

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 №2.1 (4)**  
**дисциплины**  
**«Программирование на Python»**

Выполнил:  
Ибрагимов Муса Айнудинович  
2 курс, группа ИВТ-6-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:  
Воронкин Роман Александрович

---

(подпись)

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

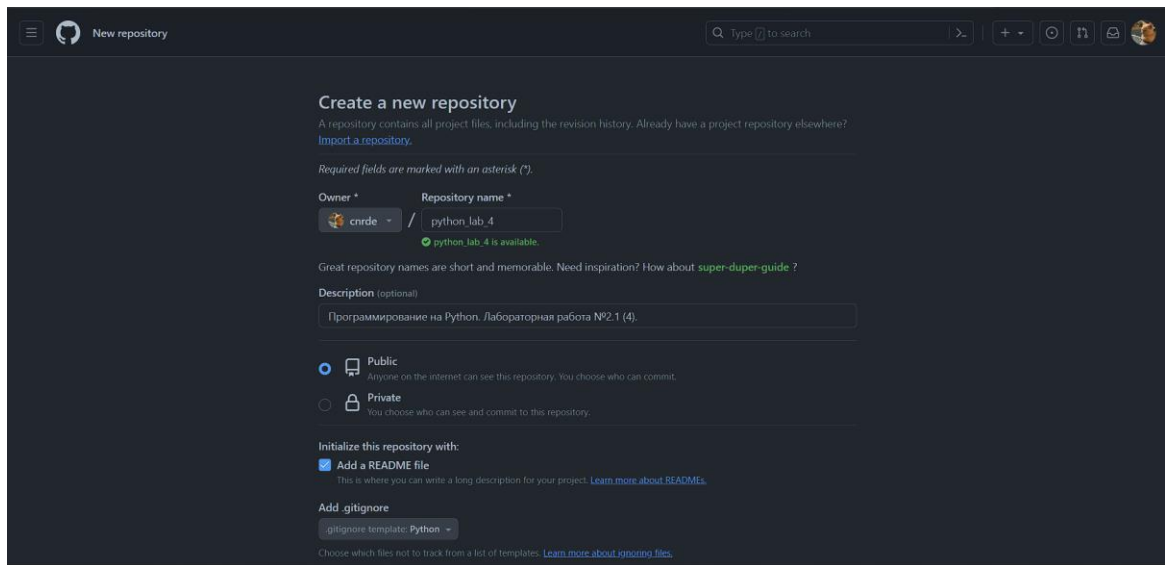
Ставрополь, 2023 г.

Тема: Основы языка Python.

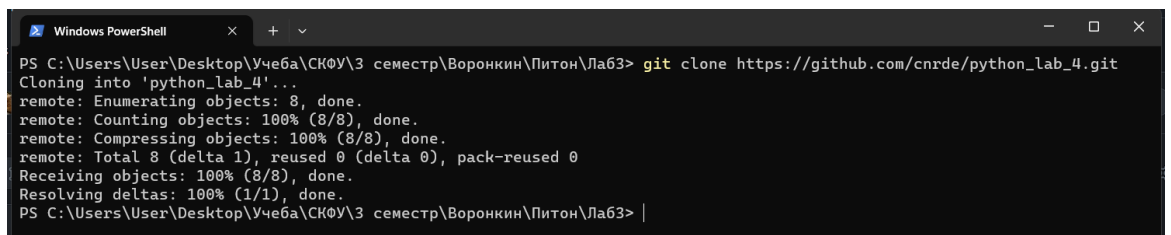
Цель: Исследование процесса установки и базовых возможностей языка Python версии 3.x.

