

**Федеральное государственное бюджетное образовательное  
учреждение высшего образования  
«Уфимский университет науки и технологий»**

**Кафедра информатики**

**ЛИНЕЙНЫЕ И РАЗВЕТВЛЕННЫЕ  
КОНСТРУКЦИИ В PYTHON**

**Лабораторный практикум  
по дисциплинам  
«Языки программирования»,  
«Информатика»**

**Уфа 2024**

Федеральное государственное бюджетное образовательное  
учреждение высшего образования  
«Уфимский университет науки и технологий»

Кафедра информатики

# ЛИНЕЙНЫЕ И РАЗВЕТВЛЕННЫЕ КОНСТРУКЦИИ В PYTHON

Лабораторный практикум  
по дисциплинам  
«Языки программирования»,  
«Информатика»

Уфа 2024

Составители: Н. А. Гарифуллина, ФИО

Линейные и разветвленные конструкции в Python:  
Лабораторный практикум по дисциплинам «Языки  
программирования», «Информатика» / [Электронный ресурс] /  
Уфимск. ун-т науки и техн.; Сост.: Н. А. Гарифуллина, ФИО – Уфа:  
УУНиТ, 2024.

Содержит начальные сведения о языке программирования Python , а также описание создания приложений линейной и разветвляющей структуры.

Предназначен для студентов первого и второго курсов технических и экономических специальностей и направлений УУНиТ, изучающих дисциплины «Языки программирования» и «Информатика» в рамках базового курса, а также всех тех, кто изучает язык программирования Python .

Рецензент: к.т.н., доцент каф. информатики Р. Р. Каримов

© Уфимский университет  
науки и технологий, 2024

# ОГЛАВЛЕНИЕ

<b>ВВЕДЕНИЕ.....</b>	<b>4</b>
<b>1. PYTHON: ОСОБЕННОСТИ И НЕДОСТАТКИ .....</b>	<b>6</b>
<b>2. УСТАНОВКА PYTHON .....</b>	<b>10</b>
2.1. Установка Python 3 для Windows и Mac OS X.....	11
2.2. Установка Python 3 для Linux .....	12
<b>3. ВВЕДЕНИЕ В ЯЗЫК PYTHON.....</b>	<b>14</b>
3.1. Моя первая программа .....	14
3.2. Среды программирования.....	14
3.3. Комментарии.....	17
3.4. Типы данных.....	18
3.5. Преобразование типов данных .....	23
3.6. Имена идентификаторов.....	24
3.7. Вывод данных .....	25
3.8. Вывод данных по формату .....	26
3.9. Ввод данных .....	28
3.10. Арифметические операции.....	30
3.11. Знаки сравнения.....	32
3.12. Логические операции .....	34
3.13. Побитовые операции .....	35
3.14. Сокращенная запись .....	35
3.15. Приоритет выполнения операций .....	36
3.16. Основные алгебраические функции.....	37
<b>4. СОЗДАНИЕ ПРИЛОЖЕНИЙ ЛИНЕЙНОЙ СТРУКТУРЫ.....</b>	<b>39</b>
<b>5. СОЗДАНИЕ ПРИЛОЖЕНИЙ РАЗВЕТВЛЯЮЩЕЙ СТРУКТУРЫ .....</b>	<b>44</b>
5.1. Оператор IF.....	44
5.2. Оператор IF-ELSE.....	45
5.3. Оператор IF внутри другого IF-ОПЕРАТОРА .....	48
5.4. Оператор IF-ELSE внутри условия ELSE.....	49
5.5. Оператор IF-ELIF-ELSE.....	49
5.6. Оператор выбора .....	53
<b>6. ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ.....</b>	<b>57</b>
6.1. Ввод-вывод .....	57
6.2. Линейные конструкции. Задания первого уровня сложности .....	60
6.3. Линейные конструкции. Задания второго уровня сложности .....	64
6.4. Разветвленные конструкции. Задания первого уровня сложности .....	70
6.5. Разветвленные конструкции. Задания второго уровня сложности .....	78
<b>7. КОНТРОЛЬНЫЕ ВОПРОСЫ.....</b>	<b>87</b>
<b>СПИСОК ЛИТЕРАТУРЫ.....</b>	<b>88</b>

## **ВВЕДЕНИЕ**

### ***Цель и задачи лабораторной работы:***

Целью работы является изучение основ программирования с использованием базовых конструкций следования, а также механизмов ввода/вывода данных в консольных приложениях с применением языка программирования Python .

### ***Требования к выполнению лабораторных работ:***

1. Перед началом лабораторной работы повторить лекционный материал по данной теме.
2. Получить у преподавателя вариант для выполнения индивидуальных заданий.
3. В порядке, описанном в практикуме, изучить разобранные задачи лабораторной работы.
4. Выполнить индивидуальные задания, приведенные в конце лабораторной работы.
5. Подготовить отчет по лабораторной работе. Ответить на контрольные вопросы, приведенные в конце лабораторной работы.
6. Продемонстрировать выполненную работу преподавателю, ответить на поставленные вопросы, уметь внести изменения в созданный проект.

### ***Отчетность:***

Отчетом (ПЗ) по лабораторной работе является документ, подготовленный с использованием текстового процессора MS Word, который должен быть оформлен в соответствие с требованиями ГОСТ 2.105-95. ЕСКД. Общие требования к текстовым документам и ГОСТ 2.104-68. ЕСКД. Основные надписи. Алгоритмы задачи должны быть представлены блок-схемами, выполненными согласно ГОСТ 19.701-90. ЕСКД. Схемы алгоритмов программ, данных и систем.

### ***Структура ПЗ:***

#### **Введение**

#### **1. Задание 1**

##### **1.1. Постановка задачи**

##### **1.2. Алгоритм решения задачи**

##### **1.3. Программирование**

##### **1.4. Тестирование и отладка, результаты выполнения**

#### **2. Задание 2**

...

Контрольные вопросы

Список использованных источников

***Защита лабораторных работ:***

При защите лабораторной работы студент должен сформулировать постановку задач, уметь объяснить алгоритмы решения задач, используемые операторы языка программирования и конструкции, необходимые для программной реализации алгоритма, а также уметь внести изменения в созданный проект.

## **1. PYTHON: ОСОБЕННОСТИ И НЕДОСТАТКИ**

Язык программирования можно инициализировать как набор команд, направленный на понимание компьютером инструкции к выполнению той или иной программы, написанной на соответствующем языке программирования.

Если же говорить о таком языке программирования как Python, то нельзя не заметить его тенденцию роста популярности. Он используется не только отдельными пользователями, но и целыми компаниями для создания продуктов, приносящих прибыль. Например, компания Google использует Python в своей поисковой системе; платформа YouTube в значительной степени реализована на этом языке. Python ориентирован на повышение производительности разработчика и облегчение задач при написании и читаемости кода. Этот язык программирования был изобретен в 1991 году голландским программистом Гвидо ван Россумом.



*Rис 1. Разработчик языка Python – голландский программист Гвидо ван Россум*

Свое имя Python получил от названия британского комедийного телешоу "Летающий цирк Монти Пайтона".

Получить дополнительные сведения о Python и скачать его дистрибутив можно с помощью официального сайта: <https://www.python.org>.

Python является высокоуровневым языком программирования.  
**Основные особенности языка:**

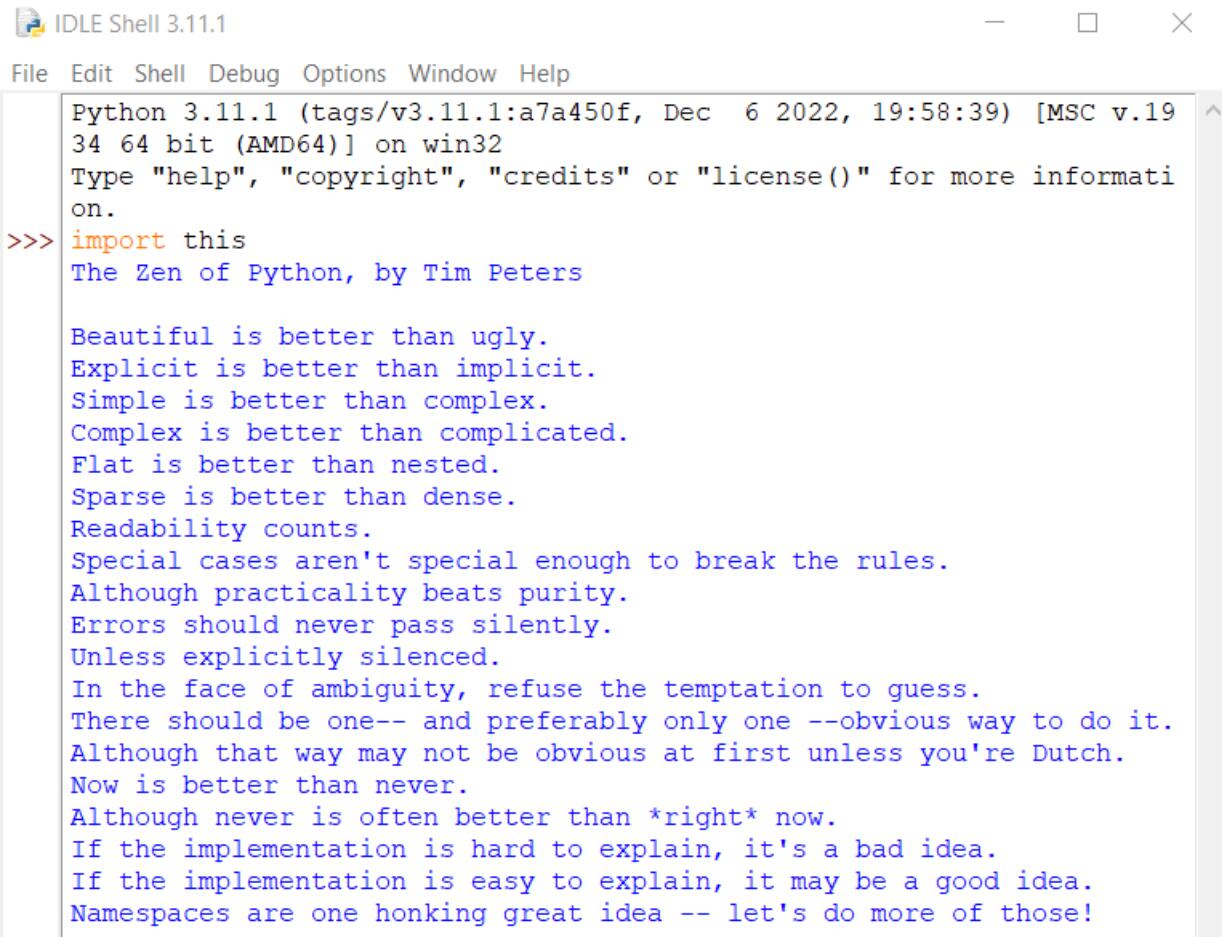
- Python поддерживает структурное, функциональное и объектно-ориентированное программирование.
- Python – интерпретируемый язык программирования. Это означает, что программный код на языке Python не компилируется перед его исполнением в отличие от таких языков, как C и Fortran. Код выполняется специальной программой-интерпретатором. Интерпретатор считывает высокоуровневую программу – исходный код – и, напрямую взаимодействуя с операционной системой, выполняет программу. Преобразование и выполнение программы осуществляется построчно.
- Python поддерживает динамическую типизацию — связывание переменной с типом в момент присваивания ей значения
- В Python реализовано автоматическое управление памятью. Это означает, что пользователю нет необходимости вручную выделять и удалять память для динамических переменных.
- Python поддерживает интерактивный режим работы.
- Для Python написано большое количество свободно распространяемых библиотек, в том числе библиотек для научных расчётов.
- Синтаксис языка Python является легко читаемым и неперегруженным.
- Интерпретатор Python является свободно распространяемым программным обеспечением с открытым исходным кодом. Этalonной реализацией интерпретатора является CPython (URL: <https://www.python.org>).
- Кроссплатформенность: Python портирован на большинство существующих платформ, в том числе Windows, Linux и Mac OS X. Программный код, написанный на Python , может выполняться на любой платформе, на которой установлен интерпретатор Python.

### ***Недостатки Python:***

Скорость выполнения программ несколько ниже по сравнению с компилируемыми языками (C или C++), поскольку Python транслирует инструкции исходного программного кода в байт-код, а затем интерпретирует этот байт-код. Байт-код обеспечивает переносимость программ, поскольку это платформонезависимый формат. Как результат, имеет место преимущество в скорости разработки с потерей скорости выполнения.

### ***Дзен Python'a:***

Если интерпретатору в Python дать команду import this, то, как показано на Рис. 2, выведется, так называемый, «Zen of Python», иллюстрирующий идеологию и особенности данного языка. Глубокое понимание дзена приходит тем, кто сможет освоить язык Python в полной мере.



```

File Edit Shell Debug Options Window Help
Python 3.11.1 (tags/v3.11.1:a7a450f, Dec 6 2022, 19:58:39) [MSC v.19
34 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

>>> import this
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
...

```

Рис. 2. Zen of Python

Таблица 1

Перевод «Zen of Python»

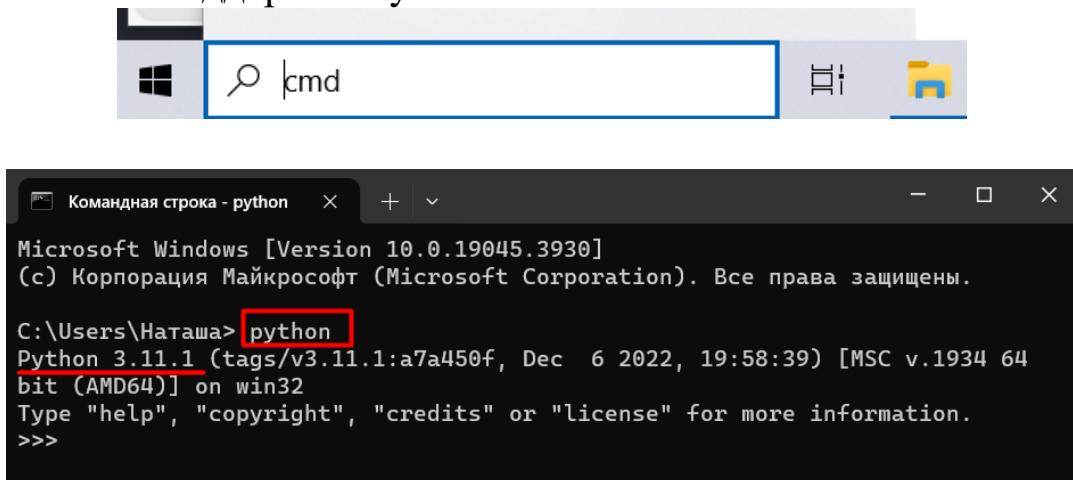
	<b>Фраза</b>	<b>Перевод</b>
1	<i>Beautiful is better than ugly</i>	Красивое лучше уродливого
2	<i>Explicit is better than implicit</i>	Явное лучше неявного
3	<i>Simple is better than complex</i>	Простое лучше сложного
4	<i>Complex is better than complicated</i>	Сложное лучше усложненного
5	<i>Flat is better than nested</i>	Плоское лучше вложенного
6	<i>Sparse is better than dense</i>	Разреженное лучше плотного
7	<i>Readability counts</i>	Удобочитаемость важна
8	<i>Special cases aren't special enough to break the rules</i>	Частные случаи не настолько существенны, чтобы нарушать правила
9	<i>Although practicality beats purity</i>	Однако практичность важнее чистоты

10	<i>Errors should never pass silently</i>	Ошибки никогда не должны замалчиваться
11	<i>Unless explicitly silenced</i>	За исключением замалчивания, которое задано явно
12	<i>In the face of ambiguity, refuse the temptation to guess</i>	В случае неоднозначности сопротивляйтесь искушению угадать
13	<i>There should be one – and preferably only one – obvious way to do it</i>	Должен существовать один – и, желательно, только один – очевидный способ сделать это
14	<i>Although that way may not be obvious at first unless you're Dutch</i>	Хотя он может быть с первого взгляда не очевиден, если ты не голландец
15	<i>Now is better than never</i>	Сейчас лучше, чем никогда
16	<i>Although never is often better than «right» now</i>	Однако, никогда чаще лучше, чем прямо сейчас
17	<i>If the implementation is hard to explain, it's a bad idea</i>	Если реализацию сложно объяснить – это плохая идея
18	<i>If the implementation is easy to explain, it may be a good idea</i>	Если реализацию легко объяснить – это может быть хорошая идея
19	<i>Namespaces are one honking great idea – let's do more of those!</i>	Пространства имен – прекрасная идея, давайте делать их больше!

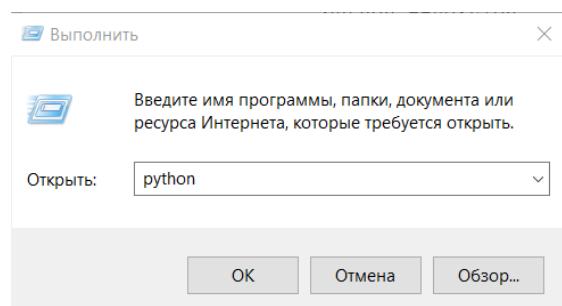
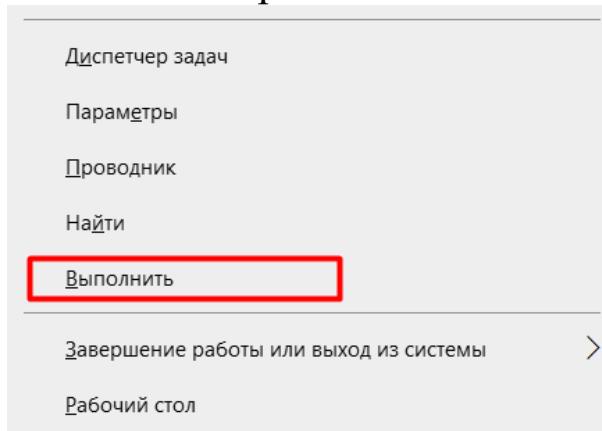
## 2. УСТАНОВКА PYTHON

Для начала проверьте, установлена ли поддержка Python в вашей системе. Сделать это можно несколькими способами:

- 1) В меню «Пуск» введите cmd, в появившемся командном окне введите команду python в нижнем регистре и нажимаем "Ok". Если на экране появится приглашение >>>, значит, в системе установлена поддержка Python.



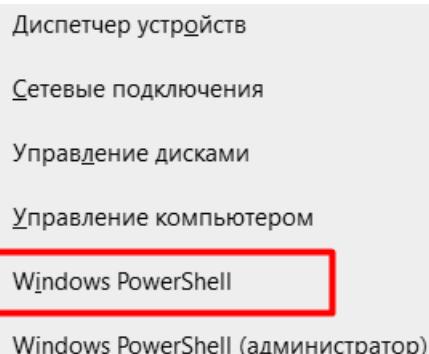
- 2) Нажав сочетание клавиш Win + "X", во всплывшем списке меню нажимаем "Выполнить", появится окно, в появившемся окне вводим python и нажимаем "Ok". Если Python установлен, мы увидим номер установленной версии.



```
C:\Users\Наташа\AppData\Local\Programs\Python\Python311\python.exe
Python 3.11.1 (tags/v3.11.1:a7a450f, Dec 6 2022, 19:58:39) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

3) Также во всплывающем меню можно выбрать «Windows PowerShell». Результат, появившаяся командная строка Windows. Проверить, установлен ли Python в системе, можно командной:

```
python --version
```

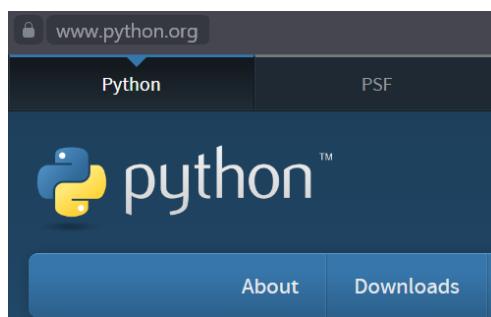


```
Windows PowerShell
Windows PowerShell
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

Попробуйте новую кроссплатформенную оболочку PowerShell (https://aka.ms/pscore6)
PS C:\Users\Наташа> python --version
Python 3.11.1
PS C:\Users\Наташа>
```

## 2.1. Установка python 3 для Windows и Mac OS X

Скачивать Python рекомендуется только с официального сайта <https://www.python.org/>, где можно найти всю доступную информацию о языке программирования.



**Download**

Python source code and installers are available for download for all versions!

Latest: [Python 3.12.1](#)

**Скачать**

Исходный код Python и установщики доступны для скачивания для всех версий!

Последняя версия: [Python 3.12.1](#)

## Files

Version	Operating System	Description	MD5 Sum
<a href="#">Gzipped source tarball</a>	Source release		51c5c22dc698483734dff5c8028606
<a href="#">XZ compressed source tarball</a>	Source release		50f827c800483776c8ef86e6a53831fa
<a href="#">macOS 64-bit universal2 installer</a>	macOS	for macOS 10.9 and later	eae2d617cbd978a4a6c167924b287572
<a href="#">Windows embeddable package (32-bit)</a>	Windows		acc28815c74facc402469e917c8f8433
<a href="#">Windows embeddable package (64-bit)</a>	Windows		019788d34af2c60a7be45bf8273e361f
<a href="#">Windows embeddable package (ARM64)</a>	Windows		18058aa3c8ccbf3e8fee53386d38c711
<a href="#">Windows installer (32-bit)</a>	Windows		37a89a09138888e6331ec279d68fea8e
<a href="#">Windows installer (64-bit)</a>	Windows	Recommended	3e3b6550e58772d324f7519bfa8066dc
<a href="#">Windows installer (ARM64)</a>	Windows	Experimental	25fb741f175dc98d5630520f2df931ec

После скачивания производится установка python 3. Сразу после установки, можно приступить к разработке программ. Помимо самого языка в установщик включена простая среда разработки, называемая IDLE.

## 2.2. Установка python 3 для Linux

Скачивать Python для Linux нет необходимости. В большинстве дистрибутивов он уже установлен. Для проверки установки введите в терминале Linux:

```
python 3
```

Если он уже установлен, терминал запустит окружение python версии 3. Если же он не был установлен по умолчанию, введите в терминале (некоторые системы вместо apt-get используют yum):

```
sudo apt-get install python 3
```

или аналогичную команду для дистрибутивов, не поддерживающих apt-get. Python можно также установить через

стандартный менеджер приложений (обычно в нем содержатся старые версии).

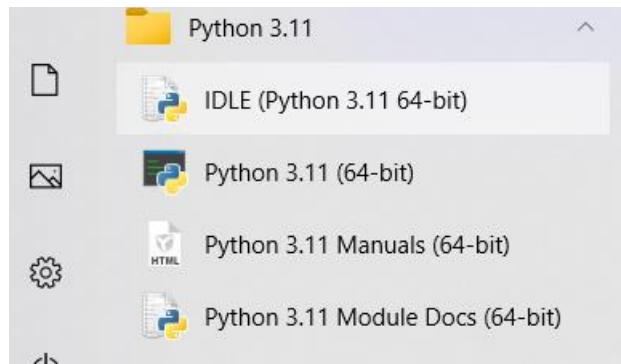
По умолчанию с Python для Linux IDLE не распространяется. Установить IDLE можно командой (некоторые системы вместо apt-get используют yum):

```
sudo apt-get install idle3
```

### **3. ВВЕДЕНИЕ В ЯЗЫК PYTHON**

#### **3.1. Моя первая программа**

После загрузки и установки Python необходимо открыть IDLE, среда разработки на языке Python поставляется вместе с дистрибутивом:



После запуска IDLE (изначально запускается в интерактивном режиме) можно начинать писать первую программу.

В мире программирования издавна принято начинать освоение нового языка с программы, выводящей на экран сообщение `Hello world!` – считается, что это приносит удачу. На языке Python программа состоит всего из одной строки:

```
print("Hello world")
Hello world
```

*Rus. 3. Пример: «Hello, World»*

Для вывода на экран сообщения «`Hello, World!`» достаточно обратиться к стандартной функции `print()` и передать ей в качестве параметра строку `"Hello, World!"`.

#### **3.2. Среды программирования**

##### **1) Python IDLE**

The screenshot shows two windows. The top window is titled 'summa.py - D:/2023-2024 учебный год/Языки программирования/Методички Python создают/summa.py'. It contains the following Python code:

```
a = int(input("Введите а: "))
b = int(input("Введите б: "))
print("Сумма:", a + b)
```

The bottom window is titled 'IDLE Shell 3.11.1'. It shows the Python environment information and a command-line session:

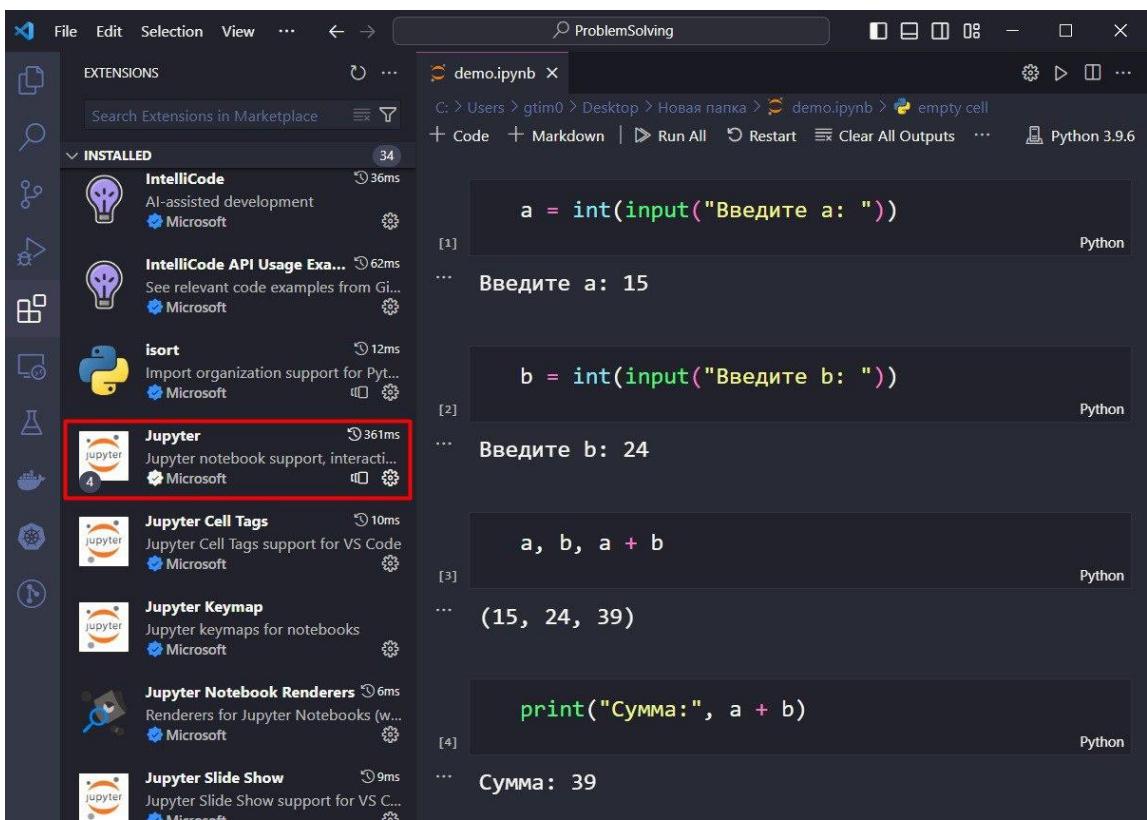
```
Python 3.11.1 (tags/v3.11.1:a7a450f, Dec  6 2022, 19:58:39) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

>>>
= RESTART: D:/2023-2024 учебный год/Языки программирования/Методички Python создают/summa.py
Введите а: 15
Введите б: 24
Сумма: 39
```

## 2) Visual Studio Code

The screenshot shows the Visual Studio Code interface with a terminal window open. The terminal window is titled 'Code - 1\_contest' and displays the following command-line session:

```
PS C:\Users\gtim0\source\repos\ProblemSolving> cd "c:\Users\gtim0\source\repos\ProblemSolving\ProblemSolving\TinkoffVsosh\1_contest"
PS C:\Users\gtim0\source\repos\ProblemSolving\ProblemSolving\TinkoffVsosh\1_contest> python -u "c:\Users\gtim0\source\repos\ProblemSolving\ProblemSolving\TinkoffVsosh\1_contest\1.py"
Введите а: 15
Введите б: 24
Сумма: 39
PS C:\Users\gtim0\source\repos\ProblemSolving\ProblemSolving\TinkoffVsosh\1_contest>
```



### 3) Блокнот Jupyter Notebook

jupyter demo Last Checkpoint: несколько секунд назад (autosaved)

The screenshot shows a Jupyter Notebook interface. A single code cell is visible, containing Python code to input values 'a' and 'b', output their sum, and print the result. The cell's output shows the user input 'Ведите a: 15' and 'Ведите b: 24', the calculated sum '(15, 24, 39)', and the printed result 'Сумма: 39'.

```
In [1]: a = int(input("Ведите a: "))

Ведите a: 15

In [2]: b = int(input("Ведите b: "))

Ведите b: 24

In [3]: a, b, a + b

Out[3]: (15, 24, 39)

In [4]: print("Сумма:", a + b)

Сумма: 39
```

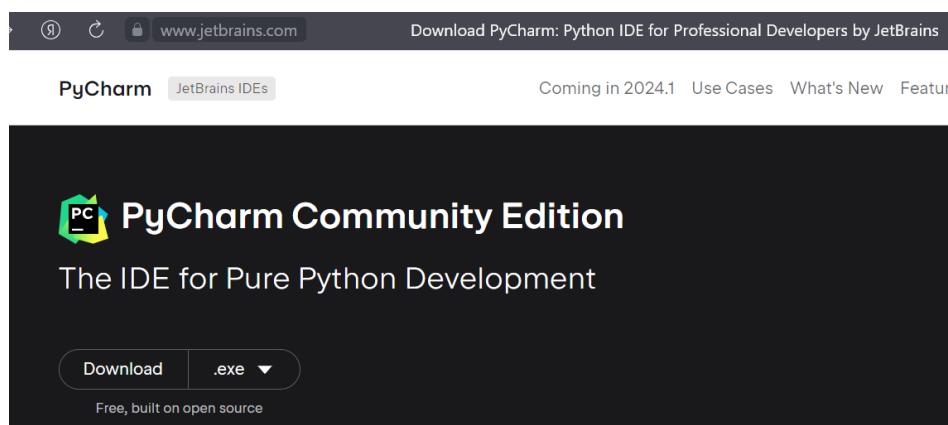
### 4) PyCharm

```
a = int(input("Введите а: "))
b = int(input("Введите б: "))
print("Сумма:", a + b)
```

Run: main

C:\Users\gtim0\AppData\Local\Programs\Python\Python39\python  
Введите а: 15  
Введите б: 24  
Сумма: 39  
Process finished with exit code 0

При установке PyCharm нужно выбрать PyCharm Community Edition:  
<https://www.jetbrains.com/pycharm/download/?section=windows>



### 3.3. Комментарии

Комментарии – это то, что пишется после # и представляет интерес лишь как заметка для читающего программу. Текст программы говорит о том, КАК, а комментарии должны объяснять, ПОЧЕМУ.

- Комментарии могут встречаться в тексте программы как отдельные строки или могут быть размещены справа от операторов в строке.

- Текст после символов # пропускается интерпретатором как примечание, предназначенное для человека, читающего данную программу.
- При копировании кода Вы можете игнорировать комментарии, т.к. они носят исключительно информативный характер.
- Грамотное использование комментариев может в значительной степени повысить читаемость, облегчить понимание и, как следствие, улучшить поддерживаемость Вашего программного кода.

### 3.4. Типы данных

Python – язык программирования со строгой динамической типизацией.

**Строгая** означает, что язык не производит неявные преобразования типов и не создаёт сюрпризов при их случайном смешении.

**Динамическая** означает, что типы объектов определяются в процессе исполнения программы (runtime). Поэтому можно не указывать типы переменных. Переменные в Python – это всего лишь указатели на объекты, они не содержат информации о типе.

Типы данных в Python можно разделить на **изменяемые** и **неизменяемые**.

Объекты встроенных типов, таких как int, float, bool, str, tuple, unicode, являются **неизменяемыми**. Объекты встроенных типов, таких как list, set, dict, являются **изменяемыми**.

Когда мы присваиваем новое значение неизменяемому объекту, Python не перезаписывает его, а создаёт новый объект с тем же именем. Чтобы в этом убедиться, достаточно проверить id – уникальный номер, который присваивается каждому объекту в Python:

```
#числовые типы в Python – неизменяемые
int_obj = 10

print("id      of      int_obj:      ",      id(int_obj))
#140717895746096

int_obj += 5
```

```
print("id      of      int_obj:      ",      id(int_obj))
#140717895746256
```

Когда мы прибавляем 5 к переменной `int_obj`, на её месте создаётся новый объект с тем же именем. Заметьте: при первом выводе `id` равен 140717895746096, а после выполнения операции сложения – 140717895746256.

**Числовые** типы данных: число с плавающей точкой (`float`), целое число (`int`), логический тип (`bool`), комплексное число (`complex`). При присваивании переменных числового типа копируется значение. Вещественные числа типа `float` в Python реализованы с двойной точностью, аналогично числам типа `double` в языке С.

Можно использовать функцию `type()`, чтобы узнать класс переменной или значения, и функцию `isinstance()` для проверки принадлежности объекта определённому классу:

```
>>> a = 5
>>> print(a, "is of type", type(a))
5 is of type <class 'int'>
>>> a = 2.0
>>> print(a, "is of type", type(a))
2.0 is of type <class 'float'>
>>> a = 1+2j
>>>     print(a,      "is      complex      number?", 
isinstance(1+2j, complex))
(1+2j) is complex number? True
```

Целые числа могут быть любой длины, они ограничиваются лишь доступной памятью.

Числа с плавающей запятой имеют ограниченную точность. Визуально разницу между целым числом и числом с плавающей запятой можно заметить в консоли по наличию точки: 1 – целое число, 1.0 – с плавающей запятой.

Комплексные числа записываются в форме `x+yj`, где `x` – действительная часть числа, а `y` – мнимая.

Вот несколько примеров:

```
>>> a = 1234567890123456789
>>> a
1234567890123456789
```

```
>>> b = 0.1234567890123456789  
>>> b  
0.12345678901234568  
>>> c = 1+2j  
>>> c  
(1+2j)
```

Переменные типа `bool` принимают всего два значения: `True` (истина) и `False` (ложь).

**Список** (`list`) 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, 2.2, 'python']
```

Можно использовать оператор `[]` для извлечения элемента (такая операция называется “доступ по индексу”) или диапазона элементов (такая операция называется “извлечение среза”) из списка. В Python индексация начинается с нуля:

```
>>> a = [5, 10, 15, 20, 25, 30, 35, 40]  
>>> print("a[2] =", a[2])  
a[2] = 15  
>>> print("a[0:3] =", a[0:3])  
a[0:3] = [5, 10, 15]  
>>> print("a[5:] =", a[5:])  
a[5:] = [30, 35, 40]
```

Списки являются изменяемым типом, т.е. значения его элементов можно изменить:

```
>>> a = [1, 2, 3]  
>>> a[2] = 4  
>>> a  
[1, 2, 4]
```

Так же как и список, **кортеж** (`tuple`) является упорядоченной последовательностью элементов. Вся разница заключается в том, что кортежи неизменяемы.

Кортежи используются для защиты данных от перезаписи и обычно работают быстрее, чем списки, т.к. их нельзя изменять.

Для создания кортежа нужно поместить внутрь круглых скобок элементы, разделённые запятой:

```
>>> t = (5, 'program', 1+3j)
```

Можно использовать оператор извлечения среза [ ] для извлечения элементов, но мы не можем менять их значения:

```
>>> t = (5, 'program', 1+3j)
>>> print("t[1] =", t[1])
t[1] = program
>>> print("t[0:3] =", t[0:3])
t[0:3] = (5, 'program', (1+3j))
# Приводит к ошибке, т.к.
# кортежи неизменяемы
>>> t[0] = 10
```

**Строка** (str) представляет собой последовательность символов. Мы можем использовать одинарные или двойные кавычки для создания строки. Многострочные строки можно обозначить тройными кавычками, " " или "'''":

```
>>> s = "Простая строка"
>>> s = '''многострочная
строка'''
```

Как и в случае со списками и кортежами, мы можем использовать оператор [ ] и со строками. Стоит отметить, что строки в Python относятся к категории неизменяемых последовательностей, то есть все функции и методы могут лишь создавать новую строку.

**Множество** является неупорядоченной уникализированной последовательностью. Объявляется множество с помощью элементов, разделённых запятой, внутри фигурных скобок:

```
>>> a = {5, 2, 3, 1, 4}
# вывод переменной множества
>>> print("a =", a)
a = {1, 2, 3, 4, 5}
# тип данных переменной a
>>> print(type(a))
<class 'set'>
```

Над множествами можно выполнять такие операции, как объединение и пересечение. Т.к. элементы во множестве должны быть уникальны, они автоматически удаляют дубликаты:

```
>>> a = {1, 2, 2, 3, 3, 3}
```

```
>>> a  
{1, 2, 3}
```

Поскольку множество является неупорядоченной последовательностью, оператор извлечения среза здесь не работает:

```
>>> a = {1,2,3}  
>>> a[1]  
Traceback (most recent call last):  
  File "<stdin>", line 1, in <module>  
TypeError: 'set' object does not support indexing
```

**Словари** (dict) – неупорядоченные наборы пар ключ-значение.

Они используются, когда нужно сопоставить каждому из ключей значение и иметь возможность быстро получать доступ к значению, зная ключ. В других языках словари обычно называются map, hash или object. Словари оптимизированы для извлечения данных. Чтобы извлечь значение, нужно знать ключ.

Словарь объявляется парами элементов в форме ключ:значение, заключенными в фигурные скобки:

```
>>> d = {1:'value', 'key':2}  
>>> type(d)  
<class 'dict'>
```

Значение может быть любого типа, а вот ключ – только неизменяемого.

Мы используем ключ, чтобы получить соответствующее ему значение. Но не наоборот:

```
>>> d = {1:'value', 'key':2}  
>>> print("d[1] =", d[1]);  
d[1] = value  
>>> print("d['key'] =", d['key']);  
d['key'] = 2  
# Приводит к ошибке  
>>> print("d[2] =", d[2]);
```

Особенностью Python является динамическая типизация – переменная связывается с типом в момент присваивания значения, а не в момент объявления переменной. В таком подходе нет необходимости явно объявлять тип переменной до того, как ей будет присвоено какое-либо значение.

### 3.5. Преобразование типов данных

Можно преобразовывать значения из одного типа в другой с помощью таких функций, как `int()`, `float()`, `str()` и т.д.

```
>>> float(5)  
5.0
```

При преобразовании числа с плавающей запятой в целое будет утеряна часть после запятой:

```
>>> int(10.6)  
10  
>>> int(-10.6)  
-10
```

Для преобразования из/в строку должны использоваться совместимые значения:

```
>>> float('2.5')  
2.5  
>>> str(25)  
'25'  
>>> int('1p')  
Traceback (most recent call last):  
  File "<stdin>", line 1, in <module>  
ValueError: invalid literal for int() with base  
10: '1p'
```

Можно преобразовывать одну последовательность в другую:

```
>>> set([1,2,3])  
{1, 2, 3}  
>>> tuple({5,6,7})  
(5, 6, 7)  
>>> list('hello')  
['h', 'e', 'l', 'l', 'o']
```

**Примечание.** Для преобразования списка из символов обратно в строку нельзя вызвать `str` (список), так как в результате получим строковое представление списка (наподобие того, что видим, когда выводим список на экран). Вместо этого нужно сделать следующее:

```
'.join(['h', 'e', 'l', 'l', 'o'])
```

Для преобразования в словарь каждый элемент последовательности должен быть парой:

```
>>> dict([[1,2],[3,4]])
```

```
{1: 2, 3: 4}
>>> dict([(3, 26), (4, 44)])
{3: 26, 4: 44}
```

### 3.6. Имена идентификаторов

Переменные – это частный случай идентификаторов.

Идентификаторы – это имена, присвоенные чему-то для его обозначения.

При выборе имен идентификаторов необходимо соблюдать следующие правила:

- Первым символом идентификатора должна быть буква из алфавита (символ ASCII в верхнем или нижнем регистре, или символ Unicode), а также символ подчеркивания («\_»);
- Остальная часть идентификатора может состоять из букв (символы ASCII в верхнем или нижнем регистре, а также символы Unicode), знаков подчеркивания «\_» или цифр (0 – 9);
- Имена идентификаторов чувствительны к регистру!

**Допустимые** имена: i, \_\_my\_name, name\_23, a1b2\_c3

**Недопустимые** имена: 2things, my-name, >a1b2\_c3

Определяемые имена не могут совпадать с ключевыми словами в Python (*Таблица 2*).

*Таблица 2*

False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	import	pass	break
except	in	raise		

```
# Пример создания переменных
Age = 19
Name = 'Ivan'
isActive = True
name = 'Petr'
```

```
Name = 'Ivan'  
# name и Name - это разные переменные  
# Имена переменных не могут начинаться с цифры!
```

В Python есть 4 примитивных типа данных:

```
# int (целые числа)  
age = 18  
# float (дробные числа)  
fraction = 2.5  
# str (строки)  
fruit = 'apple'  
# bool (правда или ложь)  
isReady = True # всего два значения: True и False
```

Мы можем преобразовывать один тип данных в другой с помощью одноименных функций. Например, число может стать строкой, строка – числом, дробное число – целым.

```
age = '22' # str -> '22'  
age = int(age) # int -> 22  
age = float(age) # float -> 22.0  
age = bool(age) # bool -> True
```

### 3.7. Вывод данных

Для печати значений в Python есть функция `print()`. Внутри круглых скобок через запятую мы пишем то, что хотим вывести. Вот программа, которая делает несколько вычислений:

```
print(5 + 10)  
>> 15  
print(3 * 7, (17 - 2) * 8)  
>> 21 120  
print(2 ** 16)      # две звёздочки означают  
возведение в степень  
>> 65536  
print(37 / 3)      # один слэш – это деление с  
ответом-дробью  
>> 12.33333333333334  
print(37 // 3)      # два слэша считают частное от  
деления нацело, это как операция div в других  
языках  
>> 12
```

```
print(37 % 3)      # процент считает остаток от
деления нацело, это как операция mod в других
языках
>> 1
```

Функция `print()` может принимать несколько входных параметров.

```
print(1, 2, 3)
# Вывод
>> 1 2 3
```

Каждый `print()` выводит данные на новой строке. По умолчанию завершающий символ строки равен символу новой строки (`\n`).

```
print('Hello')
print('world')
# Вывод
>> Hello
>> world
```

Завершающий символ строки в функции `print()` можно изменять.

```
print('Hello', end=' ')
print('world')
# Вывод
>> Hello world
```

### 3.8. Вывод данных по формату

Форматирование может выполняться в так называемом старом стиле или с помощью строкового метода `format`. Старый стиль также называют Стилем, так как он схож с тем, как происходит вывод на экран в языке С.

```
a = 1
b = 2
c = 3
print("a = %s, b = %d, c = %.2f" % (a,b,c))
# выведет a = 1, b = 2, c = 3.00
```

Здесь вместо трех комбинаций символов `%s`, `%d`, `%f` подставляются значения переменных `a`, `b`, `c`. Буквы `s`, `d`, `f` обозначают типы данных – строку, целое число, вещественное число. Если бы требовалось подставить три строки, то во всех случаях использовалось бы сочетание `%s`. Мы можем указать, сколько

требуется знаков после запятой, записав перед буквой `f` точку с желаемым числом знаков в дробной части.

