

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
дисциплины «Программирование на Python»

Выполнил:
Магдаев Даламбек Магомедович
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., 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.

Порядок выполнения работы:

1. Создал репозиторий Lab1.4, настроил .gitignore:

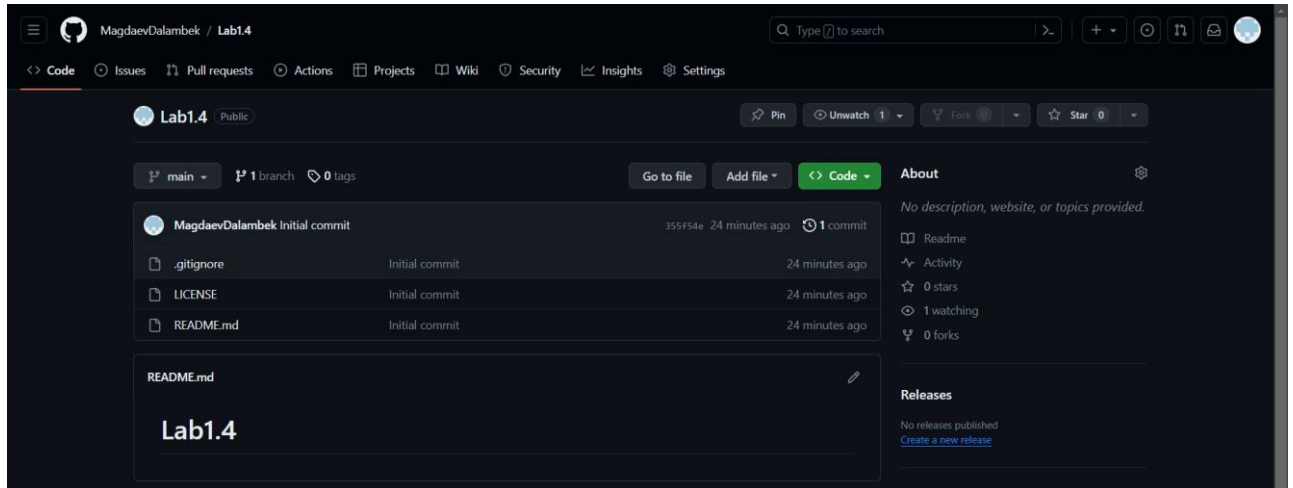


Рисунок 1. Новый репозиторий Lab1.4

2. Клонировал репозиторий и создал ветку developer:

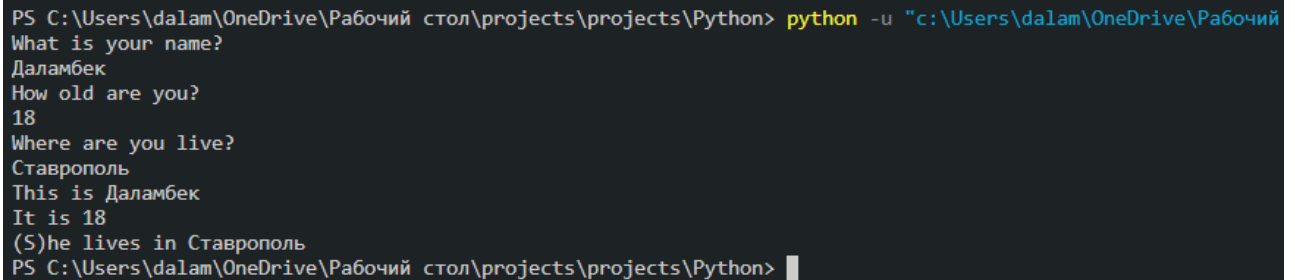
```
PS C:\Users\dalam> git clone https://github.com/MagdaevDalambek/Lab1.4.git
Cloning into 'Lab1.4'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
PS C:\Users\dalam> cd Lab1.4
PS C:\Users\dalam\Lab1.4> git checkout -b developer
Switched to a new branch 'developer'
PS C:\Users\dalam\Lab1.4> |
```

Рисунок 2. Создание ветки developer

3. Выполнил задание 8: 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 live in `место_жительства`".