### Практическая часть:

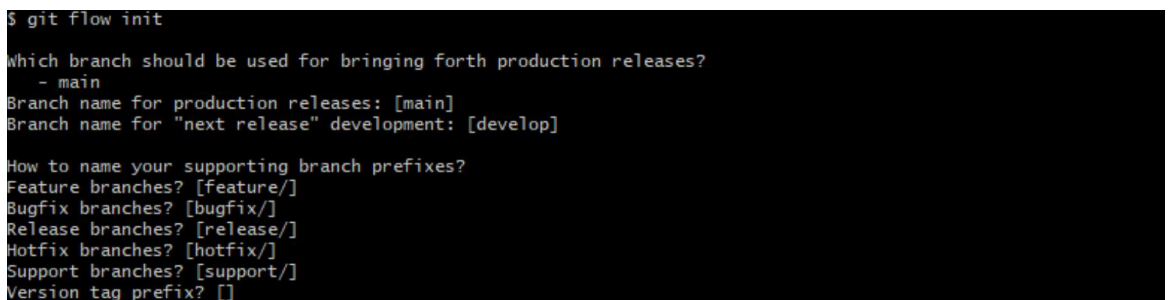
1. Изучил теоретический материал работы.
2. Создал общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.



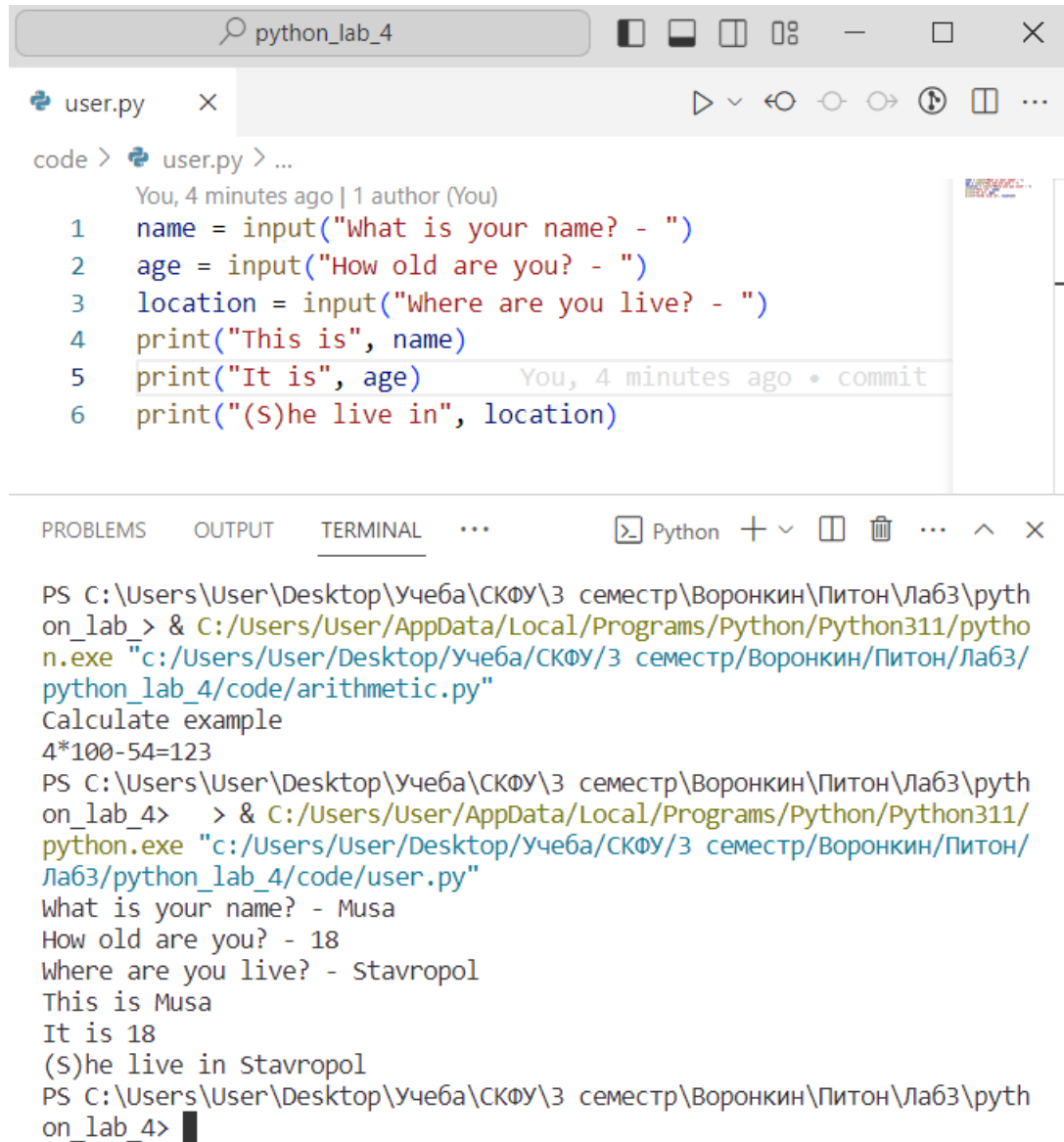
3. Выполнил клонирование созданного репозитория.



