МИНИСТЕРСТВО СЕЛЬСКОГО ХОЗЯЙСТВА РОССИЙСКОЙ ФЕДЕРАЦИИ
ДЕПАРТАМЕНТ НАУЧНО-ТЕХНОЛОГИЧЕСКОЙ ПОЛИТИКИ И ОБРАЗОВАНИЯ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ «ДОНСКОЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО Донской ГАУ)

АЗОВО-ЧЕРНОМОРСКИЙ ИНЖЕНЕРНЫЙ ИНСТИТУТ – ФИЛИАЛ
ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО ОБРАЗОВАТЕЛЬНОГО
УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ» В Г. ЗЕРНОГРАДЕ
(Азово-Черноморский инженерный институт ФГБОУ ВО Донской ГАУ)

Кафедра математики и биоинформатики

ИНФОРМАТИКА И ЦИФРОВЫЕ ТЕХНОЛОГИИ В АГРОИНЖЕНЕРИИ

Лабораторный практикум
для бакалавров, обучающихся по направлению подготовки
35.03.06 «Агроинженерия»

Часть 1

Зерноград – 2026

© Грачева Н.Н., Руденко Н.Б., Литвинов В.Н., Т.В. Жидченко, М.Н. Середина, Н.М. Удинцова, В.В. Серёгина, Д.В. Степовой 2026
© Азово-Черноморский инженерный институт ФГБОУ ВО Донской ГАУ, 2026

Об издании – 1, 2
Содержание

УДК 681.3

Публикуется по решению методического совета энергетического факультета Азово-Черноморского инженерного института – филиала Федерального государственного бюджетного образовательного учреждения высшего образования «Донской государственный аграрный университет» в г. Зерограде

Рецензент:
Доцент кафедры «Эксплуатация энергетического оборудования и электрических машин», кандидат технических наук, *А.Н. Токарева*

Информатика и цифровые технологии в агроинженерии. Часть 1 [Электронный ресурс]: лабораторный практикум. Часть 1 / Н.Б. Руденко, Н.Н. Грачева, В.Н. Литвинов, Т.В. Жидченко, М.Н. Середина, Н.М. Удинцова, В.В. Серёгина, Д.В. Степовой. – Зерноград: Азово-Черноморский инженерный институт ФГБОУ ВО ДГАУ, 2026. – 186 с.

Лабораторный практикум предназначен для студентов 1, 2 курсов очной и заочной форм обучения, изучающих дисциплину «Информатика и цифровые технологии» и обучающихся по направлению подготовки 35.03.06 – «Агроинженерия».

В часть 1 практикума включены следующие темы курса «Информатика и цифровые технологии»: системы счисления, логические основы работы компьютера, основы работы в операционной системе Linux, структурное программирование на языке высокого уровня Python.

Лабораторный практикум составлен таким образом, чтобы студент мог, при необходимости, самостоятельно изучить некоторые разделы дисциплины и выполнить контрольные работы.

Рассмотрено и одобрено на заседании кафедры математики и биоинформатики
Протокол № 2 от 28.09.2026 г.

Рассмотрено и одобрено методическим советом
Энергетического факультета
Протокол № 1 от 30.09.2026 г.

© Руденко Н.Б., Грачева Н.Н., Литвинов В.Н.,
Жидченко Т.В., Середина М.Н., Удинцова Н.М.,
Серёгина В.В., Степовой Д.В., 2026
© Азово-Черноморский инженерный институт
ФГБОУ ВО Донской ГАУ, 2026

СОДЕРЖАНИЕ

Введение 5

РАЗДЕЛ 1 ПРИМЕНЕНИЕ СИСТЕМ СЧИСЛЕНИЯ И ЛОГИЧЕСКИХ ОСНОВ РАБОТЫ КОМПЬЮТЕРА В ИНЖЕНЕРНЫХ ЗАДАЧАХ 7

Лабораторная работа №1. Позиционные системы счисления. Перевод чисел из одной системы счисления в другую 7

Лабораторная работа №2. Логические основы работы компьютера. Логические функции. Таблицы истинности. Упрощение логических функций 19

Лабораторная работа №3. Логические основы работы компьютера. Основные логические элементы. Логические схемы 28

Лабораторная работа №4. *Контрольная работа № 1. «Обработка и анализ информации, представление информации в различных форматах».* Примерный вариант 43

Лабораторная работа №5. ОС Linux. Основные принципы функционирования операционной системы. Работа в режиме командной строки 47

Лабораторная работа №6. *Контрольная работа № 2. «Использование операционной системы Linux для решения типовых задач в области агроинженерии».* Примерный вариант 68

РАЗДЕЛ 2 ИСПОЛЬЗОВАНИЕ ЯЗЫКА ПРОГРАММИРОВАНИЯ ВЫСОКОГО УРОВНЯ ДЛЯ РЕШЕНИЯ ТИПОВЫХ ЗАДАЧ В ОБЛАСТИ АГРОИНЖЕНЕРИИ 72

Лабораторные работы №7.

Основные языковые конструкции языка программирования высокого уровня. Линейный вычислительный процесс 72

Лабораторная работа №8. Управляющие операторы разветвляющегося вычислительного процесса. Операторы if и if-else 98

Лабораторная работа №9. Управляющие операторы разветвляющегося вычислительного процесса. Оператор if-elif-else 105

Лабораторная работа №10-11. Управляющие операторы циклического вычислительного процесса 113

Лабораторная работа №12. *Контрольная работа № 3. «Использование языка программирования высокого уровня для решения типовых задач в области агроинженерии».* Примерный вариант 126

Лабораторная работа №13. Комбинированные алгоритмы 130

Лабораторная работа №14. Алгоритмы накопления суммы и произведения 139

Лабораторная работа №15. Обработка одномерных массивов. Одномерные массивы: накопление суммы и произведения, поиск максимального (минимального) элемента массива 147

Лабораторная работа №16. *Контрольная работа № 4. «Типовые алгоритмы поиска информации для решения типовых задач в области агроинженерии».* Примерный вариант 159

Приложения 164

Глоссарий 178

Литература 182

Введение

Применение цифровых технологий в современном мире становится всё более значимым и широко используется в различных отраслях деятельности. Сегодня агропромышленный комплекс переходит на качественно новый уровень. Активная цифровизация сельского хозяйства, роботизация производства и искусственный интеллект делают отрасль всё более высокотехнологичной.

Такое техническое и технологическое перевооружение сельскохозяйственного производства требует от образовательных учреждений подготовки высококвалифицированных кадров в области цифровых технологий, независимо от специальности или направления подготовки.

Учебное пособие «Информатика и цифровые технологии в агроинженерии. Часть 1» соответствует Федеральному государственному образовательному стандарту высшего образования (ФГОС ВО) по направлению подготовки 35.03.06 «Агроинженерия».

В учебное пособие включены следующие разделы курса «Информатика и цифровые технологии»: применение систем счисления и логических основ работы компьютера в инженерных задачах, использование языка программирования высокого уровня для решения типовых задач в области агроинженерии.

Учебное пособие состоит из лабораторных работ, заданий для самостоятельной работы и вариантов контрольных работ. Пособие предназначено для студентов очной и заочной форм обучения и составлено таким образом, чтобы студенты заочной формы обучения могли самостоятельно изучить необходимые разделы дисциплины и выполнить контрольные работы.

В результате изучения материала, представленного в учебном пособии, студент должен овладеть следующими компетенциями:

– способен решать типовые задачи профессиональной деятельности на основе знаний основных законов математических и естественных наук с применением информационно-коммуникационных технологий (ОПК-1);

– применяет информационно-коммуникационные технологии в решении типовых задач в области агроинженерии (ОПК-1.3);

– пользуется специальными программами и базами данных при разработке и расчете энергетического оборудования, средств автоматизации и электрификации сельского хозяйства (ОПК-1.4);

– способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности (ОПК-7);

– понимает принципы работы современных информационных технологий (ОПК-7.1);

– реализует принципы работы современных информационных технологий для решения задач профессиональной деятельности (ОПК-7.2).

В результате изучения дисциплины «Информатика и цифровые технологии» студент должен:

– знать информационно-коммуникационные технологии в решении типовых задач в области агроинженерии; специальные программы и базы данных при разработке и расчете энергетического оборудования, средств автоматизации и электрификации сельского хозяйства; принципы работы современных информационных технологий для решения задач профессиональной деятельности;

– уметь применять информационно-коммуникационные технологии в решении типовых задач в области агроинженерии; применять специальные программы и базы данных при разработке и расчете энергетического оборудования, средств автоматизации и электрификации сельского хозяйства; применять современные информационные технологии для решения задач профессиональной деятельности;

– владеть навыками решения типовых задач профессиональной деятельности с применением информационно-коммуникационных технологий и специальных программ и баз данных при разработке и расчёте энергетического оборудования, средств автоматизации и электрификации сельского хозяйства; навыками применения современных информационных технологий для решения задач профессиональной деятельности.

РАЗДЕЛ 1
ПРИМЕНЕНИЕ СИСТЕМ СЧИСЛЕНИЯ И ЛОГИЧЕСКИХ ОСНОВ РАБОТЫ КОМПЬЮТЕРА В ИНЖЕНЕРНЫХ ЗАДАЧАХ

ЛАБОРАТОРНАЯ РАБОТА № 1

Тема: **ПОЗИЦИОННЫЕ СИСТЕМЫ СЧИСЛЕНИЯ. ПЕРЕВОД ЧИСЕЛ ИЗ ОДНОЙ СИСТЕМЫ СЧИСЛЕНИЯ В ДРУГУЮ**

1 ЦЕЛЬ РАБОТЫ

Познакомиться:
- с позиционными системами счисления, применяемыми для вычислений в компьютере.

Научиться:
- переводить числа из одной системы счисления в другую;
- выполнять основные арифметические операции в позиционных системах счисления.

2 ОБЪЁМ РАБОТЫ И ПОРЯДОК ЕЁ ВЫПОЛНЕНИЯ

Данная работа выполняется в течение двух часов. Выполнение работы идет одновременно с изучением теоретической части. Для закрепления материала необходимо выполнить упражнения для самостоятельной работы.

3 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ. ТЕХНОЛОГИЯ ВЫПОЛНЕНИЯ РАБОТЫ

3.1 Основные понятия

Система счисления – это совокупность правил для обозначения и наименования чисел.

Системы счисления делят на две большие группы: непозиционные и позиционные.

Непозиционными системами счисления называются системы счисления, в которых каждой цифре соответствует величина, не зависящая от её места в записи числа. Примером непозиционной системы счисления может служить римская система счисления, символы алфавита которой и обозначаемое ими количество представлено в таблице 1.1.

Таблица 1.1 – Римская система счисления

Римские цифры	I	V	X	L	C	D	M
Значение (обозначаемое количество)	1	5	10	50	100	500	1000

Позиционными системами счисления называются системы счисления, в которых вклад каждой цифры в величину числа зависит от её положения (позиции) в последовательности цифр, изображающей число. Примером может служить десятичная система счисления, считать в которой учат в школе. Значение каждой цифры в числе зависит от ее местоположения. Сравните два числа: 50 и 500. В первом числе пять стоит в разряде десятков, во втором числе – в разряде сотен.

Основными достоинствами любой позиционной системы счисления являются:

– во-первых, ограниченное количество символов для записи чисел;

– во-вторых, простота выполнения арифметических операций.

Основанием позиционной системы счисления (q) называют количество символов, используемое для записи числа.

Основание системы счисления показывает, во сколько раз изменится количественное значение цифры при перемещении ее на соседнюю позицию.

При записи числа основание системы счисления указывают в виде нижнего индекса: 54_{10}, 1011_2, $A1E_{16}$.

В компьютере используют двоичную, десятичную, восьмеричную и шестнадцатеричную системы счисления.

От системы счисления будет зависеть:

– скорость вычислений;

– емкость памяти;

– сложность алгоритма выполнения арифметических и логических операций.

Двоичная система счисления используется для организации машинных операций по преобразованию информации:

q = 2, алфавит: 0, 1.

Десятичная система счисления – для ввода и вывода информации:

q = 10, алфавит: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Восьмеричная и шестнадцатеричная системы счисления – для составления программ.

В восьмеричной системе счисления:

q = 8, алфавит: 0, 1, 2, 3, 4, 5, 6, 7.

В шестнадцатеричной системе счисления:

q = 16, алфавит: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

3.2 Правила перевода чисел из одной системы счисления в другую

3.2.1 Перевод чисел из десятичной системы счисления в двоичную, восьмеричную и шестнадцатеричную системы счисления

Для перевода числа из десятичной системы счисления в двоичную (восьмеричную, шестнадцатеричную) системы счисления необходимо:

– целую часть числа разделить на основание системы счисления, в которую производится перевод (в двоичную – на 2, в восьмеричную – на 8, в шестнадцатеричную – на 16). Деление целочисленное, с остатком. Деление

продолжать до тех пор, пока делимое не станет меньше делителя (2, 8, 16). Затем выписать остатки, начиная с последнего. Таким образом, получим целую часть числа в новой системе счисления;

– дробную часть числа умножаем на основание системы счисления, в которую производится перевод (в двоичную – на 2, в восьмеричную – на 8, в шестнадцатеричную – на 16). Если при умножении получается целое число, то в следующем действии оно не учитывается, а умножается дробная часть и т.д. Умножение заканчивается тогда, когда дробная часть равна 0. Если этого не происходит, то умножение производить до третьего знака после запятой. В дробную часть переведенного числа выписываем полученные целые числа.

В таблице 1.2 (см. ниже) приведены правила перевода целой и дробной частей чисел десятичной системы счисления в двоичную, восьмеричную и шестнадцатеричную системы счисления.

Таблица 1.2 – Перевод чисел десятичной системы счисления в двоичную, восьмеричную и шестнадцатеричную системы счисления

Направление перевода	Правила перевода
Целая часть числа N_{10} → N_2	деление на 2
N_{10} → N_8	деление на 8
N_{10} → N_{16}	деление на 16
Дробная часть числа N_{10} → N_2	умножение на 2
N_{10} → N_8	умножение на 8
N_{10} → N_{16}	умножение на 16

Поясним правила на примерах.

Пример 1

Переведите число $67{,}25_{10}$ из десятичной системы счисления в двоичную.

Перевод целой части

```
_ 7 | 2
  6 | _3 | 2
  1 |  2 | _1 | 2
    |  1 |  0 | _0 | 2
    |  0 |    |  0 | 1
    |    |    |    | 0
```

Перевод дробной части

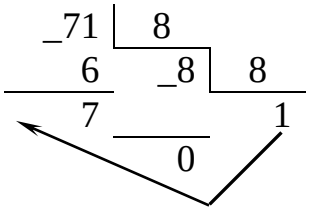
```
0, | 25
   |  2
0  | 50
   |  2
1  | 00
```

Ответ: $67{,}25_{10} = 1000011{,}01_2$

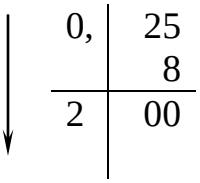
Пример 2

Переведите число $71{,}25_{10}$ из десятичной системы счисления в восьмеричную.

Перевод целой части



Перевод дробной части

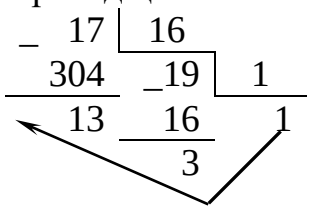


Ответ: $71{,}25_{10} = 107{,}2_8$

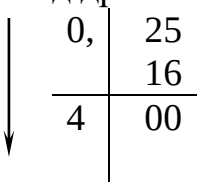
Пример 3

Переведите число $317{,}25_{10}$ из десятичной системы счисления в шестнадцатеричную.

Перевод целой части



Перевод дробной части



Ответ: $317{,}25_{10} = 13D{,}4_{16}$

3.2.2 Перевод чисел из двоичной, восьмеричной и шестнадцатеричной систем счисления в десятичную систему счисления

Для перевода числа из двоичной (восьмеричной, шестнадцатеричной) системы счисления в десятичную необходимо:

– над каждой цифрой числа, которое необходимо перевести, поставить номер разряда, на котором стоит цифра. Разряды выставляются в целой части числа справа налево (начиная с 0), в дробной части – слева направо (начиная с –1);

– умножаем каждую цифру переводимого числа на основание системы счисления (из которой происходит перевод), возведенное в степень, равную номеру разряда, на котором стоит соответствующая цифра. Затем складываем эти произведения;

– если число переводится из шестнадцатеричной системы счисления, то буква, при умножении, заменяется соответствующим этой букве значением в десятичной системе счисления (см. таблицу 1.3).

Таблица 1.3 – Таблица соответствия между системами счисления

Десятичная	**Двоичная**	**Восьмеричная**	**Шестнадцатеричная**
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14
21	10101	25	15
22	10110	26	16
23	10111	27	17
24	11000	30	18
25	11001	31	19

В таблице 1.4 приведены правила перевода чисел из двоичной, восьмеричной и шестнадцатеричной систем счисления в десятичную систему счисления.

Таблица 1.4 – Перевод чисел из двоичной, восьмеричной и шестнадцатеричной систем счисления в десятичную систему счисления

Направление перевода	Правила перевода
N_2 $N_8 \rightarrow N_{10}$ N_{16}	представить число в развернутой форме, например: $\overset{3\,2\,1\,0}{1011_2} = 1\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 = 8+2+1 = 11_{10}$ $\overset{1\,0}{75_8} = 7\cdot 8^1 + 5\cdot 8^0 = 56+5 = 61_{10}$ $\overset{1\,0}{A3_{16}} = A\cdot 16^1 + 3\cdot 16^0 = 10\cdot 16+3 = 163_{10}$

Поясним правила на примерах.

Пример 4

Переведите число $57{,}4_8$ из восьмеричной системы счисления в десятичную.

$$\overset{1\ 0\ -1}{5\,7,\,4}_8 = 5\cdot 8^1 + 7\cdot 8^0 + 4\cdot 8^{-1} = 40 + 7 + \frac{4}{8} = 47\,\frac{1}{2} = 47{,}5_{10}.$$

Ответ: $57{,}4_8 = 47{,}5_{10}$

Пример 5

Переведите число $A7{,}48_{16}$ из шестнадцатеричной системы счисления в десятичную.

$$\overset{1\ 0\ -1\ -2}{\mathrm{A}\,7,\,4\ 8}_{16} = 10\cdot 16^1 + 7\cdot 16^0 + 4\cdot 16^{-1} + 8\cdot 16^{-2} = 160 + 7 + \frac{4}{16} + \frac{8}{256} = 167\,\frac{9}{32}_{10}.$$

Ответ: $A7{,}48_{16} = 167\ \frac{9}{32}_{10}$

3.2.3 Перевод чисел из двоичной системы счисления в восьмеричную и шестнадцатеричную системы счисления

Для перевода числа из двоичной системы счисления в восьмеричную (шестнадцатеричную) системы счисления необходимо:

– двоичное число разбить на тройки цифр – триады (для восьмеричной системы счисления) или на четверки цифр – тетрады (для шестнадцатеричной системы счисления). Разбиение начинать от запятой, разделяющей целую и дробную части, влево и вправо соответственно;

– если в отделенной последовательности количество цифр меньше трёх (четырех), то недостающие цифры заменить нулями. Для целой части нули приписать перед числом, для дробной части – после числа;

– выписать соответственно каждой отделенной группе (триаде или тетраде) по таблице соответствия между системами счисления восьмеричную или шестнадцатеричную цифру (или букву) (см. таблицу 1.3).

В таблице 1.5 приведены правила перевода чисел из двоичной системы счисления в восьмеричную и шестнадцатеричную системы счисления.

Таблица 1.5 – Перевод чисел из двоичной системы счисления в восьмеричную и шестнадцатеричную системы счисления

Направление перевода	Правила перевода
$N_2 \rightarrow N_8$	замена триад цифрами восьмеричной системы счисления
$N_2 \rightarrow N_{16}$	замена тетрад цифрами шестнадцатеричной системы счисления

Поясним правила на примерах.

Пример 6

Переведите число $1111101{,}01011_2$ из двоичной в восьмеричную и шестнадцатеричную системы счисления.

$$1111101{,}01011_2 = \underset{1}{001}\ \underset{7}{111}\ \underset{5}{101},\ \underset{2}{010}\ \underset{6}{110_2} = 175{,}26_8$$

$$1111101{,}01011_2 = \underset{7}{0111}\ \underset{D}{1101},\ \underset{5}{0101}\ \underset{8}{1000_2} = 7D{,}58_{16}$$

Ответ: $1111101{,}01011_2 = 175{,}26_8$; $1111101{,}01011_2 = 7D{,}58_{16}$

3.2.4 Перевод чисел из восьмеричной и шестнадцатеричной систем счисления в двоичную систему счисления

Для перевода числа из восьмеричной (шестнадцатеричной) системы счисления в двоичную систему счисления необходимо:

– выписать соответственно каждой цифре восьмеричного числа тройку цифр – триаду по таблице соответствия между системами счисления триаду (см. таблицу 1.3). Аналогично поступаем и с шестнадцатеричным числом, но выписываем четверку цифр – тетраду;

– в записи полученного двоичного числа нули в начале целой части и в конце дробной части необходимо опустить.

В таблице 1.6 приведены правила перевода чисел из двоичной системы счисления в восьмеричную и шестнадцатеричную системы счисления.

Таблица 1.6 – Перевод чисел из двоичной системы счисления в восьмеричную и шестнадцатеричную системы счисления

Направление перевода	Правила перевода
N_8 → N_2	замена восьмеричных цифр триадами
N_{16} → N_2	замена шестнадцатеричных цифр тетрадами

Поясним правила на примерах.

Пример 7

Переведите числа $567{,}561_8$ и $4FE, AD_{16}$ в двоичную систему счисления.

$$567{,}561_8 = \underset{101}{5}\ \underset{110}{6}\ \underset{111}{7},\ \underset{101}{5}\ \underset{110}{6}\ \underset{001}{1_8} = 101110111{,}101110001_2$$

$$4FE, AD_{16} = \underset{0100}{4}\ \underset{1111}{F}\ \underset{1110}{E},\ \underset{1010}{A}\ \underset{1101}{D_{16}} = 10011111110{,}10101101_2$$

Ответ: $567{,}561_8 = 101110111{,}101110001_2$;
$4FE, AD_{16} = 10011111110{,}10101101_2$.

3.2.5 Перевод чисел из восьмеричной системы счисления в шестнадцатеричную систему счисления (или из шестнадцатеричной системы счисления в восьмеричную систему счисления)

Для перевода числа из восьмеричной системы счисления в шестнадцатеричную (из шестнадцатеричной в восьмеричную) необходимо:

– осуществить перевод восьмеричного (или шестнадцатеричного) числа в двоичную систему счисления (см. п.3.2.4);

– осуществить перевод числа из двоичной системы счисления в шестнадцатеричную (или восьмеричную) систему счисления (см. п.3.2.3).

Поясним правила на примерах.

Пример 8

Переведите число $4FE, AD_{16}$ в восьмеричную систему счисления, а число $567{,}561_8$ в шестнадцатеричную систему счисления.

$$4FE, AD_{16} = 0100\ 1111\ 1110, 1010\ 1101_2 = 10011111110{,}10101101_2$$
$$= 010\ 011\ 111\ 110, 101\ 011\ 010_2 = 2376{,}532_8$$

$$567{,}561_8 = 101\ 110\ 111, 101\ 110\ 001_2 = 101110111{,}101110001_2$$
$$= 0001\ 0111\ 0111, 1001\ 1001_2 = 177, B9_{16}$$

3.3 Арифметические операции в позиционных системах счисления

Арифметические операции в двоичной, восьмеричной и шестнадцатеричной системе счисления сходны с аналогичными операциями в десятичной системе счисления.

Все операции производятся столбиком. Рассмотрим несколько примеров.

Пример 9

Выполните сложение, вычитание и умножение двоичных чисел $1100{,}11_2$ и $101{,}10_2$.

сложение	вычитание	умножение
1100,11 + 101,10 10010,01	1100,11 – 101,10 111,01	1100,11 × 101,1 110011 + 110011 110011 10000110,001

Пример 10

Выполните сложение, вычитание и умножение чисел в восьмеричной системе счисления: $215{,}4_8$ и $73{,}6_8$.

Необходимо помнить:

– если при сложении или умножении двух цифр в разряде получилось число больше восьми, то полученное число необходимо разделить на 8 (целочисленное деление). Остаток от деления оставить в текущем разряде, а результат деления прибавить к старшему разряду;

– если при вычитании двух цифр уменьшаемая цифра меньше вычитаемой, то занимаем в старшем разряде 1 (единицу, для текущего разряда это будет 8). 8 прибавляем к уменьшаемому и из полученной суммы отнимаем вычитаемое.

сложение	вычитание	умножение
1 1 1 215,4 + 73,6 311,2 4+6=10=8+2 5+3+1=9=8+1 1+7+1=9=8+1 2+1=3	1 1 1 215,4 – 73,6 121,6 8+4-6=6 4-3=1 1+8-7=2	215,4 × 73,6 15210 + 6504 17364 20406,50

Пример 11

Выполните сложение и вычитание чисел в шестнадцатеричной системе счисления: $8D,8_{16}$ и $3B,C_{16}$.

Необходимо помнить:

– если при сложении или умножении двух цифр в разряде получилось число больше шестнадцати, то полученное число необходимо разделить на 16 (целочисленное деление). Остаток от деления оставить в текущем разряде, а результат деления прибавить к старшему разряду;

– если при вычитании двух цифр уменьшаемая цифра меньше вычитаемой, то занимаем в старшем разряде 1 (для текущего разряда это будет 16). 16 прибавляем к уменьшаемому и из полученной суммы отнимаем вычитаемое.

сложение	вычитание	умножение
1 1 8D,8 + 3B,C C9,4 8+2=20=16+4 13+11+1=25=16+9 8+3+1=12=C	1 1 8D,8 − 3B,C 51,C 8+16=24-12=12=C 12-11=1 8-3=5	8D,8 × 3B,C 6A20 + 6148 1A88 2106,A0

4 СОДЕРЖАНИЕ ОТЧЕТА

Отчет должен содержать тему и цель работы, выполненные задания для самостоятельной работы по указанному преподавателем варианту и ответы на контрольные вопросы.

5 УПРАЖНЕНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

1. Переведите числа из десятичной системы счисления в двоичную, восьмеричную, шестнадцатеричную системы счисления.

1) $57{,}9_{10}$	9) $97{,}5_{10}$	17) $59{,}5_{10}$	25) $110{,}1_{10}$
2) $123{,}01_{10}$	10) $55{,}55_{10}$	18) $29{,}3_{10}$	26) $25{,}45_{10}$
3) $113{,}1_{10}$	11) $80{,}25_{10}$	19) $67{,}5_{10}$	27) $38{,}35_{10}$
4) $39{,}2_{10}$	12) $17{,}17_{10}$	20) $91{,}25_{10}$	28) $71{,}1_{10}$
5) $91{,}95_{10}$	13) $11{,}95_{10}$	21) $77{,}7_{10}$	29) $44{,}9_{10}$
6) $89{,}5_{10}$	14) $83{,}65_{10}$	22) $45{,}2_{10}$	30) $31{,}23_{10}$
7) $76{,}45_{10}$	15) $22{,}22_{10}$	23) $34{,}25_{10}$	31) $86{,}2_{10}$
8) $190{,}2_{10}$	16) $78{,}05_{10}$	24) $66{,}5_{10}$	32) $29{,}35_{10}$

2. Переведите числа в десятичную систему счисления.

1) $101{,}10_2$	9) $2{,}3_8$	17) $17{,}17_8$	25) 100001_2
2) 555_{16}	10) 11001_2	18) $77{,}6_8$	26) 190_{16}
3) 113_8	11) $BA2{,}A_{16}$	19) 1010101_2	27) $11{,}75_8$
4) 1010_2	12) 22_8	20) $A38{,}1_{16}$	28) $1101{,}1_2$
5) 788_{16}	13) 111_2	21) $17{,}5_8$	29) $F6{,}B_{16}$
6) 35_8	14) $AB6_{16}$	22) 10001_2	30) $71{,}3_8$
7) 11011_2	15) 52_8	23) $F80_{16}$	31) $11{,}101_2$
8) $CC3{,}5_{16}$	16) 111101_2	24) $11{,}2_8$	32) $BB3{,}3_{16}$

3. Переведите числа в восьмеричную и шестнадцатеричную системы счисления.

1) 1000000_2	9) $1010{,}11_2$	17) 1001100_2	25) 101000_2
2) $100{,}001_2$	10) 1110_2	18) 1000001_2	26) $1001{,}01_2$
3) $111{,}001_2$	11) $101{,}10_2$	19) $101{,}10_2$	27) $101{,}10_2$
4) $1{,}0001_2$	12) $111{,}0011_2$	20) $11{,}1011_2$	28) $10{,}0101_2$
5) 10101_2	13) 11000_2	21) 10100101_2	29) $1110{,}1_2$
6) $110{,}11_2$	14) 1000101_2	22) $10{,}001001_2$	30) $100{,}0001_2$
7) $101{,}1_2$	15) 110011_2	23) $110{,}001_2$	31) $1100{,}11_2$
8) $100{,}100_2$	16) $11{,}0101_2$	24) $1000{,}01_2$	32) $11001{,}1_2$

4. Переведите числа в двоичную систему счисления.

1) $65{,}8_{16}$	9) $67{,}5_8$	17) BA_{16}	25) $51, A_{16}$
2) $123{,}8_{16}$	10) $55{,}55_8$	18) 29_{16}	26) $B7_{16}$
3) 143_{16}	11) 35_8	19) 62_{16}	27) 47_{16}
4) 39_{16}	12) $17{,}17_8$	20) 130_8	28) $8C_{16}$
5) $F16_{16}$	13) $37, A_{16}$	21) $74{,}6_8$	29) $13{,}6_{16}$
6) $89{,}4_{16}$	14) $86{,}5_{16}$	22) $43{,}2_8$	30) $41{,}7_{16}$
7) $76{,}4_8$	15) $22{,}67_{16}$	23) $41{,}1_8$	31) 36_8
8) 50_8	16) $78{,}32_{16}$	24) 500_8	32) $15{,}5_8$

5. Переведите числа из шестнадцатеричной системы счисления в восьмеричную, из восьмеричной системы счисления в шестнадцатеричную.

1) $65{,}8_{16}$	9) $67{,}5_8$	17) BA_{16}	25) $51, A_{16}$
2) $123{,}8_{16}$	10) $55{,}55_8$	18) 29_{16}	26) $B7_{16}$
3) 143_{16}	11) 35_8	19) 62_{16}	27) 47_{16}
4) 39_{16}	12) $17{,}17_8$	20) 130_8	28) $8C_{16}$
5) $F16_{16}$	13) $37, A_{16}$	21) $74{,}6_8$	29) $13{,}6_{16}$
6) $89{,}4_{16}$	14) $86{,}5_{16}$	22) $43{,}2_8$	30) $41{,}7_{16}$
7) $76{,}4_8$	15) $22{,}67_{16}$	23) $41{,}1_8$	31) 36_8
8) 50_8	16) $78{,}32_{16}$	24) 500_8	32) $15{,}5_8$

6. Выполните сложение чисел, а затем проверьте результаты, выполнив соответствующие действия в десятичной системе счисления.

1) $101111_2 + 11101_2$	11) $35_8 + 17_8$	21) $57{,}25_8 + 71{,}6_8$
2) $1011{,}101_2 + 10{,}11_2$	12) $13{,}17_8 + 16_8$	22) $75_8 + 17_8$
3) $1011_2 + 110_2$	13) $F1_{16} + A_{16}$	23) $E, 5_{16} + A, B_{16}$
4) $1011_2 + 111_2$	14) $19_{16} + C_{16}$	24) $3D_{16} + E_{16}$
5) $110{,}1_2 + 11{,}1_2$	15) $E, 2_{16} + A, B_{16}$	25) $10111_2 + 111_2$
6) $1010_2 + 1111_2$	16) $F8_{16} + 9_{16}$	26) $19_{16} + F_{16}$
7) $35_8 + 75_8$	17) $BA_{16} + C_{16}$	27) $B9_{16} + 12_{16}$
8) $165_8 + 37_8$	18) $209_{16} + 77_{16}$	28) $100111_2 + 111_2$
9) $7{,}5_8 + 14{,}6_8$	19) $101{,}1_2 + 11{,}1_2$	29) $79_{16} + D_{16}$
10) $55{,}55_8 + 36_8$	20) $1010_2 + 111_2$	30) $8A_{16} + 17_{16}$

7. Выполните вычитание чисел, а затем проверьте результаты, выполнив соответствующие действия в десятичной системе счисления.

1) $101111_2 - 11101_2$
2) $1011{,}101_2 - 10{,}11_2$
3) $1011_2 - 110_2$
4) $1011_2 - 111_2$
5) $110{,}1_2 - 11{,}1_2$
6) $1010_2 + 11_2$
7) $35_8 - 7_8$
8) $165_8 - 37_8$
9) $7{,}5_8 - 1{,}6_8$
10) $55{,}55_8 - 36_8$
11) $35_8 - 17_8$
12) $13{,}17_8 - 6_8$
13) $F1_{16} - A_{16}$
14) $19_{16} - C_{16}$
15) $E5{,}2_{16} - A, B_{16}$
16) $F8_{16} - 9_{16}$
17) $BA_{16} - C_{16}$
18) $209_{16} - 77_{16}$
19) $101{,}1_2 - 11{,}1_2$
20) $1010_2 - 111_2$
21) $57{,}25_8 - 7{,}6_8$
22) $75_8 - 17_8$
23) $E2{,}5_{16} - A, B_{16}$
24) $3D_{16} - E_{16}$
25) $10111_2 - 111_2$
26) $C9_{16} - F_{16}$
27) $B9_{16} - 12_{16}$
28) $100111_2 - 111_2$
29) $D9_{16} - D_{16}$
30) $8A_{16} - 17_{16}$

8. Выполните умножение чисел, а затем проверьте результаты, выполнив соответствующие действия в десятичной системе счисления.

1) $1011101_2 \cdot 11101_2$
2) $1011{,}101_2 \cdot 10{,}11_2$
3) $1011_2 \cdot 110_2$
4) $1011_2 \cdot 101_2$
5) $110{,}1_2 \cdot 11{,}1_2$
6) $1010_2 \cdot 11_2$
7) $35_8 \cdot 71_8$
8) $165_8 \cdot 37_8$
9) $7{,}5_8 \cdot 1{,}6_8$
10) $55_8 \cdot 36_8$
11) $35_8 \cdot 17_8$
12) $13{,}17_8 \cdot 6_8$
13) $F1_{16} \cdot A_{16}$
14) $19_{16} \cdot A_{16}$
15) $E, 2_{16} \cdot A, B_{16}$
16) $F8_{16} \cdot 9_{16}$
17) $BA_{16} \cdot C_{16}$
18) $209_{16} \cdot 77_{16}$
19) $101{,}1_2 \cdot 11{,}1_2$
20) $1010_2 \cdot 111_2$
21) $75_8 \cdot 16_8$
22) $17{,}17_8 \cdot 6{,}6_8$
23) $E, 5_{16} \cdot A, B_{16}$
24) $3D_{16} \cdot E_{16}$
25) $10111_2 \cdot 111_2$
26) $19_{16} \cdot F_{16}$
27) $B1_{16} \cdot 12_{16}$
28) $100111_2 \cdot 111_2$
29) $79_{16} \cdot D_{16}$
30) $8A_{16} \cdot 17_{16}$

6 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что называется системой счисления? Перечислите виды систем счисления.
2. Какая система счисления называется позиционной, непозиционной?
3. Что называется основанием позиционной системы счисления?
4. Что называется алфавитом системы счисления? Перечислите алфавиты для двоичной, восьмеричной, десятичной и шестнадцатеричной систем счисления.
5. Сформулируйте правило перевода из десятичной системы счисления в двоичную, восьмеричную, шестнадцатеричную системы счисления.
6. Сформулируйте правило перевода из двоичной, восьмеричной, шестнадцатеричной системы счисления в десятичную систему счисления.
7. Сформулируйте правила перевода из двоичной системы счисления в восьмеричную и шестнадцатеричную систему счисления и обратно.

ЛАБОРАТОРНАЯ РАБОТА № 2

Тема: **ЛОГИЧЕСКИЕ ОСНОВЫ РАБОТЫ КОМПЬЮТЕРА. ЛОГИЧЕСКИЕ ФУНКЦИИ. ТАБЛИЦЫ ИСТИННОСТИ. УПРОЩЕНИЕ ЛОГИЧЕСКИХ ФУНКЦИЙ**

1 ЦЕЛЬ РАБОТЫ

Познакомиться:
- с логическими основами работы компьютера;
- с логическими функциями;
- с основными законами и тождествами алгебры логики.

Научиться:
- составлять таблицы истинности для логических функций;
- упрощать логические функции.

2 ОБЪЁМ ЛАБОРАТОРНОЙ РАБОТЫ И ПОРЯДОК ЕЁ ВЫПОЛНЕНИЯ

Данная работа выполняется в течение двух часов. Выполнение работы идет одновременно с изучением теоретической части. Для закрепления материала необходимо выполнить упражнения для самостоятельной работы.

3 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ. ТЕХНОЛОГИЯ ВЫПОЛНЕНИЯ РАБОТЫ

3.1 Логические основы работы компьютера

Основу компьютера составляют элементарные логические схемы. Работа этих схем основана на законах и правилах алгебры логики, которая оперирует двумя понятиями: истинности и ложности высказывания.

3.1.1 Основные понятия алгебры логики

Алгебра логики (Булева алгебра) – это раздел математики, изучающий высказывания, рассматриваемые со стороны их логических значений (истинности или ложности) и логических операций над ними.

Логическое высказывание – это любое повествовательное предложение, в отношении которого можно однозначно сказать, истинно оно или ложно.

Высказывания бывают простыми и сложными.

Простое высказывание (логическая переменная) содержит только одну простую мысль.

Сложное высказывание (логическая функция) содержит несколько простых мыслей, соединенных между собой с помощью логических операций.

Логические переменные обычно обозначаются буквами латинского алфавита: ***A, B, C …***

Основными логическими операциями являются:

– отрицание (инверсия);

– умножение (конъюнкция);

– сложение (дизъюнкция);

Дополнительными логическими операциями являются:

– следование (импликация);

– равнозначность (эквиваленция).

Значение сложного высказывания (**логической функции**) можно определить с помощью таблицы истинности.

Таблица истинности – это таблица, в которой перечислены все возможные значения входящих логических переменных и соответствующие им значения функции.

3.1.2 Логические операции

Для реализации любых логических операций над двоичными сигналами достаточно элементов трех типов – элементов, реализующих три основные логические операции: **отрицание (НЕ)**, **логическое умножение (И)**, **логическое сложение (ИЛИ)**.

Отрицание (инверсия)

Операция, выражаемая словом **НЕ**, называется **инверсией** или **отрицанием**. Применяется к одной переменной или логическому выражению.

Обозначается: не $A, \bar{A}, \neg A, notA$.

Таблица истинности:

A	$\bar{A}$
0	1
1	0

Высказывание $\overline{A}$ истинно, когда A ложно, и ложно, когда A истинно.

Пример:

«Луна — спутник Земли» (A)

«Луна — не спутник Земли» ($\bar{A}$)

Логическое умножение (конъюнкция)

Соответствует союзу **И**. Применяется к двум переменным или логическим выражениям.

Обозначается: $*$, и,$\wedge, \&, and$.

Таблица истинности:

A	B	$A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

Конъюнкция истинна, тогда и только тогда, когда оба высказывания истинны.

Любое сложное высказывание можно записать с помощью основных логических операций.

Логическое сложение (дизъюнкция)

Соответствует союзу **ИЛИ**. Применяется к двум переменным или логическим выражениям.

Обозначается: $+$, или,$\vee$, or.

Таблица истинности:

A	B	$A \vee B$
0	0	0
0	1	1
1	0	1
1	1	1

Дизъюнкция истинна, если хотя бы одно из высказываний истинно.

Кроме основных логических операций могут быть использованы и дополнительные логические операции. Ниже представлены две из них.

Логическое следование (импликация)

Соответствует речевому обороту **ЕСЛИ … ТО**. Применяется к двум переменным или логическим выражениям.

Обозначается: $\rightarrow$, $\Rightarrow$.

Таблица истинности:

A	B	$A \rightarrow B$
0	0	1
0	1	1
1	0	0
1	1	1

Импликация истинна всегда, за исключением случая, когда А истинно, а В ложно.

Равнозначность (эквиваленция)

Соответствует речевым оборотам: **ЭКВИВАЛЕНТНО**, **НЕОБХОДИМО И ДОСТАТОЧНО**, **ТОГДА И ТОЛЬКО ТОГДА**. Применяется к двум переменным или логическим выражениям.

Обозначается: **=, ↔, ⇔.**

Таблица истинности:

A	B	$A \Leftrightarrow B$
0	0	1
0	1	0
1	0	0
1	1	1

Эквиваленция истинна тогда и только тогда, когда оба высказывания одновременно либо истинны, либо ложны.

Порядок выполнения логических операций:

1. операция в скобках;
2. $\overline{\text{A}}$, ¬A – логическое отрицание;
3. $\times$, &, $\wedge$ – логическое умножение;
4. $+$, $\vee$ – логическое сложение;
5. $\longrightarrow$, $\Rightarrow$ – импликация;
6. $=$, $\Leftrightarrow$, $\leftrightarrow$ – эквиваленция.

Рассмотрим на примере построение таблицы истинности для логической функции.

Пример 1

Постройте таблицу истинности для логической функции $F = \bar{X} \vee X \vee Y$.

Решение:

1) Подсчитаем количество переменных в функции. В нашей функции две переменные – X и Y.

2) Построим таблицу для двух переменных и впишем все возможные сочетания значений двух переменных:

X	Y
0	0
0	1
1	0
1	1

3) Расставим в логической функции порядок выполнения операций.

$$\overset{1}{\bar{X}} \quad \overset{2}{\vee} \; X \quad \overset{3}{\vee} \; Y$$

4) Согласно полученному порядку впишем операции в таблицу истинности и подсчитаем значения. Значения логических операций выписываем из таблиц истинности для каждой конкретной логической операции.

X	Y	$\bar{X}$	$\bar{X} \vee X$	$\bar{X} \vee X \vee Y$
0	0	1	1	1
0	1	1	1	1
1	0	0	1	1
1	1	0	1	1

5) По последней таблице делаем **вывод**: функция тождественно истинна.

Так как в последнем столбце значения функции – все единицы, то функция тождественно истинна. Таким образом, функция принимает значение истина (1) при любых значениях входящих в нее переменных X и Y.

Если в последнем столбце будут все нули, то функция тождественно ложна. Таким образом, функция принимает значение ложь (0) при любых значениях входящих в нее переменных X и Y.

Если в последнем столбце значения функции – набор нулей и единиц, то функция не является ни тождественно ложной, ни тождественно истинной. В таком случае выписываются значения входящих в функцию переменных, при которых функция принимает значения истина (1).

3.1.3 Тождества и законы алгебры логики, формулы поглощения и склеивания

Для упрощения сложных логических выражений применяются тождества, законы алгебры логики, формулы поглощения и склеивания.

Тождества алгебры логики

Тождества логического сложения	Тождества логического умножения
$A \vee 0 = A$	$A \wedge 0 = 0$
$A \vee 1 = 1$	$A \wedge 1 = A$
$A \vee A = A$	$A \wedge A = A$
$A \vee \overline{A} = 1$	$A \wedge \overline{A} = 0$
$\overline{\overline{A}} = A$ – (двойное отрицание)	

Законы алгебры логики

Переместительный закон	$A \vee B = B \vee A$ $A \wedge B = B \wedge A$
Сочетательный закон	$(A \vee B) \vee C = A \vee (B \vee C)$ $(A \wedge B) \wedge C = A \wedge (B \wedge C)$
Распределительный закон	$(A \vee B) \wedge C = A \wedge C \vee B \wedge C$ $A \wedge B \vee C = (A \vee C) \wedge (B \vee C)$
Законы де Моргана	$\overline{A \vee B} = \overline{A} \wedge \overline{B}$ $\overline{A \wedge B} = \overline{A} \vee \overline{B}$ $A \longrightarrow B = \overline{A} \vee B$ $A \Leftrightarrow B = (\overline{A} \vee B) \wedge (A \vee \overline{B})$

Формулы поглощения и склеивания

Формулы поглощения	$A \vee (A \wedge B) = A$ $A \wedge (A \vee B) = A$ $A \vee (\overline{A} \wedge B) = A \vee B$ $A \wedge (\overline{A} \vee B) = A \wedge B$
Формулы склеивания	$A \wedge B \vee A \wedge \overline{B} = A$ $(A \vee B) \wedge (A \vee \overline{B}) = A$

Рассмотрим применение законов и тождеств алгебры логики для упрощения логических функций.

Пример 2

Используя логические законы и тождества, упростите логическое выражение: $A \wedge (A \vee B) \wedge (B \wedge \bar{B})$.

Решение:

1) Первую скобку раскроем по распределительному закону, во второй скобке воспользуемся тождеством логического умножения, получим:

$$A \wedge (A \vee B) \wedge (B \wedge \bar{B}) = (A \wedge A \vee A \wedge B) \wedge 0 = (A \vee A \wedge B) \wedge 0.$$

Можно и далее преобразовывать выражение в скобках. Но согласно тождеству $A \wedge 0 = 0$,

$$A \wedge (A \vee B) \wedge (B \wedge \bar{B}) = (A \wedge A \vee A \wedge B) \wedge 0 = (A \vee A \wedge B) \wedge 0 = 0.$$

Таким образом, исходное логическое выражение тождественно ложно: принимает значение 0 при любых значениях переменных A и B.

3.3 Формы логических функций

Одна и та же логическая функция может быть записана различным образом.

Для исключения неоднозначности записи логические функции представляют в унифицированных формах. Такими формами являются: **дизъюнктивная** и **конъюнктивная**. В них используются элементарные дизъюнкции и конъюнкции.

Элементарной называется **конъюнкция**, в которую входят только переменные и их отрицания.

Например: $X_1 \wedge X_2;\ X_1 \wedge \bar{X}_2;\ \bar{X}_1 \wedge X_2;\ X_1 \wedge X_2 \wedge X_3;\ X_1 \wedge \bar{X}_2 \wedge \bar{X}_3.$

Элементарной называется **дизъюнкция**, представляющая собой логическую сумму переменных и их отрицаний.

Например: $X_1 \vee X_2;\ X_1 \vee \bar{X}_2;\ X_1 \vee \bar{X}_2 \vee \bar{X}_3.$

В элементарные конъюнкции (дизъюнкции) не могут входить одинаковые переменные, а также переменные с их отрицаниями. Такие дизъюнкции (конъюнкции) должны преобразовываться. При этом они упрощаются, а также превращаются в 0 или 1.

Например: $X_1 \wedge X_1 \wedge X_1 = X_1;\ X_1 \wedge \bar{X}_1 = 0;\ X_1 \vee X_1 \vee X_1 = X_1;$
$X_2 \vee \bar{X}_2 = 1.$

Правильность преобразований может быть проверена подстановкой значений переменных. Элементарная конъюнкция (дизъюнкция) может характеризоваться рангом, равным количеству переменных в конъюнкции (дизъюнкции).

Понятия элементарной конъюнкции и дизъюнкции позволяют достаточно просто определить дизъюнктивную и конъюнктивную формы записи логических функций.

Дизъюнктивная нормальная форма (ДНФ) – это форма, в которой логическая функция представляется в виде дизъюнкции элементарных конъюнкций, например:

$$F = X_1 \wedge X_2 \vee \bar{X}_1 \wedge X_3 \vee X_1 \wedge X_2 \wedge X_3.$$

Конъюнктивной нормальной формой (КНФ) называется такая форма, в которой функция представляется в виде конъюнкции элементарных дизъюнкций, например:

$$F = (X_1 \vee \bar{X}_2) \wedge (\bar{X}_1 \vee X_2 \vee X_3).$$

Использование нормальных форм не устраняет полностью неоднозначности записи логических функций. Поэтому среди нормальных форм выделяются такие, в которых функции записываются единственным образом. Их называют совершенными. Применяются **совершенная**

дизъюнктивная нормальная форма (СДНФ) и **совершенная конъюнктивная нормальная форма (СКНФ)**.

Формы СДНФ и СКНФ имеют две отличительные особенности:

– все элементарные конъюнкции и дизъюнкции имеют одинаковый ранг;

– в элементарные конъюнкции (дизъюнкции) входят все те переменные или их отрицания, от которых зависит функция.

4 СОДЕРЖАНИЕ ОТЧЕТА

Отчет должен содержать тему и цель работы, выполненные задания для самостоятельной работы по указанному преподавателем варианту и ответы на контрольные вопросы.

5 УПРАЖНЕНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

1. Постройте таблицы истинности для следующих логических функций и сделайте вывод об истинности функций:

1) $\bar{A} \wedge \bar{B} \longrightarrow C$	2) $(\bar{A} \longrightarrow C) \wedge D$	3) $A \Leftrightarrow B \vee C$
4) $A \wedge \bar{B} \longrightarrow C$	5) $A \Leftrightarrow B \wedge C$	6) $(A \Leftrightarrow \bar{B}) \vee C$
7) $\bar{A} \wedge (B \longrightarrow C)$	8) $A \vee (C \Leftrightarrow B)$	9) $A \vee (C \Leftrightarrow \bar{B})$
10) $A \wedge (B \Leftrightarrow C)$	11) $A \wedge (\bar{B} \Leftrightarrow C)$	12) $\overline{A \Leftrightarrow B} \wedge C$

2. Используя логические законы и тождества, упростите следующие логические выражения:

1) $\bar{X} \vee \overline{(\bar{Y} \vee X)}$	2) $A \wedge B \wedge (C \longrightarrow D) \vee B \wedge \bar{C} \wedge A \vee B \vee B \wedge A$
3) $(A \vee B) \wedge (\bar{A} \vee B)$	4) $\bar{B} \wedge \bar{A} \vee A \wedge B \vee (A \longrightarrow B) \vee \bar{A} \wedge \bar{B}$
5) $\bar{B} \wedge (A \vee \overline{\bar{A} \wedge \bar{B}})$	6) $\bar{A} \wedge B \vee \bar{B} \wedge A \vee \bar{A} \wedge (B \longrightarrow A)$
7) $(A \vee B) \wedge (A \vee \bar{B})$	8) $A \wedge B \wedge \overline{(C \longrightarrow D)} \vee B \vee B \wedge A$
9) $\overline{\bar{X} \wedge \bar{Y}} \vee \bar{X}$	10) $\bar{B} \wedge \bar{A} \vee A \wedge B \vee \overline{(A \longrightarrow B)} \vee \bar{A} \wedge \bar{B}$
11) $\bar{X} \wedge \bar{Y} \vee \bar{X} \wedge Y \vee \bar{X} \wedge Y$	12) $\bar{X} \wedge (\bar{Y} \longrightarrow \bar{X}) \wedge Y \vee \bar{X} \wedge Y$

6 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Дайте определение алгебры логики.
2. Что называется логическим высказыванием?
3. Какое логическое высказывание называется простым? Сложным?
4. Перечислите основные и дополнительные логические операции.
5. Что называется таблицей истинности?
6. Напишите таблицы истинности для основных и дополнительных логических операций.
7. Напишите тождества алгебры логики.
8. Напишите законы алгебры логики.

ЛАБОРАТОРНАЯ РАБОТА № 3

Тема: **ЛОГИЧЕСКИЕ ОСНОВЫ РАБОТЫ КОМПЬЮТЕРА. ОСНОВНЫЕ ЛОГИЧЕСКИЕ ЭЛЕМЕНТЫ. ЛОГИЧЕСКИЕ СХЕМЫ**

1 ЦЕЛЬ РАБОТЫ

Познакомиться:
- с правилами построения СКНФ и СДНФ;
- с основными логическими элементами.

Научиться:
- составлять СКНФ и СДНФ по таблице истинности логической функции;
- составлять логические схемы.

2 ОБЪЁМ ЛАБОРАТОРНОЙ РАБОТЫ И ПОРЯДОК ЕЁ ВЫПОЛНЕНИЯ

Данная работа выполняется в течение двух часов. Выполнение работы идет одновременно с изучением теоретической части. Для закрепления материала необходимо выполнить упражнения для самостоятельной работы.

3 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ. ТЕХНОЛОГИЯ ВЫПОЛНЕНИЯ РАБОТЫ

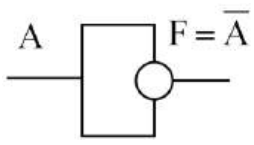
3.1 Основные логические элементы

Средством обработки двоичных сигналов в ЭВМ являются логические элементы.

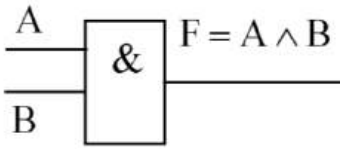
Для реализации любых логических операций над двоичными сигналами достаточно элементов трех типов – элементов, реализующих три основные логические операции: **НЕ, И , ИЛИ**.

Логические элементы (вентили) – это электронные схемы с одним или несколькими входами и одним выходом, через которые проходят электрические сигналы, представляющие цифры 0 и 1.

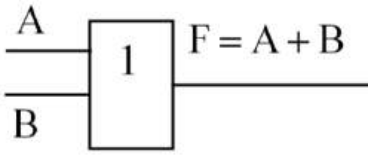
1. Элемент **НЕ** (**инвертор**). У инвертора один вход и один выход. Сигнал на выходе появится, если на входе его нет, и наоборот. Инвертор реализует логическую операцию **инверсия** (**логическое отрицание**).

Функция	Таблица истинности	Логический элемент
$F = \bar{A}$	A | F 0 | 1 1 | 0	

2. Элемент **И** (**конъюнктор**). Элемент **И** имеет не менее двух входов и один выход. **A** и **B** – входные сигналы, **F** – выходной сигнал. Конъюнктор реализует логическую операцию **конъюнкция** (**логическое умножение**).

