

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития  
Кафедра инфокоммуникаций

**ОТЧЕТ**  
**ПО ЛАБОРАТОРНОЙ РАБОТЕ №4**  
**дисциплины «Программирование на Python»**

Выполнил:  
Говоров Егор Юрьевич  
2 курс, группа ИВТ-б-о-22-1,  
09.03.01 «Информатика и  
вычислительная техника»,  
направленность (профиль)  
«Программное обеспечение средств  
вычислительной техники и  
автоматизированных систем», очная  
форма обучения

---

(подпись)

Руководитель практики:  
Воронкин Р. А., канд. технических  
наук, доцент кафедры  
инфокоммуникаций

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

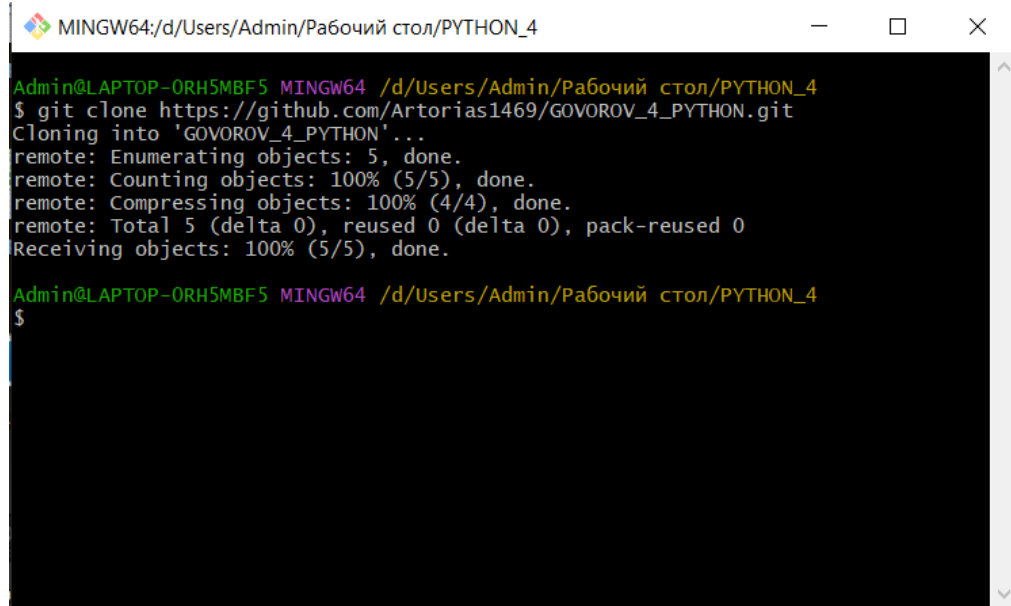
Ставрополь, 2023 г.

Тема: Основы языка Python

Цель: исследование процесса установки и базовых возможностей языка Python версии 3.x.

### Ход работы

1. Создал общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python. Выполнил клонирование созданного репозитория.

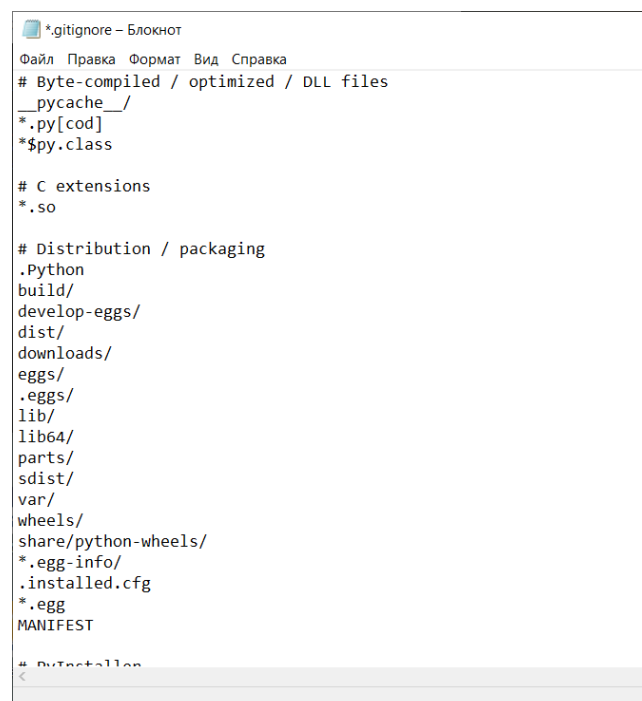


```
Admin@LAPTOP-0RH5MBF5 MINGW64 /d/Users/Admin/Рабочий стол/PYTHON_4
$ git clone https://github.com/Artorias1469/GOVOROV_4_PYTHON.git
Cloning into 'GOVOROV_4_PYTHON'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.

Admin@LAPTOP-0RH5MBF5 MINGW64 /d/Users/Admin/Рабочий стол/PYTHON_4
$
```