4. Организовал свой репозиторий в соответствии с моделью ветвления git-flow.



5. Написал программу user.py, которая запрашивает у пользователя имя, возраст и место жительства. После чего выводит 3 строки.



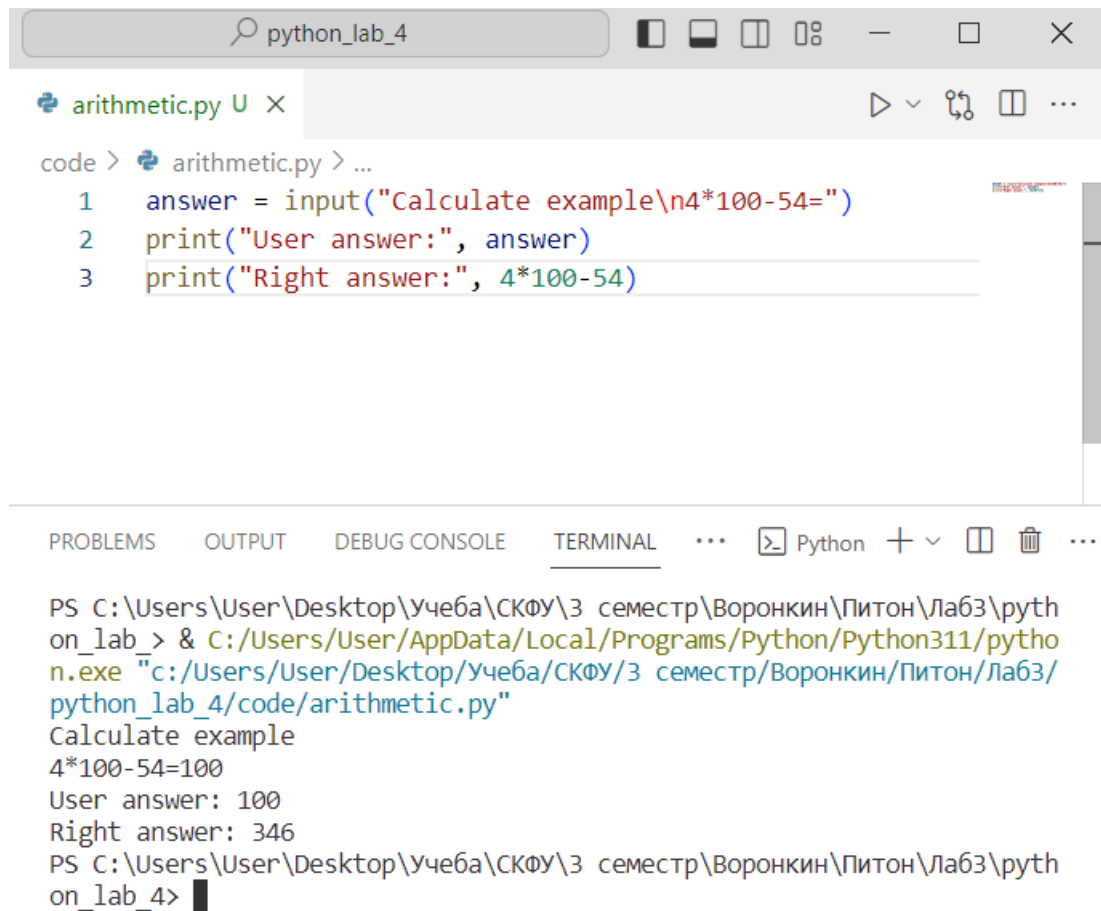
The screenshot shows a code editor window titled 'python\_lab\_4' with a file named 'user.py' open. The code in the editor is as follows:

```
code > user.py > ...
    You, 4 minutes ago | 1 author (You)
1  name = input("What is your name? - ")
2  age = input("How old are you? - ")
3  location = input("Where are you live? - ")
4  print("This is", name)
5  print("It is", age)
6  print("(S)he live in", location)
```

