

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 №4
дисциплины «Программирование на Python»
Вариант 29

Выполнил:
Саенко Андрей Максимович
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», заочная
форма обучения

(подпись)

Руководитель практики:
Воронкин Р.А., канд. технических
наук, 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:

Задание 1. Изучить теоретический материал работы.

Задание 2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.

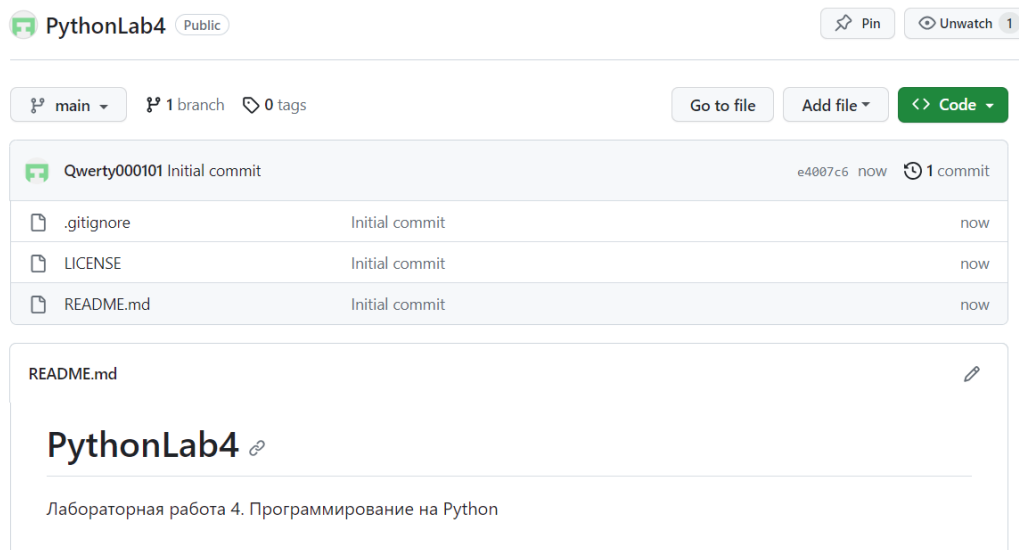


Рисунок 1 – Созданный репозиторий

Задание 3. Выполните клонирование созданного репозитория.

```
C:\Users\HAIER\Desktop\Задания\Программирование на Python>git clone https://github.com/Qwerty000101/PythonLab4.git
Cloning into 'PythonLab4'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
C:\Users\HAIER\Desktop\Задания\Программирование на Python>
```

Рисунок 2 – Клонирование репозитория

Задание 4. Дополните файл .gitignore необходимыми правилами для работы с IDE PyCharm