Рисунок 1. Клонирование репозитория

2. Дополнил файл .gitignore необходимыми правилами.



```
*.gitignore - Блокнот
Файл  Правка  Формат  Вид  Справка
# Byte-compiled / optimized / DLL files
__pycache__/
*.py[co]
*$py.class

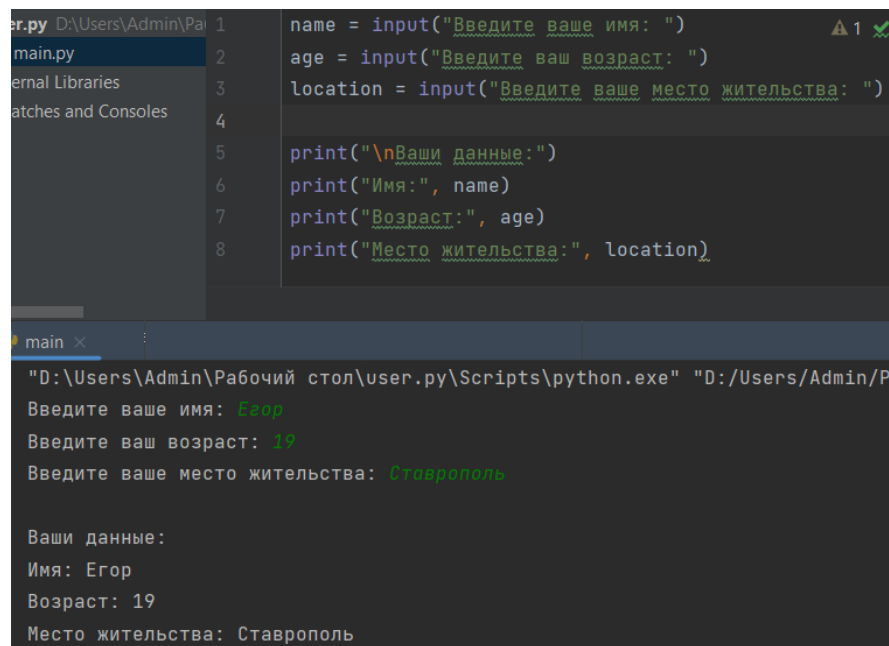
# C extensions
*.so

# Distribution / packaging
.Python
build/
develop-eggs/
dist/
downloads/
eggs/
.eggs/
lib/
lib64/
parts/
sdist/
var/
wheels/
share/python-wheels/
*.egg-info/
.installed.cfg
*.egg
MANIFEST

# PyInstaller
<
```

Рисунок 2. Дополнение .gitignore

3. Написал программу (файл user.py), которая запрашивает у пользователя его имя, возраст и место жительства, а после этого выводит данные, введенные пользователем.