Код программы:

```
name = input("What is your name?\n")
age = input("How old are you?\n")
location = input("Where are you live?\n")print("This is " + name)
print("It is " + age)
print("(S)he lives in " + location)
```



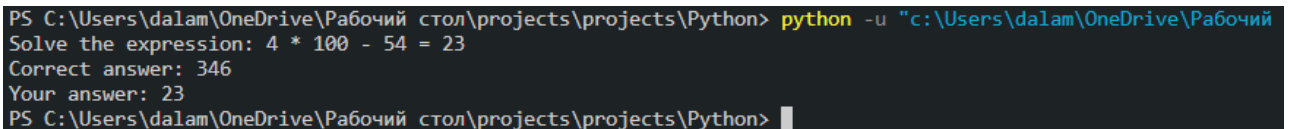
```
PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python> python -u "c:\Users\dalam\OneDrive\Рабочий
What is your name?
Даламбек
How old are you?
18
Where are you live?
Ставрополь
This is Даламбек
It is 18
(S)he lives in Ставрополь
PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python>
```

Рисунок 3. Вывод результата программы

4. Выполнил задание 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.

Код программы:

```
expression = "4 * 100 - 54"
expected_result = eval(expression)
user_answer = input("Solve the expression: 4 * 100 - 54 = ")
print("Correct answer:", expected_result)
print("Your answer:", user_answer)
```



```
PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python> python -u "c:\Users\dalam\OneDrive\Рабочий
Solve the expression: 4 * 100 - 54 = 23
Correct answer: 346
Your answer: 23
PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python>
```

Рисунок 4. Вывод результата программы

5. Выполнил задание 9: 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.

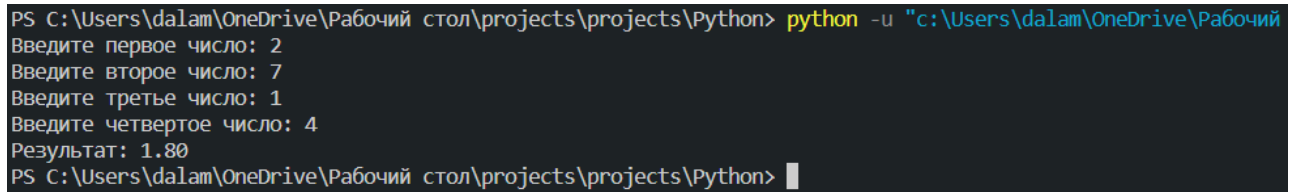
Код программы:

```
num1 = float(input("Введите первое число: "))
```

```

num2 = float(input("Введите второе число: "))
num3 = float(input("Введите третье число: "))
num4 = float(input("Введите четвертое число: "))
sum1 = num1 + num2
sum2 = num3 + num4
result = sum1 / sum2
print("Результат: {:.2f}".format(result))

```



```

PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python> python -u "c:\Users\dalam\OneDrive\Рабочий
Введите первое число: 2
Введите второе число: 7
Введите третье число: 1
Введите четвертое число: 4
Результат: 1.80
PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python>

```

Рисунок 5. Вывод результата программы

6. Выполнил индивидуальное задание варианта 9: Треугольник задан координатами своих вершин. Найти периметр и площадь треугольника.

Код программы:

```

import math

# ввод координат вершин треугольника
print('Enter the coordinates of the vertex A: ')
x1, y1 = map(float, input().split())
print('Enter the coordinates of the vertex B: ')
x2, y2 = map(float, input().split())
print('Enter the coordinates of the vertex C: ')
x3, y3 = map(float, input().split())

# вычисление длин сторон треугольника
AB = math.sqrt((x2 - x1)**2 + (y2 - y1)**2)
AC = math.sqrt((x3 - x1)**2 + (y3 - y1)**2)
BC = math.sqrt((x3 - x2)**2 + (y3 - y2)**2)
p = (AB + AC + BC) # Периметр треугольника ABC
print("")
print('Perimeter = {:.2f}'.format(p)) # Выводим периметр
p = p / 2 # полупериметр

```

Площадь треугольника ABC

$S = \text{math.sqrt}(p * (p - AB) * (p - AC) * (p - BC))$

`print('Square = {:.2f}'.format(S))`

```
Enter the coordinates of the vertex A:
2 0
Enter the coordinates of the vertex B:
3 7
Enter the coordinates of the vertex C:
1 4

Perimeter = 14.80
Square = 5.50
PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python>
```

Рисунок 6. Вывод результата программы

7. Выполнил первое задание повышенной сложности: 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.

