

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ
ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное
учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Кафедра инфокоммуникаций

«Основы языка Python»

Отчет по лабораторной работе № 2.1

по дисциплине «Основы программной инженерии»

Выполнил студент группы ПИЖ-б-о-21-1

Гребенкин Е. А. « » 2022г.

Подпись студента _____

Работа защищена « » _____2022г.

Проверил Воронкин Р.А. _____

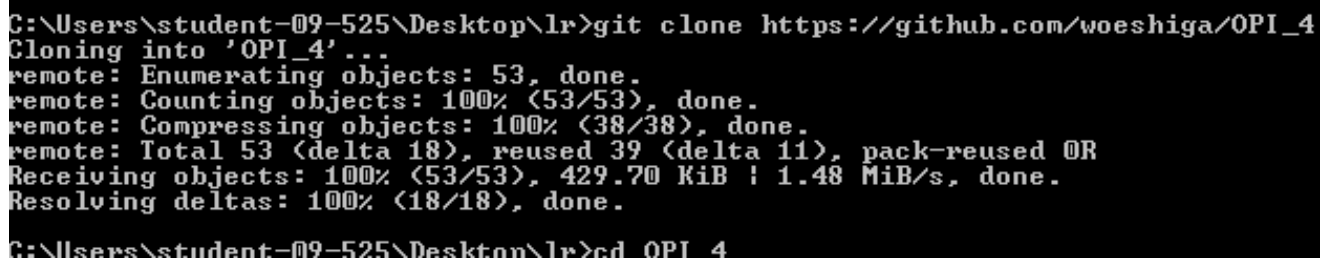
(подпись)

Ставрополь 2022

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

Ход работы:

1. Изучить теоретический материал работы.
2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.
3. Выполните клонирование созданного репозитория.



```
C:\Users\student-09-525\Desktop\lr>git clone https://github.com/woeshiga/OPI_4
Cloning into 'OPI_4'...
remote: Enumerating objects: 53, done.
remote: Counting objects: 100% (53/53), done.
remote: Compressing objects: 100% (38/38), done.
remote: Total 53 (delta 18), reused 39 (delta 11), pack-reused 0R
Receiving objects: 100% (53/53), 429.70 KiB | 1.48 MiB/s, done.
Resolving deltas: 100% (18/18), done.
C:\Users\student-09-525\Desktop\lr>cd OPI_4
```

Рисунок 2 – Клонирование репозитория

4. Организуйте свой репозиторий в соответствие с моделью ветвления gitflow.

```
C:\Users\dimu7\Desktop\ОПИ\Laba_4>git flow init

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main] main
Branch name for "next release" development: [develop] develop

How to name your supporting branch prefixes?
Feature branches? [feature/] feature
Bugfix branches? [bugfix/] bugfix
Release branches? [release/] release
Hotfix branches? [hotfix/] hotfix
Support branches? [support/] support
Version tag prefix? [] tags
Hooks and filters directory? [C:/Users/dimu7/Desktop/ОПИ/Laba_4/.git/hooks]

C:\Users\dimu7\Desktop\ОПИ\Laba_4>git branch
* develop
main
```

Рисунок 3 – Модель ветвления git-flow

5. Создайте проект PyCharm в папке репозитория.
6. Решите следующие задачи с помощью языка программирования Python3 и IDE PyCharm:
7. Напишите программу (файл user.py), которая запрашивала бы у пользователя:
 - его имя (например, "What is your name?")
 - возраст ("How old are you?")
 - место жительства ("Where are you live?")

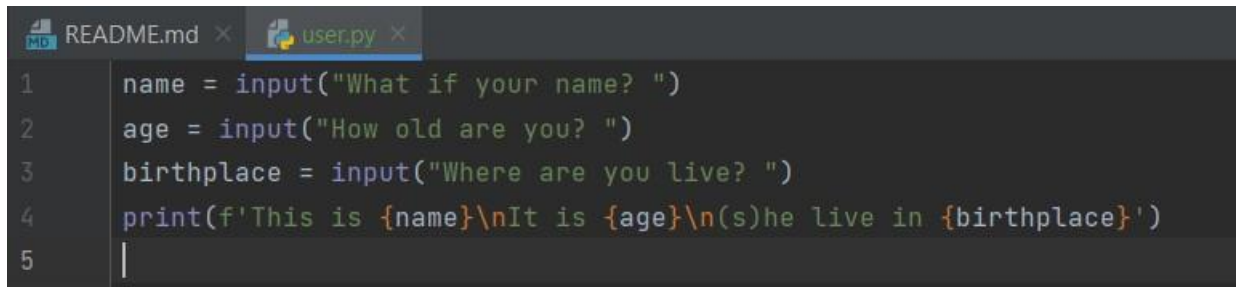
После этого выводила бы три строки:

"This is `имя`"

"It is `возраст`"

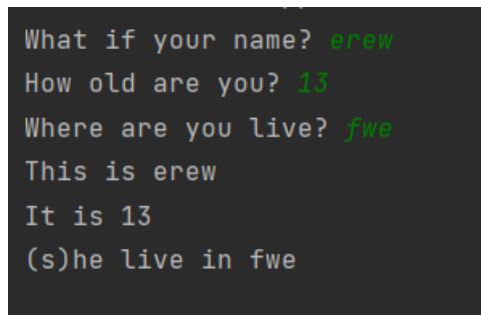
"(S)he lives in `место_жительства`"

Вместо имя, возраст, место_жительства должны быть данные, введенные пользователем.



```
1 name = input("What if your name? ")
2 age = input("How old are you? ")
3 birthplace = input("Where are you live? ")
4 print(f'This is {name}\nIt is {age}\n(s)he live in {birthplace}')
5
```

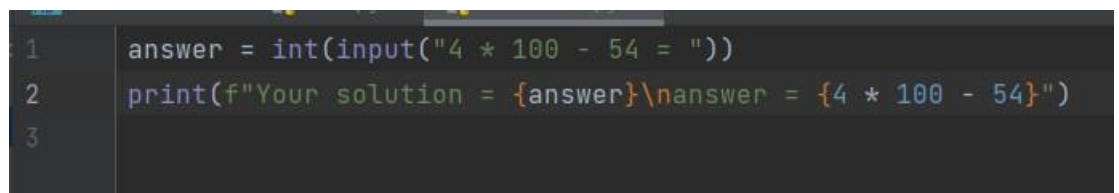
Рисунок 4 – Код программы



```
What if your name? erew
How old are you? 13
Where are you live? fwe
This is erew
It is 13
(s)he live in fwe
```

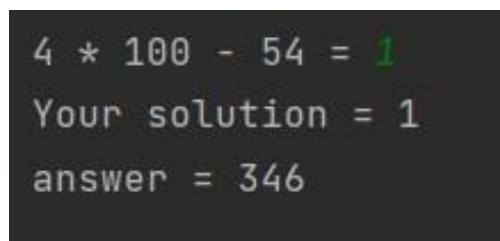
Рисунок 5 – Результат ее выполнения в IDE PyCharm

9. Напишите программу (файл arithmetic.py), которая предлагала бы пользователю решить пример $4 * 100 - 54$. Потом выводила бы на экран правильный ответ и ответ пользователя. Подумайте, нужно ли здесь преобразовывать строку в число.



```
1 answer = int(input("4 * 100 - 54 = "))
2 print(f"Your solution = {answer}\nanswer = {4 * 100 - 54}")
3
```

Рисунок 7 – Код программы



```
4 * 100 - 54 = 1
Your solution = 1
answer = 346
```

Рисунок 8 – Результат ее выполнения в IDE PyCharm

```
[(base) svetik@MacBook-Air-Svetik LR_2.1 % python arithmetic.py  
solve: 4*100 - 54 = 324  
Your solution=324  
answer=346  
(base) svetik@MacBook-Air-Svetik LR_2.1 %
```

Рисунок 9 – Результат выполнение программы в командной строке

10. Запросите у пользователя четыре числа (файл numbers.py). Отдельно сложите первые два и отдельно вторые два. Разделите первую сумму на вторую. Выведите результат на экран так, чтобы ответ содержал две цифры после запятой.

```
1 number1 = float(input("Enter 1 number: "))  
2 number2 = float(input("Enter 2 number: "))  
3 number3 = float(input("Enter 3 number: "))  
4 number4 = float(input("Enter 4 number: "))  
5 answer = (number1 + number2) / (number3 + number4)  
6 print("Answer = %.2f" % answer)  
7
```