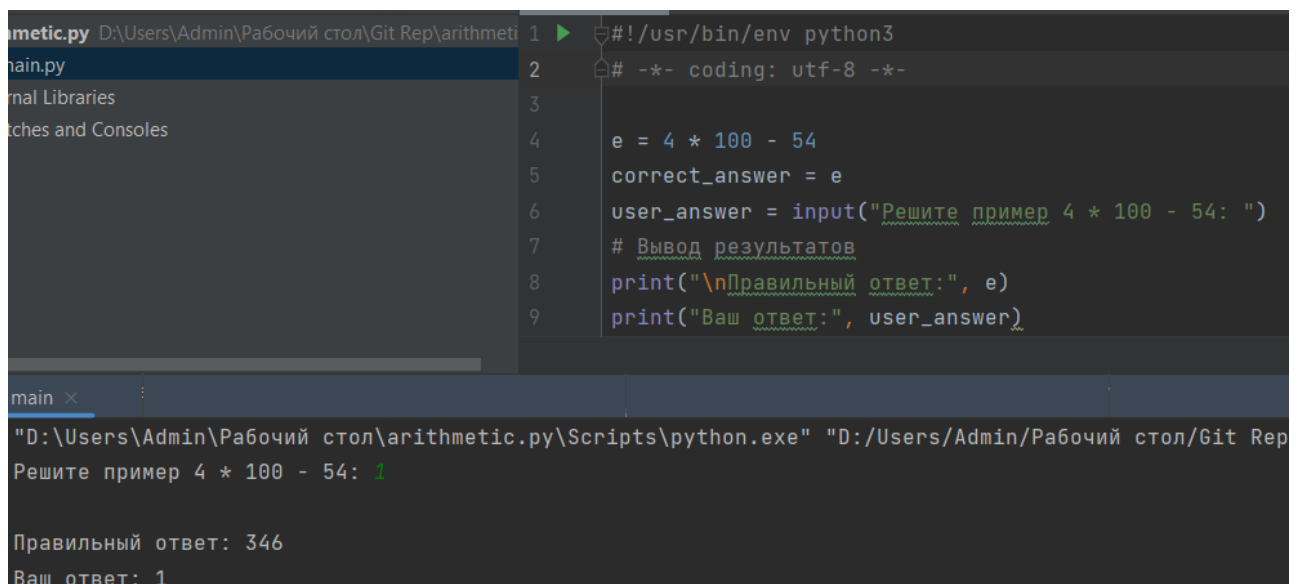
```
er.py D:\Users\Admin\Рабочий стол\user.py 1 name = input("Введите ваше имя: ")
main.py 2 age = input("Введите ваш возраст: ")
ernal Libraries 3 location = input("Введите ваше место жительства: ")
atches and Consoles 4
5 print("\nВаши данные:")
6 print("Имя:", name)
7 print("Возраст:", age)
8 print("Место жительства:", location)

main x
"D:\Users\Admin\Рабочий стол\user.py\Scripts\python.exe" "D:/Users/Admin/Рабочий стол/user.py"
Введите ваше имя: Егор
Введите ваш возраст: 19
Введите ваше место жительства: Ставрополь

Ваши данные:
Имя: Егор
Возраст: 19
Место жительства: Ставрополь
```

Рисунок 3. Результат работы программы user.py

4. Написал программу (файл arithmetic.py), которая предлагает пользователю решить пример  $4 * 100 - 54$ . Потом выводит на экран правильный ответ и ответ пользователя.



