

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 №3**

Выполнила:
Кузнецова Алена Валерьевна
1 курс, группа ИВТ-б-о-21-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной техники и
автоматизированных систем», очная
форма обучения

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2022 г.

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

Ход работы:

1. Создали общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python.

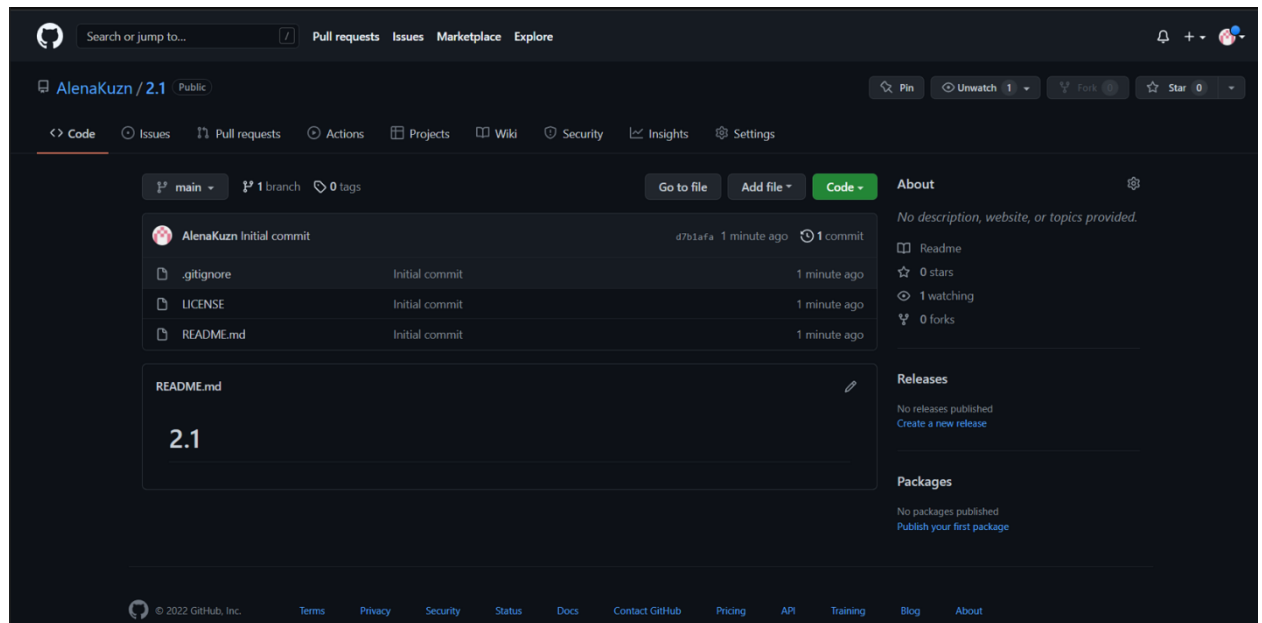


Рисунок 1 – Созданный репозиторий

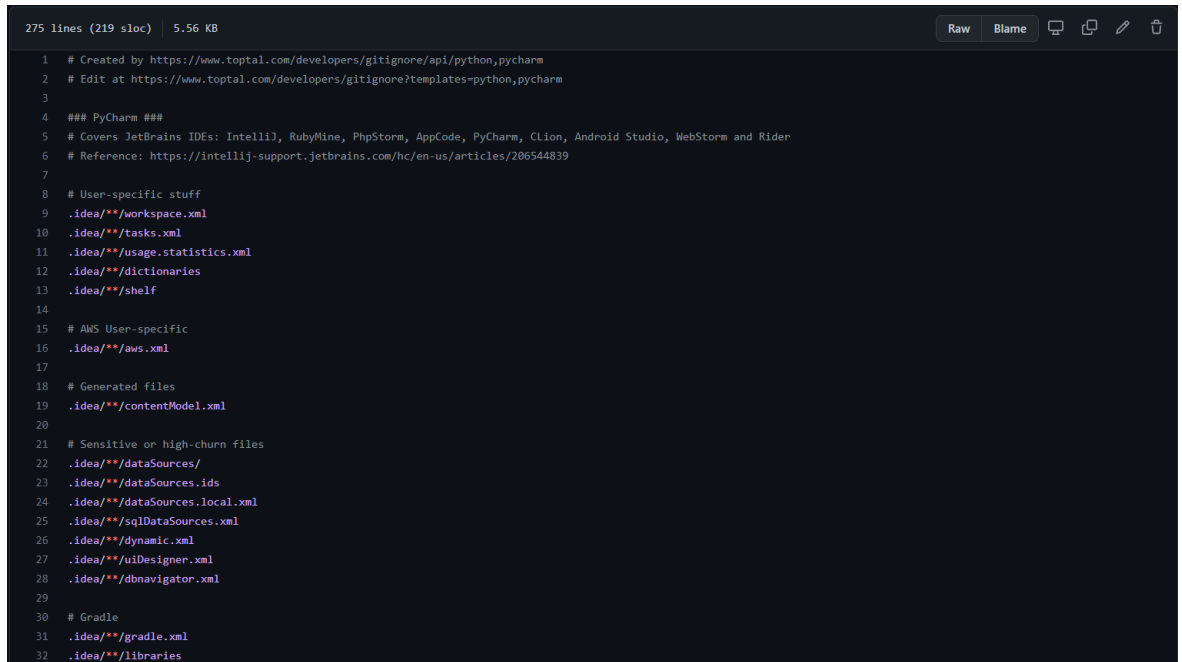
2. Выполнили клонирование созданного репозитория.

```
C:\Users\akuzn>cd /d C:\Users\akuzn

C:\Users\akuzn>git clone https://github.com/AlenaKuzn/2.1.git
Cloning into '2.1'...
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.
```