Теперь посмотрим на метод `format()`:

```
c = 3
print("a = {0}, b = {1}, c = {2}".format("a", 1,
c))
# выведет a = a, b = 1, c = 3
24
```

В строке в фигурных скобках указаны номера данных, которые будут сюда подставлены. Далее к строке применяется метод `format()`. В его скобках указываются сами данные, можно использовать переменные. На нулевое место подставится первый аргумент метода `format()`, на место с номером 1 – второй и т. д. Следует отметить, что возможности метода `format()` существенно шире, но нам пока будет достаточно этого.

Начиная с версии 3.6 в языке Python, появился еще один способ форматирования строк – `f`-строки. Этот способ является более кратким, удобным к восприятию и менее подверженным ошибкам, чем другие способы форматирования, при этом и быстрее всех остальных.

Чтобы воспользоваться `f`-строками, достаточно сначала поставить букву `f`, затем открыть кавычки, где указать необходимые для вывода данные, здесь следует отметить, что имена переменных, выражения и другие инструкции следует указывать в фигурных скобках.

Значением, являющимся результатом работы `f`-строк, является строка, поэтому взаимодействие с ним происходит как со строкой, оно может быть передано функции или присвоено переменной.

Рассмотрим несколько примеров применения `f`-строк.

```
name = "Ivan"
age = 10
print(f"Hello, {name}. You are {age}.")
# выведет Hello, Ivan. You are 10.
```

Следует отметить, что для применения `f`-строк можно использовать и заглавную букву `F`.

Передавать в `f`-строки можно как переменные, так и все допустимые выражения Python.

```
print(f" x = {2 * 3 - 1}")  
# выведет x = 5
```

Вызов функций также доступен в f-строках.

```
print(f"{len('value')}")  
# выведет 5
```

В приведенном коде, в f-строки передана функция определения длины последовательности, в данном случае последовательностью является слово 'value'.

### 3.9. Ввод данных

Функция `input()` принимает пользовательский ввод данных.

```
name = input()  
print('Hello ' + name)  
# После запуска скрипта Python будет ожидать ввода  
данных
```

Функция `input` может принимать всего лишь один аргумент – строку, которая выводится перед входной строкой.

```
name = input('Enter your name: ')  
print('Hello, ' + name)
```

Данные, полученные с помощью функции `input()`, имеют строковый тип данных (`str`).

Строки можно складывать друг с другом, такое сложение называется их **конкатенацией** или **объединением**.

```
# Сумма двух строчных чисел  
number1 = input('Введите число: ')  
number2 = input('Введите число: ')  
print(number1 + number2)  
# Ввод:  
>> 1  
>> 2  
# Вывод:  
>> 12
```

Преобразуем строковый тип в целое число (`str -> int`).

```
# Исправленная сумма двух чисел  
number1 = int(input('Введите число: '))  
number2 = int(input('Введите число: '))  
print(number1 + number2)
```

```
# Ввод:  
>> 1  
>> 2  
# Вывод:  
>> 3
```

### Пример 1. Произведение

```
# Произведение двух введенных чисел  
a = int(input('Введите число: '))  
b = int(input('Введите число: '))  
print(a * b)  
# Ввод:  
>> 4  
>> 3  
# Вывод:  
>> 12
```

### Пример 2. Приветствие

```
# Приветствие пользователя по его имени  
firstname = input('Введите свое имя: ') # здесь  
приводить к типу int не нужно  
lastname = input('Введите свою фамилию: ')  
print('Здравствуйте, ' + firstname + ' ' +  
lastname) # не забудьте про пробел между словами  
# Ввод:  
>> Иван  
>> Иванов  
# Вывод:  
>> Здравствуйте, Иван Иванов
```

### Пример 3. Остаток

```
# Операция % позволяет получить остаток от деления  
print(10 % 2) # 0, так как 10 делится на 2 нацело  
print(10 % 3) # 1, остаток равен 1  
print(10 % 4) # 2, остаток равен 2  
# Вывод:  
>> 0  
>> 1  
>> 2
```

### Пример 4. Деление нацело

```

# Операция // позволяет получить целую часть от
деления
print(10 // 2) # 5
print(10 // 3) # 3
print(10 // 4) # 2
# Вывод:
>> 5
>> 3
>> 2

```

### 3.10. Арифметические операции

Результатом арифметической операции является число. Надо отметить, что в арифметическом выражении могут участвовать и данные логического типа. В таком случае True интерпретируется как 1, False – как 0.

Операция	Обозначение	Пример использования
Унарный минус	-	-10
Сложение	+	2 + 2 # 4 False + 3 # 3
Вычитание	-	10 - 18 # -8 True - 5 # -4
Умножение	*	4 * 5 # 20
Деление (с сохранением остатка)	/	10 / 4 # 2.5
Возвведение в степень	**	2 ** 4 # 16
Целочисленное деление (деление с округлением в меньшую сторону)	//	10 // 2 # 5 13 // 5 # 2 -12 // 7 # -2 13 // -6 # -3 -10 // -3 # 3 10 // 2.6 # 3.0
Остаток от деления (деление по модулю)	%	10 % 2 # 0 13 % 5 # 3 -12 % 7 # 2 13 % -6 # -5 -10 % -3 # -1 10 % 2.6 # 2.2

Взятие модуля (абсолютное значение)	<code>abs(x)</code>	<code>abs(-10) # 10</code> <code>abs(1000) # 1000</code>
Округление	<code>round(x)</code>	<code>round(2.512) # 3</code> <code>round(2.462) # 2</code> <code>round(2.516, 2) # 2.52</code>
Суммирование	<code>sum(x1, x2)</code>	<code>sum(3, 1, 2, 4, 5)</code> # 15
Минимальное значение	<code>min(x1, x2)</code>	<code>min(3, 1, 2, 4, 5)</code> # 1
Максимальное значение	<code>max(x1, x2)</code>	<code>max(3, 1, 2, 4, 5)</code> # 5
Пара ( $x // y$ , $x \% y$ )	<code>divmod(x, y)</code>	<code>divmod(5, 2) #(2, 1)</code>

Операция «унарный минус» делает из числа (подвыражения) в начале выражения отрицательное значение. Так при записи  $-10 + 20$  сначала выполнится операция унарного минуса « $-10$ » и только затем произойдет сложение « $-10$ » и « $20$ ».

Функции целочисленного деления и нахождения остатка от деления работают по следующему правилу: если число  $a$  не делится нацело на  $b$ , то знак остатка должен совпадать со знаком делителя. Или соблюдается одно из следующих условий.

$$\begin{cases} 0 \leq r < b \\ b < r \leq 0 \end{cases}, \text{ где } b - \text{делитель, } r - \text{остаток.}$$

Результат целочисленного деления в таком случае можно представить, как  $a // b = \left\lfloor \frac{a}{b} \right\rfloor$ , где  $\lfloor A \rfloor$  – округление вниз (до меньшего или равного целого).

$$\text{Соответственно, } a \% b = a - (a // b) \cdot b$$

$$\text{Поэтому } -12 // 7 = \left\lfloor \frac{-12}{7} \right\rfloor = \left\lfloor -2 \frac{5}{7} \right\rfloor = -2$$

$$-12 \% 7 = -12 - (-12 // 7) \cdot 7 = -12 - (-2) \cdot 7 = 2$$

Или

$$13 // (-6) = \left\lfloor \frac{13}{-6} \right\rfloor = \left\lfloor -2 \frac{1}{6} \right\rfloor = -3$$

$$13 \% (-6) = 13 - 13 // (-6) \cdot (-6) = -12 - (-3) \cdot (-6) = 13 - 18 = -5$$

Также данные операции могут быть применены к вещественным числам.

Логика вычисления остается такая же – результат целочисленного деления является вещественным числом с нулевой

дробной частью, результат для остатка от деления вычисляется по приведенной выше формуле.

Существует и другой подход к работе с остатками, когда остаток не может быть отрицательным. Например, остаток от деления 13 на -6 будет равен 1, а сам результат целочисленного деления равен 2.

При этом формула

$$a = b \cdot r + q$$

где  $b$  – делитель,  $r$  – результат целочисленного деления,  $q$  – остаток, справедлива для обоих случаев (Python и данного определения).

### 3.11. Знаки сравнения

Операция	Обозначение	Пример использования
Равно	$==$	<code>4+2 == 3+3 # True</code> <code>False == True # False</code>
Не равно	$!=$	<code>3+1 != 2+2 # False</code>
Больше	$>$	<code>2 &gt; 2 - 3 # True</code>
Больше или равно	$\geq$	<code>2 \geq 1+1 # True</code>
Меньше	$<$	<code>3 &lt; 10 # True</code>
Меньше или равно	$\leq$	<code>5 \leq 1 # False</code> <code>True \leq False # False</code>

При использовании операторов присваивания с логическими значениями, последние сначала преобразуются в числовое представление – `True` в 1, `False` в 0.

Также в Python есть механизм цепочек сравнения (*chaining comparison operators*) с помощью которого, в числе прочего, удобно обозначать диапазоны значений числовых переменных.

**Пример 5.** Число  $x$  в диапазоне [10; 110].

`10 \leq x \leq 110`

- 1) Оператор равенства `==` возвращает `True`, если числа равны, и `False` в противном случае.

```
a = 10
b = 10
print(a == b)
# Вывод
>> True
a = 8
b = 7
```

```
print(a == b)
# Вывод
>> False
```

- 2) Оператор неравенства `!=` возвращает `True`, если числа не равны, и `False` в противном случае.

```
a = 8
b = 7
print(a != b)
# Вывод
>> True
```

- 3) Оператор больше `>` возвращает `True`, если первое число больше второго, и `False` в противном случае.

```
a = 8
b = 7
print(a > b)
print(b > a)
# Вывод
>> True # a > b
>> False # b > a
```

- 4) Оператор меньше `<` возвращает `True`, если первое число меньше второго, и `False` в противном случае.

```
c = 100
d = 200
print(c < d)
print(d < c)
# Вывод
>> True # c < d
>> False # d < c
```

- 5) Иногда требуется выполнение нескольких операторов сравнения сразу. Для таких целей существует оператор `and` (оператор логического умножения, конъюнкция).

```
print(10 > 0 and 5 > 0)
print(10 % 2 == 0 and 12 % 2 == 0) # оба числа
четные
# Вывод:
>> True
>> True
```

- 6) Если хотя бы один из операторов равен `False`, то результат оператора будет равен `False`. Конъюнкция истинна в том случае, когда все условия истинны.

```
print(10 > 100 and 5 > 0 and 10 > 0) # False
# Вывод:
>> False
```

- 7) Последний из операторов – это оператор инверсии `not`. Оператор `not` изменяет (инвертирует) значение на противоположное.

```
print(not False) # True
print(not True) # False
print(not 2 == 0) # True
# Вывод:
>> True
>> False
>> True
```

- 8) Оператор `not` выполняется в приоритете.

```
print(not 1 == 0 or 2 == 0) # True, значение первого условия инвертировано
# Вывод:
>> True
```

### 3.12. Логические операции

Операция	Обозначение	Пример использования
Отрицание	<code>not</code>	<code>not True # False</code>
Дизъюнкция	<code>or</code>	<code>False or True # True</code>
Конъюнкция	<code>and</code>	<code>True and True # True</code>

В языке Python всего три логические операции. В качестве альтернативы можно использовать, например, оператор `<=` для операции импликации и оператор `==` для эквивалентности. Однако стоит помнить о порядке выполнения операций в записываемом выражении.

**Пример 6.** Число  $x$  одновременно четно и больше 10.

```
x % 2 == 0 and x > 10
```

**Пример 7.** Число оканчивается на 5 или принадлежит промежутку  $(-\infty; 0) \cup (100; +\infty)$ .

```
x % 10 == 5 or 0 >= x <= 100
```

Любое выражение в Python вычисляется слева направо, поэтому при вычислении значения логического выражения выражение может быть вычислено досрочно. Выражение считается вычисленным, если результат левого операнда дизъюнкции равен True или результат левого операнда конъюнкции равен False. Так как нет смысла вычислять значение выражения 1 or expr или 0 and expr, потому что результат первого всегда будет истинным, второго – ложным.

### Пример 8.

```
a = 10
b = 15
c = 20
# True or ... == True
if a < b or c > a + b: c = 10
```

На этом принципе построены некоторые алгоритмы проверок значений. Например, при работе с конъюнкцией правильнее писать в качестве левой части выражения самое редко встречающееся условие, тогда записанные правее условия будут проверяться только при истинности самого редко срабатывающего условия.

## 3.13. Побитовые операции

Операция	Обозначение	Пример использования
Побитовое отрицание	$\sim a$	$\sim 5 \ # -6$
Побитовое умножение	$a \& b$	$13 \& 10 \ # 8$
Побитовое сложение	$a \mid b$	$13 \mid 6 \ # 15$
Побитовое исключающее или	$a \wedge b$	$13 \wedge 6 \ # 11$

Побитовое отрицание производит инвертирование бит, в том числе бита, отвечающего за знак. Поэтому в результате получается значение  $-(a+1)$ .

## 3.14. Сокращенная запись

Операция	Обозначение	Эквивалент
Сложение	$a += b$	$a = a + b$
Вычитание	$a -= b$	$a = a - b$
Умножение	$a *= b$	$a = a * b$
Деление	$a /= b$	$a = a / b$

Целочисленное деление	$a //= b$	$a = a // b$
Остаток от деления	$a \% = b$	$a = a \% b$
Побитовое умножение	$a \&= b$	$a = a \& e$
Побитовое сложение	$a  = b$	$a   b$
Побитовое исключающее или	$a ^= b$	$a ^ b$

Сокращенные операторы выполняются всегда в самом конце, после вычисления выражения справа от оператора.

### 3.15. Приоритет выполнения операций

В любом выражении можно задавать порядок выполнения операций с помощью скобок. В остальных случаях нужно знать, в каком порядке выполняются операции в выражении.

Приоритет выполнения операций следующий

Операция
$\star \star$
$\sim$
$\star, /, //, \%$
$+, -$
$\&$
$ $
$^$
$==, !=, <, <=, >, >=$
$not$
$and$
$or$

Говоря иначе, чем выше операция в таблице, тем раньше она выполняется. Если несколько операций записаны в одной строке, то они имеют одинаковый приоритет. При наличии их в одном выражении, выполняются они слева направо, то есть в первую очередь выполняется самая левая одноприоритетная операция.

1      3      2      4      9      7      5      6      8  
 a  $\star \star$  4 + b \* c  $>=$  100 and 5  $^$  c / 10 \* 2  $<=$  50

*Rис. 4. Порядок выполнения операций в выражении*

```
((a ** 4) + (b * c)) >= 100) and ((5 ^ ((c / 10) * 2)) <= 50)
```

Рис. 5. Расстановка приоритета операций с помощью скобок

Стоит заметить, что почти все операции являются левоассоциативными, то есть при совпадении приоритета у соседних операций сначала выполняется операция, которая левее, затем правая.

Единственное исключение – операция возведения в степень, являющаяся правоассоциативной.

Например, запись  $a^{**}b^{**}c$  будет вычисляться, как  $a^{b^c}$ , а не как  $(ab)^c$ .

### 3.16. Основные алгебраические функции

В стандартную поставку Python входит библиотека `math`, в которой содержится большое количество часто используемых математических функций.

Для работы с данным модулем его предварительно нужно импортировать.

```
import math # импорт всех функций библиотеки  
# используем в программе math.cos(x)
```

Возможен импорт отдельный функций:

```
From math import cos, sqrt # импорт нескольких  
функций библиотеки – cos(x) и sqrt(x)  
# используем в программе cos(x), sqrt(x)
```

В Таблице 2 представлены основные функции модуля `math`.

Таблица 3

Функция	Описание
<code>math.cos(a)</code>	Вычисление косинуса числа a. Угол в радианах
<code>math.sin(a)</code>	Вычисление синуса числа a. Угол в радианах
<code>math.tan(a)</code>	Вычисление тангенса числа a. Угол в радианах
<code>math.acos(a)</code>	Вычисление арккосинуса числа a. Результат в радианах
<code>math.asin(a)</code>	Вычисление арксинуса числа a. Результат в радианах
<code>math.atan(a)</code>	Вычисление арктангенса числа a. Результат в радианах

<b>Функция</b>	<b>Описание</b>
math.exp(a)	Вычисление экспоненты ( $e^a = e**a$ )
math.log(x [, base])	С одним аргументом возвращает натуральный логарифм
math.log10(a)	Вычисление десятичного логарифма ( $\lg(a)$ )
math.sqrt(a)	Вычисление корня квадратного числа a ( $a \geq 0$ )
math.pow(a, b)	Возведение числа a в степень b ( $a^b$ )
math.comb(n, k)	Возвращает количество способов выбрать k элементов из n элементов без повторений и без учета порядка (число сочетаний из n элементов по k элементов)
math.factorial(a)	Возвращает целое – факториал x. Если аргумент действительное или отрицательное число возбуждается исключение ValueError
math.floor(a)	Возвращает ближайшее целое число большее, чем a, $\text{floor}(1.5) == 1$ , $\text{floor}(-1.5) == -2$
math.ceil(a)	Округление числа вверх, $\text{ceil}(1.5) == 2$ , $\text{ceil}(-1.5) == -1$

Таблица 4

#### Запись некоторых математических выражений

<b>Математическая запись</b>	<b>Python</b>
$y = \sqrt{\pi x}$	<code>y = Math.Sqrt(Math.pi*x);</code>
$z = \sin\left(-\frac{\pi}{2}\right)$	<code>z = Math.Sin(-Math.pi/2);</code>
$a = \sin^2\left(-\frac{\pi}{2}\right)$ $a = \left(\sin\left(-\frac{\pi}{2}\right)\right)^2$	<code>a=Math.Pow(Math.Sin(-Math.pi/2), 2);</code>

## 4. СОЗДАНИЕ ПРИЛОЖЕНИЙ ЛИНЕЙНОЙ СТРУКТУРЫ

Линейные алгоритмические структуры. Линейный алгоритм (следование) состоит из последовательности операций, выполняющихся только один раз в порядке их следования. Для графического способа записи линейной алгоритмической структуры используется блок «Процесс».



Рис. 6. Линейная структура алгоритма

### Пример 9.

Напишите программу, которая запрашивала бы у пользователя:

- ФИО ("Ваши фамилия, имя, отчество?")
- возраст ("Сколько Вам лет?")
- место жительства ("Где вы живете?")