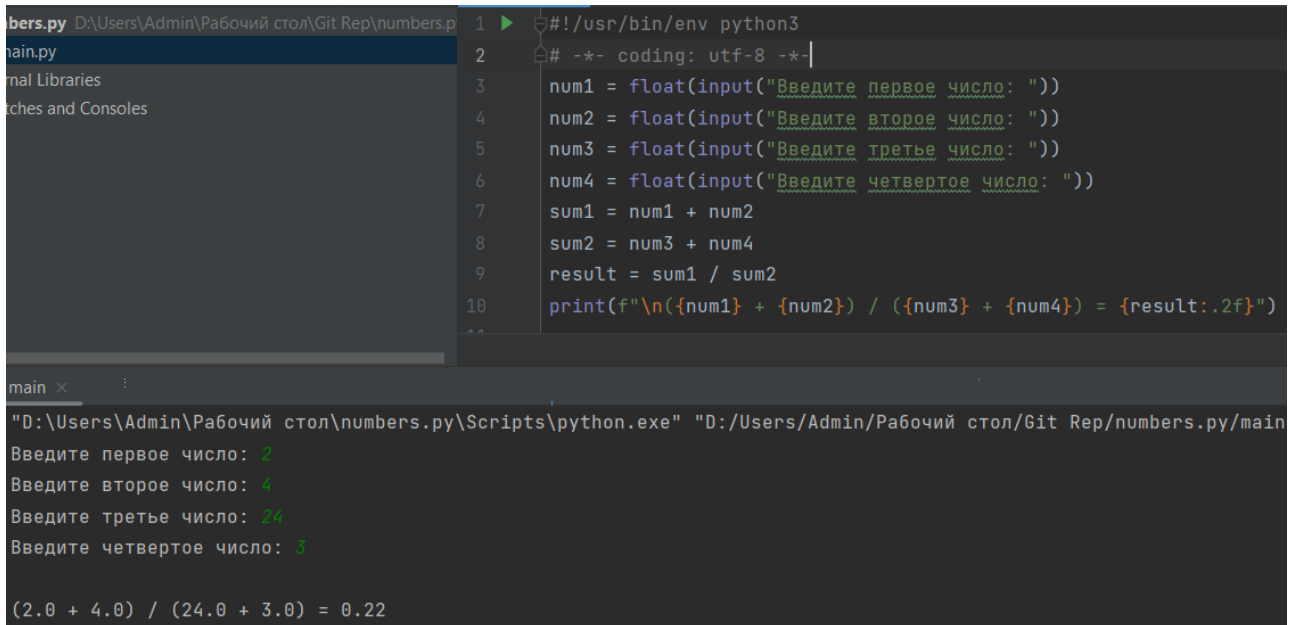
```
arithmetic.py D:\Users\Admin\Рабочий стол\Git Rep\arithmetic.py 1 #!/usr/bin/env python3
main.py 2 # -*- coding: utf-8 -*-
ernal Libraries 3
atches and Consoles 4
5 e = 4 * 100 - 54
6 correct_answer = e
7 user_answer = input("Решите пример 4 * 100 - 54: ")
8 # Вывод результатов
9 print("\nПравильный ответ:", e)
10 print("Ваш ответ:", user_answer)

main x
"D:\Users\Admin\Рабочий стол\arithmetic.py\Scripts\python.exe" "D:/Users/Admin/Рабочий стол/Git Rep/arithmetic.py"
Решите пример 4 * 100 - 54: 1

Правильный ответ: 346
Ваш ответ: 1
```

Рисунок 4. Результат работы программы arithmetic.py

5. Написал программу, которая запрашивает у пользователя четыре числа (файл numbers.py), отдельно складывает первые два и отдельно вторые два, далее делит первую сумму на вторую и выводит результат на экран так, чтобы ответ содержал две цифры после запятой.



```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 num1 = float(input("Введите первое число: "))
4 num2 = float(input("Введите второе число: "))
5 num3 = float(input("Введите третье число: "))
6 num4 = float(input("Введите четвертое число: "))
7 sum1 = num1 + num2
8 sum2 = num3 + num4
9 result = sum1 / sum2
10 print(f"\n({num1} + {num2}) / ({num3} + {num4}) = {result:.2f}")
```

main x

"D:\Users\Admin\Рабочий стол\numbers.py\Scripts\python.exe" "D:/Users/Admin/Рабочий стол/Git Rep/numbers.py/main

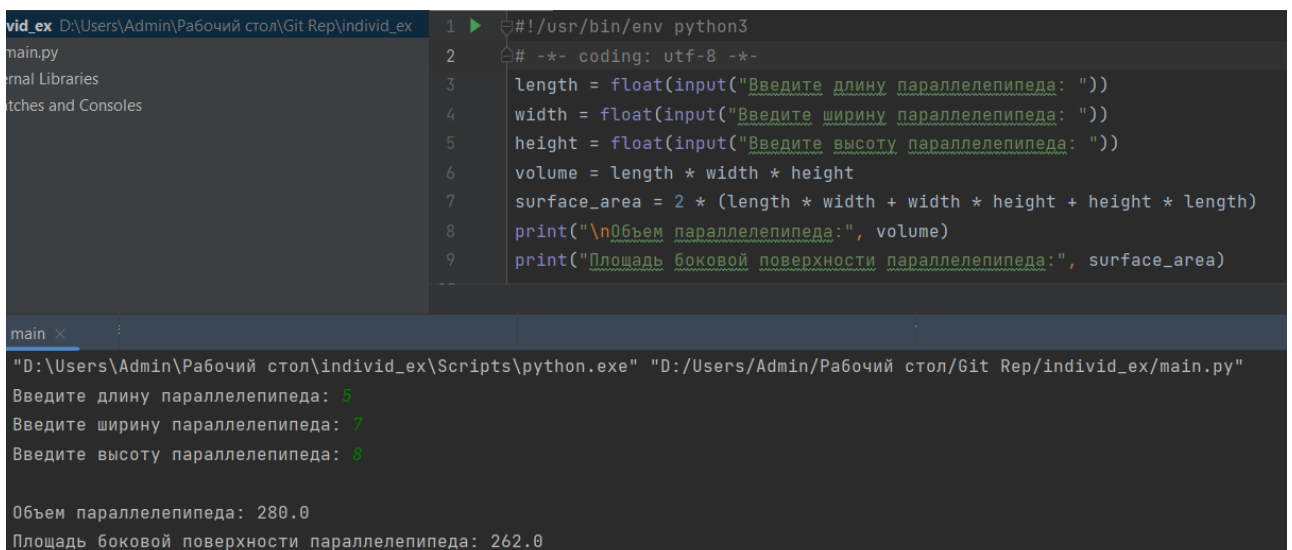
Введите первое число: 2  
Введите второе число: 4  
Введите третье число: 24  
Введите четвертое число: 3