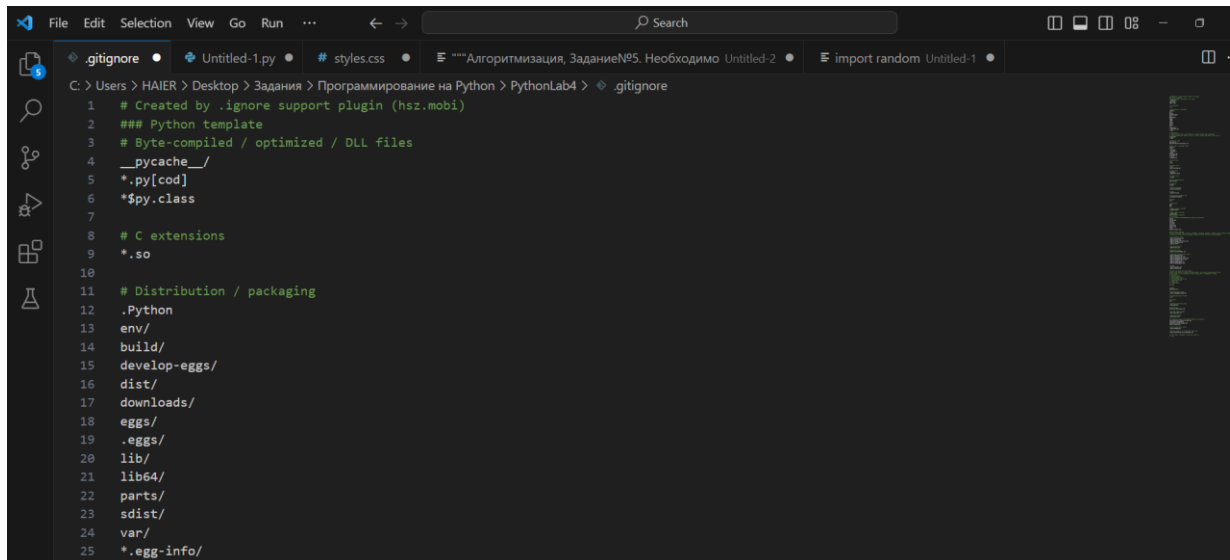


Рисунок 3 – Дополнение файла .gitignore для работы с pycharm

Задание 5. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.

```
C:\Users\HAIER\Desktop\Задания\Программирование на Python\PythonLab4>git branch
* develop
  main

C:\Users\HAIER\Desktop\Задания\Программирование на Python\PythonLab4>git flow status
usage: git flow <subcommand>

Available subcommands are:
  init      Initialize a new git repo with support for the branching model.
  feature   Manage your feature branches.
  bugfix    Manage your bugfix branches.
  release   Manage your release branches.
  hotfix    Manage your hotfix branches.
  support   Manage your support branches.
  version   Shows version information.
  config    Manage your git-flow configuration.
  log       Show log deviating from base branch.

Try 'git flow <subcommand> help' for details.

C:\Users\HAIER\Desktop\Задания\Программирование на Python\PythonLab4>
```

Рисунок 4 – Репозиторий, организованный для работы с git-flow

Задание 6. Создайте проект PyCharm в папке репозитория.

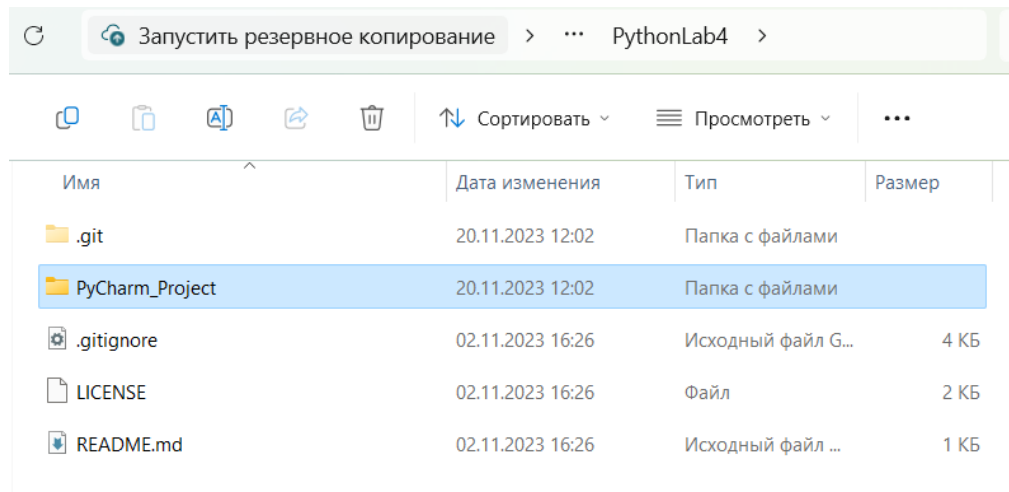


Рисунок 5 – Проект в папке репозитория

7. Решите следующие задачи с помощью языка программирования Python3 и IDE PyCharm:

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`"

Решение:

Код программы:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
name = input("What is your name?\n")
age = int(input("How old are you?\n"))
country = input("Where are you live?\n")
print("This is %s" % (name))
print("It is %d" % (age))
print("(S)he live in %s" % (country))
```

Результат работы программы:

```
What is your name?  
Andrew  
How old are you?  
18  
Where are you live?  
Russia  
This is Andrew  
It is 18  
(S)he live in Russia
```

Рисунок 6 – Результат работы программы

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.

Решение:

Код программы:

```
#!/usr/bin/env python3  
# -*- coding: utf-8 -*-  
answer = int(input("Решите пример: \n4*100-54\n"))  
print("Правильный ответ: %d \nВаш ответ: %d" % ((4*100-54),answer))
```

Результат работы программы:

```
Решите пример:  
4*100-54  
346  
Правильный ответ: 346  
Ваш ответ: 346
```

Рисунок 7 – Результат работы программы

10. Запросите у пользователя четыре числа (файл `numbers.py`). Отдельно сложите первые два и отдельно вторые два. Разделите первую

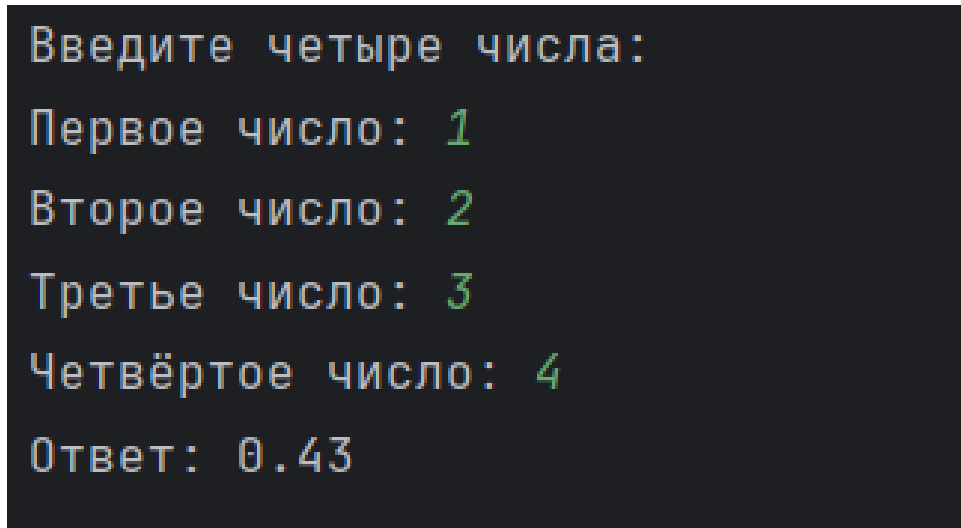
сумму на вторую. Выведите результат на экран так, чтобы ответ содержал две цифры после запятой.

Решение:

Код программы:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
print("Введите четыре числа:")
first_number = float(input("Первое число: "))
second_number = float(input("Второе число: "))
third_number = float(input("Третье число: "))
fourth_number = float(input("Четвёртое число: "))
first_sum = first_number + second_number
second_sum = third_number + fourth_number
answer = first_sum/second_sum
print("Ответ: %.2f" %(answer))
```

Результат работы программы:



```
Введите четыре числа:
Первое число: 1
Второе число: 2
Третье число: 3
Четвёртое число: 4
Ответ: 0.43
```

Рисунок 8 – Результат работы программы

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

Индивидуальное задание:

Даны длины сторон прямоугольного параллелепипеда. Найти его объем и площадь боковой поверхности.