Функция	Таблица истинности	Логический элемент
$F = A \wedge B$	A | B | F 0 | 0 | 0 0 | 1 | 0 1 | 0 | 0 1 | 1 | 1	

3. Элемент **ИЛИ** (**дизъюнктор**). Элемент **ИЛИ** имеет не менее двух входов и один выход. **A** и **B** – входные сигналы, **F** – выходной сигнал. Дизъюнктор реализует логическую операцию **дизъюнкция** (**логическое сложение**).

Функция	Таблица истинности	Логический элемент
$F = A \vee B$	A | B | F 0 | 0 | 0 0 | 1 | 1 1 | 0 | 1 1 | 1 | 1	

С помощью логических элементов **НЕ**, **И**, **ИЛИ** можно реализовать (собрать как из конструктора) типовые функциональные узлы (блоки) ЭВМ:

- триггеры;
- сумматоры;
- шифраторы;
- регистры;
- счетчики;
- дешифраторы.

Чтобы понять, как работает интересующее нас устройство, необходимо понять логику его работы, т.е. найти соответствие между входными и выходными сигналами, для этого необходимо:

1. Составить таблицу истинности.
2. По таблице записать логическую функцию.
3. Упростить построенную логическую функцию (если возможно).
4. Построить логическую схему.

3.2 Построение логической схемы по заданной таблице истинности логической функции с помощью СДНФ

1. По заданной таблице определяется логическая функция (формула) с помощью следующего метода, называемого совершенная дизъюнктивная нормальная форма (СДНФ):
   1.1. В заданной таблице выбираются наборы переменных, при которых значение функции равно 1.
   1.2. Для каждого такого набора записываются конъюнкции (∧) всех входных переменных, имеющих значение 1. При этом те переменные, которые имеют значение 0, записываются с отрицанием.
   1.3. Все полученные конъюнкции объединяются знаком дизъюнкции (∨). Это и будет искомая логическая функция, которую можно будет упростить (минимизировать) по законам Булевой алгебры.
2. По упрощенной логической функции строится логическая схема.

Пример 1

По заданной таблице истинности записать логическую функцию, упростить её и построить логическую схему.

X	Y	Z	F
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

Решение:

1. Построим СДНФ по данной таблице истинности. Запишем конъюнкцию для каждой строки, где значение функции равно 1 (первая, вторая, третья и седьмая строки). Переменные, значения которых равны 0, запишем с отрицанием (см. таблицу ниже).

X	Y	Z	F	*Вспомогательный столбец*
0	0	0	1	$\bar{X} \wedge \bar{Y} \wedge \bar{Z}$
0	0	1	1	$\bar{X} \wedge \bar{Y} \wedge Z$
0	1	0	1	$\bar{X} \wedge Y \wedge \bar{Z}$
0	1	1	0	
1	0	0	0	
1	0	1	0	
1	1	0	1	$X \wedge Y \wedge \bar{Z}$
1	1	1	0	

2. Объединив полученные конъюнкции дизъюнкцией, получим следующую логическую функцию.

$$F = (\bar{X} \wedge \bar{Y} \wedge \bar{Z}) \vee (\bar{X} \wedge \bar{Y} \wedge Z) \vee (\bar{X} \wedge Y \wedge \bar{Z}) \vee (X \wedge Y \wedge \bar{Z}).$$

3. Упростим:

$$F = (\bar{X} \wedge \bar{Y} \wedge \bar{Z}) \vee (\bar{X} \wedge \bar{Y} \wedge Z) \vee (\bar{X} \wedge Y \wedge \bar{Z}) \vee (X \wedge Y \wedge \bar{Z}) =$$
$$= \bar{X} \wedge \bar{Y} \wedge (\bar{Z} \vee Z) \vee Y \wedge \bar{Z} \wedge (\bar{X} \vee X) = \bar{X} \wedge \bar{Y} \vee Y \wedge \bar{Z}.$$

4. По полученной функции построим логическую схему (рисунок 3.1).

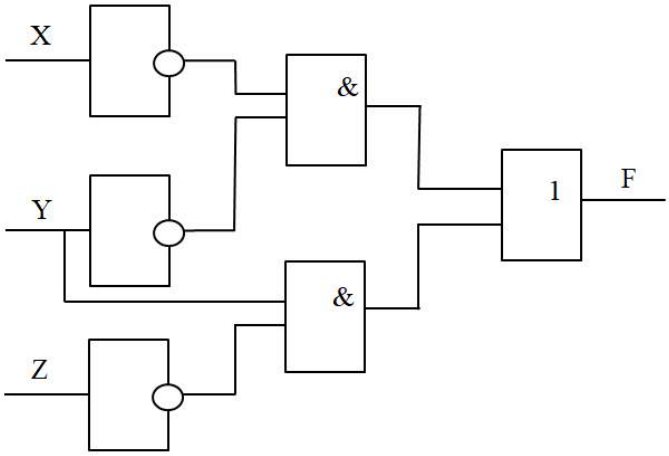


Рисунок 3.1 – Логическая схема

3.3 Построение логической схемы по заданной таблице истинности логической функции с помощью СКНФ

1. По заданной таблице определяется логическая функция (формула) с помощью следующего метода, называемого совершенная конъюнктивная нормальная форма (СКНФ):
   1.1. В заданной таблице выбираются наборы переменных, при которых значение функции равно 0.
   1.2. Для каждого такого набора записываются дизъюнкция (∨) всех входных переменных, имеющих значение 0. При этом те переменные, которые имеют значение 1, записываются с отрицанием.
   1.3. Все полученные дизъюнкции объединяются знаком конъюнкции (∧). Это и будет искомая логическая функция, которую можно будет упростить (минимизировать) по законам Булевой алгебры.
2. По упрощенной логической функции строится логическая схема.

Пример 2

По заданной таблице истинности записать логическую функцию, упростить её и построить логическую схему.

X	Y	Z	F
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

Решение:

1. Построим СКНФ по данной таблице истинности. Запишем дизъюнкции для каждой строки, где значение функции равно 0 (четвёртая строка). Переменные, значения которых равны 1, запишем с отрицанием.

X	Y	Z	F	*Вспомогательный столбец*
0	0	0	1	
0	0	1	1	
0	1	0	1	
0	1	1	0	$X \vee \bar{Y} \vee \bar{Z}$
1	0	0	0	$\bar{X} \vee Y \vee Z$
1	0	1	0	$\bar{X} \vee Y \vee \bar{Z}$
1	1	0	1	
1	1	1	0	$\bar{X} \vee \bar{Y} \vee \bar{Z}$

2. Объединив полученные конъюнкции дизъюнкцией, получим следующую логическую функцию.

$$F = (X \vee \bar{Y} \vee \bar{Z}) \wedge (\bar{X} \vee Y \vee Z) \wedge (\bar{X} \vee Y \vee \bar{Z}) \wedge (\bar{X} \vee \bar{Y} \vee \bar{Z}).$$

3. Упростим:

$$\mathrm{F} = (X \vee \bar{Y} \vee \bar{Z}) \wedge (\bar{X} \vee Y \vee Z) \wedge (\bar{X} \vee Y \vee \bar{Z}) \wedge (\bar{X} \vee \bar{Y} \vee \bar{Z}) =$$
$$= (X \vee \bar{Y} \vee \bar{Z}) \wedge (\bar{X} \vee \bar{Y} \vee \bar{Z}) \wedge (\bar{X} \vee Y \vee Z) \wedge (\bar{X} \vee Y \vee \bar{Z}) =$$
$$= \big((\bar{Y} \vee \bar{Z}) \vee X\big) \wedge \big((\bar{Y} \vee \bar{Z}) \vee \bar{X}\big) \wedge ((\bar{X} \vee Y) \vee Z) \wedge ((\bar{X} \vee Y) \vee \bar{Z}) =$$
$$= (\bar{Y} \vee \bar{Z}) \wedge (\bar{X} \vee Y).$$

4. По полученной функции построим логическую схему (рисунок 3.2).

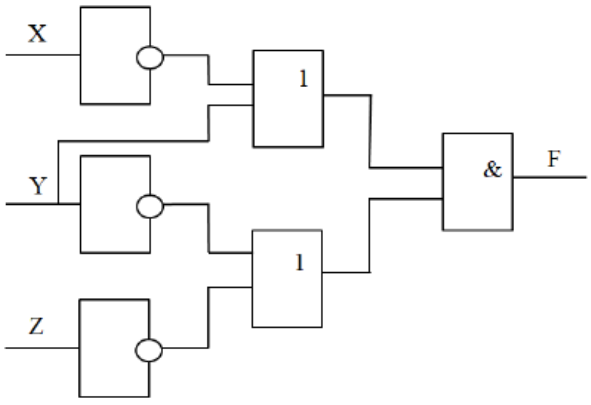


Рисунок 3.2 – Логическая схема

Вывод: при решении примеров 1 и 2 была использована таблица истинности для одной и той же логической функции, но методы нахождения искомой логической функции были разные (СКНФ и СДНФ). В результате были получены различные логические схемы, но результат работы полученных логических схем будет одинаковый (проверить это можно, построив таблицы истинности по полученным логическим схемам).

3.4 Построение таблицы истинности по заданной логической схеме

Пример 3

По заданной логической схеме запишите логическую функцию и постройте таблицу истинности (рисунок 3.3).

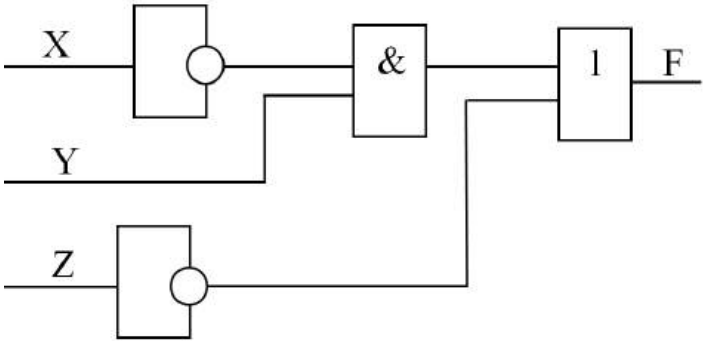


Рисунок 3.3 – Логическая схема

Решение:

1. Для записи функции необходимо записать значения для каждого элемента схемы:
   1) $\bar{X}$
   2) $\bar{X} \wedge Y$
   3) $\bar{Z}$
   4) $(\bar{X} \wedge Y) \vee \bar{Z}$

2. Следовательно, получится функция:

$$F = (\bar{X} \wedge Y) \vee \bar{Z}.$$

3. По полученной функции построим таблицу истинности:

X	Y	Z	$\bar{X}$	$\bar{X} \wedge Y$	$\bar{Z}$	$(\bar{X} \wedge Y) \vee \bar{Z}$
0	0	0	1	0	1	1
0	0	1	1	0	0	0
0	1	0	1	1	1	1
0	1	1	1	1	0	1
1	0	0	0	0	1	1
1	0	1	0	0	0	0
1	1	0	0	0	1	1
1	1	1	0	0	0	0

3.5 Построение логической схемы технологического процесса

В инженерной практике часто возникает необходимость разработки таких систем автоматизации, в которых к управляющей вычислительной машине подключены дискретные датчики, и работа управляющих органов может быть осуществлена дискретными сигналами. В отдельных случаях при проектировании указанных систем следует использовать методику синтеза дискретной системы управления технологическим процессом, первым этапом которой является анализ технологического процесса и построение таблицы истинности. Второй этап – составление логических функций для каждого из дискретных входов и запись логической схемы работы.

Ниже рассмотрим пример составления логической функции и логической схемы технологического процесса на основе составленной таблицы истинности.

Пример 3

Дана таблица истинности, построенная на основе анализа технологического процесса. Запишите логическую функцию и упростите её. По полученной логической функции составьте логическую схему технологического процесса.

A	*B*	*C*	*F*
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Решение**:**

1. Составим логическую функцию по данной таблице:

$$F = (A \wedge \bar{B} \wedge C) \vee (A \wedge B \wedge \bar{C}) \vee (A \wedge B \wedge C).$$

2. Упростим полученную функцию:

$$F = (A \wedge \bar{B} \wedge C) \vee (A \wedge B \wedge \bar{C}) \vee (A \wedge B \wedge C) =$$
$$= (A \wedge \bar{B} \wedge C) \vee \left(A \wedge B \wedge (\bar{C} \vee C)\right) = A \wedge \bar{B} \wedge C \vee A \wedge B =$$
$$= A \wedge \overline{(B \wedge C \vee B)}.$$

3. По полученной функции построим логическую схему (рисунок 3.4).

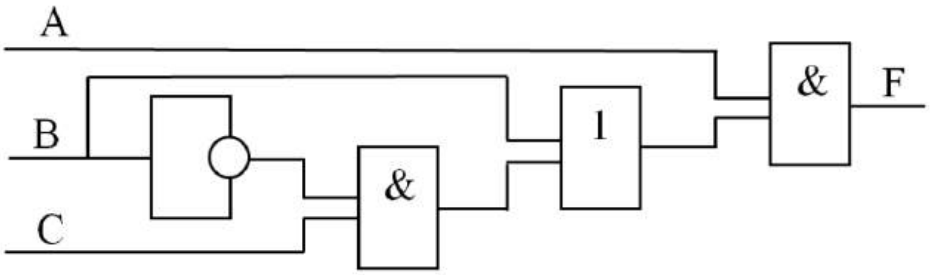


Рисунок 3.4 – Логическая схема

4 СОДЕРЖАНИЕ ОТЧЕТА

Отчет должен содержать тему и цель работы, выполненные задания для самостоятельной работы по указанному преподавателем варианту и ответы на контрольные вопросы.

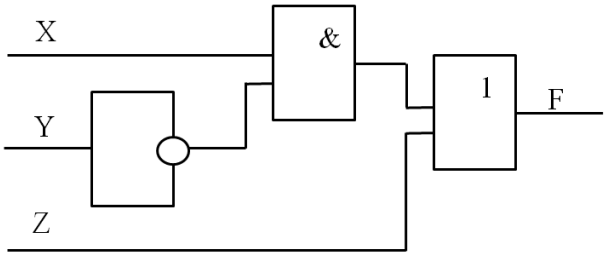
5 УПРАЖНЕНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

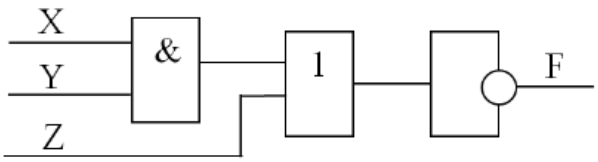
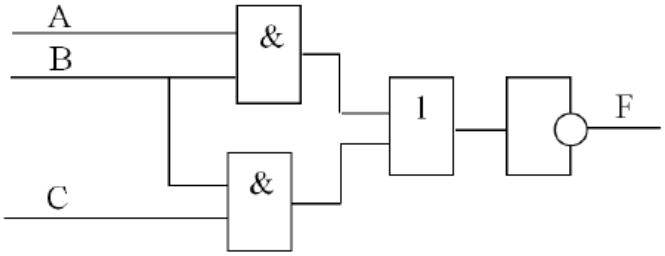
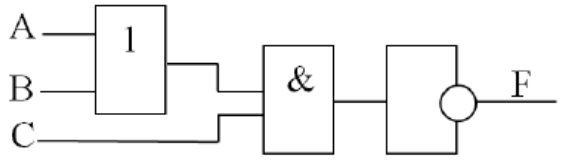
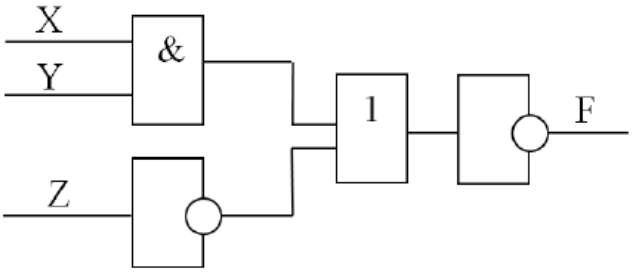
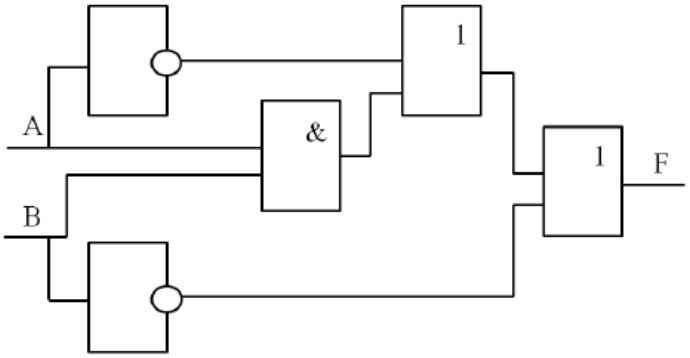
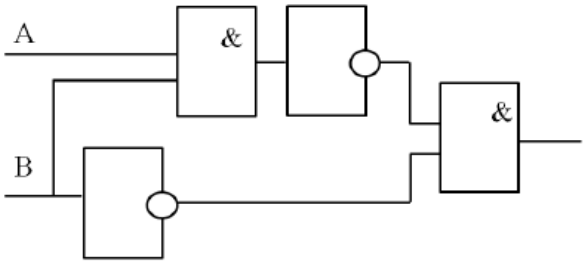
1. Составьте логические схемы для следующих логических выражений:

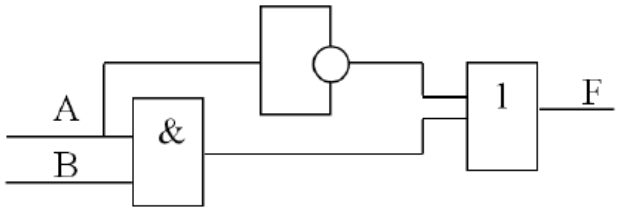
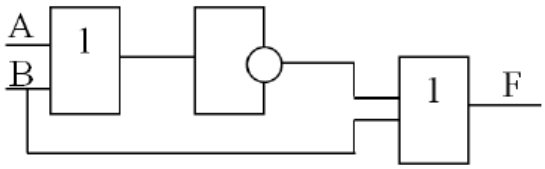
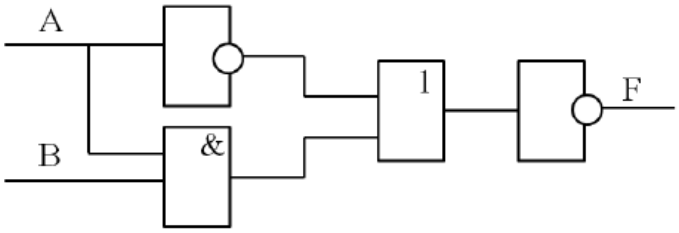
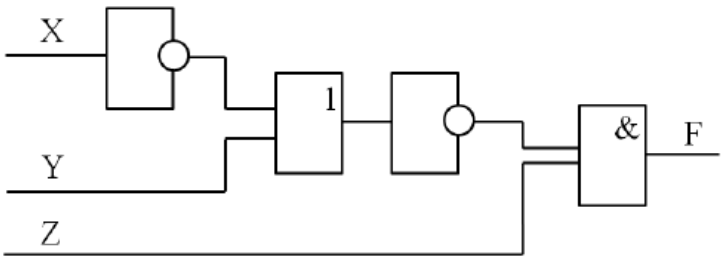
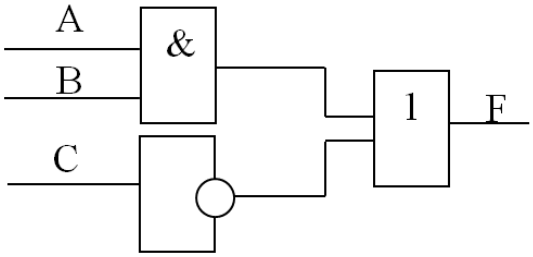
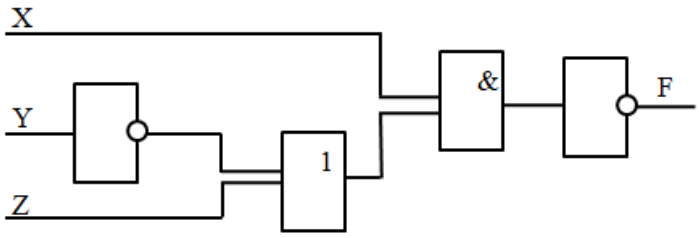
1) $\overline{\left(X \wedge (\bar{X} \to \bar{Y})\right)} \vee Y$
2) $\overline{(A \wedge B \wedge \bar{C} \vee C)} \vee \bar{B} \vee C$
3) $\bar{A} \wedge B \vee A \wedge B \vee \bar{A}$
4) $X \vee \overline{(\bar{Y} \vee X)} \to Y$
5) $\bar{B} \wedge (A \vee \overline{A \wedge B}) \vee \bar{A} \wedge B$
6) $(A \vee B) \wedge (\bar{A} \vee B)$
7) $(A \vee B) \wedge (A \to \bar{B})$
8) $\bar{X} \wedge (\bar{Y} \longrightarrow \bar{X}) \wedge Y \vee \bar{X} \wedge Y$
9) $\bar{X} \wedge \bar{Y} \vee \bar{X} \wedge Y \vee \bar{X} \wedge Z$
10) $\bar{X} \vee \overline{X \vee Y} \vee \overline{Y \wedge X \wedge Z}$
11) $\overline{X \vee Z \vee \overline{\bar{X} \wedge \bar{Y}}} \wedge \overline{Y \vee X}$
12) $(A \Leftrightarrow B) \vee \bar{A} \wedge B$

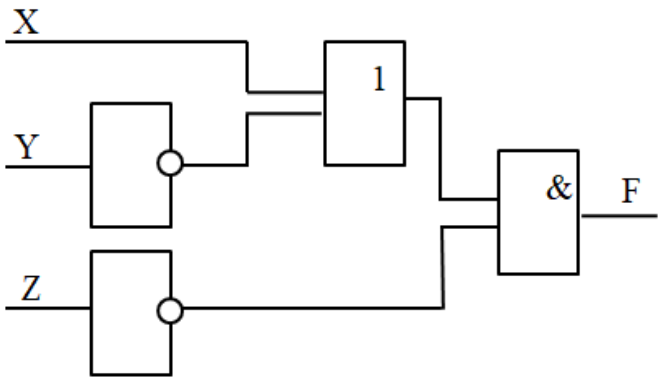
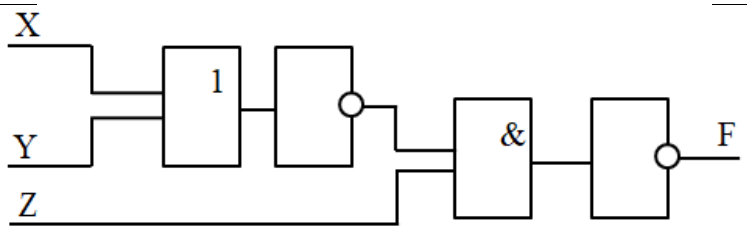
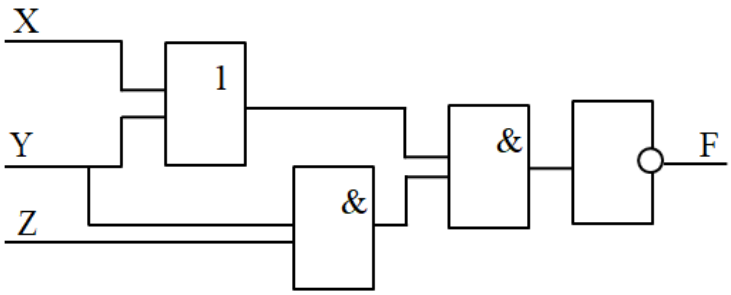
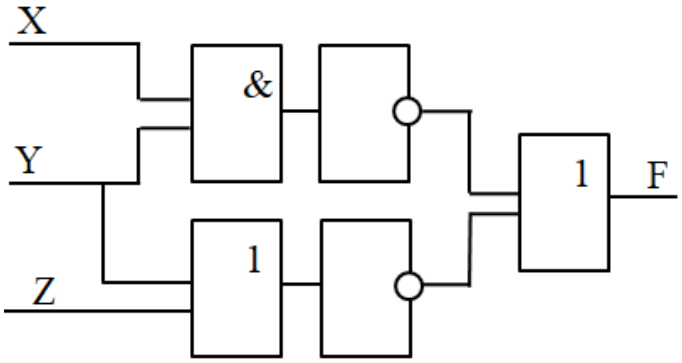
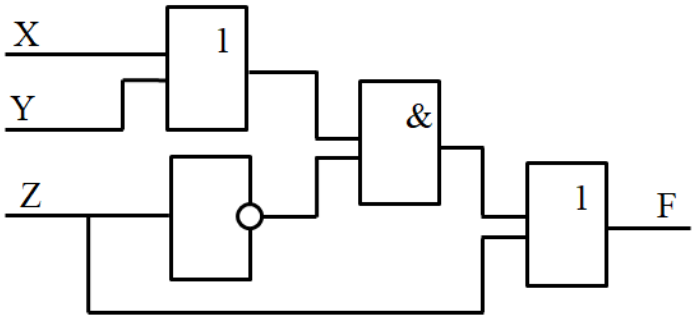
2. По логическим схемам, полученным в примерах 1 и 2 лабораторной работы, постройте таблицы истинности и сравните результаты. Сделайте вывод.

3. По заданным логическим схемам составить соответствующие им логические выражения и упростите их:

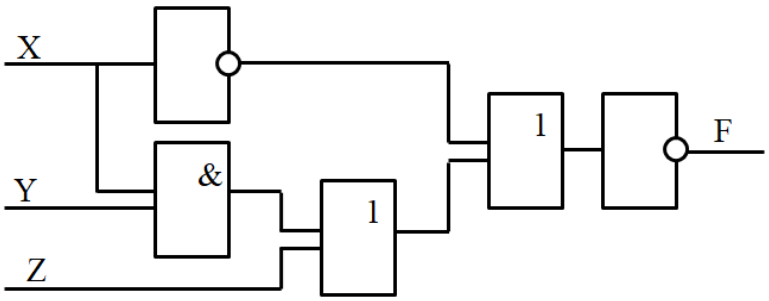
№	Схема
1	2
1)	

1	2
2)	X Y Z & 1 F
3)	A B C & & 1 F
4)	A B C 1 & F
5)	X Y Z & 1 F
6)	A B & 1 1 F
7)	A B & &

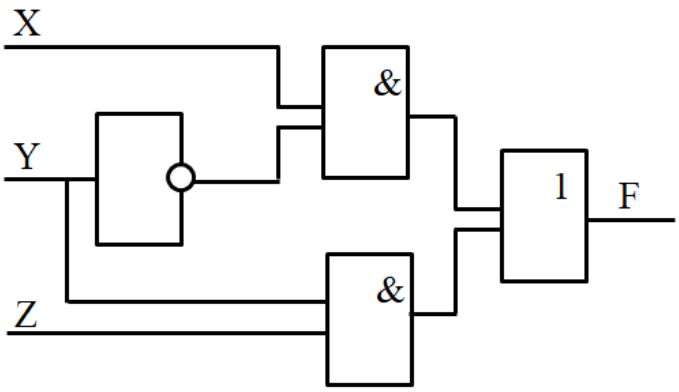
1	2
8)	
9)	
10)	
11)	
12)	
13)	

14)	X Y Z 1 & F
15)	X Y Z 1 & F
16)	X Y Z 1 & & F
17)	X Y Z & 1 1 F
18)	X Y Z 1 & 1 F

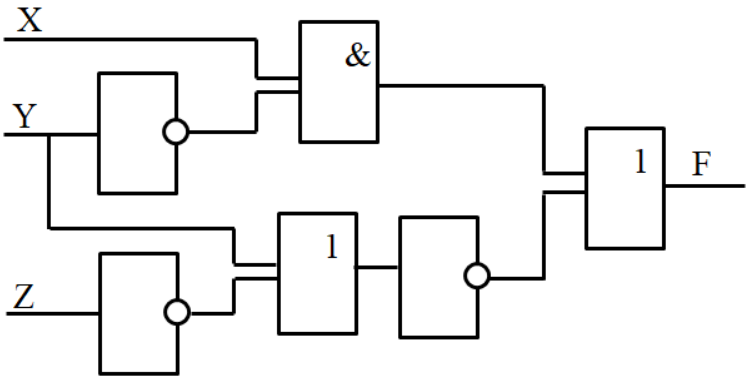
19)



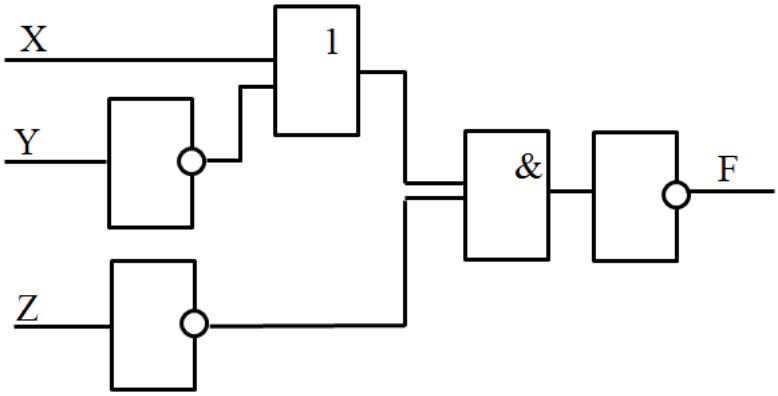
20)



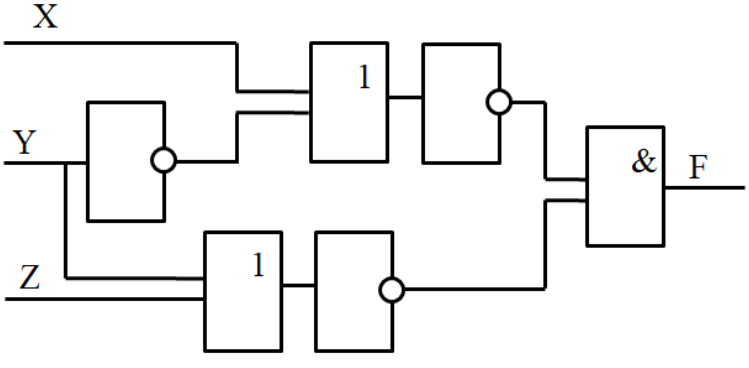
21)



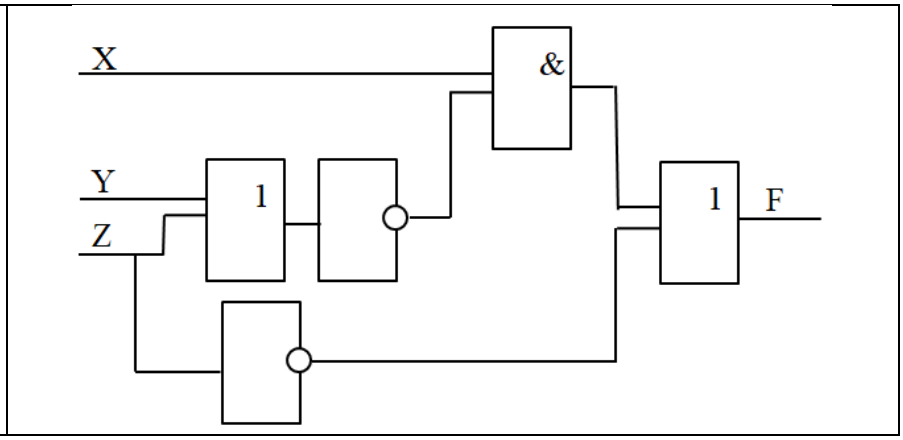
22)



23)



24)



4. Дана таблица истинности, построенная на основе анализа технологического процесса. Запишите логическую функцию и упростите её. По полученной логической функции составьте логическую схему.

№	Таблица	№	Таблица
1	2	3	4

1)

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

2)

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

3)

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

4)

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

5)

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

6)

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

7)

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

8)

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

9)

A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

10)

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

11)

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

12)

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

13)

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

14)

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

15)

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

16)

A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

17)

A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

18)

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

19)

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

20)

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

21)

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

22)

A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

23)

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

24)

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

6 КОНТРОЛЬНЫЕ ВОПРОСЫ

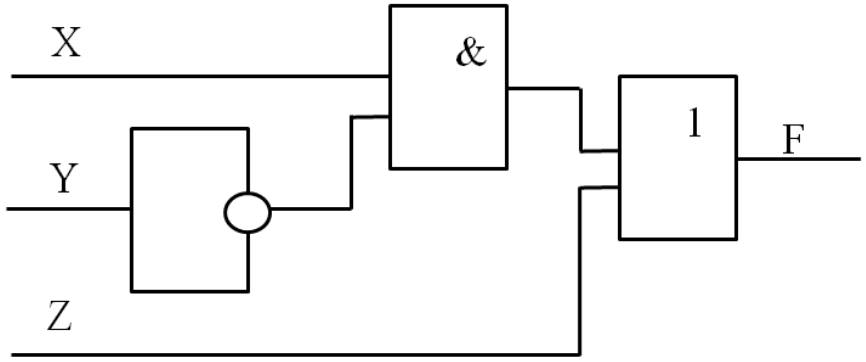
1. Что называется логическим элементом?
2. Напишите основные логические функции и соответствующие им логические элементы.
3. Какие типовые узлы ЭВМ можно реализовать с помощью основных логических элементов?
4. Опишите алгоритм построения логической схемы по заданной таблице истинности логической функции?
5. Опишите алгоритм построения логической функции и её таблицы истинности по заданной логической схеме?

ЛАБОРАТОРНАЯ РАБОТА № 4

КОНТРОЛЬНАЯ РАБОТА № 1.
«ОБРАБОТКА И АНАЛИЗ ИНФОРМАЦИИ,
ПРЕДСТАВЛЕНИЕ ИНФОРМАЦИИ В РАЗЛИЧНЫХ ФОРМАТАХ»

Примерный вариант

1. Переведите число $81{,}4_{10}$ из десятичной системы счисления в двоичную.
2. Переведите число $11{,}75_8$ в десятичную систему счисления.
3. Решите задачу. Сколько дискет объемом 1,44 Мбайт необходимо для хранения энциклопедии из 60 томов по 500 страниц, на каждой из которых по две полосы, содержащей 80 строк из 45 знаков?
4. Составьте таблицу истинности для логического выражения: $F = \bar{A} \to \bar{B}$.
5. По заданной логической схеме составьте соответствующее ей логическое выражение и упростите его:



Результат выполнения контрольной работы № 1

1. Переведите число $81{,}4_{10}$ из десятичной системы счисления в двоичную.

Решение:

Перевод целой части

81	40	20	10	5	2	1
80	40	20	10	4	2	0
1	0	0	0	1	0	

Перевод дробной части

0,	4
	2
0	8
	2
1	6
	2
1	2

2. Переведите число $11{,}75_8$ в десятичную систему счисления.

Решение:

$$11{,}75_8 = 1 \cdot 8^1 + 1 \cdot 8^0 + 7 \cdot 8^{-1} + 5 \cdot 8^{-2}$$

(1 0 -1 -2)

3. Составьте таблицу истинности для логического выражения: $F = \bar{A} \to \bar{B}$.

Решение:

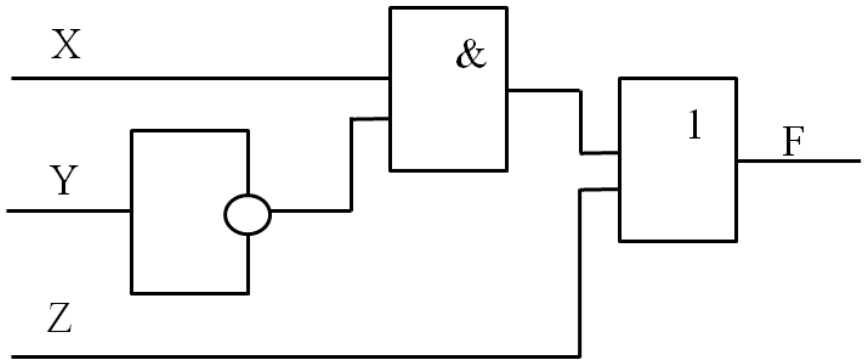
1) Расставим порядок операций над логическим выражением:

$$F = \overset{1}{\bar{A}} \overset{3}{\to} \overset{2}{\bar{B}}$$

2) Составим таблицу истинности, учитывая порядок выполнения логических операций:

A	B	$\bar{A}$	$\bar{B}$	$\bar{A} \to \bar{B}$
0	0	1	1	1
0	1	1	0	0
1	0	0	1	1
1	1	0	0	1

4. По заданной логической схеме составьте соответствующее ей логическое выражение и упростите его:



Решение:

1) Необходимо учесть, что количество логических элементов в логической схеме соответствует количеству логических операций в логическом выражении. В задаче в логической схеме 3 логических элемента, следовательно в логическом выражении будет 3 логические функции.
2) Запишем, в порядке следования в логической схеме, логические операции. В результате получится логическое выражение: $\bar{Y} \wedge X \vee Z$

5. По заданной таблице истинности запишите логическую функцию, упростите ее и составьте логическую схему.

A	B	F
0	0	0
0	1	1
1	0	1
1	1	1

Решение:

1) Запишем конъюнкцию для каждой строки, где значение функции равно 1 (вторая, третья и четвертая строки). Переменные, значения которых равны 0, запишем с отрицанием

A	B	F	Вспомогательный столбец
0	0	0	
0	1	1	$\bar{A} \wedge B$
1	0	1	$A \wedge \bar{B}$
1	1	1	$A \wedge B$

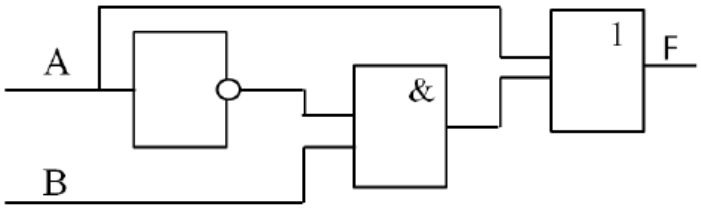
2) Объединив полученные конъюнкции дизъюнкцией, получим следующую логическую функцию:

$$F = (\bar{A} \wedge B) \vee (\mathrm{A} \wedge \bar{B}) \vee (\mathrm{A} \wedge \mathrm{B})$$

3) Упростим функцию:

$$F = (\bar{A} \wedge B) \vee (\mathrm{A} \wedge \bar{B}) \vee (\mathrm{A} \wedge \mathrm{B}) = \bar{A} \wedge B \vee \mathrm{A} \wedge (\bar{B} \vee B) = \bar{A} \wedge B \vee \mathrm{A} \wedge 1$$
$$= \bar{A} \wedge B \vee \mathrm{A}$$

4) По полученной функции построим логическую схему:



ЛАБОРАТОРНАЯ РАБОТА № 5

Тема: **ОС LINUX. ОСНОВНЫЕ ПРИНЦИПЫ ФУНКЦИОНИРОВАНИЯ ОПЕРАЦИОННОЙ СИСТЕМЫ. РАБОТА В РЕЖИМЕ КОМАНДНОЙ СТРОКИ**

1 ЦЕЛЬ РАБОТЫ

Познакомиться:
- с интерфейсом рабочего стола ОС Simply Linux;
- с Меню запуска приложений;
- с файловым менеджером Thunar;
- с файловой системой ОС Simply Linux.

Научиться:
- запускать ОС Simply Linux;
- работать с Меню запуска приложений;
- работать с файловым менеджером;
- запускать Терминал;
- создавать в Терминале файлы и папки;
- выполнять в Терминале основные команды работы с файлами и папками.

2 ОБЪЁМ ЛАБОРАТОРНОЙ РАБОТЫ И ПОРЯДОК ЕЁ ВЫПОЛНЕНИЯ

Данная работа выполняется в течение двух часов. Выполнение работы идет одновременно с изучением теоретической части. Для закрепления материала необходимо выполнить упражнения для самостоятельной работы.

3 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ. ТЕХНОЛОГИЯ ВЫПОЛНЕНИЯ РАБОТЫ

3.1 Основные понятия

Операционная система (ОС) Simply Linux – это простая в установке и удобная в работе операционная система, дающая возможность пользователю решать все повседневные задачи, не опасаясь вирусов и не затрачивая время на поиск необходимых программ в сети Интернет и на полках магазинов. ОС Simply Linux легка в использовании, не требовательна к ресурсам компьютера, функциональна и надёжна.

В комплект Simply Linux включено более 30 полностью русифицированных программ, ориентированных на решение большинства потребностей пользователя.

Если на вашем компьютере установлена ОС Windows, то ОС Simply Linux можно установить на виртуальную машину. Где скачать и как

установить виртуальную машину и ОС Simply Linux показано в Приложении А и Приложении Б соответственно.

3.2 Загрузка Simply Linux

Для того чтобы загрузить Simply Linux, достаточно просто включить компьютер. После загрузки на экране появится окно входа в систему (рисунок 5.1).

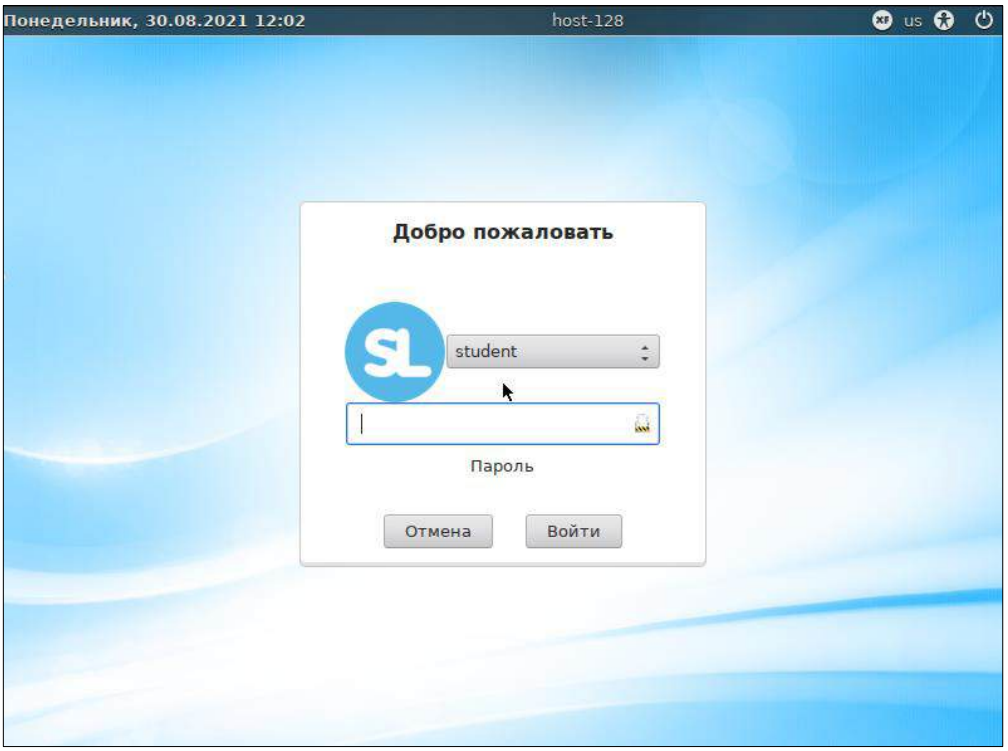


Рисунок 5.1 – Окно входа в систему

После выбора пользователя student необходимо ввести пароль.

Обратите внимание!

Указатель языка ввода с клавиатуры находится в правом верхнем углу окна: на рисунке 5.1 указано, что язык ввода с клавиатуры английский – us. Поэтому для ввода пароля НЕ НЕОБХОДИМО менять раскладку клавиатуры.

В поле ввода введите пароль: student. Операционная система загрузится и на экране появится настольная рабочая среда Xfce 4 (см. рисунок 5.2).

Внимание!

В процессе работы с ОС Simply Linux может произойти блокировка экрана (см. рисунок 5.3). Для разблокирования экрана введите пароль student и щелкните по кнопке Разблокировать.



Рисунок 5.2 – Настольная рабочая среда Xfce 4

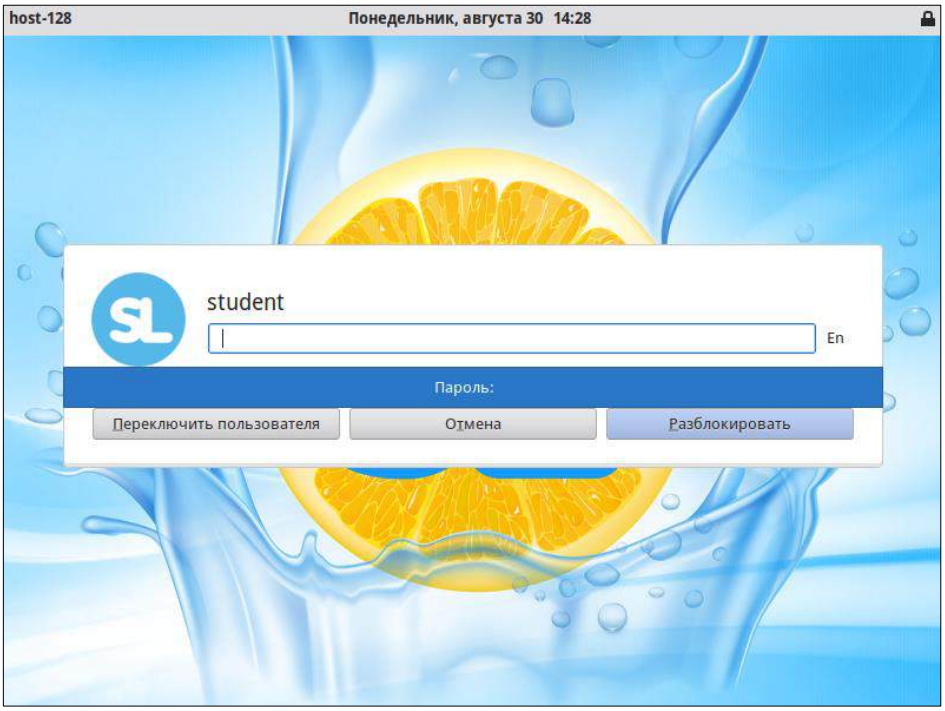


Рисунок 5.3 – Окно блокировки экрана

На рабочем столе Xfce 4 есть две особых области:

1. Область рабочего стола (рабочая область в центре, занимающая большую часть экрана).
2. Панель Xfce (полоса внизу экрана).

Область рабочего стола включает в себя значки:

– **Домашний каталог (Домашняя папка)** – доступ к домашнему каталогу пользователя /home/student/. В этом каталоге (по умолчанию) хранятся пользовательские файлы, например, аудиозаписи, видеозаписи, документы.

– **Корзина** – доступ к «удаленным файлам». Обычно, при удалении файла, он не удаляется из системы. Вместо этого он помещается в «Корзину». С помощью этого значка можно просмотреть или восстановить «удаленные файлы». Чтобы удалить файл из системы, необходимо очистить «Корзину».

При двойном щелчке по значку **Домашний каталог** запустится файловый менеджер с открытым домашним каталогом (рисунок 5.4). При таком же действии по отношению к значку **Корзина** запустится файловый менеджер с открытой корзиной.

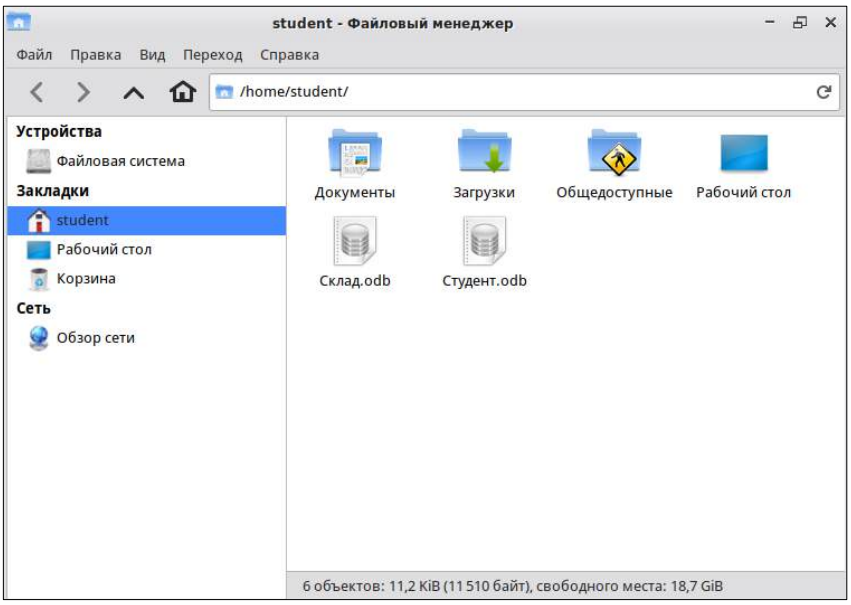


Рисунок 5.4 – Файловый менеджер с открытым домашним каталогом

Щелчок правой кнопкой мыши на свободной области рабочего стола открывает контекстное меню со списком команд.

3.3 Панель Xfce

Панель **Xfce** расположена в нижней части экрана (рисунок 5.5). Панель **Xfce** универсальна: она может содержать значки загрузчика, панели задач, переключатель окон или любое другое сочетание; и её можно удобно настроить. Панель **Xfce** имеет как вертикальный, так и горизонтальный режимы отображения, возможность автоматического скрытия с экрана, легко настраиваемые, всплывающие меню и ярлыки запуска приложений. Панель **Xfce** поддерживает прозрачность.



Рисунок 5.5 – Панель Xfce

На левой части панели расположены:

–  – Меню запуска приложений, обеспечивающее доступ ко всем приложениям и изменениям настроек;

–  – кнопка Скрыть окна и показать рабочий стол, которая позволяет свернуть все открытые окна для быстрого доступа к рабочему столу или восстановить свернутые окна;

–  – значок запуска веб-браузера;

–  – переключатель рабочих мест, позволяющий переходить между доступными рабочими столами с помощью мыши.

На правой части панели находятся:

–  – кнопка переключения между языковыми раскладками клавиатуры;

–  – системный раздел, обеспечивающий доступ к настройке сети, программы обмена сообщениями размещают в нём значки уведомлений;

–  – регулятор громкости для звуковой карты;

–  – часы;

–  – меню Переход, предоставляющее быстрый доступ к каталогам, документам и съёмным носителям ;

–  – кнопка Завершение сеанса..., позволяющая выполнить выход из системы, блокировку экрана и другие системные действия.

В центральной части панели расположена область списка задач, в которой отображаются запущенные приложения.

Упражнение 1. В этом упражнение вы изучите содержимое Меню запуска приложений.

1. Откройте программу **Калькулятор** и выполните вычисления: $\sqrt{(3 * 45 - 35)}$. Для этого выполните команду Меню запуска приложений – Стандартные – Калькулятор. Результат вычислений равен **10**.

2. Откройте домашний каталог с помощью Меню запуска приложений. Для этого выполните команду Меню запуска приложений – Файловый менеджер. На экране откроется домашний каталог текущего пользователя, в данном случае **Student**. Откройте каталог **Документы**.

3. Просмотрите информацию об операционной системе, выполнив команду Меню запуска приложений – Системные – Simply Linux 9.1 (Стартовая страница). В появившемся окне щелкните по пункту **Руководство пользователя** и изучите его содержимое. Закройте все окна.

3.4 Файловый менеджер Thunar

Thunar – это новый, современный файловый менеджер для рабочей среды **Xfce**. **Thunar** отличается быстротой работы и простотой использования. Он быстро запускается и открывает каталоги.

Файловый менеджер **Thunar** является точкой доступа, как к вашим файлам, так и к приложениям. Используя файловый менеджер, вы можете:

- создавать каталоги и документы;
- просматривать файлы и каталоги;
- управлять файлами и каталогами;
- настраивать и выполнять особые действия;
- получать доступ к съёмным носителям.

Окно файлового менеджера (рисунок 5.6) состоит из боковой панели слева, основной области справа и панели адреса, расположенной над основной областью. На боковой панели размещены закладки на различные каталоги вашей системы. Основная область отображает содержимое текущего каталога. Панель адреса всегда показывает путь к текущему каталогу.

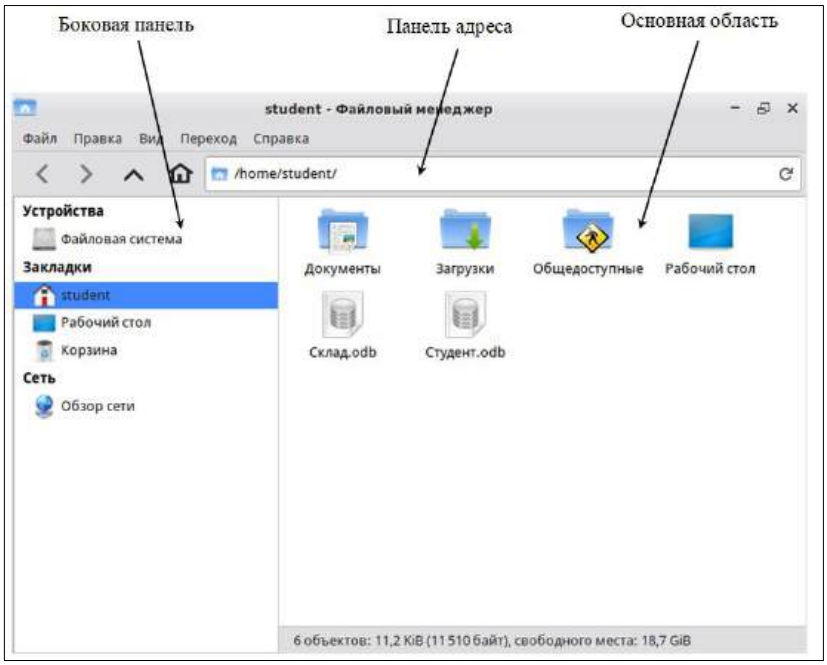


Рисунок 5.6 – Окно файлового менеджера

Чтобы просмотреть свойства файла (каталога), необходимо выделить файл (каталог) и выполнить одно из следующих действий:

- в контекстном меню файла (каталога) выбрать пункт Свойства...;
- в Файловом менеджере необходимо выполнить команду Файл – Свойства...