(2.0 + 4.0) / (24.0 + 3.0) = 0.22

Рисунок 5. Результат работы программы numbers.py

6. Написал программу (файл individ\_ex.py) для решения индивидуального задания.

Вариант 5: Даны длины прямоугольного параллелепипеда. Найти объем и площадь боковой поверхности



```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 length = float(input("Введите длину параллелепипеда: "))
4 width = float(input("Введите ширину параллелепипеда: "))
5 height = float(input("Введите высоту параллелепипеда: "))
6 volume = length * width * height
7 surface_area = 2 * (length * width + width * height + height * length)
8 print(f"\nОбъем параллелепипеда: ", volume)
9 print(f"Площадь боковой поверхности параллелепипеда: ", surface_area)
```

main x

"D:\Users\Admin\Рабочий стол\individ\_ex\Scripts\python.exe" "D:/Users/Admin/Рабочий стол/Git Rep/individ\_ex/main.py"

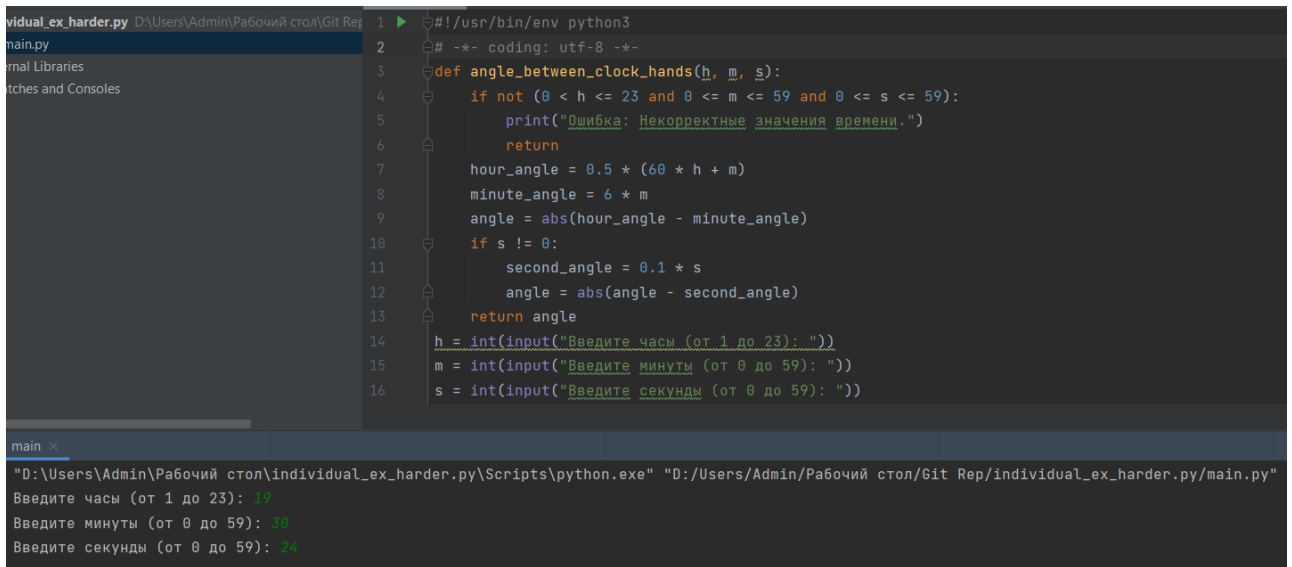
Введите длину параллелепипеда: 5  
Введите ширину параллелепипеда: 7  
Введите высоту параллелепипеда: 8