После этого выводила бы три строки:

"Ваше имя"  
"Ваш возраст"  
"Вы живете в"

```
a = input('Введите Ваши фамилию, имя, отчество ')
b = input('Сколько Вам лет? ')
c = input('Где Вы живете? ')
print('Ваше имя ', a)
print('Ваш возраст ', b)
print('Вы живете ', c)
```

Результат

```
Введите Ваши фамилию, имя, отчество Гарифуллина Наталья Анатольевна
Сколько Вам лет? 48
Где Вы живете? Уфа
Ваше имя Гарифуллина Наталья Анатольевна
Ваш возраст 48
Вы живете Уфа
```

### Пример 10.

Вычислить выражение при  $x=1$ :

$$\left( \arctan(\sqrt{x} - 2.7 \times 10^{-3}) \right)^2 + \frac{e^{-x}}{\cos(x)}$$

```

import math

x = 1
y = math.atan(x ** 0.5 - 2.7e-3) ** 2 + math.exp(-x) /
math.cos(x)
print(y)

>> 1.2956057140344002

```

### Пример 11.

Исходные данные:  $x=3.3$ . Вывести на экран полученные значения функций:

$$y_1 = \frac{x^2 + 2x - 3 + (x + 1)\sqrt{x^2 - 9}}{x^2 - 2x - 3 + (x - 1)\sqrt{x^2 - 9}}$$

$$y_2 = \frac{\sqrt{x + 3}}{\sqrt{x - 3}}$$

#### 1) импорт всех функций библиотеки

```

import math # импорт всех функций библиотеки

x = 3.3
a = math.sqrt(x ** 2 - 9)
b = x ** 2 + 2 * x - 3 + (x + 1) * a
c = x ** 2 - 2 * x - 3 + (x - 1) * a
y1 = b / c
y2 = math.sqrt(x + 3) / math.sqrt(x - 3)

print("y1 =", y1)
print("y2 =", y2)

>> y1 = 4.58257569495584
>> y2 = 4.582575694955842

```

#### 2) импорт одной функции sqrt (x)

```

from math import sqrt #импорт sqrt(x)

x = 3.3
a = sqrt(x ** 2 - 9)
b = x ** 2 + 2 * x - 3 + (x + 1) * a
c = x ** 2 - 2 * x - 3 + (x - 1) * a
y1 = b / c
y2 = sqrt(x + 3) / sqrt(x - 3)

print("y1 =", y1)

```

```

print("y2 =", y2)

>> y1 = 4.58257569495584
>> y2 = 4.582575694955842

```

### 3) без использования функций

```

x = 3.3                      # без функций
a = (x ** 2 - 9) ** 0.5
b = x ** 2 + 2 * x - 3 + (x + 1) * a
c = x ** 2 - 2 * x - 3 + (x - 1) * a
y1 = b / c
y2 = (x + 3) ** 0.5 / (x - 3) ** 0.5

print("y1 =", y1)
print("y2 =", y2)

```

**Пример 12.** Для предыдущей задачи вывести результат по формату.

```

from math import sqrt

x = 3.3
a = sqrt(x ** 2 - 9)
b = x * x + 2 * x - 3 + (x + 1) * a
c = x * x - 2 * x - 3 + (x - 1) * a
y_1 = b / c
y_2 = sqrt(x + 3) / sqrt(x - 3)

print("y1 =", "{:.3f}".format(y_1)) # метод format
print("y2 = %.5f" %(y_2))         # старый стиль

>> y1 = 4.583
>> y2 = 4.58258

```

**Пример 13.** Вычислите значение функции  $y = f(x)$  для заданного аргумента  $x$  из отрезка  $[a,b]$ . Используйте форматный и бесформатный вывод.

$$y = \sin\left(\frac{x-1}{5^{x+1}}\right) - e^x + \cos(2^{2x-1})$$

```

from math import sin, cos, pow, exp

x = float(input("Введите аргумент x "))

```

```

y = sin(pow(5, (x - 1) / (x + 1))) - exp(x) + cos(pow(2,
2 * x - 1))
# метод format
print("Значение функции y = ", "{:.3f}".format(y))
# без формата
print("Значение функции y = ", y)

```

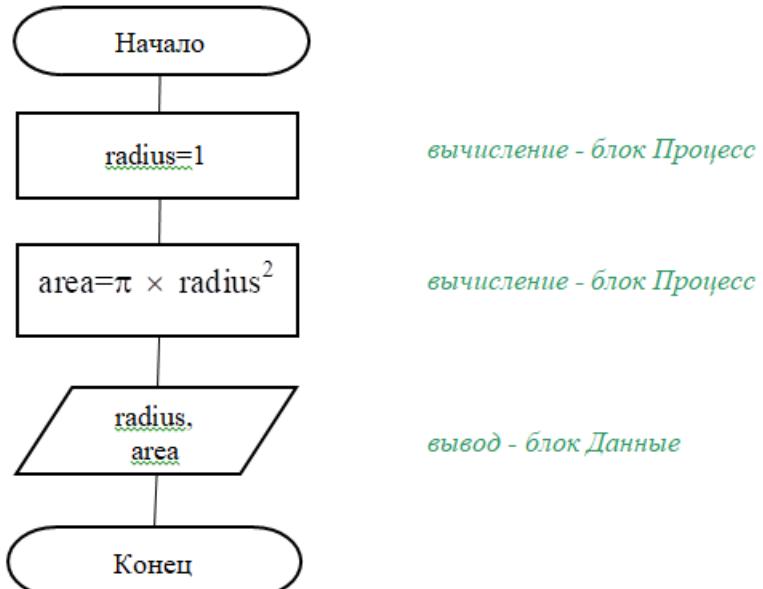
Результаты работы программы:

Ведите аргумент x 1

Значение функции y = -2.293

Значение функции y = -2.292957680198291

**Пример 14.** Вычислить площадь круга по известному радиусу.



```

from math import pi

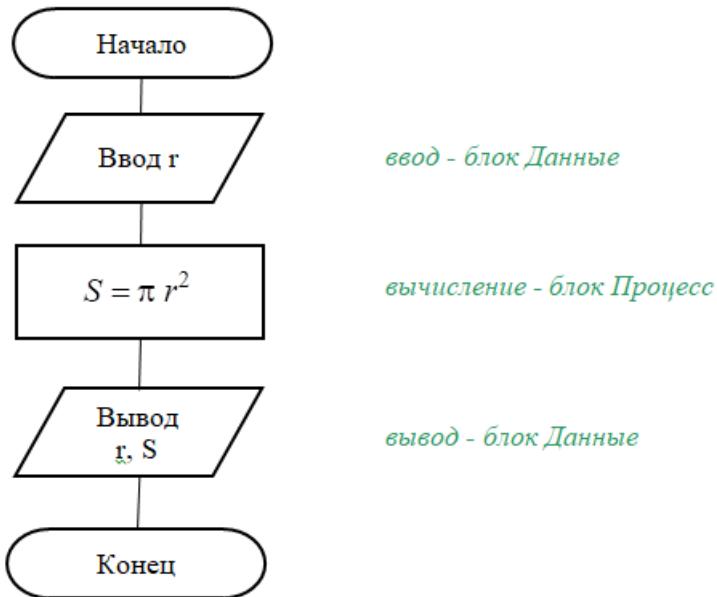
radius = 1
area = pi * radius ** 2

print("радиус окружности = ", radius)
print("Площадь круга = ", area)

>> радиус окружности = 1
>> Площадь круга = 3.141592653589793

```

**Пример 15.** Вычислить площадь круга по введому пользователем радиусу.



```

from math import pi

r = float(input(" ВВедите радиус круга " ))
S = pi * r ** 2

print(" Радиус окружности = ", r)
print(" Площадь круга = ", S)

>> Введите радиус круга 5.47
>> Радиус окружности = 5.47
>> Площадь круга = 93.99927962879482

>> Введите радиус круга 2.89
>> Радиус окружности = 2.89
>> Площадь круга = 26.238896002047312

```

## 5. СОЗДАНИЕ ПРИЛОЖЕНИЙ РАЗВЕТВЛЯЮЩЕЙ СТРУКТУРЫ

Структуру разветвляющегося алгоритма можно реализовать с помощью ветвления – выбора одного из двух вариантов действий.

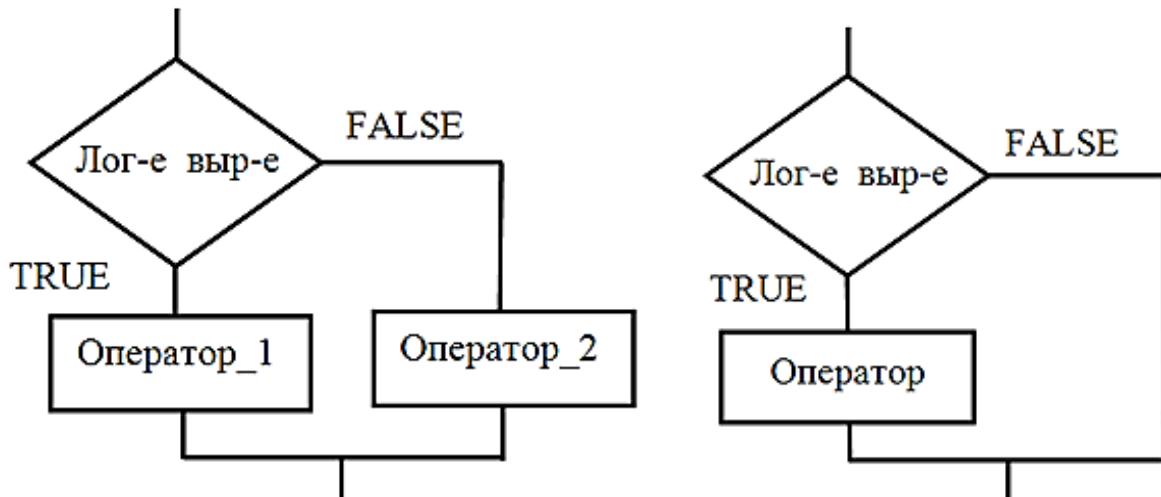


Рис. 7. Структуры операторов ветвления

### 5.1. Оператор `if`

Синтаксис оператора:

```
if Условие:  
    оператор 1  
    оператор 2  
    ...
```

Первая строчка оператора, то есть `if Условие:` — это условие `if`, а `Условие` — это логическое выражение, которое возвращает `True` или `False`. В следующей строке блок инструкций. Блок представляет собой одну или больше инструкций. Если он идет следом за условием `if`, такой блок называют блоком `if`.

Стоит обратить внимание, что у каждой инструкции в блоке `if` одинаковый отступ от слова `if`. Многие языки, такие как C, C++, Java и PHP, используют фигурные скобки (`{}`), чтобы определять начало и конец блока, но в Python используются отступы.

#### Пример 16.

```
number = int(input("Введите число: "))  
  
if number > 10:
```

```

print("Число больше 10")
# Ввод
>> Введите число: 100
# Вывод
>> Число больше 10

# Ввод
>> Введите число: 5
# Вывод
>>                               # ничего не выведет

```

### Пример 17.

```

number = int(input("Введите число: "))
if number > 10:
    print("первая строка")
    print("вторая строка")
    print("третья строка")

print("Выполняется каждый раз, когда вы запускаете
программу")
print("Конец")

# Ввод
>> Введите число: 45
# Вывод
>> первая строка
>> вторая строка
>> третья строка
>> Выполняется каждый раз, когда вы запускаете программу
>> Конец

# Ввод
>> Введите число: 4
# Вывод
>> Выполняется каждый раз, когда вы запускаете программу
>> Конец

```

## 5.2. Оператор **if-else**

Синтаксис оператора:

<b>if</b> Условие:	# блок if
оператор 1	
оператор 2	
...	

```
else:                                # блок else
    оператор 1
    оператор 2
...
...
```

Когда оператор `if-else` исполняется, условие проверяется, и если оно возвращает `True`, когда инструкции в блоке `if` исполняются. Но если возвращается `False`, тогда исполняются инструкции из блока `else`.

**Пример 18.** Вывести абсолютное значение (модуль) числа.

```
x = int(input())
if x > 0:
    print(x)
else:
    print(-x)

# Ввод
>> -4
# Вывод
>> 4
```

В этой программе используется условная инструкция `if` (если). После слова `if` указывается проверяемое условие (`x > 0`), завершающееся двоеточием. После этого идет блок (последовательность) инструкций, который будет выполнен, если условие истинно, в нашем примере это вывод на экран величины `x`. Затем идет слово `else` (иначе), также завершающееся двоеточием, и блок инструкций, который будет выполнен, если проверяемое условие неверно, в данном случае будет выведено значение `-x`.

**Пример 19.** Вывести абсолютное значение (модуль) числа без `else`.

```
x = int(input())
if x < 0:
    x = -x
print(x)
```

В этом примере переменной `x` будет присвоено значение `-x`, но только в том случае, когда `x<0`. А вот инструкция `print(x)` будет выполнена всегда, независимо от проверяемого условия.

**Пример 20. «Четное»-«нечетное»**

```
# Выведите строку 'четное', если введенное число четно, и
строку 'нечетное', если число нечетно.
```

```

a = int(input())
if a % 2 == 0:
    print('четное')
else:
    print('нечетное')
# Ввод
>> 10
# Вывод
>> четное

# Ввод
>> 11
# Вывод
>> нечетное

```

**Пример 21.** Пользователь вводит два числа: координаты шахматной клетки. Выведите YES, если клетка белая, и NO, если - черная.

```

x = int(input('Введите координату x: '))
y = int(input('Введите координату y: '))
# Идея такая: если четность координат совпадает,
# то это черная клетка, а если - нет, то белая.

if (x + y) % 2 == 1:
    print('YES')
else:
    print('NO')
# Ввод:
>> 1
>> 1
# Вывод:
>> NO

# Ввод:
>> 1
>> 1
# Вывод:
>> NO

```

**Пример 22.** Случайные координаты шахматной клетки. Выведите YES, если клетка белая, и NO, если - черная.

```

from random import randint # импортирование функции randint
x = randint(1,8) # функция randint вернет случайное число от 1 до 8

```

```

y = randint(1,8)
print(x, y) # вывод пары случайных чисел

if (x + y) % 2 == 1:
    print('YES')
else:
    print('NO')

```

### 5.3. Оператор `if` внутри другого `if`-оператора

**Пример 23.** Программа, проверяющая, имеет ли студент право на кредит.

```

gre_score = int(input("Введите текущий лимит: "))
per_grad = int(input("Введите кредитный рейтинг: "))

if per_grad > 70:
    # внешний блок if
    if gre_score > 150:
        # внутренний блок if
        print("Поздравляем, вам выдан кредит")
else:
    print("Извините, вы не имеете права на кредит")

# Ввод
>> Введите текущий лимит: 160
>> Введите кредитный рейтинг: 75
# Вывод
>> Поздравляем, вам выдан кредит

# Ввод
>> Введите текущий лимит: 160
>> Введите кредитный рейтинг: 60
# Вывод
>> Извините, вы не имеете права на кредит

# Ввод
>> Введите текущий лимит: 140
>> Введите кредитный рейтинг: 80
# Вывод
>>                                # не выведет ничего

```

Здесь оператор `if` используется внутри другого `if`-оператора. Внутренним называют вложенный оператором `if`. В этом случае внутренний оператор `if` относится к внешнему блоку `if`, а у

внутреннего блока `if` есть только одна инструкция, которая выводит “Поздравляем, вам выдан кредит”.

**Пример 24.** Программа, проверяющая, имеет ли студент право на кредит. Инструкция `if-else` внутри другого оператора `if`.

```
gre_score = int(input("Введите текущий лимит: "))
per_grad = int(input("Введите кредитный рейтинг: "))

if per_grad > 70:
    if gre_score > 150:
        print("Поздравляем, вам выдан кредит")
    else:
        print("У вас низкий кредитный лимит")
else:
    print("Извините, вы не имеете права на кредит")

# Ввод
>> Введите текущий лимит: 140
>> Введите кредитный рейтинг: 80
# Вывод
>> У вас низкий кредитный лимит
```

## 5.4. Оператор `if-else` внутри условия `else`

**Пример 25.** Программа для определения оценки студента на основе введенных баллов.

```
score = int(input("Введите вашу оценку: "))

if score >= 91:
    print("Отлично! Ваша оценка 5")
else:
    if score >= 74:
        print("Здорово! Ваша оценка - 4")
    else:
        if score >= 61:
            print("Хорошо! Ваша оценка - 3")
        else:
            print("Ваша оценка - 2. Вы не сдали экзамен")
```

## 5.5. Оператор `if-elif-else`

Синтаксис оператора:

```
if Условие_1:                      # блок if
    оператор 1
```

```

    оператор 2
    ...
elif Условие_2:           # первый блок elif
    оператор 1
    оператор 2
    ...
elif condition_3:         # второй блок elif
    оператор 1
    оператор 2
    ...
...
else:
    оператор 1
    оператор 2
    ...

```

Команда `if` может проверить только одно условие. Если необходимо осуществить передачу управления в зависимости от результатов проверки нескольких условий, то их можно задать с помощью оператора `elif`:

**Пример 26.** Переписать пример 25 с помощью `if-elif-else`.

```

score = int(input("Введите вашу оценку: "))

if score >= 91:
    print("Отлично! Ваша оценка 5")
elif score >= 74:
    print("Здорово! Ваша оценка - 4")
elif score >= 61:
    print("Хорошо! Ваша оценка - 3")
else:
    print("Ваша оценка - 2. Вы не сдали экзамен")

```

**Пример 27. Много `elif`'ов**

```

# Если в вашем коде больше трех elif, значит, что-то не
# так и стоит придумать новый алгоритм.
# Это сугубо академический пример, так писать не стоит.
a = int(input('Введите число от одного до 100'))
if a < 10:
    print('Ваше число меньше 10')
elif a < 20:
    print('Ваше число меньше двадцатки')
elif a < 30:
    print('30 - это потолок')

```

```

elif a < 40:
    print('Ваше число меньше 40')
elif a < 50:
    print('Много, но не больше полтинника')
elif a < 60:
    print('Число меньше, чем шесть*десять')
elif a < 70:
    print('Ваше число расположено в восьмом десятке')
elif a < 80:
    print('Ваше число меньше 80')
elif a < 90:
    print('Ваше число находится в 9 десятке')
elif a < 100:
    print('Сотня больше вашего числа')
else:
    print('Хитро, но меня не обманешь, число слишком
большое')

# Ввод
>> 11
# Вывод
>> Ваше число меньше двадцатки

# Ввод
>> 91
# Вывод
>> Сотня больше вашего числа

# Ввод
>> 58
# Вывод
>> Число меньше, чем шесть*десять

```

**Пример 28.** Написать программу, вычисляющую значение функции  $y = \sin(x) - \cos(x^2) + 2.25$ . Значение величины  $x$ , вводимое пользователем с клавиатуры, должно проверяться на корректность, т. е. должно принадлежать указанному в задании диапазону  $[2,5;3,5]$ .

```

from math import sin, cos

a = 2.5
b = 3.5
x = float(input("Введите аргумент x из диапазона
[2.5;3.5]: "))
if (x >= a) and (x <= b):

```

```

y = sin(x) - cos(x ** 2) + 2.25
# метод format
print("Значение функции y = ", "{:.3f}".format(y))
# без формата
print("Значение функции y = ", y)
else:
    print("Вы не попали в нужный диапазон")

>> Введите аргумент x из диапазона [2.5;3.5]: 2
>> Вы не попали в нужный диапазон

>> Введите аргумент x из диапазона [2.5;3.5]: 3
>> Значение функции y = 3.302
>> Значение функции y = 3.302250269944544

```

**Пример 29.** Составить алгоритм, находящий значение  $y$

$$y = \begin{cases} x & \text{при } x < 0 \\ 0 & \text{при } 0 \leq x < 30. \\ x^2 & \text{при } x \geq 30 \end{cases}$$

```
x = float(input('Введите значение x = '))
```

```

if x < 0:
    y = x
elif x < 30:
    y = 0
else:
    y = x ** 2
print('x = ', x, ' y = ', y)

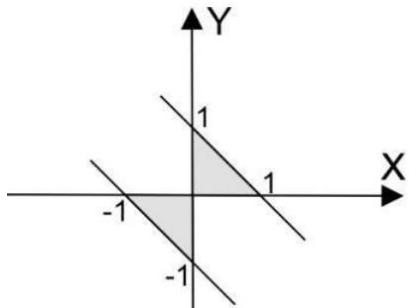
>> Введите значение x = -1
>> x = -1.0 y = -1.0

>> Введите значение x = 25
>> x = 25.0 y = 0

>> Введите значение x = 30
>> x = 30.0 y = 900.0

```

**Пример 30.** Координаты точки вводятся с клавиатуры. Определить, попадает ли точка в указанную область. Использовать оператор if.



Определим условие попадания точки в указанную область:

- 1) условие попадания в I или III квадрант  $x \cdot y > 0$ ;
- 2) условие попадания между двумя наклонными прямыми  
 $-(x + 1) < y < -(x - 1)$ .

```

x = float(input('Введите координату x : '))
y = float(input('Введите координату y : '))

if x * y >= 0 and y >= (-x - 1) and y <= (-x+1):
    print('Точка (', x, ', ', y, ') попадает в область')
else:
    print('Точка (', x, ', ', y, ') не попадает в
область')

>> Введите координату x : 1
>> Введите координату y : 1
>> Точка ( 1.0 , 1.0 ) не попадает в область

>> Введите координату x : 0.2
>> Введите координату y : 0.4
>> Точка ( 0.2 , 0.4 ) попадает в область

>> Введите координату x : -1
>> Введите координату y : 0
>> Точка ( -1.0 , 0.0 ) попадает в область

```

## 5.6. Оператор выбора

Структуру разветвляющегося алгоритма можно реализовать с помощью выбора одного варианта из нескольких (Рис. 8) в зависимости от значения некоторой величины (селектора).

Начиная с версии 3.10, в Python появилась конструкция `switch-case`, которая называется `match-case`.

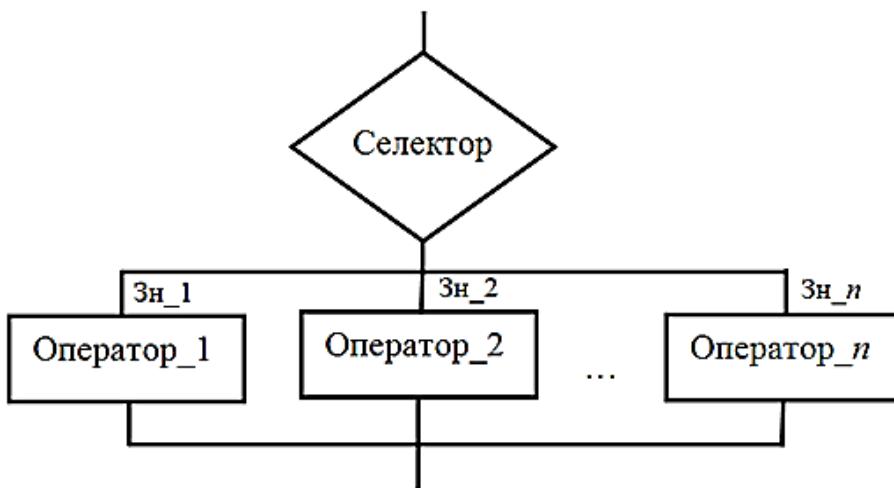


Рис. 8. Структура оператора выбора

**Синтаксис оператора:**

```

match селектор:
    case Значение1 : #Проверяется равенство значению1
        Оператор1 #Выполняется, если селектор=Значению1
    case Значение2 :
        Оператор2 #Выполняется, если селектор=Значению2
    ...
    case Значение N :
        Оператор N
    case _ :
        Оператор # Выполняется, если значение селектора
не равно ни одному из указанных

```

Выполнение оператора начинается с вычисления выражения (селектора). Самый простой вариант использования `match-case` – это сопоставление с литеральными шаблонами. Литерал, с которым можно сравнивать, может быть: числом, строкой, `None`, `True`, `False`. Затем управление передается первому оператору из списка `case`, помеченному константным выражением, значение которого совпало с вычисленным значением селектора.

Если ни одного совпадения не произошло, выполняются операторы, расположенные после слова `case _` (а при его отсутствии управление передается следующему за `match` оператору).

### Пример 31.

```

x = int(input('Введите значение x = {1,2,3} '))
match x:
    case 1:
        print('x = 1')

```

```

case 2:
    print('x = 2')
case 3:
    print('x = 3')
case _:
    print('Некорректное значение.')

>> Введите значение x = {1,2,3}1
>> x = 1

>> Введите значение x = {1,2,3} 5
>> Некорректное значение.

```

### **Пример 32. Светофор**

```

color = input('Введите цвет сигнала светофора : ')
match color:
    case 'красный' | 'жёлтый':
        print('Стоп.')
    case 'зелёный':
        print('Можно ехать.')
    case _:
        print('Некорректное значение.')

>> Введите цвет сигнала светофора красный
>> Стоп.

>> Введите цвет сигнала светофора синий
>> Некорректное значение.

>> Введите цвет сигнала светофора зелёный
>> Можно ехать.