Окно Свойства объекта показывает подробную информацию о любом файле, каталоге или другом объекте в файловом менеджере (какие именно сведения будут доступны, определяется типом объекта):

– **имя файла или каталога** – можно ввести новое имя, и файл или каталог будут переименованы после нажатия кнопки Закрыть;

– **тип** – тип объекта (например, файл или каталог);

– **адрес** – системный путь к объекту (указывает местонахождение объекта относительно корня системы);

– **изменён** – дата и время последнего изменения объекта;

– **дата доступа** – дата и время последнего просмотра объекта;

– **размер** – количество пространства, которое файл занимает на жестком диске (носителе);

– **использование** – объём свободного пространства на носителе, на котором находится каталог (наибольшее количество данных, которые можно скопировать в этот каталог).

С помощью окна Свойства объекта можно выполнить следующие действия:

– изменить эмблему объекта;

– изменить права на доступ к объекту;

– выбрать, с помощью какого приложения следует открывать данный объект и другие объекты того же типа.

3.5 Файловая система

Файловая система представляет собой иерархическую, древовидную структуру. Самый верхний уровень файловой системы – это корневой каталог, который обозначается символом /.

В ОС Linux всё, включая жесткие диски, разделы и сменные носители, считается файлами. Это значит, что все файлы и каталоги (включая диски и разделы) располагаются в корневом каталоге (рисунок 5.7).

Под корневым каталогом находится ряд важных системных каталогов, которые используются для работы ОС. Список системных каталогов находится в Приложении В.

Цепочка имен каталогов, через которые необходимо пройти для доступа к заданному файлу, вместе с именем этого файла образуют **путь к файлу**. Путь может быть абсолютным (например, /temp/foo – от корневого каталога) или относительным (например, book4/filesystem – от текущего каталога).

Существует ограничение на длину имени файла – не более 255 символов. В имя нельзя включать символ косой черты и нулевые символы. Также есть ограничение на длину пути, который передается ядру в качестве аргумента системного вызова – 4095 байт.

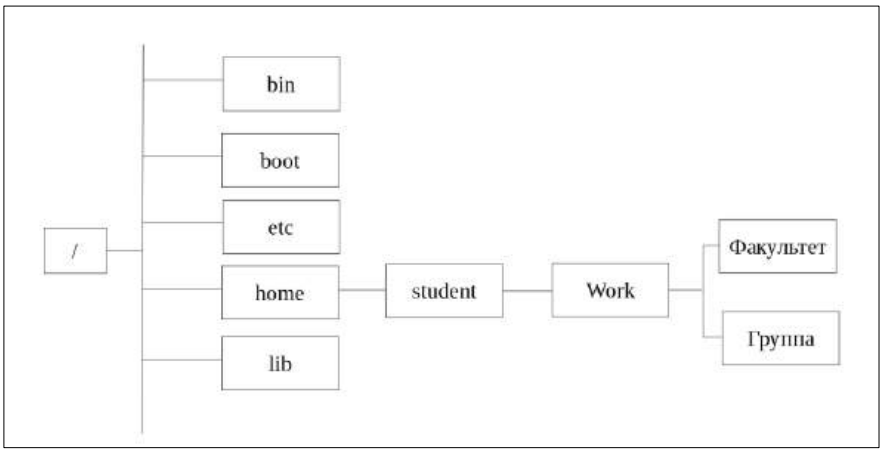


Рисунок 5.7 – Файловая система ОС Linux

3.6 Домашний каталог (Домашняя папка)

Все файлы и каталоги пользователя хранятся в системе внутри домашнего каталога /home/student. Открыть его можно, щёлкнув на значке **Домашний каталог** на рабочем столе. Откроется файловый менеджер **Thunar**, позволяющий просматривать содержимое дерева каталогов, удалять, переименовывать и производить прочие операции над файлами и каталогами.

Домашний каталог есть у каждого пользователя системы, и по умолчанию содержащиеся в нём файлы недоступны для других пользователей (даже для чтения).

В домашнем каталоге по умолчанию находятся несколько стандартных каталогов:

– **Документы** – каталог, предназначенный для хранения документов;

– **Загрузки** – в данный каталог по умолчанию загружаются файлы из Интернета;

– **Рабочий стол** – содержит файлы, каталоги и значки, отображающиеся на рабочем столе.

Кроме того, в домашнем каталоге и его подкаталогах можно создавать другие каталоги при помощи команды контекстного меню Создать каталог...

Упражнение 2. В этом упражнении вы изучите навигацию с помощью Файлового менеджера.

1. Откройте домашний каталог /home/student/. Обратите внимание, что в адресной строке указан путь к каталогу student, текущим является каталог /home/student/.

2. Создайте в текущем каталоге каталог **Work**. Для этого выполните команду Файл – Создать каталог..., введите **Work** и нажмите **Enter**. Создать каталог можно также с помощью команды Создать каталог... контекстного меню, щелкнув правой кнопкой мыши по свободному месту Основной области.

3. Откройте каталог **Work** и создайте в нём два новых каталога – **Факультет** и **Группа**. Таким образом, получим следующую структуру каталогов (рисунок 5.8).

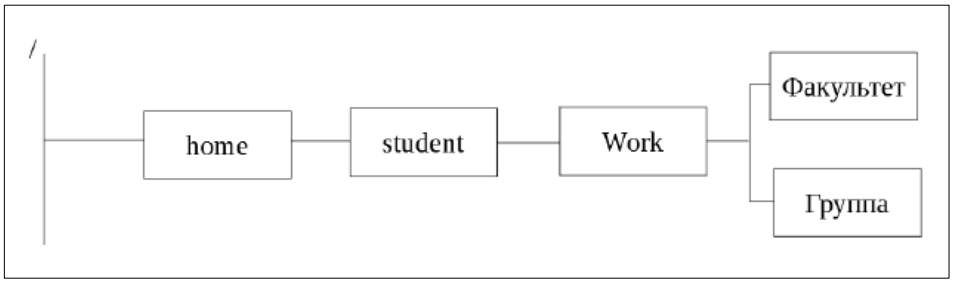


Рисунок 5.8 – Структура каталогов

4. Выведите файловую систему в иерархическом виде (Дерево). Для этого выполните команду Вид – Боковая панель – Дерево. Изучите содержимое вашего домашнего каталога (см. рисунок 5.9).

5. Перейдите в каталог **Факультет** и создайте текстовый документ с именем **Пример**, с помощью команды Файл – Создать – Пустой файл. Содержимое файла – ваша фамилия, имя, отчество и номер группы. Сохраните файл и закройте его.

6. Скопируйте файл **Пример** в каталог **Группа**. Откройте файл, добавьте в содержимое название факультета и сохраните файл в каталог **Группа** с новым именем **Студент**.

7. Удалите каталог **Work**.

8. Закройте **Файловый менеджер**.

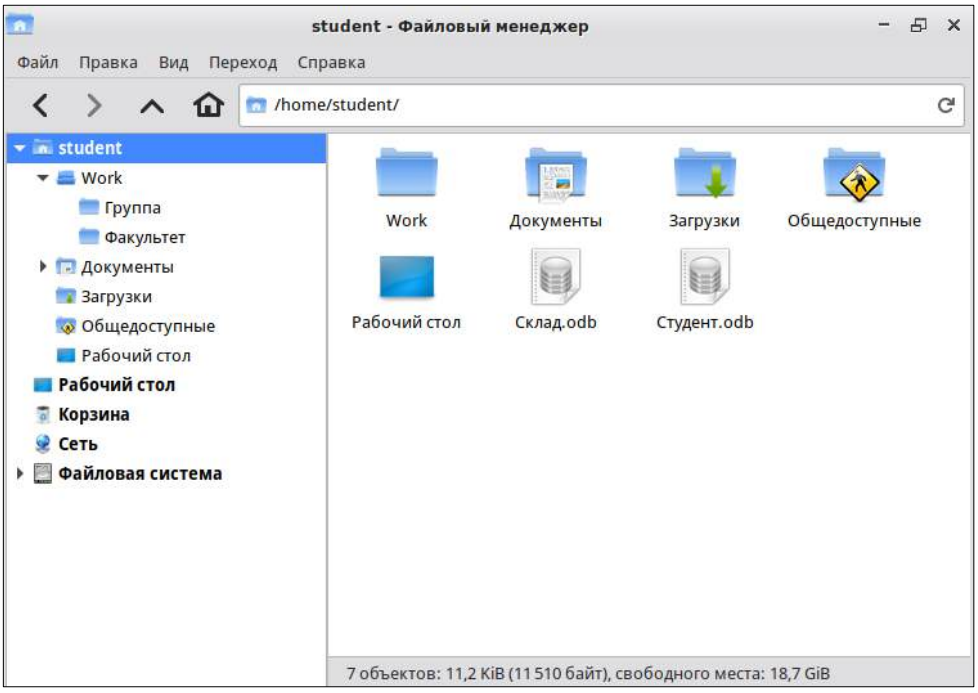


Рисунок 5.9 – Отображение структуры папок в Файловом менеджере

3.7 Терминал и консоль: основные понятия

В Simply Linux существует два вида интерфейса: графический интерфейс пользователя и интерфейс командной строки.

Графический интерфейс пользователя *(англ. Graphical user interface,* ***GUI)*** – управление программами с помощью графических кнопок, всплывающих меню, окон и других элементов. Множество действий можно выполнять с помощью мыши. Преимущества использования GUI: визуальное отображение программ и их содержимого, возможности программ можно изучать без чтения документации.

Интерфейс командной строки *(англ. Command Line Interface,* ***CLI)*** – управление программами с помощью команд. Команды состоят из букв, цифр, символов, набираются построчно, выполняются после нажатия клавиши **Enter**. Основной инструмент здесь клавиатура. Данный интерфейс встроен в ядро системы, он будет доступен, даже если графический интерфейс не запустится.

Преимущества использования командной строки:

1. Командная строка обеспечивает более быстрый доступ к некоторым возможностям операционной системы, а иногда – это единственный способ запустить тот или иной процесс.
2. Текстовый интерфейс менее требовательный к ресурсам компьютера, чем графический.

Терминал – графическая программа, эмулирующая консоль. Такие программы позволяют, не выходя из графического режима, выполнять команды в командной строке. Терминал, по сравнению с консолью, имеет дополнительный функционал (управление мышью, контекстное меню, полосу прокрутки, вкладки, возможность запуска нескольких окон, главное меню, графические настройки) (рисунок 5.10).

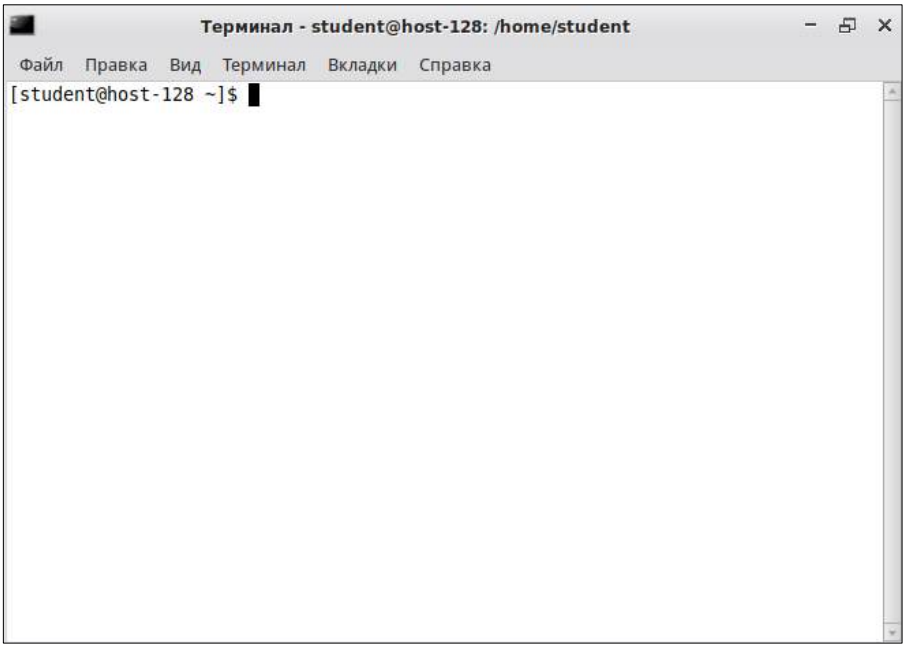


Рисунок 5.10 – Окно Терминала

Консолью называют программное обеспечение, реализующее текстовый интерфейс.

Эмулятор терминала – это графическая программа, которая предоставляет вам доступ к **tty** или **pts** терминалу. Например, Gnome Terminal, Konsole, Terminix, Xterm и многие другие.

Командная оболочка – интерпретатор команд ОС, обеспечивающий интерфейс для взаимодействия пользователя с функциями системы. Командных оболочек достаточно много, это bash, sh, zsh, ksh и другие, но чаще всего применяется bash.

Командная строка – это то место, куда вы будете вводить свои команды, строка приглашения в терминале для ввода.

Запустить терминал можно через Меню запуска приложений, выполнив команду Меню запуска приложений – Эмулятор терминала или в любом открытом каталоге с помощью команды контекстного меню Открыть терминал (графический режим терминала). Для работы в текстовом режиме следует выполнить комбинацию клавиш **Ctrl+Alt+F2** (выход из текстового режима **Ctrl+Alt+F1**).

После запуска терминала, в окне появится приглашение к работе вида **[student@host-128 ~]$** (см. рисунок 5.10). Оно несет в себе информацию об имени пользователя (**student**), имени компьютера (**host-128**) и текущем каталоге (~). Тильда (~) обозначает домашний каталог.

Если на том же компьютере зайдет другой пользователь, то изменится первая строка (до знака @), при работе на другом компьютере изменится также строка после @. А если поменять текущий каталог, то знак тильды будет заменен на имя нового текущего каталога.

3.8 Команды работы с каталогами и файлами

Существует много команд для работы с каталогами и файлами. Эти команды позволяют выполнить различные задачи (получать информацию, работать с текстом, программировать и др.), управлять операционной системой и компьютером (железом).

Чтобы выполнить команду ее необходимо записать после приглашения и нажать **Enter**.

Синтаксис команды:

имя_команды [короткие_ключи] [длинные_ключи] [аргументы]

– имя_команды – командой является любой исполняемый файл (например, ls – команда показывает список файлов в текущем каталоге);

– [короткие_ключи] – короткие ключи состоят из одного символа и начинаются с символа “-“ и указываются через пробел (например, ls -l -a). Для компактности короткие ключи могут объединяться в один ключ (например, ls -la).

– [длинные_ключи] – длинные ключи состоят из нескольких символов, образующих осмысленные слова и начинаются с двух символов, указываются через пробел (например, ls **--**all) и НЕ МОГУТ объединяться в один ключ.

Краткую справку о команде можно получить вызовом команды с ключом **--**help. Перемещение по тексту вниз и вверх с помощью стрелок и клавиш **[PgUp]** и **[PgDn]**. Выход нажатием клавиши **[q]**.

Команда pwd

Назначение: выводит имя текущего каталога, т. е., каталога, с которым пользователь работает в данный момент времени. Текущий каталог необходимо знать при указании относительных путей к файлам и каталогам, также некоторые программы сохраняют результаты своего выполнения в текущий каталог.

Формат: pwd

Упражнение 3.

1. Запустите терминал из домашнего каталога.
2. Выполните команду pwd. В терминале появится строка /home/student, где / – корневой каталог, home – каталог, находящийся в корневом каталоге, student – текущий каталог, вложенный в каталог home.

Команда ls

Назначение: выводит список файлов и каталогов, поддерживает несколько различных форматов вывода.

Формат: ls [ключ1] [ключ2] ... [ключN] [путь к каталогу]

Ключи:

-a – выводит список всех файлов и каталогов, в том числе и скрытые файлы (краткий формат);

-l – выводит список файлов в расширенном формате, показывая тип каждого элемента, полномочия, владельца, размер и дату последней модификации (полный формат);

-r – выводит список в порядке, обратном заданному;

-s – выводит размеры каждого файла;

-t – перечисляет файлы и каталоги в соответствии с датой их последней модификации;

-u – перечисляет файлы и каталоги в порядке, обратном в соответствии с датой их последней модификации.

Если команда ls применяется без указания пути к каталогу, то по умолчанию на экран выводится содержимое текущего каталога.

Упражнение 4.

1. Выполните команду ls. На экран выведутся каталоги, содержащиеся в текущем (рисунок 5.11).

2. Выполните команду ls -l. На экран выведется список каталогов, содержащихся в текущем, с атрибутами (рисунок 5.12).

3. Выполните команду ls -la. На экран выведется список каталогов и файлов, содержащихся в текущем каталоге, с атрибутами (рисунок 5.13).

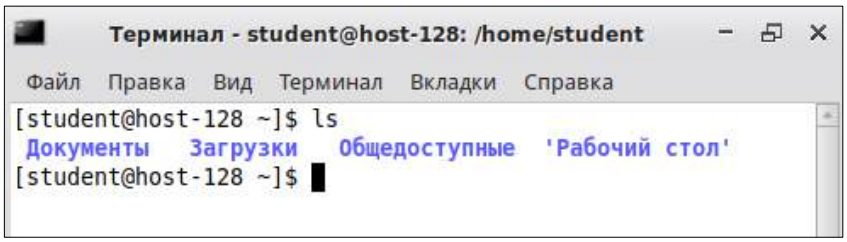


Рисунок 5.11 – Результат работы команды ls

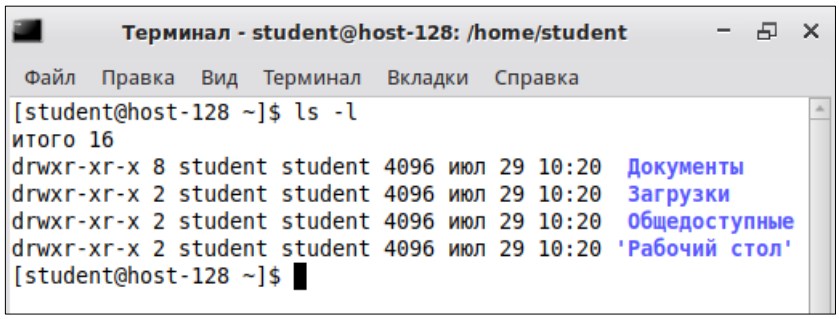


Рисунок 5.12 – Результат работы команды ls -l

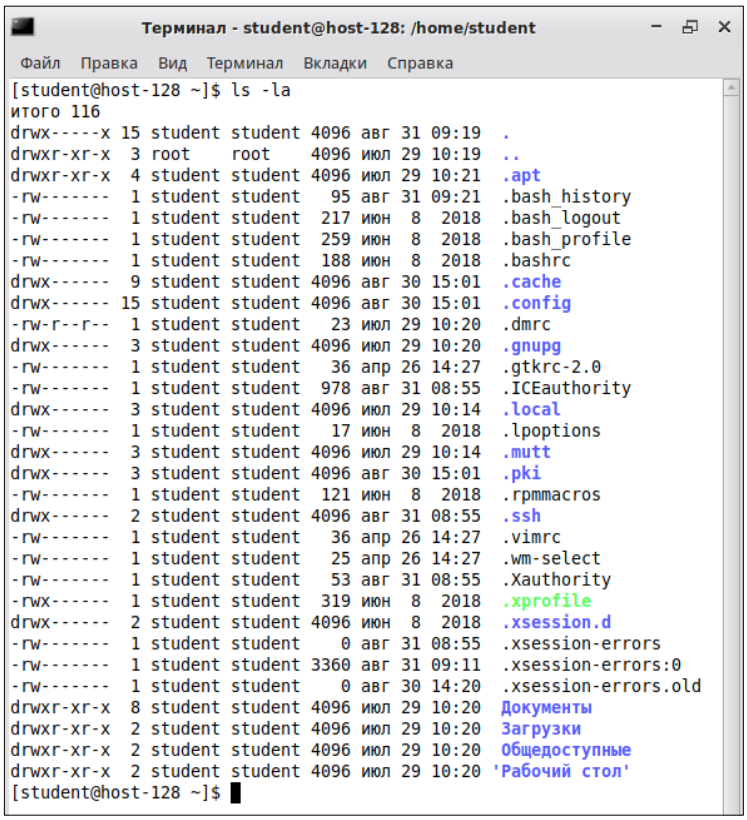
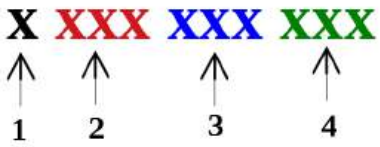


Рисунок 5.13 – Результат работы команды ls -la

Обратите внимание, что перед названием файла или каталога выведены атрибуты, указывающие на принадлежность файла (каталога), возможность чтения, записи и выполнения различными группами пользователей.

Атрибуты представляют собой цепочку из десяти символов **x xxx xxx xxx**:



1 – первый символ байта атрибутов определяет тип файла и может интерпретироваться со следующими значениями:

- – обычный файл;
d –каталог;
l – символическая связь.

2 – вторая группа символов атрибутов определяет доступ к каталогам и файлам для владельца.

3 – третья группа символов атрибутов определяет доступ к каталогам и файлам для членов группы.

4 – четвертая группа символов атрибутов определяет доступ к каталогам и файлам для остальных пользователей.

Для каждого уровня доступа существуют свои байты атрибутов, значение которых расшифровывается следующим образом:

r – разрешение на чтение;
w – разрешение на запись;
x – разрешение на выполнение;
- – отсутствие разрешения.

В домашнем каталоге пользователь имеет полный доступ к файлам (READ, WRITE, EXECUTE; r, w, x).

После байтов атрибутов на экран выводится следующая информация о файле:

- число связей файла;
- имя владельца файла;
- размер файла в байтах;
- дата создания файла (или модификации);
- время;
- имя файла.

Например, с использованием команды ls -la выведем содержимое каталога student и рассмотрим атрибуты папки **Документы**:

d rwx r-x r-x. 2 student 4096 Сен 10 14:19 Документы,

где d – каталог;

rwx – разрешение на чтение, запись и выполнение для владельца (student);

r-x – разрешение на чтение и выполнение для членов группы;

r-x – разрешение на чтение и выполнение для остальных пользователей.

Выполните команду clear для очистки содержимого терминала.

Команда mkdir

Назначение: создание нового каталога.

Формат: mkdir имя_каталога

Упражнение 5.

1. Выполните команду mkdir Work.
2. Откройте домашний каталог в **Обозревателе файлов** и убедитесь, что в домашнем каталоге появился каталог Work.
3. Создайте в каталоге Work два каталога Факультет и Группа. Для этого выполните команды mkdir Work/Факультет и mkdir Work/Группа.

Команда cd

Назначение: изменение текущего каталога.

Формат: cd [имя_каталога]

Упражнение 6.

1. Проверьте, какой каталог является текущим с помощью команды pwd.
2. Измените текущий каталог на Факультет. Для этого выполните команду cd Work/Факультет.
3. Убедитесь, что текущим стал каталог Факультет.
4. Перейдите в каталог Группа с помощью команды cd ~/Work/Группа.
5. Выполните команду cd -. Таким образом, мы вернулись в предыдущий каталог Факультет.
6. Перейдите в каталог Work – родительский каталог для каталога Факультет. Выполните команду cd .. Таким образом, мы вернулись на один уровень выше.
7. Перейдите в домашний каталог, выполнив команду cd. Команда cd без параметров возвращает оболочку в домашний каталог.

Команда cat

Назначение: просмотр содержимого файлов, создание текстовых файлов, объединение содержимого нескольких файлов.

Формат:
cat [ключи][путь/]имя файла1 [[путь/]имя файла2]... [[путь/]имя файлаN]

Рассмотрим основные ключи:
-b – нумеровать только непустые строки;
-E – показывать символ $ в конце каждой строки;
-n – нумеровать все строки;
-s – удалять пустые повторяющиеся строки;
-T – отображать табуляции в виде ^I.

Упражнение 7.

1. Проверьте, какой каталог является текущим с помощью команды pwd.
2. Измените текущий каталог на Work.
3. Создайте в текущем каталоге текстовый файл text1. Для этого выполните команду cat > text1. На экране курсор перейдет на новую строку и оболочка будет ожидать ввода текста. Переключите раскладку клавиатуры на русский язык (**[Alt+Shift]**) и введите текст по предложенному образцу, указывая свои личные данные, в конце каждой строки нажимайте **Enter**:

Иванов Иван Иванович
дата рождения 11 октября 2005 г.
студент группы ТА-11 Энергетического факультета Азово-Черноморского инженерного института
год поступления – 2022

4. После окончания ввода текста нажмите комбинацию клавиш **[Ctrl+d]**. Оболочка выведет в терминал приглашение к работе. Файл text1 создан.
5. Проверьте наличие файла text1 в каталоге Work.
6. Выведите в терминал содержимое созданного файла, выполнив команду cat text1.
7. Создайте в текущем каталоге файл text2. Содержимым файла является список фамилий студентов вашей группы.
8. Выполните объединение содержимого файлов text1 и text2 и перенаправьте полученный текст в новый файл text3. Для этого выполните команду cat text1 text2 > text3.
9. Просмотрите содержимое файла text3.
10. Просмотрите содержимое каталога Work.
11. Сделайте текущим домашний каталог.

Команда cp

Назначение: копирует файлы и каталоги.

Формат: cp [ключ1] ... [ключN] [путь/]источник [путь/]приёмник

источник – каталог, который мы хотим скопировать;

приёмник – каталог, в который мы хотим скопировать.

Ключи:

-f,-force – перезаписывать существующие файлы;

-i,-interactive – спрашивать, необходимо ли перезаписывать существующие файлы;

-n – не перезаписывать существующие файлы;

-r – копировать каталог рекурсивно, т. е. с учетом вложенных каталогов;

-u – скопировать файл, только если он был изменён;

-p – сохранять владельца, временные метки и флаги доступа при копировании;

-t – считать файл-приемник директорией и копировать файл-источник в эту директорию.

Упражнение 8.

1. Проверьте, какой каталог является текущим.
2. Сделайте текущим каталог Work.
3. Скопируйте каталог Группа в каталог Факультет. Для этого выполните команду cp -a Группа Факультет. Копирование происходит внутри каталога Work, который является текущим. Поэтому мы не указываем путь к каталогам.
4. В каталоге Факультет создайте каталог Группа2.
5. Убедитесь, что текущим является каталог Work. Если это не так, сделайте каталог Work текущим.
6. Скопируйте файл text2 в каталог Группа2. Для этого выполните команду cp text2 Факультет/Группа2.

Команда mv

Назначение: перемещение, переименование файла или каталога.

Формат: mv [ключ1] ... [ключN] [путь/]источник [путь/]приёмник

источник – каталог или файл, которые мы хотим переименовать или переместить;

приёмник – новое имя каталога или файла при переименовании, имя каталога, в который перемещаем каталог или файл.

Упражнение 9.

1. Проверьте, какой каталог является текущим.
2. Сделайте текущим каталог Work.

3. Переименуйте каталог Факультет в Энергетический_Факультет, выполнив команду mv Факультет Энергетический_Факультет.

4. В каталоге Work переименуйте файл text3 на АЭ-11, выполнив команду mv text3 АЭ-11.

5. Переместите файл АЭ-11 в каталог Энергетический_Факультет, выполнив команду mv АЭ-11 Энергетический_Факультет.

Команда rm

Назначение: удаление файла или каталога.
Формат: rm [ключ1] ... [ключN] каталог (или файл)

Ключи:

-r – происходит рекурсивное удаление каталога вместе с его содержимым;

-i – перед удалением каждого файла запрашивается подтверждение.

Упражнение 10.

1. Проверьте, какой каталог является текущим.
2. Сделайте текущим каталог Work.
3. Удалите файл text1, выполнив команду rm text1.
4. Сделайте текущим домашний каталог.
5. Удалите каталог Work вместе со всем содержимым.
6. Для выхода из Терминала выполните команду exit.

Упражнение 11. В этом упражнении вы научитесь правильно выключать систему.

1. Закройте все открытые программы и окна.
2. В Меню запуска приложений щелкните по кнопке Выйти 

, которая находится в правом верхнем углу меню. В появившемся диалоговом окне щелкните по кнопке Выключить.

4 СОДЕРЖАНИЕ ОТЧЕТА

Отчет должен содержать тему и цель работы, выполненные задания для самостоятельной работы (выпишите команды терминала) и ответы на контрольные вопросы.

5 УПРАЖНЕНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

1. С помощью команд терминала создайте в домашнем каталоге следующую структуру каталогов (рисунок 5.14).

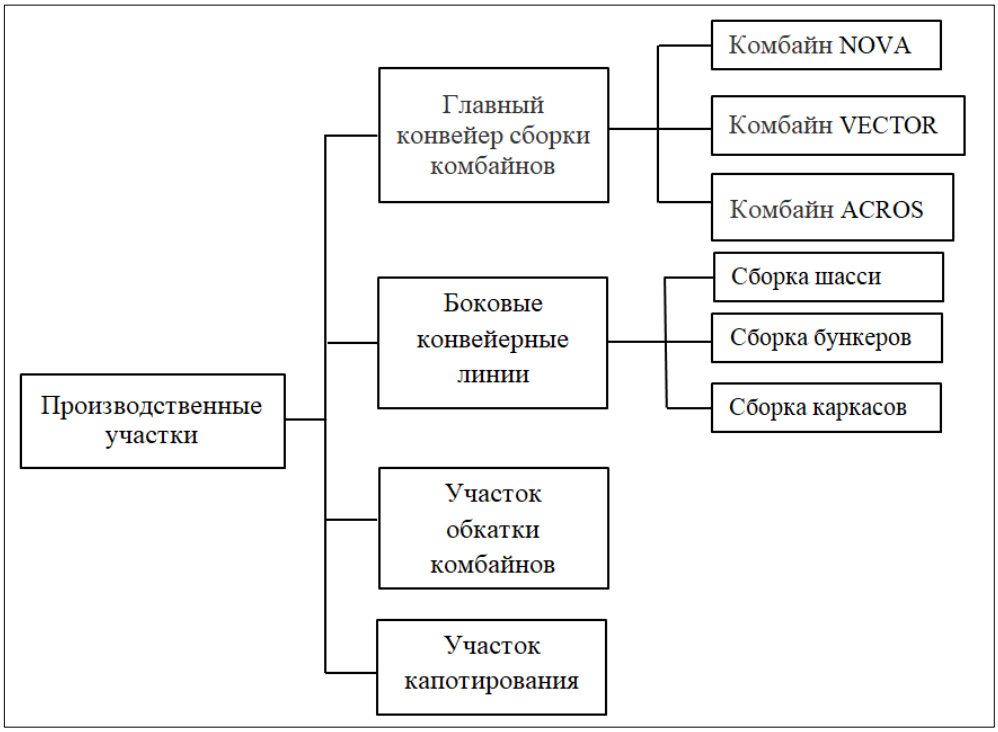


Рисунок 5.14 – Структура каталогов

2. В каталоге **Участок обкатки комбайнов** создайте текстовый файл **Отчет1**, в каталоге **Участок капотирования** создайте текстовый файл **Отчет2**.

3. Скопируйте файлы **Отчет1**, **Отчет2** в каталог **Главный конвейер сборки комбайнов** с именами **Отчет3** и **Отчет4** соответственно.

4. Перенесите файлы **Отчет3**, **Отчет4** в каталог **Производственные участки**.

5. Удалите файлы **Отчет1**, **Отчет2**.

6. Переименуйте файлы **Отчет3**, **Отчет4** каталога **Производственные участки** в файлы **Третий_Отчет**, **Четвертый_Отчет**.

7. Покажите результаты работы преподавателю.

8. Удалите каталог **Производственные участки**.

9. Создайте в режиме командной строки в домашнем каталоге следующую структуру каталогов (рисунок 5.15).

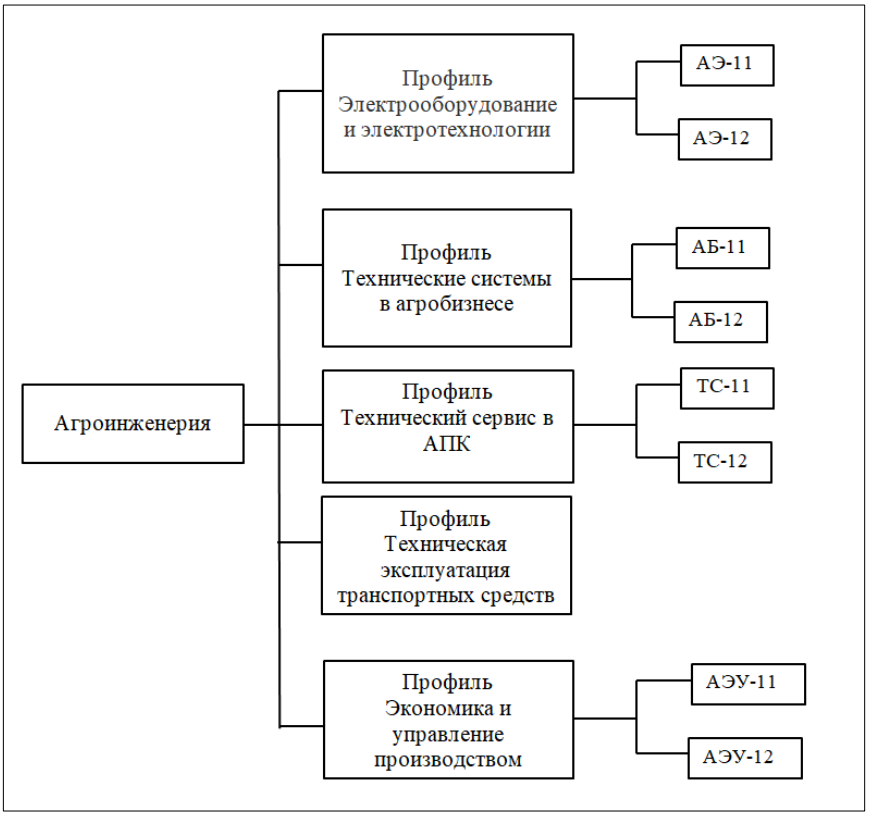


Рисунок 5.15 – Структура каталогов

10. В каталоге **Профиль Электрооборудование и электротехнологии** создайте текстовый файл **Сессия1**, в каталоге **Профиль Технический сервис в АПК** создайте текстовый файл **Сессия2**. Содержимым файлов будет являться перечень дисциплин с оценками.

11. Скопируйте файлы **Сессия1**, **Сессия2** в папку **Агроинженерия**.

12. Перенесите файлы **Сессия1, Сессия2** из папки **Агроинженерия** в папку **Профиль Экономика и управление производством**.

13. Переименуйте файлы **Сессия1**, **Сессия2** в папке **Профиль Экономика и управление производством** в файлы **Первая_сессия**, **Вторая_сессия**.

14. Покажите результаты работы преподавателю.

15. Удалите каталог **Агроинженерия**.

6 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Из каких компонентов состоит среда рабочего стола ОС Simply Linux?
2. Где находится **Меню запуска приложений**?
3. Для чего предназначен файловый менеджер **Thunar**?
4. Как открыть каталог в **Обозревателе файлов**?
5. Что представляет собой файловая система ОС Simply Linux?

6. Что такое путь к файлу?
7. Какой каталог в ОС Simply Linux является домашним? Как она обозначается?
8. Какой каталог называется текущим?
9. Что такое **Терминал**?
10. Для чего предназначен **Терминал** в ОС Simply Linux?
11. Что такое **Консоль**?
12. Что такое **Командная оболочка**?
13. Для чего предназначена **Командная строка**?
14. Назначение и формат команды **pwd**?
15. Назначение и формат команды **ls**?
16. Назначение и формат команды **mkdir**?
17. Назначение и формат команды **cd**?
18. Назначение и формат команды **cat**?
19. Назначение и формат команды **cp**?
20. Назначение и формат команды **mv**?
21. Назначение и формат команды **rm**?
22. Назначение и формат команды **exit**?
23. Назначение и формат команды **clear**?

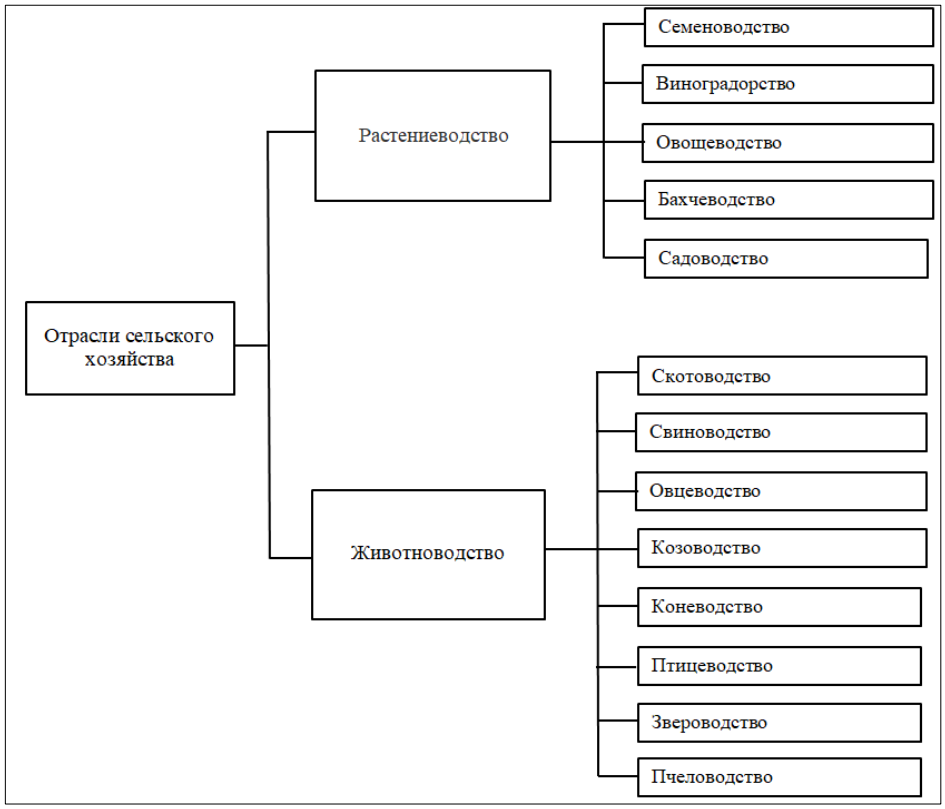
ЛАБОРАТОРНАЯ РАБОТА № 6

КОНТРОЛЬНАЯ РАБОТА № 2. «ИСПОЛЬЗОВАНИЕ ОПЕРАЦИОННОЙ СИСТЕМЫ LINUX ДЛЯ РЕШЕНИЯ ТИПОВЫХ ЗАДАЧ В ОБЛАСТИ АГРОИНЖЕНЕРИИ»

Примерный вариант

Все задания выполняются в режиме командной строки. Ответом на задание является соответствующая команда. Напишите команды, необходимые для выполнения каждого задания.

1. Текущим является корневой каталог. Сделайте текущим домашний каталог.
2. Создайте в домашнем каталоге следующую структуру каталогов:



3. В каталоге **Растениеводство** создайте текстовый файл с именем **oughtem**. Содержимым файла являются ответ на вопрос: **Какой каталог называется текущим?** Ответ на вопрос напишите на листе и укажите команду окончания ввода текста.
4. Скопируйте файл **oughtem** из каталога **Растениеводство** в каталог **Овощеводство**.
5. В каталоге **Овощеводство** переименуйте файл **oughtem** на **winter**.
6. Удалите файл **oughtem** из каталога **Растениеводство**.

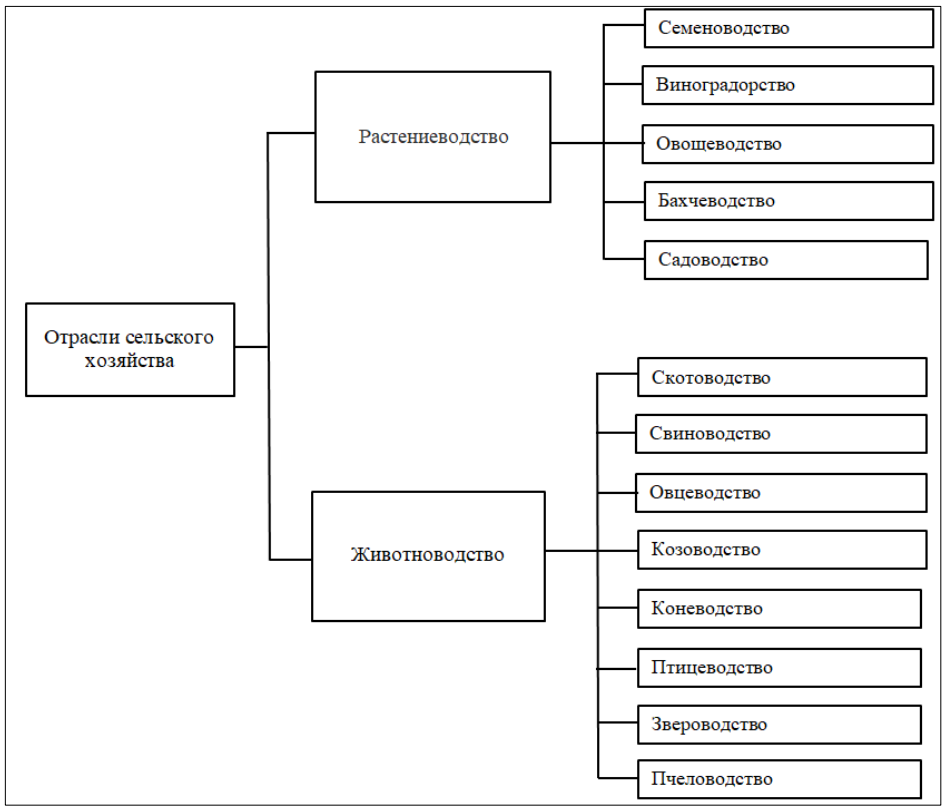
Результат выполнения контрольной работы № 2

1. Текущим является корневой каталог. Сделайте текущим домашний каталог.

Решение:

Сделать текущим домашний каталог можно тремя различными способами:
1) командой cd ~
2) командой cd home/student
3) командой cd

2. Создайте в домашнем каталоге следующую структуру каталогов:



Решение:

1) Создадим каталог Отрасли_сельского_хозяйства. Текущим является Домашний каталог, поэтому путь указывать не будем.
Команда:

mkdir Отрасли_сельского_хозяйства

2) Сделаем текущим созданный в п.1 каталог. Команда:

cd Отрасли_сельского_хозяйства

Создадим в этом каталоге два каталога: Растениеводство и Животноводство. Команды:

mkdir Растениеводство
mkdir Животноводство

3) Сделаем текущим каталог Растениеводство. Команда:

cd Растениеводство

Создадим в каталоге Растениеводство пять каталогов (см. задание). Команды:

mkdir Семеноводство
mkdir Виноградарство
mkdir Овощеводство
mkdir Бахчеводство
mkdir Садоводство

4) Сделаем текущим каталог Животноводство. Команда:

cd ../Животноводство

Создадим в каталоге Животноводство восемь каталогов (см. задание). Команды:

mkdir Скотоводство
mkdir Свиноводство
mkdir Овцеводство
mkdir Козловодство
mkdir Коневодство
mkdir Птицеводство
mkdir Звероводство
mkdir Пчеловодство

1. В каталоге **Растениеводство** создайте текстовый файл с именем **oughtem**. Содержимым файла являются ответ на вопрос: **Какой каталог называется текущим?** Ответ на вопрос напишите на листе и укажите команду окончания ввода текста.

Решение:

1) Сделаем текущим каталог Растениеводство. Команда:

cd ../Растениеводство

2) Создаём текстовый файл oughtem. Команда:

cat >oughtem

3) После нажатия клавиши [Enter] введём следующий текст:
Текущим называется каталог, с которым пользователь работает в данный момент времени.
4) Нажимаем комбинацию клавиш [Ctrl]+[D] для завершения создания файла oughtem.

2. Скопируйте файл **oughtem** из каталога **Растениеводство** в каталог **Овощеводство**.

Решение:

Команда:

cp oughtem Овощеводство

3. В каталоге **Овощеводство** переименуйте файл **oughtem** на **winter**.

Решение:

1) Сделаем текущим каталог Овощеводство. Команда:

cd Овощеводство

2) Переименуем файл oughtem на winter. Команда:

mv oughtem winter

4. Удалите файл **oughtem** из каталога **Растениеводство**.

Решение:

1) Сделаем текущим каталог Растениеводство. Команда:

cd ..

2) Удалим файл oughtem. Команда:

rm oughtem

РАЗДЕЛ 2

ИСПОЛЬЗОВАНИЕ ЯЗЫКА ПРОГРАММИРОВАНИЯ ВЫСОКОГО УРОВНЯ ДЛЯ РЕШЕНИЯ ТИПОВЫХ ЗАДАЧ В ОБЛАСТИ АГРОИНЖЕНЕРИИ

ЛАБОРАТОРНАЯ РАБОТА № 7

Тема: **ОСНОВНЫЕ ЯЗЫКОВЫЕ КОНСТРУКЦИИ ЯЗЫКА ПРОГРАММИРОВАНИЯ ВЫСОКОГО УРОВНЯ. ЛИНЕЙНЫЙ ВЫЧИСЛИТЕЛЬНЫЙ ПРОЦЕСС**

1 ЦЕЛЬ РАБОТЫ

Познакомиться:
- с основными языковыми конструкциями;
- с базовыми синтаксическими конструкциями языка Python.

Научиться:
- создавать программы, реализующие линейный вычислительный процесс.

2 ОБЪЁМ РАБОТЫ И ПОРЯДОК ЕЁ ВЫПОЛНЕНИЯ

Данная работа выполняется в течение двух часов. Выполнение работы идет одновременно с изучением теоретической части. Для закрепления материала необходимо выполнить упражнения для самостоятельной работы и оформить отчет.

3 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ. ТЕХНОЛОГИЯ ВЫПОЛНЕНИЯ РАБОТЫ

3.1 Основные понятия

Python – современный язык программирования, работающий на всех распространенных операционных системах для настольных компьютеров. В настоящее время активно используется две версии языка – более старая версия 2 и современная версия 3.

Достоинства языка программирования **Python**:

1. Кроссплатформенность и бесплатность.
2. Простой синтаксис и богатые возможности, позволяющие записывать программы очень кратко, но в то же время понятно.
3. По простоте освоения язык сравним с Бейсиком.
4. Богатая стандартная библиотека, возможность разработки промышленных приложений (для работы с сетью, GUI, базами данных и т. д.).

Программы на языке Python выполняются интерпретатором. В ОС Linux интерпретатор можно вызвать, набрав команду **python** в

терминале. После запуска интерпретатора появляется подсказка >>>, в которой можно начать отладку операторов программы в простом цикле чтения/выполнения (рисунок 7.1). Такой режим работы интерпретатора называют интерактивным.

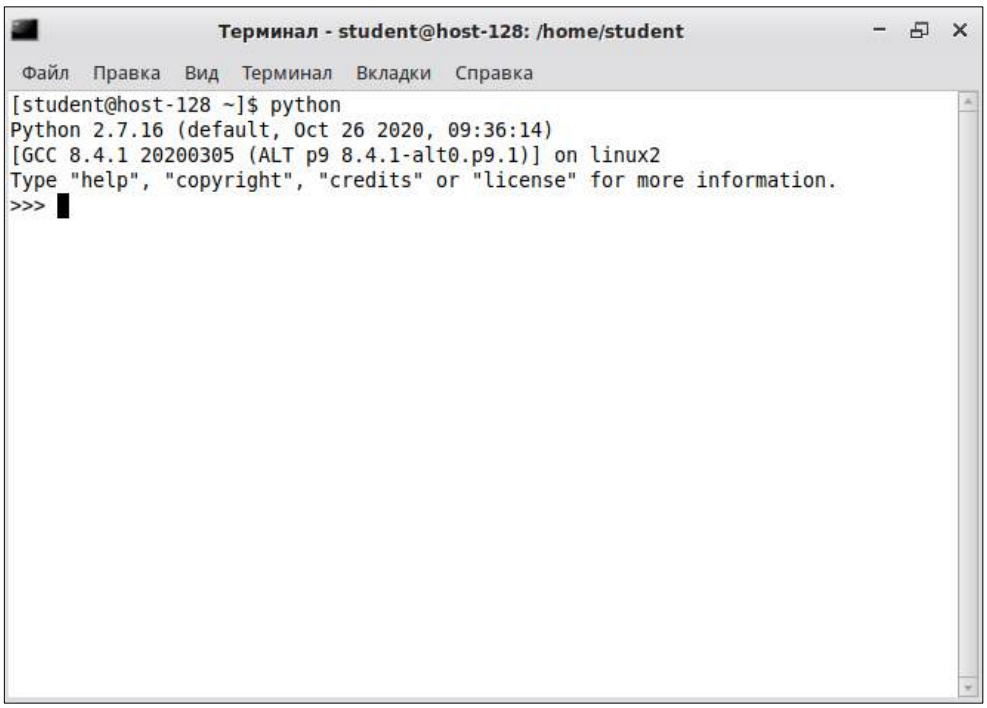


Рисунок 7.1 – Результат запуска Python в интерактивном режиме

Несмотря на удобства интерактивного режима работы с Python, часто требуется сохранить исходный программный код для последующего использования. В таком случае подготавливаются файлы, которые передаются затем интерпретатору на исполнение. По отношению к интерпретируемым языкам программирования часто исходный код называют скриптом. Файлы с кодом на Python обычно имеют расширение **.py**.

Скрипты можно писать в любом текстовом редакторе (например, стандартная программа medit с подсветкой синтаксиса Python). Также существуют специальные программы для разработки, предоставляющие дополнительные возможности и удобства.

Запускать подготовленные файлы можно в Терминале с помощью команды **python адрес/имя_файла**.

3.2 Переменные в Python

Совокупность величин, с которыми работает компьютер, принято называть **данными**. По отношению к программе, данные делятся на **исходные**, **результаты** (окончательные данные) и **промежуточные**, которые получаются в процессе вычислений.

Для успешного освоения программирования необходимо усвоить следующее правило: всякая величина занимает свое определенное место в памяти ЭВМ (иногда говорят – ячейку памяти). Хотя термин «ячейка», если

учесть архитектуру современных ЭВМ, несколько устарел, однако в учебных целях его удобно использовать.

У всякой величины имеются три основных свойства: **имя**, **значение** и **тип**. В алгоритмах и языках программирования величины делятся на **константы** и **переменные**.

Константа – величина, которая не изменяет свое значение в процессе выполнения программы. В программе она представляется собственным значением, например, 15, 34.7, ‘k’, true и пр.

Переменная – это простейшая именованная структура данных, в которой может быть сохранён промежуточный или конечный результат работы программы. Переменная может изменять свое значение в процессе выполнения программы и представляется символическим именем – **идентификатором**, например, X, S2, cod15 и пр.

Правила написания имен переменных (констант):

1. Переменная должна иметь уникальное имя, состоящее из букв латинского алфавита, цифр и знака подчеркивания.
2. Имя переменной должно начинаться с буквы.
3. В имени переменной нельзя использовать ключевые слова и встроенные команды (например, import, help…).
4. При написании имени переменной необходимо учитывать регистр букв: X и x разные переменные.

Переменную в Python создать очень просто – необходимо присвоить некоторому идентификатору значение при помощи оператора присваивания «=».

Упражнение 1. В этом упражнении вы научитесь работать с Python в режиме интерпретатора.

1. Откройте домашнюю папку. Откройте терминал, выполнив команду контекстного меню Открыть терминал в домашней папке.
2. Запустите интерпретатор Python. Для этого наберите в командной строке команду python и нажмите клавишу **[Enter]**.
3. После приглашения к вводу команд интерпретатора >>> введите следующие команды, нажимая клавишу **[Enter]** после каждой строки.

```
a = 10
b = 3.1415926
c = "Hello"
d = [1, 2, 3]
```

4. Просмотрите значения переменных a, b, c, d. Для этого в интерпретаторе введите имя переменной и нажмите **[Enter]**.

В этом упражнении используются четыре переменные:

– переменная a хранит значение типа int (целое число),

– переменная b хранит значение типа float (действительное число),

– переменная c хранит значение типа str (строка),

– переменная d хранит значение типа list (список, в данном случае из трех целых чисел).

Никакого специального объявления переменных не требуется, первое присваивание переменной значения и является ее объявлением.

Python – язык с динамической типизацией: каждая переменная в каждый момент времени имеет определенный тип, но этот тип может меняться по ходу выполнения программы, достаточно просто присвоить ей новое значение другого типа.

На самом деле переменная в Python является лишь ссылкой на объект в памяти. При создании любой переменной (число, строка или массив) в неё записывается ссылка на объект, а сам объект находится где-то в оперативной памяти далеко от самой переменной со ссылкой. Таким образом, несколько переменных могут указывать на один объект, и при изменении объекта (например, списка) изменится результат обращения к нему с использованием каждой переменной.

3.3 Структура простейшей программы в Python

Как правило, программа на Python должна состоять из следующих частей:

1. Считывание данных.
2. Решение задачи.
3. Вывод результата.

Например, программа, считающая сумму двух чисел может иметь следующий вид:

```
a = int(input())
b = int(input())
sum = a + b
print "sum= ", sum
```

В данной программе с клавиатуры вводятся значения переменных a и b , вычисляется их сумма sum, и выводится значение переменной sum на экран.

3.4 Синтаксис языка Python

Программа на Python – обычный текстовый файл с инструкциями. Каждая инструкция располагается на отдельной строке. Если инструкция не является вложенной, то она начинается с начала строки (отступ вложенной инструкции может быть, например, 4 пробела).

В конце инструкции обычно ничего не ставится. Но, если строка содержит несколько инструкций, то ставится точка с запятой (;). Длинную инструкцию в строке можно разбить на несколько частей с помощью \ или заключив инструкцию в круглые скобки.

Вложенные инструкции объединяются в блоки по величине отступов. Отступ может быть любым, главное, чтобы в пределах одного вложенного блока отступ был одинаков.

Вложенные инструкции в Python записываются в соответствии с одним и тем же шаблоном, когда основная инструкция завершается двоеточием, вслед за которым располагается вложенный блок кода, обычно с отступом под строкой основной инструкции.