Объем параллелепипеда: 280.0  
Площадь боковой поверхности параллелепипеда: 262.0

Рисунок 6. Результат работы программы individ\_ex.py

7. Написал программу (файл individual\_ex\_harder.py) для решения индивидуального задания повышенной сложности.

Вариант 5: Даны целые числа ( $0 < h \leq 23$ ,  $0 \leq m \leq 59$ ,  $0 \leq s \leq 59$ ), указывающие момент времени: "h часов, m минут, s секунд". Определить угол (в градусах) между положением часовой стрелки в начале суток в указанный момент времени



```
individual_ex_harder.py D:\Users\Admin\Рабочий стол\Git Rep
main.py
ernal Libraries
atches and Consoles

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  def angle_between_clock_hands(h, m, s):
4      if not (0 < h <= 23 and 0 <= m <= 59 and 0 <= s <= 59):
5          print("Ошибка: Некорректные значения времени.")
6          return
7      hour_angle = 0.5 * (60 * h + m)
8      minute_angle = 6 * m
9      angle = abs(hour_angle - minute_angle)
10     if s != 0:
11         second_angle = 0.1 * s
12         angle = abs(angle - second_angle)
13     return angle
14     h = int(input("Введите часы (от 1 до 23): "))
15     m = int(input("Введите минуты (от 0 до 59): "))
16     s = int(input("Введите секунды (от 0 до 59): "))

main x
"D:\Users\Admin\Рабочий стол\individual_ex_harder.py\Scripts\python.exe" "D:/Users/Admin/Рабочий стол/Git Rep/individual_ex_harder.py/main.py"
Введите часы (от 1 до 23): 19
Введите минуты (от 0 до 59): 30
Введите секунды (от 0 до 59): 24
```

Рисунок 7. Результат работы программы individual\_ex\_harder.py

## Контрольные вопросы

### 1. Опишите основные этапы установки Python в Windows и Linux.

В Windows: Скачать установочный файл Python с официального сайта (python.org). Запустить установщик и следовать инструкциям. Выбрать опцию "Add Python to PATH" (добавить Python в переменную среды PATH) для удобства использования Python из командной строки. Завершить процесс установки.

В Linux: Многие дистрибутивы Linux уже имеют Python предустановленным. В противном случае, можно установить его с помощью пакетного менеджера вашего дистрибутива, например, в Ubuntu: `sudo apt-get install python3`. После установки можно проверить версию с помощью команды `python3 --version`.

### 2. В чем отличие пакета Anaconda от пакета Python, скачиваемого с официального сайта?

Anaconda - это дистрибуция Python, предназначенная для научных вычислений и анализа данных. Основное отличие заключается в том, что Anaconda включает в себя множество предустановленных библиотек и инструментов, таких как NumPy, Pandas, Matplotlib, Jupyter и многие другие, что делает ее идеальным выбором для работы в области анализа данных и машинного обучения. Стандартный пакет Python с официального сайта включает только базовые библиотеки.

### 3. Как осуществить проверку работоспособности пакета Anaconda?

Для проверки работоспособности Anaconda можно запустить интерактивную оболочку IPython или Jupyter Notebook. Также можно создать новое окружение и установить несколько библиотек для проверки.

### 4. Как задать используемый интерпретатор языка Python в IDE PyCharm?

В PyCharm можно задать интерпретатор Python в настройках проекта или в глобальных настройках IDE. Для этого перейдите в "File" -> "Settings"

(или "Preferences" на macOS) -> "Project: [имя проекта]" -> "Python Interpreter" и выберите нужный интерпретатор Python.

#### 5. Как осуществить запуск программы с помощью IDE PyCharm?

Для запуска программы в PyCharm можно нажать кнопку "Run" (Запустить) или "Debug" (Отладка) в верхней панели. Также можно использовать комбинации клавиш, например, Shift + F10 для запуска.

