

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»  
ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ**

**Отчет о лабораторной работе №4 по дисциплине основы программной  
инженерии**

Выполнил:  
Выходцев Егор Дмитриевич,  
2 курс, группа ПИЖ-б-о-20-1,

Проверил:  
Доцент кафедры инфокоммуникаций,  
Воронкин Р.А.

Ставрополь, 2021 г.

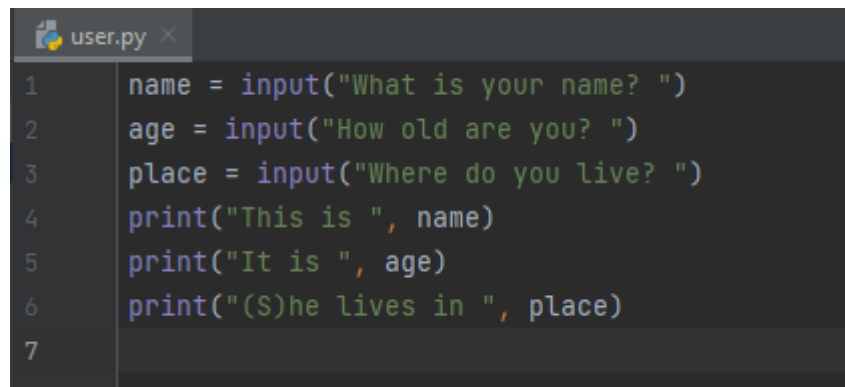
## 1. Основы языка Python

### 1.1 Создание ветки «develop» (рис. 1).

```
C:\Users\Evil\work\Laboratornaya4>git checkout develop
Switched to a new branch 'develop'
Branch 'develop' set up to track remote branch 'develop' from 'origin'.
C:\Users\Evil\work\Laboratornaya4>
```

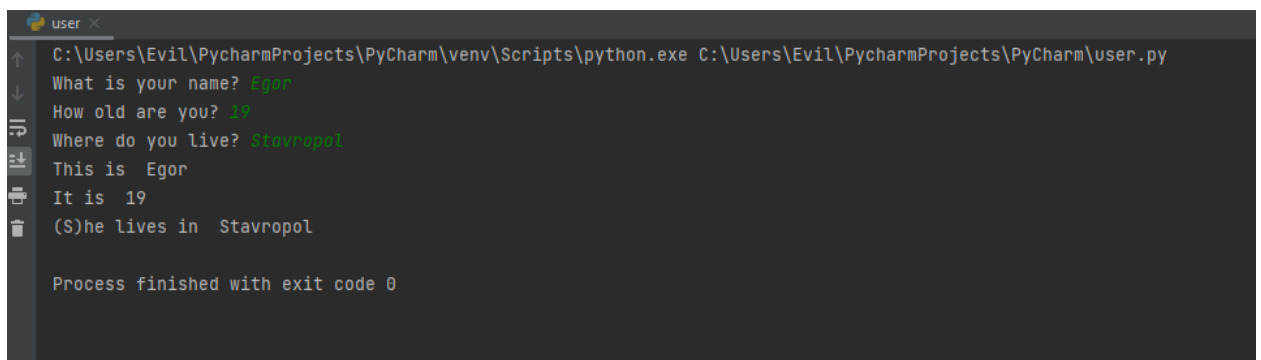
Рисунок 1 – Процесс создания ветки отслеживания удалённой ветки «develop»

### 1.2 Код программы «user.py» (рис. 2, 3).



```
user.py x
1 name = input("What is your name? ")
2 age = input("How old are you? ")
3 place = input("Where do you live? ")
4 print("This is ", name)
5 print("It is ", age)
6 print("(S)he lives in ", place)
7
```

Рисунок 2 – Вид программы «user.py»