Решение:

Код программы:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
```

```

height = int(input("Введите длину первого ребра параллелепипеда: "))
first_width = int(input("Введите длину второго ребра параллелепипеда: "))
second_width = int(input("Введите длину третьего ребра параллелепипеда: "))
first_edge = height*first_width
second_edge = height*second_width
area = 2*(first_edge + second_edge)
volume = height*first_width*second_width
print("Площадь боковой поверхности параллелепипеда:\nОбъем параллелепипеда равен: %d"%(area,volume))

```

Результат работы программы:

```

Введите длину первого ребра параллелепипеда: 2
Введите длину второго ребра параллелепипеда: 3
Введите длину третьего ребра параллелепипеда: 4
Площадь боковой поверхности параллелепипеда:28
Объем параллелепипеда равен: 24

```

Рисунок 9 – Результат работы программы

Задача повышенной сложности:

Даны целые числа h, m, s , указывающие момент времени: « h часов, m минут, s секунд». Определить угол (в градусах) между положением часовой стрелки в начале суток и в указанный момент времени.

Решение:

Код программы:

```

#!/usr/bin/env python3
# -*- coding: utf-8 -*-
h = int(input("Количество часов: "))
m = int(input("Количество минут: "))
s = int(input("Количество секунд: "))
sum = h*3600 + m*60 + s
print("Угол равен %.3f градуса(ов)"%(abs(sum/120 - 360*round(sum/(360*120)))))

```

Результат работы программы:

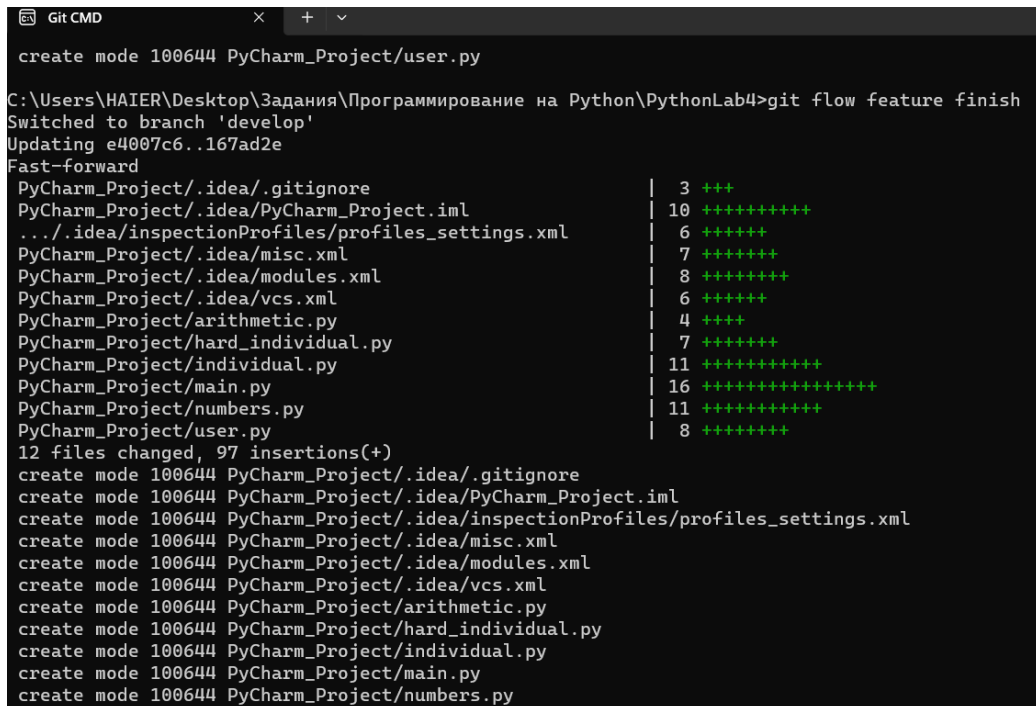
```

Количество часов: 9
Количество минут: 1
Количество секунд: 15
Угол равен 89.375 градуса(ов)

```

Рисунок 10 – Результат работы программы

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