#### 6. В чем суть интерактивного и пакетного режимов работы Python?

Интерактивный режим позволяет вводить команды Python построчно и немедленно видеть результат. Пакетный режим используется для выполнения скриптов и программ, которые выполняются целиком.

#### 7. Почему язык программирования Python называется языком динамической типизации?

Язык Python называется динамическим из-за того, что типы данных переменных определяются автоматически во время выполнения программы, а не во время компиляции.

#### 8. Какие существуют основные типы в языке программирования Python?

Основные типы данных в Python включают числа (int, float), строки (str), списки (list), кортежи (tuple), множества (set), словари (dict), булевы значения (bool), и многое другое.

#### 9. Как создаются объекты в памяти? Каково их устройство? В чем заключается процесс объявления новых переменных и работа операции присваивания?

Переменные - это ссылки на объекты. Процесс объявления переменных заключается в присвоении им значений, и Python автоматически выделяет память для хранения объектов.

#### 10. Как получить список ключевых слов в Python?

Получение списка ключевых слов в Python можно сделать с помощью модуля keyword. Используйте import keyword и keyword.kwlist для получения списка ключевых слов.

11. Каково назначение функций `id()` и `type()`?

Функция `id()` возвращает уникальный идентификатор объекта в памяти, а функция `type()` возвращает тип объекта.

12. Что такое изменяемые и неизменяемые типы в Python.

Изменяемые типы данных могут быть изменены после создания, например, списки. Неизменяемые типы данных, такие как кортежи и строки, не могут быть изменены после создания.

13. Чем отличаются операции деления и целочисленного деления?

Операция деления (`/`) возвращает результат в виде числа с плавающей точкой, а операция целочисленного деления (`//`) возвращает результат в виде целого числа.

14. Какие имеются средства в языке Python для работы с комплексными числами?

Для работы с комплексными числами в Python используется встроенный тип `complex`.

15. Каково назначение и основные функции библиотеки (модуля) `math`? По аналогии с модулем `math` изучите самостоятельно назначение и основные функции модуля `cmath`.

Библиотека (модуль) `math` в Python предоставляет функции и константы для выполнения математических операций и вычислений. Основное назначение и функции библиотеки `math` включают в себя:

Вычисления математических функций: `math` предоставляет функции для выполнения различных математических операций, таких как корень, логарифмы, тригонометрические функции и др.

Константы: `math` содержит константы, такие как число  $\pi$  ( $\pi$ ) и экспонента ( $e$ ), которые можно использовать в вычислениях.

Округление и модуль: `math` предоставляет функции для округления чисел, нахождения модуля числа, а также другие функции для работы с числами.



Тригонометрические функции: `math` включает в себя тригонометрические функции, такие как синус, косинус, тангенс и другие, которые позволяют выполнять вычисления связанные с углами.

Экспоненциальные и логарифмические функции: `math` предоставляет функции для работы с экспоненциальными и логарифмическими вычислениями, такие как возведение в степень, натуральный логарифм и другие.

Библиотека `math` является полезным инструментом для выполнения разнообразных математических операций в Python, и она широко используется при решении математических задач, научных вычислений и инженерных задач.

Модуль `cmath` предоставляет аналогичные функции для комплексных чисел.

16. Каково назначение именных параметров `sep` и `end` в функции `print()`?

Параметры `sep` и `end` в функции `print()` используются для настройки разделителей между значениями и окончания вывода.

17. Каково назначение метода `format()`? Какие еще существуют средства для форматирования строк в Python? Примечание: в дополнение к рассмотренным средствам изучите самостоятельно работу с f-строками в Python.

Метод `format()` используется для форматирования строк, позволяя вставлять значения в строку. Также в Python есть f-строки для удобного форматирования строк с использованием выражений.

18. Каким образом осуществить ввод с консоли значения целочисленной и вещественной переменной в языке Python?

Для ввода значений с консоли в Python используются функции `input()` для строк, `int(input())` для целых чисел и `float(input())` для вещественных чисел.

Вывод: В ходе выполнения лабораторной работы приобретены базовые навыки программирования на этом языке, что дало возможность решать задачи с использованием Python.