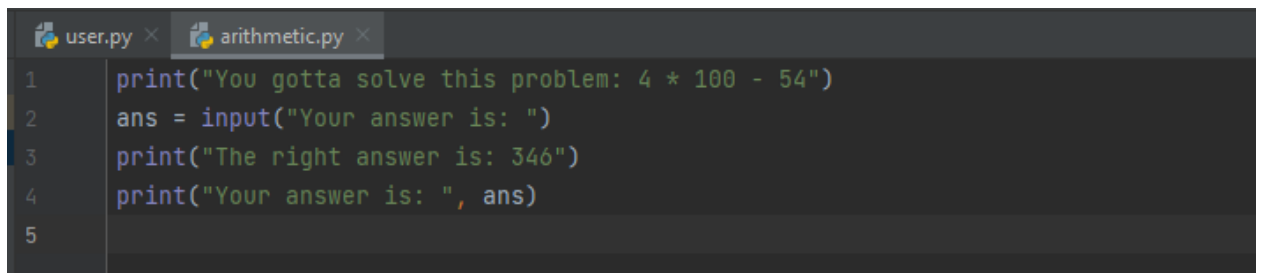


```
user x
C:\Users\Evil\PycharmProjects\PyCharm\venv\Scripts\python.exe C:\Users\Evil\PycharmProjects\PyCharm\user.py
What is your name? Egor
How old are you? 19
Where do you live? Stavropol
This is Egor
It is 19
(S)he lives in Stavropol

Process finished with exit code 0
```

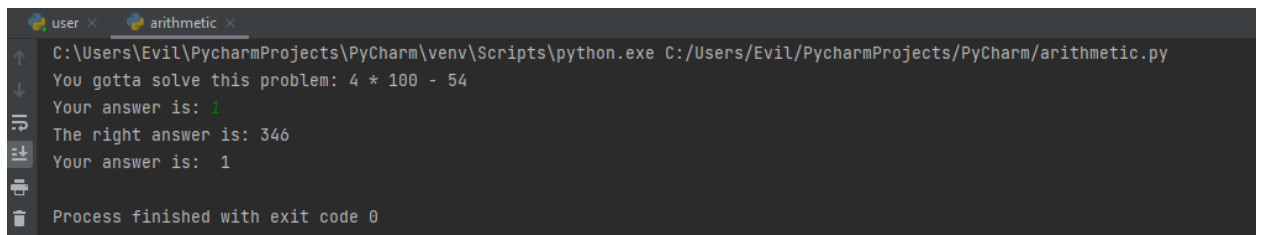
Рисунок 3 – Пример работы программы

### 1.3 Код программы «arithmetic.py» (рис. 4, 5).



```
1 print("You gotta solve this problem: 4 * 100 - 54")
2 ans = input("Your answer is: ")
3 print("The right answer is: 346")
4 print("Your answer is: ", ans)
5
```

Рисунок 4 – Вид программы «arithmetic.py»

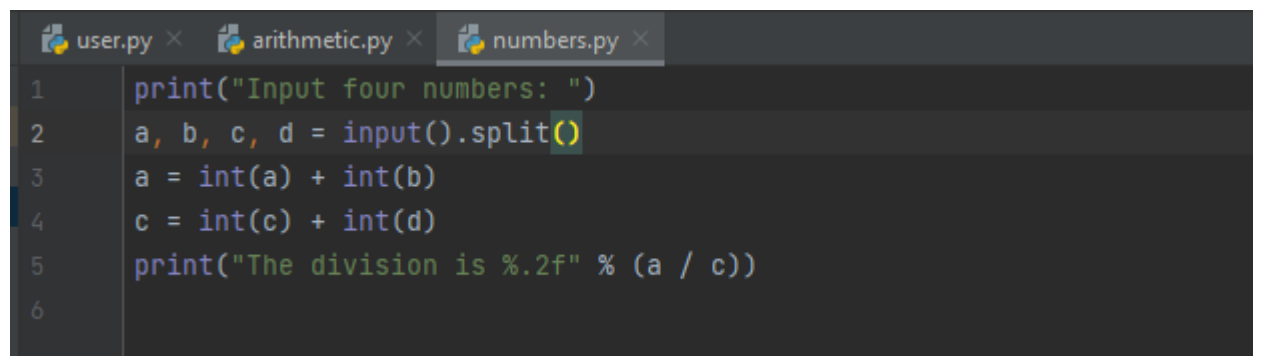


```
C:\Users\Evil\PycharmProjects\PyCharm\venv\Scripts\python.exe C:/Users/Evil/PycharmProjects/PyCharm/arithmetic.py
You gotta solve this problem: 4 * 100 - 54
Your answer is: 1
The right answer is: 346
Your answer is: 1
Process finished with exit code 0
```