Рисунок 10 – Код программы

```
/Users/svetik/.conda/envs/LR_2.1/bin/python  
Enter 1 number: 1  
Enter 2 number: 2  
Enter 3 number: 3  
Enter 4 number: 4  
Answer= 0.43  
  
Process finished with exit code 0
```

Рисунок 11 – Результат ее выполнения в IDE PyCharm

```

answer=0.43
(base) svetik@MacBook-Air-Svetik LR_2.1 % python numbers.py
Enter 1 number: 1
Enter 2 number: 2
Enter 3 number: 3
Enter 4 number: 4
Answer= 0.43
(base) svetik@MacBook-Air-Svetik LR_2.1 %

```

Рисунок 12 – Результат выполнения программы в командной строке

11. Напишите программу (файл individual.py) для решения индивидуального задания. Вариант индивидуального задания уточните у преподавателя.

4. Даны два числа. Найти их сумму, разность, произведение, а также частное от деления первого числа на второе.

Рисунок 13 – Индивидуальное задание

```

1  number1 = float(input("Enter 1 number: "))
2  number2 = float(input("Enter 2 number: "))
3  print(f"summa = {number1 + number2}\n"
4      f"difference = {number1 - number2}\n"
5      f"multiplication = {number1 * number2}\n"
6      f"division = {number1 / number2}")
7

```

Рисунок 14 – Код программы

```

C:\Users\dimu7\AppData\Local
Enter 1 number: 10
Enter 2 number: 1
summa = 11.0
difference = 9.0
multiplication = 10.0
division = 10.0

```

Рисунок 15 – Результат ее выполнения в IDE PyCharm

12. Выполните коммит файлов user.py, arithmetic.py, numbers.py и individual.py в репозиторий git в ветку для разработки.

```
(base) svetik@MacBook-Air-Svetik LR_2.1 % git add .
(base) svetik@MacBook-Air-Svetik LR_2.1 % fit commit -m "Tasks"
zsh: command not found: fit
(base) svetik@MacBook-Air-Svetik LR_2.1 % git commit -m "Tasks"
[main e7f7b61] Tasks
 10 files changed, 56 insertions(+)
 create mode 100644 .idea/.gitignore
 create mode 100644 .idea/LR_2.1.iml
 create mode 100644 .idea/inspectionProfiles/profiles_settings.xml
 create mode 100644 .idea/misc.xml
 create mode 100644 .idea/modules.xml
 create mode 100644 .idea/vcs.xml
 create mode 100644 arithmetic.py
 create mode 100644 individual.py
 create mode 100644 numbers.py
 create mode 100644 user.py
(base) svetik@MacBook-Air-Svetik LR_2.1 % git push
Enumerating objects: 15, done.
Counting objects: 100% (15/15), done.
Delta compression using up to 8 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (14/14), 2.07 KiB | 2.07 MiB/s, done.
Total 14 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/kucherenkosveta/LR_2.1.git
 28e6018..e7f7b61  main -> main
(base) svetik@MacBook-Air-Svetik LR_2.1 %
```

Рисунок 16 – Коммит заданий

13. Выполните слияние ветки для разработки с веткой master.

```
(base) svetik@MacBook-Air-Svetik LR_2.1 % git merge work
Updating e7f7b61..6d0aae8
Fast-forward
 .DS_Store      | Bin 0 -> 6148 bytes
 arithmetic.py | 2 +--
 2 files changed, 1 insertion(+), 1 deletion(-)
 create mode 100644 .DS_Store
(base) svetik@MacBook-Air-Svetik LR_2.1 %
```

Рисунок 17 – Слияние веток

14. Отправьте сделанные изменения на сервер GitHub.

Задания повышенной сложности:

1. Даны цифры двух целых чисел: двузначного a_2a_1 и однозначного b , где a_1 – число единиц, a_2 – число десятков. Получить цифры числа, равного сумме заданных чисел (известно, что это число двузначное). Слагаемое – двузначное число и число-результат не определять; условный оператор не использовать.