Код программы:

`number = int(input("Введите двузначное число a2a1: "))`

`b = int(input("Введите однозначное число b: "))`

`a2 = number // 10 a1 = number % 10`

`print("a2 =", a2, "a1 =", a1, "b =", b)`

`result_a1 = (b + a1) % 10`

`result_a2 = a2 + (b + a1) // 10`

`print("Результирующий a2:", result_a2)`

`print("Результирующий a1:", result_a1)`

```
PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python> python -u "c:\Users\dalam\OneDrive
Введите двузначное число a2a1: 88
Введите однозначное число b: 7
a2 = 8 a1 = 8 b = 7
Результирующий a2: 9
Результирующий a1: 5
PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python>
```

Рисунок 7. Вывод терминала

8. Выполнил коммит файлов в ветке developer:

```
PS C:\Users\dalam\Lab1.4> git commit -m "add programs"
[developer d62441c] add programs
5 files changed, 92 insertions(+)
create mode 100644 arithmetic.py
create mode 100644 hard_1.py
create mode 100644 individual.py
create mode 100644 numbers.py
create mode 100644 user.py
```

Рисунок 8. Коммит файлов программ

9. Слил ветку developer в main:

```
PS C:\Users\dalam\Lab1.4> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\dalam\Lab1.4> git rebase developer
Successfully rebased and updated refs/heads/main.
```

Рисунок 9. Результат выполнения git rebase developer из main

10. Отправил все изменения ветки main на удаленный репозиторий:

```
PS C:\Users\dalam\Lab1.4> git push
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 12 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 2.42 KiB | 2.42 MiB/s, done.
Total 7 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/MagdaevDalambek/Lab1.4.git
355f54e..d62441c main -> main
```

Рисунок 10. Отправка изменений

Ответы на контрольные вопросы:

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

Порядок установки Python в Windows:

- запустите скачанный установочный файл;
- выберете способ установки;
- отметьте необходимые опций установки (доступно при выборе Customize installation);
- выберете место установки (доступно при выборе Customize installation);
- после успешной установки вас ждет соответствующее сообщение.

Порядок установки Python в Linux: чаще всего интерпретатор Python уже входит в состав дистрибутива.

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

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. 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.

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

Указать путь до интерпретатора в настройках.

5. Как осуществить запуск программы с помощью IDE PyCharm?

Нажать на соответствующий значек в виде зеленого треугольника либо в углу окна, либо в контекстном меню.

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.

Пакетный: сначала записывается вся программа, потом она выполняется полностью.

7. Почему язык программирования Python называется языком динамической типизации?

В языке программирования Python тип переменной определяется непосредственно при выполнении программы.

8. Какие существуют основные типы в языке программирования Python?

К основным встроенным типам относятся:

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) 1. str

- 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?

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 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. 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.

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()`?

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. 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`). 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 спомощью функции `id()`.

Если тип данных изменяемый, то можно менять значение объекта.

Например, создадим список `[1, 2]`, а потом заменим второй элемент на 3.

13. Чем отличаются операции деления и целочисленного деления?

При целочисленном делении дробная часть отбрасывается, при обычном – нет.

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 и мнимую части (`x.real` и `x.imag`).

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

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

Основные функции:

`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 π .

`math.e` - число e .

Теперь рассмотрим модуль `cmath`. Он предоставляет функции для

выполнения математических операций с комплексными числами в Python.

Основные функции модуля `cmath` включают:

`cmath.sqrt(x)`: Возвращает квадратный корень из комплексного числа x .

`cmath.exp(x)`: Возвращает значение экспоненты e в степени x . `cmath.log(x)`:

Возвращает натуральный логарифм комплексного числа x . `cmath.sin(x)`:

Возвращает синус комплексного числа x .

`cmath.cos(x)`: Возвращает косинус комплексного числа x . `cmath.tan(x)`:

Возвращает тангенс комплексного числа x . `cmath.phase(x)`: Возвращает фазу комплексного числа x .

`cmath.polar(x)`: Возвращает полярные координаты комплексного числа x в виде (r, ϕ) , где r - модуль числа, а ϕ - аргумент числа.

`cmath.rect(r, phi)`: Возвращает комплексное число в декартовой форме по его полярным координатам (r, ϕ) .

Модуль `cmath` позволяет работать с комплексными числами и выполнять на них различные математические операции.

16. Каково назначение именованных 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. По умолчанию происходит переход на новую строку. Однако это действие можно отменить, указав любой другой символ или 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.

Форматирование может выполняться в так называемом старом стиле или с помощью строкового метода `format`. Старый стиль также называют Си-

стилем, так как он схож с тем, как происходит вывод на экран в языке 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. Сами значения записываются в скобках после знака процента(%)

Метод format()

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.

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

Указать перед input() тип данных: int(input()), а для вещественной переменной float(input()).

Вывод: в результате выполнения работы были исследованы процесс установки и базовые возможности языка Python версии 3.x.