Рисунок 5 – Пример работы программы

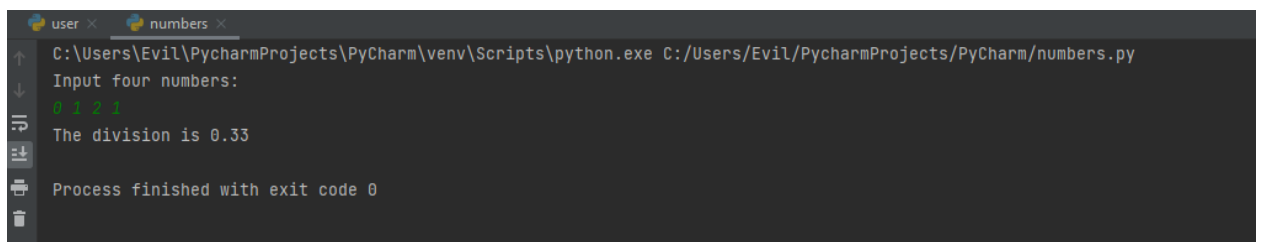
Переводить строку в число не нужно, так как вычислительные операции в данной программе не совершаются. Ответ пользователя можно вывести как строку.

#### 1.4 Код программы «numbers.py» (рис. 6, 7).



```
1 print("Input four numbers: ")
2 a, b, c, d = input().split()
3 a = int(a) + int(b)
4 c = int(c) + int(d)
5 print("The division is %.2f" % (a / c))
6
```

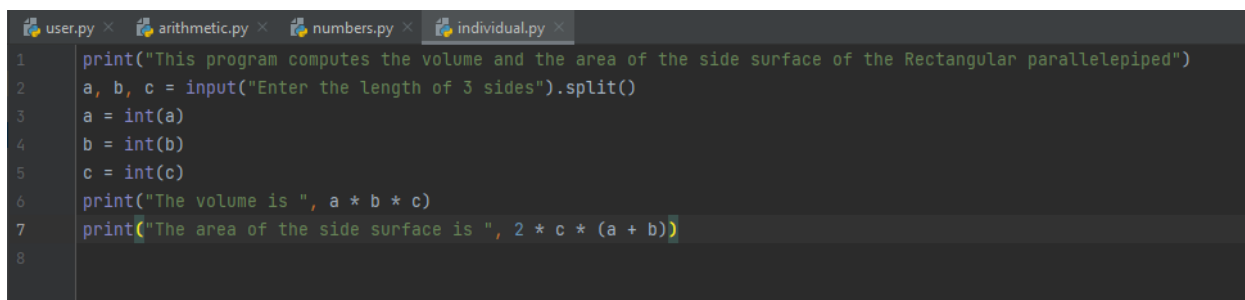
Рисунок 6 – Вид программы «numbers.py»



```
C:\Users\Evil\PycharmProjects\PyCharm\venv\Scripts\python.exe C:/Users/Evil/PycharmProjects/PyCharm/numbers.py
Input four numbers:
0 1 2 3
The division is 0.33
Process finished with exit code 0
```