Below the code editor, the 'TERMINAL' tab is active, showing the command prompt output:

```
PS C:\Users\User\Desktop\учеба\СКФУ\3 семестр\Воронкин\Питон\Лаб3\python_lab_4> & C:/Users/User/AppData/Local/Programs/Python/Python311/python.exe "c:/Users/User/Desktop/учеба/СКФУ/3 семестр/Воронкин/Питон/Лаб3/python_lab_4/code/arithmetic.py"
Calculate example
4*100-54=123
PS C:\Users\User\Desktop\учеба\СКФУ\3 семестр\Воронкин\Питон\Лаб3\python_lab_4> & C:/Users/User/AppData/Local/Programs/Python/Python311/python.exe "c:/Users/User/Desktop/учеба/СКФУ/3 семестр/Воронкин/Питон/Лаб3/python_lab_4/code/user.py"
What is your name? - Musa
How old are you? - 18
Where are you live? - Stavropol
This is Musa
It is 18
(S)he live in Stavropol
PS C:\Users\User\Desktop\учеба\СКФУ\3 семестр\Воронкин\Питон\Лаб3\python_lab_4>
```

6. Написал программу arithmetic.py, которая предлагала бы пользователю решить пример  $4 * 100 - 54$ . Потом выводила бы на экран правильный ответ и ответ пользователя.