```

### **Пример 33. Создание калькулятора:**

```

a = float(input('Введите первое число '))
b = float(input('Введите второе число '))
x = int(input('Выберите действие: 1-сложение; 2-вычитание; 3-
умножение; 4-деление: '))
match x:
    case 1:
        print(a + b)
    case 2:
        print(a - b)
    case 3:
        print(a * b)
    case 4:
        if b == 0:
            print('На нуль делить нельзя!')

```

```
else:
    print(a / b)
case _:
    print('Некорректное значение')

>> Введите первое число 5
>> Введите второе число 6
>> Выберите действие: 1-сложение; 2-вычитание; 3-умножение;
4-деление: 1
>> 11.0

>> Введите первое число 5
>> Введите второе число 6
>> Выберите действие: 1-сложение; 2-вычитание; 3-умножение;
4-деление: 2
>> -1.0

>> Введите первое число 5
>> Введите второе число 6
>> Выберите действие: 1-сложение; 2-вычитание; 3-умножение;
4-деление: 3
>> 30.0

>> Введите первое число 5
>> Введите второе число 6
>> Выберите действие: 1-сложение; 2-вычитание; 3-умножение;
4-деление: 4
>> 0.833333333333334

>> Введите первое число 5
>> Введите второе число 0
>> Выберите действие: 1-сложение; 2-вычитание; 3-умножение;
4-деление: 4
>> На нуль делить нельзя!

>> Введите первое число 5
>> Введите второе число 6
>> Выберите действие: 1-сложение; 2-вычитание; 3-умножение;
4-деление: 10
>> Некорректное значение
```

## **6. ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ**

### **6.1. Ввод-вывод**

**Задание 1.** Напишите программу, которая запрашивала бы у пользователя данные, а потом выводила их.

<p>Вариант 1</p> <p><b>Имя, Фамилия, Возраст, Место жительства</b></p> <ul style="list-style-type: none"> <li>- фамилия, имя ("Ваши фамилия, имя?")</li> <li>- возраст ("Сколько Вам лет?")</li> <li>- место жительства ("Где вы живете?")</li> </ul> <p>После этого выводила бы три строки: "Ваши фамилия, имя" "Ваш возраст" "Вы живете в"</p>	<p>Вариант 2</p> <p><b>Имя, Дата рождения, Образование</b></p> <ul style="list-style-type: none"> <li>- имя ("Ваше, имя?")</li> <li>- дата рождения ("Ваша дата рождения?")</li> <li>- образование ("Где Вы учитесь?")</li> </ul> <p>После этого выводила бы три строки: "Ваше имя" "Дата рождения" "Вы учитесь в "</p>
<p>Вариант 3</p> <p><b>Фамилия, Место жительства</b></p> <ul style="list-style-type: none"> <li>- Фамилия ("Ваша фамилия?")</li> <li>- место жительства ("Где Вы живете?")</li> </ul> <p>После этого выводила бы две строки: "Ваша фамилия" "Вы живете в"</p>	<p>Вариант 4</p> <p><b>Фамилия, Место рождения, любимая музыка</b></p> <ul style="list-style-type: none"> <li>- Фамилия, ("Ваша фамилия?")</li> <li>- место рождения ("Где Вы родились?")</li> <li>- музыка ("Какая музыка нравится? ")</li> </ul> <p>После этого выводила бы три строки: "Ваши имя, фамилия" "Вы родились в" "Ваша любимая музыка "</p>
<p>Вариант 5</p> <p><b>Имя, Фамилия, ФИО мамы, ФИО отца</b></p> <ul style="list-style-type: none"> <li>- ФИО (например, "Ваши фамилия, имя, отчество?")</li> <li>- возраст ("Сколько Вам лет?")</li> <li>- место жительства ("Где Вы живете?")</li> </ul> <p>После этого выводила бы три строки: "Ваши имя, фамилия, отчество"</p>	<p>Вариант 6</p> <p><b>Имя, Любимый предмет в ВУЗе, год обучения</b></p> <ul style="list-style-type: none"> <li>- имя ("Ваше имя?")</li> <li>- любимый предмет ("Какой Ваш любимый предмет в ВУЗе?")</li> <li>- номер класса ("На каком курсе Вы учитесь?")</li> </ul> <p>После этого выводила бы три строки:</p>

<p>"Ваш возраст" "Вы живете в"</p>	<p>"Ваше имя" "Ваш любимый предмет в ВУЗе" "Вы учитесь на курсе "</p>
<p>Вариант 7</p> <p><b>Имя, Фамилия, Отчество, Хобби</b>            - ФИО (например, "Ваши фамилия, имя, отчество?")            - хобби ("Чем Вы увлекаетесь?")</p> <p>После этого выводила бы две строки:            "Ваши имя, фамилия, отчество"            "Ваше хобби"</p>	<p>Вариант 8</p> <p><b>Имя, Фамилия, отделение</b>            - Фамилия, имя ("Ваши фамилия, имя?")            - образование ("В каком ВУЗе Вы учитесь?")            ("На каком отделении Вы учитесь?")</p> <p>После этого выводила бы три строки:            "Ваши имя, фамилия"            "Вы учитесь в "            "Вы учитесь на отделении "</p>
<p>Вариант 9</p> <p><b>Имя, Фамилия, Любимый предмет в институте, ФИО куратора</b>            - Фамилия, имя ("Ваши фамилия, имя?")            - любимый предмет в ВУЗе ("Какой Ваш любимый предмет в ВУЗе?")            - ФИО куратора ("ФИО Вашего куратора?")</p> <p>После этого выводила бы три строки:            "Ваши имя, фамилия"            "Ваш любимый предмет в ВУЗе "            "ФИО Вашего куратора"</p>	<p>Вариант 10</p> <p><b>Имя, Фамилия, Возраст, Дата рождения</b>            - Фамилия, имя ("Ваши фамилия, имя?")            - возраст ("Сколько Вам лет?")            - дата рождения ("Когда Вы родились?")</p> <p>После этого выводила бы три строки:            "Ваши имя, фамилия"            "Ваш возраст"            "Дата Вашего рождения"</p>
<p>Вариант 11</p> <p><b>Имя, Фамилия, Место жительства, Месторождения</b>            - Фамилия, имя ("Ваши фамилия, имя?")            - место рождения ("Где Вы родились?")            - место жительства ("Где Вы живете?")</p> <p>После этого выводила бы три строки:            "Ваши имя, фамилия"            "Вы родились в "</p>	<p>Вариант 12</p> <p><b>Имя, Фамилия, Возраст, Номер телефона</b>            - Фамилия, имя ("Ваши фамилия, имя?")            - возраст ("Сколько тебе лет?")            - номер телефона ("Номер Вашего телефона?")</p> <p>После этого выводила бы три строки:            "Ваши имя, фамилия"            "Ваш возраст"</p>

"Вы живете в"	"Ваш номер телефона"
Вариант 13	Вариант 14
<b>Имя, Фамилия, Страна, Край, Город</b> - Фамилия, имя ("Ваши фамилия, имя?") - страна ("В какой стране Вы живете?") - город ("В каком городе Вы живете?")  После этого выводила бы три строки: "Ваши имя, фамилия" "Вы живете в стране" "Вы живете в крае" "Вы живете в городе"	<b>Имя, Фамилия, Любимый фильм</b> - Фамилия, имя ("Ваши фамилия, имя?") - страна ("В какой стране Вы живете?") - фильм ("Какой любимый фильм?")  После этого выводила бы три строки: "Ваши имя, фамилия" "Вы живете в стране" "Ваш любимый фильм"
Вариант 15	Вариант 16
<b>Имя, Фамилия, Республика, Город</b> - Фамилия, имя ("Ваши фамилия, имя?") - Республика ("В какой республике Вы живете?") - город ("В каком городе Вы живете?")  После этого выводила бы три строки: "Ваши имя, фамилия" "Вы живете в республике" "Вы живете в городе"	<b>Имя, Фамилия, Любимый актер</b> - Фамилия, имя ("Ваши фамилия, имя?") - страна ("В какой стране Вы живете?") - актер ("Какой любимый актер?")  После этого выводила бы три строки: "Ваши имя, фамилия" "Вы живете в стране" "Ваш любимый актер"
Вариант 17	Вариант 18
<b>Имя, Фамилия, Город, Дом</b> - Фамилия, имя ("Ваши фамилия, имя?") - город ("В каком городе Вы живете?") - номер дома ("Укажите номер Вашего дома?")  После этого выводила бы три строки: "Ваши имя, фамилия" "Вы живете в городе" "Вы живете в доме номер"	<b>Имя, Фамилия, Любимый писатель</b> - Фамилия, имя ("Ваши фамилия, имя?") - страна ("В какой стране Вы живете?") - писатель ("Кто любимый писатель?")  После этого выводила бы три строки: "Ваши имя, фамилия" "Вы живете в стране" "Ваш любимый писатель "

## 6.2. Линейные конструкции. Задания первого уровня сложности

**Задание 2.** Вычислите значение функции  $y = f(x)$  для заданного аргумента  $x$  из отрезка  $[a,b]$  (проверять попадание в отрезок не нужно). Результат выведите без формата и по формату.

№ вар.	Функция	Диапазон изменения аргумента $[a,b]$
1.	$y =  x^2 - 5x + 6  - 0.5$	[1,4]
2.	$y = 2 \cos^2 x + 5 \sin x + 1$	[3,6]
3.	$y = \cos x - e^{-\frac{x^2}{2}} + x - 1$	[1;2]
4.	$y = (x^2 - 4x + 4)^{x-1.5} - 1$	[0,2]
5.	$y = \sin^4 x - \cos^4 x - 0.5$	[0,3]
6.	$y = \sqrt{3} \sin 3x + \cos 3x - 1$	[0,2,5]
7.	$y = e^x \sin x$	[0,5]
8.	$y = 0,1x^2 - x \ln x$	[1;2]
9.	$y = x - 1/(3 + \sin 3,6x)$	[0;0,85]
10.	$y =  x  - 0.5$	[-4,4]
11.	$y = \lg(x) + \sqrt[3]{x} - 1.56$	[1,3]
12.	$y = x + \cos(x^{0,52} + 2)$	[0,5;1]

<b>№ вар.</b>	<b>Функция</b>	<b>Диапазон изменения аргумента [a,b]</b>
13.	$y = \frac{\sin x}{x} + 0.03$	[5,7]
14.	$y = (\sqrt{3x})^{x-1} - 1$	[0,2;2]
15.	$y = 3 \cdot 2^{2x} + 3^{2x+3} - 10$	[-1,1]
16.	$y = 27^{\frac{2x-1}{x}} - \sqrt{9^{2x-1}}$	[2,4]
17.	$y = \lg^2 x - \lg x^3 + 2$	[1,10]
18.	$y = \sqrt{10 - 18 \cos x} - 6 \cos x - 5$	[\pi/2,3\pi/2]
19.	$y = 3 \sin \sqrt{x} + 0,35x - 3,8$	[2,3]
20.	$y = x + \sqrt{x} + \sqrt[3]{x} - 2,5$	[0,4;1]
21.	$y = \sin(\ln x) - \cos(\ln x) + 2 \ln x$	[1;3]
22.	$y = \cos \frac{2}{x} - 2 \sin \frac{1}{x} + \frac{1}{x}$	[1;2]
23.	$y = 3x - 4 \ln x - 5$	[2;4]
24.	$y = \sqrt{1-x} - \cos \sqrt{1-x}$	[0;1]
25.	$y = 3 \ln^2 x + 6 \ln x - 5$	[1;3]
26.	$y = 3x - 14 + e^x - e^{-x}$	[1;3]
27.	$y = \sqrt{1-x} - \operatorname{tg} x$	[0;1]

<i>№ вар.</i>	<i>Функция</i>	<i>Диапазон изменения аргумента [a,b]</i>
28.	$y = \operatorname{tg}x - \frac{1}{3}\operatorname{tg}^3x + \frac{1}{5}\operatorname{tg}^5x - \frac{1}{3}$	[0;0,8]
29.	$y = \sin x - \cos x^2 + 0,25$	[2,5;3,5]
30.	$y = e^x \cos x$	[0,5]
31.	$y =  x - 1  - 0,5$	[-5,5]

**Задание 3.** В соответствии с номером варианта выполнить индивидуальное задание.

<i>№ вар.</i>	<i>Формулировка задачи</i>
1.	Вычислить площадь и периметр прямоугольника, если задана длина одной стороны ( $a$ ) и коэффициент $n$ (%), позволяющий вычислить длину второй стороны ( $b=n*a$ ).
2.	Дана длина ребра куба. Найти площадь грани, площадь полной поверхности и объем этого куба.
3.	Вычислить периметр и площадь прямоугольного треугольника по заданным длинам двух катетов $a$ и $b$ .
4.	Вычислить площади геометрических фигур: прямоугольника и треугольника по заданным сторонам.
5.	По известному радиусу вычислить объем и площадь поверхности шара.
6.	Даны два числа. Найти среднее арифметическое кубов этих чисел и среднее геометрическое модулей этих чисел.
7.	Известен объем информации в байтах. Выразить его в мегабайтах, гигабайтах и битах.
8.	Длина выражена в сантиметрах. Выразить ее в метрах, миллиметрах, дюймах (1 дюйм = 2,5 см).
9.	Перевести значение веса, заданное в граммах, в килограммы, тонны, унции (1 унция = 28,3 грамма).

<i>№ вар.</i>	<i>Формулировка задачи</i>
10.	Вычислить путь, пройденный лодкой по течению и против течения, если известна ее скорость в стоячей воде, скорость течения реки и время движения.
11.	Вычислить площади геометрических фигур: трапеции и круга.
12.	Заданы длины трех сторон треугольника. Вычислить его периметр и площадь.
13.	Дана сторона равностороннего треугольника. Найти площадь этого треугольника и его высоту.
14.	Дана сторона равностороннего треугольника. Найти радиусы вписанной и описанной окружностей.
15.	Вычислить объем и площадь полной поверхности цилиндра, если известны высота и радиус основания.
16.	Заданы стороны прямоугольника. Определить его периметр, площадь и длину диагонали.
17.	Заданы длина, ширина и высота прямоугольного параллелепипеда. Определить объем и площадь поверхности параллелепипеда.
18.	Переменной А присвоить ее значение, увеличенное в N раз, в $2N$ раз, в $3N$ раз.
19.	Заданы три положительных числа. Вычислить их среднее арифметическое и среднее геометрическое.
20.	Известен объем информации в байтах. Выразить его в килобайтах, мегабайтах и битах.
21.	Известен объем информации в килобайтах. Выразить его в мегабайтах, гигабайтах и битах.
22.	Известен объем информации в мегабайтах. Выразить его в байтах, гигабайтах и килобайтах.
23.	Вычислить время движения, затраченное лодкой при движении по течению и против течения, если известна ее скорость в стоячей воде, скорость течения реки и расстояние.
24.	Задана длина стороны куба. Определить объем, площадь поверхности и диагональ.

<i>№ вар.</i>	<i>Формулировка задачи</i>
25.	Задана сторона ромба и его высота. Определить его периметр и площадь.
26.	Задана сторона ромба и один из его углов. Определить периметр и площадь ромба.
27.	По известному радиусу вычислить объем и площадь поверхности шара.
28.	Переменной А присвоить ее значение, увеличенное в N раз, в $2N$ раз, в $3N$ раз.
29.	Заданы длина, ширина и высота прямоугольного параллелепипеда. Определить объем и площадь поверхности параллелепипеда.
30.	Длина выражена в сантиметрах. Выразить ее в метрах, миллиметрах, дюймах (1 дюйм = 2,5 см).
31.	Дана сторона равностороннего треугольника. Найти площадь этого треугольника и его высоту.

### 6.3. Линейные конструкции. Задания второго уровня сложности

Создать пользовательские функции.

**Задание 4.** Исходные данные вводятся с клавиатуры. Вычислить значения функций.

<i>№ вар.</i>	<i>Расчетные формулы</i>	<i>Исходные данные</i>
1.	$a = \frac{2 \cos(x - \pi / 6)}{1/2 + \sin^2 y};$ $b = 1 + \frac{z^2}{3 + z^2 / 5}.$	$x = 1,426$ $y = -1,22$ $z = 3,5$
2.	$\gamma = \left  x^{y/x} - \sqrt[3]{y/x} \right ;$ $\varphi = (y-x) \frac{y-z/(y-x)}{1+(y-x)^2}.$	$x = 1,825$ $y = 18,225$ $z = -3,298$

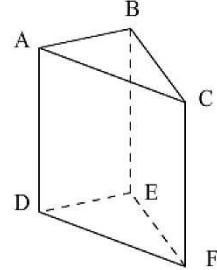
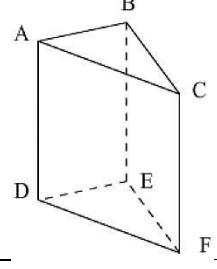
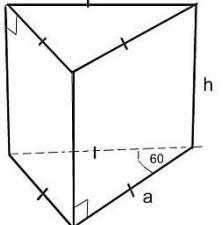
<i>№ вар.</i>	<i>Расчетные формулы</i>	<i>Исходные данные</i>
3.	$s = \frac{e^x + \cos^2 e^y}{(x+y)^2};$ $z = \sqrt{(x+y)}(\sin x^3 + \cos^2 y).$	$x = 0,335$ $y = 0,025$
4.	$y = e^{-bt} \sin(at+b) - \sqrt{ bt+a };$ $s = b \sin(at^2 \cos 2t) - 1.$	$a = -0,5$ $b = 1,7$ $t = 0,44$
5.	$\omega = \sqrt{x^2 + b} - b^2 \sin^3(x+a)/x;$ $y = \cos^2 x^3 - x/\sqrt{a^2 + b^2}.$	$a = 1,5$ $b = 15,5$ $x = -2,9$
6.	$s = x^3 \operatorname{tg}^2(x+b)^2 + a/\sqrt{x+b};$ $q = \frac{bx^2 - a}{e^{ax} - 1}.$	$a = 16,5$ $b = 3,4$ $x = 0,61$
7.	$R = x^2(x+1)/b - \sin^2(x+a);$ $s = \sqrt{xb/a} + \cos^2(x+b)^2.$	$a = 0,7$ $b = 0,05$ $x = 0,5$
8.	$y = \sin^3(x^2 + a)^2 - \sqrt{x/b};$ $z = \frac{x^2}{a} + \cos(x+b)^3.$	$a = 1,1$ $b = 0,004$ $x = 0,2$
9.	$f = \sqrt[3]{mctb +  c \sin t };$ $z = m \cos(bt \sin t) + c.$	$m = 2$ $c = 1$ $t = 1,2$ $b = 0,7$
10.	$y = abx^2 - \frac{a}{\sin^2(x/a)};$ $d = ae^{-\sqrt{a}} \cos(bx/a).$	$a = 3,2$ $b = 17,5$ $x = -4,8$
11.	$f = \ln(a + x^2) + \sin^2(x/b);$ $z = e^{-cx} \frac{x + \sqrt{x+a}}{x - \sqrt{ x-b }}.$	$a = 10,2$ $b = 9,2$ $c = 0,5$ $x = 2,2$

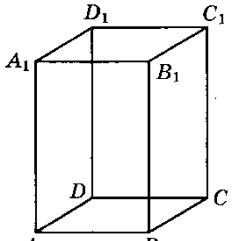
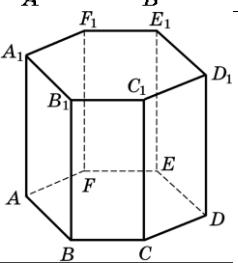
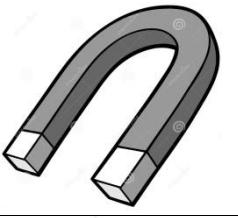
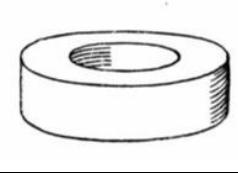
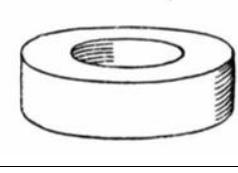
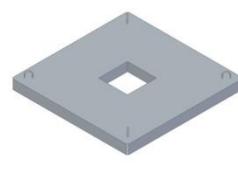
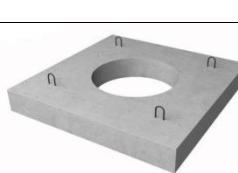
<i>№ вар.</i>	<i>Расчетные формулы</i>	<i>Исходные данные</i>
12.	$y = \frac{a^{2x} + b^{-x} \cos(a+b)x}{x+1};$ $R = \sqrt{x^2 + b - b^2 \sin^3(x+a)/x}.$	$a = 0,3$ $b = 0,9$ $x = 0,61$
13.	$z = \sqrt{ax \sin 2x + e^{-2x}(x+b)};$ $\omega = \cos^2 x^3 - x / \sqrt{a^2 + b^2}.$	$a = 0,5$ $b = 3,1$ $x = 1,4$
14.	$U = \frac{a^2 x + e^{-x} \cos bx}{bx - e^{-x} \sin bx + 1};$ $f = e^{2x} \ln(a+x) - b^{3x} \ln(b-x).$	$a = 0,5$ $b = 2,9$ $x = 0,3$
15.	$z = \frac{\sin x}{\sqrt{m^2 + \sin^2 x}} - cm \ln mx;$ $s = e^{-ax} \sqrt{x+1} + e^{-bx} \sqrt{x+1.5}.$	$m = 0,5$ $c = 1,5$ $x = 2$ $a = 1,5$ $b = 1,3$
16.	$a = \frac{2 \cos(x - \pi/6)}{1/2 + \sin^2 y};$ $b = 1 + \frac{z^2}{3 + z^2/5}.$	$x = 1,426$ $y = -1,22$ $z = 3,5$
17.	$\gamma = \left  x^{y/x} - \sqrt[3]{y/x} \right ;$ $\varphi = (y-x) \frac{y-z/(y-x)}{1+(y-x)^2}.$	$x = 1,825$ $y = 18,225$ $z = -3,298$
18.	$s = \frac{e^x + \cos^2 e^y}{(x+y)^2};$ $z = \sqrt{(x+y)} (\sin x^3 + \cos^2 y).$	$x = 0,335$ $y = 0,025$
19.	$y = e^{-bt} \sin(at+b) - \sqrt{ bt+a };$ $s = b \sin(at^2 \cos 2t) - 1.$	$a = -0,5$ $b = 1,7$ $t = 0,44$