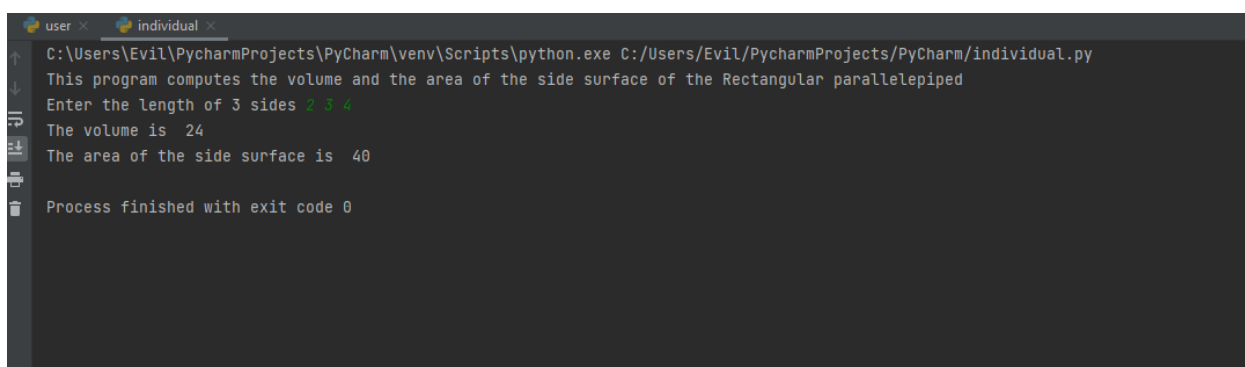
Рисунок 7 – Пример работы программы

### 1.5 Код программы «individual.py», вариант №5 (рис. 8, 9).



```
1 print("This program computes the volume and the area of the side surface of the Rectangular parallelepiped")
2 a, b, c = input("Enter the length of 3 sides").split()
3 a = int(a)
4 b = int(b)
5 c = int(c)
6 print("The volume is ", a * b * c)
7 print("The area of the side surface is ", 2 * c * (a + b))
8
```

Рисунок 8 – Вид программы «individual.py»

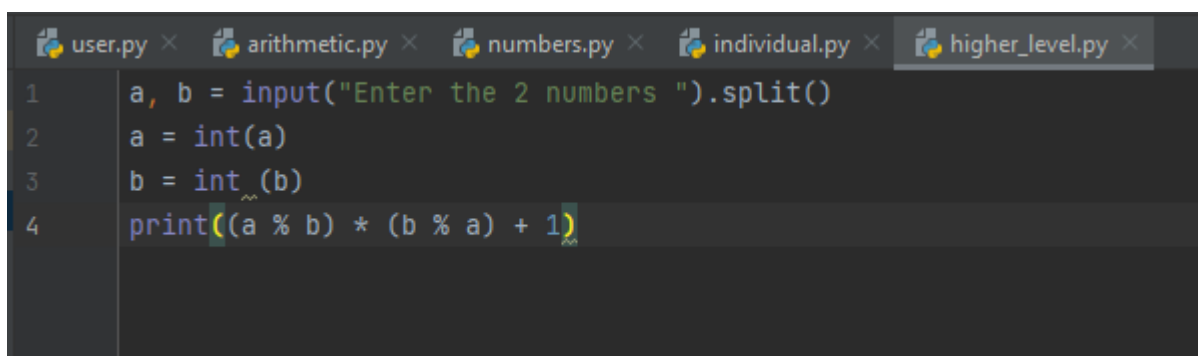


```
user x individual x
C:\Users\Evil\PycharmProjects\PyCharm\venv\Scripts\python.exe C:/Users/Evil/PycharmProjects/PyCharm/individual.py
This program computes the volume and the area of the side surface of the Rectangular parallelepiped
Enter the length of 3 sides 2 3 4
The volume is 24
The area of the side surface is 40
Process finished with exit code 0
```