3.5 Комментарии

Комментарии в программном коде предназначены для пояснений и игнорируются интерпретатором. Текст комментария всегда начинается со знака # (октоторп). В Python можно использовать только однострочный комментарий.

Упражнение 2. В этом упражнении вы научитесь писать комментарии.

1. Для корректного отображения символов русского алфавита необходимо после запуска интерпретатора ввести команду

```
#coding=utf-8
```

2. В режиме интерпретатора введите следующую команду:

```
print “Первая программа на Python”  # Вывод текста на экран
```

Обратите внимание, что в результате выполнения команды на экране появится текст, заключенный в кавычки в операторе print, а комментарий выводиться не будет.

3.6 Типы данных

В языке Python выделяют несколько типов данных: целые числа, числа с плавающей точкой (вещественные), строки, логический тип и т.д.

Тип каждой переменной может динамически изменяться по ходу выполнения программы. Определить, какой тип имеет переменная, можно с помощью команды type(имя переменной).

3.6.1 Тип Numbers (числа)

Python поддерживает четыре различных числовых типа: целые числа (int), длинные целые числа (long), числа с плавающей запятой или вещественные (float) и комплексные числа (complex).

Целые числа являются положительными или отрицательными целыми числами без десятичной точки. Длинные целые числа имеют неограниченную точность, действительные числа записываются с десятичной точкой, отделяющие целые и дробные части. Комплексные числа имеют вещественную и мнимую часть, a + bj, где a – вещественная часть, а b – мнимая часть.

Целые числа (int) и длинные целые числа (long)

Целое число в Python имеет тип int, длинное целое – тип long. Тип int представляет целое число, для хранения которого используется 4 байта в памяти компьютера. Числа данного типа имеют 32-битную точность (или допустимый диапазон от -2147483648 до +2147483647). Длинные целые числа ограничений не имеют.

Вещественные числа (float)

Вещественные числа или числа с плавающей запятой имеют тип float. Представляют числа с плавающей точкой, для хранения которого используется 8 байт, например, 1.2 или 34.76. Точность чисел с плавающей точкой равна 15 десятичным знакам в дробной части. Обратите внимание, что **в Python разделителем целой и дробной частей числа является точка.**

Комплексные числа (complex)

Тип данных complex относится к категории неизменяемых и хранит пару значений типа float, одно из которых представляет действительную часть комплексного числа, а другое – мнимую. Принцип описания комплексных чисел в Python похож на описание комплексных чисел в математике. Комплексные числа записываются в форме a+bj, где a – действительная часть числа, b – мнимая часть числа.

Упражнение 3. В этом упражнении вы научитесь выполнять арифметические операции с комплексными числами.

1. В режиме интерпретатора введите следующие команды:

```
f = 3.5 + 2j
print "f=", f
e=4j
print "e=", e
```

Если действительная часть равна 0, то эту часть при записи комплексного числа можно опустить.

Отдельные части комплексного числа доступны в виде атрибутов real и imag.

2. В режиме интерпретатора введите следующие команды:

```
m = -4.5-5j
print m.real, '\n', m.imag  #Вывод действительной и мнимой части числа m
```

'\n' – перевод курсора на новую строку.

В Python с комплексными числами можно выполнять следующие арифметические операции:

a + b – сумма;
a - b – разность;
a * b – произведение;
a / b – частное;
a * * b – возведение в степень.

Указанные арифметические операции можно выполнять как с самим комплексным числом, так и с его частями.

3. В режиме интерпретатора введите следующие команды:

```
n = e + f
print "n=", n
```

Основные операции с целыми и вещественными числами

a + b – сумма;
a - b – разность;
a * b – произведение;
a / b – частное (результатом этого действия является вещественное или целое число);
a % b – возвращает остаток от деления a на b;
a / / b – возвращает целую часть от деления a на b (целочисленное деление);
a * * b – возведение a в степень b;
pow(a, b) – возведение a в степень b;
-a – отрицание a;
divmod(a, b) – возвращает пару (a / / b, a % b), т. е. целую часть от деления и остаток от деления.

Упражнение 4. В этом упражнении вы научитесь выполнять арифметические операции с целыми и вещественными числами.

1. В режиме интерпретатора введите следующие команды:

```
a = 25
b = 4.3
c = a * b
print "c=", c  #  Вывод результата произведения двух чисел
d = a % b
print "d=", d  #  Вывод остатка от деления
```

Приоритеты выполнения операций

Приоритеты операций в Python совпадают с приоритетом операций в математике:

1. Возведение в степень справа налево, т.е. 3* *3* *3 это 3* *(3* *3).
2. Унарный минус (отрицание).
3. Умножение и деление слева направо. Операции умножения и деления имеют одинаковый приоритет.
4. Сложение и вычитание слева направо. Операции сложения и вычитания имеют одинаковый приоритет.
5. Для изменения порядка действий необходимо использовать скобки.

Арифметические операции с присвоением (унарные операции)

Ряд специальных операций позволяют присвоить результат операции первому операнду:

`+=` – присвоение результата сложения;
`-=` – присвоение результата вычитания;
`*=` – присвоение результата умножения;
`/=` – присвоение результата от деления;
`//=` – присвоение результата целочисленного деления;
`**=` – присвоение результата возведения в степень;
`%=` – присвоение остатка от деления.

2. В режиме интерпретатора введите следующие команды:

```
a = 5
a+=2
print “a=”, a
d = 23
d//=3
print “d=”, d
```

3.6.2 Тип String (строки)

Строки в Python имеют тип `str`. Строкой называется последовательность символов: букв, цифр, знаков препинания и т.д., представленных в кавычках. Python позволяет использовать пару одиночных или двойных кавычек. Подмножества строк могут быть взяты с использованием оператора «извлечения среза» или «доступа по индексу» (`[]` и `[:]`) с индексами, начинающимися с 0 в начале строки., с -1, если отсчет идет с конца строки.

Основные операции со строками

`a + b` – конкатенация (строка `b` приписывается к строке `a`);
`a * n` – повторение строки `a` `n` раз, значение `n` должно быть целого типа.

Упражнение 5. В этом упражнении вы научитесь выполнять операции со строками.

1. В режиме интерпретатора введите следующие команды:

```
stroka1 = “primer 1”
print stroka1  # Вывод всей строки
```

```
print stroka1[0]  # Вывод первого символа строки
print stroka1[2:5]  # Вывод символов строки с третьего по пятый
print stroka1[2:]  # Вывод строки с третьего символа
print stroka1*2  # Вывод строки дважды
r1 = "Python"
r2 = " - язык программирования высокого уровня"
r3 = r1 + r2  # Операция конкатенация
print r3
```

3.6.3 Тип List (списки)

Список представляет собой упорядоченную последовательность элементов. Он очень гибкий и является одним из самых используемых типов в Python. Элементы списка не обязательно должны быть одного типа.

Объявить список довольно просто. Внутрь квадратных скобок помещаются элементы списка, разделённые запятой:

```
a = [1, 2.4, "Python"]
```

Для извлечения элемента из списка используется оператор []. В Python индексация начинается с нуля.

```
a = [5, 20, 13, 15, 17, 18]
print "a[2]=", a[2]  # Вывод элемента списка с индексом 2
```

3.7 Преобразование типов в Python

Иногда бывает полезно целое число записать как строку. И, наоборот, если строка состоит из цифр, то полезно эту строку представить в виде числа, чтобы дальше можно было выполнять арифметические операции с ней. Для этого используются функции, название которых совпадает с именем типа, то есть int, float, str.

Существует несколько встроенных функций для преобразования из одного типа данных в другой. Эти функции возвращают новый объект, представляющий преобразованное значение (таблица 7.1).

Таблица 7.1 – Преобразование типов в Python

№	Функция	Описание
1	2	3
1	int(x [,base])	Преобразует x в целое число
2	float(x)	Преобразует x в число с плавающей запятой
3	complex(real [,imag])	Создает комплексное число
4	str(x)	Преобразует объект x в строковое представление
5	repr(x)	Преобразует объект x в строковое выражение

Продолжение таблицы 7.1

1	2	3
6	eval(str)	Обрабатывает строку и возвращает объект. Используется для выполнения программного кода представленного строкой
7	list(s)	Преобразует s в список
8	chr(x)	Преобразует целое число в символ
9	unichr(x)	Преобразует целое число в символ Unicode
10	ord(x)	Преобразует один символ в его целочисленное значение

3.8 Оператор присваивания

Оператором присваивания является символ =. Выполняется оператор стандартным образом: сначала вычисляется выражения справа от знака равенства, а затем полученное значение записывается в переменную, указанную слева от знака равенства.

```
a = 3.14
b = a + 10
```

3.9 Встроенные функции

В среде Python без дополнительных операций импорта доступно более сотни встроенных объектов, в основном, функций и исключений.

Для удобства функции условно разделены по категориям:

1. Функции преобразования типов и классы: **coerce**, **str**, **repr**, **int**, **list**, **tuple**, **long**, **float**, **complex**, **dict**, **super**, **file**, **bool**, **object**.
2. Числовые и строковые функции: **abs**, **divmod**, **ord**, **pow**, **len**, **chr**, **unichr**, **hex**, **oct**, **cmp**, **round**, **unicode**.
3. Функции обработки данных: **apply**, **map**, **filter**, **reduce**, **zip**, **range**, **xrange**, **max**, **min**, **iter**, **enumerate**, **sum**.
4. Функции определения свойств: **hash**, **id**, **callable**, **issubclass**, **isinstance**, **type**.
5. Функции для доступа к внутренним структурам: **locals**, **globals**, **vars**, **intern**, **dir**.
6. Функции компиляции и исполнения: **eval**, **execfile**, **reload**, **__import__**, **compile**.
7. Функции ввода-вывода: **input**, **raw_input**, **open**.
8. Функции для работы с атрибутами: **getattr**, **setattr**, **delattr**, **hasattr**.
9. Функции-"украшатели" методов классов: **staticmethod**, **classmethod**, **property**.
10. Прочие функции: **buffer**, **slice**.

3.10 Функция для ввода данных

Язык Python включает много уже определенных, встроенных в него, функций. Программист не видит их определений. Достаточно знать, что эти функции принимают и что возвращают, то есть их синтаксис.

Ввод данных осуществляется при помощи функции input().
Синтаксис:
input([строка])

Когда вызывается эта функция, программа останавливает свое выполнение и ждет, когда пользователь введет текст. После этого, когда он нажмет [**Enter]**, функция input() заберет введенный текст и передаст его программе, которая уже будет обрабатывать его согласно своим алгоритмам.

Если в интерактивном режиме ввести команду input(), то компьютер будет ждать, когда вы что-нибудь введете и нажмете [**Enter]** или просто нажмете [**Enter]**. Введенное значение сразу же отобразиться на экране.

Упражнение 6. В этом упражнении вы научитесь использовать функцию input().

Функцию input() можно вызвать двумя способами.

Способ 1. Функция input() передает введенные данные в программу, которые можно присвоить переменной. В этом случае интерпретатор не выводит строку сразу же.

Способ 2. Функция input() вводит входные значения как строку символов. Поэтому, чтобы присвоить переменной целочисленное значение, следует воспользоваться функцией преобразования типов, например int().

1. В режиме интерпретатора введите следующие команды. Ввод каждой команды заканчивается нажатием клавиши [Enter].

```
input()  #Ввод в интерактивном режиме
a = input("Введите значение переменной a= ")  #Ввод значения переменной a
b = int(input())  #Ввод значения целочисленной переменной b
```

Обратите внимание, что в каждом из трёх случаев применения функции ввода данных input(), интерпретатор ожидает ввода данных с клавиатуры!

3.11 Оператор вывода данных

Для вывода данных используется оператор print, который может выводить не только значения переменных, но и значения любых выражений
Синтаксис:

```
print [object, ...][, sep=' '][, end='\n'][, file=sys.stdout]
```

где object – объекты, которые требуется отправить в файл;

sep=' ' – строка, которой следует разделить объекты, None – использовать значение по умолчанию;

end='\n' – строка, которой следует поставить после всех объектов, None – использовать значение по умолчанию;

file=sys.stdout – ожидается объект, реализующий метод write(string).

При помощи оператора print можно выводить значение не одного, а нескольких выражений, для этого необходимо перечислить их через запятую.

Упражнение 7. В этом упражнении вы научитесь использовать оператор print.

1. В режиме интерпретатора введите следующие команды:

```
a = 1
b = 2
print "a=", a, '\n', "b=", b, '\n', "a+b=", a + b
```

В данном случае будет напечатан текст: a=1, b=2, a+b=3. Сначала выводится значение переменной a, затем значение переменной b, затем значение суммы a + b.

3.12 Основные стандартные модули Python

Одним из важных преимуществ языка Python является наличие большой библиотеки модулей и пакетов, входящих в стандартную комплектацию.

В соответствии с модульным подходом к программированию большая задача разбивается на несколько более мелких, каждую из которых (в идеале) решает отдельный модуль. При построении модульной структуры программы важнее составить такую композицию модулей, которая позволила бы свести к минимуму связи между ними. Набор классов и функций, имеющий множество связей между своими элементами, было бы логично расположить в одном модуле.

Модуль должен иметь удобный интерфейс: набор функций, классов и констант, который он предлагает своим пользователям.

Модули выполняют как минимум три важных функции:

1. Повторное использование кода: такой код может быть загружен много раз во многих местах.

2. Управление адресным пространством: модуль – это высокоуровневая организация программ, это пакет имен, который избавляет вас от конфликтов. Каждый объект «проживает» свой цикл внутри своего модуля, поэтому модуль – это средство для группировки системных компонентов.

3. Глобализация сервисов и данных: для реализации объекта, который используется во многих местах, достаточно написать один модуль, который будет импортирован.

В программе на Python ниже представлен модуль math, атрибутами которого являются имена, определенные в нём:

```
from math import *
c = cos(0)
print "c=", c
```

В данном примере импортируется модуль math. В результате работы оператора import в текущем пространстве имен появляется объект с именем math.

Модули для использования в программах на языке Python по своему происхождению делятся на обычные (написанные на Python) и **модули расширения**, написанные на другом языке программирования (как правило, на C). С точки зрения пользователя они могут отличаться быстродействием.

Python распространяется с библиотекой стандартных модулей, включающей в себя более 200 модулей, которые выполняют платформенно-зависимую поддержку таких задач, как: интерфейс к операционной системе, управление объектами, поиск, сеть + интернет, GUI и т.д.

3.12.1 Модули math и cmath

В этих модулях собраны математические функции для действительных и комплексных аргументов. В таблице 7.2 даны функции модуля math.

Там, где аргумент обозначен буквой z, аналогичная функция определена и в модуле cmath.

Таблица 7.2 – Основные функции модуля math для работы с математическими функциями

Функция или константа	Описание функции	Математическая запись
1	2	
cos(z)	косинус аргумента z	$\cos(z)$
sin(z)	синус аргумента z	$\sin(z)$
tan(z)	тангенс аргумента z	$\mathrm{tg}(z)$
acos(z)	арккосинус аргумента z	$\arccos(z)$
asin(z)	арксинус аргумента z	$\arcsin(z)$
atan(z)	арктангенс аргумента z	$\mathrm{arctg}(z)$
pi	константа пи	π
exp(z)	экспонента аргумента z	e^z
pow(x,y)	возведение числа x в степень y	x^y

Продолжение таблицы 7.2

1	2	3
sqrt(z)	корень квадратный от z	$\sqrt{z}$
abs(z)	модуль z	$\|z\|$
floor(z)	наибольшее целое, меньшее или равное z	–
fmod(x,y)	остаток от деления x на y	–
hypot(x,y)	корень квадратный из суммы квадратов x и y	$\sqrt{x^2+y^2}$
log(z)	натуральный логарифм аргумента z	$ln(z)$
log10(z)	десятичный логарифм аргумента z	$lg(z)$
log(a)/log(b)	логарифм по основанию b аргумента a	$log_b(a)$
modf(z)	возвращает пару (y,q) — дробную и целую части числа z; обе части имеют знак исходного числа	–
ceil(z)	наименьшее целое, большее или равное z	–

Рассмотрим пример записи математической функции $y = ln(x) - \frac{x^3}{|x+1|}$ с использованием модуля math. На языке Python эта функция будет выглядеть так:

```
y = log(x) — x**3 / abs(x + 1)
```

3.13 Линейный вычислительный процесс

Линейным называется процесс обработки данных, если составляющие его действия выполняются последовательно одно за другим.

Упражнение 8. В этом упражнении вы составите блок-схему алгоритма и разработаете программу на языке Python для вычисления значения функции $y = a + \frac{sin(10*b)}{e^{c+2}}$, где a, b, c – произвольные числа, введенные с клавиатуры.

1. Составьте блок-схему алгоритма (см.рисунок 7.2).
2. Выйдите из интерактивного режима Python, нажав комбинацию клавиш **[Ctrl+d]**.
3. В домашней папке создайте файл **Lab7_1.py**

4. Откройте созданный файл в программе **medit** и наберите текст программы (см. рисунок 7.3).

5. Сохраните файл.

6. Запустите программу на выполнение. Для этого в Терминале введите команду:

```
python Lab7_1.py
```

7. После выполнения программы интерпретатор Python будет ожидать ввода значений указанных переменных. После ввода каждого значения нажимайте **[Enter]**. На экране появится результат работы вашей программы.

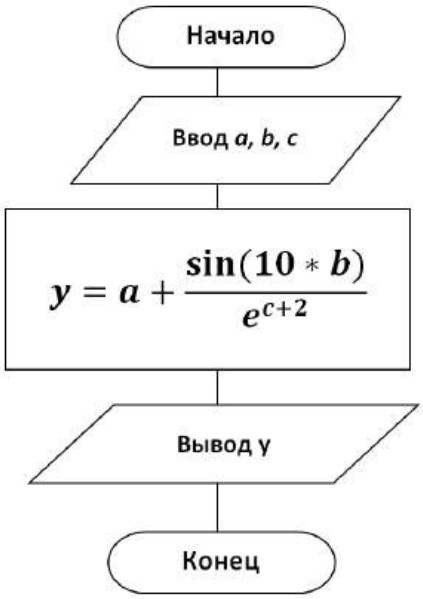


Рисунок 7.2 – Блок-схема алгоритма

```
#coding=utf-8
from math import *                  #Подключение модуля math
a=float(input('Введите a= '))       #Ввод значения вещественной переменной a
b=float(input('Введите b= '))       #Ввод значения вещественной переменной b
c=float(input('Введите c= '))       #Ввод значения вещественной переменной c
y=a+sin(10*b)/exp(c+2)              #Вычисление y
print 'y= ', y                     #Вывод y
```

Рисунок 7.3 – Текст программы

4 СОДЕРЖАНИЕ ОТЧЕТА

Отчет должен содержать:

- тему лабораторной работы;
- цель лабораторной работы;
- номер варианта, указанный преподавателем;
- тексты задач соответствующего варианта;

- блок-схемы алгоритмов;
- тексты программ **Lab7_2.py** и **Lab7_3.py**;
- исходные данные;
- результаты выполнения программ;
- ответы на контрольные вопросы.

5 УПРАЖНЕНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

1. Составьте блок-схему алгоритма и напишите программу для **Задачи 1**, согласно варианту, указанному преподавателем. Имя файла – **Lab7_2.py**

Задача 1. Разработать программу для расчета параметров электрической схемы N, приведенной в таблице исходных данных (N – номер варианта, заданный преподавателем). Значения ЭДС (E1) и сопротивлений (R1 – R4) должны вводиться пользователем. В результате расчета должны быть выведены следующие значения:

а) эквивалентное сопротивление электрической цепи, Ом;

б) токи в ветвях электрической цепи, А;

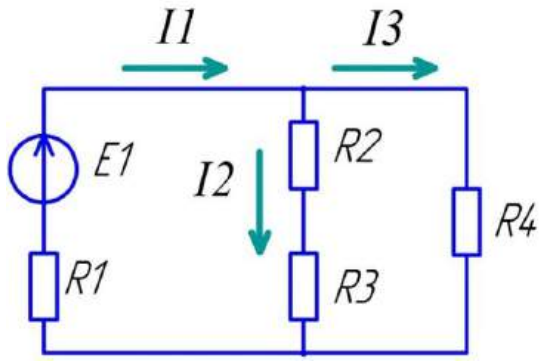
в) мощности, потребляемые активными сопротивлениями, Вт;

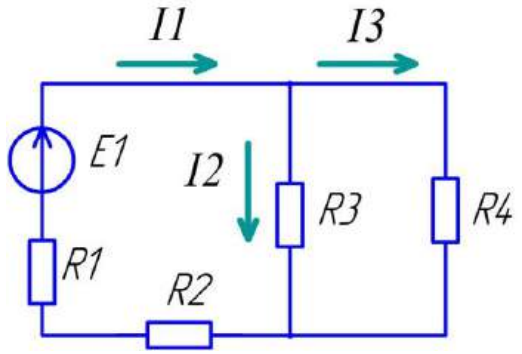
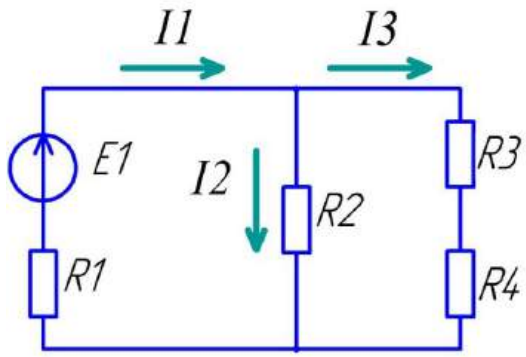
г) суммарная мощность, потребляемая нагрузкой, Вт;

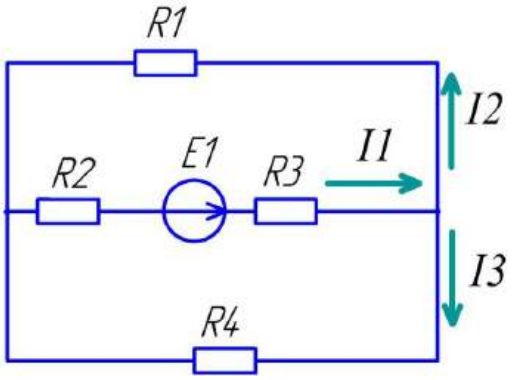
д) мощность, генерируемая источником питания, Вт.

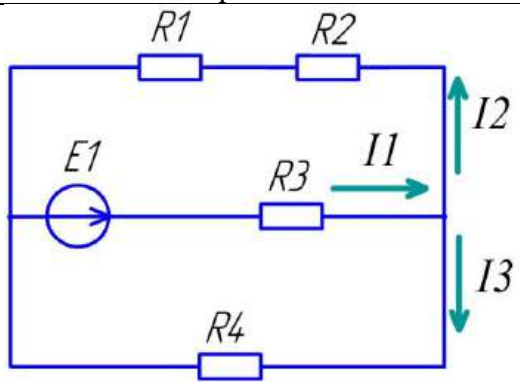
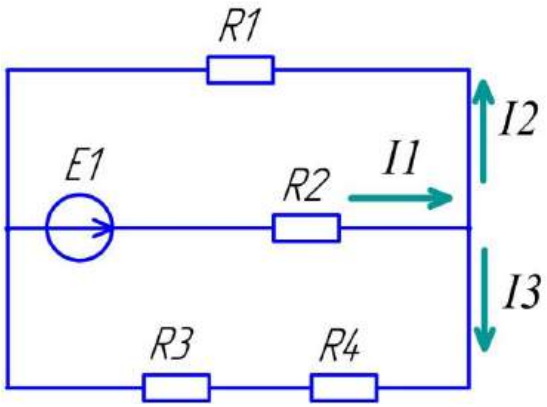
Все числовые данные, выводимые на консоль, округлять до сотых и приводить поясняющие надписи.

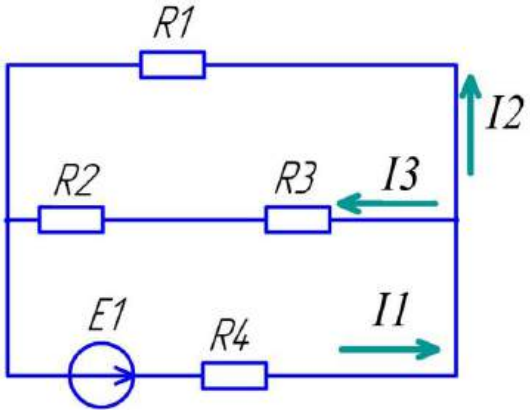
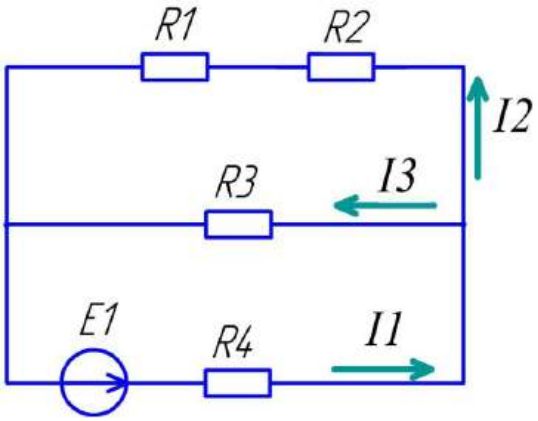
Таблица исходных данных

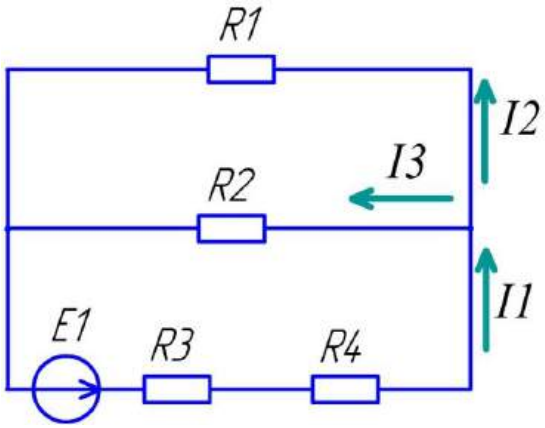
N	Электрическая схема	Порядок расчета
1		1) Эквивалентное сопротивление цепи, Ом $Rekv = R1 + \frac{(R2+R3)R4}{R2+R3+R4}$ 2) Ток в первой ветви, А $I1 = \frac{E1}{Rekv}$ 3) Ток во второй ветви, А $I2 = \frac{E1 - I1 \cdot R1}{R2+R3}$ 4) Ток в третьей ветви, А $I3 = \frac{E1 - I1 \cdot R1}{R4}$ 5) Мощность, потребляемая сопротивлением *R1*, Вт $P1 = I1^2 R1$ 6) Мощность, потребляемая сопротивлением *R2*, Вт $P2 = I2^2 R2$ 7) Мощность, потребляемая сопротивлением *R3*, Вт $P3 = I2^2 R3$ 8) Мощность, потребляемая сопротивлением *R4*, Вт

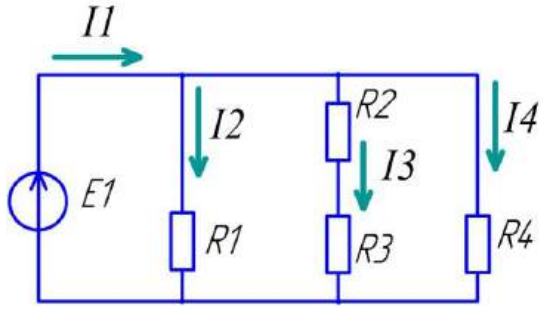
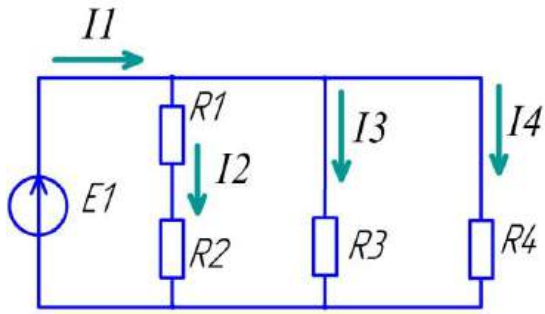
N	Электрическая схема	Порядок расчета
		$P4 = I3^2 R4$ 9) Суммарная мощность, потребляемая нагрузкой, Вт $Pnagr = P1 + P2 + P3 + P4$ 10) Мощность, генерируемая источником питания, Вт $Pist = E1 \cdot I1$
2		1) Эквивалентное сопротивление цепи, Ом $Rekv = R1 + R2 + \frac{R3 \cdot R4}{R3 + R4}$ 2) Ток в первой ветви, А $I1 = \frac{E1}{Rekv}$ 3) Ток во второй ветви, А $I2 = \frac{E1 - I1 \cdot (R1 + R2)}{R3}$ 4) Ток в третьей ветви, А $I3 = \frac{E1 - I1 \cdot (R1 + R2)}{R4}$ 5) Мощность, потребляемая сопротивлением *R1*, Вт $P1 = I1^2 R1$ 6) Мощность, потребляемая сопротивлением *R2*, Вт $P2 = I1^2 R2$ 7) Мощность, потребляемая сопротивлением *R3*, Вт $P3 = I2^2 R3$ 8) Мощность, потребляемая сопротивлением *R4*, Вт $P4 = I3^2 R4$ 9) Суммарная мощность, потребляемая нагрузкой, Вт $Pnagr = P1 + P2 + P3 + P4$ 10) Мощность, генерируемая источником питания, Вт $Pist = E1 \cdot I1$
3		1) Эквивалентное сопротивление цепи, Ом $Rekv = R1 + \frac{R2(R3 + R4)}{R2 + R3 + R4}$ 2) Ток в первой ветви, А $I1 = \frac{E1}{Rekv}$ 3) Ток во второй ветви, А $I2 = \frac{E1 - I1 \cdot R1}{R2}$ 4) Ток в третьей ветви, А $I3 = \frac{E1 - I1 \cdot R1}{(R3 + R4)}$ 5) Мощность, потребляемая сопротивлением *R1*, Вт $P1 = I1^2 R1$

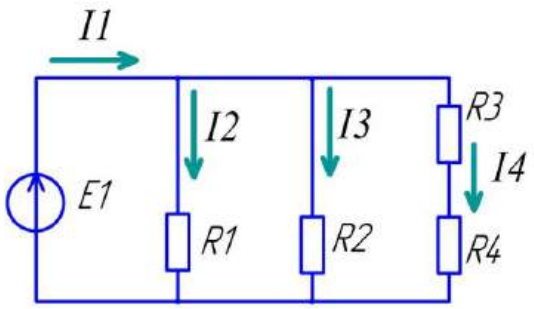
N	Электрическая схема	Порядок расчета
		6) Мощность, потребляемая сопротивлением *R2*, Вт $$P2 = I2^2 R2$$ 7) Мощность, потребляемая сопротивлением *R3*, Вт $$P3 = I3^2 R3$$ 8) Мощность, потребляемая сопротивлением *R4*, Вт $$P4 = I3^2 R4$$ 9) Суммарная мощность, потребляемая нагрузкой, Вт $$Pnagr = P1 + P2 + P3 + P4$$ 10) Мощность, генерируемая источником питания, Вт $$Pist = E1 \cdot I1$$
4		1) Эквивалентное сопротивление цепи, Ом $$Rekv = R2 + R3 + \frac{R1R4}{R1 + R4}$$ 2) Ток в первой ветви, А $$I1 = \frac{E1}{Rekv}$$ 3) Ток во второй ветви, А $$I2 = \frac{E1 - I1 \cdot (R2 + R3)}{R1}$$ 4) Ток в третьей ветви, А $$I3 = \frac{E1 - I1 \cdot (R2 + R3)}{R4}$$ 5) Мощность, потребляемая сопротивлением *R1*, Вт $$P1 = I2^2 R1$$ 6) Мощность, потребляемая сопротивлением *R2*, Вт $$P2 = I1^2 R2$$ 7) Мощность, потребляемая сопротивлением *R3*, Вт $$P3 = I1^2 R3$$ 8) Мощность, потребляемая сопротивлением *R4*, Вт $$P4 = I3^2 R4$$ 9) Суммарная мощность, потребляемая нагрузкой, Вт $$Pnagr = P1 + P2 + P3 + P4$$ 10) Мощность, генерируемая источником питания, Вт $$Pist = E1 \cdot I1$$
5		1) Эквивалентное сопротивление цепи, Ом $$Rekv = R3 + \frac{(R1 + R2)R4}{R1 + R2 + R4}$$ 2) Ток в первой ветви, А $$I1 = \frac{E1}{Rekv}$$ 3) Ток во второй ветви, А

N	Электрическая схема	Порядок расчета
		$I2 = \frac{E1 - I1 \cdot R3}{R1 + R2}$ 4) Ток в третьей ветви, А $I3 = \frac{E1 - I1 \cdot R3}{R4}$ 5) Мощность, потребляемая сопротивлением *R1*, Вт $P1 = I2^2 R1$ 6) Мощность, потребляемая сопротивлением *R2*, Вт $P2 = I2^2 R2$ 7) Мощность, потребляемая сопротивлением *R3*, Вт $P3 = I1^2 R3$ 8) Мощность, потребляемая сопротивлением *R4*, Вт $P4 = I3^2 R4$ 9) Суммарная мощность, потребляемая нагрузкой, Вт $Pnagr = P1 + P2 + P3 + P4$ 10) Мощность, генерируемая источником питания, Вт $Pist = E1 \cdot I1$
6		1) Эквивалентное сопротивление цепи, Ом $Rekv = R2 + \frac{R1(R3 + R4)}{R1 + R3 + R4}$ 2) Ток в первой ветви, А $I1 = \frac{E1}{Rekv}$ 3) Ток во второй ветви, А $I2 = \frac{E1 - I1 \cdot R2}{R1}$ 4) Ток в третьей ветви, А $I3 = \frac{E1 - I1 \cdot R2}{R3 + R4}$ 5) Мощность, потребляемая сопротивлением *R1*, Вт $P1 = I2^2 R1$ 6) Мощность, потребляемая сопротивлением *R2*, Вт $P2 = I1^2 R2$ 7) Мощность, потребляемая сопротивлением *R3*, Вт $P3 = I3^2 R3$ 8) Мощность, потребляемая сопротивлением *R4*, Вт $P4 = I3^2 R4$

N	Электрическая схема	Порядок расчета
		9) Суммарная мощность, потребляемая нагрузкой, Вт $$Pnagr = P1 + P2 + P3 + P4$$ 10) Мощность, генерируемая источником питания, Вт $$Pist = E1 \cdot I1$$
7		1) Эквивалентное сопротивление цепи, Ом $$Rekv = R4 + \frac{R1(R2 + R3)}{R1 + R2 + R3}$$ 2) Ток в первой ветви, А $$I1 = \frac{E1}{Rekv}$$ 3) Ток во второй ветви, А $$I2 = \frac{E1 - I1 \cdot R4}{R1}$$ 4) Ток в третьей ветви, А $$I3 = \frac{E1 - I1 \cdot R4}{R2 + R3}$$ 5) Мощность, потребляемая сопротивлением *R1*, Вт $$P1 = I2^2 R1$$ 6) Мощность, потребляемая сопротивлением *R2*, Вт $$P2 = I3^2 R2$$ 7) Мощность, потребляемая сопротивлением *R3*, Вт $$P3 = I3^2 R3$$ 8) Мощность, потребляемая сопротивлением *R4*, Вт $$P4 = I1^2 R4$$ 9) Суммарная мощность, потребляемая нагрузкой, Вт $$Pnagr = P1 + P2 + P3 + P4$$ 10) Мощность, генерируемая источником питания, Вт $$Pist = E1 \cdot I1$$
8		1) Эквивалентное сопротивление цепи, Ом $$Rekv = R4 + \frac{(R1 + R2)R3}{R1 + R2 + R3}$$ 2) Ток в первой ветви, А $$I1 = \frac{E1}{Rekv}$$ 3) Ток во второй ветви, А $$I2 = \frac{E1 - I1 \cdot R4}{R1 + R2}$$ 4) Ток в третьей ветви, А $$I3 = \frac{E1 - I1 \cdot R4}{R3}$$ 5) Мощность, потребляемая сопротивлением *R1*, Вт $$P1 = I2^2 R1$$ 6) Мощность, потребляемая сопротивлением

N	Электрическая схема	Порядок расчета
		R2, Вт $P2 = I2^2 R2$ 7) Мощность, потребляемая сопротивлением *R3*, Вт $P3 = I3^2 R3$ 8) Мощность, потребляемая сопротивлением *R4*, Вт $P4 = I1^2 R4$ 9) Суммарная мощность, потребляемая нагрузкой, Вт $Pnagr = P1 + P2 + P3 + P4$ 10) Мощность, генерируемая источником питания, Вт $Pist = E1 \cdot I1$
9		1) Эквивалентное сопротивление цепи, Ом $Rekv = R3 + R4 + \frac{R1 \cdot R2}{R1 + R2}$ 2) Ток в первой ветви, А $I1 = \frac{E1}{Rekv}$ 3) Ток во второй ветви, А $I2 = \frac{E1 - I1(R3 + R4)}{R1}$ 4) Ток в третьей ветви, А $I3 = \frac{E1 - I1(R3 + R4)}{R2}$ 5) Мощность, потребляемая сопротивлением *R1*, Вт $P1 = I2^2 R1$ 6) Мощность, потребляемая сопротивлением *R2*, Вт $P2 = I3^2 R2$ 7) Мощность, потребляемая сопротивлением *R3*, Вт $P3 = I1^2 R3$ 8) Мощность, потребляемая сопротивлением *R4*, Вт $P4 = I1^2 R4$ 9) Суммарная мощность, потребляемая нагрузкой, Вт $Pnagr = P1 + P2 + P3 + P4$ 10) Мощность, генерируемая источником питания, Вт $Pist = E1 \cdot I1$
10		1) Эквивалентное сопротивление цепи, Ом $Rekv = \frac{1}{\frac{1}{R1} + \frac{1}{R2 + R3} + \frac{1}{R4}}$ 2) Ток в первой ветви, А $I1 = \frac{E1}{Rekv}$

N	Электрическая схема	Порядок расчета
		3) Ток во второй ветви, А $I2 = \frac{E1}{R1}$ 4) Ток в третьей ветви, А $I3 = \frac{E1}{R2 + R3}$ 5) Ток в четвертой ветви, А $I4 = \frac{E1}{R4}$ 6) Мощность, потребляемая сопротивлением *R1*, Вт $P1 = I2^2 R1$ 7) Мощность, потребляемая сопротивлением *R2*, Вт $P2 = I3^2 R2$ 8) Мощность, потребляемая сопротивлением *R3*, Вт $P3 = I3^2 R3$ 9) Мощность, потребляемая сопротивлением *R4*, Вт $P4 = I4^2 R4$ 10) Суммарная мощность, потребляемая нагрузкой, Вт $Pnagr = P1 + P2 + P3 + P4$ 11) Мощность, генерируемая источником питания, Вт $Pist = E1 \cdot I1$
11		1) Эквивалентное сопротивление цепи, Ом $Rekv = \frac{1}{\frac{1}{R1 + R2} + \frac{1}{R3} + \frac{1}{R4}}$ 2) Ток в первой ветви, А $I1 = \frac{E1}{Rekv}$ 3) Ток во второй ветви, А $I2 = \frac{E1}{R1 + R2}$ 4) Ток в третьей ветви, А $I3 = \frac{E1}{R3}$ 5) Ток в четвертой ветви, А $I4 = \frac{E1}{R4}$ 6) Мощность, потребляемая сопротивлением *R1*, Вт $P1 = I2^2 R1$ 7) Мощность, потребляемая сопротивлением *R2*, Вт $P2 = I2^2 R2$ 8) Мощность, потребляемая сопротивлением *R3*, Вт $P3 = I3^2 R3$ 9) Мощность, потребляемая сопротивлением *R4*, Вт

N	Электрическая схема	Порядок расчета
		$P4 = I4^2 R4$ 10) Суммарная мощность, потребляемая нагрузкой, Вт $Pnagr = P1 + P2 + P3 + P4$ 11) Мощность, генерируемая источником питания, Вт $Pist = E1 \cdot I1$
12		1) Эквивалентное сопротивление цепи, Ом $Rekv = \frac{1}{\frac{1}{R1} + \frac{1}{R2} + \frac{1}{R3 + R4}}$ 2) Ток в первой ветви, А $I1 = \frac{E1}{Rekv}$ 3) Ток во второй ветви, А $I2 = \frac{E1}{R1}$ 4) Ток в третьей ветви, А $I3 = \frac{E1}{R2}$ 5) Ток в четвертой ветви, А $I4 = \frac{E1}{R3 + R4}$ 6) Мощность, потребляемая сопротивлением *R1*, Вт $P1 = I2^2 R1$ 7) Мощность, потребляемая сопротивлением *R2*, Вт $P2 = I3^2 R2$ 8) Мощность, потребляемая сопротивлением *R3*, Вт $P3 = I4^2 R3$ 9) Мощность, потребляемая сопротивлением *R4*, Вт $P4 = I4^2 R4$ 10) Суммарная мощность, потребляемая нагрузкой, Вт $Pnagr = P1 + P2 + P3 + P4$ 11) Мощность, генерируемая источником питания, Вт $Pist = E1 \cdot I1$

2. Составьте блок-схему алгоритма и напишите программу для **Задачи 2**, согласно варианту, указанному преподавателем. Имя файла – **Lab7_3.py**

Задача 2.

1. Вычислите значение функции:

$$Y = \frac{(x^2 - 12{,}5)^3}{x + b} + \sqrt[4]{|b - x^2|}, \quad b = 3{,}27, \qquad x - \text{произвольное.}$$

2. Вычислите значение функции:

$$Z = cos^2(3{,}24 - x^3) - tg(x + 2{,}31)^2, \qquad x - \text{произвольное.}$$

3. Вычислите значение функции:

$$Y = ctg^3 \frac{\sqrt{|a - 3{,}5 + sin^2(a - b)|}}{a^2 + e^{(b+1)}}, \quad b = -3{,}07, \qquad a - \text{произвольное.}$$

4. Вычислите значение функции:

$$Y = sin^2\left(14{,}8 - \frac{\sqrt[3]{|x^2 - c \cdot d|}}{\cos(d + c) - x}\right), \quad d = -24{,}1, c = 1{,}8, \qquad x - \text{произвольное.}$$

5. Вычислите значение функции:

$$W = \frac{ln\sqrt{|n + 3{,}2 \cdot (n - 2{,}3)|}}{(n + 3{,}1)^2 + n}, \quad n - \text{произвольное.}$$

6. Вычислите значение функции:

$$M = \sqrt[3]{|sin^4(d^2 - c \cdot tg(d + c))|}, \quad c = -6{,}5, \qquad d - \text{произвольное.}$$

7. Вычислите значение функции:

$$L = cos^2 \frac{a - f^3 + e^{(a-f)}}{a + f}, \quad f = 5, \qquad a - \text{произвольное.}$$

8. Вычислите значение функции:

$$B = \frac{2{,}4 + \sin(x^2 - 3{,}1)^3}{\sqrt[3]{|x^3 - 1{,}2 \cdot \cos(x + 3{,}1)|}}, \quad x - \text{произвольное.}$$

9. Вычислите значение функции:

$$N = ln^2 \left|\frac{7{,}1 - cos^2(k + \sin(k + 1))}{k + 1{,}8}\right|, \quad k - \text{произвольное.}$$

10. Вычислите значение функции:

$$Y = \frac{z^2 + cos^3(z + 14{,}5)}{\lg(|z - 3{,}9|)}, \quad z - \text{произвольное.}$$

11. Вычислите значение функции:

$$D = \sqrt[5]{\frac{a + 4x - \cos(x-a)^4}{7{,}2 - ln^2(2{,}4 + a)}},\ \ a = 1{,}6, \qquad x - \text{произвольное.}$$

12. Вычислите значение функции:

$$W = \frac{\cos(2{,}14 + x) - 41}{arctg(x - 10)},\quad x - \text{произвольное.}$$

13. Вычислите значение функции:

$$Y = \frac{2x - 3sin(x - 1{,}35)^2}{\sqrt{5x + lnx^2}},\ \ x - \text{произвольное}, x \neq 0\ .$$

14. Вычислите значение функции:

$$Z = \frac{cos(\pi + x) - 45}{arctg(x - 11)},\ \ x - \text{произвольное.}$$

15. Вычислите значение функции:

$$R = \sqrt[3]{|cos^3(d \cdot \sin(d + c))|},\ \ d = 7{,}43, \qquad c = 6{,}87.$$

16. Вычислите значение функции:

$$Y = cosx + sin^2(a + b) - e^x,\ \ a = log_5(1 + c^3),\ \ b = \frac{\sqrt[3]{|x|}}{c + 1},$$

$c = 2{,}5x,\ \ x -$ произвольное.

17. Вычислите значение функции:

$$K = \sqrt[5]{\frac{x^2 + a - sin(x-a)^3}{2.4 - lg^2(3.6 + a)}},\ \ a = 1{,}6, \qquad x - \text{произвольное.}$$

18. Вычислите значение функции:

$$S = \sqrt[4]{|tg^3(c \cdot \cos(d + c))|},\ \ d = 6{,}33, \qquad c = 7{,}27.$$

19. Вычислите значение функции:

$$Y = \frac{e^{2+x} - 5sin(x + 5{,}32)^2}{\sqrt{4x + lgx^2}},\ \ x - \text{произвольное положительное число.}$$

20. Вычислите значение функции:

$$N = ln^3 \left| \frac{4{,}5 + sin^2(k + \cos(k+1))}{k + 1{,}8} \right|, \quad k - \text{произвольное.}$$

21. Вычислите значение функции:

$$Y = tg^2 \frac{\sqrt{|b - 3{,}5 + sin^2(a+b)|}}{b^2 + e^{(a+1)}}, \quad b = -3{,}07, \qquad a - \text{произвольное.}$$

22. Вычислите значение функции:

$$K = ctg^2 \frac{a - b^3 + e^{(a+b)}}{a + b}, \quad b = 2, \qquad a - \text{произвольное.}$$

23. Вычислите значение функции:

$$Y = \frac{(b^2 - 12{,}5x)^3}{(x+b)^2} + \sqrt[3]{|b + x^2|}, \quad b = 3{,}27, \qquad x - \text{произвольное.}$$

24. Вычислите значение функции:

$$Y = \frac{za^2 + tg^2(z + 14{,}5a)}{\ln(|z - 3{,}9|)}, \quad a, z - \text{произвольное.}$$

6 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какая величина называется константой?
2. Какая величина называется переменной?
3. Что представляет собой программа на языке Python?
4. Для чего нужны комментарии в программе?
5. Перечислите основные типы данных в Python.
6. Перечислите основные операции с вещественными и целыми числами?
7. Приоритет выполнения операций при записи арифметических выражений.
8. Как записываются унарные операции?
9. Перечислите основные операции со строками.
10. Что представляет собой тип список?
11. Синтаксис функции для ввода данных.
12. Синтаксис оператора вывода данных.
13. Какой вычислительный процесс называется линейным?

ЛАБОРАТОРНАЯ РАБОТА № 8

Тема: **УПРАВЛЯЮЩИЕ ОПЕРАТОРЫ РАЗВЕТВЛЯЮЩЕГОСЯ ВЫЧИСЛИТЕЛЬНОГО ПРОЦЕССА. ОПЕРАТОРЫ IF И IF-ELSE**

1 ЦЕЛЬ РАБОТЫ

Познакомиться:

– с управляющими операторами разветвляющегося вычислительного процесса IF, IF-ELSE.

Научиться:

– реализовывать разветвляющиеся алгоритмы на языке программирования Python.

2 ОБЪЁМ РАБОТЫ И ПОРЯДОК ЕЁ ВЫПОЛНЕНИЯ

Данная работа выполняется в течение двух часов. Выполнение работы идет одновременно с изучением теоретической части. Для закрепления материала необходимо выполнить упражнения для самостоятельной работы и оформить отчет.

3 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ. ТЕХНОЛОГИЯ ВЫПОЛНЕНИЯ РАБОТЫ

Вычисления, в процессе которых необходимо совершить те или иные действия в зависимости от результата проверки какого-либо условия, получили название **разветвляющегося вычислительного процесса**.

Разветвляющийся вычислительный процесс реализуется с помощью условных операторов.

3.1 Оператор условного перехода if

Оператор if содержит в себе логическое условие, в котором производится сравнение данных и по результату выбирается дальнейшее действие.

Синтаксис оператора:

```
if логическое_выражение:
    инструкция1
    инструкция2
...
    инструкцияN
```

Если логическое выражение принимает значение истина (True), то выполняются инструкции, определяемые данным оператором. Блок кода, который необходимо выполнить в случае истинности логического выражения отделяется четырьмя пробелами слева либо количеством пробелов, кратным четырем.

Структура «Разветвление» для оператора if выглядит, как показано на рисунке 8.1.



Рисунок 8.1 – Структура «Разветвление» для оператора if

Упражнение 1. В этом упражнении вы составите блок-схему и разработаете программу для следующей задачи.

Задача. Даны два числа a и b. Возвести в квадрат эти числа, если a больше b. Вывести результат на экран.

Решение:

1. Составьте блок-схему алгоритма (рисунок 8.2).

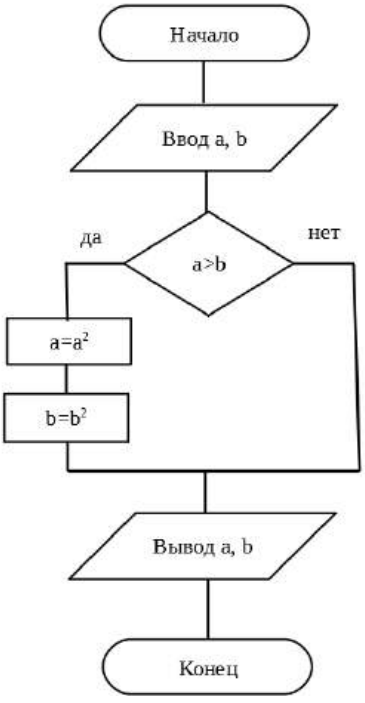


Рисунок 8.2 – Блок-схема алгоритма

2. Создайте файл с именем **Lab8_1.py**
3. Откройте созданный файл и наберите текст программы (рисунок 8.3).

```
#coding=utf-8
from math import *
a=float(input('Введите a= '))
b=float(input('Введите b= '))
if a>b:
        a**=2
        b**=2
print 'a= ', a, '\n', 'b= ', b
```

Рисунок 8.3 – Текст программы

4. Сохраните файл и закройте его.
5. Выполните отладку программы для значений a=3, b=7 и a=8, b=2.

3.2 Логические выражения. Логический тип данных

Для записи условий используются шесть знаков **отношений**: > (больше), < (меньше), >= (больше или равно), <= (меньше или равно), == (равно), != (не равно).

Операция отношения может иметь только одно из двух результирующих значений – True (Истина) и False (Ложь), которые могут быть присвоены переменным типа Boolean или определенному свойству объекта.

С переменными типа Boolean (логическими переменными) можно выполнять ряд специальных операций, которые называются логическими или булевыми. К ним относят:

and – конъюнкция (логическое умножение);

or – дизъюнкция (логическое сложение);

not – инверсия (логическое отрицание).

Так же, как и для операций отношения, результатом выполнения логической операции может быть True или False.

Логические выражения типа a >= 1023 являются простыми, так как в них выполняется только одна логическая операция. Однако на практике нередко возникает необходимость в более сложных выражениях.

В таких случаях используются специальные операторы, объединяющие два и более простых логических выражения. Широко используются два оператора – так называемые логические и (and) и или (or).

Допустим, переменной x было присвоено значение **2** (x = 2), переменной y присвоили **21** (y = 21). Логическое выражение y<23 and x>8 будет выполняться следующим образом. Сначала выполнится выражение y<23. Его результатом будет **True**. Затем выполнится выражение x>8. Его результатом будет **False**. Далее выражение сведется к **True and False**, что вернет **False**.

В случае с оператором or второе простое выражение проверяется, если первое вернуло ложь, и не проверяется, если уже первое вернуло истину. Так как для истинности всего выражения достаточно единственного **True**, неважно по какую сторону от or оно стоит.

В языке Python есть еще унарный логический оператор not, т. е. отрицание. Он превращает истину в ложь, а ложь в истину. Унарный он потому, что применяется к одному выражению, стоящему после него, а не справа и слева от него как в случае бинарных and и or.

3.3 Оператор условного перехода if–else

Оператор if–else содержит в себе логическое условие, в котором производится сравнение данных и по результату выбирается дальнейшее действие.

Синтаксис оператора:

```
if логическое_выражение:
    инструкция1
    инструкция2
...
    инструкцияN
else:
    инструкция1
    инструкция2
...
    инструкцияN
```

Если логическое выражение принимает значение истина (True), то выполняются инструкции, определяемые оператором if. Если логическое выражение принимает значение истина (False), то выполняются инструкции, определяемые оператором else (рисунок 8.4).



Рисунок 8.4 – Структура «Разветвление» для оператора if-else

Упражнение 2. В этом упражнении вы составите блок-схему и напишите программу, вычисляющую значение функции

$$y = \begin{cases} \frac{1}{\sqrt{|x-a|}} + ln^2(x+a), \text{если } x > a \\ cos^3(x+a)^2, \text{ если } x \le a \end{cases},$$

где x, a – произвольные числа, введённые с клавиатуры.

Решение:

1. Составьте блок-схему алгоритма (рисунок 8.5).