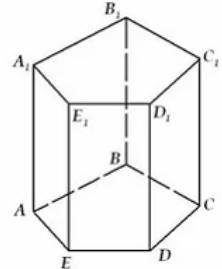
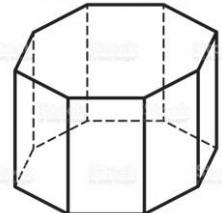
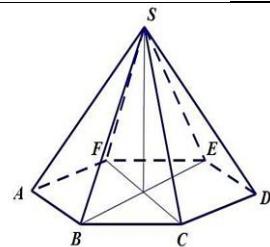
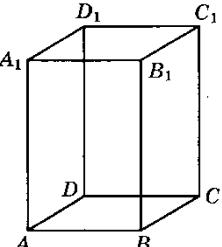
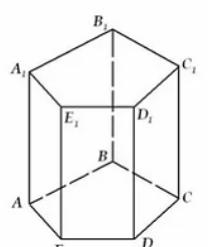
<i>№ вар.</i>	<i>Расчетные формулы</i>	<i>Исходные данные</i>
20.	$\omega = \sqrt{x^2 + b} - b^2 \sin^3(x+a)/x;$ $y = \cos^2 x^3 - x/\sqrt{a^2 + b^2}.$	$a = 1,5$ $b = 15,5$ $x = -2,9$
21.	$s = x^3 \operatorname{tg}^2(x+b)^2 + a/\sqrt{x+b};$ $q = \frac{bx^2 - a}{e^{ax} - 1}.$	$a = 16,5$ $b = 3,4$ $x = 0,61$
22.	$R = x^2(x+1)/b - \sin^2(x+a);$ $s = \sqrt{xb/a} + \cos^2(x+b)^2.$	$a = 0,7$ $b = 0,05$ $x = 0,5$
23.	$y = \sin^3(x^2 + a)^2 - \sqrt{x/b};$ $z = \frac{x^2}{a} + \cos(x+b)^3.$	$a = 1,1$ $b = 0,004$ $x = 0,2$
24.	$f = \sqrt[3]{mctb +  c \sin t };$ $z = m \cos(bt \sin t) + c.$	$m = 2$ $c = 1$ $t = 1,2$ $b = 0,7$
25.	$y = abx^2 - \frac{a}{\sin^2(x/a)};$ $d = ae^{-\sqrt{a}} \cos(bx/a).$	$a = 3,2$ $b = 17,5$ $x = -4,8$
26.	$f = \ln(a + x^2) + \sin^2(x/b);$ $z = e^{-cx} \frac{x + \sqrt{x+a}}{x - \sqrt{ x-b }}.$	$a = 10,2$ $b = 9,2$ $c = 0,5$ $x = 2,2$
27.	$y = \frac{a^{2x} + b^{-x} \cos(a+b)x}{x+1};$ $R = \sqrt{x^2 + b} - b^2 \sin^3(x+a)/x.$	$a = 0,3$ $b = 0,9$ $x = 0,61$
28.	$z = \sqrt{ax \sin 2x + e^{-2x}(x+b)};$ $\omega = \cos^2 x^3 - x/\sqrt{a^2 + b^2}.$	$a = 0,5$ $b = 3,1$ $x = 1,4$

<i>№ вар.</i>	<i>Расчетные формулы</i>	<i>Исходные данные</i>
29.	$U = \frac{a^2 x + e^{-x} \cos bx}{bx - e^{-x} \sin bx + 1};$ $f = e^{2x} \ln(a + x) - b^{3x} \ln(b - x).$	$a = 0,5$ $b = 2,9$ $x = 0,3$
30.	$z = \frac{\sin x}{\sqrt{m^2 + \sin^2 x}} - cm \ln mx;$ $s = e^{-ax} \sqrt{x+1} + e^{-bx} \sqrt{x+1.5}.$	$m = 0,5$ $c = 1,5$ $x = 2$ $a = 1,5$ $b = 1,3$
31.	$U = \frac{a^2 x + e^{-x} \cos bx}{bx - e^{-x} \sin bx + 1};$ $f = e^{2x} \ln(a + x) - b^{3x} \ln(b - x).$	$a = 0,5$ $b = 2,9$ $x = 0,3$

**Задание 5.** Определить площадь поверхности, объем и массу изделия в виде трехмерного объекта в соответствии с вариантом задания.

<i>№ вар.</i>	<i>Форма изделия</i>	<i>Входные данные</i>	<i>Материал</i>	
1.	Треугольная призма	Длина стороны, величина прилежащих углов, высота призмы	Сталь	
2.	Треугольная призма	Длины двух сторон, угол между ними, высота призмы	Сталь	
3.	Правильная треугольная призма	Радиус описанной окружности, высота	Стекло	

4.	Правильная четырехугольная призма	Радиус описанной окружности, высота	Стекло	
5.	Правильная шестиугольная призма	Радиус описанной окружности, высота	Стекло	
6.	U-образный магнит	Внешний и внутренний радиусы, внешние габаритные размеры	Сталь	
7.	Кольцо с образующей в форме окружности	Внешний и внутренний радиусы кольца	Резина	
8.	Кольцо с образующей в форме окружности	Внешний радиус, радиус образующей	Резина	
9.	Кольцо с образующей в форме прямоугольника	Внешний радиус, высота и толщина кольца	Бетон	
10.	Кольцо с образующей в форме прямоугольника	Внешний и внутренний радиусы, высота кольца	Бетон	
11.	Плита с прямоугольным отверстием	Внешние габаритные размеры плиты, внутренние размеры отверстия	Бетон	
12.	Плита с круглым отверстием	Внешние габаритные размеры плиты, радиус отверстия	Бетон	

13.	Правильная пятиугольная призма	Радиус описанной окружности, высота	Стекло	
14.	Правильная восьмиугольная призма	Радиус описанной окружности, высота	Сталь	
15.	Правильная шестиугольная пирамида	Радиус описанной окружности около основания, высота	Стекло	
16.	Правильная четырехугольная призма	Длина стороны основания, высота	Стекло	
17.	Правильная пятиугольная призма	Длина стороны основания, высота	Стекло	

#### 6.4. Разветвленные конструкции. Задания первого уровня сложности

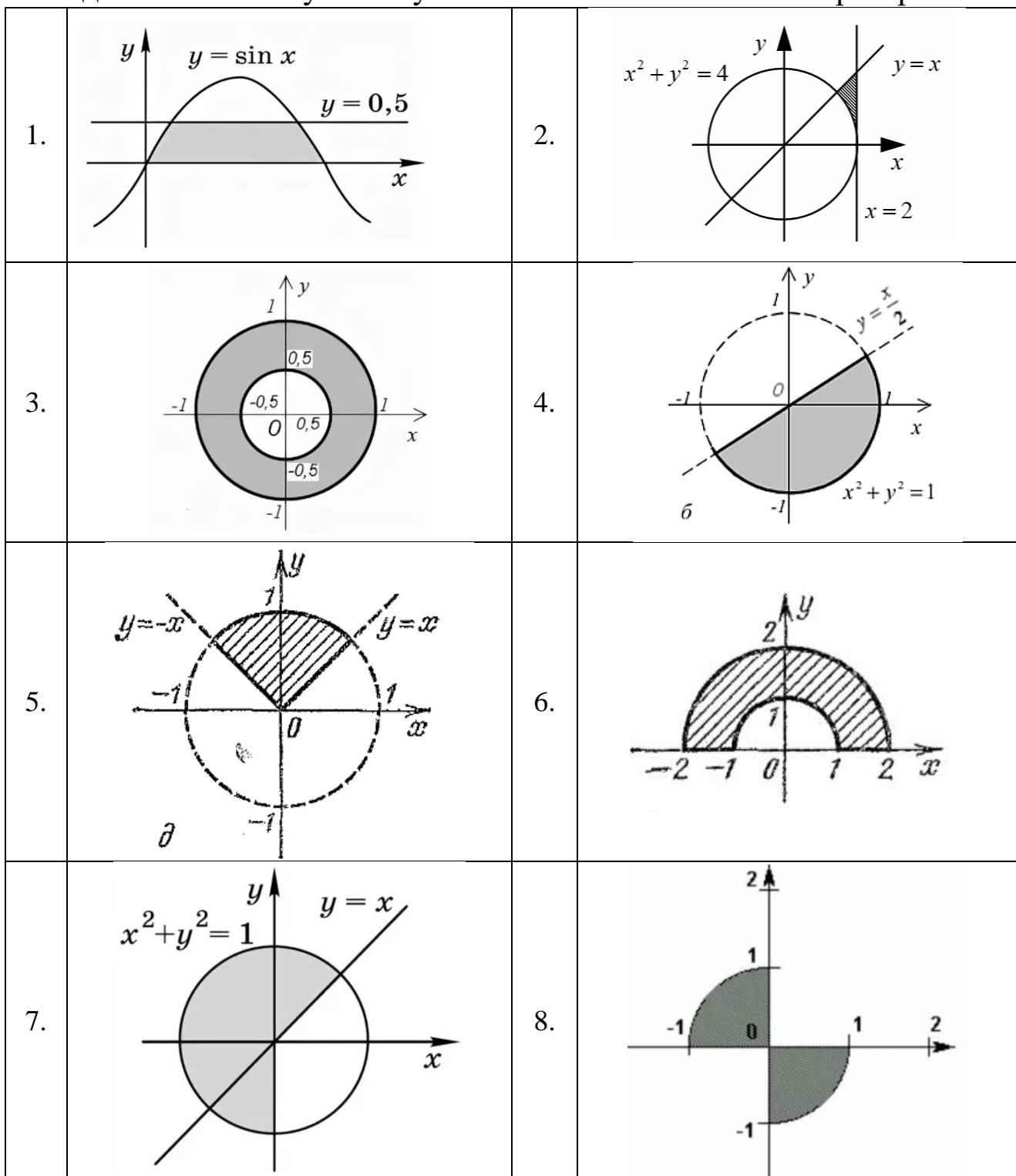
**Задание 1.** Написать программу, вычисляющую значение функции  $y = f(x)$ . Значение величины  $x$ , вводимое пользователем с клавиатуры, должно проверяться на корректность, т. е. должно принадлежать указанному в задании диапазону  $[a, b]$ .

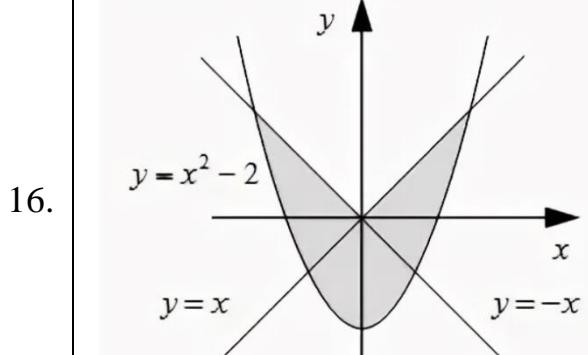
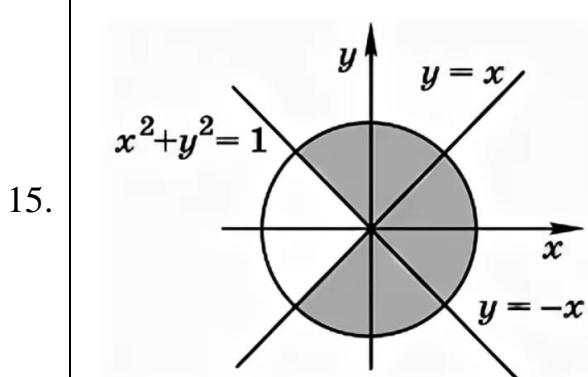
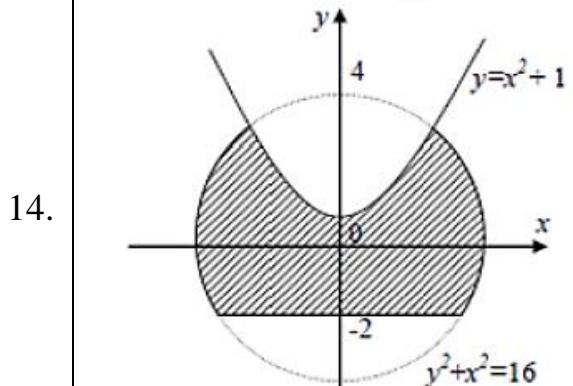
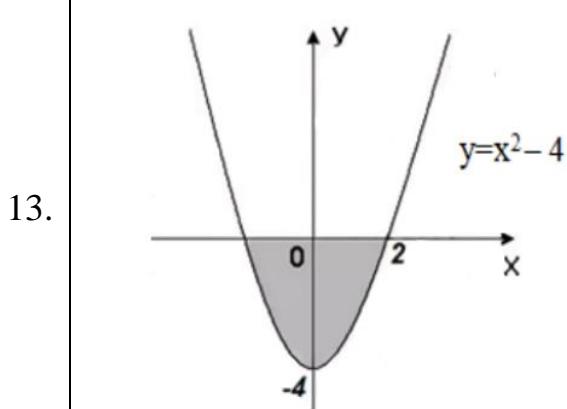
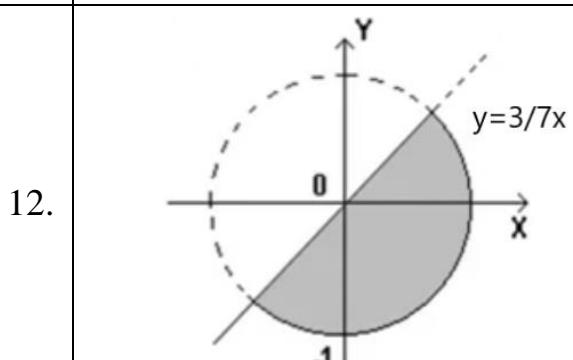
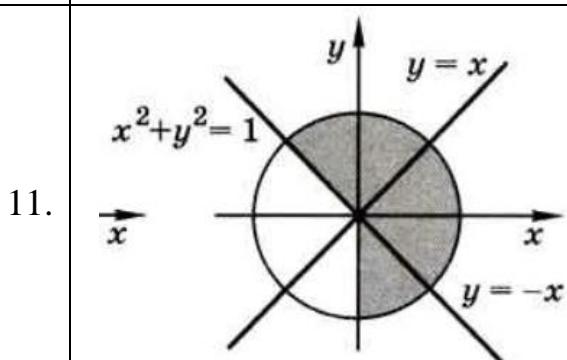
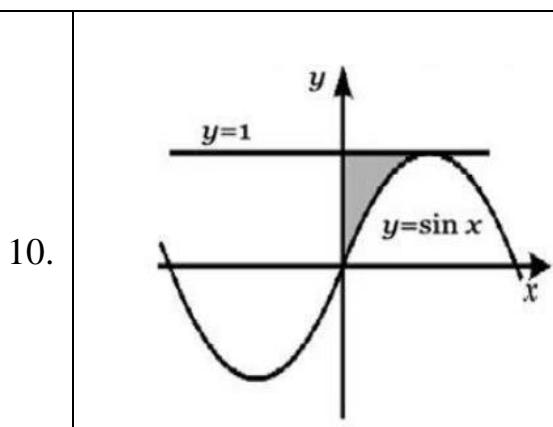
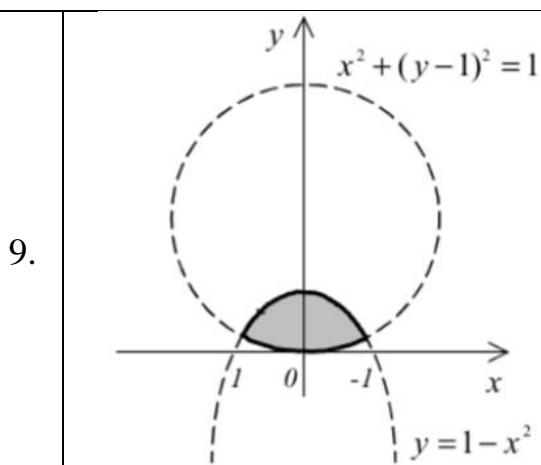
<b>№вар.</b>	<b>Функция</b>	<b>Диапазон изменения аргумента [a,b]</b>
1.	$y =  x^2 - 5x + 6  - 0.5$	[1,4]
2.	$y = 2 \cos^2 x + 5 \sin x + 1$	[3,6]
3.	$y = \cos x - e^{-\frac{x^2}{2}} + x - 1$	[1;2]
4.	$y = (x^2 - 4x + 4)^{x-1.5} - 1$	[0,2]
5.	$y = \sin^4 x - \cos^4 x - 0.5$	[0,3]
6.	$y = \sqrt{3} \sin 3x + \cos 3x - 1$	[0,2,5]
7.	$y = e^x \sin x$	[0,5]
8.	$y = 0,1x^2 - x \ln x$	[1;2]
9.	$y = x - 1/(3 + \sin 3,6x)$	[0;0,85]
10.	$y =  x  - 0.5$	[-4,4]
11.	$y = \lg(x) + \sqrt[3]{x} - 1.56$	[1,3]
12.	$y = x + \cos(x^{0,52} + 2)$	[0,5;1]
13.	$y = \frac{\sin x}{x} + 0.03$	[5,7]
14.	$y = (\sqrt{3x})^{x-1} - 1$	[0,2;2]
15.	$y = 3 \cdot 2^{2x} + 3^{2x+3} - 10$	[-1,1]