Рисунок 9 – Пример работы программы

### 1.6 Задача повышенной сложности №8 (рис. 10).

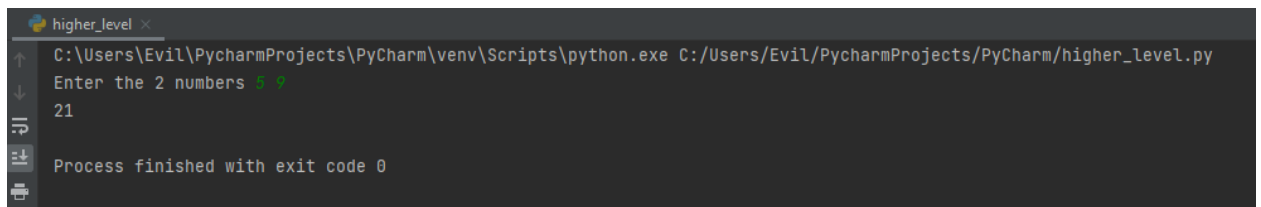
Условие: Даны два целых числа  $a$  и  $b$ . Если  $a$  делится на  $b$  или  $b$  делится на  $a$ , то вывести 1, иначе – любое другое число. Условные операторы и операторы цикла не использовать.



```
1 a, b = input("Enter the 2 numbers ").split()
2 a = int(a)
3 b = int(b)
4 print((a % b) * (b % a) + 1)
```

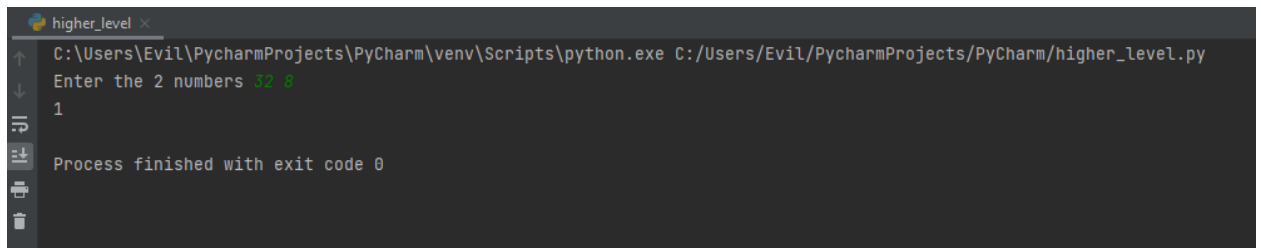
Рисунок 10 – Код программы

Примеры выполнения программы представлены на рисунках 11 и 12.



```
higher_level x
C:\Users\Evil\PycharmProjects\PyCharm\venv\Scripts\python.exe C:/Users/Evil/PycharmProjects/PyCharm/higher_level.py
Enter the 2 numbers 8 21
21
Process finished with exit code 0
```