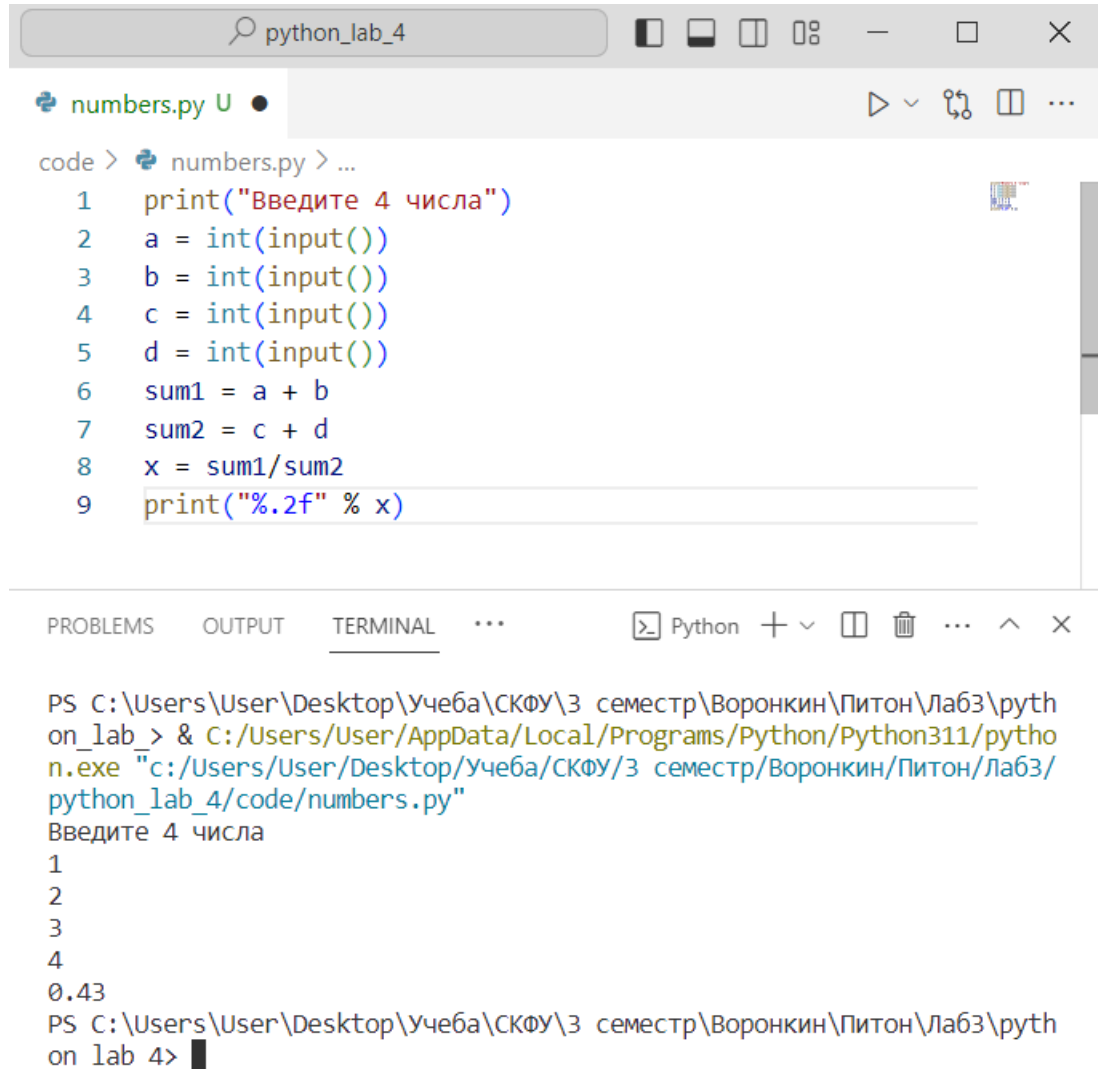
The screenshot shows an IDE window titled 'python\_lab\_4'. The editor displays a Python script named 'arithmetic.py' with the following code:

```
code > arithmetic.py > ...
1  answer = input("Calculate example\n4*100-54=")
2  print("User answer:", answer)
3  print("Right answer:", 4*100-54)
```

Below the editor, the 'TERMINAL' tab is active, showing the command prompt output:

```
PS C:\Users\User\Desktop\Учеба\СКФУ\3 семестр\Воронкин\Питон\Лаб3\python_lab_4> & C:/Users/User/AppData/Local/Programs/Python/Python311/python.exe "c:/Users/User/Desktop/Учеба/СКФУ/3 семестр/Воронкин/Питон/Лаб3/python_lab_4/code/arithmetic.py"
Calculate example
4*100-54=100
User answer: 100
Right answer: 346
PS C:\Users\User\Desktop\Учеба\СКФУ\3 семестр\Воронкин\Питон\Лаб3\python_lab_4>
```

7. Написал программу numbers.py, которая запрашивает у пользователя 4 числа, отдельно складывает первые два и вторые два, затем делит первую сумму на вторую, после выводит результат на экран с точностью до сотых.