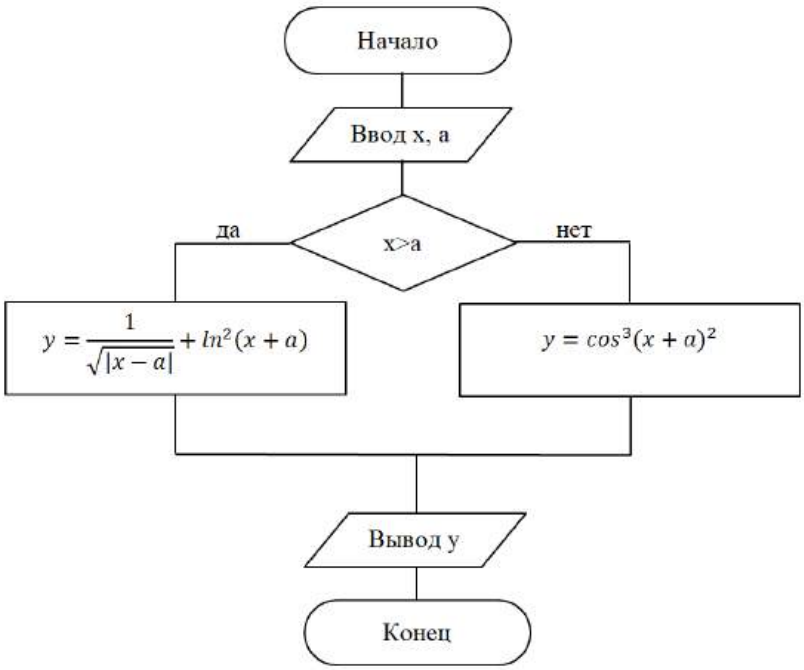


Рисунок 8.5 – Блок-схема алгоритма

2. Создайте файл с именем **Lab8_2.py**
3. Откройте созданный файл и наберите текст программы (рисунок 8.6).

```
#coding=utf-8
from math import *
a=float(input('Введите a= '))
x=float(input('Введите x= '))
if x>a:
        y=1/sqrt(abs(x-a))+log(x+a)**2
else:
        y=cos((x+a)**2)**3
print 'y= ', y
```

Рисунок 8.6 – Текст программы

4. Сохраните файл и запустите программу на выполнение.

4 СОДЕРЖАНИЕ ОТЧЕТА

Отчет должен содержать:
- тему лабораторной работы;
- цель лабораторной работы;
- номер варианта, указанный преподавателем;
- текст задачи, соответствующего варианта;
- блок-схему алгоритма;
- текст программы **Lab8_3.py**;
- исходные данные;
- результат выполнения программы;
- ответы на контрольные вопросы.

5 УПРАЖНЕНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Выполните задание, согласно варианту, указанному преподавателем. Имя файла – **Lab8_3.py**

1. Составьте блок-схему алгоритма и напишите программу. Даны два числа
x, y. Возвести эти числа в квадрат, если x < y, в противном случае – увеличить каждое число на 2.

2. Составьте блок-схему алгоритма и напишите программу. Даны два числа. Вывести на экран те из них, которые принадлежат полуинтервалу $[3, 9)$.

3. Составьте блок-схему алгоритма и напишите программу. Даны три числа
x, y, z. Возвести в квадрат отрицательные числа.

4. Составьте блок-схему алгоритма и напишите программу. Даны три вещественных числа. Возвести в квадрат отрицательные числа и в пятую степень – положительные, число 0 не изменять.

5. Составьте блок-схему алгоритма и напишите программу для проверки условия: является ли треугольник со сторонами a, b равнобедренным.

6. Составьте блок-схему алгоритма и напишите программу для проверки условия: является ли треугольник со сторонами a, b, c прямоугольным.

7. Составьте блок-схему алгоритма и напишите программу. Даны вещественные числа x, y ($x \neq y$). Каждое из чисел заменить модулем, если оба числа отрицательные, в противном случае – увеличить числа в три раза.

8. Составьте блок-схему алгоритма и напишите программу. Даны вещественные числа x, y ($x \neq y$). Заменить суммой чисел меньшее из них,

удвоенным произведением чисел – большее из них.

9. Составьте блок-схему алгоритма и напишите программу. Даны три точки
x, y, z на числовой оси. Определить, какая из двух последних точек (y или z) расположена ближе к x. Вывести координату этой точки и расстояние до точки x.

10. Составьте блок-схему алгоритма и напишите программу. Даны координаты точки на координатной плоскости. Вывести на экран: 0 – если точка не лежит на координатных осях; 1 – в противном случае.

11. Составьте блок-схему алгоритма и напишите программу для проверки условия: лежит ли точка с координатами (x, y) в первой четверти координатной плоскости.

12. Составьте блок-схему алгоритма и напишите программу. Даны два числа – x, y. Вывести на экран те числа, которые кратны 5.

6 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какой вычислительный процесс называется разветвляющимся?
2. С помощью каких операторов реализуются условия разветвляющегося вычислительного процесса?
3. Синтаксис оператора if.
4. Синтаксис оператора if-else.

ЛАБОРАТОРНАЯ РАБОТА № 9

Тема: УПРАВЛЯЮЩИЕ ОПЕРАТОРЫ РАЗВЕТВЛЯЮЩЕГОСЯ ВЫЧИСЛИТЕЛЬНОГО ПРОЦЕССА. ОПЕРАТОР IF-ELIF-ELSE

1 ЦЕЛЬ РАБОТЫ

Познакомиться:

– с управляющим оператором разветвляющегося вычислительного процесса **IF-ELIF-ELSE**.

Научиться:

– реализовывать разветвляющиеся алгоритмы на языке программирования Python.

2 ОБЪЁМ РАБОТЫ И ПОРЯДОК ЕЁ ВЫПОЛНЕНИЯ

Данная работа выполняется в течение двух часов. Выполнение работы идет одновременно с изучением теоретической части. Для закрепления материала необходимо выполнить упражнения для самостоятельной работы и оформить отчет.

3 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ. ТЕХНОЛОГИЯ ВЫПОЛНЕНИЯ РАБОТЫ

Вычисления, в процессе которых необходимо совершить те или иные действия в зависимости от результата проверки какого-либо условия, получили название **разветвляющегося вычислительного процесса**.

Разветвляющийся вычислительный процесс реализуется с помощью условных операторов.

3.1 Оператор условного перехода if-elif-else

Синтаксис оператора:

```
if логическое_выражение_1:
    инструкция1
...
    инструкцияN
[elif логическое_выражение_2:
    инструкция1
...
    инструкцияN]
...
[else:
    инструкция1
...
    инструкцияN]
```

Оператор elif позволяет выполнять множественную проверку выражения и выполнять блок кода, как только результат будет считаться истиной.

Упражнение 1. В этом упражнении вы составите блок-схему и напишите программу, вычисляющую значение функции

$$S = \begin{cases} \sqrt[3]{ab^2}, \text{если } a \le 3 \\ a - b, \text{если } 3 < a \le 8 \\ a + b, \text{если} a > 8 \end{cases},$$

где a, b – произвольные положительные числа, введённые с клавиатуры.

Решение:

1. Составьте блок-схему алгоритма (рисунок 9.1).

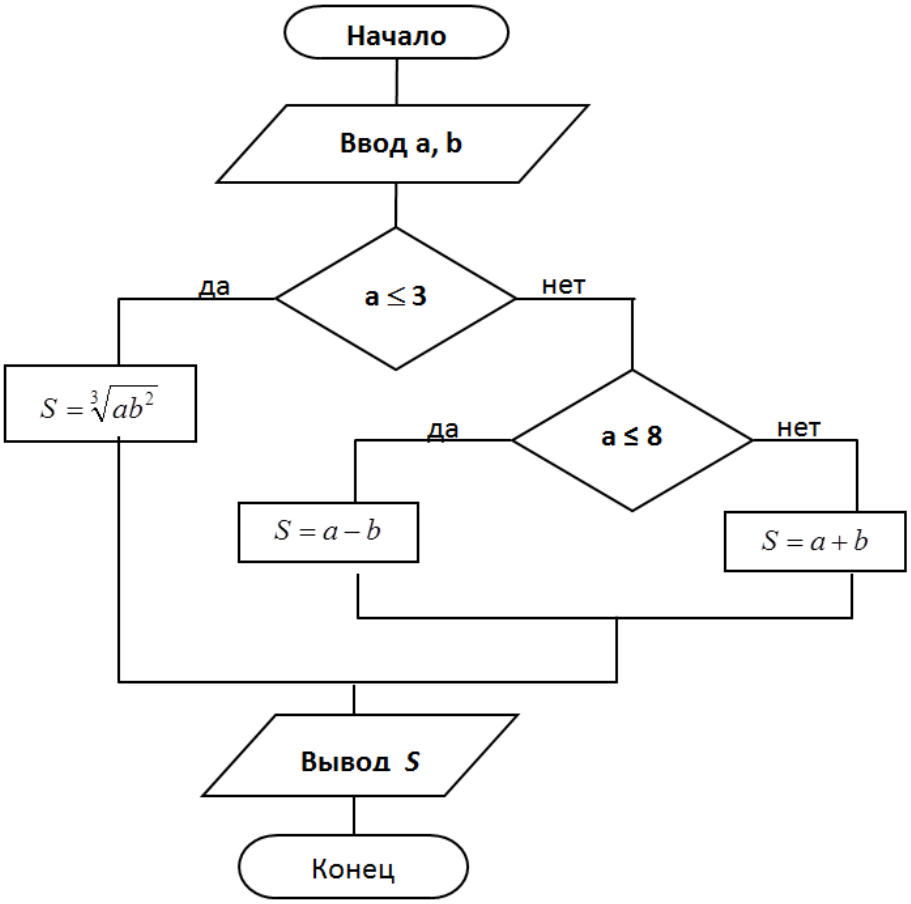


Рисунок 9.1 – Блок-схема алгоритма

2. Создайте файл с именем **Lab9_1.py**
3. Откройте созданный файл и наберите текст программы (рисунок 9.2).
4. Сохраните файл и запустите программу на выполнение.
5. Выполните отладку программы для значений **b=3**, **a=1**, **a=7**, **a=10**.

```
#coding=utf-8
from math import *
a=float(input('Введите a= '))
b=float(input('Введите b= '))
if a<=3:
        S=(a*b**2)**(1.0/3)
elif a<=8:
        S=a-b
else:
        S=a+b
print 'S= ', S
```

Рисунок 9.2 – Текст программы

Упражнение 2. В этом упражнении вы составите блок-схему и напишите программу для следующей задачи.

Задача. Дана точка А с координатами (x, y), x и y отличны от нуля. Укажите, какой четверти координатной плоскости принадлежит данная точка.

Решение:

1. Составьте блок-схему алгоритма (рисунок 9.3).

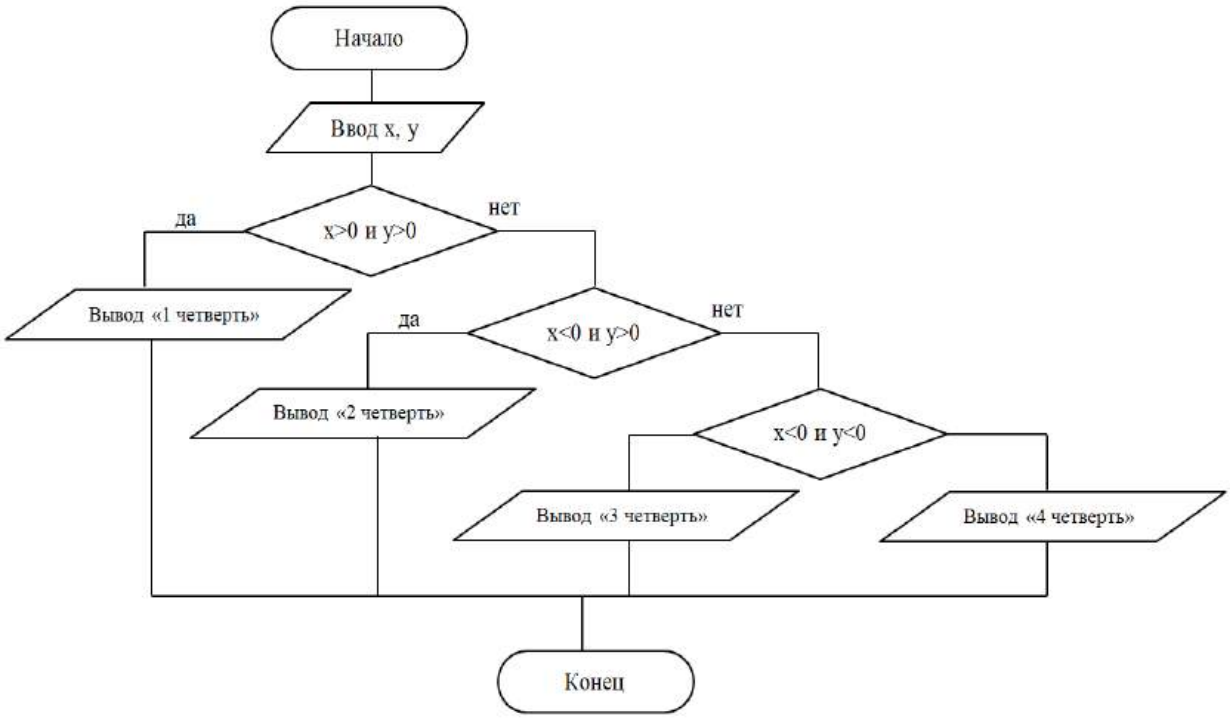


Рисунок 9.3 – Блок-схема алгоритма

2. Создайте файл с именем **Lab9_2.py**
3. Откройте созданный файл и наберите текст программы (рисунок 9.4).
4. Сохраните файл и запустите программу на выполнение.
5. Выполните отладку программы для различных значений x и y.

```
#coding=utf-8
from math import *
x=float(input('Введите координату x= '))
y=float(input('Введите координату y= '))
if x>0 and y>0:
        print '1 четверть координатной плоскости'
elif x<0 and y>0:
        print '2 четверть координатной плоскости'
elif x<0 and y<0:
        print '3 четверть координатной плоскости'
else:
        print '4 четверть координатной плоскости'
```

Рисунок 9.4 – Текст программы

4 СОДЕРЖАНИЕ ОТЧЕТА

Отчет должен содержать:

- тему лабораторной работы;
- цель лабораторной работы;
- номер варианта, указанный преподавателем;
- текст задачи соответствующего варианта;
- блок-схему алгоритма;
- текст программы **Lab9_3.py**;
- исходные данные;
- результат выполнения программы;
- ответы на контрольные вопросы.

5 УПРАЖНЕНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Выполните задание, согласно варианту, указанному преподавателем. Имя файла – **Lab9_3.py**

1. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции Z:

$$Z = \begin{cases} k^3 + \sqrt{k}, & \text{при } k > 7 \\ e^{k+1} - sin^2 k, & \text{при } -2{,}4 < k \le 7 \\ \sin(|k - 7{,}6|), & \text{при } k \le -2{,}4 \end{cases},$$

где k – произвольное число из области допустимых значений.

2. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции Y:

$$Y = \begin{cases} \dfrac{\sqrt[4]{|x+2|}}{c - lg^3(x+7)}, & \text{если } -2 < x < 5 \\ e^{(c-x)^2}, & \text{если } x \ge 5 \\ arctg(x-1), & \text{в остальных случаях} \end{cases},$$

где $c = cosx + sin^2(1+x) - e^x, x$ – произвольное.

3. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции Z:

$$Z = \begin{cases} y, & \text{при } x = 0 \\ e^{x+1} - sin^2 y, & \text{при } x > 0, \\ xy - cos20°, & \text{при } x < 0 \end{cases}$$

где x, y – произвольные.

4. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции F:

$$F = \begin{cases} |x - tgb|, & \text{при } x < 5 \\ e^{x+1,2}, & \text{при } 5 \le x < 12, \\ b + log_3(x+1), & \text{при } x \ge 12 \end{cases}$$

где x, b – произвольные.

5. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции Y:

$$Y = \begin{cases} ax + ln(x+b), & \text{если } x < b \\ tgx + \frac{ctg(a+x)}{b-1}, & \text{если } b \le x \le a, \\ \sqrt[3]{ax + b \cdot lgx}, & \text{если } x > a \end{cases}$$

где $a = 7{,}8, b = 1{,}12, x$ – произвольное.

6. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции z:

$$z = \begin{cases} \frac{ln(x+b)^2}{b \cdot e^{ax}}, & \text{если } x < 0 \\ ctgx + b \cdot sin^2(ax), & \text{если } 0 \le x \le 5, \\ tgx + |x - a|, & \text{если } x > 5 \end{cases}$$

где $a = 3{,}2, b = 7, x$ – произвольное.

7. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции f:

$$f = \begin{cases} \sqrt{|t+8|} - \frac{1}{at-c}, & \text{если } t < 0 \\ cos^3(a+t) + ct \cdot arctg(at), & \text{если } 0 \le t \le 3, \\ \frac{\sqrt{|t-c|}}{ln^3 t}, & \text{если } t > 3 \end{cases}$$

где $a = -4{,}3, c = 6, t$ – произвольное.

8. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции z:

$$z = \begin{cases} \cos(x+1), & \text{если } y < 0 \\ tg|y-1| - e^{x+2,5}, & \text{если } 0 \le y < 1, \\ y + log_5 x, & \text{если } y \ge 1 \end{cases}$$

где $y = cos^3 x + sin^3 x, x$ – произвольное положительное число.

9. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции w:

$$w = \begin{cases} \sqrt{t}, & \text{если } t = 20 \\ e^{k+1} - sin^4 k, & \text{если } t > 20, \\ cos20°, & \text{если } t < 20 \end{cases}$$

где $t = \frac{lgk + \sqrt[3]{tg(k+7)}}{7}$, k – произвольное.

10. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции r:

$$r = \begin{cases} \frac{\sqrt[7]{x}}{lg(x+0,1)-c}, & \text{если } 0,2 < x < 3 \\ tgx + |x - c|, & \text{если } x \geq 3, \\ tg(x + c), & \text{в остальных случаях} \end{cases}$$

где $c = cos^3(1 - x) - e^{x+1}$, x – произвольное положительное число.

11. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции f:

$$f = \begin{cases} |x + ctg(b)|, & \text{если } x < 5 \\ e^{x+1,2}, & \text{если } 5 \leq x \leq 12, \\ b + ln(x + 1)^2, & \text{в остальных случаях} \end{cases}$$

где $b = cosx^2$, x – произвольное положительное число.

12. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции Y:

$$Y = \begin{cases} ax + ln(x + b), & \text{если } x > b \\ \frac{\sqrt{|x+ab|}}{ln^3(x+10)}, & \text{если } a \leq x \leq b, \\ e^{x+1,2}, & \text{если } x < a \end{cases}$$

где $a = -1,8, b = 2$, x – произвольное.

13. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции Z:

$$Z = \begin{cases} k^3 + \sqrt{k + 1}, & \text{при } k > 7 \\ e^{k+1} - cos^2 k, & \text{при } -2,4 < k \leq 7, \\ e^{k+1,5}, & \text{при } k \leq -2,4 \end{cases}$$

где k – произвольное.

14. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции F:

$$F = \begin{cases} |x - ctgb|, & \text{при } x < 5 \\ bx^3 + \sqrt{|x|}, & \text{при } 5 \leq x < 12, \\ b + log_2|x + 1|, & \text{при } x \geq 12 \end{cases}$$

где x, b – произвольные.

15. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции Y:

$$Y=\begin{cases}\dfrac{ln^3(x+7)}{c-\sqrt[4]{|x+2|}}, & \text{если } 1<x<6\\ e^{(-c-x)^2}, & \text{если } x\geq 6\\ ctg(x+1), & \text{в остальных случаях}\end{cases}$$

где $c=tg^2(1+x)-e^x, x$ – произвольное.

16. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции Y:

$$Y=\begin{cases}tgx+\frac{ctg(a+x)}{b+1}, & \text{если } x<b\\ e^{x+1}-cos^2x, & \text{если } b\leq x\leq a\\ ax+ln(|x+b|), & \text{если } x>a\end{cases},$$

где $a=7{,}8; b=1{,}12; x$ – произвольное.

17. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции f:

$$f=\begin{cases}\sqrt{|x+8|}-\frac{1}{ax+c}, & \text{если } x<0\\ sin^3(a+x)+cx\cdot ctg(ax), & \text{если } 0\leq x\leq 3\\ \frac{\sqrt{|x-c|}}{ln^3x}, & \text{если } x>3\end{cases},$$

где $a=-4{,}3, c=6, x$ – произвольное.

18. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции w:

$$w=\begin{cases}\mathrm{t}+\mathrm{ctg(k)}, & \text{если } t=20\\ e^{k+1}+sin^4k, & \text{если } t>20\\ sin45°, & \text{если } t<20\end{cases},$$

где $t=\frac{lgk+\sqrt[3]{tg(k+7)}}{5}, k$ – произвольное.

19. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции z:

$$z=\begin{cases}tgx+|x-a|, & \text{если } x<0\\ tgx+b\cdot sin^2(ax), & \text{если } 0\leq x\leq 5\\ \frac{ln(x+b)^2}{b\cdot e^{ax}}, & \text{если } x>5\end{cases},$$

где $a=3{,}2, b=7, x$ – произвольное.

20. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции w:

$$w = \begin{cases} \sqrt{t+k}, & \text{если } t = 20 \\ \ln(k+3.5) - sin^4 k, & \text{если } t > 20, \\ e^{k+1}, & \text{если } t < 20 \end{cases}$$

где $t = \frac{\sqrt[3]{tg(k+7)}}{7}$, k – произвольное.

21. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции r:

$$r = \begin{cases} lg(x+0{,}1) - c, & \text{если } 0.5 < x < 4 \\ lgx + |x+c|, & \text{если } x \geq 4 \\ cos^3(1-x) - e^{x+1}, & \text{в остальных случаях} \end{cases},$$

где $c = 3.4$, x – произвольное.

22. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции F:

$$F = \begin{cases} |ctgbx + 5|, & \text{при } x < 5 \\ \sqrt{x} + bx^3, & \text{при } 5 \leq x < 12, \\ log_5(x+1) - b, & \text{при } x \geq 12 \end{cases}$$

где x, b – произвольные.

23. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции z:

$$z = \begin{cases} \sqrt{y+x}, & \text{если } y < 0 \\ cos|y-1| - e^{x+2,5}, & \text{если } 0 \leq y < 1, \\ \ln x - y, & \text{если } y \geq 1 \end{cases}$$

где $y = 5x + 8$, x – произвольное

24. Составьте блок-схему алгоритма и напишите программу для вычисления значения функции f:

$$f = \begin{cases} \frac{\sqrt{|x-c|}}{ln^3 x}, & \text{если } x < 0 \\ cx \cdot ctg(ax), & \text{если } 0 \leq x \leq 3 \\ \sqrt{|x+8|}, & x > 3 \end{cases},$$

где $a = 2{,}3$; $c = 7$, x – произвольное.

6 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какой вычислительный процесс называется разветвляющимся?
2. С помощью каких операторов реализуются условия разветвляющегося вычислительного процесса?
3. Синтаксис оператора if-elif-else.

ЛАБОРАТОРНАЯ РАБОТА № 10-11

Тема: **УПРАВЛЯЮЩИЕ ОПЕРАТОРЫ ЦИКЛИЧЕСКОГО ВЫЧИСЛИТЕЛЬНОГО ПРОЦЕССА**

1 ЦЕЛЬ РАБОТЫ

Познакомиться:

– с управляющими операторами циклического вычислительного процесса.

Научиться:

– реализовывать циклические алгоритмы на языке программирования Python.

2 ОБЪЁМ РАБОТЫ И ПОРЯДОК ЕЁ ВЫПОЛНЕНИЯ

Данная работа выполняется в течение четырёх часов. Выполнение работы идет одновременно с изучением теоретической части. Для закрепления материала необходимо выполнить упражнения для самостоятельной работы и оформить отчет.

3 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ. ТЕХНОЛОГИЯ ВЫПОЛНЕНИЯ РАБОТЫ

Процесс, в котором оператор или группа операторов повторяются многократно при различных значениях входящих величин, называется *циклическим*.

Циклический вычислительный процесс реализуется с помощью следующих операторов цикла:

– while;
– for.

3.1 Оператор цикла while

С помощью оператора цикла while реализуется циклический вычислительный процесс с предусловием – условие выхода из цикла проверяется до выполнения тела цикла (в этом случае цикл может не выполниться ни разу).

Синтаксис оператора цикла while:

```
while логическое_выражение:
    инструкция1
...
    инструкцияN
```

Семантика оператора цикла while. В этом цикле инструкции выполняются до тех пор, пока значение выражения равно true (истина). Как только значение условия становится ложным, управление программой передаётся строке кода, следующей непосредственно после цикла.

Упражнение 1. В этом упражнении вы составите блок-схему и разработаете программу, вычисляющую элементы последовательности a_n с заданной точностью ε =0,1.

$$a_n = \frac{1}{n+2}$$

Решение:

1. Составьте блок-схему алгоритма (рисунок 10.1).

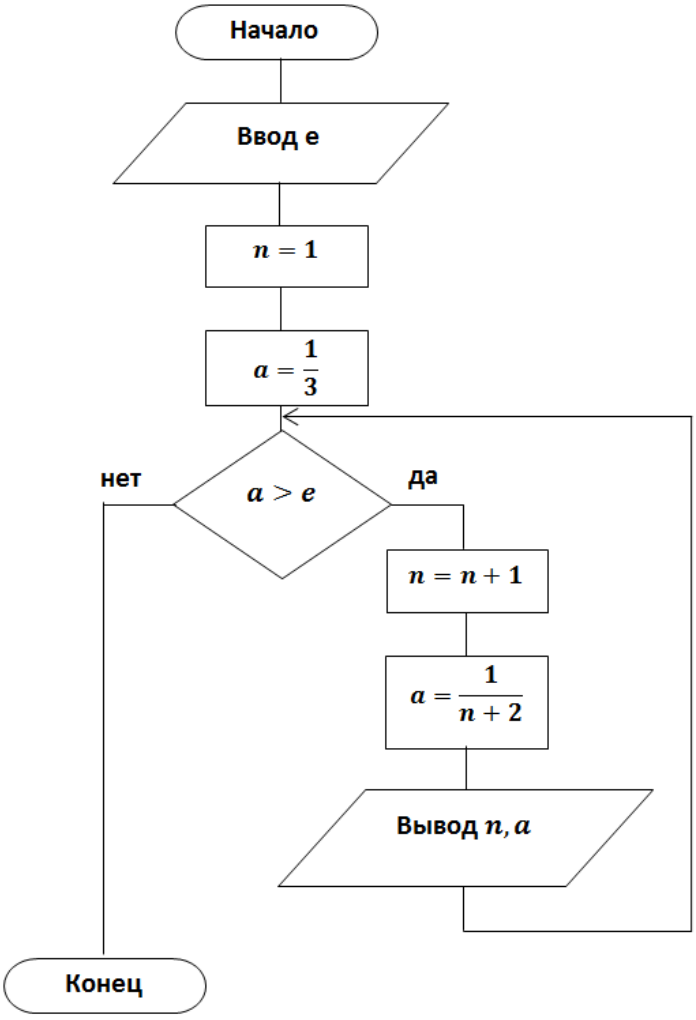


Рисунок 10.1 – Блок-схема алгоритма

2. Создайте файл с именем **Lab10_1.py**
3. Откройте созданный файл и наберите текст программы (см. рисунок 10.2).
4. Сохраните файл и запустите программу на выполнение. Результат выполнения программы сравните с рисунком 10.3.

```
#coding=utf-8
from math import *
e=float(input('Введите e= '))
n=1
a=1.0/3
while a>e:
        n=n+1
        a=1.0/(n+2)
        print 'n= ', n, '    a= ', a
```

Рисунок 10.2 – Текст программы

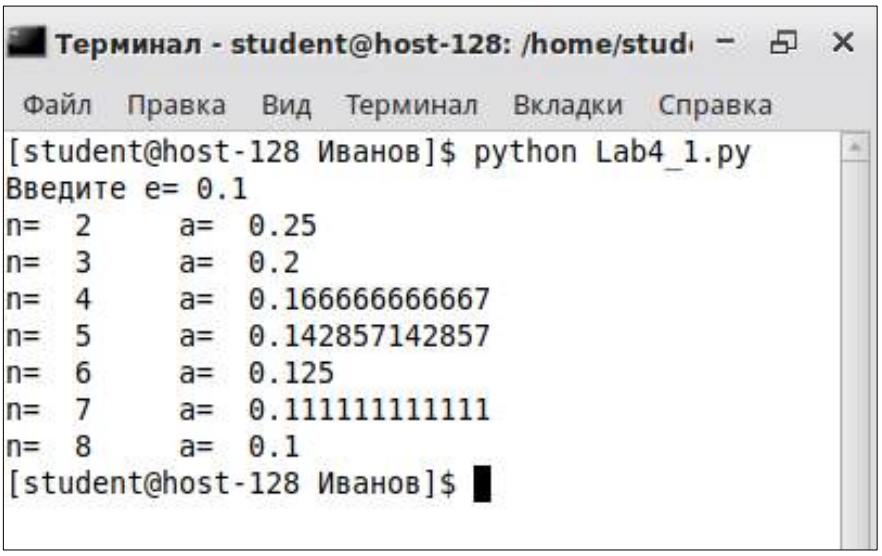


Рисунок 10.3 – Результат выполнения программы

Упражнение 2. В этом упражнении вы составите блок-схему и напишите программу, вычисляющую значения функции z:

$$z = \frac{\sqrt[3]{bx^2} + cos(x - y)}{e^{x+1,2}},$$

где $x \in [3; 6],\ \Delta x = 0{,}2,\ y = 4{,}1,\ b = 7{,}3.$

Решение:

1. Составьте блок-схему алгоритма (рисунок 10.4).
2. Создайте файл с именем **Lab10_2.py**
3. Откройте созданный файл и наберите текст программы (см. рисунок 10.5).

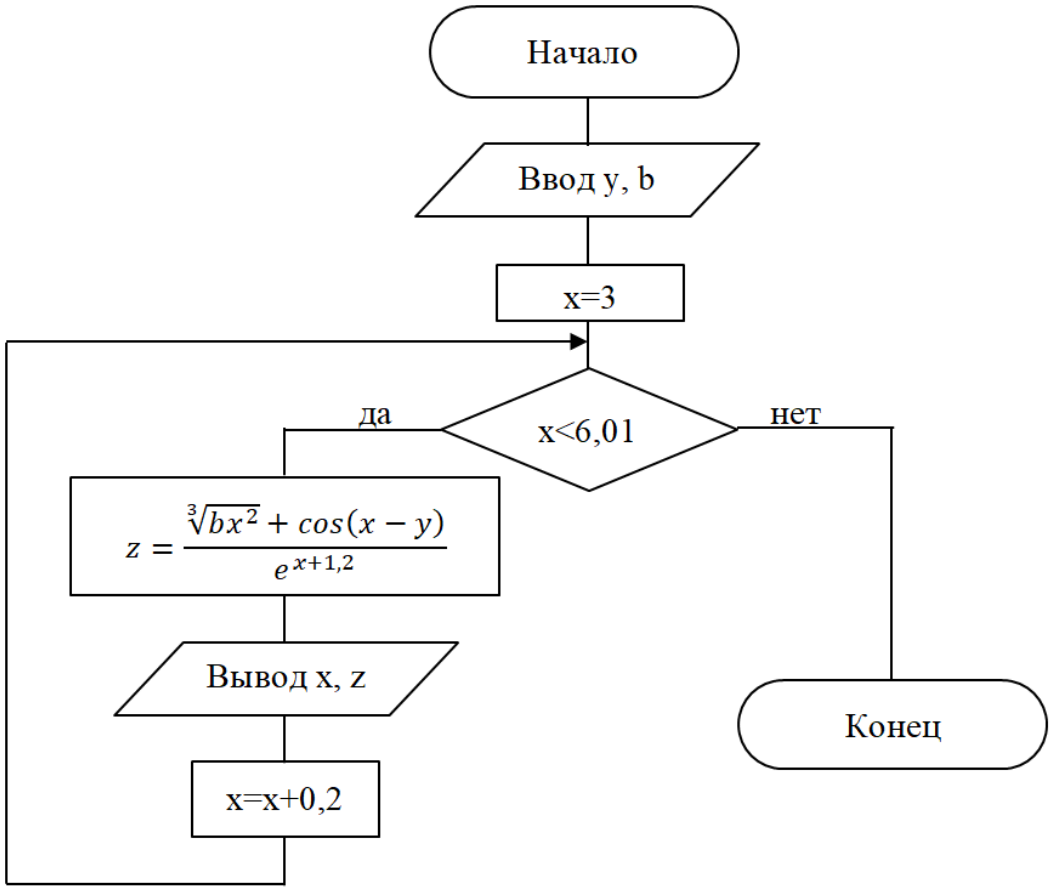


Рисунок 10.4 – Блок-схема алгоритма

```
#coding=utf-8
from math import *
y=float(input('Введите y= '))
b=float(input('Введите b= '))
x=3
while x<6.01:
        z=((b*x**2)**(1.0/3)+cos(x-y))/exp(x+1.2)
        print 'x= ', x, '    z= ', z
        x+=0.2
```

Рисунок 10.5 – Текст программы

4. Сохраните файл и запустите программу на выполнение. Результат выполнения программы сравните с рисунком 10.6.

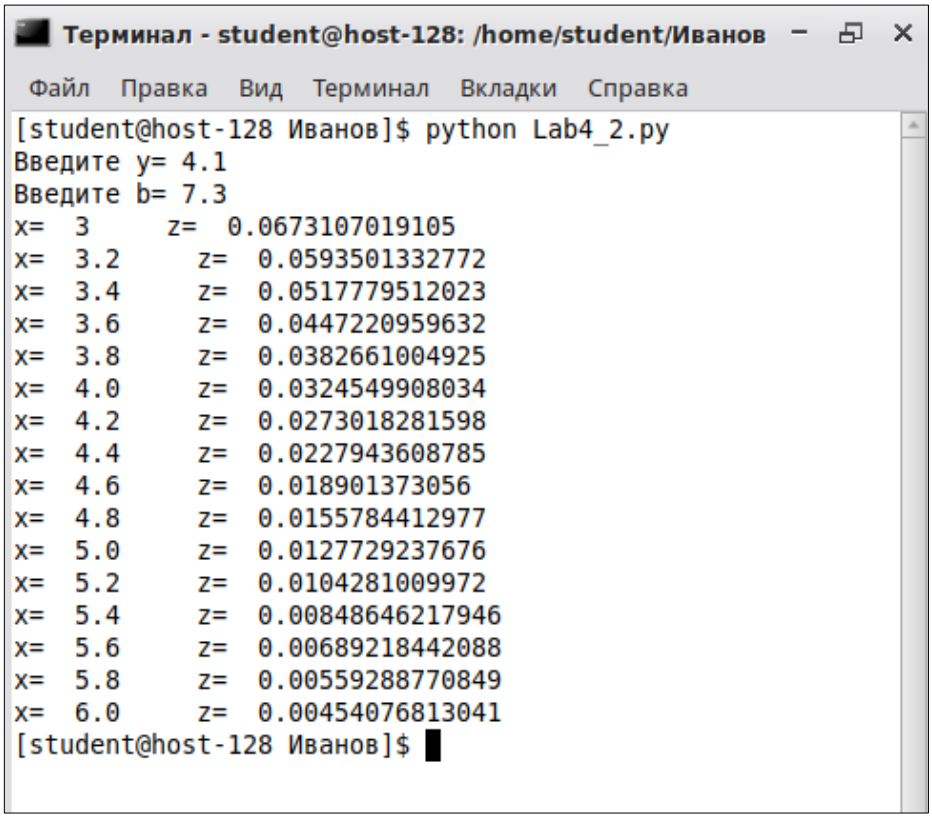


Рисунок 10.6 – Результат выполнения программы

3.2 Операторы break и continue

При работе с циклами используются операторы break и continue. Оператор break предназначен для досрочного прерывания работы цикла while и выход из него.

Оператор continue вызывает немедленный переход в начало цикла. Этот оператор иногда позволяет избежать использования вложенных инструкций.

3.3 Оператор цикла for

Оператор цикла for используется в том случае, если известно количество повторений каких-либо операций. Оператор for выполняет указанный набор инструкций заданное количество раз, которое определяется количеством элементов в наборе.

Синтаксис оператора:

```
for переменная in диапазон:
    тело цикла
```

Тело цикла – инструкции, выполняемые многократно.

Блок кода после заголовка выполняется, пока переменная принадлежит указанному диапазону (причём этот диапазон может быть списком, числовой последовательностью, массивом каких-либо значений).

3.4 Функция range

Range() является универсальной функцией Python для создания списков (list), содержащих арифметическую прогрессию. Чаще всего она используется в циклах for.

Функция range() может принимать от одного до трех аргументов, при этом аргументами должны быть целые числа (int).

Синтаксис функции:

```
range(старт, стоп, шаг)
```

где старт – начальное значение списка, если старт не задан, то равняется нулю;

шаг – разность между соседними элементами списка, если шаг не задан, то он равен единице, шаг может принимать целые положительные и отрицательные значения, не должен равняться нулю;

стоп – конечное значение списка.

Внимание! Оператор цикла for можно использовать только в случае, если шаг является целым числом, неравным нулю. В противном случае необходимо использовать цикл while!

Рассмотрим применение функции range() на следующих примерах.

```
>>>range(10)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Эта функция возвращает все целые числа в диапазоне от 0 до числа 10, не включая само число 10.

```
>>>range(5, 10)
[5, 6, 7, 8, 9]
```

Эта функция возвращает все целые числа в диапазоне от 5 до числа 10, не включая само число 10, но включая начальное число 5.

```
>>>range(0, 12, 3)
[0, 3, 6, 9]
```

Эта функция возвращает все целые числа в диапазоне от 0 до числа 10 с шагом 3, включая начальное число 0, но не включая число 12.

```
>>>range(-10, -100, -30)
[-10, -40, -70]
```

Эта функция возвращает все целые отрицательные числа в диапазоне от -10 до числа -100 с шагом -30, включая начальное число -10, но не включая число -100.

Упражнение 3. В этом упражнении вы составите блок-схему и напишите программу, вычисляющую значения функции z:

$$z = \cos\left(\frac{a^2 - b}{27}\right)$$

для $b \in [-2; 14]$, $\Delta b = 1, a$ – любое число.

Решение:

1. Составьте блок-схему алгоритма (рисунок 10.7).
2. Создайте файл с именем **Lab10_3.py**

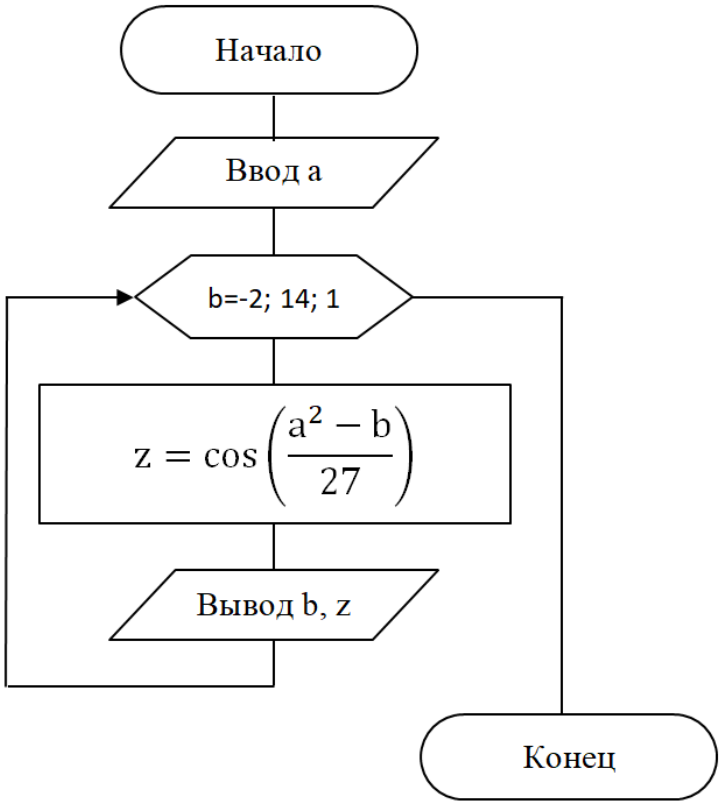


Рисунок 10.7 – Блок-схема алгоритма

3. Откройте созданный файл и наберите текст программы (рисунок 10.8). Обратите внимание на запись функции range(-2, 15). Параметры функции указывают на начальное значение (старт) переменной b = -2, и на конечное значение (стоп) переменной b = 15. Конечное значение отличается от значения в задаче, т.к. для достижения значения 14 необходимо указать значение больше, согласно семантике функции range(). Третьего значения у функции range() нет, т.к. шаг равен 1.

4. Сохраните файл и запустите программу на выполнение. Результат выполнения программы сравните с рисунком 10.9.

```
#coding=utf-8
from math import *
a=float(input('Введите a= '))
for b in range(-2, 15):
        z=cos((a**2-b)/27)
        print 'b = ', b, '     z = ', z
```

Рисунок 10.8 – Текст программы

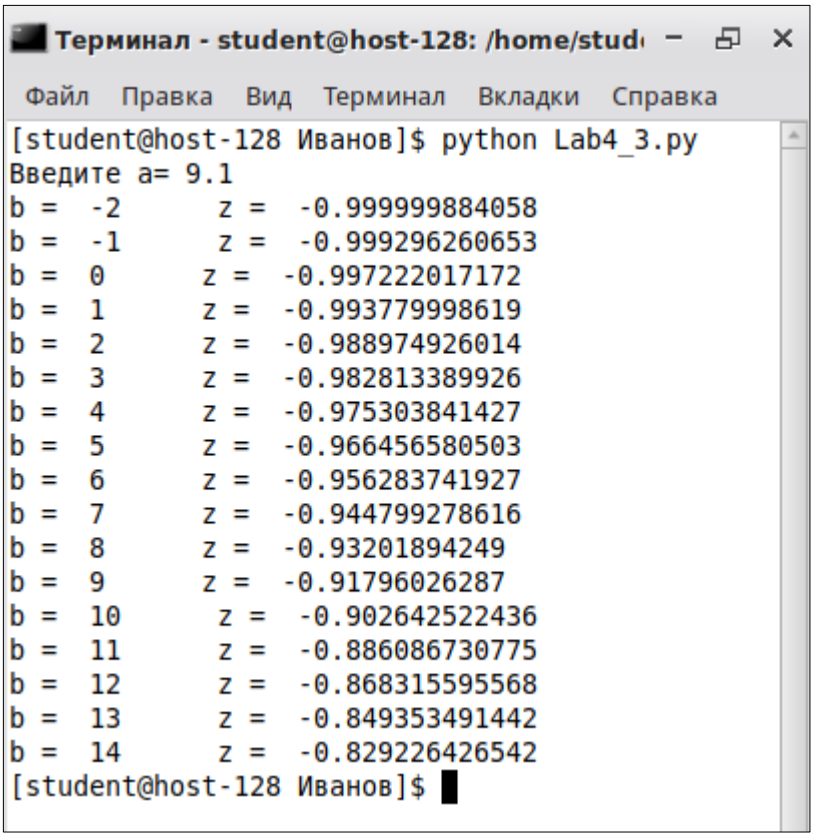


Рисунок 10.9 – Результат выполнения программы

4 СОДЕРЖАНИЕ ОТЧЕТА

Отчет должен содержать:

- тему лабораторной работы;
- цель лабораторной работы;
- номер варианта, указанный преподавателем;
- тексты задач соответствующего варианта;
- блок-схемы алгоритмов;
- тексты программ **Lab10_4.py** и **Lab10_5.py**;
- исходные данные;
- результаты выполнения программ;
- ответы на контрольные вопросы.

5 УПРАЖНЕНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

1. Выполните задачу 1, согласно варианту, указанному преподавателем. Имя файла – **Lab10_4.py**

Задача 1.

1. Составьте блок-схему алгоритма и напишите программу. Вычислите значение функции $z = \lg(m)$ для $m \in [2;7],\ \Delta m = 1$.

2. Составьте блок-схему алгоритма и напишите программу. Вычислите значение функции $y = 3x + \sqrt{|ax - e^{3x}|}$ для $x \in [2;16],\ \Delta x = 2,\ a = \lg(x-1)$.

3. Составьте блок-схему алгоритма и напишите программу. Вычислите значение функции $z = \sin^2(t+b) - \sqrt[4]{|t-b|}$ для $t \in [-1;9],\ \Delta t = 1,\ b = e^{2t-t^2}$.

4. Составьте блок-схему алгоритма и напишите программу. Вычислите значение функции $f = tg(x+ax)^2$ для $x \in [0;8],\ \Delta x = 1,\ a = \frac{\ln|3b-2|}{7},\ b = 2{,}1$.

5. Составьте блок-схему алгоритма и напишите программу. Вычислите значение функции $R = \ln|x+ax| - \sqrt[3]{|x-ba^2|}$ для $x \in [-1;11],\ \Delta x = 2,\ a = 2b + \cos^2(b),\ b = 3{,}1$.

6. Составьте блок-схему алгоритма и напишите программу. Вычислите значение функции $f = \sin(2x+y) - a$ для $x \in [1;20],\ \Delta x = 1,\ a = 4{,}2,\ y = 1{,}3$.

7. Составьте блок-схему алгоритма и напишите программу. Вычислите значение функции $M = x + 2{,}4y^3 - b$ для $y \in [2;13],\ \Delta y = 2,\ b = 1{,}3,\ x = 5{,}3$.

8. Составьте блок-схему алгоритма и напишите программу. Вычислите значение функции $z = \cos(x-y) + bx^2$ для $y \in [3;8],\ \Delta y = 1,\ x = 4{,}1,\ b = 7{,}3$.

9. Составьте блок-схему алгоритма и напишите программу. Вычислите значение функции $f = 2\sin(3t-y) + bt$ для $t \in [1;8],\ \Delta t = 1,\ y = 5{,}1,\ b = 1{,}2$.

10. Составьте блок-схему алгоритма и напишите программу. Вычислите значение функции $f = 2cd - 3x^2$ для $x \in [1;9],\ \Delta x = 1,\ c = 2{,}3,\ d = 4{,}2$.

11. Составьте блок-схему алгоритма и напишите программу. Вычислите значение функции $z = \ln(x^3 - ax^2) - y$ для $x \in [2;9],\ \Delta x = 2,\ a = 0{,}21,\ y = 3{,}4$.

12. Плотность воздуха убывает с высотой h по закону $\rho = \rho_0 \cdot e^{-hz}$, где $\rho_0 = 1{,}3 кг/м^3$, $z = 1{,}4 \cdot 10^{-4} 1/м$. Составьте блок-схему алгоритма и напишите программу, которая выводит на экран таблицу зависимости плотности воздуха от высоты для значений h, изменяющихся от 0 до 1000 м с шагом 20 м.

2. Выполните задачу 2, согласно варианту, указанному преподавателем. Имя файла – **Lab10_5.py**

Задача 2.

1. Переходный процесс описывается уравнением $i = i_{пр} + Ae^{pt}$. Разработайте программу, позволяющую пользователю ввести параметры: $i_{пр}$, A, p ($p < 0$). Параметр t изменяется от 0 до $6/p$ с шагом $0{,}5/p$. На основе введенных данных программа должна вычислить значения токов в указанном временном диапазоне с заданным шагом и вывести результаты расчётов.

2. Переходный процесс описывается уравнением $u = Ae^{pt}$. Разработайте программу, позволяющую пользователю ввести параметры: A, p ($p < 0$). Параметр t изменяется от 0 до $5/p$ с шагом $0{,}25/p$. На основе введенных данных программа должна вычислить значения напряжений в указанном временном диапазоне с заданным шагом и вывести результаты расчётов.

3. Активное сопротивление (R_1, Ом) и катушка индуктивности (L_k, Гн – индуктивность катушки) соединены последовательно и подключены к идеальному источнику переменного напряжения. Для заданной электрической цепи разработать программу, позволяющую исследовать зависимость изменения модуля полного сопротивления от частоты питающего напряжения в заданном пользователем диапазоне (f_1, f_2) с заданным пользователем шагом (df). Использовать следующие зависимости:

$w = 2\pi f$ – угловая частота, рад/с;

$X_L = wL_k$ – индуктивное сопротивление катушки;

$R_{ekv} = R_1 + R_k$, Ом – полное активное сопротивление цепи, Ом;

$X_{ekv} = X_L$, Ом – полное реактивное сопротивление цепи, Ом;

$Z = \sqrt{R_{ekv}^2 + X_{ekv}^2}$ – полное сопротивление цепи, Ом.

4. Активное сопротивление (R_1, Ом) и катушка индуктивности (R_k, Ом – активное сопротивление катушки; L_k, Гн – индуктивность катушки) соединены последовательно и подключены к идеальному источнику переменного напряжения (U). Для заданной электрической цепи разработать программу, позволяющую исследовать зависимость изменения модуля тока от частоты питающего напряжения в заданном пользователем диапазоне (f_1, f_2) с заданным пользователем шагом (df). Использовать следующие зависимости:

$w = 2\pi f$ – угловая частота, рад/с;

$X_L = wL_k$ – индуктивное сопротивление катушки;

$R_{ekv} = R_1 + R_k$, Ом – полное активное сопротивление цепи, Ом;

$X_{ekv} = X_L$, Ом – полное реактивное сопротивление цепи, Ом;

$Z = \sqrt{R_{ekv}^2 + X_{ekv}^2}$ – полное сопротивление цепи, Ом.

$I = \frac{U}{Z}$, А.

5. Активное сопротивление (R_1, Ом) и катушка индуктивности (R_k, Ом – активное сопротивление катушки; L_k, Гн – индуктивность катушки) соединены последовательно и подключены к идеальному источнику переменного напряжения с частотой f. Для заданной электрической цепи разработать программу, позволяющую исследовать зависимость изменения модуля полного сопротивления от величины индуктивности в заданном пользователем диапазоне (L_{k1}, L_{k2}) с заданным пользователем шагом (dL_k). Использовать следующие зависимости:

$w = 2\pi f$ – угловая частота, рад/с;

$X_L = wL_k$ – индуктивное сопротивление катушки;

$R_{ekv} = R_1 + R_k$, Ом – полное активное сопротивление цепи, Ом;

$X_{ekv} = X_L$, Ом – полное реактивное сопротивление цепи, Ом;

$Z = \sqrt{R_{ekv}^2 + X_{ekv}^2}$ – полное сопротивление цепи, Ом.

6. Активное сопротивление (R_1, Ом) и катушка индуктивности (R_k, Ом – активное сопротивление катушки; L_k, Гн – индуктивность катушки) соединены последовательно и подключены к идеальному источнику переменного напряжения $(U, \text{В})$ с частотой f. Для заданной электрической цепи разработать программу, позволяющую исследовать зависимость изменения модуля тока от величины индуктивности в заданном пользователем диапазоне (L_{k1}, L_{k2}) с заданным пользователем шагом (dL_k). Использовать следующие зависимости:

$w = 2\pi f$ – угловая частота, рад/с;

$X_L = wL_k$ – индуктивное сопротивление катушки;

$R_{ekv} = R_1 + R_k$, Ом – полное активное сопротивление цепи, Ом;

$X_{ekv} = X_L$, Ом – полное реактивное сопротивление цепи, Ом;

$Z = \sqrt{R_{ekv}^2 + X_{ekv}^2}$ – полное сопротивление цепи, Ом.

$I = \frac{U}{Z}$, А.

7. Активное сопротивление (R_1, Ом) и конденсатор C (– ёмкость конденсатора, Ф) соединены последовательно и подключены к идеальному источнику переменного напряжения f. Для заданной электрической цепи разработать программу, позволяющую исследовать зависимость изменения модуля полного сопротивления от частоты питающего напряжения в заданном пользователем диапазоне (f_1, f_2) с заданным пользователем шагом (df). Использовать следующие зависимости:

$w = 2\pi f$ – угловая частота, рад/с;

$X_C = \frac{1}{wC}$ – емкостное сопротивление конденсатора, Ом;

$Z = \sqrt{R_1^2 + X_C^2}$ – полное сопротивление цепи, Ом.

8. Активное сопротивление (R_1, Ом) и конденсатор (C – ёмкость конденсатора, Ф) соединены последовательно и подключены к идеальному источнику переменного напряжения (U, В) с частотой f. Для заданной электрической цепи разработать программу, позволяющую исследовать зависимость изменения модуля тока от частоты питающего напряжения в заданном пользователем диапазоне (f_1, f_2) с заданным пользователем шагом (df). Использовать следующие зависимости:

$w = 2\pi f$ – угловая частота, рад/с;

$X_C = \frac{1}{wC}$ – емкостное сопротивление конденсатора, Ом;

$Z = \sqrt{R_1^2 + X_C^2}$ – полное сопротивление цепи, Ом.

$I = \frac{U}{Z}$, А.

9. Активное сопротивление (R_1, Ом) и конденсатор (C – ёмкость конденсатора, Ф) соединены последовательно и подключены к идеальному источнику переменного напряжения с частотой f. Для заданной электрической цепи разработать консольное приложение, позволяющее исследовать зависимость изменения модуля полного сопротивления Z от величины ёмкости конденсатора в заданном пользователем диапазоне (C_1, C_2) с заданным пользователем шагом (dC). Использовать следующие зависимости:

$w = 2\pi f$ – угловая частота, рад/с;

$X_C = \frac{1}{wC}$ – ёмкостное сопротивление конденсатора, Ом;

$Z = \sqrt{R_1^2 + X_C^2}$ – полное сопротивление цепи, Ом.

10. Активное сопротивление (R_1, Ом) и конденсатор (C – ёмкость конденсатора, Ф) соединены последовательно и подключены к идеальному источнику переменного напряжения (U, В) с частотойf. Для заданной электрической цепи разработать программу, позволяющую исследовать зависимость изменения модуля тока I от величины ёмкости конденсатора в заданном пользователем диапазоне (C_1, C_2) с заданным пользователем шагом (dC). Использовать следующие зависимости:

$w = 2\pi f$ – угловая частота, рад/с;

$X_C = \frac{1}{wC}$ – емкостное сопротивление конденсатора, Ом;

$Z = \sqrt{R_1^2 + X_C^2}$ – полное сопротивление цепи, Ом.

$I = \frac{U}{Z}$, А.

11. Активное сопротивление (R, Ом) и конденсатор C (C – ёмкость конденсатора, Ф) соединены последовательно и подключены к идеальному

источнику переменного напряжения с частотой f. Для заданной электрической цепи разработать программу, позволяющую исследовать зависимость изменения модуля полного сопротивления от величины активного сопротивления в заданном пользователем диапазоне (R_1, R_2) с заданным пользователем шагом (dR). Использовать следующие зависимости:

$w = 2\pi f$ – угловая частота, рад/с;

$X_C = \frac{1}{wC}$ – емкостное сопротивление конденсатора, Ом;

$Z = \sqrt{R^2 + X_C^2}$ – полное сопротивление цепи, Ом.

12. Активное сопротивление (R, Ом) и конденсатор (C – ёмкость конденсатора, Ф) соединены последовательно и подключены к идеальному источнику переменного напряжения $(U, \text{В})$ с частотой f. Для заданной электрической цепи разработать программу, позволяющую исследовать зависимость изменения модуля тока от величины активного сопротивления в заданном пользователем диапазоне (R_1, R_2) с заданным пользователем шагом (dR). Использовать следующие зависимости:

$w = 2\pi f$ – угловая частота, рад/с;

$X_C = \frac{1}{wC}$ – емкостное сопротивление конденсатора, Ом;

$Z = \sqrt{R^2 + X_C^2}$ – полное сопротивление цепи, Ом.

$I = \frac{U}{Z}$, А.

6 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какой процесс называется циклическим?
2. Какие вы знаете операторы цикла?
3. Оператор цикла while. Его синтаксис и семантика.
4. Назначение операторов break и continue.
5. Оператор цикла for. Его синтаксис.
6. Назначение и синтаксис функции range().

ЛАБОРАТОРНАЯ РАБОТА № 12

КОНТРОЛЬНАЯ РАБОТА № 3
«ИСПОЛЬЗОВАНИЕ ЯЗЫКА ПРОГРАММИРОВАНИЯ ВЫСОКОГО УРОВНЯ ДЛЯ РЕШЕНИЯ ТИПОВЫХ ЗАДАЧ В ОБЛАСТИ АГРОИНЖЕНЕРИИ»

Примерный вариант

Б И Л Е Т № 1

1. Составьте блок-схему алгоритма и программу. Вычислить значение характеристики амортизатора передней подвески автомобиля по формуле:

$$S_1 = \begin{cases} 2 \cdot \omega \cdot \varphi_1 \cdot v, & v \le 0, \\ 2 \cdot \omega \cdot \varphi_2 \cdot v, & v > 0,' \end{cases}$$

где $\omega = \sqrt{9{,}8 \cdot \frac{k}{l}}$,

ω, φ_1, φ_2 – параметры амортизатора;
v – скорость деформации упругого элемента;
k – показатель политропы;
l – идеализированная высота столба газа

2. Активное сопротивление (R_1, Ом) и конденсатор (C – ёмкость конденсатора, Ф) соединены последовательно и подключены к идеальному источнику переменного напряжения f. Для заданной электрической цепи разработайте программу, позволяющую исследовать зависимость изменения модуля полного сопротивления от частоты питающего напряжения в заданном пользователем диапазоне (f_1, f_2) с заданным пользователем шагом (df). Используйте следующие зависимости:

$w = 2\pi f$ – угловая частота, рад/с;
$X_C = \frac{1}{wC}$ – емкостное сопротивление конденсатора, Ом;
$Z = \sqrt{R_1^2 + X_C^2}$ – полное сопротивление цепи, Ом.

Результат выполнения контрольной работы № 3

1. Составьте блок-схему алгоритма и программу. Вычислить значение характеристики амортизатора передней подвески автомобиля по формуле:

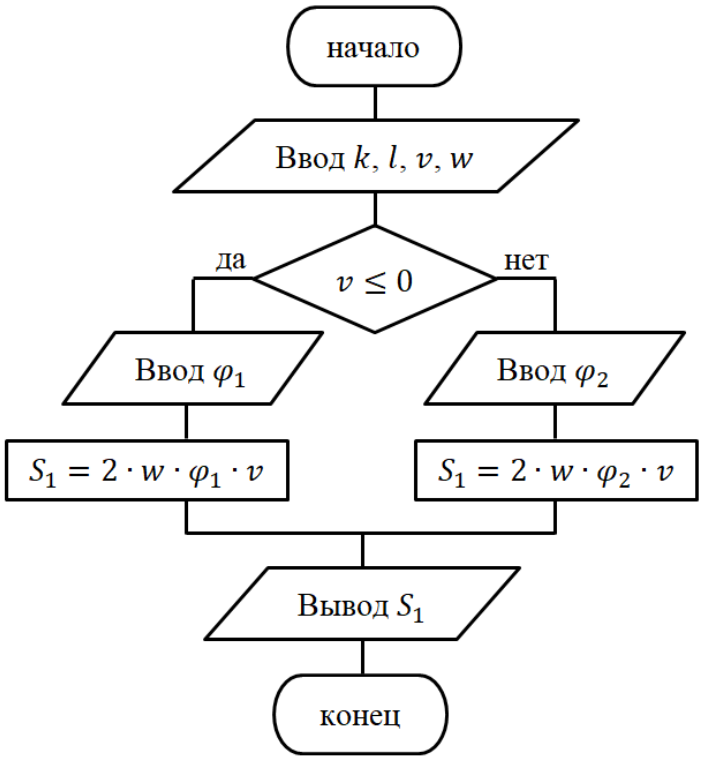
$$S_1 = \begin{cases} 2 \cdot \omega \cdot \varphi_1 \cdot v, & v \le 0, \\ 2 \cdot \omega \cdot \varphi_2 \cdot v, & v > 0,' \end{cases}$$

где $\omega = \sqrt{9{,}8 \cdot \frac{k}{l}}$,

ω, φ_1, φ_2 – параметры амортизатора;
v – скорость деформации упругого элемента;
k – показатель политропы;
l – идеализированная высота столба газа

Решение:

1) Составим блок-схему алгоритма:



2) По блок-схеме алгоритма составим программу:

```
#coding=utf-8
from math import *
k=float(input('k= '))
l=float(input('l= '))
v=float(input('v= '))
w=sqrt(9.8*k/l)
```

```
if v<=0:
        fi1=float(input('fi1= '))
        S1=2*w*fi1*v
else:
        fi2=float(input('fi2= '))
        S1=2*w*fi2*v
print 'S1= ', S1
```

2. Активное сопротивление (R_1, Ом) и конденсатор (C – ёмкость конденсатора, Ф) соединены последовательно и подключены к идеальному источнику переменного напряжения f. Для заданной электрической цепи разработайте программу, позволяющую исследовать зависимость изменения модуля полного сопротивления от частоты питающего напряжения в заданном пользователем диапазоне (f_1, f_2) с заданным пользователем шагом (df). Используйте следующие зависимости:

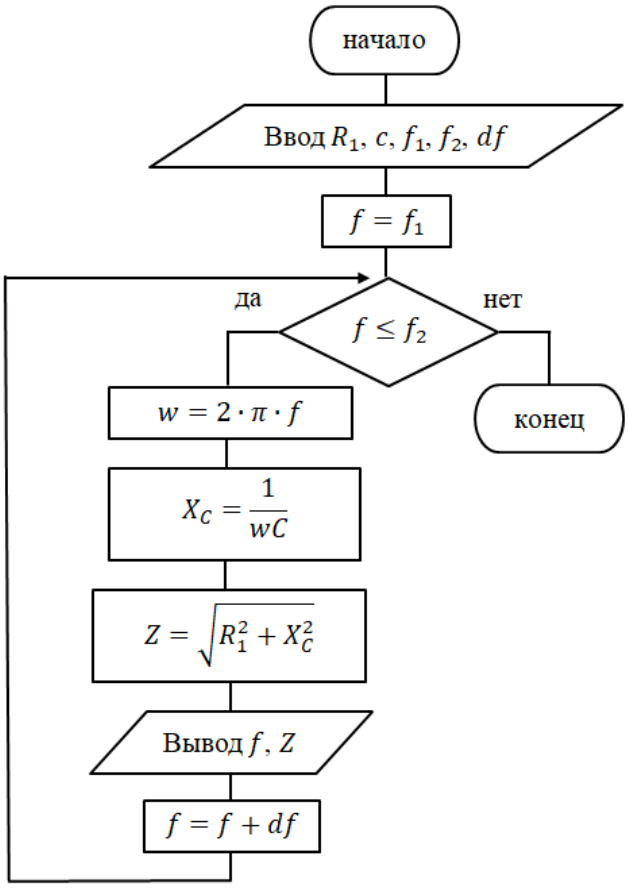
$w = 2\pi f$ – угловая частота, рад/с;

$X_C = \frac{1}{wC}$ – емкостное сопротивление конденсатора, Ом;

$Z = \sqrt{R_1^2 + X_C^2}$ – полное сопротивление цепи, Ом.

Решение:

1) Составим блок-схему алгоритма:



2) По блок-схеме алгоритма составим программу:

```
#coding=utf-8
from math import *
R1=float(input('R1= '))
c=float(input('c= '))
f1=float(input('f1= '))
f2=float(input('f2= '))
df=float(input('df= '))
f=f1
while f<=f2:
        w=2*pi*f
        Xc=1/(w*c)
        Z=sqrt(pow(R1,2)+pow(Xc,2))
        print 'f= ', f, '    Z= ', Z
        f=f+df
```

ЛАБОРАТОРНАЯ РАБОТА № 13

Тема: **КОМБИНИРОВАННЫЕ АЛГОРИТМЫ**

1 ЦЕЛЬ РАБОТЫ

Научиться:
– реализовывать комбинированные алгоритмы на языке программирования Python.

2 ОБЪЁМ РАБОТЫ И ПОРЯДОК ЕЁ ВЫПОЛНЕНИЯ

Данная работа выполняется в течение двух часов. Выполнение работы идет одновременно с изучением теоретической части. Для закрепления материала необходимо выполнить упражнения для самостоятельной работы и оформить отчет.

3 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ. ТЕХНОЛОГИЯ ВЫПОЛНЕНИЯ РАБОТЫ

К *комбинированным алгоритмам* относятся алгоритмы, в которых для решения задачи используется несколько алгоритмических структур. Например, комбинирование разветвляющейся структуры и циклической или двух циклических структур.

Упражнение 1. В этом упражнении вы составите блок-схему и напишите программу, вычисляющую значения функции Y:

$$Y = \begin{cases} a + \cos(x + b), & \text{если } x < b \\ tgx + \frac{ctg(a+x)}{b-1}, & \text{если } b \le x \le a, \\ \sqrt[3]{|a + b \cdot lgx|}, & \text{если } x > a \end{cases}$$

где $a = 5{,}6,\ b = 1{,}2,\ x \in [1; 10],\ \Delta x = 1.$

Решение:

1. Составьте блок-схему алгоритма (рисунок 13.1).

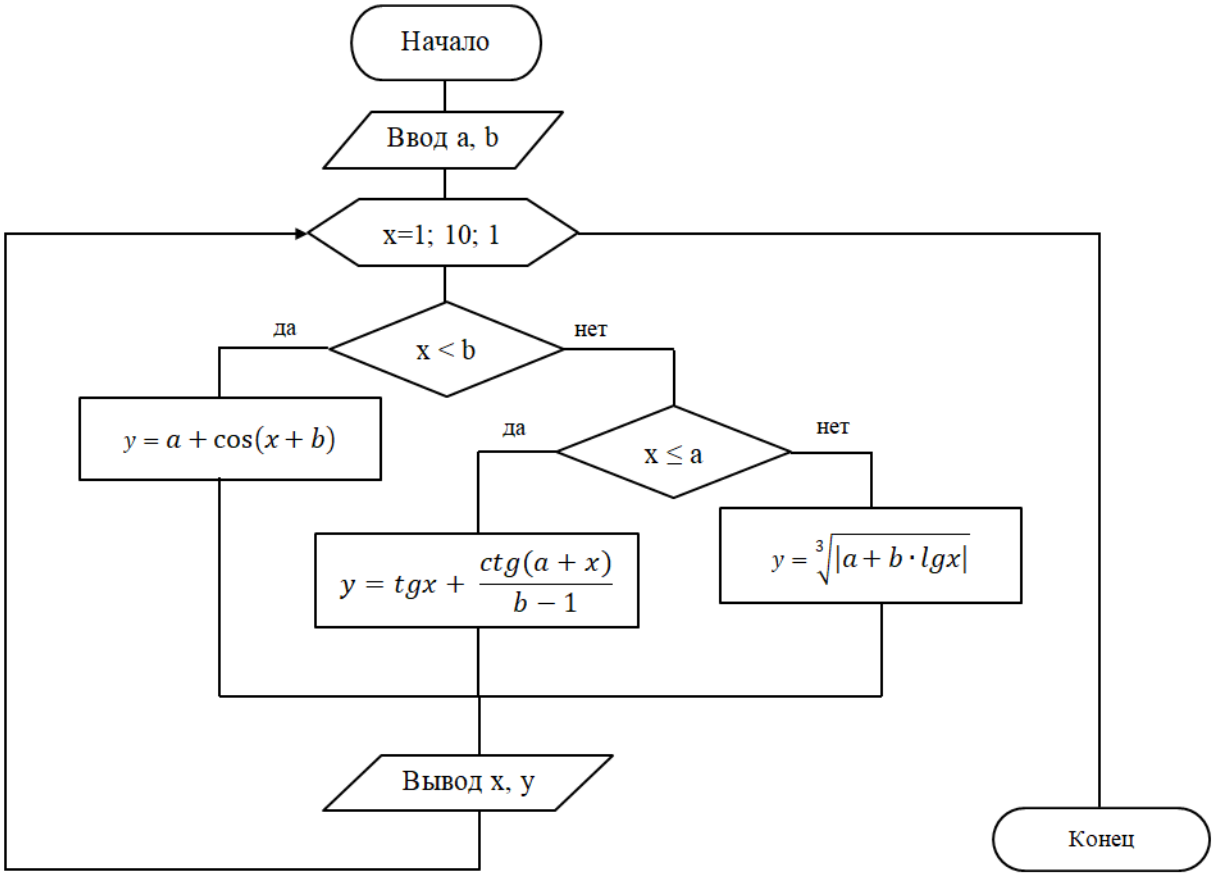


Рисунок 13.1 – Блок-схема алгоритма

2. Создайте файл с именем **Lab13_1.py**
3. Откройте созданный файл и наберите текст программы (рисунок 13.2).

```
#coding=utf-8
from math import *
a=float(input('Введите a= '))
b=float(input('Введите b= '))
for x in range(1, 11):
        if x<b:
                y=a+cos(x+b)
        elif x<=a:
                y=tan(x)+1.0/(tan(a+x)*(b-1))
        else:
                y=abs(a+b*log(x))**(1.0/3)
        print 'x = ', x, '     y = ', y
```

Рисунок 13.2 – Текст программы

4. Сохраните файл и запустите программу на выполнение. Результат выполнения программы сравните с рисунком 13.3.

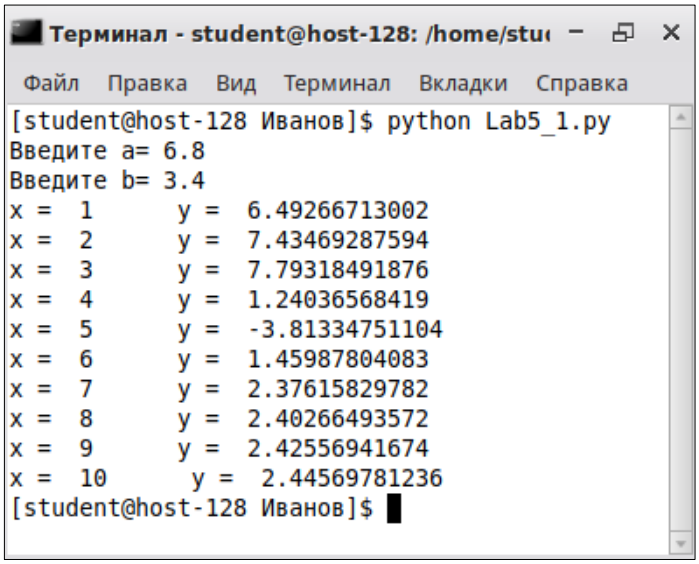


Рисунок 13.3 – Результат выполнения программы

Упражнение 2. В этом упражнении вы составите блок-схему и напишите программу, вычисляющую значения функции z:

$$z = \frac{\lg(x+y)}{\sqrt{(x+y)^2+1}},$$

где $x \in [1;10],\ \Delta x = 1,\ y \in [0{,}1;1],\ \Delta y = 0{,}1.$

Решение:

1. Составьте блок-схему алгоритма (рисунок 13.4).

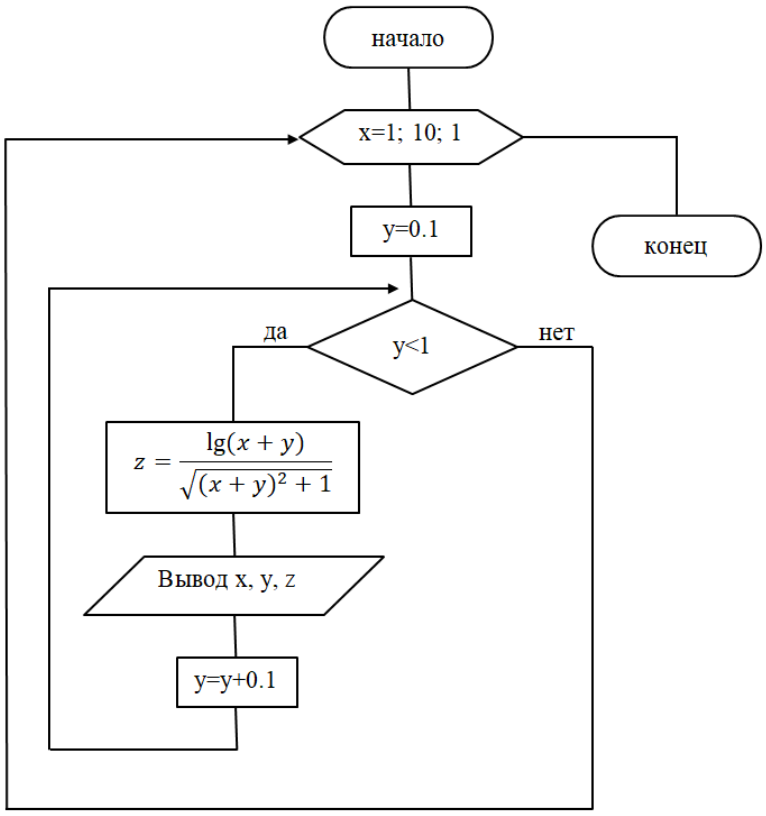


Рисунок 13.4 – Блок-схема алгоритма

2. Создайте файл с именем **Lab13_2.py**.

3. Откройте созданный файл и наберите текст программы (рисунок 13.5).

```
#coding=utf-8
from math import *
for x in range(1, 11):
        y=0.1
        while(y<1):
                z=log10(x+y)/sqrt((x+y)**2+1)
                print 'x = ', x, '     y = ', y, '     z= ', z
                y+=0.1
```

Рисунок 13.5 – Текст программы

4. Сохраните файл и запустите программу на выполнение. Результат выполнения программы сравните с рисунком 13.6.

```
Терминал - student@host-128: /home/student/Ива
Файл  Правка  Вид  Терминал  Вкладки  Справка
[student@host-128 Иванов]$ python Lab5_2.py
x =  1      y =  0.1      z=  0.0278437331764
x =  1      y =  0.2      z=  0.0506905984657
x =  1      y =  0.3      z=  0.0694724880288
x =  1      y =  0.4      z=  0.0849351955093
x =  1      y =  0.5      z=  0.0976778559517
x =  1      y =  0.6      z=  0.108183374441
x =  1      y =  0.7      z=  0.116842241284
x =  1      y =  0.8      z=  0.123971287628
x =  1      y =  0.9      z=  0.129828495283
x =  1      y =  1.0      z=  0.134624706714
x =  2      y =  0.1      z=  0.138532895056
x =  2      y =  0.2      z=  0.14169551352
x =  2      y =  0.3      z=  0.144230333446
x =  2      y =  0.4      z=  0.146235092966
x =  2      y =  0.5      z=  0.147791208969
x =  2      y =  0.6      z=  0.148966750392
x =  2      y =  0.7      z=  0.149818827875
x =  2      y =  0.8      z=  0.15039552115
x =  2      y =  0.9      z=  0.150737439301
x =  2      y =  1.0      z=  0.150878988499
x =  3      y =  0.1      z=  0.150849405955
x =  3      y =  0.2      z=  0.150673606316
x =  3      y =  0.3      z=  0.150372877122
x =  3      y =  0.4      z=  0.149965452301
x =  3      y =  0.5      z=  0.149466986797
x =  3      y =  0.6      z=  0.148890950739
x =  3      y =  0.7      z=  0.148248957911
x =  3      y =  0.8      z=  0.147551040371
x =  3      y =  0.9      z=  0.146805878801
x =  3      y =  1.0      z=  0.146020996306
x =  4      y =  0.1      z=  0.145202921971
x =  4      y =  0.2      z=  0.144357329285
x =  4      y =  0.3      z=  0.14348915363
x =  4      y =  0.4      z=  0.14260269227
```

Рисунок 13.6 – Результат выполнения программы

4 СОДЕРЖАНИЕ ОТЧЕТА

Отчет должен содержать:

- тему лабораторной работы;
- цель лабораторной работы;

- номер варианта, указанный преподавателем;
- тексты задач соответствующего варианта;
- блок-схемы алгоритмов;
- тексты программ **Lab13_3.py** и **Lab13_4.py**;
- исходные данные для обеих задач;
- результаты выполнения обеих программ;
- ответы на контрольные вопросы.

5 УПРАЖНЕНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

1. Выполните задание, согласно варианту, указанному преподавателем. Имя файла – **Lab13_3.py**.

Задача 1.

1. Составьте блок-схему алгоритма и напишите программу. Вычислить значения функции z:

$$z = \begin{cases} \frac{\ln^2(x+b)^2}{b+e^{ax}}, & \text{если } x < 0 \\ arctgx + \ b \cdot sin^2(ax), & \text{если } 0 \le x \le 2 \\ tgx + |x-a|, & \text{если } x > 2 \end{cases},$$

где $a = 2,\ b = 8,\ x\ \in [-2; 4],\ \Delta x = 0{,}5.$

2. Составьте блок-схему алгоритма и напишите программу. Вычислить значения функции f:

$$f = \begin{cases} \sqrt{|t + \mathrm{c}|} - \frac{1}{at+c}, & \text{если } t < 0 \\ sin^3(c+t) + at \cdot arctg(at), & \text{если } 0 \le t \le 3 \\ \frac{\sqrt{|t-c|}}{ln^3(t+10)}, & \text{если } t > 3 \end{cases},$$

где $a = -3{,}7,\ c = 10,\ t \in [-4; 7],\ \Delta t = 1.$

3. Составьте блок-схему алгоритма и напишите программу. Вычислить значения функции z:

$$z = \begin{cases} \frac{1}{\sqrt{x+10}}, & \text{если } y < 0 \\ tg|y-1| - e^{x+2{,}5}, & \text{если } 0 \le y < 1 \\ y + log_5(x + 7{,}7), & \text{если } y \ge 1 \end{cases},$$

где $y = \sqrt{|x+3|},\ x \in [-6; 3],\ \Delta x = 0{,}5.$

4. Составьте блок-схему алгоритма и напишите программу. Вычислить значения функции w:

$$w = \begin{cases} \sqrt{|t+8|}, & \text{если } t = 20 \\ e^{2k+1} - sin^4 k, & \text{если } t > 20 \\ cos20°, & \text{если } t < 20 \end{cases},$$

где $t = \frac{lgk + \sqrt[3]{tg(k+7)}}{7}$, $k \in [5;9]$, $\Delta k = 0{,}5$.

5. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции r:

$$r = \begin{cases} \frac{\sqrt[7]{x}}{\lg(|x+0{,}1|)-c}, & \text{если } 0{,}2 < x < 3 \\ tg(x+c), & \text{в остальных случаях} \end{cases},$$

где c $= sin^3(1-x) - e^{7x-1}$, $x \in [-3;6]$, $\Delta x = 1$.

6. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции f:

$$f = \begin{cases} \sqrt{|x + ctg(b)|}, & \text{если } x < 1 \\ e^{x+1{,}2}, & \text{если } 1 \le x \le 4{,}7 \\ b + \ln(x+1)^2, & \text{в остальных случаях} \end{cases},$$

где $b = cosx^2$, $x \in [0;6]$, $\Delta x = 0{,}2$.

7. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции Z:

$$Z = \begin{cases} k^3 + \sqrt{k}, & \text{при } k > 0 \\ e^{k+1} - sin^2 k, & \text{при } k \le 0 \end{cases},$$

где $k \in [-2;3]$, $\Delta k = 0{,}5$.

8. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции Y:

$$Y = \begin{cases} \frac{\sqrt[4]{|x+2|}}{c - lg^3|x|}, & \text{если } -2 < x < 5 \\ arctg(x-1), & \text{в остальных случаях} \end{cases},$$

где $c = cosx + sin^2(1+x) - e^x, x \in [-3.2;6.4]$, $\Delta x = 0{,}6$.

9. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции Z:

$$Z = \begin{cases} y, & \text{при } x = 0 \\ e^{x+1} - sin^2 y, & \text{при } x > 0, \\ xy - cos20°, & \text{при } x < 0 \end{cases}$$

где $x -$ произвольное, $y \in [-2; 12]$, $\Delta y = 2$.

10. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции F:

$$F = \begin{cases} |x - tgb|, & \text{при } x < 5 \\ e^{x+1,2}, & \text{при } 5 \le x < 12, \\ b + log_3(x+1), & \text{при } x \ge 12 \end{cases}$$

где $x \in [2; 18]$, $\Delta x = 3, b -$ произвольное.

11. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции Y:

$$Y = \begin{cases} ax + \ln(x+b), & \text{если } x < b \\ tgx + \frac{ctg(a+x)}{b-1}, & \text{если } b \le x \le a, \\ \sqrt[3]{ax + b \cdot lgx}, & \text{если } x > a \end{cases}$$

где $a = 7{,}8$, $b = 1{,}12$, $x \in [0; 8]$, $\Delta x = 1{,}5$.

12. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции m:

$$m = \begin{cases} \sqrt{|n + cos(a)|}, & \text{если } n < 3 \\ e^{n-1,7}, & \text{если } 3 \le n \le 6, \\ a + \ln(n+1)^2, & \text{если } n > 6 \end{cases}$$

где $a = tgn$, $n \in [2; 7]$, $\Delta n = 1$.

2. Выполните задание, согласно варианту, указанному преподавателем. Имя файла – **Lab13_4.py**

Задача 2.

1. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции

$$f = \sqrt[3]{1 + xy},$$

где $x \in [3; 5]$, $\Delta x = 0{,}25$, $y \in [-1; 1]$, $\Delta y = 0{,}4$.

2. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции

$$f = \ln(ax) + 1{,}3e^{8y-x},$$

где $x \in [2;8],\ \Delta x = 0{,}3,\ y \in [1;7],\ \Delta y = 1,\ a = 3{,}1.$

3. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции

$$z = cos2y + tg^2(ay),$$

где $y \in [2;3],\ \Delta y = 0{,}2,\ a \in [1;8],\ \Delta a = 1{,}2,\ a = 1{,}2.$

4. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции

$$y = \sqrt{x} + 1{,}1a^2b,$$

где $x \in [1{,}5;2{,}5],\ \Delta x = 0{,}1,\ a \in [2;12],\ \Delta a = 2,\ b -$ произвольное.

5. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции

$$y = 1 + e^{ax} - \sqrt{bx},$$

где $x \in [1;2],\ \Delta x = 0{,}1,\ b \in [2;11],\ \Delta b = 1, a = 5 + \frac{x}{10}.$

6. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции

$$f = e^{x+1} - sin^2y,$$

где $x \in [4;5{,}5],\ \Delta x = 0{,}3,\ y \in [2;7],\ \Delta y = 1.$

7. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции

$$f = \frac{\sqrt[4]{|x+b|}}{y - lg^3x},$$

где $x \in [2;7],\ \Delta x = 0{,}5,\ y \in [-3;1],\ \Delta y = 0{,}2, b -$ произвольное.

8. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции

$$z = \sqrt{|x + ctg(b)|},$$

где $x \in [2;3],\ \Delta x = 0{,}2,\ b \in [1;8],\ \Delta b = 1{,}2.$

9. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции

$$y = sin^3(c + t) + at \cdot arctg(at),$$

где $c \in [1; 2],\ \Delta c = 0{,}1,\ t \in [2; 11],\ \Delta t = 1,\ a = 5 - c.$

10. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции

$$M = \sqrt[3]{|sin^4(d^2 - c \cdot tg(d + c))|}$$

где $d \in [0; 7],\ \Delta d = 0{,}5,\ c \in [1; 6],\ \Delta c = 1.$

11. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции

$$L = cos^2 \frac{a - f^3 + e^{(a-f)}}{a + f},$$

где $a \in [1{,}5; 2{,}5],\ \Delta a = 0{,}1,\ f \in [6; 22],\ \Delta f = 2.$

12. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции

$$Y = \frac{z^2 + cos^3(n + 14{,}5)}{|n - 3{,}9|}$$

где $n \in [1; 4],\ \Delta n = 0{,}25,\ z \in [2; 11],\ \Delta z = 2.$

6 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какой процесс называется циклическим?
2. Какие вы знаете операторы цикла?
3. Назначение и синтаксис функции range().
4. Функция range() записана в виде: range(2,7). Перечислите параметры данной функции.

ЛАБОРАТОРНАЯ РАБОТА № 14

Тема: **АЛГОРИТМЫ НАКОПЛЕНИЯ СУММЫ И ПРОИЗВЕДЕНИЯ**

1 ЦЕЛЬ РАБОТЫ

Научиться:

– реализовывать алгоритмы накопления суммы и произведения на языке Python.

2 ОБЪЁМ РАБОТЫ И ПОРЯДОК ЕЁ ВЫПОЛНЕНИЯ

Данная работа выполняется в течение двух часов. Выполнение работы идет одновременно с изучением теоретической части. Для закрепления материала необходимо выполнить упражнения для самостоятельной работы и оформить отчет.

3 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ. ТЕХНОЛОГИЯ ВЫПОЛНЕНИЯ РАБОТЫ

Типовыми алгоритмами для циклического процесса являются задачи с вычислением суммы нескольких слагаемых или произведения нескольких сомножителей. Вычисление суммы или произведения сводится к их накоплению в какой-либо ячейке памяти.

Например, необходимо найти сумму значений числа x или произведение значений числа x. Накопление суммы часто обозначают символом $\sum$, а накопление произведения символом $\prod$. Если параметры цикла известны, то они указываются, например

$$\sum_{x=1}^{10} x^2.$$

Накапливать сумму будем при помощи переменной S, а произведение – при помощи переменной P. Принцип накопления реализуется с использованием рекуррентных выражений:

$$S = S + x;$$
$$P = P \cdot x.$$

Причем до начала процесса накопления (до цикла) необходимо в ячейку для накопления суммы положить 0 (ноль, S = 0), а в ячейку для накопления произведения положить 1 (единицу, P = 1).

Накопленные сумма и произведение выводятся за циклом.

Упражнение 1. В этом упражнении вы составите блок-схему и напишите программу, вычисляющую L по формуле

$$L = \sum_{a=3}^{7} \frac{15}{(a+1)}.$$

Решение:

1. Составьте блок-схему алгоритма (рисунок 14.1).

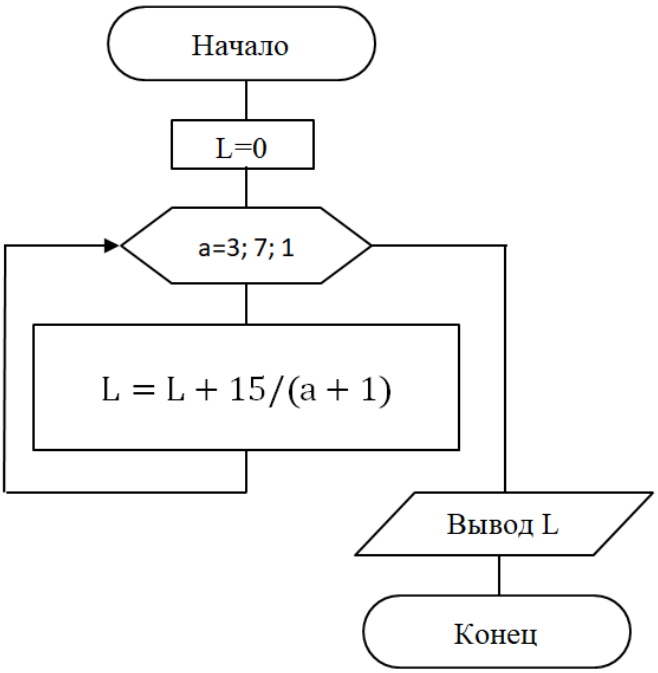


Рисунок 14.1 – Блок-схема алгоритма

2. Создайте файл с именем **Lab14_1.py**.
3. Откройте созданный файл и наберите текст программы (рисунок 14.2).

```
#coding=utf-8
from math import *
L=0
for a in range(3, 8):
        L+=15.0/(a+1)
print 'L =', L
```

Рисунок 14.2 – Текст программы

4. Сохраните файл и запустите программу на выполнение. Результат выполнения программы сравните с рисунком 14.3.

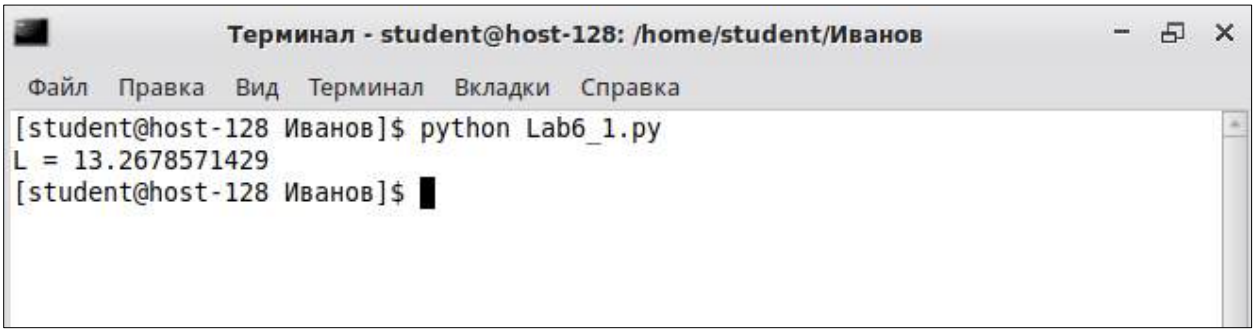


Рисунок 14.3 – Результат выполнения программы

Упражнение 2. В этом упражнении вы составите блок-схему и напишите программу, вычисляющую сумму ряда S с точностью ε по формуле:

$$S = \frac{1}{x} - \frac{2}{x^2} + \frac{3}{x^3} - \frac{4}{x^4} + \cdots + (-1)^{i-1}\frac{i}{x^i} + \cdots$$

Решение:

1. Составьте блок-схему алгоритма (рисунок 14.4).

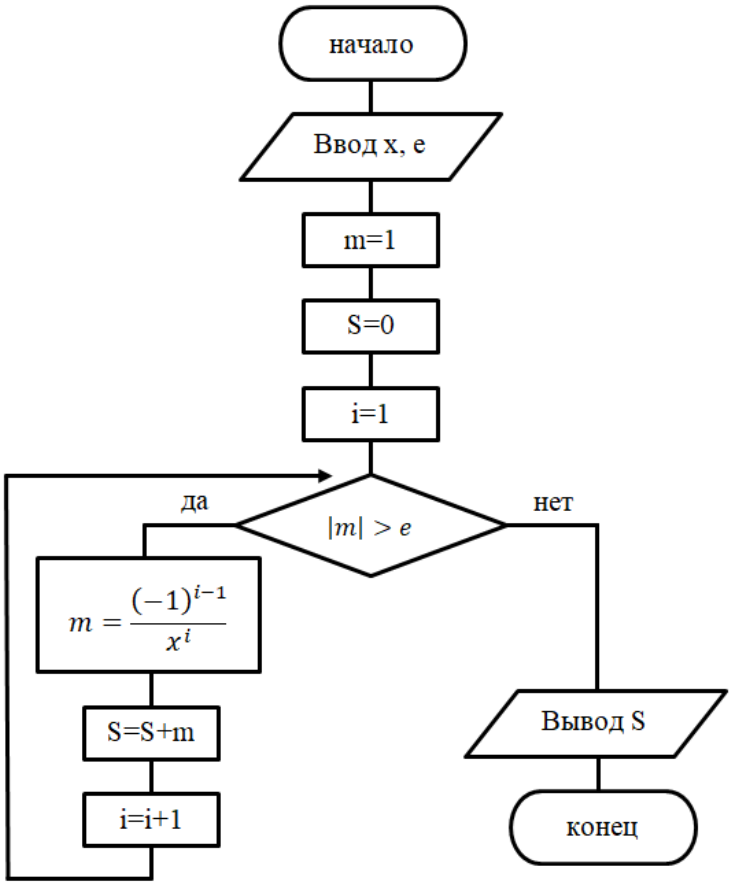


Рисунок 14.4 – Блок-схема алгоритма

2. Создайте файл с именем **Lab14_2.py**.
3. Откройте созданный файл и наберите текст программы (рисунок 14.5).

```
#coding=utf-8
from math import *
x=float(input());
e=float(input());
m=1
S=0
i=1
while (abs(m)>e):
        m=(-1)**(i-1)*i/(x**i)
        S+=m
        i+=1
print 'Сумма ряда S=', S
```

Рисунок 14.5 – Текст программы

4. Сохраните файл и запустите программу на выполнение. Выполните отладку программы для значений $\varepsilon = 0{,}01$ и $x = 8$.
5. Сравните результат выполнения программы с рисунком 14.6.

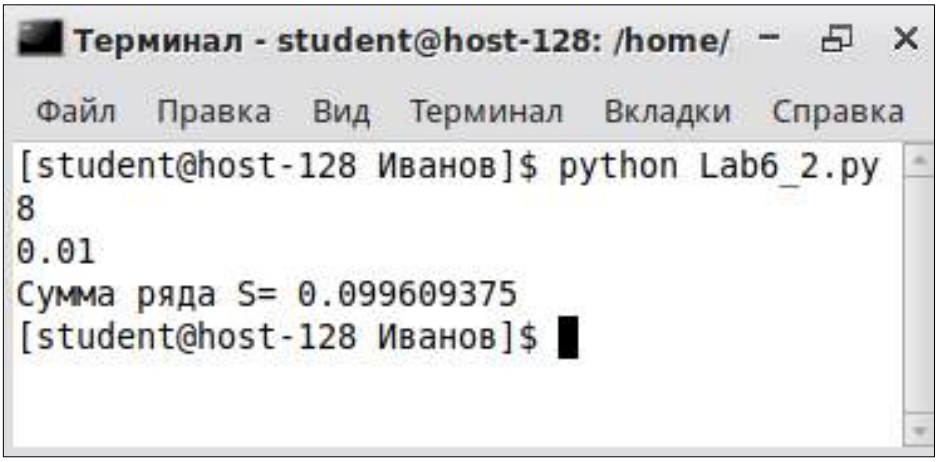


Рисунок 14.6 – Результат выполнения программы

Упражнение 3. В этом упражнении вы составите блок-схему алгоритма и напишите программу для вычисления z по формуле

$$z = \prod_{y=2}^{6}(2ab + y),\ \ \Delta y = 0{,}5,\ \ a = -1{,}58,\qquad b = 2.$$

Решение:

1. Составьте блок-схему алгоритма (рисунок 14.7).

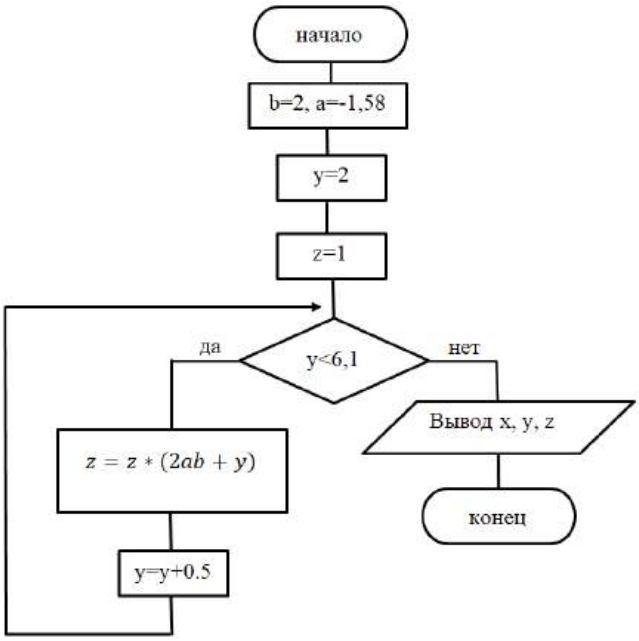


Рисунок 14.7 – Блок-схема алгоритма

2. Создайте файл с именем **Lab14_3.py**.
3. Откройте созданный файл и наберите текст программы (рисунок 14.8).

Lab6_3.py

```
#coding=utf-8
from math import *
b=2
a=-1.58
y=2
z=1
while (y<6.1):
        z*=2*a*b+y
        y+=0.5
print 'z =', z
```

Рисунок 14.8 – Текст программы

4. Сохраните файл и запустите программу на выполнение.
5. Сравните результат выполнения программы с рисунком 14.9.

Терминал - student@host-128: /home/student/Иванов

Файл Правка Вид Терминал Вкладки Справка

```
[student@host-128 Иванов]$ python Lab6_3.py
z = -225.959932607
[student@host-128 Иванов]$
```

Рисунок 14.9 – Результат выполнения программы

4 СОДЕРЖАНИЕ ОТЧЕТА

Отчет должен содержать:

- тему лабораторной работы;
- цель лабораторной работы;
- номер варианта, указанный преподавателем;
- текст задачи соответствующего варианта;
- блок-схему алгоритма;
- текст программы **Lab14_4.py**;
- исходные данные;
- результат выполнения программы.

5 УПРАЖНЕНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Выполните задание, согласно варианту, указанному преподавателем. Имя файла – **Lab14_4.py**

1. Составьте блок-схему алгоритма и напишите программу для вычисления f по формуле

$$f = \sum_{x=1}^{5} 2\sin(3x - y), \quad \Delta x = 0{,}4, \quad y = 5{,}1.$$

2. Составьте блок-схему алгоритма и напишите программу для вычисления p по формуле

$$p = \prod_{x=2}^{6} \frac{3x}{\sqrt{|sinx^3|}}, \quad \Delta x = 0{,}2.$$

3. Составьте блок-схему алгоритма и напишите программу для вычисления k по формуле

$$k = \sum_{x=1}^{2} bx^2, \quad \Delta x = 0{,}2, \quad b = 4{,}5.$$

4. Составьте блок-схему алгоритма и напишите программу для вычисления h по формуле

$$h = \prod_{n=5}^{15} \frac{1}{(3n-2)(3n+1)}.$$

5. Составьте блок-схему алгоритма и напишите программу для вычисления p по формуле

$$p = \sum_{y=-2}^{2} (2x^2 + 3y) - a, \;\; \Delta y = 0{,}1, \;\; a = 3{,}7, \qquad x = 8{,}3.$$

6. Составьте блок-схему алгоритма и напишите программу для вычисления y по формуле

$$y = \prod_{x=1}^{3} \frac{1}{2} sin\frac{x}{4} + 10, \;\; \Delta x = 0{,}1.$$

7. Составьте блок-схему алгоритма и напишите программу для вычисления z по формуле

$$z = \sum_{x=2}^{5} \ln(x^3 - ax^2) - y, \;\; \Delta x = 0{,}2, \;\; a = 0{,}2, \;\; y = 4{,}3.$$

8. Составьте блок-схему алгоритма и напишите программу для вычисления f по формуле

$$f = \prod_{a=3}^{9} \frac{a^{\ln(a)}}{(\ln(a))^a}.$$

9. Составьте блок-схему алгоритма и напишите программу для вычисления M по формуле

$$M = \sum_{x=-\frac{\pi}{2}}^{x=\frac{\pi}{2}} tg(-ax)^2, \;\; \Delta x = \frac{\pi}{6}, \;\; a = ln|3b - 2|, \;\; b = 2{,}1.$$

10. Составьте блок-схему алгоритма и напишите программу для вычисления y по формуле

$$y = k - \prod_{x=0}^{1} \frac{sinx + tg^3 x}{\sqrt[3]{x+3}}, \;\; \Delta x = 0{,}1, \;\; k = -4{,}58.$$

11. Составьте блок-схему алгоритма и напишите программу для вычисления z по формуле

$$z = \sum_{x=2}^{8} 3x + \sqrt{|ax - e^{3x}|},\ \ \Delta x = 1{,}2,\ \ a = 3{,}5.$$

12. Составьте блок-схему алгоритма и напишите программу для вычисления g по формуле

$$g = \prod_{n=1}^{8} \frac{2n-1}{2^n},\ \ \Delta n = 0{,}5.$$

ЛАБОРАТОРНАЯ РАБОТА № 15

Тема: **ОБРАБОТКА ОДНОМЕРНЫХ МАССИВОВ. ОДНОМЕРНЫЕ МАССИВЫ: НАКОПЛЕНИЕ СУММЫ И ПРОИЗВЕДЕНИЯ, ПОИСК МАКСИМАЛЬНОГО (МИНИМАЛЬНОГО) ЭЛЕМЕНТА МАССИВА**

1 ЦЕЛЬ РАБОТЫ

Познакомиться:

– с основными характеристиками массивов, с типами массивов;

– с основными способами обработки массивов на языке Python.

Научиться:

– реализовывать алгоритмы обработки массивов на языке программирования Python.

2 ОБЪЁМ ЛАБОРАТОРНОЙ РАБОТЫ И ПОРЯДОК ЕЁ ВЫПОЛНЕНИЯ

Данная лабораторная работа выполняется в течение двух часов. Выполнение работы идет одновременно с изучением теоретической части. Для закрепления материала необходимо выполнить упражнения для самостоятельной работы и оформить отчет.

3 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ. ТЕХНОЛОГИЯ ВЫПОЛНЕНИЯ РАБОТЫ

3.1 Массивы: основные понятия и определения

Массив – это упорядоченное множество однотипных переменных (элементов массива), объединенных общим именем и отличающихся номерами (*индексами*).

Массивы применяются там, где возникает необходимость обработки (поиска, перестановки, сортировки) большого количества данных, как числовых, так и символьных.

Идея дать многим переменным одно имя и разрешить использовать в качестве индекса переменную позволяет создавать очень компактные программы по обработке больших наборов данных.

Для характеристики массивов используют следующие величины:

– *Размерность* – это число индексов. Чаще всего используются одномерные (один индекс) и двумерные (два индекса) массивы.

– *Размер (глубина)* – это количество элементов в массиве.

Работа с массивами сводится к работе с его элементами. *Для обращения к элементу массива* необходимо указать *имя массива* и *индекс (порядковый номер)* элемента массива.

В одномерном массиве положение элемента (его порядковый номер) определяется одним индексом, который обозначается буквой i. Аналогом одномерного массива является линейная таблица (вектор).

Нумерация элементов массива начинается с нуля. Например, пусть X – одномерный массив, содержащий 5 вещественных чисел:

i	0	1	2	3	4
x_i	10,1	5	4,8	8,2	10

Тогда, $X_0 = 10{,}1;\ X_1 = 5;\ X_2 = 4{,}8;\ X_3 = 8{,}2;\ X_4 = 10.$

3.2 Объявление и инициализация одномерных массивов

Одномерный массив в Python представляет собой список элементов.

Для работы с массивами в Python необходимо импортировать модуль array, который определяет массивы. Для этого в программе необходимо написать команду

```
from array import *
```

Для инициализации (определения) массива в программе необходимо указать его имя, тип и значения элементов массива, например,

```
x = array('i',[2, 4, 6, 8])
```

Данная команда создает новый массив x и инициализирует его значения. Значения элементов массива перечисляются через запятую в квадратных скобках []. Любой элемент массива вызывается по индексу (номеру). Номера элементов массива начинаются с нуля, т.е. x[0]=2, x[1]=4, x[2]=6, x[3]=8.

В объявлении массива 'i' – это typecode, который сообщает Python, что массив будет хранить целые числа. Этот параметр в Python является необязательным.

Функция len() возвращает количество элементов массива.

Для перечисления элементов массива используется оператор цикла for. С помощью оператора for можно перебирать элементы массива или их индексы:

```
for x in [2, 4, 6, 8]
```

или

```
for i in range(0,5).
```

3.3 Обработка одномерных массивов

Рассмотрим на примерах инициализацию и обработку одномерных массивов на языке Python.

Упражнение 1. В этом упражнении вы составите блок-схему и напишите программу для следующей задачи.

Даны массивы X и Y, состоящие из 10 вещественных чисел каждый. Образовать массив Z, каждый элемент которого вычисляется по правилу:

$$Z_i = \sqrt[3]{X_i^2 + Y_i^2}.$$

Полученный массив вывести на экран.

Решение:

1. Составьте блок-схему алгоритма (рисунок 15.1).

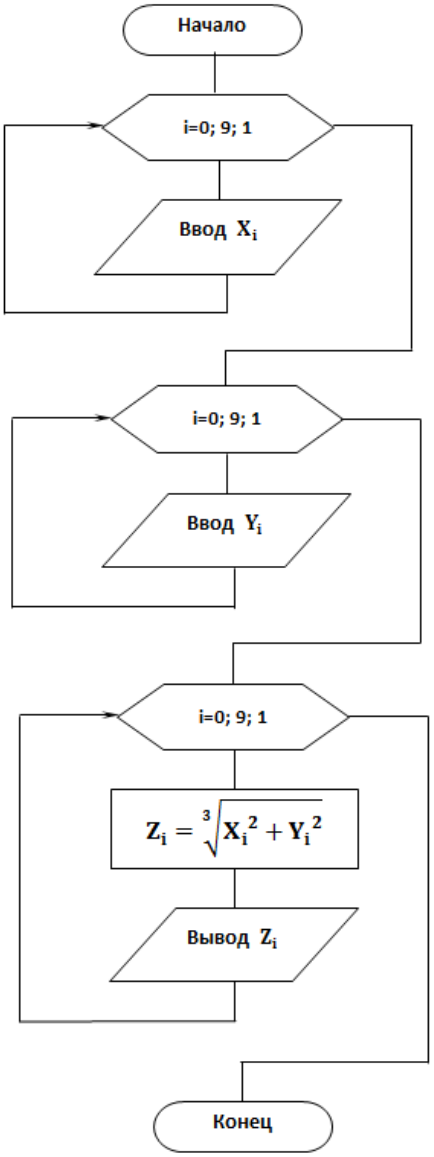


Рисунок 15.1 – Блок-схема алгоритма

2. Создайте файл с именем **Lab15_1.py**
3. Откройте созданный файл и наберите текст программы (рисунок 15.2).

```
#coding=utf-8
from math import *                                  #Импорт содержимого модуля math
from array import *                                 #Импорт содержимого модуля array
a=array("f", [0, 0, 0, 0, 0, 0, 0, 0, 0, 0])       #Инициализация массива a
b=array("f", [0, 0, 0, 0, 0, 0, 0, 0, 0, 0])       #Инициализация массива b
z=array("f", [0, 0, 0, 0, 0, 0, 0, 0, 0, 0])       #Инициализация массива z
for i in range(0, 10):
        a[i]=float(input('Введите элемент массива a ')) #Ввод элементов массива a
for i in range(0, 10):
        b[i]=float(input('Введите элемент массива b ')) #Ввод элементов массива b
for i in range(0, 10):
        z[i]=(a[i]**2+b[i]**2)**(1.0/3)             #Вычисление элементов массива z
        print 'z[', i, ']=', z[i]                   #Вывод элементов массива z
```

Рисунок 15.2 – Текст программы

4. Сохраните файл и запустите программу на выполнение.
5. Сравните результат выполнения программы с рисунком 15.3.