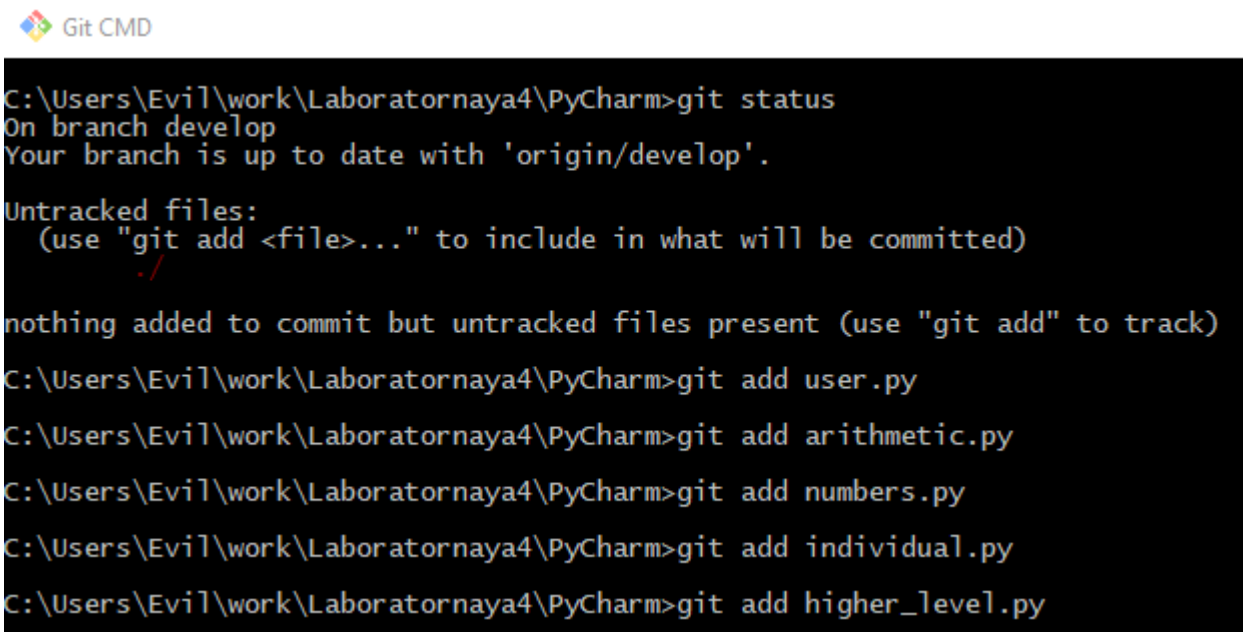
Рисунок 11 – Вывод программы при вводе двух чисел, не делящихся друг на друга



```
higher_level x
C:\Users\Evil\PycharmProjects\PyCharm\venv\Scripts\python.exe C:/Users/Evil/PycharmProjects/PyCharm/higher_level.py
Enter the 2 numbers 32 8
1
Process finished with exit code 0
```

Рисунок 12 – Вывод программы при вводе двух чисел, где одно из них делится на другое

1.7 Добавление файлов в ветке «develop» под версионный контроль (рис. 13).



```
Git CMD
C:\Users\Evil\work\Laboratornaya4\PyCharm>git status
On branch develop
Your branch is up to date with 'origin/develop'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  ./

nothing added to commit but untracked files present (use "git add" to track)
C:\Users\Evil\work\Laboratornaya4\PyCharm>git add user.py
C:\Users\Evil\work\Laboratornaya4\PyCharm>git add arithmetic.py
C:\Users\Evil\work\Laboratornaya4\PyCharm>git add numbers.py
C:\Users\Evil\work\Laboratornaya4\PyCharm>git add individual.py
C:\Users\Evil\work\Laboratornaya4\PyCharm>git add higher_level.py
```

Рисунок 13 – Добавлены файлы программ

1.8 Слияние веток (рис. 14).

```
Git CMD
C:\Users\Evil\work\Laboratornaya4>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

C:\Users\Evil\work\Laboratornaya4>git merge develop
Updating 4d79c33..7998251
Fast-forward
 PyCharm/arithmic.py      | 4 ++++
 PyCharm/higher_level.py | 4 ++++
 PyCharm/individual.py    | 7 ++++++
 PyCharm/numbers.py       | 5 +++++
 PyCharm/user.py          | 6 ++++++
 5 files changed, 26 insertions(+)
 create mode 100644 PyCharm/arithmic.py
 create mode 100644 PyCharm/higher_level.py
 create mode 100644 PyCharm/individual.py
 create mode 100644 PyCharm/numbers.py
 create mode 100644 PyCharm/user.py

C:\Users\Evil\work\Laboratornaya4>git push
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (8/8), 1.13 KiB | 577.00 KiB/s, done.
Total 8 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/EgorVyhodcev/Laboratornaya4.git
 4d79c33..7998251  main -> main

C:\Users\Evil\work\Laboratornaya4>
```

Рисунок 14 – Слияние ветки «main» с веткой «develop» и загрузка изменений

## 2. Ответы на вопросы

1. Windows: Для операционной системы Windows дистрибутив распространяется либо в виде исполняемого файла (с расширением exe), либо в виде архивного файла (с расширением zip). Если вы используете Windows 7, не забудьте установить Service Pack 1!

