

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ
ФЕДЕРАЦИИ**
**Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Кафедра инфокоммуникаций

Институт цифрового развития

ОТЧЁТ

по лабораторной работе №3

Дисциплина: «Основы кроссплатформенного программирования»

Тема: «Условные операторы и циклы в языке Python»

Выполнил студент группы

ИВТ-б-о-21-1

Артемов Андрей Витальевич

« » _____ 20__ г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил доцент

Кафедры инфокоммуникаций, старший
преподаватель

Воронкин Р.А.

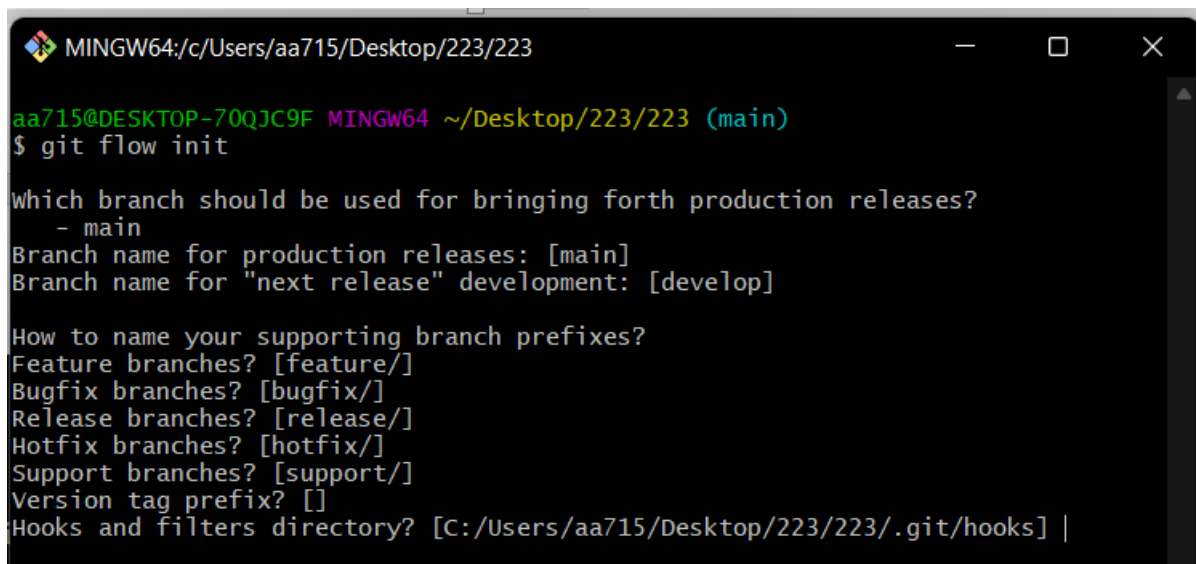
(подпись)

Ставрополь 2022

Ход работы:

```
1  # Byte-compiled / optimized / DLL files
2  __pycache__/
3  *.py[cod]
4  *$py.class
5
6  # C extensions
7  *.so
8
9  # Distribution / packaging
10 .Python
11 build/
12 develop-eggs/
13 dist/
14 downloads/
15 eggs/
16 .eggs/
17 lib/
18 lib64/
19 parts/
20 sdist/
21 var/
22 wheels/
23 pip-wheel-metadata/
24 share/python-wheels/
25 *.egg-info/
26 .installed.cfg
27 *.egg
28 MANIFEST
29
30 # PyInstaller
31 # Usually these files are written by a python script from a template
32 # before PyInstaller builds the exe, so as to inject date/other infos into it.
33 *.manifest
34 *.spec
35
36 # Installer logs
```

Рисунок 1. Добавление правил в .gitignore



```
MINGW64:/c/Users/aa715/Desktop/223/223
aa715@DESKTOP-70QJC9F MINGW64 ~/Desktop/223/223 (main)
$ git flow init

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

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [C:/Users/aa715/Desktop/223/223/.git/hooks] |
```

Рисунок 2. Организация репозитория согласно модели ветвления git-flow

```
1 print("What is your name?")
2 name = str(input())
3 print("How old are you?")
4 age = int(input())
5 print("Where are you from?")
6 place = str(input())
7 print("This is", name)
8 print("(S)he", age, "years old")
9 print("(S)he is", age, "years old")
10 print("(S)he live in", place)
```

main x

↑ 18
↓ Where are you from?
↶ Stavropol
↷ This is Andrey
↵ (S)he 18 years old
↵ (S)he is 18 years old
↵ (S)he live in Stavropol

Process finished with exit code 0

Рисунок 3. Задача 1

The image shows a PyCharm IDE window. The top pane displays a Python script named `main.py` with the following code:

```
1 print("4 * 100 - 54 =")
2 print("Please,write an answer:")
3 answer = int(input())
4 print("Correct answer is",4 * 100 -54)
5 print("Your answer:",answer)
```

The bottom pane shows the output of the script execution. The path to the script is `C:\Users\aa715\PycharmProjects\pythonProject\venv\Scripts\python.exe`. The output is as follows:

```
4 * 100 - 54 =
Please,write an answer:
256
Correct answer is 346
Your answer: 256

Process finished with exit code 0
```

Рисунок 4. Задача 2

```
1 print("Write 4 numbers:")
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 s = sum1/sum2
9 d = round(s,2)
10 print(d)
```

3333 x

D:\programmi\python.exe C:/Users/aa715/Downloads/3333.py

Write 4 numbers:

9

8

9

8

1.0

Process finished with exit code 0

Рисунок 5. Задача 3

The image shows a PyCharm IDE window with a Python script named 'indiv 1.py' and its execution output in the console.