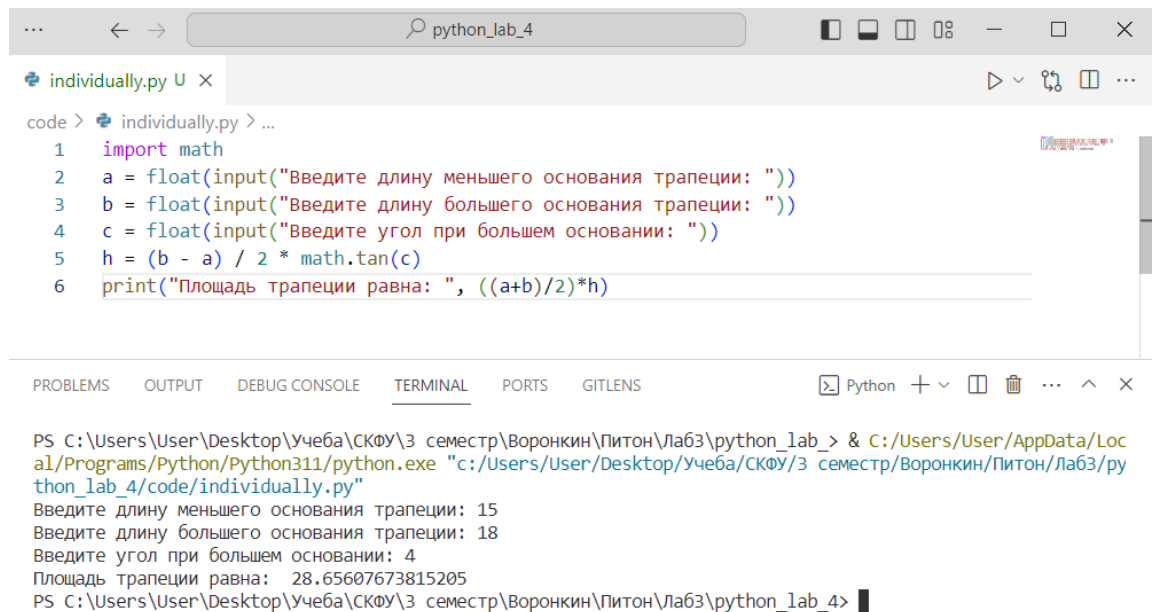
The screenshot shows a code editor window titled 'python\_lab\_4' with a file named 'numbers.py' open. The code in the editor is as follows:

```
code > numbers.py > ...
1  print("Введите 4 числа")
2  a = int(input())
3  b = int(input())
4  c = int(input())
5  d = int(input())
6  sum1 = a + b
7  sum2 = c + d
8  x = sum1/sum2
9  print("%.2f" % x)
```

Below the code editor, the 'TERMINAL' tab is active, showing the command prompt execution of the script:

```
PS C:\Users\User\Desktop\учеба\СКФУ\3 семестр\Воронкин\Питон\Лаб3\python_lab_4> & C:/Users/User/AppData/Local/Programs/Python/Python311/python.exe "c:/Users/User/Desktop/учеба/СКФУ/3 семестр/Воронкин/Питон/Лаб3/python_lab_4/code/numbers.py"
Введите 4 числа
1
2
3
4
0.43
PS C:\Users\User\Desktop\учеба\СКФУ\3 семестр\Воронкин\Питон\Лаб3\python_lab_4>
```

8. Индивидуальное задание (Вариант 8). Даны основания равнобедренной трапеции и угол при большем основании. Найти площадь трапеции.