Порядок установки.

1) Запустите скачанный установочный файл.

2) Выберите способ установки.

В данном окне предлагается два варианта Install Now и Customize installation. При выборе Install Now, Python установится в папку по указанному пути. Помимо самого интерпретатора будет установлен IDLE (интегрированная среда разработки), pip (пакетный менеджер) и

документация, а также будут созданы соответствующие ярлыки и установлены связи файлов, имеющие расширение .py с интерпретатором Python. Customize installation – это вариант настраиваемой установки. Опция Add python 3.5 to PATH нужна для того, чтобы появилась возможность запускать интерпретатор без указания полного пути до исполняемого файла при работе в командной строке.

3) Отметьте необходимые опции установки (доступно при выборе Customize installation)

На этом шаге нам предлагается отметить дополнения, устанавливаемые вместе с интерпретатором Python. Рекомендуется выбрать все опции.

Documentation – установка документаций.

pip – установка пакетного менеджера pip.

tcl/tk and IDLE – установка интегрированной среды разработки (IDLE) и библиотеки для построения графического интерфейса (tkinter).

4) Выберите место установки (доступно при выборе Customize installation)

Помимо указания пути, данное окно позволяет внести дополнительные изменения в процесс установки с помощью опций:

Install for all users – Установить для всех пользователей. Если не выбрать данную опцию, то будет предложен вариант инсталляции в папку пользователя, устанавливающего интерпретатор.

Associate files with Python – Связать файлы, имеющие расширение .py, с Python. При выборе данной опции будут внесены изменения в Windows, позволяющие запускать Python скрипты по двойному щелчку мыши.

Create shortcuts for installed applications – Создать ярлыки для запуска приложений.

Add Python to environment variables – Добавить пути до интерпретатора Python в переменную PATH.

Precompile standard library – Провести прекомпиляцию стандартной библиотеки.

Последние два пункта связаны с загрузкой компонентов для отладки, их мы устанавливать не будем.

Linux: Чаще всего интерпретатор Python уже входит в состав дистрибутива. Это можно проверить, набрав в терминале «python» или «python3»

В первом случае, вы запустите Python 2 во втором – Python 3. В будущем, скорее всего, во всех дистрибутивах Linux, включающих Python, будет входить только третья версия. Если у вас, при попытке запустить Python, выдается сообщение о том, что он не установлен, или установлен, но не тот, что вы хотите, то у вас есть два пути: а) собрать Python из исходников; б) взять из репозитория.

Для установки из репозитория в Ubuntu воспользуйтесь командой «sudo apt-get install python3»

2. Для удобства запуска примеров и изучения языка Python, настоятельно рекомендуется установить на свой ПК пакет Anaconda. Этот пакет включает в себя интерпретатор языка Python (есть версии 2 и 3), набор наиболее часто используемых библиотек и удобную среду разработки и исполнения, запускаемую в браузере.

3. Для выполнения проверки работоспособности Anaconda необходимо вначале запустить командный процессор с поддержкой виртуальных окружений Anaconda. В Windows это можно сделать, выбрав следующий пункт главного меню системы Пуск Anaconda3 (64-bit) Anaconda Prompt. В появившейся командной строке необходимо ввести «jupyter notebook», запустится веб-сервер и среда разработки в браузере. Создайте ноутбук для



разработки, для этого нажмите на кнопку New (в правом углу окна) и в появившемся списке выберете Python. В результате будет создана новая страница в браузере с ноутбуком. Введите в первой ячейке команду «`print("Hello, World!")`» и нажмите Alt+Enter на клавиатуре. Ниже ячейки должна появиться соответствующая надпись.

4. При создании нового проекта нужно будет указать путь до него и интерпретатор.

5. После создания нового проекта нужно добавить python файл в проект.

6. Интерактивный режим – непосредственное выполнение команд одна за другой в консоли. Пакетный режим – запуск программы из файла.