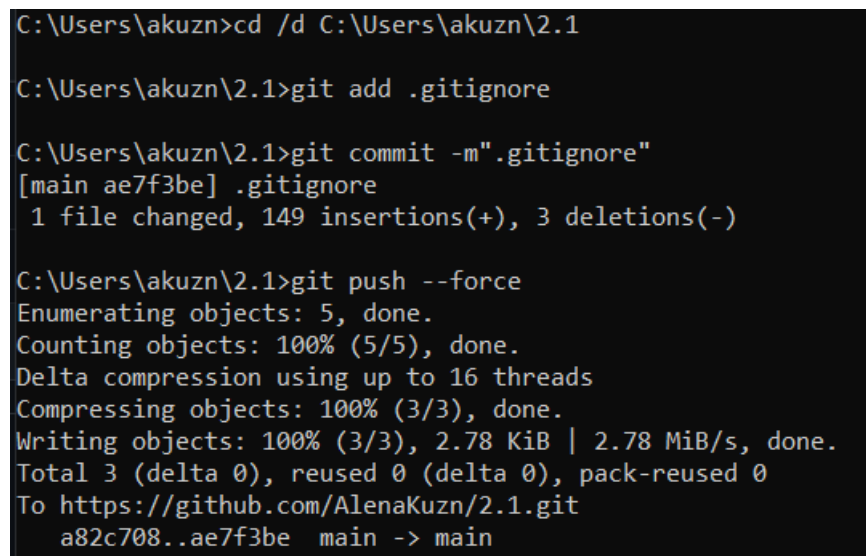
Рисунок 2 – Клонирование репозитория

3. Дополните файл .gitignore необходимыми правилами для работы с IDE PyCharm.



```
275 lines (219 sloc) | 5.56 KB
1 # Created by https://www.toptal.com/developers/gitignore/api/python,pycharm
2 # Edit at https://www.toptal.com/developers/gitignore?templates-python,pycharm
3
4 ### PyCharm ###
5 # Covers JetBrains IDEs: IntelliJ, RubyMine, PhpStorm, AppCode, PyCharm, CLion, Android Studio, WebStorm and Rider
6 # Reference: https://intellij-support.jetbrains.com/hc/en-us/articles/206544839
7
8 # User-specific stuff
9 .idea/**/workspace.xml
10 .idea/**/tasks.xml
11 .idea/**/usage.statistics.xml
12 .idea/**/dictionaries
13 .idea/**/shelf
14
15 # AWS User-specific
16 .idea/**/aws.xml
17
18 # Generated files
19 .idea/**/contentModel.xml
20
21 # Sensitive or high-churn files
22 .idea/**/dataSources/
23 .idea/**/dataSources.ids
24 .idea/**/dataSources.local.xml
25 .idea/**/sqlDataSources.xml
26 .idea/**/dynamic.xml
27 .idea/**/uiDesigner.xml
28 .idea/**/dbnavigator.xml
29
30 # Gradle
31 .idea/**/gradle.xml
32 .idea/**/libraries
```

Рисунок 3 – Дополнили файл



```
C:\Users\akuzn>cd /d C:\Users\akuzn\2.1
C:\Users\akuzn\2.1>git add .gitignore
C:\Users\akuzn\2.1>git commit -m ".gitignore"
[main ae7f3be] .gitignore
1 file changed, 149 insertions(+), 3 deletions(-)
C:\Users\akuzn\2.1>git push --force
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 2.78 KiB | 2.78 MiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/AlenaKuzn/2.1.git
a82c708..ae7f3be main -> main
```

Рисунок 4 – Отправили на удаленный репозиторий

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

```
C:\Users\akuzn\2.1>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/akuzn/2.1/.git/hooks]
```

Рисунок 5 – Организация репозитория в соответствии с моделью ветвления git-flow

5. Написали программу (файл user.py), которая запрашивает у пользователя:

- его имя (например, "What is your name?")
- возраст ("How old are you?")
- место жительства ("Where are you live?")

```
name = input("What is your name? ")
year = input("How old are you? ")
place = input("Where are you live? ")
print("This is", name)
print("It is", year)
print("Live in", place)
```

Рисунок 6 – Код программы

```
C:\Users\akuzn\PycharmProjects\user\venv\Scripts\python.exe C:/Users/akuzn/PycharmProjects/user/main.py
What is your name? Alena
How old are you? 18
Where are you live? Stavropol
This is Alena
It is 18
Live in Stavropol

Process finished with exit code 0
```

Рисунок 7 – Результат работы программы

6. Написали программу (файл arithmetic.py), которая предлагает пользователю решить пример $4 * 100 - 54$. Потом выводит на экран правильный ответ и ответ пользователя.

```
your_answer = input("solve the example 100 * 4 - 54 = ")
print("Your answer", your_answer)
print("correct answer", 100 * 4 - 54)
```

Рисунок 8 – Код программы

```
C:\Users\akuzn\PycharmProjects\arithmetic
solve the example 100 * 4 - 54 = 777
Your answer 777
correct answer 346

Process finished with exit code 0
```

Рисунок 9 – Результат работы программы

7. Запросили у пользователя четыре числа (файл numbers.py). Отдельно сложим первые два и отдельно вторые два. Разделим первую сумму на вторую. Выводим результат на экран так, чтобы ответ содержал две цифры после запятой.

```
print("Enter four numbers ")
n1 = int(input())
n2 = int(input())
n3 = int(input())
n4 = int(input())
summa1 = n1 + n2
summa2 = n3 + n4
result = summa1 / summa2
print("result", format(result, ".2"))
```

Рисунок 10 – Код программы

```
Enter four numbers
21
3
11
22
result 0.73
```

Рисунок 11 – Результат работы программы

8. Написали программу (файл individual.py) для решения индивидуального задания.

Задание:

Напишите программу, в которой вычисляется сумма, разность, произведение, частное и среднее арифметическое двух целых чисел, введенных с клавиатуры.

```
print("Enter two numbers")
n1 = int(input())
n2 = int(input())
print("Summa ", n1 + n2)
print("Raznost ", n1 - n2)
print("Proizved ", n1 * n2)
print("Delen ", n1 / n2)
print("Srednee arf ", (n1 + n2) / 2)
```

Рисунок 12 – Код программы

```
Enter two numbers
3
10
Summa 13
Raznost -7
Proizved 30
Delen 0.3
Srednee arf 6.5
```

Рисунок 13 – Результат работы программы

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

```
C:\Users\akuzn\2.1>git commit -m"numbers.py"
[develop 9a69314] numbers.py
1 file changed, 9 insertions(+)
create mode 100644 numbers.py

C:\Users\akuzn\2.1>git add individual.py

C:\Users\akuzn\2.1>git commit -m"individual.py"
[develop e4cef0e] individual.py
1 file changed, 8 insertions(+)
create mode 100644 individual.py

C:\Users\akuzn\2.1>git add "hard.py"

C:\Users\akuzn\2.1>git commit -m"hard.py"
[develop 19563f1] hard.py
1 file changed, 6 insertions(+)
create mode 100644 hard.py

C:\Users\akuzn\2.1>git add arithmetic.py

C:\Users\akuzn\2.1>git commit -m"arithmetic.py"
[develop 5610e81] arithmetic.py
1 file changed, 3 insertions(+)
create mode 100644 arithmetic.py
```

Рисунок 14 – Коммит файлов

10. Выполните слияние ветки для разработки с веткой master.

```
C:\Users\akuzn\2.1>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

C:\Users\akuzn\2.1>git merge develop
Updating ae7f3be..5610e81
Fast-forward
 arithmetic.py | 3 +++
 hard.py       | 6 ++++++
 individual.py | 8 ++++++++
 numbers.py    | 9 ++++++++
4 files changed, 26 insertions(+)
create mode 100644 arithmetic.py
create mode 100644 hard.py
create mode 100644 individual.py
create mode 100644 numbers.py
```

Рисунок 15 – Слияние ветки для разработки с веткой main

11. Задача 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 16 – Условие задачи

```
a2 = int(input("Enter number of tithes "))
a1 = int(input("Enter the number of units "))
b = int(input("Enter a single digit "))
n = a2 + (a1 + b) // 10
m = (a1 + b) % 10
print("Result", n, m)
```

Рисунок 17 – Код программы

```
Enter number of tithes 5
Enter the number of units 9
Enter a single digit 3
Result 6 2
```

Рисунок 19 – Результат выполнения программы

Контрольные вопросы:

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

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

Windows: Основные 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.

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 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 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 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 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?

```
n = int(input())
```