```
Терминал - student@host-128: /home/student/Иванов
Файл  Правка  Вид  Терминал  Вкладки  Справка
[student@host-128 Иванов]$ python Lab7_1.py
Введите элемент массива a 5.4
Введите элемент массива a 3.1
Введите элемент массива a 0.45
Введите элемент массива a -0.96
Введите элемент массива a -12
Введите элемент массива a 2
Введите элемент массива a 23
Введите элемент массива a 45
Введите элемент массива a 1
Введите элемент массива a 1
Введите элемент массива b 0.87
Введите элемент массива b 65
Введите элемент массива b -43
Введите элемент массива b 2.34
Введите элемент массива b 1
Введите элемент массива b 2
Введите элемент массива b 5
Введите элемент массива b 7
Введите элемент массива b 9.01
Введите элемент массива b 56
z[ 0 ]= 3.10436081886
z[ 1 ]= 16.178483963
z[ 2 ]= 12.2742462158
z[ 3 ]= 1.85636472702
z[ 4 ]= 5.25358772278
z[ 5 ]= 2.0
z[ 6 ]= 8.21302700043
z[ 7 ]= 12.7527227402
z[ 8 ]= 4.3476600647
z[ 9 ]= 14.6387786865
[student@host-128 Иванов]$
```

Рисунок 15.3 – Результат выполнения программы

Рассмотрим код программы подробнее. В первой и второй строках программы импортируется содержимое двух модулей math и array соответственно.

Третья, четвертая и пятая строка – инициализация массивов a, b, z с

указанием типа элементов и их количества.

Шестая и седьмая строки – оператор for для ввода элементов массива a.

Восьмая и девятая строки – оператор for для ввода элементов массива b.

Десятая, одиннадцатая и двенадцатая строки кода – оператор for для вычисления и вывода элементов массива z.

Упражнение 2. В этом упражнении вы составите блок-схему и напишите программу для следующей задачи.

В одномерном массиве D, состоящем из 12 целых чисел, элементы с четными номерами, значения которых находятся в интервале от 3 до 6, замените на ноль. Полученный массив вывести на экран.

Решение:

1. Составьте блок-схему алгоритма (рисунок 15.4).
2. Создайте файл с именем **Lab15_2.py**
3. Откройте созданный файл и наберите текст программы (рисунок 15.5).
4. Сохраните файл и запустите программу на выполнение.
5. Сравните результат выполнения программы с рисунком 15.6.

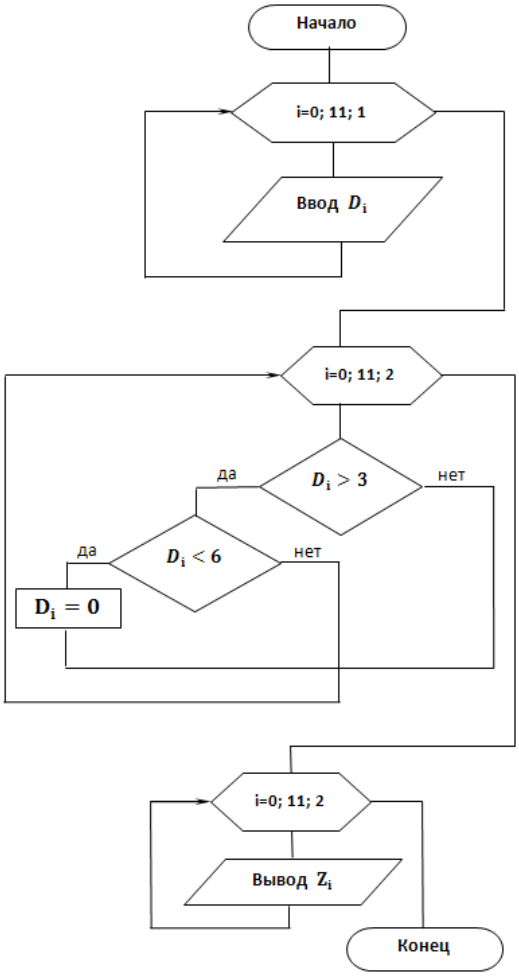


Рисунок 15.4 – Блок-схема алгоритма

```
#coding=utf-8
from math import *
from array import *
d=array("i", [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
for i in range(0, 12):
        d[i]=int(input('Введите элемент целочисленного массива d= '))
for i in range(0, 12, 2):
        if d[i]>3 and d[i]<6:
                d[i]=0
for i in range(0, 12):
        print 'd[', i, ']=', d[i]
```

Рисунок 15.5 – Текст программы

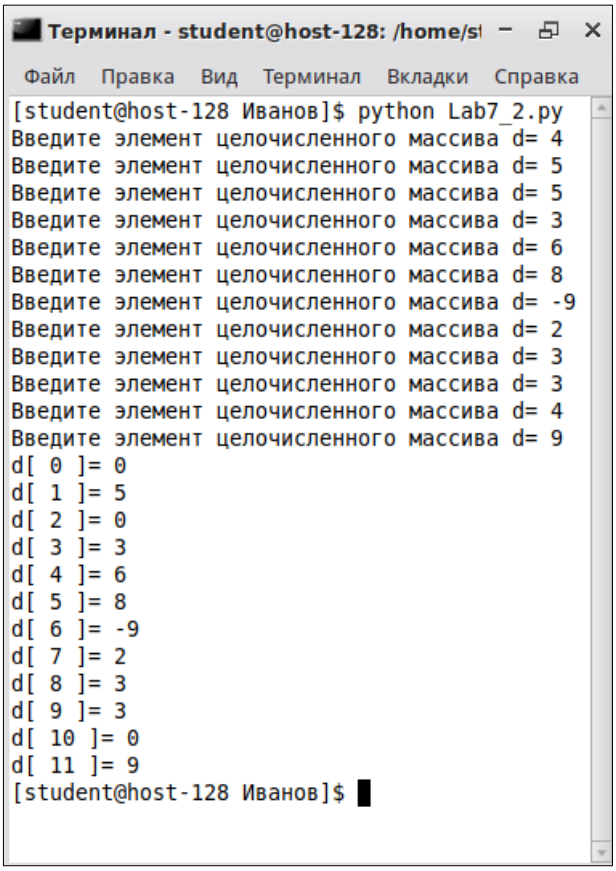


Рисунок 15.6 – Результат выполнения программы

3.4 Вычисление суммы и произведения элементов одномерного массива

В одномерных массивах вычисление происходит в цикле. Начальное значение суммы и количества равно 0, а произведения – 1. Количество элементов в массиве накапливается подобно сумме, только вместо значения элемента массива в рекуррентной формуле прибавляется 1. Накопленные величины выводятся за циклом. Рассмотрим несколько примеров.

Упражнение 3. В этом упражнении вы составите блок-схему и напишите программу для следующей задачи.

Дан массив T, состоящий из 8 элементов. Найти сумму положительных элементов массива T(8).

Решение:

1. Составьте блок-схему алгоритма (рисунок 15.7).

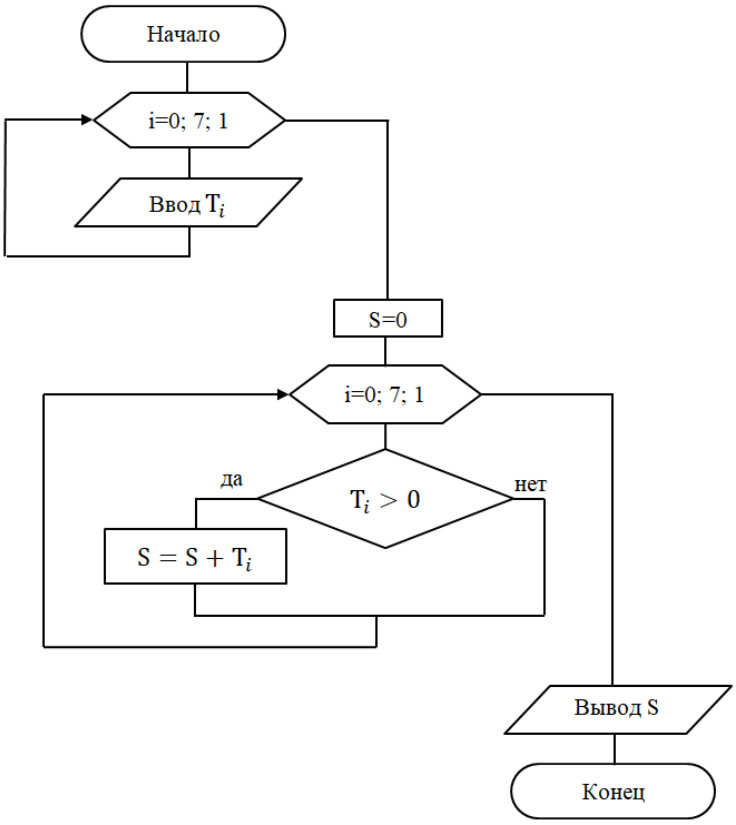


Рисунок 15.7 – Блок-схема алгоритма

2. Создайте файл с именем **Lab15_3.py**
3. Откройте созданный файл и наберите текст программы (рисунок 15.8).
4. Сохраните файл и запустите программу на выполнение.
5. Сравните результат выполнения программы с рисунком 15.9.

```
#coding=utf-8
from math import *
from array import *
T=array("f", [0, 0, 0, 0, 0, 0, 0, 0])
for i in range(0, 8):
        T[i]=float(input('Введите элемент массива T = '))
S=0
for i in range(0, 8):
        if T[i]>0:
                S+=T[i]
print "Сумма S положительных элементов массива T равна  ", S
```

Рисунок 15.8 – Текст программы

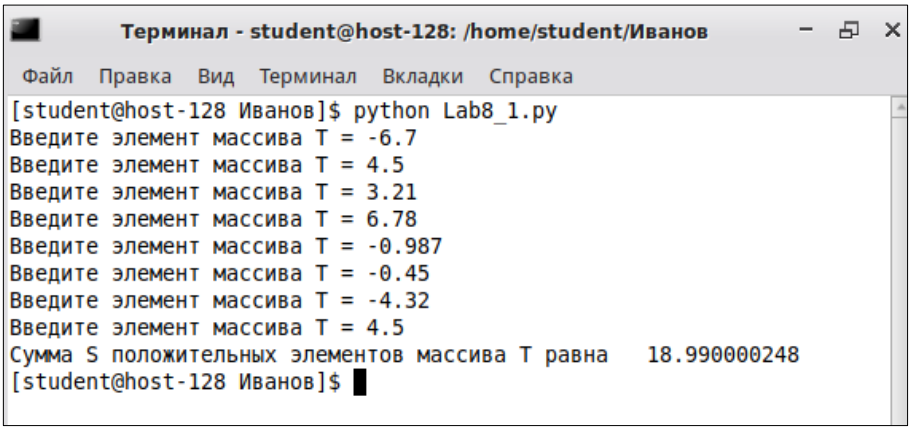


Рисунок 15.9 – Результат выполнения программы

3.5 Нахождение максимального (минимального) элемента одномерного массива

Упражнение 4. В этом упражнении вы составите блок-схему и напишите программу для следующей задачи.

Дан массив K, состоящий из 12 вещественных чисел. Найдите значение и номер максимального элемента массива K.

Решение:

1. Составьте блок-схему алгоритма (рисунок 15.10).

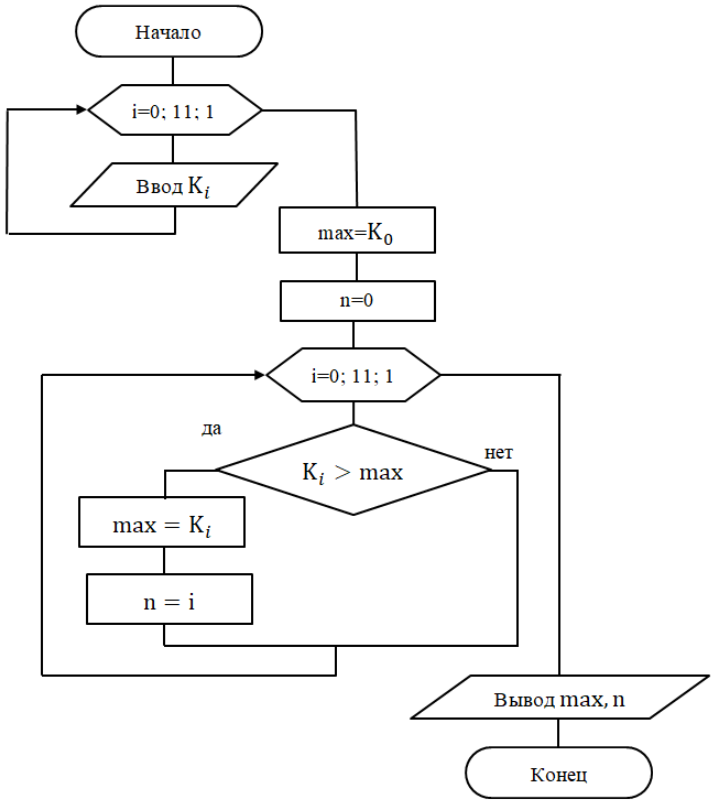


Рисунок 15.10 – Блок-схема алгоритма

2. Создайте файл с именем **Lab15_4.py**
3. Откройте созданный файл и наберите текст программы (рисунок 15.11).

```
#coding=utf-8
from math import *
from array import *
K=array("f", [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
for i in range(0, 12):
        K[i]=float(input('Введите элемент массива K = '))
max=K[0]
n=0
for i in range(0, 12):
        if K[i]>max:
                max=K[i]
                n=i
print "Максимальный элемент массива K равен  ", max, '\n', "Номер n максимального элемента массива K равен ", n
```

Рисунок 15.11 – Текст программы

4. Сохраните файл и запустите программу на выполнение.
5. Сравните результат выполнения программы с рисунком 15.12.

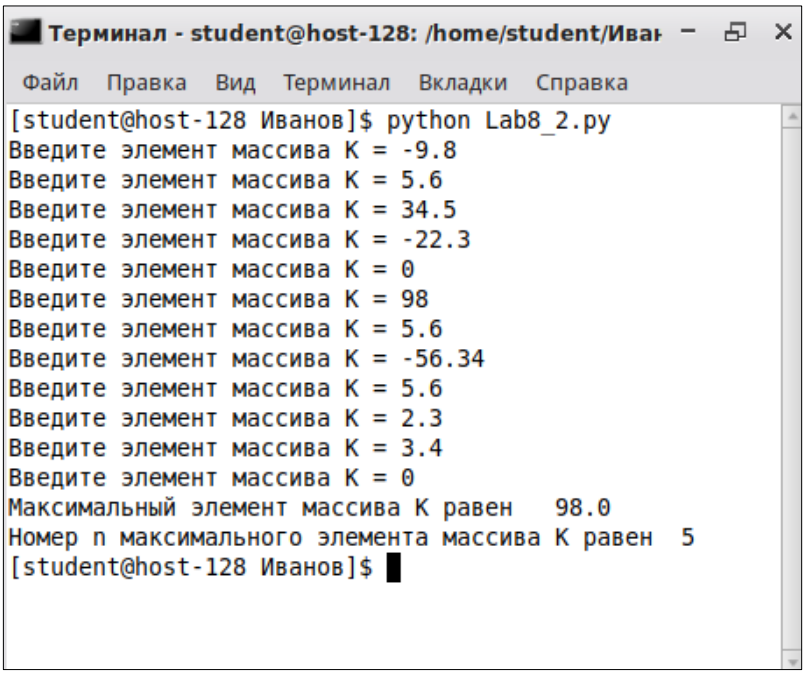


Рисунок 15.12 – Результат выполнения программы

4 СОДЕРЖАНИЕ ОТЧЕТА

Отчет должен содержать:

- тему лабораторной работы;
- цель лабораторной работы;
- номер варианта, указанный преподавателем;
- тексты задач соответствующего варианта;
- блок-схемы алгоритмов;
- тексты программы **Lab15_5.py** и **Lab15_6.py**;
- исходные данные;
- результаты выполнения программ;
- ответы на контрольные вопросы.

5 УПРАЖНЕНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

1. Выполните задание, согласно варианту, указанному преподавателем. Имя файла – **Lab15_5.py**

Задача 1.

1. Составьте блок-схему алгоритма и напишите программу для нахождения среднего арифметического элементов массива X, состоящего из 16 вещественных чисел.
2. Составьте блок-схему алгоритма и напишите программу. Дан массив A, состоящий из 10 вещественных чисел. Найдите количество отрицательных элементов массива A.
3. Составьте блок-схему алгоритма и напишите программу. Дан массив C, состоящий из 15 вещественных чисел. Найдите количество нулевых элементов массива C.
4. Составьте блок-схему алгоритма и напишите программу. Дан массив B, состоящий из 12 вещественных чисел. Найдите сумму положительных элементов массива B.
5. Составьте блок-схему алгоритма и напишите программу. Дан массив D, состоящий из 20 вещественных чисел. Найдите произведение элементов массива D, значение которых меньше заданного числа n.
6. Составьте блок-схему алгоритма и напишите программу. Дан одномерный массив вещественных чисел, состоящий из 20 элементов. Вывести на экран произведение положительных элементов массива.
7. Составьте блок-схему алгоритма и напишите программу для нахождения максимального элемента массива V(14) и его индекса.
8. Составьте блок-схему алгоритма и напишите программу для нахождения минимального элемента массива Z(10) и его индекса.
9. Составьте блок-схему алгоритма и напишите программу для подсчета числа элементов, равных максимальному в числовом массиве из 9 элементов.
10. Составьте блок-схему алгоритма и напишите программу для вычисления разности между максимальным и минимальным значениями элементов заданного целочисленного массива, состоящего из 11 вещественных чисел.
11. Составьте блок-схему алгоритма и напишите программу для нахождения в массиве C, состоящего из 7 вещественных чисел, минимального из элементов, имеющих четный индекс.
12. Составьте блок-схему алгоритма и напишите программу для нахождения наименьшего положительного элемента целочисленного массива A(29).

2. Выполните задание, согласно варианту, указанному преподавателем. Имя файла – **Lab15_6.py**

Задача 2.

1. Составьте блок-схему алгоритма и напишите программу, в результате выполнения которой были бы напечатаны номера всех нулевых элементов массива C(11).
2. Составьте блок-схему алгоритма и напишите программу, в результате выполнения которой были бы напечатаны номера всех положительных элементов массива M(9).
3. Составьте блок-схему алгоритма и напишите программу. В одномерном массиве F(10) отрицательные элементы заменить нулями, а все положительные увеличить на 2.
4. Составьте блок-схему алгоритма и напишите программу. В одномерном массиве S(11) замените на единицы все элементы, значения которых больше 5.
5. Составьте блок-схему алгоритма и напишите программу. В одномерном массиве K(10) элементы с нечетными номерами необходимо уменьшить в четыре раза. Полученный массив вывести на экран.
6. Составьте блок-схему алгоритма и напишите программу. В одномерном массиве S(11) элементы с четными номерами разделите на их порядковый номер. Полученный массив вывести на экран.
7. Составьте блок-схему алгоритма и напишите программу. В одномерном массиве A(9) элементы с четными номерами необходимо уменьшить в два раза, а элементы с нечетными номерами увеличить в три раза. Полученный массив вывести на экран.
8. Составьте блок-схему алгоритма и напишите программу. Дан массив R, состоящий из 8 вещественных чисел. Упорядочить по возрастанию элементы массива R с нечетными индексами.
9. Составьте блок-схему алгоритма и напишите программу замены всех нулевых элементов массива F(12) единицами.
10. Составьте блок-схему алгоритма и напишите программу замены всех отрицательных элементов массива K(12) их квадратами.
11. Составьте блок-схему алгоритма и напишите программу. В одномерном массиве M(13) отрицательные элементы с нечетными номерами необходимо возвести в квадрат. Полученный массив вывести на экран.
12. Составьте блок-схему алгоритма и напишите программу для вычисления значений функции Z:

$$Z = \begin{cases} \operatorname{ctg}^2\left(x_i - \dfrac{a}{b}\right), & \text{если } x_i < a \\ 1 - (x_i + a^3)^2, & \text{если } a \le x_i < b, \\ {x_i}^2 + \dfrac{1}{ab}, & \text{если } x_i \ge b \end{cases}$$

где $a = 4,\ b = 6{,}4,\ x(10) -$ массив целых чисел.

6 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что называется массивом?
2. Что такое размерность и размер массива?
3. Как обратиться к элементу массива?
4. С какого значения начинается нумерация элементов массива в Python?
5. Какой модуль необходимо использовать для работы с массивами?
6. Какой командой импортируется содержимое модуля для работы с массивами?

ЛАБОРАТОРНАЯ РАБОТА № 16

КОНТРОЛЬНАЯ РАБОТА № 4
«ТИПОВЫЕ АЛГОРИТМЫ ПОИСКА ИНФОРМАЦИИ ДЛЯ РЕШЕНИЯ ТИПОВЫХ ЗАДАЧ В ОБЛАСТИ АГРОИНЖЕНЕРИИ»

Примерный вариант

Б И Л Е Т № 1

1. Составьте блок-схему алгоритма и напишите программу для вычисления f по формуле

$$f = \sum_{x=1}^{5} 2\sin(3x - y), \quad \Delta x = 0{,}4, \quad y = 5{,}1.$$

2. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции m:

$$m = \begin{cases} \sqrt{|n + cos(a)|}, & \text{если } n < 3 \\ e^{n-1{,}7}, & \text{если } 3 \le n \le 6, \\ a + \ln(n+1)^2, & \text{если } n > 6 \end{cases}$$

где $a = tgn,\ n \in [2; 7],\ \Delta n = 1.$

3. Составьте блок-схему алгоритма и напишите программу. Даны два массива данных: массив S – посевная площадь озимой пшеницы в 8-ми районах Ростовской области за 2024 год, массив U – урожайность озимой пшеницы в 8-ми районах Ростовской области за 2024 год. Вычислить: валовый сбор озимой пшеницы для каждого из 8-ми районов Ростовской области и максимальный валовый сбор.

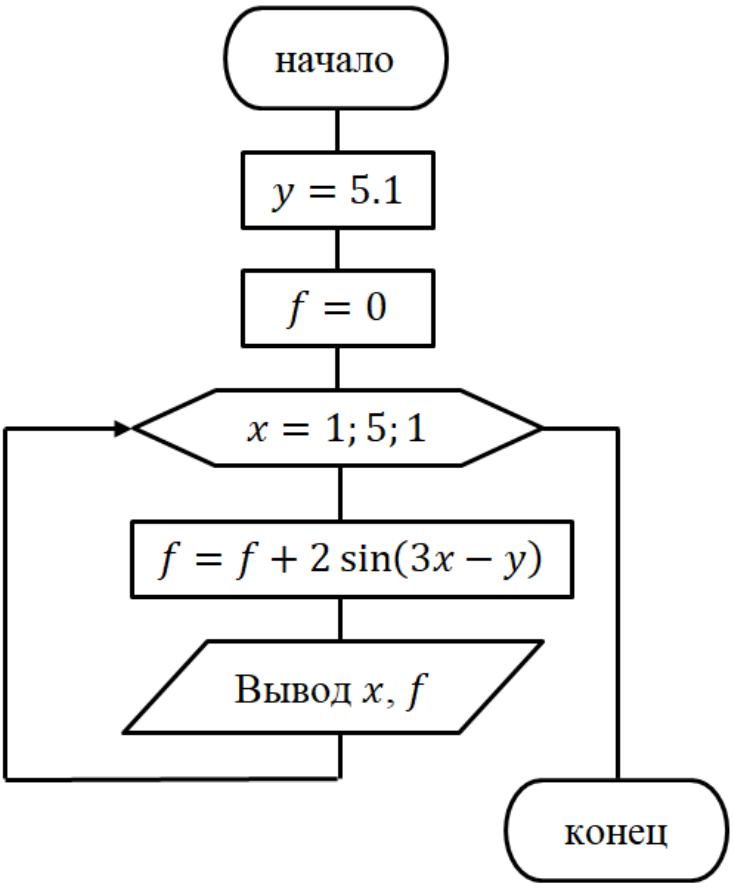
Результат выполнения контрольной работы № 4

1. Составьте блок-схему алгоритма и напишите программу для вычисления f по формуле

$$f = \sum_{x=1}^{5} 2\sin(3x - y), \quad \Delta x = 0{,}4, \quad y = 5{,}1.$$

Решение:

1) Составим блок-схему алгоритма:



2) По блок-схеме алгоритма составим программу:

```
#coding=utf-8
from math import *
y=5.1
f=0
for x in range(1, 6):
        f+=2*sin(3*x-y)
        print 'x =', x, 'f =', f
```

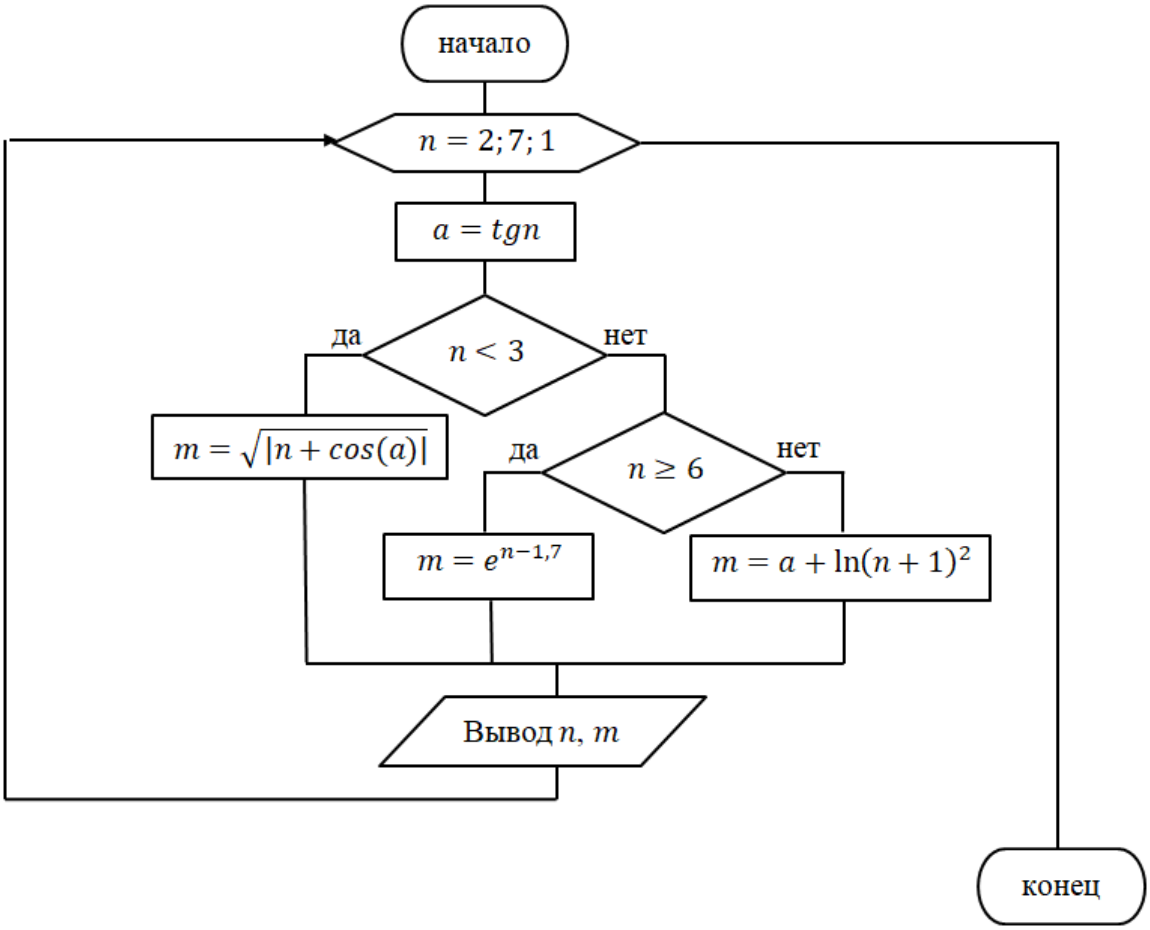
2. Составьте блок-схему алгоритма и напишите программу. Вычислить значение функции m:

$$m = \begin{cases} \sqrt{|n + cos(a)|}, & \text{если } n < 3 \\ e^{n-1,7}, & \text{если } 3 \le n \le 6, \\ a + \ln(n+1)^2, & \text{если } n > 6 \end{cases}$$

где $a = tgn,\ n \in [2;7],\ \Delta n = 1.$

Решение:

1) Составим блок-схему алгоритма:



2) По блок-схеме алгоритма составим программу:

```
#coding=utf-8
from math import *
for n in range(2, 8):
        a=tan(n)
        if n<3:
                m=sqrt(abs(n+cos(a)))
```

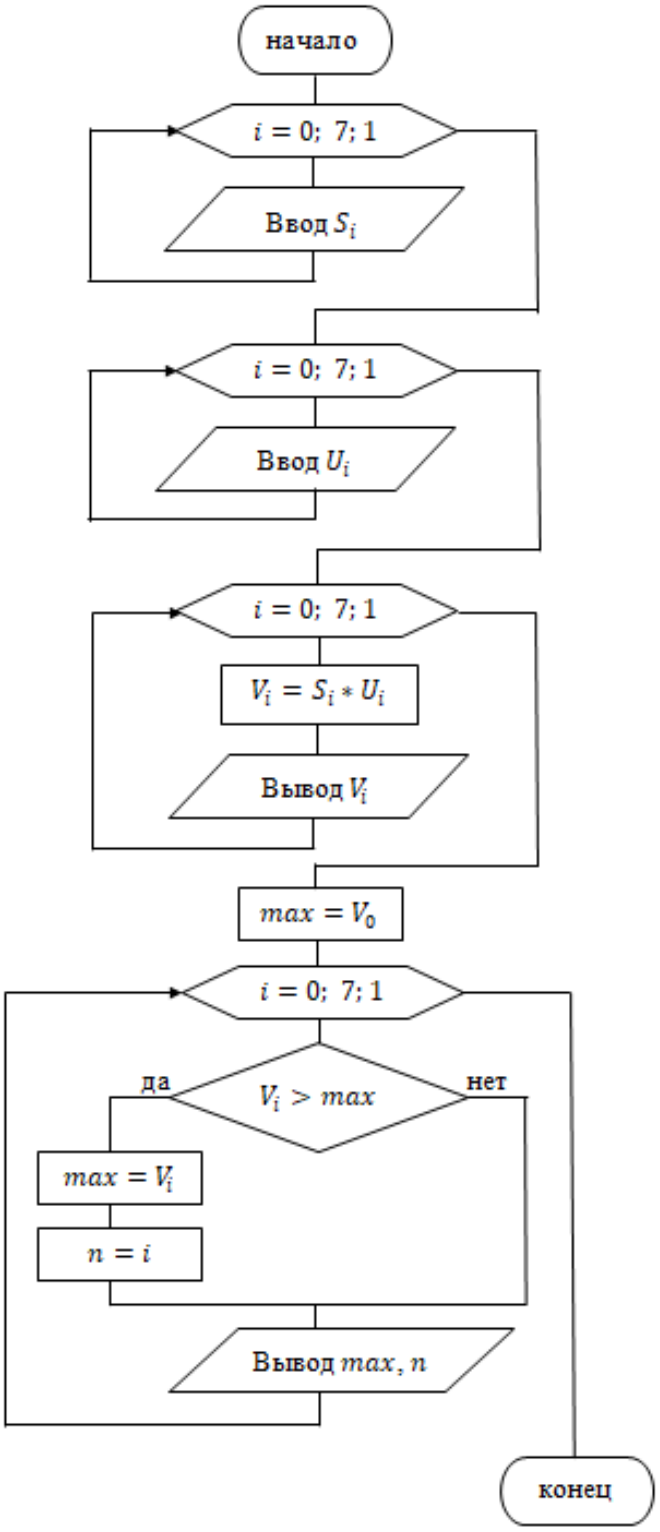
```
elif n>=3 and n<=6:
        m=exp(n-1.7)
else:
        m=a+log(pow((n+1),2))
print 'n =', n, 'm =', m
```

3. Составьте блок-схему алгоритма и напишите программу. Даны два массива данных: массив S – посевная площадь озимой пшеницы в 8-ми районах Ростовской области за 2024 год, массив U – урожайность озимой пшеницы в 8-ми районах Ростовской области за 2024 год. Вычислить: валовый сбор озимой пшеницы для каждого из 8-ми районов Ростовской области и максимальный валовый сбор.

Решение:

1) Составим блок-схему алгоритма (см. рисунок ниже).
2) По блок-схеме алгоритма составим программу:

```
#coding=utf-8
from math import *
from array import *
S=array("f", [0, 0, 0, 0, 0, 0, 0, 0])
U=array("f", [0, 0, 0, 0, 0, 0, 0, 0])
V=array("f", [0, 0, 0, 0, 0, 0, 0, 0])
for i in range(0, 8):
        S[i]=float(input('S = '))
for i in range(0, 8):
        U[i]=float(input('U = '))
for i in range(0, 8):
        V[i]=S[i]*U[i]
        print 'V[ ', i, ' ]=', V[i]
max=V[0]
for i in range(0, 8):
        if V[i]>max:
                max=V[i]
                n=i
print "Максимальный элемент ", max, '\n', "Номер максимального элемента массива V равен ", n
```


начало
$i = 0;\ 7;\ 1$
Ввод S_i
$i = 0;\ 7;\ 1$
Ввод U_i
$i = 0;\ 7;\ 1$
$V_i = S_i * U_i$
Вывод V_i
$max = V_0$
$i = 0;\ 7;\ 1$
да
$V_i > max$
нет
$max = V_i$
$n = i$
Вывод max, n
конец

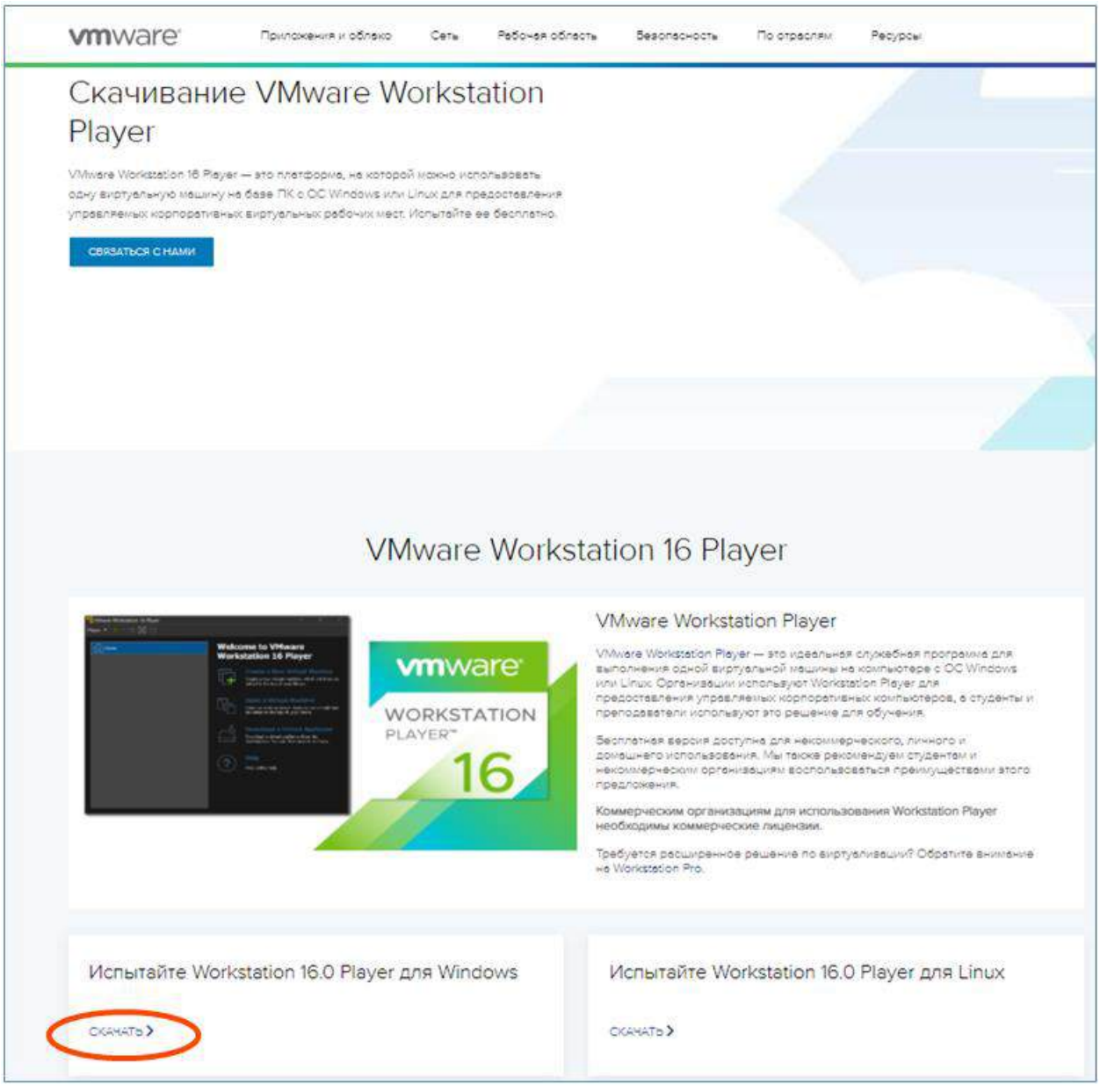
ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ А

Установка и настройка виртуальной машины VMNware Workstation Player

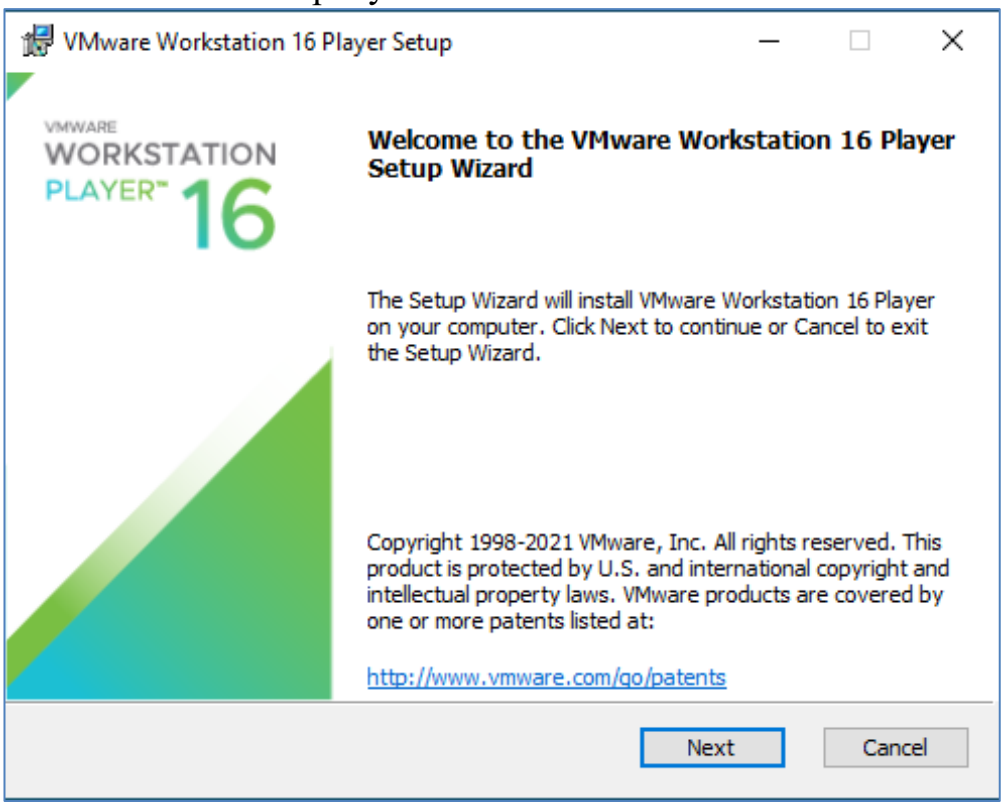
1. Скачайте установочный файл по адресу

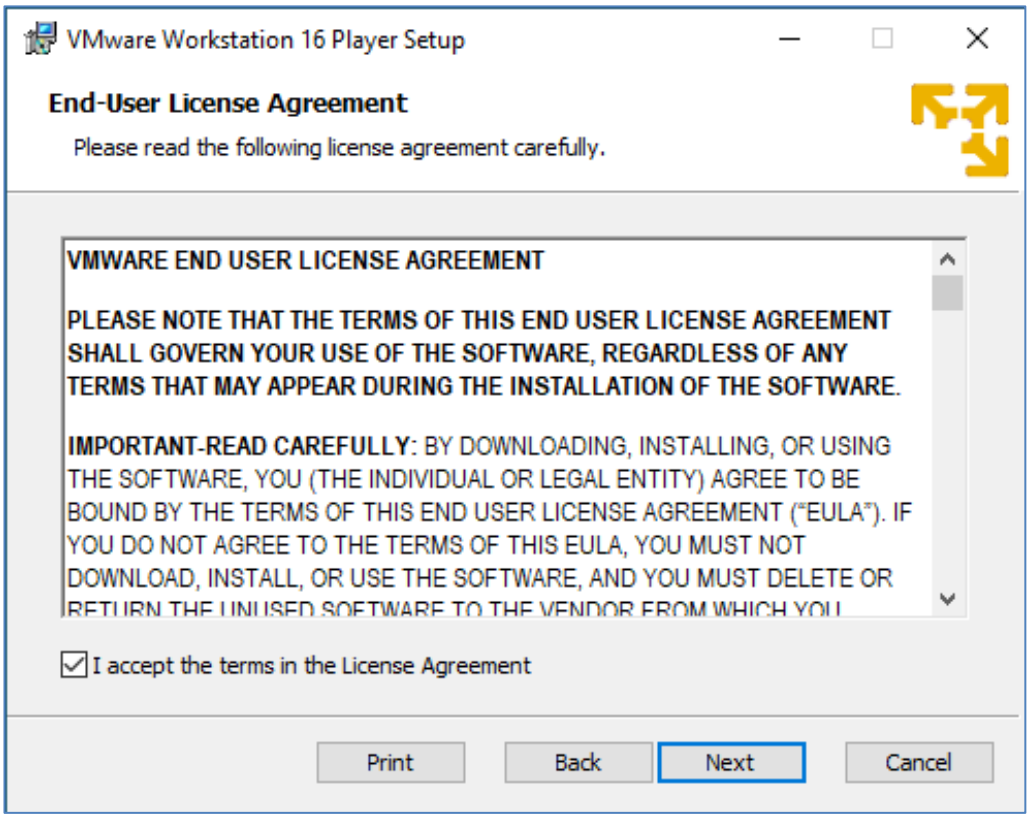
https://www.vmware.com/ru/products/workstation-player/workstation-player-evaluation.html

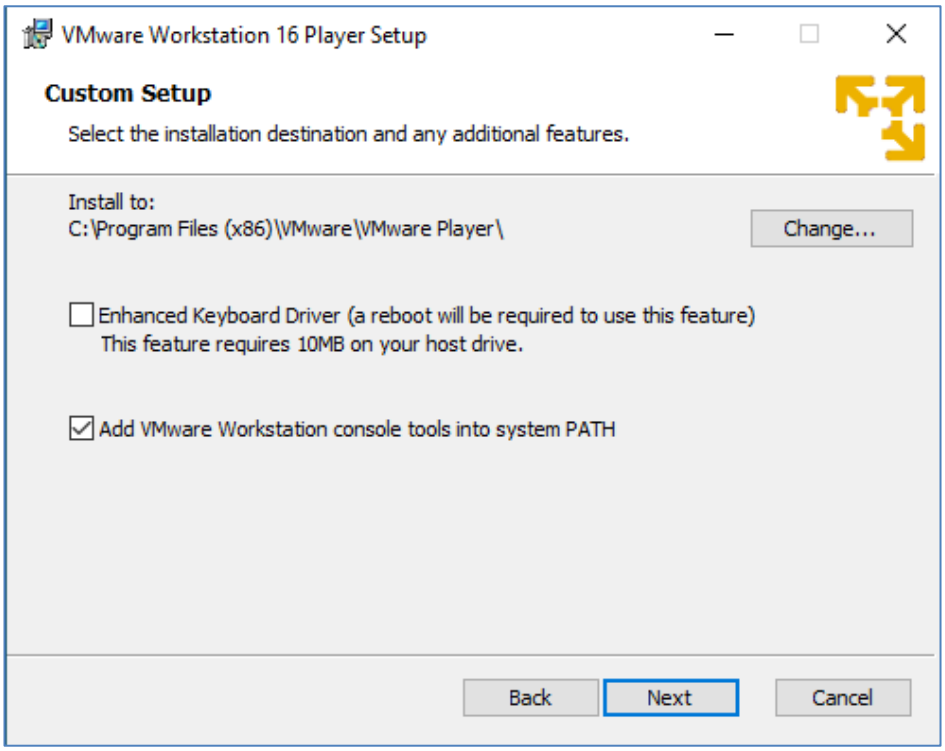


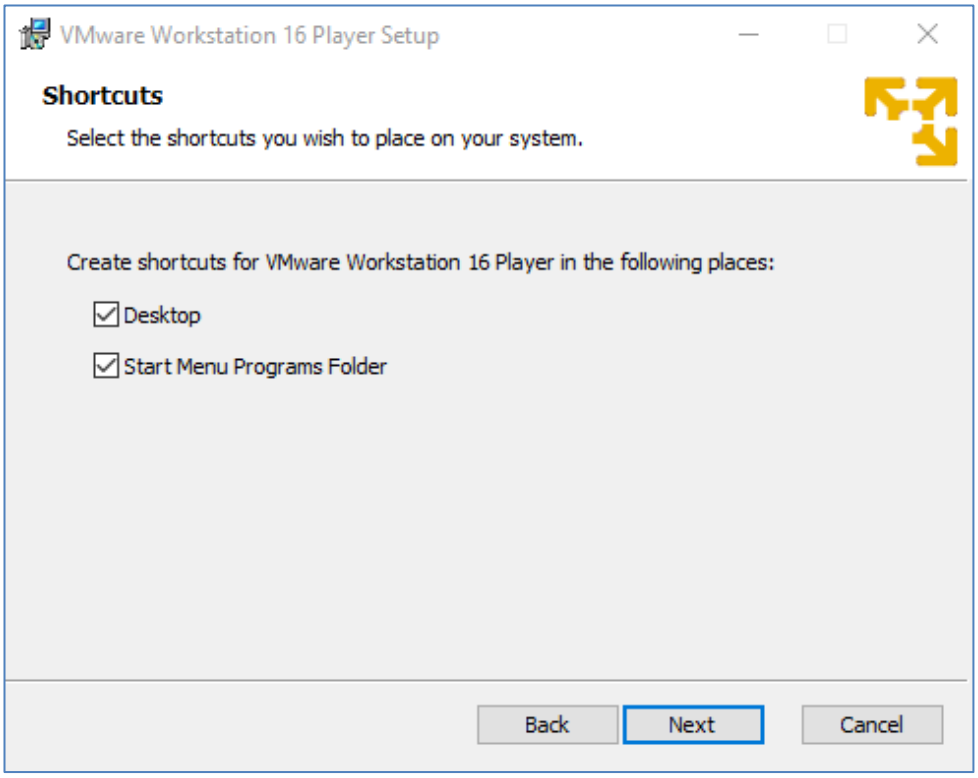
2. Щелкаем по ссылке **Скачать** в разделе **Испытайте Workstation 16.0 Player для Windows**, в появившемся диалоговом окне указываем папку, в которую необходимо скачать установочный файл и щелкаем по кнопке **Сохранить**.

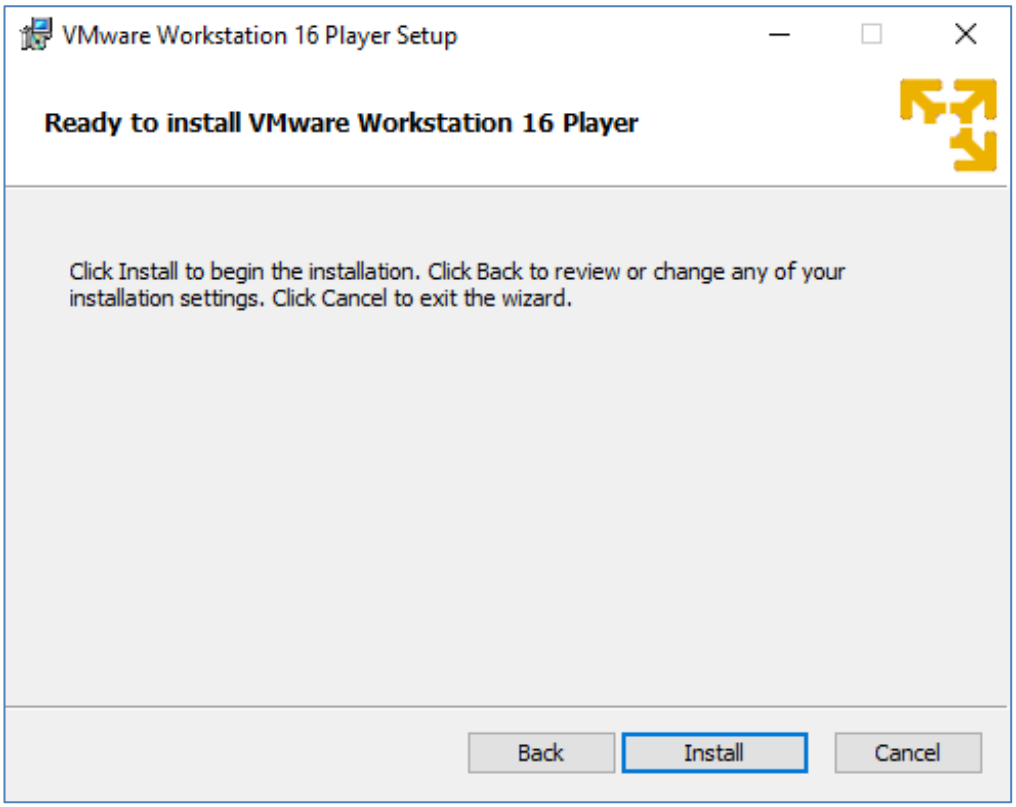
3. Находим сохраненный файл и запускаем на установку. Выполняем настройки как показано на рисунках ниже.

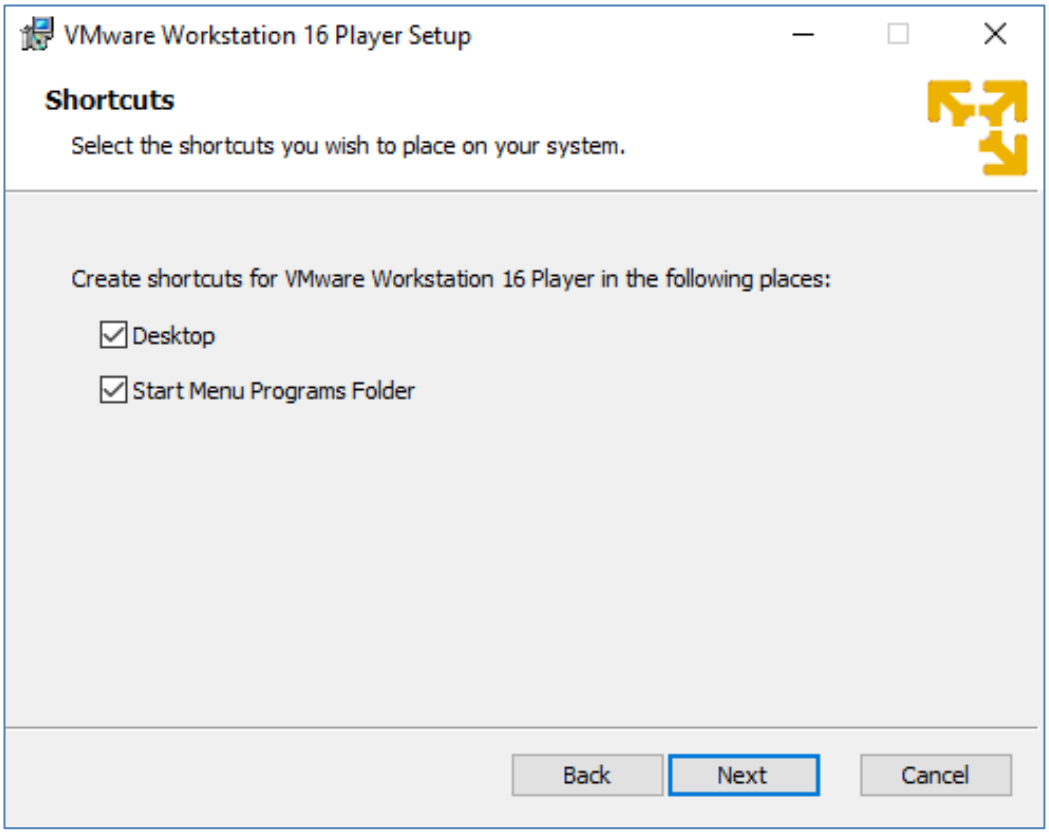


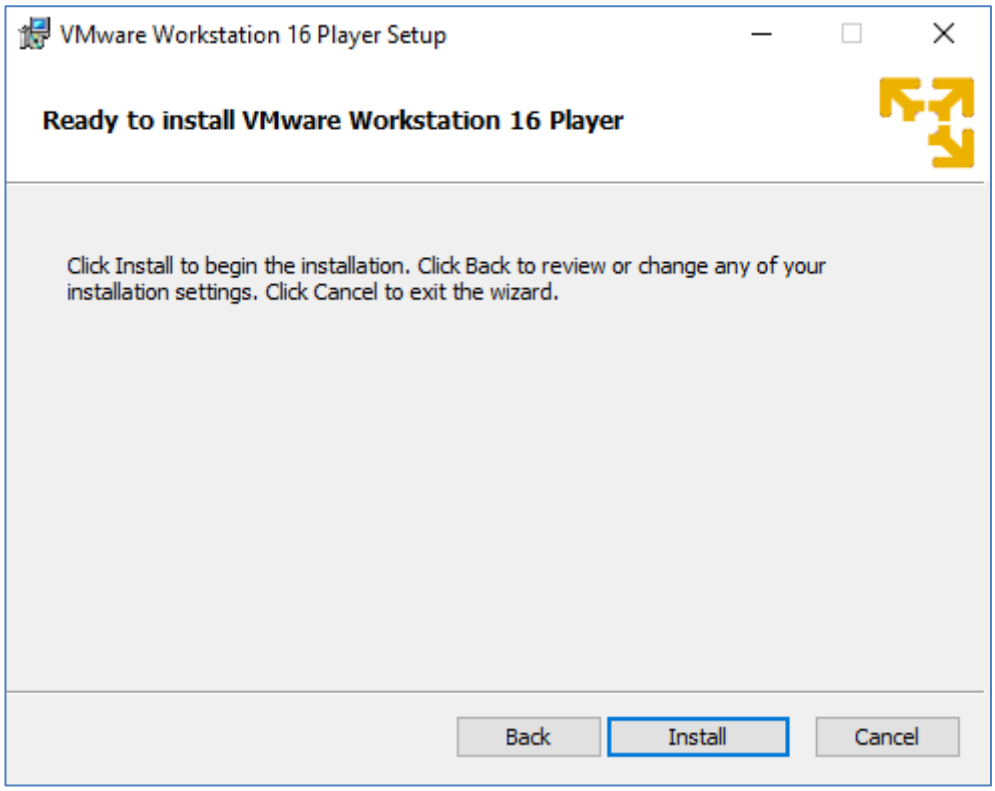


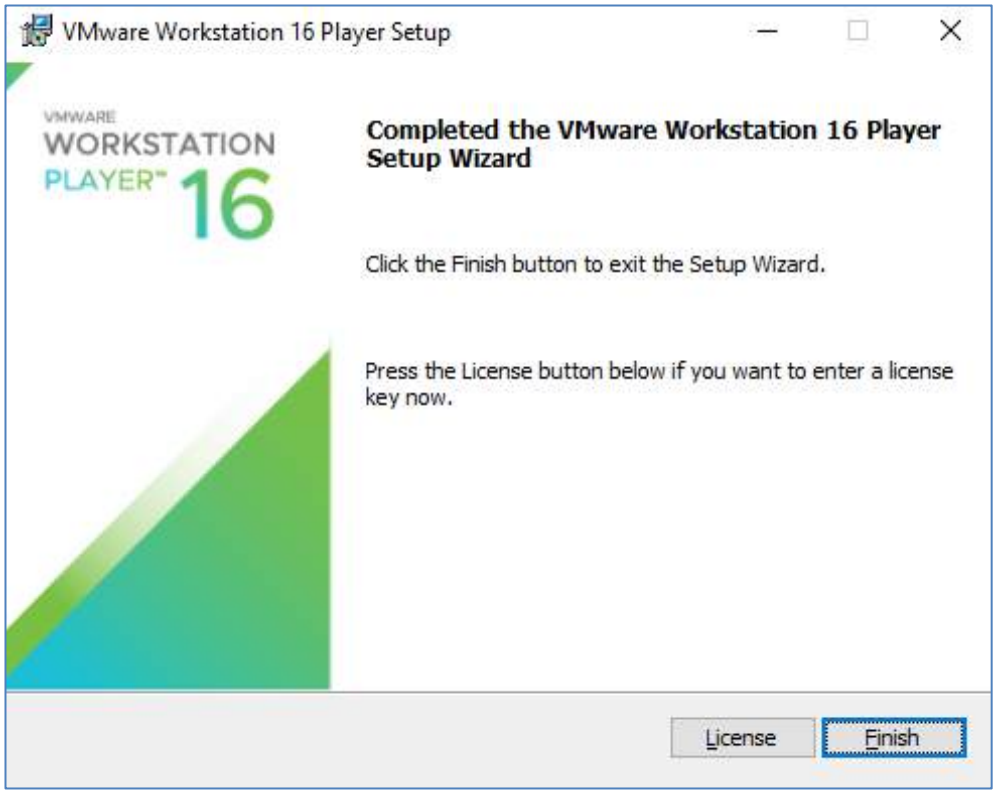

VMware Workstation 16 Player Setup
Custom Setup
Select the installation destination and any additional features.
Install to:
C:\Program Files (x86)\VMware\VMware Player\
Change...
Enhanced Keyboard Driver (a reboot will be required to use this feature)
This feature requires 10MB on your host drive.
Add VMware Workstation console tools into system PATH
Back
Next
Cancel


VMware Workstation 16 Player Setup
Shortcuts
Select the shortcuts you wish to place on your system.
Create shortcuts for VMware Workstation 16 Player in the following places:
Desktop
Start Menu Programs Folder
Back
Next
Cancel


VMware Workstation 16 Player Setup
Ready to install VMware Workstation 16 Player
Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.
Back
Install
Cancel


VMware Workstation 16 Player Setup
Shortcuts
Select the shortcuts you wish to place on your system.
Create shortcuts for VMware Workstation 16 Player in the following places:
Desktop
Start Menu Programs Folder
Back
Next
Cancel

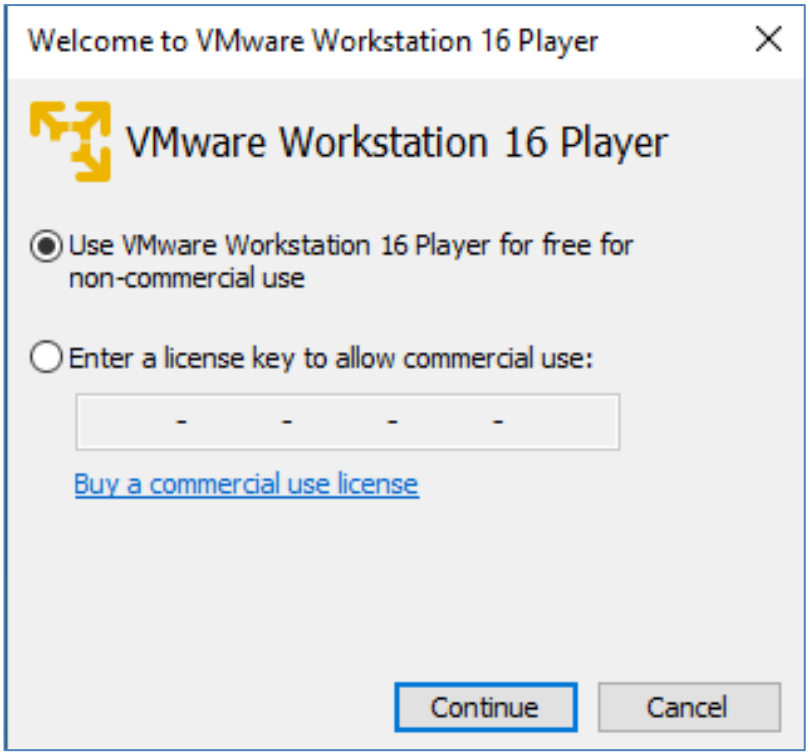


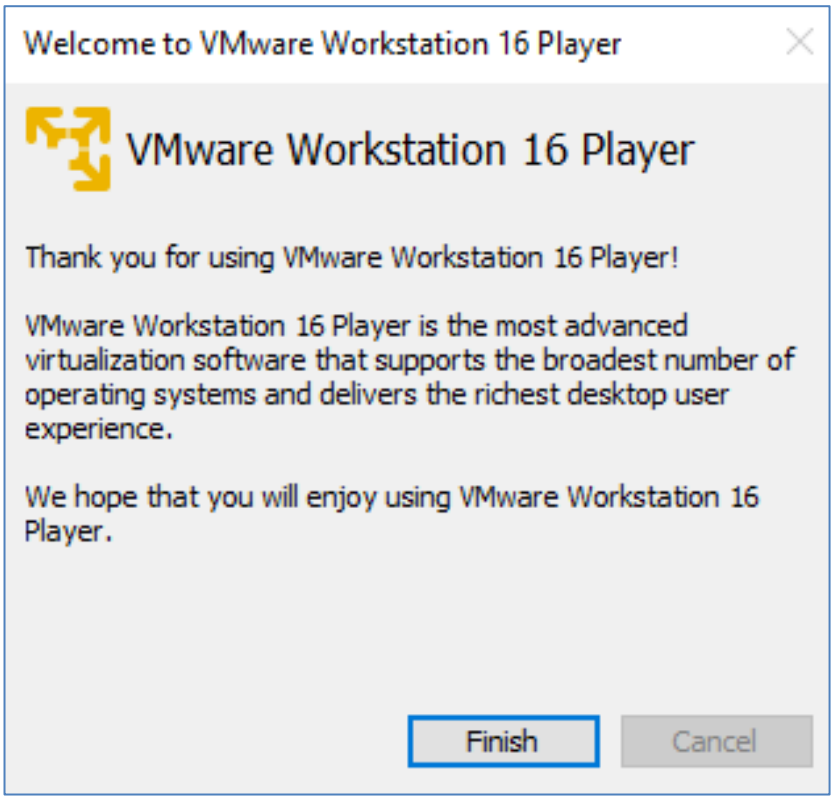


После установки в меню Пуск и на рабочем столе появится ярлык установленной программы.

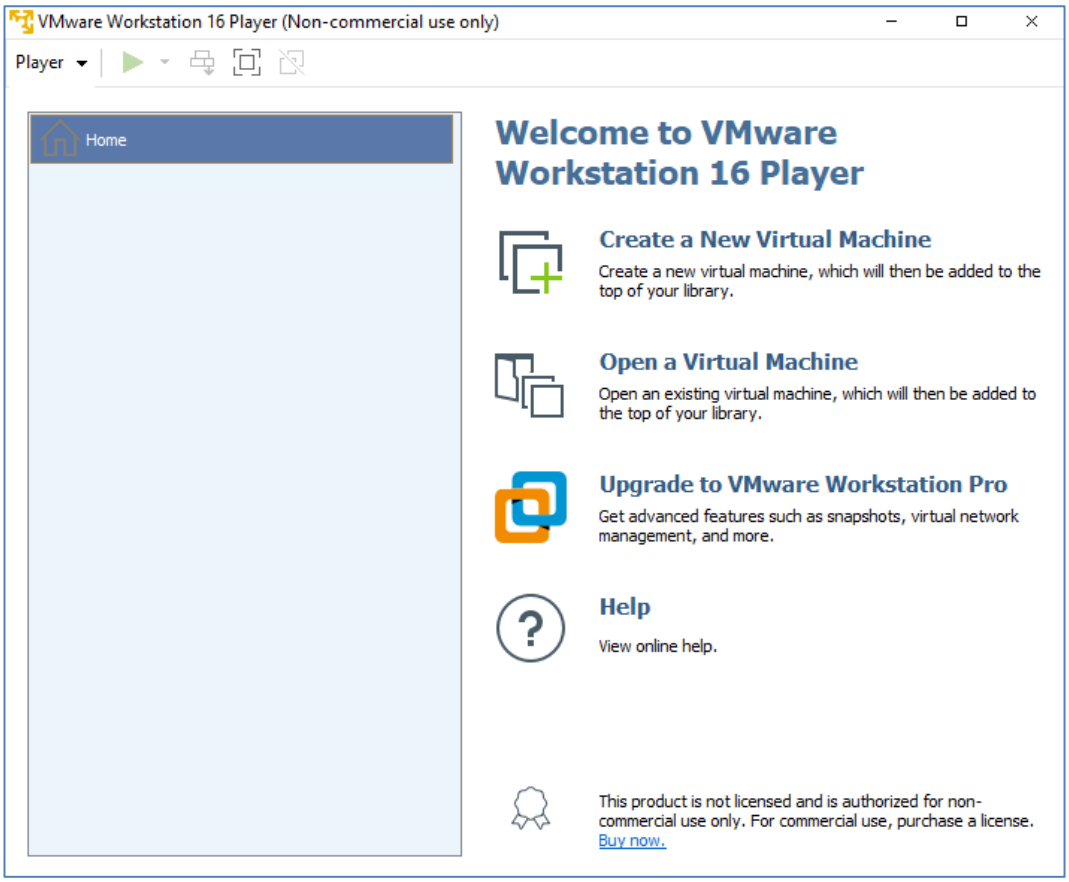


4. Запустите виртуальную машину VMNware Workstation Player. Выполните настройки виртуальной машины как показано на рисунках ниже.





5. На экране появится окно:



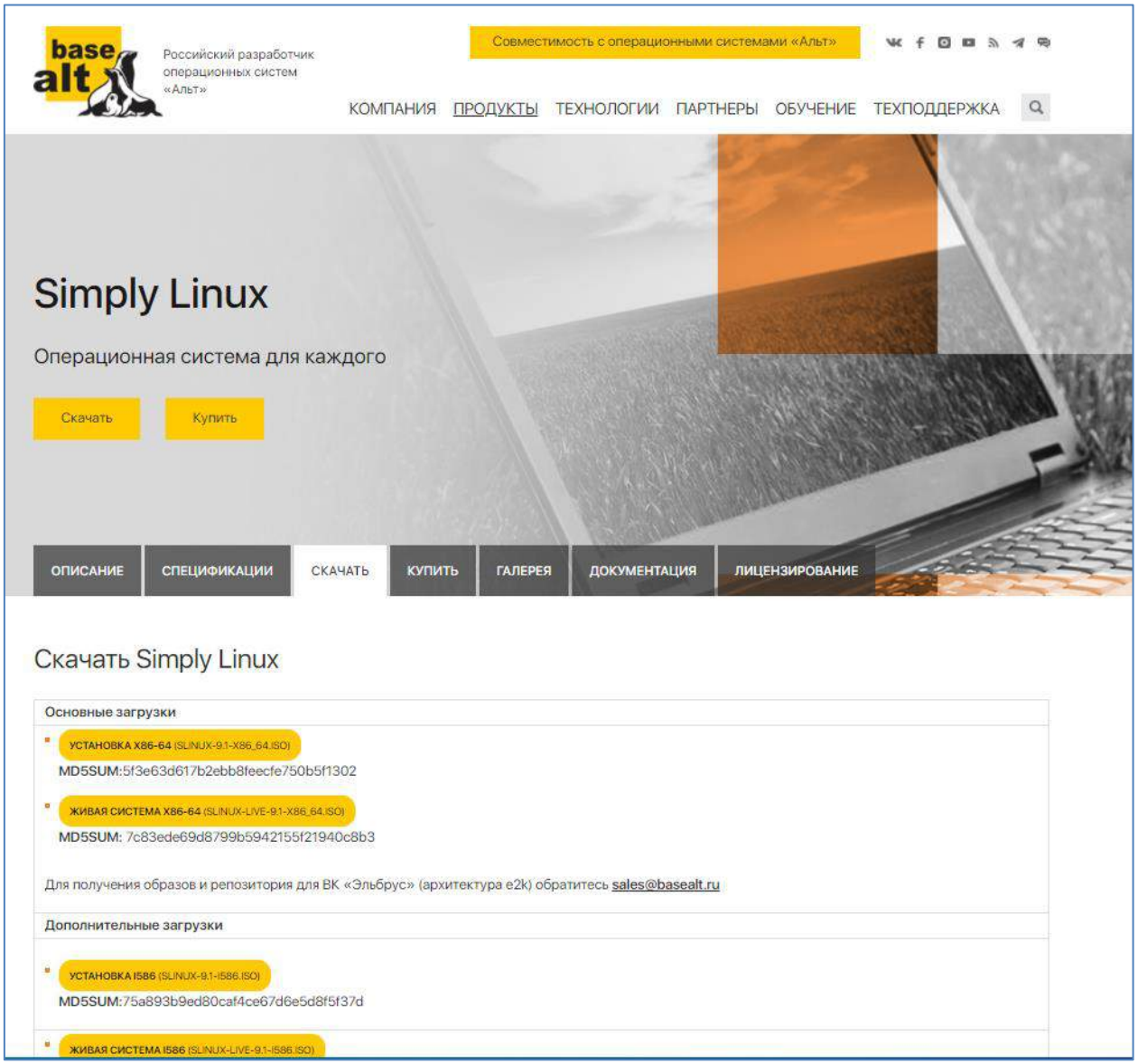
Виртуальная машина готова к работе.

ПРИЛОЖЕНИЕ Б

Установка и настройка ОС Simply Linux **на виртуальной машине VirtualBox**

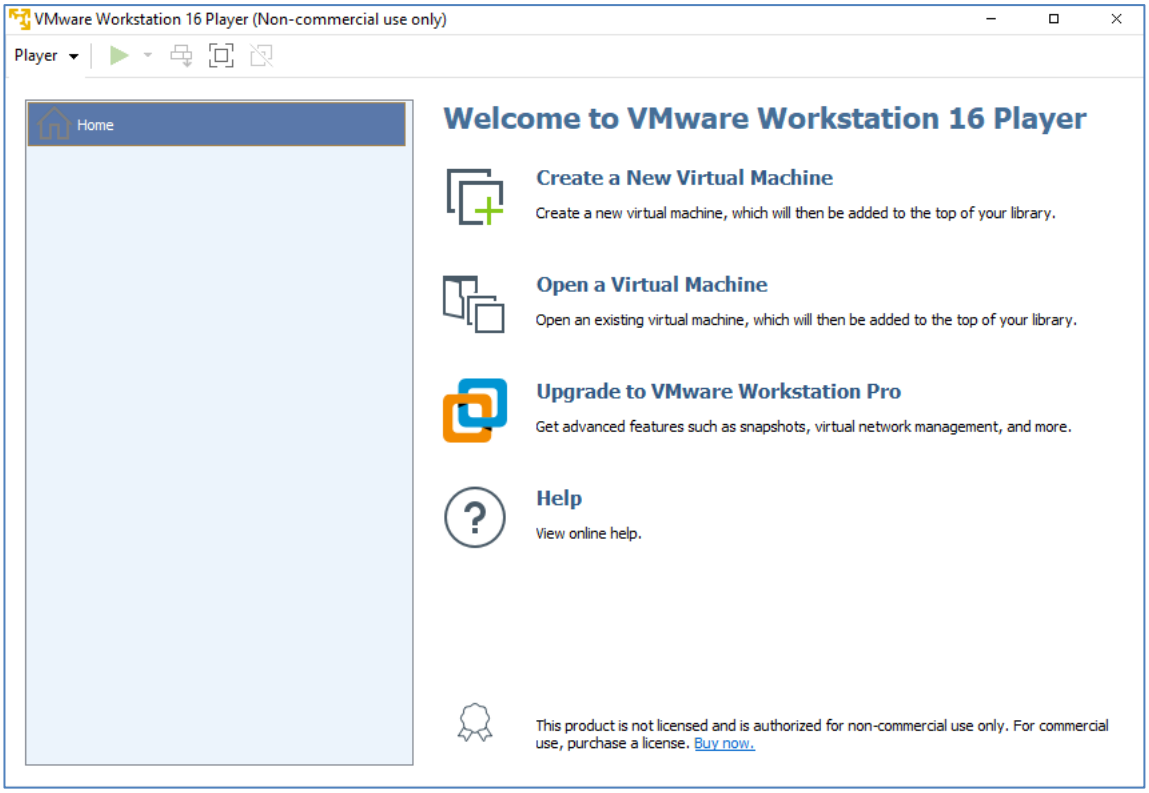
1. Скачайте установочный файл ОС Simply Linux по адресу

https://www.basealt.ru/products/skachat-1

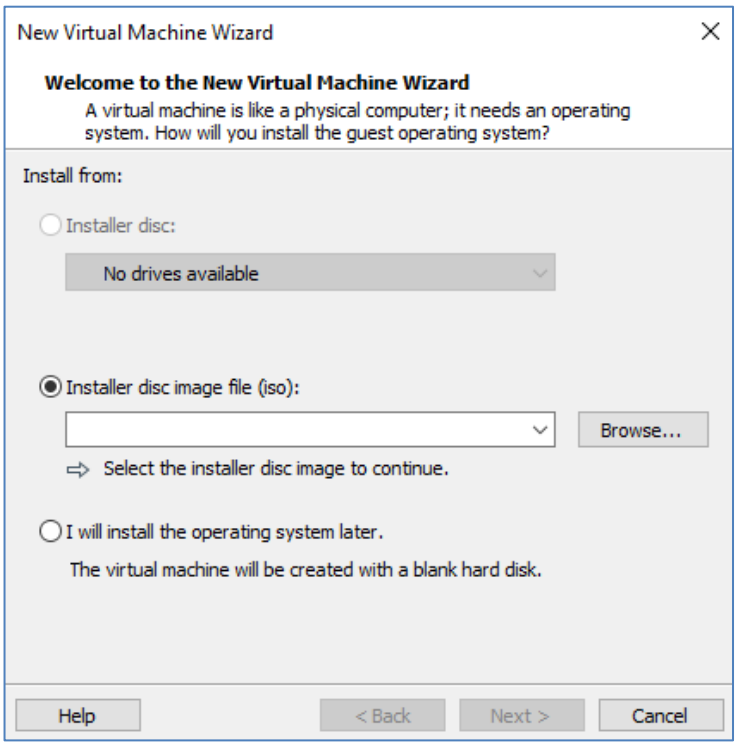


Выберите для установки ту версию **Simply Linux**, которая подходит для вышего компьютера.

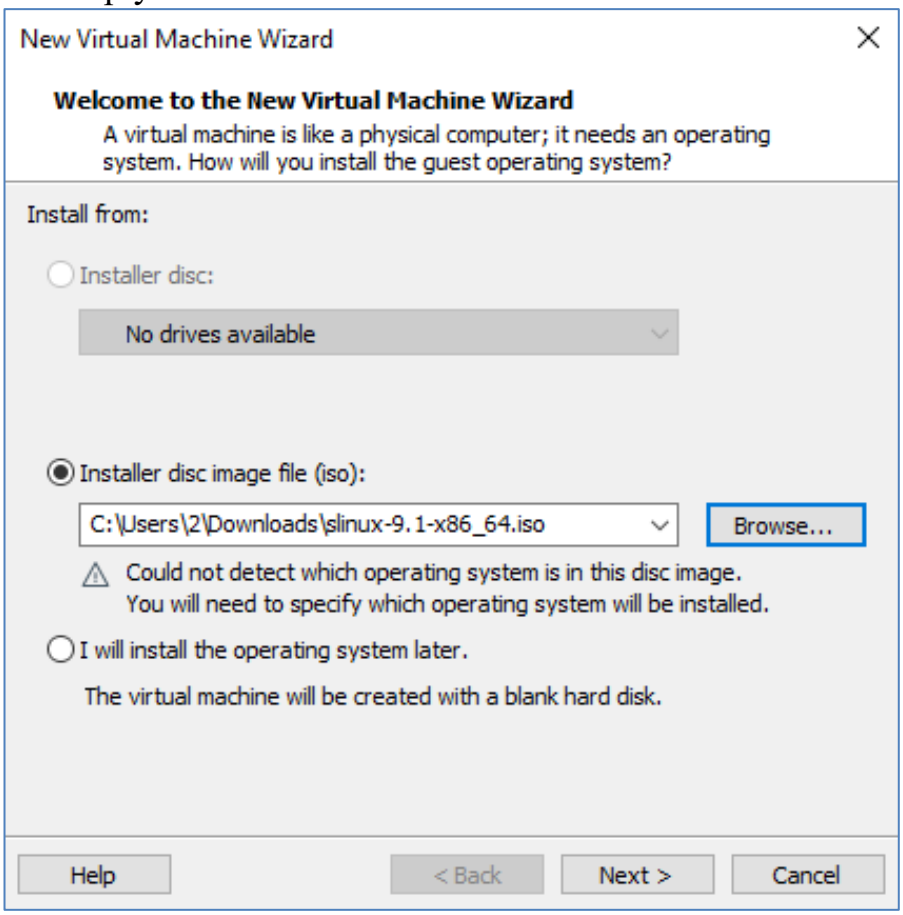
2. Запустите виртуальную машину.



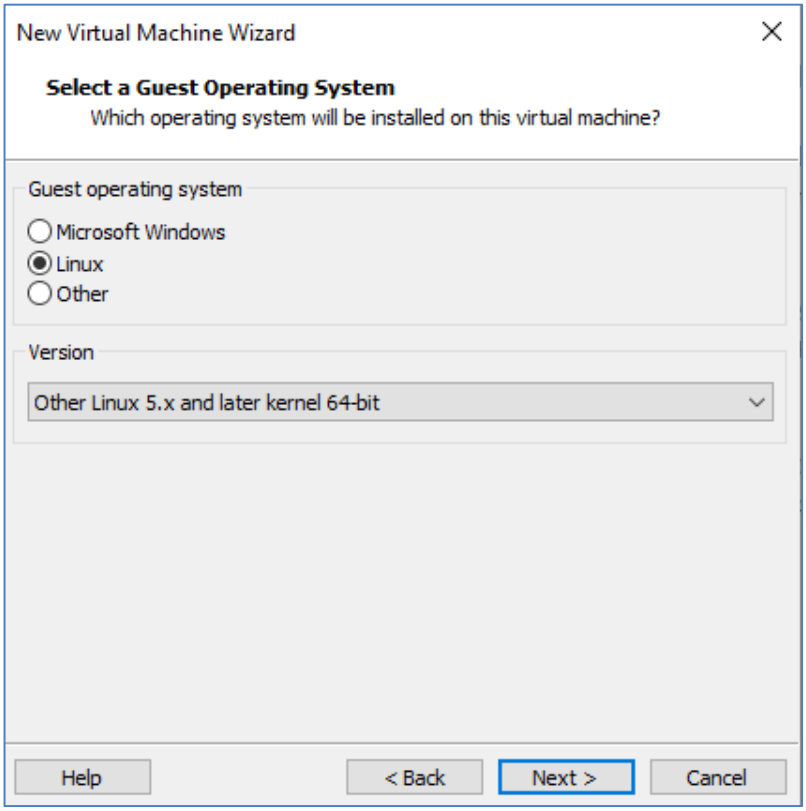
3. В окне выберите команду Create a New Virtual Machine. Появится диалоговое окно



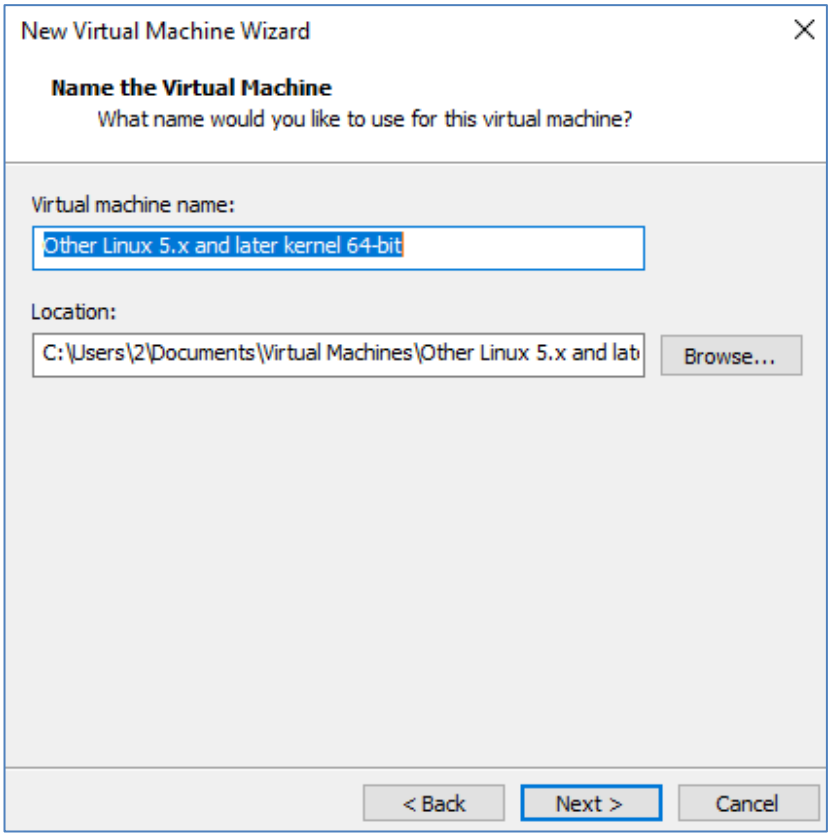
4. В поле ввода **Installer disk image file (iso):** укажите путь к установочному файлу ОС Simply Linux:

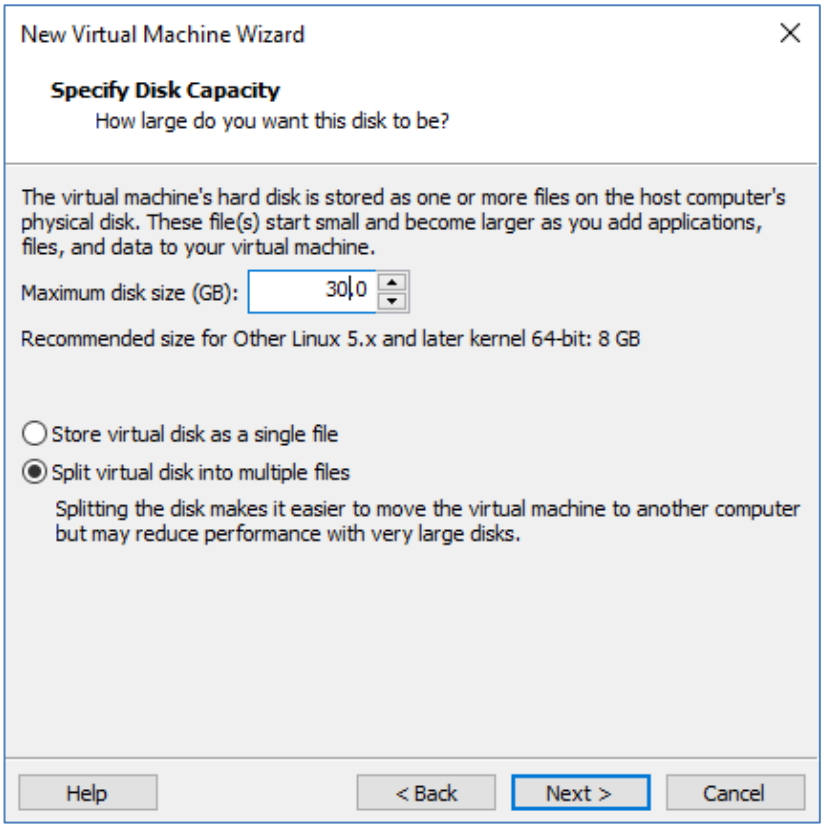


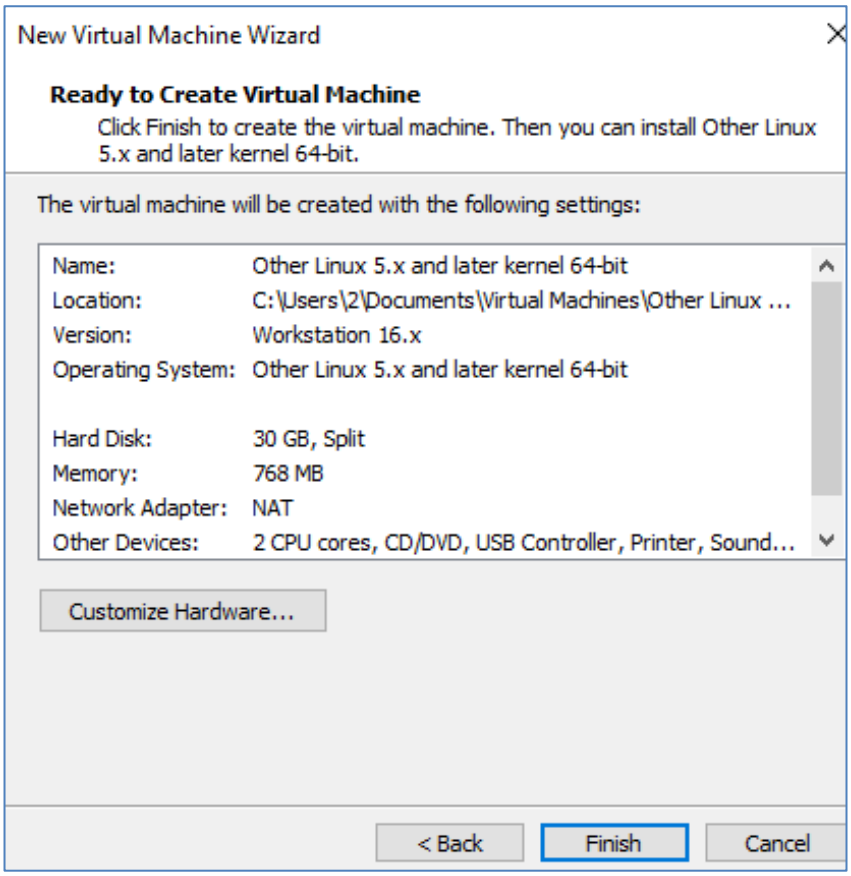
5. В следующем диалоговом окне установите переключатель на Linux:



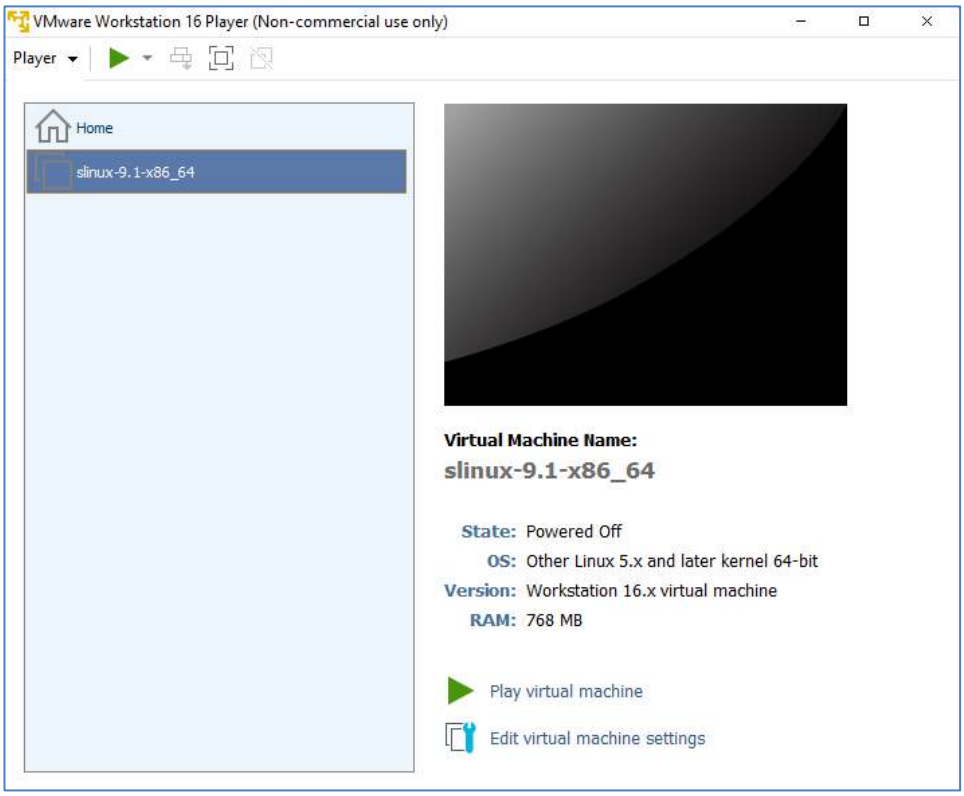
6. Выполните настройки как показано на рисунках ниже.



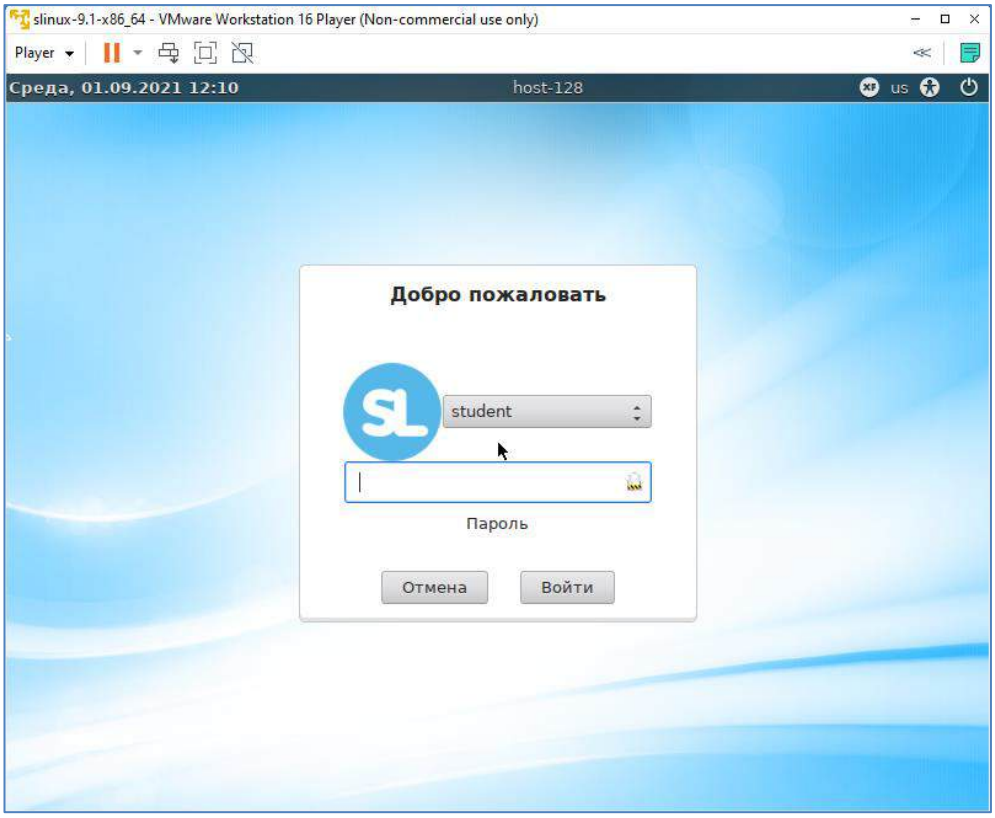




7. В окне виртуальной машины появилась строка, выбрав которую можно запустить ОС Simply Linux. Выберите ее и щелкните по кнопке .



В результате на экране в окне виртуальной машины появится окно с запросом пароля для входа в ОС Simply Linux.



ПРИЛОЖЕНИЕ В

В таблице приведены стандартные каталоги и их содержимое

Каталог	ОС	Содержимое
/bin	Все	Команды операционной системы ядра
/boot	LS	Ядро и файлы для его загрузки
/dev	Все	Файлы устройств: дисков, принтеров, псевдотерминалов и т.д.
/etc	Все	Важные файлы запуска и конфигурации системы
/home	Все	Стандартные домашние каталоги пользователей
/lib	Все	Библиотеки, совместно используемые библиотеки и компоненты компилятора языка С
/media	LS	Точки монтирования файловых системы на съемных носителях
/mnt	LSA	Временные точки монтирования
/opt	Все	Программные пакеты необязательных приложения (которые пока не находят широкого применения)
/proc	LSA	Информация о всех выполняющихся процессах
/root	LS	Домашний каталог суперпользователя (часто просто /)
/sbin	Все	Команды, необходимые для обеспечения минимальной работоспособности системы
/tmp	Все	Временные файлы, которые могут удаляться при перезагрузке
/usr	Все	Иерархия дополнительных файлов и программ
/lib64	L	64-разрядные библиотеки для 64-разрядных дистрибутивов Linux
/sbin	Все	Менее важные файлы системного администрирования
/var	Все	Системные данные и конфигурационные файлы

Примечание: L = Linux, S = Solaris, H = HP-UX, A = AIX

ГЛОССАРИЙ

Система счисления – это совокупность правил для обозначения и наименования чисел.

Непозиционные системы счисления – это системы счисления, в которых каждой цифре соответствует величина, не зависящая от её места в записи числа.

Позиционные системы счисления – это системы счисления, в которых вклад каждой цифры в величину числа зависит от её положения (позиции) в последовательности цифр, изображающей число.

Основание позиционной системы счисления – это количество символов, используемое для записи числа. Основание системы счисления показывает, во сколько раз изменится количественное значение цифры при перемещении ее на соседнюю позицию.

Алгебра логики (Булева алгебра) – это раздел математики, изучающий высказывания, рассматриваемые со стороны их логических значений (истинности или ложности) и логических операций над ними.

Логическое высказывание – это любое повествовательное предложение, в отношении которого можно однозначно сказать, истинно оно или ложно. Высказывания бывают простыми и сложными.

Простое логическое высказывание (логическая переменная) – это высказывание, которое содержит только одну простую мысль.

Сложное логическое высказывание (логическая функция) – это высказывание, которое содержит несколько простых мыслей, соединенных между собой с помощью логических операций.

Таблица истинности – это таблица, в которой перечислены все возможные значения входящих логических переменных и соответствующие им значения функции.

Элементарная конъюнкция – это конъюнкция простых высказываний или их отрицаний.

Элементарная дизъюнкция – это дизъюнкция простых высказываний или их отрицаний.

Дизъюнктивная нормальная форма (ДНФ) – это форма, в которой логическая функция представляется в виде дизъюнкции элементарных конъюнкций.

Конъюнктивной нормальной формой (КНФ) – это форма, в которой функция представляется в виде конъюнкции элементарных дизъюнкций.

Логические элементы (вентили) – это электронные схемы с одним или несколькими входами и одним выходом, через которые проходят электрические сигналы, представляющие цифры 0 и 1.

Графический интерфейс пользователя (англ. Graphical user interface, GUI) – управление программами с помощью графических кнопок, всплывающих меню, окон и других элементов.

Интерфейс командной строки (англ. Command Line Interface, CLI) – управление программами с помощью команд.

Терминал – графическая программа, эмулирующая консоль.

Консоль – это программное обеспечение, реализующее текстовый интерфейс.

Информатика – это комплексная техническая наука, которая изучает и систематизирует приёмы создания, хранения, переработки, обработки и передачи данных средствами вычислительной техники, а также принципы функционирования этих средств и методы управления ими.

Информация – это сведения о лица, предметах, фактах, событиях, явлениях и процессах независимо от формы их представления.

Носитель информации – это среда для записи, хранения и передачи информации.

Сообщение – это информация, представленная в определенной форме и предназначенная для передачи.

Данные – это информация, представленная в формализованном виде и предназначенная для обработки её техническими средствами, например ЭВМ.

Информационная технология (информационные технологии) – это совокупность методов, производственных и программно-технологических средств, объединенных в технологическую цепочку, обеспечивающую сбор, хранение, обработку, вывод и распространение информации.

Эмулятор терминала – это графическая программа, которая предоставляет вам доступ к tty или pts терминалу.

Командная оболочка – интерпретатор команд ОС, обеспечивающий интерфейс для взаимодействия пользователя с функциями системы.

Командная строка – это то место, куда вы будете вводить свои команды, строка приглашения в терминале для ввода.

Текущий каталог – это каталог, с которым пользователь работает в данный момент времени

Путь к файлу – это цепочка имен каталогов, которые необходимо указать для доступа к заданному файлу, вместе с именем этого файла.

Константа – это величина, которая не изменяет свое значение в процессе выполнения программы.

Переменная – это простейшая именованная структура данных, в которой может быть сохранён промежуточный или конечный результат работы программы. Переменная может изменять свое значение в процессе выполнения программы.

Линейный вычислительный процесс – это процесс обработки данных, в котором действия выполняются последовательно одно за другим.

Разветвляющийся вычислительный процесс – это процесс обработки данных, в котором те или иные действия совершаются в зависимости от результата проверки какого-либо условия.

Циклический вычислительный процесс – это процесс обработки данных, в котором действия повторяются многократно при различных значениях входящих величин.

Комбинированные алгоритмы – это алгоритмы, в которых для решения задачи используется несколько алгоритмических структур.

Массив – это упорядоченное множество однотипных переменных (элементов массива), объединенных общим именем и отличающихся номерами (индексами).

Размерность массива – это число индексов. Чаще всего используются одномерные (один индекс) и двумерные (два индекса) массивы.

Размер (глубина) массива – это количество элементов в массиве.

Цифровые технологии – это разнообразные инновационные средства, основанные на использовании цифровой информации и обработке данных с помощью компьютеров и электронных устройств.

Алгоритм – это набор инструкций, описывающих порядок действий исполнителя для достижения результата решения задачи за конечное число шагов.

Алгоритмизация – это процесс разработки алгоритма для решения задачи.

Блок-схема алгоритма – это графический способ представления алгоритма, при котором алгоритм изображается в виде последовательности связанных между собой функциональных блоков, каждый из которых соответствует выполнению одного или нескольких действий.

Структурное программирование – это программирование, основанное на модульной структуре программного продукта и типовых

управляющих структурах алгоритмов обработки данных различных модулей.

Модуль – самостоятельная часть программы, имеющая определенное назначение и обеспечивающая заданные функции обработки автономно от других программных модулей.

Структура – это некоторая конфигурация или конструкция из элементов.

Инструкция (оператор) языка программирования – минимальный самостоятельный элемент языка, определяющий действие, которое требуется выполнить.

ЛИТЕРАТУРА

1. Simply Linux 9.1. Документация. Руководство пользователя (редакция апрель 2021 г.) – Режим доступа: https://docs.altlinux.org/ru-RU/alt-simply/9.1/html/alt-simply/index.html.

2. Балабаева, И. Ю. Учебное пособие по курсу «Информатика»: [16+] / И. Ю. Балабаева, Н. Б. Ельчанинова, Е. Р. Мунтян; Южный федеральный университет. – Ростов-на-Дону; Таганрог: Южный федеральный университет, 2020. – Часть 3. – 117 с.: ил., табл., схем. – Режим доступа: по подписке. – Библиогр. в кн. – ISBN 978-5-9275-3657-3 (Ч. 3) - 978-5-9275-3313-8. – Текст: электронный.

3. Батенков, К. А. Основы алгоритмизации: учебное пособие / К. А. Батенков, А. В. Козленко. – Москва: РТУ МИРЭА, 2025. – 211 с. – ISBN 978-5-7339-2444-1. – Текст: электронный // Лань: электронно-библиотечная система. – URL: https://e.lanbook.com/book/493370 (дата обращения: 06.10.2025). – Режим доступа: для авториз. пользователей.

4. Вычислительные аспекты решения сеточных уравнений на графических ускорителях / Н. Н. Грачева, В. Н. Литвинов, А. В. Никитина [и др.] // Известия ЮФУ. Технические науки. – 2021. – № 5(222). – С. 93-105. – DOI 10.18522/2311-3103-2021-5-93-105.

5. Грачева, Н. Н. Информатика. Алгоритмизация и программирование. Языки программирования высокого уровня: сборник задач / Н. Н. Грачева, Н. Б. Руденко. – Зерноград: ФГБОУ ВПО АЧГАА, 2013. – 74 с.

6. Информатика: учебное пособие: [16+] / Е. Н. Гусева, И. Ю. Ефимова, Р. И. Коробков [и др.]. – 5-е изд., стер. – Москва: ФЛИНТА, 2021. – 260 с.: ил. – Режим доступа: по подписке. – Библиогр. в кн. – ISBN 978-5-9765-1194-1. – Текст: электронный.

7. Информатика [Электронный ресурс]: учебное пособие / В. Н. Литвинов, Н. Н. Грачева, Н. Б. Руденко, О. Ю. Каун. – Электрон. дан. – Зерноград: Азово-Черноморский инженерный институт ФГБОУ ВО ДГАУ, 2019. – 205 с. – Режим доступа: Локальная сеть Библиотеки Азово-Черноморского инженерного института ФГБОУ ВО Донской ГАУ.

8. Карчевская, М. П. Основы алгоритмизации инженерных задач: учебное пособие / М. П. Карчевская, О. Л. Рамбургер. – Вологда: Инфра-Инженерия, 2025. – 244 с. – ISBN 978-5-9729-2355-7. – Текст: электронный // Лань: электронно-библиотечная система. – URL: https://e.lanbook.com/book/499817 (дата обращения: 06.10.2025). – Режим доступа: для авториз. пользователей.

9. Литвинов, В. Н. Прикладное программирование: лабораторный практикум / В. Н. Литвинов, Н. Б. Руденко, Н. Н. Грачева. – Зерноград: Азово-Черноморский инженерный институт ФГБОУ ВО ДГАУ, 2021. – 166 с.

10. Литвинов, В. Н. Разработка модели параллельно-конвейерного вычислительного процесса для решения системы сеточных уравнений / В. Н. Литвинов, Н. Б. Руденко, Н. Н. Грачева // Advanced Engineering Research (Rostov-on-Don). – 2023. – Т. 23, № 3. – С. 329-339. – DOI 10.23947/2687-

1653-2023-23-3-329-339.

11. Литвинов, В. Н. Современное компьютерное программирование: лабораторный практикум / В. Н. Литвинов, Н. Б. Руденко, Н. Н. Грачева. – Зерноград: Азово-Черноморский инженерный институт ФГБОУ ВПО ДГАУ, 2015. – 163 с.

12. Модель расчёта момента силы трения при вращательном движении тела / Ф. В. Постолов, И. В. Егорова, Н. В. Петренко [и др.] // АгроЭкоИнфо. – 2024. – № 4(64). – DOI 10.51419/202144413.

13. Определение удельного сопротивления проводника / Ю. В. Лобко, И. В. Егорова, Н. В. Петренко [и др.] // АгроЭкоИнфо. – 2024. – № 4(64). – DOI 10.51419/202144414.

14. Парамонов, А. И. Алгоритмизация и компьютерные вычисления на языках программирования высокого уровня: учебно-методическое пособие / А. И. Парамонов, А. Г. Савенко. – БГУИР: БГУИР, 2024. – 212 с. – ISBN 978-985-543-738-4. – Текст: электронный// Лань: электронно-библиотечная система. – URL: https://e.lanbook.com/book/479597 (дата обращения: 06.10.2025). – Режим доступа: для авториз. пользователей.

15. Руденко, Н. Б. Информатика: Понятие информации. Общая характеристика процессов сбора, передачи, обработки и накопления информации. Позиционные системы счисления. Количество информации. Логические основы работы компьютера: методические указания / Н. Б. Руденко, Н. Н. Грачева, В. Н. Литвинов. – Зерноград: Азово-Черноморский инженерный институт ФГБОУ ВПО ДГАУ, 2015. – 58 с.

16. Руденко, Н. Б. Информатика: учебное пособие / Н. Б. Руденко, Н. Н. Грачева, В. Н. Литвинов. – Зерноград: Азово-Черноморский инженерный институт ФГБОУ ВО Донской ГАУ, 2017. – 144 с.

17. Свидетельство о государственной регистрации программы для ЭВМ № 2020610567 Российская Федерация. КИС ВУЗ. Сведения об образовательной организации. Направления и результаты научной деятельности: № 2019667157: заявл. 23.12.2019: опубл. 16.01.2020 / В. Н. Литвинов, Н. Н. Грачева, Н. Б. Руденко, О. Ю. Каун; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Донской государственный аграрный университет» (ФГБОУ ВО Донской ГАУ).

18. Свидетельство о государственной регистрации программы для ЭВМ № 2020610266 Российская Федерация. КИС ВУЗ. Сведения об образовательной организации. Информация по образовательным программам: № 2019667194: заявл. 23.12.2019: опубл. 13.01.2020 / В. Н. Литвинов, Н. Н. Грачева, Н. Б. Руденко, О. Ю. Каун; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Донской государственный аграрный университет» (ФГБОУ ВО Донской ГАУ).

19. Свидетельство о государственной регистрации программы для ЭВМ № 2021619896 Российская Федерация. КИС ВУЗ. Сведения об образовательной организации. Информация об администрации

образовательной организации: № 2021619005: заявл. 09.06.2021: опубл. 18.06.2021 / В. Н. Литвинов, Н. Н. Грачева, Н. Б. Руденко, О. Ю. Каун; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Донской государственный аграрный университет».

20. Свидетельство о государственной регистрации программы для ЭВМ № 2021617395 Российская Федерация. КИС ВУЗ. Сведения об образовательной организации. Информация о составе педагогических (научно-педагогических) работников: № 2021616566: заявл. 27.04.2021: опубл. 13.05.2021 / В. Н. Литвинов, Н. Н. Грачева, Н. Б. Руденко, О. Ю. Каун; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Донской государственный аграрный университет».

21. Свидетельство о государственной регистрации программы для ЭВМ № 2022662945 Российская Федерация. Расчет оплаты труда научно-педагогических работников с учетом эффективного контракта: № 2022661514: заявл. 21.06.2022: опубл. 07.07.2022 / Н. А. Глечикова, Н. Н. Грачева, Н. Б. Руденко, А. А. Серегин; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Донской государственный аграрный университет».

22. Свидетельство о государственной регистрации программы для ЭВМ № 2024683028 Российская Федерация. Расчет момента силы трения при вращательном движении тела: № 2024682058: заявл. 24.09.2024: опубл. 03.10.2024 / Ф. В. Постолов, Н. Н. Грачева, Н. Б. Руденко [и др.]; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Донской государственный аграрный университет».

23. Свидетельство о государственной регистрации программы для ЭВМ № 2024681237 Российская Федерация. Модуль выбора оптимального алгоритма вычисления скалярного произведения векторов большой размерности в гетерогенной вычислительной системе: № 2024680236: заявл. 30.08.2024: опубл. 06.09.2024 / В. Н. Литвинов, Н. Н. Грачева, Н. Б. Руденко; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Донской государственный аграрный университет».

24. Свидетельство о государственной регистрации программы для ЭВМ № 2024680535 Российская Федерация. Расчет удельного сопротивления проводника: № 2024669594: заявл. 20.08.2024: опубл. 29.08.2024 / Ю. В. Лобко, Н. Н. Грачева, Н. Б. Руденко [и др.]; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Донской государственный аграрный университет».

25. Свидетельство о государственной регистрации программы для ЭВМ № 2024680402 Российская Федерация. Модель цифровизации технологического процесса уборки и транспортировки зерновых культур с учетом различных видов потерь на каждом агромероприятии: № 2024669415: заявл. 19.08.2024: опубл. 28.08.2024 / Н. А. Глечикова, Н. Н. Грачева,

Н. Б. Руденко; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Донской государственный аграрный университет».

26. Свидетельство о государственной регистрации программы для ЭВМ № 2024615870 Российская Федерация. Модель цифровой трансформации технологического процесса посева и возделывания зерновых культур с учетом видов потерь и проводимого агромероприятия: № 2024613967: заявл. 27.02.2024: опубл. 13.03.2024 / Н. А. Глечикова, Н. Н. Грачева, Н. Б. Руденко [и др.]; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Донской государственный аграрный университет».

27. Свидетельство о государственной регистрации программы для ЭВМ № 2025662061 Российская Федерация. Расчет теплового баланса автономной зимней теплицы: заявл. 18.04.2025: опубл. 16.05.2025 / М. Ю. Попов, Н. Н. Грачева, Н. Б. Руденко; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Донской государственный аграрный университет».

28. Свидетельство о государственной регистрации программы для ЭВМ № 2024614194 Российская Федерация. Расчет вероятности выполнения работ и коэффициента загрузки ремонтных бригад при устранении массовых повреждений в электрических сетях: № 2024612626: заявл. 09.02.2024: опубл. 21.02.2024 / А. М. Исупова, Н. Б. Руденко, В. Я. Хорольский, А. И. Самарцев; заявитель Федеральное государственное бюджетное образовательное учреждение высшего образования «Донской государственный аграрный университет».

29. Технология программирования: учебное пособие / Ю.Ю. Громов, О.Г. Иванова, М.П. Беляев, Ю.В. Минин. – Тамбов: Изд-во ФГБОУ ВПО «ТГТУ», 2013. – 172 с.

Учебное издание

Руденко Нелли Борисовна
кандидат технических наук, доцент

Грачева Наталья Николаевна
кандидат технических наук, доцент

Литвинов Владимир Николаевич
кандидат технических наук, доцент

Жидченко Татьяна Викторовна
кандидат технических наук, доцент

Марина Николаевна Середина
кандидат технических наук, доцент

Удинцова Надежда Михайловна
кандидат технических наук, доцент

Серёгина Виктория Викторовна
кандидат социологических наук, доцент

Степовой Дмитрий Владимирович
кандидат физико-математических наук, доцент

ИНФОРМАТИКА И ЦИФРОВЫЕ ТЕХНОЛОГИИ В АГРОИНЖЕНЕРИИ

Лабораторный практикум

Часть 1

Издается в авторской редакции

Объем ЭИ: 10,3 Мб
Формат ЭИ: Portable Document Format (PDF)

Вернуться к содержанию