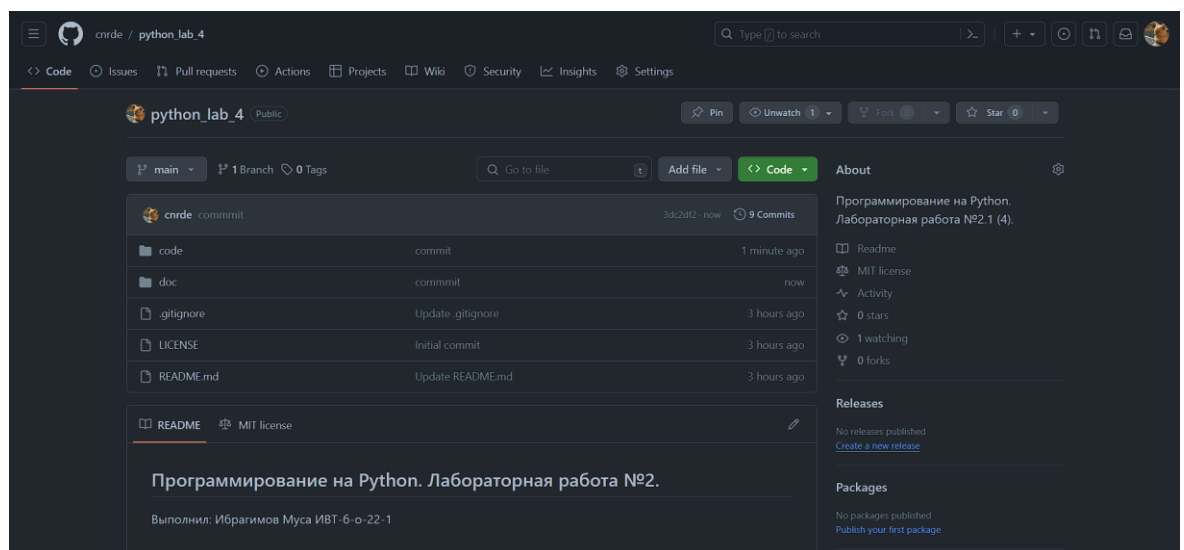
The screenshot shows a Python IDE with a file named `individually.py`. The code defines a function to calculate the area of an isosceles trapezoid given its bases and the angle at the larger base. The terminal output shows the user inputting values 15, 18, and 4, resulting in an area of 28.65607673815205.

```
code > individually.py > ...
1 import math
2 a = float(input("Введите длину меньшего основания трапеции: "))
3 b = float(input("Введите длину большего основания трапеции: "))
4 c = float(input("Введите угол при большем основании: "))
5 h = (b - a) / 2 * math.tan(c)
6 print("Площадь трапеции равна: ", ((a+b)/2)*h)
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS GITLENS Python + -