7. Потому что тип переменной определяется непосредственно при выполнении программы.

8. К основным встроенным типам относятся:

1. None (неопределенное значение переменной)

2. Логические переменные (Boolean Type)

3. Числа (Numeric Type)

1. int – целое число

2. float – число с плавающей точкой

3. complex – комплексное число

4. Списки (Sequence Type)

1. list – список

2. tuple – кортеж

3. range – диапазон

5. Строки (Text Sequence Type )

1. str

## 6. Бинарные списки (Binary Sequence Types)

1. bytes – байты

2. bytearray – массивы байт

3. memoryview – специальные объекты для доступа к внутренним данным объекта через protocol buffer

## 7. Множества (Set Types)

1. set – множество

2. frozenset – неизменяемое множество

## 8. Словари (Mapping Types)

1. dict – словарь

9. Для того, чтобы объявить и сразу инициализировать переменную необходимо написать её имя, потом поставить знак равенства и значение, с которым эта переменная будет создана. Целочисленное значение 5 в рамках языка Python по сути своей является объектом. Объект, в данном случае – это абстракция для представления данных, данные – это числа, списки, строки и т.п. При этом, под данными следует понимать, как непосредственно сами объекты, так и отношения между ними (об этом чуть позже). Каждый объект имеет три атрибута – это идентификатор, значение и тип. Идентификатор – это уникальный признак объекта, позволяющий отличать объекты друг от друга, а значение – непосредственно информация, хранящаяся в памяти, которой управляет интерпретатор. При инициализации переменной, на уровне интерпретатора, происходит следующее:

создается целочисленный объект 5 (можно представить, что в этот момент создается ячейка и 5 кладется в эту ячейку);

данный объект имеет некоторый идентификатор, значение: 5, и тип: целое число;

посредством оператора “=” создается ссылка между переменной b и целочисленным

объектом 5 (переменная b ссылается на объект 5).

Имя переменной не должно совпадать с ключевыми словами интерпретатора Python.

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

11. Для того, чтобы посмотреть на объект с каким идентификатором ссылается данная переменная, можно использовать функцию id(). Тип переменной можно определить с помощью функции type().

12. 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).

Как уже было сказано ранее, при создании переменной, вначале создается объект, который имеет уникальный идентификатор, тип и значение, после этого переменная может ссылаться на созданный объект.

Неизменяемость типа данных означает, что созданный объект больше не изменяется. Например, если мы объявим переменную k = 15, то будет создан объект со значением 15, типа int и идентификатором, который можно узнать с помощью функции id().

13. При обычном делении результатом операции будет вещественное число с плавающей точкой. При целочисленном делении результатом будет целое число, показывающее количество целых чисел  $b$  в числе  $a$ , к примеру.

14. 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 и мнимую части.

15. 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. Библиотека «cmath» содержит в себе функции для работы с комплексными числами.

16. Через параметр «sep» можно указать отличный от пробела разделитель строк. Параметр «end» позволяет указывать, что делать, после вывода строки. По умолчанию происходит переход на новую строку. Однако это действие можно отменить, указав любой другой символ или строку.

17. В строке в фигурных скобках указаны номера данных, которые будут 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.

«f-строки»: Форматирование, которое появилось в Python 3.6 (PEP 498). Этот способ похож на форматирование с помощью метода `format()`, но гибче, читабельней и быстрее. Пример:

```
>>> name = "Дмитрий"
```

```
>>> age = 25
```

```
>>> print(f"Меня зовут {name} Мне {age} лет.")
```

```
>>> Меня зовут Дмитрий. Мне 25 лет.
```

18. Даже, если ввести число, функция `input()` все равно вернет его строковое представление. Чтобы получить число, нужно использовать функции преобразования типов. Пример:

```
qtyOranges = int(input("Сколько апельсинов? "))
```

```
priceOrange = float(input("Цена одного апельсина? "))
```

```
sumOranges = qtyOranges * priceOrange
```