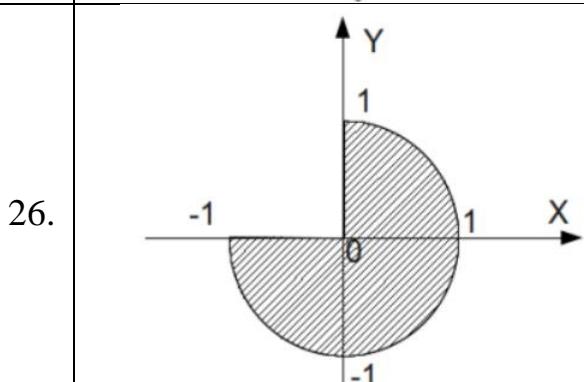
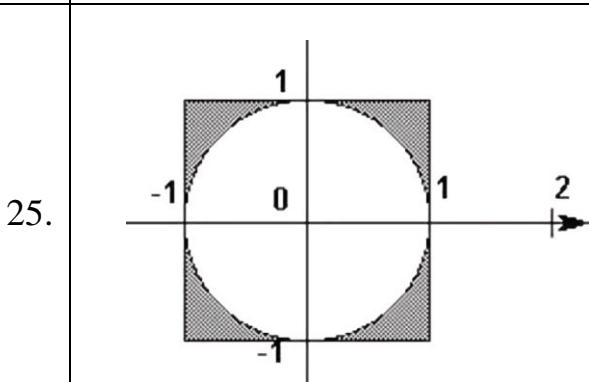
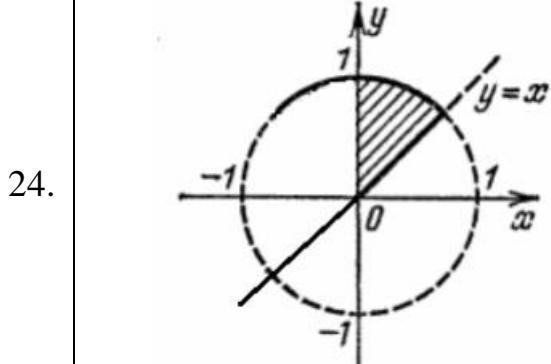
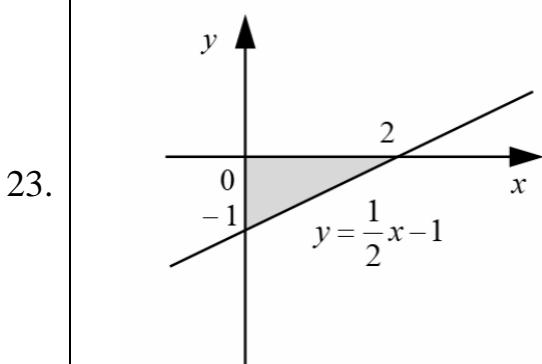
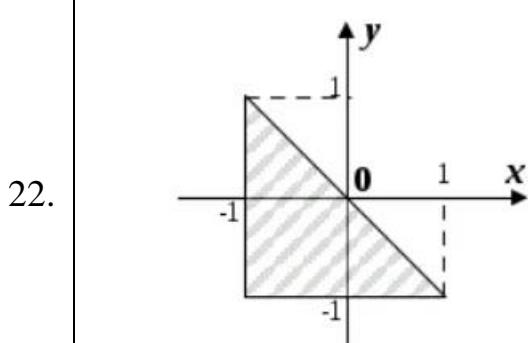
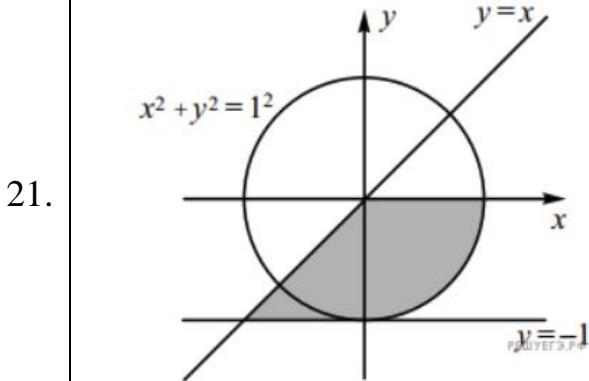
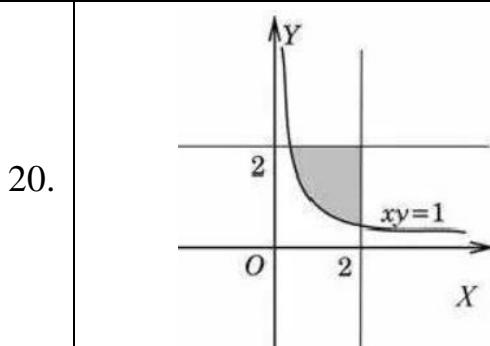
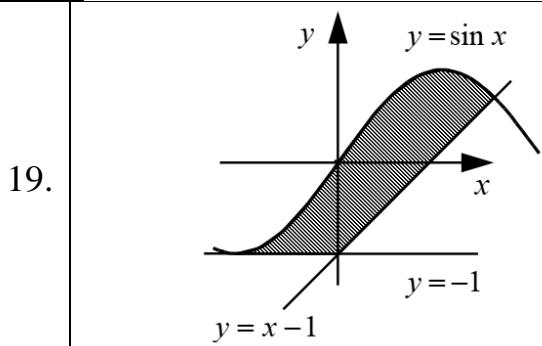
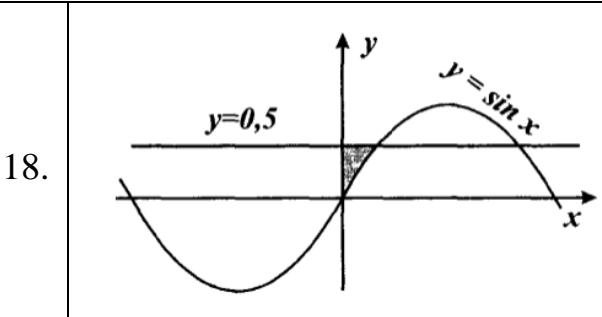
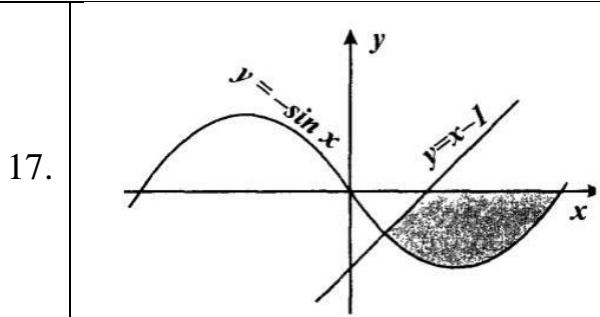
<b>№ вар.</b>	<b>Функция</b>	<b>Диапазон изменения аргумента [a,b]</b>
16.	$y = 27^{\frac{2x-1}{x}} - \sqrt{9^{2x-1}}$	[2,4]
17.	$y = \lg^2 x - \lg x^3 + 2$	[1,10]
18.	$y = \sqrt{10 - 18 \cos x} - 6 \cos x - 5$	[\pi/2, 3\pi/2]
19.	$y = 3 \sin \sqrt{x} + 0,35x - 3,8$	[2,3]
20.	$y = x + \sqrt{x} + \sqrt[3]{x} - 2,5$	[0,4;1]
21.	$y = \sin(\ln x) - \cos(\ln x) + 2 \ln x$	[1;3]
22.	$y = \cos \frac{2}{x} - 2 \sin \frac{1}{x} + \frac{1}{x}$	[1;2]
23.	$y = 3x - 4 \ln x - 5$	[2;4]
24.	$y = \sqrt{1-x} - \cos \sqrt{1-x}$	[0;1]
25.	$y = 3 \ln^2 x + 6 \ln x - 5$	[1;3]
26.	$y = 3x - 14 + e^x - e^{-x}$	[1;3]
27.	$y = \sqrt{1-x} - \operatorname{tg} x$	[0;1]
28.	$y = \operatorname{tg} x - \frac{1}{3} \operatorname{tg}^3 x + \frac{1}{5} \operatorname{tg}^5 x - \frac{1}{3}$	[0;0,8]
29.	$y = \sin x - \cos x^2 + 0,25$	[2,5;3,5]
30.	$y = e^x \cos x$	[0,5]

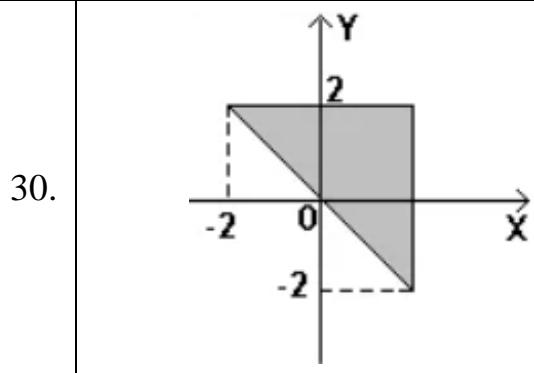
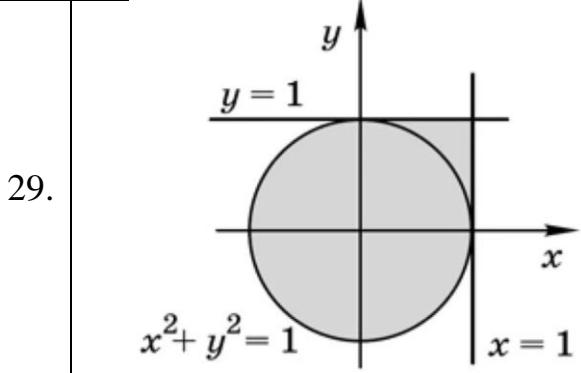
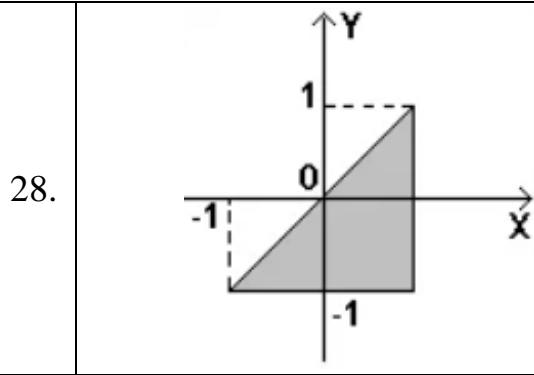
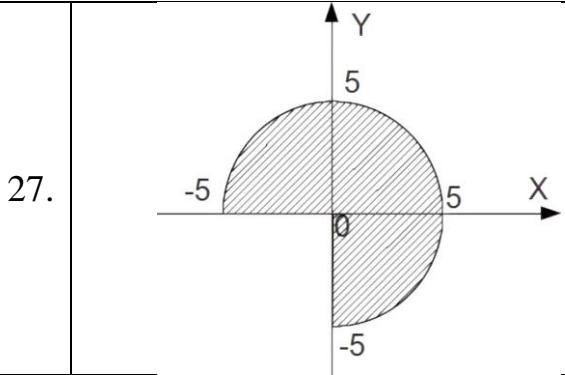
№ вар.	Функция	Диапазон изменения аргумента [a,b]
31.	$y =  x - 1  - 0.5$ $y = x^2 - 4$	-4[-5,5]

**Задание 2.** Координаты точки вводятся с клавиатуры. Определить, попадает ли точка в указанную область. Использовать оператор `if`.









**Задание 3.** Исходные данные вводятся с клавиатуры. Для решения задач использовать оператор `match-case`.

<i>№ вар.</i>	<i>Формулировка задачи</i>
1.	Вычислить площадь и периметр прямоугольника, если задана длина одной стороны ( $a$ ) и коэффициент $n$ (%), позволяющий вычислить длину второй стороны ( $b=n*a$ ).
2.	Дана длина ребра куба. Найти площадь грани, площадь полной поверхности и объем этого куба.
3.	Вычислить периметр и площадь прямоугольного треугольника по заданным длинам двух катетов $a$ и $b$ .
4.	Вычислить площади геометрических фигур: прямоугольника и треугольника по заданным сторонам.
5.	По известному радиусу вычислить объем и площадь поверхности шара.
6.	Даны два числа. Найти среднее арифметическое кубов этих чисел и среднее геометрическое модулей этих чисел.
7.	Известен объем информации в байтах. Выразить его в мегабайтах, гигабайтах и битах.

<i>№ вар.</i>	<i>Формулировка задачи</i>
8.	Длина выражена в сантиметрах. Выразить ее в метрах, миллиметрах, дюймах (1 дюйм = 2,5 см).
9.	Перевести значение веса, заданное в граммах, в килограммы, тонны, унции (1 унция = 28,3 грамма).
10.	Вычислить путь, пройденный лодкой по течению и против течения, если известна ее скорость в стоячей воде, скорость течения реки и время движения.
11.	Вычислить площади геометрических фигур: трапеции и круга.
12.	Заданы длины трех сторон треугольника. Вычислить его периметр и площадь.
13.	Дана сторона равностороннего треугольника. Найти площадь этого треугольника и его высоту.
14.	Дана сторона равностороннего треугольника. Найти радиусы вписанной и описанной окружностей.
15.	Вычислить объем и площадь полной поверхности цилиндра, если известны высота и радиус основания.
16.	Заданы стороны прямоугольника. Определить его периметр, площадь и длину диагонали.
17.	Заданы длина, ширина и высота прямоугольного параллелепипеда. Определить объем и площадь поверхности параллелепипеда.
18.	Заданы три положительных числа. Вычислить их среднее арифметическое и среднее геометрическое.
19.	Известен объем информации в килобайтах. Выразить его в мегабайтах, гигабайтах и битах.
20.	Известен объем информации в мегабайтах. Выразить его в байтах, гигабайтах и килобайтах.
21.	Задана сторона ромба и его высота. Определить его периметр и площадь.
22.	Задана длина стороны куба. Определить объем, площадь поверхности и диагональ.

<i>№ вар.</i>	<i>Формулировка задачи</i>
23.	Дано целое число $N = 1, 2, 3$ , которое есть номер функции. По значению переменной $x$ вычислить значение соответствующей функции: 1) $-2x^2 - 4$ ; 2) $5x+2$ ; 3) $15-3x$ .
24.	Задана сторона ромба и один из его углов. Определить периметр и площадь ромба.
25.	Дано целое число $N= 1, 2, 3$ , которое есть номер функции. По значению переменной $x$ вычислить значение соответствующей функции: 1) $-2x + 4$ ; 2) $5x+2$ ; 3) $15-3x^2$ .
26.	Вводится число от 1 до 4, определяющее пору года. Дать название этой поры года (1 – зима, 2 – весна, 3 – лето, 4 – осень).
27.	Вводится число от 1 до 6, определяющее месяц года. Дать название этого месяца года (1 – январь, 2 – февраль, ..., 6 – июнь).
28.	Вводится число от 7 до 12, определяющее месяц года. Дать название этого месяца года (7 – июль, 8 – август, ..., 12 – декабрь).
29.	Вводится число от 1 до 7, определяющее день недели. Дать название этого дня (1 – понедельник, 2 – вторник, ..., 7 – воскресенье).
30.	Вводится число от 1 до 5. Дать название этого числа (1 – один , 2 – два , ..., 5 – пять).
31.	В спортивных соревнованиях Шарик, кот Матроскин, дядя Фёдор и почтальон Печкин заняли соответственно 1, 2, 3 и 4 места. Составить программу, которая по номеру места выдаёт имя участника соревнований.

## 6.5. Разветвленные конструкции. Задания второго уровня сложности

**Задание 4.** Исходные данные вводятся с клавиатуры. Вычислить значения функций. При выполнении можно использовать операторы `if` или `match-case` (1 – первая функция, 2 – вторая).

<i>№ вар.</i>	<i>Расчетные формулы</i>	<i>Исходные данные</i>
1.	$a = \frac{2 \cos(x - \pi / 6)}{1 / 2 + \sin^2 y};$ $b = 1 + \frac{z^2}{3 + z^2 / 5}.$	$x = 1,426$ $y = -1,22$ $z = 3,5$

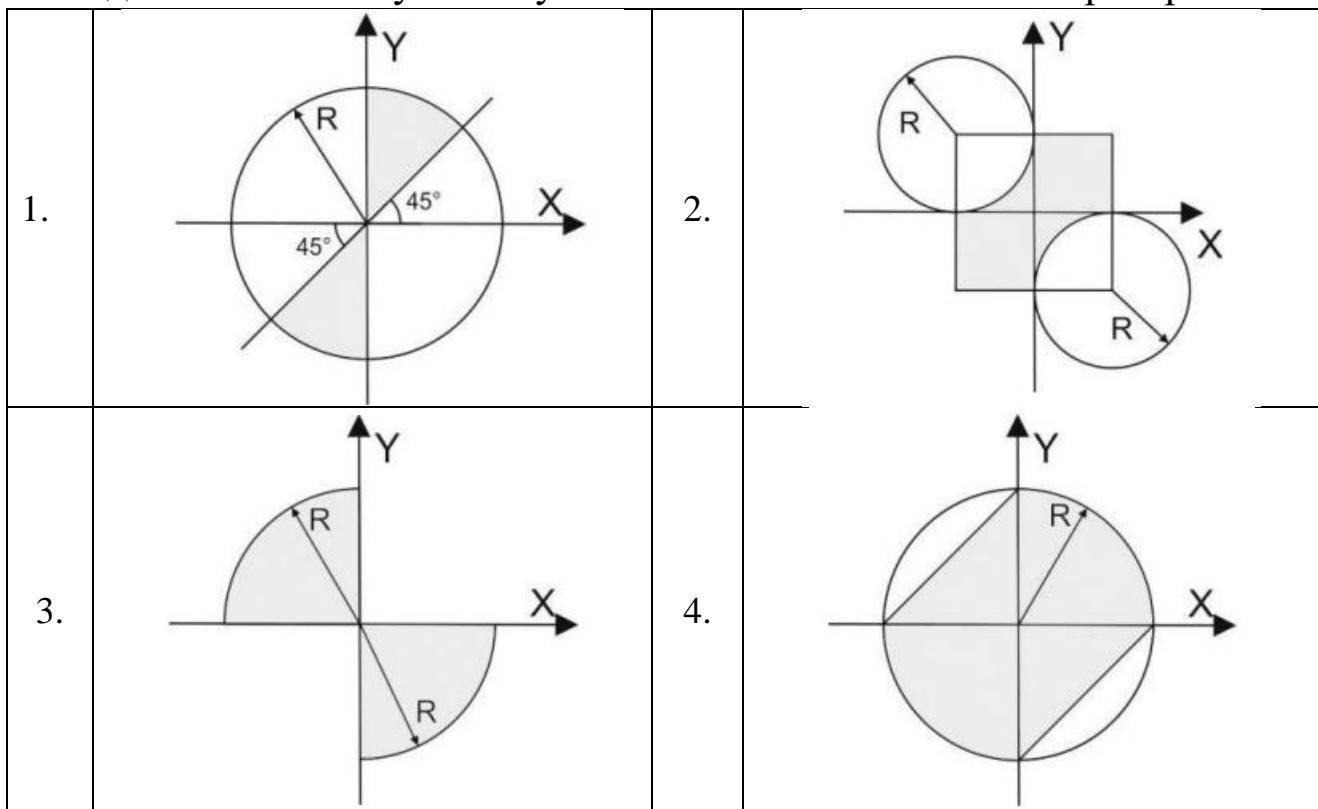
<i>№ вар.</i>	<i>Расчетные формулы</i>	<i>Исходные данные</i>
2.	$\gamma = \left  x^{y/x} - \sqrt[3]{y/x} \right ;$ $\varphi = (y-x) \frac{y-z/(y-x)}{1+(y-x)^2}.$	$x=1,825$ $y=18,225$ $z=-3,298$
3.	$s = \frac{e^x + \cos^2 e^y}{(x+y)^2};$ $z = \sqrt{(x+y)}(\sin x^3 + \cos^2 y).$	$x=0,335$ $y=0,025$
4.	$y = e^{-bt} \sin(at+b) - \sqrt{ bt+a };$ $s = b \sin(at^2 \cos 2t) - 1.$	$a=-0,5$ $b=1,7$ $t=0,44$
5.	$\omega = \sqrt{x^2 + b} - b^2 \sin^3(x+a)/x;$ $y = \cos^2 x^3 - x/\sqrt{a^2 + b^2}.$	$a=1,5$ $b=15,5$ $x=-2,9$
6.	$s = x^3 \operatorname{tg}^2(x+b)^2 + a/\sqrt{x+b};$ $q = \frac{bx^2 - a}{e^{ax} - 1}.$	$a=16,5$ $b=3,4$ $x=0,61$
7.	$R = x^2(x+1)/b - \sin^2(x+a);$ $s = \sqrt{xb/a} + \cos^2(x+b)^2.$	$a=0,7$ $b=0,05$ $x=0,5$
8.	$y = \sin^3(x^2 + a)^2 - \sqrt{x/b};$ $z = \frac{x^2}{a} + \cos(x+b)^3.$	$a=1,1$ $b=0,004$ $x=0,2$
9.	$f = \sqrt[3]{mctb +  c \sin t };$ $z = m \cos(bt \sin t) + c.$	$m=2$ $c=1$ $t=1,2$ $b=0,7$
10.	$y = abx^2 - \frac{a}{\sin^2(x/a)};$ $d = ae^{-\sqrt{a}} \cos(bx/a).$	$a=3,2$ $b=17,5$ $x=-4,8$

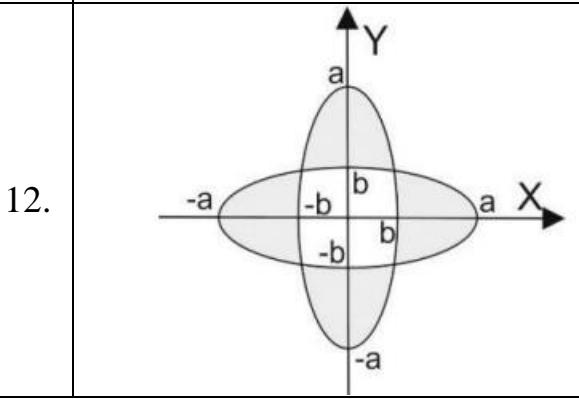
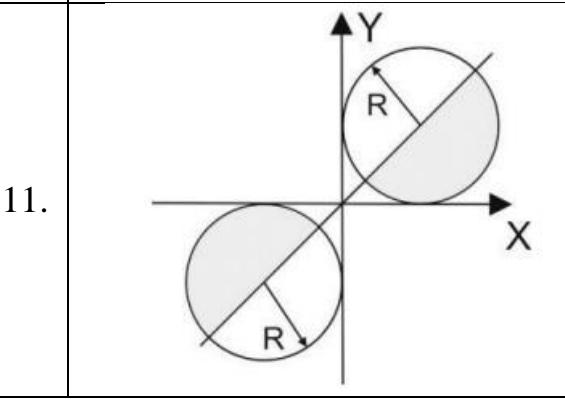
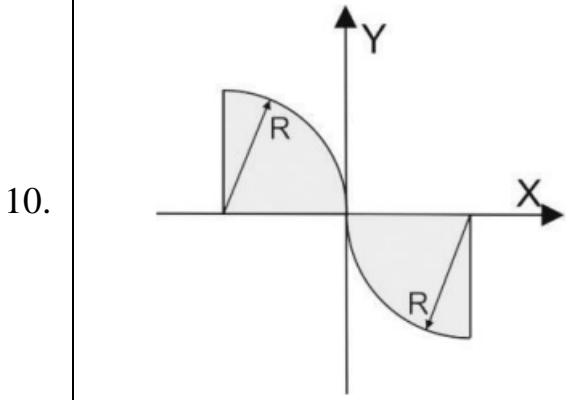
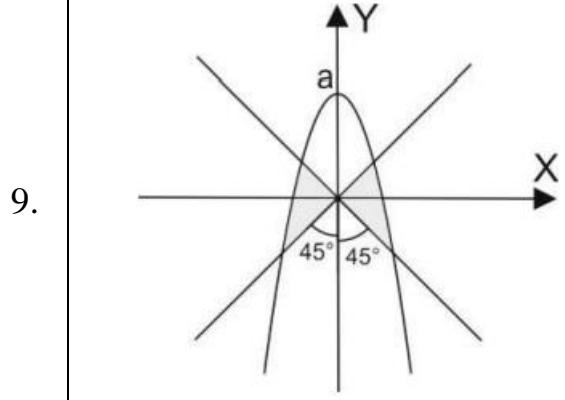
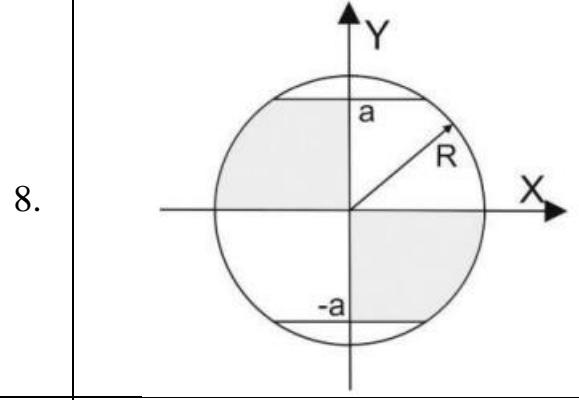
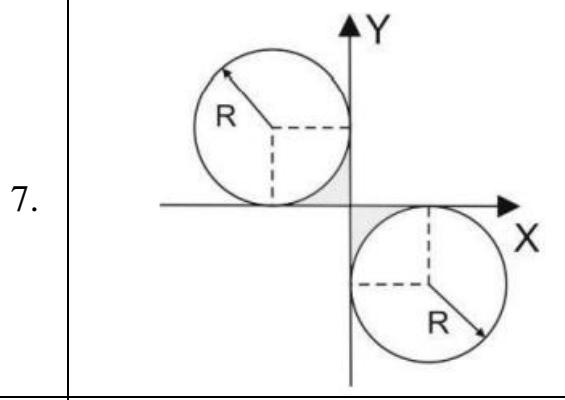
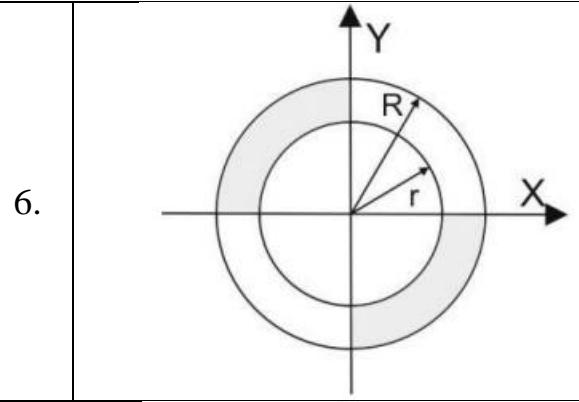
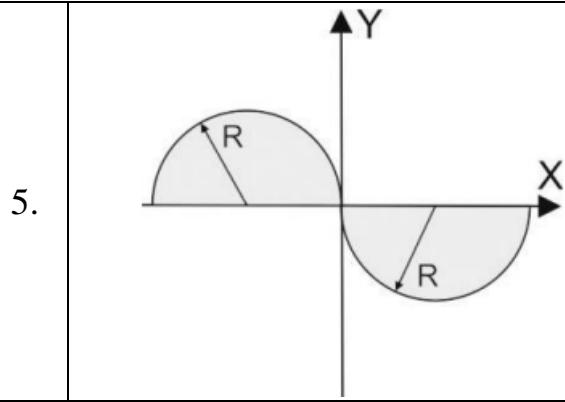
<i>№ вар.</i>	<i>Расчетные формулы</i>	<i>Исходные данные</i>
11.	$f = \ln(a + x^2) + \sin^2(x/b);$ $z = e^{-cx} \frac{x + \sqrt{x+a}}{x - \sqrt{ x-b }}.$	$a = 10,2$ $b = 9,2$ $c = 0,5$ $x = 2,2$
12.	$y = \frac{a^{2x} + b^{-x} \cos(a+b)x}{x+1};$ $R = \sqrt{x^2 + b} - b^2 \sin^3(x+a)/x.$	$a = 0,3$ $b = 0,9$ $x = 0,61$
13.	$z = \sqrt{ax \sin 2x + e^{-2x}(x+b)};$ $\omega = \cos^2 x^3 - x/\sqrt{a^2 + b^2}.$	$a = 0,5$ $b = 3,1$ $x = 1,4$
14.	$U = \frac{a^2 x + e^{-x} \cos bx}{bx - e^{-x} \sin bx + 1};$ $f = e^{2x} \ln(a+x) - b^{3x} \ln(b-x).$	$a = 0,5$ $b = 2,9$ $x = 0,3$
15.	$z = \frac{\sin x}{\sqrt{m^2 + \sin^2 x}} - cm \ln mx;$ $s = e^{-ax} \sqrt{x+1} + e^{-bx} \sqrt{x+1.5}.$	$m = 0,5$ $c = 1,5$ $x = 2$ $a = 1,5$ $b = 1,3$
16.	$a = \frac{2 \cos(x - \pi/6)}{1/2 + \sin^2 y};$ $b = 1 + \frac{z^2}{3 + z^2/5}.$	$x = 1,426$ $y = -1,22$ $z = 3,5$
17.	$\gamma = \left  x^{y/x} - \sqrt[3]{y/x} \right ;$ $\varphi = (y-x) \frac{y-z/(y-x)}{1+(y-x)^2}.$	$x = 1,825$ $y = 18,225$ $z = -3,298$
18.	$s = \frac{e^x + \cos^2 e^y}{(x+y)^2};$ $z = \sqrt{(x+y)}(\sin x^3 + \cos^2 y).$	$x = 0,335$ $y = 0,025$