Рисунок 18 - Задание

```
1 a1 = int(input('enter a1: ')) * 10
2 a2 = int(input('enter a2: '))
3 b1 = int(input('enter b1: '))
4
5 result = a1 + a2 + b1
6 print(f'the first figure {str(result)[0]}\n'
7       f'the second figure {str(result)[1]}')
8
```

Рисунок 19 – Код программы

```
C:\Users\dim07\AppData\Lo
enter a1: 1
enter a2: 3
enter b1: 5
the first figure 1
the second figure 8
```

Рисунок 20 – Результат выполнение программы

Вопросы для защиты работы

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

1. Для установки интерпретатора Python первое, что нужно сделать – это скачать дистрибутив. Загрузить его можно с официального сайта, перейдя по ссылке <https://www.python.org/downloads/>.

2. Запустить скачанный установочный файл.

3. Выбрать способ установки.
4. Отметить необходимые опции установки (доступно при выборе Customize installation).
5. Выберите место установки (доступно при выборе Customize installation).

При установке для Linux, в случае ошибки необходимо либо собрать Python из исходников либо взять из репозитория. Для установки из репозитория в Ubuntu воспользуйтесь командой «`sudo apt-get install python3`»

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

Anaconda является дистрибутивом языков программирования таких как Python и R для науки о данных и машинного обучения, а Python — это язык программирования высокого уровня общего назначения.

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. Как осуществить проверку работоспособности пакета Anaconda?

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`», в результате чего отобразится процесс загрузки веб-среды Jupyter Notebook, после чего 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 выберите Python. В результате

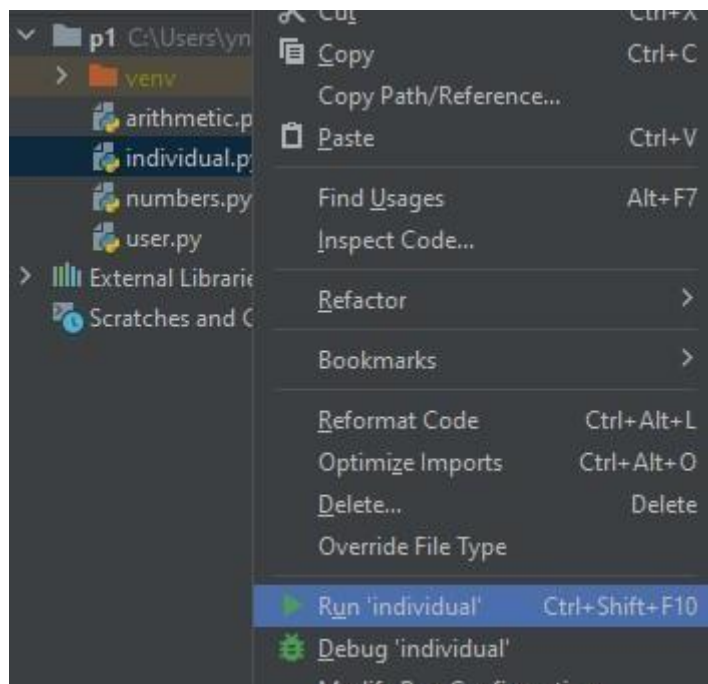
будет создана новая страница в браузере с ноутбуком. Введите в первой ячейке команду «`print("Hello, World!")`» и нажмите Alt+Enter на клавиатуре. Ниже ячейки должна появиться соответствующая надпись.

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

При создании нового проекта в PyCharm есть возможность выбрать путь до проекта и интерпретатор.

5. Как осуществить запуск программы с помощью IDE PyCharm?

Правой кнопкой в любом месте или по файлу слева и выбрать из появившегося меню пункт «Run»



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

Интерактивный режим позволяет вводить команды, которые сразу же будут обрабатываться, это можно использовать в качестве калькулятора.

Пакетный режим позволяет запустить файл с исходным кодом.

7. Почему язык программирования Python называется языком динамической типизации?

Это означает, что одна и та же переменная в разное время может ссылаться на данные разного типа.

8. Какие существуют основные типы в языке программирования Python?