Code in indiv 1.py:

```
1 a=int(input("Введите a:"))
2 b=int(input("Введите b:"))
3 print(f"Среднее арифметическое: {(a+b)/2}\nСреднее геометрическое: {(a+b)**0.5}")
```

Console Output:

```
D:\programmi\python.exe "C:/Users/aa715/AppData/Roaming/JetBrains/PyCharmCE2022.1/sc
Введите a:8
Введите b:2
Среднее арифметическое: 5.0
Среднее геометрическое: 3.1622776601683795
Process finished with exit code 0
```

Рисунок 6. Индивидуальная задача

```
MINGW64:/c/Users/aa715/a21
aa715@DESKTOP-70QJC9F MINGW64 ~/a21 (main)
$ git merge develop
Updating 972058b..7750924
Fast-forward
"\320\272\320\276\320\264\1main.py" | 10 ++++++++
"\320\272\320\276\320\264\2.py" | 5 +++++
"\320\272\320\276\320\264\3main.py" | 10 ++++++++
"\320\272\320\276\320\264\indiv 1.py" | 3 +++
.../1.py" | 8 ++++++++
.../ind.py" | 8 -----
6 files changed, 36 insertions(+), 8 deletions(-)
create mode 100644 "\320\272\320\276\320\264\1main.py"
create mode 100644 "\320\272\320\276\320\264\2.py"
create mode 100644 "\320\272\320\276\320\264\3main.py"
create mode 100644 "\320\272\320\276\320\264\indiv 1.py"
create mode 100644 "\321\203\321\201\320\273\320\276\320\266\320\275\320\265\320\275\320\275\321\213\320\265 \320\267\320\260\320\264\320\260\320\275\320\270\321\217\1.py"
delete mode 100644 "\321\203\321\201\320\273\320\276\320\266\320\275\320\265\320\275\320\275\321\213\320\265 \320\267\320\260\320\264\320\260\320\275\320\270\321\217\ind.py"
aa715@DESKTOP-70QJC9F MINGW64 ~/a21 (main)
$ |
```

Рисунок 7. Слияние веток

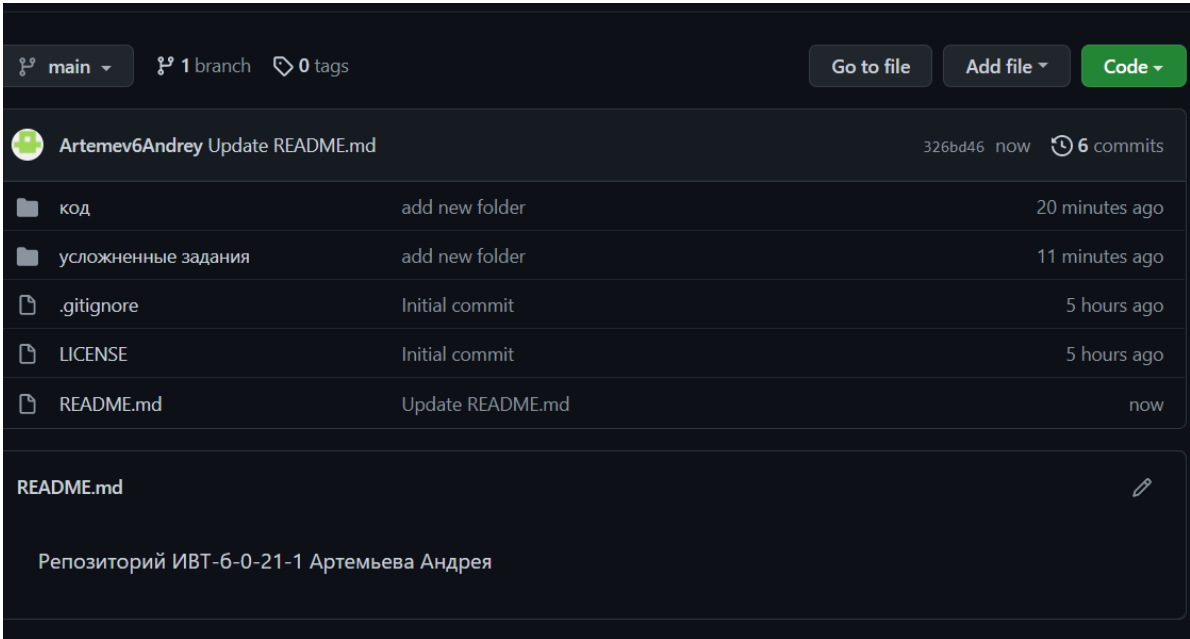


Рисунок 8. Изменения в удаленном репозитории


```
a2 = int(input('add a2 '))
a1 = int(input('add a1 '))
b2 = int(input('add b2 '))
b1 = int(input('add b1 '))

c2 = a2 + b2 + (a1 + b1) // 10
c1 = (a1 + b1) % 10

print(c2, c1)
```

2 x

```
D:\programmi\python.exe C:/Users/aa715/Downloads/2.py
add a2 10
add a1 5
add b2 2
add b1 4
12 9

Process finished with exit code 0
```

Рисунок 9. Задание повышенной сложности

Контрольные вопросы:

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

Чаще всего интерпретатор Python уже входит в состав дистрибутива.

Windows: Осн. 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 отобразится 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 в выборе 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.

Пакетный. 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 ЯП Python проверка типа происходит во время выполнения, а не компиляции.

8. Какие существуют основные типы в языке программирования Python?

Типы в ЯП 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 комплексносопряженного числа 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`. Осн. 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 π .

`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: исследовали процесс установки и базовые возможности языка Python версии 3.