<i>№ вар.</i>	<i>Расчетные формулы</i>	<i>Исходные данные</i>
19.	$y = e^{-bt} \sin(at + b) - \sqrt{ bt + a };$ $s = b \sin(at^2 \cos 2t) - 1.$	$a = -0,5$ $b = 1,7$ $t = 0,44$
20.	$\omega = \sqrt{x^2 + b} - b^2 \sin^3(x + a)/x;$ $y = \cos^2 x^3 - x/\sqrt{a^2 + b^2}.$	$a = 1,5$ $b = 15,5$ $x = -2,9$
21.	$s = x^3 \operatorname{tg}^2(x + b)^2 + a/\sqrt{x + b};$ $q = \frac{bx^2 - a}{e^{ax} - 1}.$	$a = 16,5$ $b = 3,4$ $x = 0,61$
22.	$R = x^2(x + 1)/b - \sin^2(x + a);$ $s = \sqrt{xb/a} + \cos^2(x + b)^2.$	$a = 0,7$ $b = 0,05$ $x = 0,5$
23.	$y = \sin^3(x^2 + a)^2 - \sqrt{x/b};$ $z = \frac{x^2}{a} + \cos(x + b)^3.$	$a = 1,1$ $b = 0,004$ $x = 0,2$
24.	$f = \sqrt[3]{mctb +  c \sin t };$ $z = m \cos(bt \sin t) + c.$	$m = 2$ $c = 1$ $t = 1,2$ $b = 0,7$
25.	$y = abx^2 - \frac{a}{\sin^2(x/a)};$ $d = ae^{-\sqrt{a}} \cos(bx/a).$	$a = 3,2$ $b = 17,5$ $x = -4,8$
26.	$f = \ln(a + x^2) + \sin^2(x/b);$ $z = e^{-cx} \frac{x + \sqrt{x+a}}{x - \sqrt{ x-b }}.$	$a = 10,2$ $b = 9,2$ $c = 0,5$ $x = 2,2$
27.	$y = \frac{a^{2x} + b^{-x} \cos(a+b)x}{x+1};$ $R = \sqrt{x^2 + b} - b^2 \sin^3(x + a)/x.$	$a = 0,3$ $b = 0,9$ $x = 0,61$

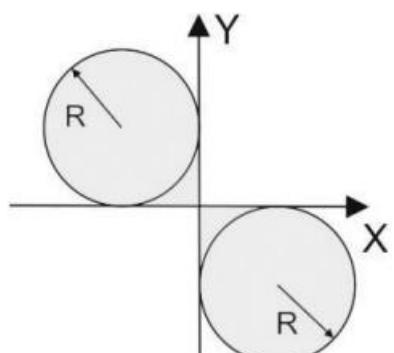
<i>№ вар.</i>	<i>Расчетные формулы</i>	<i>Исходные данные</i>
28.	$z = \sqrt{ax \sin 2x + e^{-2x}(x+b)}$ ; $\omega = \cos^2 x^3 - x / \sqrt{a^2 + b^2}$ .	$a = 0,5$ $b = 3,1$ $x = 1,4$
29.	$U = \frac{a^2 x + e^{-x} \cos bx}{bx - e^{-x} \sin bx + 1}$ ; $f = e^{2x} \ln(a+x) - b^{3x} \ln(b-x)$ .	$a = 0,5$ $b = 2,9$ $x = 0,3$
30.	$z = \frac{\sin x}{\sqrt{m^2 + \sin^2 x}} - cm \ln mx$ ; $s = e^{-ax} \sqrt{x+1} + e^{-bx} \sqrt{x+1.5}$ .	$m = 0,5$ $c = 1,5$ $x = 2$ $a = 1,5$ $b = 1,3$
31.	$U = \frac{a^2 x + e^{-x} \cos bx}{bx - e^{-x} \sin bx + 1}$ ; $f = e^{2x} \ln(a+x) - b^{3x} \ln(b-x)$ .	$a = 0,5$ $b = 2,9$ $x = 0,3$

**Задание 5.** Координаты точки вводятся с клавиатуры. Определить, попадает ли точка в указанную область. Использовать оператор `if`.

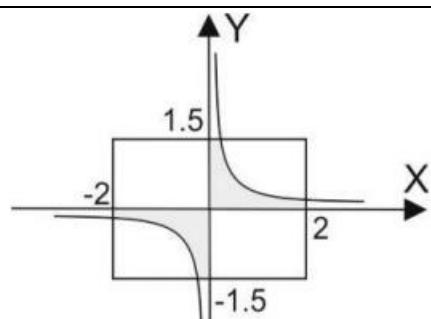




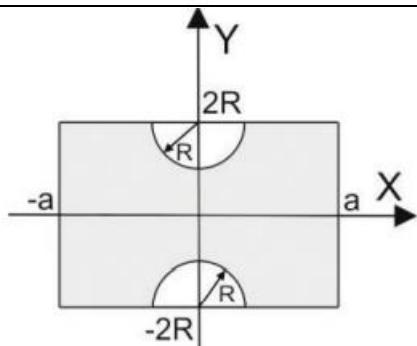
13.



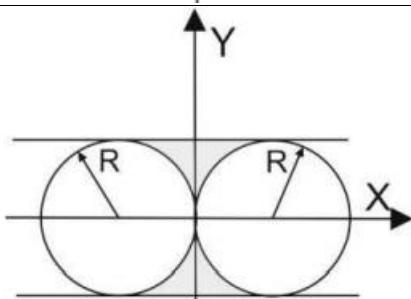
14.



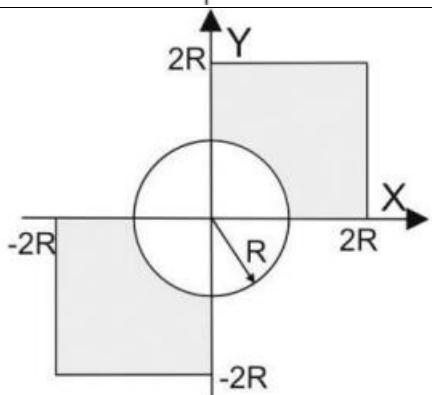
15.



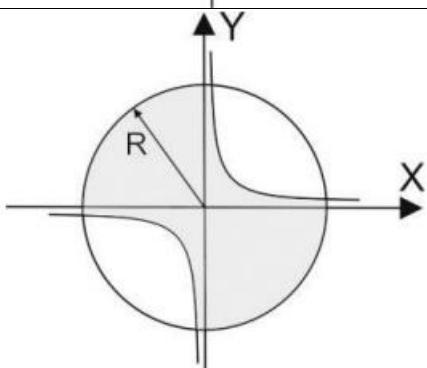
16.



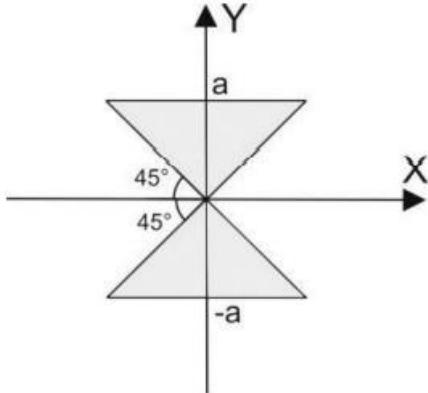
17.



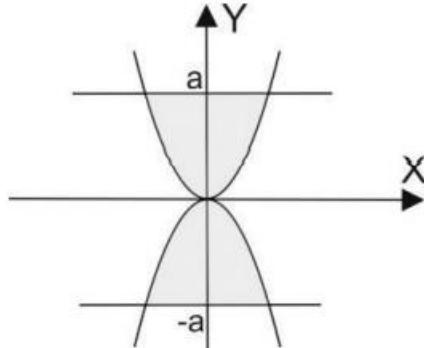
18.

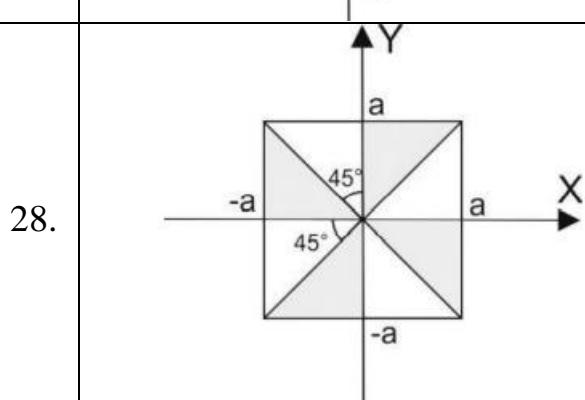
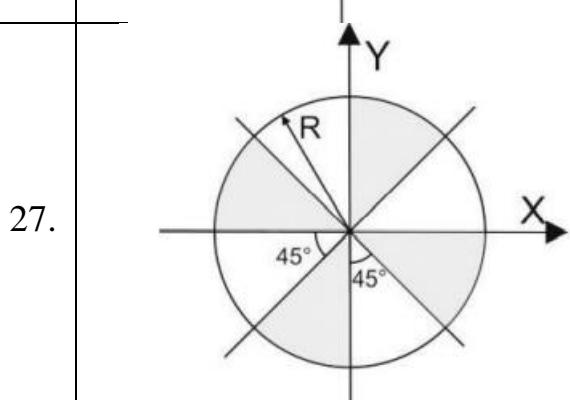
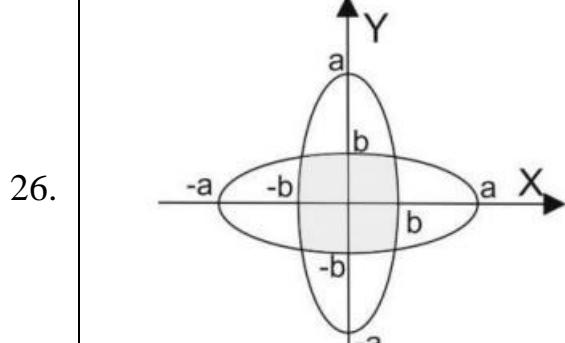
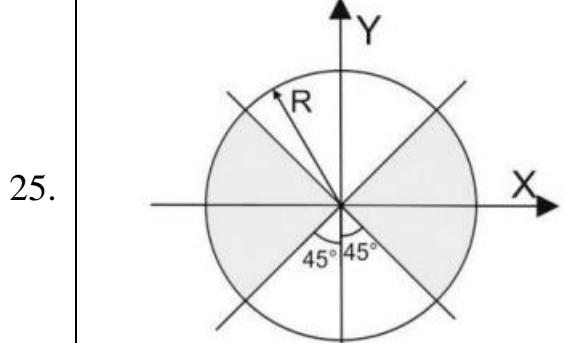
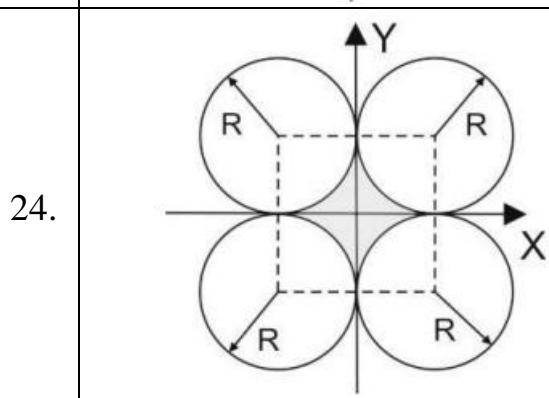
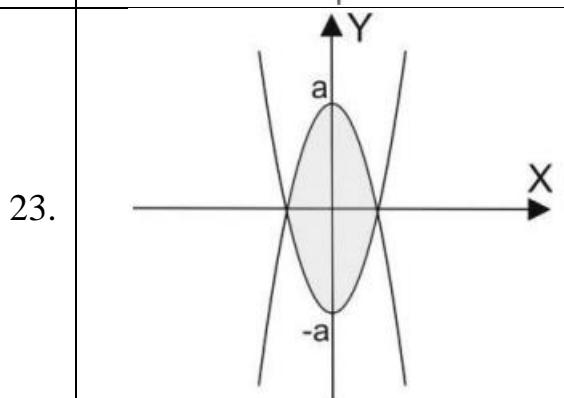
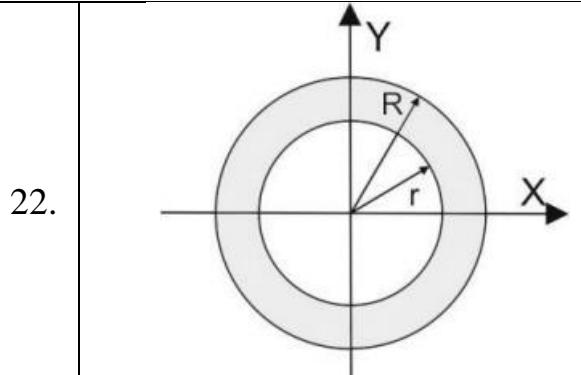
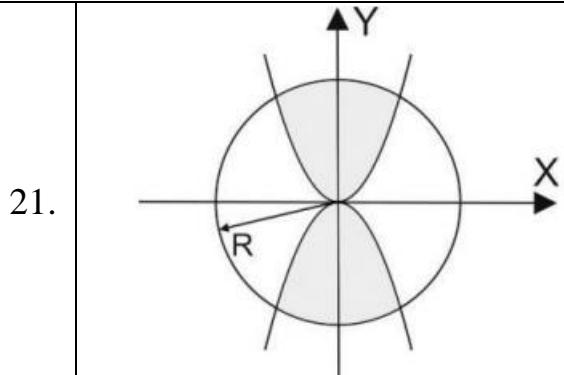


19.

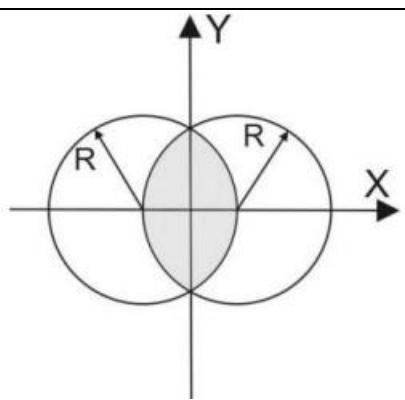


20.

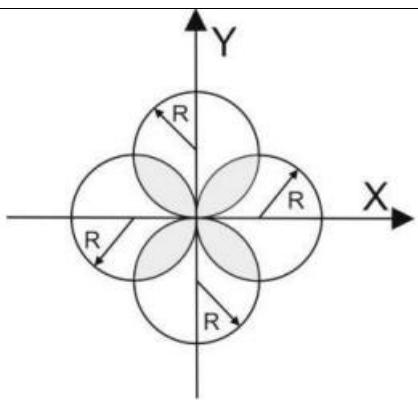




29.



30.



## 7. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какие символы используются для обозначения комментариев в программном коде?
2. Перечислите типы данных, используемые в Python.
3. Опишите формат объявления констант и переменных.
4. Каким образом осуществляется *преобразования типов* данных в Python?
5. Какие методы используются для вывода по формату? Приведите примеры их использования.
6. Какие операции используются в арифметических выражениях?
7. Назовите основные логические операции, используемые для написания логических выражений.
8. Перечислите знаки отношений.
9. Назовите основные логические операции, используемые для написания логических выражений.
10. С помощью какого оператора реализуется алгоритмическая структура «Развилка»? Нарисуйте ее блок-схему.
11. Когда применяется условный оператор?
12. Приведите пример на использование условного оператора.
13. С помощью какого оператора реализуется алгоритмическая структура «Выбор»? Нарисуйте ее блок-схему.
14. Что позволяет делать оператор выбора?
15. Что такое селектор в операторе выбора?
16. Приведите пример использования оператора выбора.

## **СПИСОК ЛИТЕРАТУРЫ**

1. Карчевская, М. П. Основы алгоритмизации инженерных задач [Электронный ресурс]: [учебное пособие для студентов специальностей 27.05.01 "Специальные организационно-технические системы"; 10.05.05 "Безопасность информационных технологий в правоохранительной сфере"; 09.05.01 "Применение и эксплуатация автоматизированных систем специального назначения"] / М. П. Карчевская, О. Л. Рамбургер; Уфимский государственный авиационный технический университет (УГАТУ). — Электронные текстовые данные (1 файл: 6,06 МБ). — Уфа: УГАТУ, 2019.
2. Воробьев, А. В. Методы языка Python для представления знаний в информационных системах : лабораторный практикум по дисциплине "Представление знаний в информационных системах" / А. В, Воробьев, Н. А. Мириянов ; Уфимский государственный авиационный технический университет (УГАТУ) .— Учебное электронное издание .— Уфа : УГАТУ, 2020 .— 1 электрон. опт. диск (CD-ROM) ; 12 см .— Текст: электронный. — Систем. требования: Pentium 300 МГц, Windows 98, MS Internet Explorer 6.0, CD-ROM 12x и выше, 32 Mb RAM, видеокарта и монитор, поддерживающий режим 800x600 16 бит, мышь, звуковая карта (дата обращения: 26.01.2024).
3. Основы разработки программного обеспечения на языке программирования высокого уровня Python: учебное пособие для студентов очной и заочной форм обучения, обучающихся по направлению подготовки бакалавров 09.03.03 Прикладная информатика / Л. А. Кромина, Л. Е. Родионова, А. Р. Фахруллина, Р. А. Ярцева; Уфимский государственный авиационный технический университет (УГАТУ). — Электронные текстовые данные (1 файл: 1,04 МБ). — Уфа: УГАТУ, 2020. — Электронная версия печатной публикации. — Заглавие с титул. экрана. — Доступ из сети Интернет по логину и паролю. Анонимный доступ из корпоративной сети УГАТУ. — Систем. требования: Adobe Reader. — <URL:[http://e-library.ufa-rb.ru/dl/lib\\_net\\_r/Kromina\\_L\\_A\\_Osn\\_razr\\_progr\\_obesp\\_na\\_yaz\\_2020.pdf](http://e-library.ufa-rb.ru/dl/lib_net_r/Kromina_L_A_Osn_razr_progr_obesp_na_yaz_2020.pdf)>. — Текст: электронный.