1. None (неопределенное значение переменной)
2. Логические переменные (Boolean Type)
3. Числа (Numeric Type)
 - 3.1. int – целое число
 - 3.2. float – число с плавающей точкой
 - 3.3. complex – комплексное число
4. Списки (Sequence Type)
 - 4.1. list – список
 - 4.2. tuple – кортеж
 - 4.3. range – диапазон
5. Строки (Text Sequence Type)
6. Бинарные списки (Binary Sequence Types)
 - 6.1. bytes – байты
 - 6.2. bytearray – массивы байт
 - 6.3. memoryview – специальные объекты для доступа к внутренним данным объекта через protocol buffer
7. Множества (Set Types)

- 7.1. set – множество
- 7.2. frozenset – неизменяемое множество
- 8. Словари (Mapping Types)
- 8.1. dict – словарь

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

При создании переменной, вначале создается объект, который имеет уникальный идентификатор, тип и значение, после этого переменная может ссылаться на созданный объект.

Создание переменных и объектов в Python происходит с помощью оператора присваивания («=»). Записывается имя переменной, оператор и значение, например, «a = 5». Множественное присваивание (позиционное присваивание) в Python реализуется следующим образом: «a, b, c = 5, 3, 1»

10. Как получить список ключевых слов в Python?

Нужно подключить модуль keyword и воспользоваться командой keyword.kwlist.

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

id() – возвращает уникальный идентификатор объекта в программе.
type() – возвращает тип переменной.

12. Что такое изменяемые и неизменяемые типы в Python.

Изменяемый объект можно изменить после его создания, а неизменяемый – нет. Во втором случае фактически мы не изменяем значение переменной, а создаем другой объект с тем же именем и присваиваем ему другое значение. Мы связываем имя переменной с новым значением. Теперь, при ее вызове, он будет выводить новое значение и ссылаться на новое местоположение.

13. Чем отличаются операции деления и целочисленного деления?

Во втором случае не учитывается остаток от деления.

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.

15. Каково назначение и основные функции библиотеки (модуля) `math`?

По аналогии с модулем `math` изучите самостоятельно назначение и основные функции модуля `cmath`.

Библиотека `math` содержит большое количество часто используемых математических функций, такие как округление, логарифмы, факториал, модуль числа, экспонента (e^x), степень, квадратный корень, синус/косинус, числа π и e и т.д.

Модуль `cmath` – предоставляет функции для работы с комплексными числами. Из отличных функций можно выделить преобразование к полярным координатам и преобразование из полярных координат.

`cmath.isfinite(x)` - True, если действительная и мнимая части конечны.

`cmath.isinf(x)` - True, если либо действительная, либо мнимая часть бесконечна. `cmath.isnan(x)` - True, если либо действительная, либо мнимая часть NaN.

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

`Sep()` устанавливает отличный от пробела разделитель строк.

`End()` указывает, что делать, после вывода строки (по умолчанию стоит переход на новую строку)

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()` позволяет форматировать выводимые строки.

Рассмотрим пример:

```
print("This is a {0}. It's {1}.".format("ball", "red"))
```

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.

Форматирование также может осуществляться в старом (Си-) стиле. Он схож с тем, как происходит вывод на экран в языке C. Пример:

```
print("It's %s, %d. Level: %f" % (pupil, old, grade))
```

Здесь вместо трех комбинаций символов %s, %d, %f подставляются значения переменных pupil, old, grade. Буквы s, d, f обозначают типы данных – строку, целое число, вещественное число.

F-Строки являются упрощенной версией метода format(). F-strings являются строковыми литералами с «f» в начале и фигурные скобки, содержащие выражения, которые в дальнейшем будут заменены своими значениями. Пример:

```
print(f"Hello, {name}. You are {2*8}.")    //name – переменная
```

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

С помощью функции input() можно получить вводимые с консоли данные. Однако они будут принадлежать к строковому типу, чтоб получить число нужно использовать функции преобразования типов:

```
c = float(input("Enter temperature (Celsius) : "))
```

Выводы: в ходе выполнения работы исследование процесса установки и базовых возможностей языка Python, получен опыт работы с IDE PyCharm, Anaconda, консоль.