```
PS C:\Users\User\Desktop\Учеба\СКФУ\3 семестр\Воронкин\Питон\Лаб3\python_lab_ > C:/Users/User/AppData/Local/Programs/Python/Python311/python.exe "c:/Users/User/Desktop/Учеба/СКФУ/3 семестр/Воронкин/Питон/Лаб3/python_lab_4/code/individually.py"
Введите длину меньшего основания трапеции: 15
Введите длину большего основания трапеции: 18
Введите угол при большем основании: 4
Площадь трапеции равна: 28.65607673815205
PS C:\Users\User\Desktop\Учеба\СКФУ\3 семестр\Воронкин\Питон\Лаб3\python_lab_4>
```

9. Выполнил коммит файлов `user.py`, `arithmetic.py`, `numbers.py` и `individual.py` в репозиторий `git` в ветку для разработки.



### Ответы на контрольные вопросы:

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

Linux: Чаще всего интерпретатор Python уже входит в состав дистрибутива.

Windows: Оsn. этапы установки Python на Windows:

- 1) Скачать дистрибутив с официального сайта;
- 2) Запустить скачанный установочный файл;
- 3) Выбрать способ установки;
- 4) Отметить необходимые опции установки;
- 5) Выбрать место установки;
- 6) Готово.

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

Пакет Anaconda содержит версии языка Python 2 и 3, набор наиболее часто используемых библиотек и удобную среду разработки и исполнения, запускаемую в браузере, а также на Anaconda удобнее запускать примеры.

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

Для выполнения проверки работоспособности Anaconda необходимо вначале запустить командный процессор с поддержкой виртуальных окружений Anaconda. В появившейся командной строке необходимо ввести `> jupyter notebook`, в результате чего отобразиться процесс загрузки веб-среды Jupyter Notebook, после чего запустится веб-сервер и среда разработки в браузере. Создать ноутбук для разработки, для этого нажать на кнопку New и в появившемся списке выбрать Python. В результате будет создана новая страница в браузере с ноутбуком. Ввести в первой ячейке команду `print("Hello, World!")` и нажать `Alt+Enter` на компьютере. Ниже ячейки должна появиться соответствующая надпись.

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

Указать путь до интерпретатора в настройках IDE, для этого:

- 1) Нажмите на шестеренку в верхнем правом углу, выберите "Add..".
- 2) Далее выберите "System Interpreter";
- 3) Нажмите на 3 точки "..." справа от поля с выбором интерпретатора;
- 4) Укажите путь до интерпретатора.

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

Сочетанием клавиш Shift+F10.

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

Интерактивный.

Python можно использовать как калькулятор для различных вычислений, а если дополнительно подключить необходимые математические библиотеки, то по своим возможностям он становится практически равным таким пакетам как Matlab, Octave и т.п.

Проектный.

В этом режиме сначала записывается вся программа, а потом эта программа выполняется полностью.

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

Т. к. в языке программирования Python проверка типа происходит во время выполнения, а не компиляции.

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

Типы в Python:

1. None
2. Логические переменные
3. Числа
4. Списки
5. Строки
6. Бинарные списки
7. Множества
8. Словари

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



Для того, чтобы объявить и сразу инициализировать переменную необходимо написать её имя, потом поставить знак равенства и значение, с которым эта переменная будет создана.

При инициализации переменной, на уровне интерпретатора, создается целочисленный объект, который имеет некоторый идентификатор, значение и тип. Посредством оператора “=” создается ссылка между переменной и объектом.

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

Список ключевых слов можно получить непосредственно в программе, для этого нужно подключить модуль keyword и воспользоваться командой keyword.kwlist.

### **11. Каково назначение функций id() и type()?**

Функция id() предназначена для получения значения идентичности объекта.

С помощью функции type() можно получить тип конкретного объекта.

### **12. Что такое изменяемые и неизменяемые типы в Python?**

К неизменяемым (immutable) типам относятся: целые числа (int), числа с плавающей точкой (float), комплексные числа (complex), логические переменные (bool), кортежи (tuple), строки (str) и неизменяемые множества (frozen set).

К изменяемым (mutable) типам относятся: списки (list), множества (set), словари (dict).

### **13. Чем отличаются операции деления и целочисленного деления?**

При целочисленном делении отбрасывается дробная часть от деления чисел, при операции деления дробная часть не отбрасывается.

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

Для создания комплексного числа можно использовать функцию complex(a, b), в которую, в качестве первого аргумента, передается действительная часть, в качестве второго – мнимая. Либо записать число в

виде  $a + bj$ . Комплексные числа можно складывать, вычитать, умножать, делить и возводить в степень. У комплексного числа можно извлечь действительную(`x.real`) и мнимую части(`x.imag`).

Для получения комплексно-сопряженного число необходимо использовать метод `conjugate()`.

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

Для выполнения математических операций необходим модуль `math`.

Основные операции библиотеки `math`:

`math.ceil(x)` - возвращает ближайшее целое число большее, чем  $x$ .

`math.fabs(x)` - возвращает абсолютное значение числа.

`math.factorial(x)` - вычисляет факториал  $x$ .

`math.floor(x)` - возвращает ближайшее целое число меньшее, чем  $x$ .

`math.exp(x)` - вычисляет  $e^{**}x$ .

`math.log2(x)` - логарифм по основанию 2.

`math.log10(x)` - логарифм по основанию 10.

`math.log(x[, base])` - по умолчанию вычисляет логарифм по основанию  $e$ , дополнительно можно указать основание логарифма.

`math.pow(x, y)` - вычисляет значение  $x$  в степени  $y$ .

`math.sqrt(x)` - корень квадратный от  $x$ .

`math.cos(x)` - косинус от  $x$ .

`math.sin(x)` - синус от  $x$ .

`math.tan(x)` - тангенс от  $x$ .

`math.acos(x)` - арккосинус от  $x$ .

`math.asin(x)` - арксинус от  $x$ .

`math.atan(x)` - арктангенс от  $x$ .

`math.pi` - число  $\pi$ .

`math.e` - число  $e$ .

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

Через параметр `sep` можно указать отличный от пробела разделитель строк.

Параметр `end` позволяет указывать, что делать, после вывода строки.

**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.**

Форматирование может выполняться в так называемом старом стиле или с помощью строкового метода `format`.

Символы `%s`, `%d`, `%f` подставляются значения переменных. Буквы `s`, `d`, `f` обозначают типы данных – строку, целое число, вещественное число.

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

Указать перед `input` тип данных: `int(input())`.

**Вывод:** Исследовал процесс установки и базовые возможности языка Python версии 3.x.