```
Git CMD
create mode 100644 PyCharm_Project/user.py

C:\Users\HAIER\Desktop\Задания\Программирование на Python\PythonLab4>git flow feature finish
Switched to branch 'develop'
Updating e4007c6..167ad2e
Fast-forward
 PyCharm_Project/.idea/.gitignore      | 3 +++
 PyCharm_Project/.idea/PyCharm_Project.iml | 10 ++++++++
 .../.idea/inspectionProfiles/profiles_settings.xml | 6 +++++
 PyCharm_Project/.idea/misc.xml        | 7 ++++++
 PyCharm_Project/.idea/modules.xml     | 8 ++++++
 PyCharm_Project/.idea/vcs.xml         | 6 +++++
 PyCharm_Project/arithmetic.py         | 4 ++++
 PyCharm_Project/hard_individual.py    | 7 ++++++
 PyCharm_Project/individual.py         | 11 ++++++++
 PyCharm_Project/main.py               | 16 ++++++++
 PyCharm_Project/numbers.py           | 11 ++++++++
 PyCharm_Project/user.py               | 8 ++++++
12 files changed, 97 insertions(+)
create mode 100644 PyCharm_Project/.idea/.gitignore
create mode 100644 PyCharm_Project/.idea/PyCharm_Project.iml
create mode 100644 PyCharm_Project/.idea/inspectionProfiles/profiles_settings.xml
create mode 100644 PyCharm_Project/.idea/misc.xml
create mode 100644 PyCharm_Project/.idea/modules.xml
create mode 100644 PyCharm_Project/.idea/vcs.xml
create mode 100644 PyCharm_Project/arithmetic.py
create mode 100644 PyCharm_Project/hard_individual.py
create mode 100644 PyCharm_Project/individual.py
create mode 100644 PyCharm_Project/main.py
create mode 100644 PyCharm_Project/numbers.py
```

Рисунок 11 – Слияние ветки feature с веткой develop

Ответы на контрольные вопросы:

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

Установка Python в Windows:

- 1) Загрузка дистрибутива.
- 2) Запуск загруженного файла.
- 3) Выбор параметров установки.
- 4) Установка.

Для установки Python в Linux необходимо ввести команду `sudo apt-get install python3`.

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

Пакет Anaconda 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

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

Для того, чтобы задать используемый интерпретатор языка Python в IDE PyCharm, необходимо выбрать соответствующий файл при создании проекта.

5. Как осуществить запуск программы с помощью IDE PyCharm?

Для запуска программы нужно нажать на кнопку «Run», которая выглядит как зелёный треугольник.

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

В интерактивном режиме пользователь сам вводит код программы в командную строку, а в пакетном режиме через консоль открывается файл с кодом.

7. Почему язык программирования Python называется языком динамической типизации?

В языке Python тип переменной определяется непосредственно при выполнении программы.

8. Какие существуют основные типы в языке программирования Python?

Основные типы данных в Python:

- 1) None (неопределенное значение переменной).
- 2) Логические переменные (Boolean Type).
- 3) Числа (Numeric Type).
- 4) Списки (Sequence Type).

5) Строки (Text Sequence Type).

7) Множества (Set Types).

8) Словари (Mapping Types).

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, происходит следующее: создается объект (можно представить, что в этот момент создается ячейка и этот объект кладется в эту ячейку). Данный объект имеет некоторый идентификатор, значение и тип. Посредством оператора «=» создается ссылка между переменной и объектом (переменная ссылается на объект).

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() возвращает идентификатор объекта, на который ссылается переменная.

Функция type() используется для определения типа переменной.

12. Что такое изменяемые и неизменяемые типы в Python.

Если тип переменной является изменяемым, то её можно изменить, а если нет, то её изменить нельзя.

13. Чем отличаются операции деления и целочисленного деления?

Целочисленное деление отличается от обычной операции деления тем, что возвращает целую часть частного, а дробная часть 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.

Для получения комплексно-сопряженного 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`.

В библиотеке `math` содержится большое количество часто используемых математических функций. Основные функции: `ceil()`, `fabs()`, `factorial()`, `floor()`, `exp()`, `log2()`, `log10()`, `log(x,[base])`, `pow(x,y)`, `sqrt(x)`, `cos(x)`, `sin(x)`, `tan(x)`, `acos(x)`, `asin(x)`, `atan(x)`, `math.e`, `math.pi`.

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.

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 строки в стиле языка Си.

Форматировать строки также можно с помощью оператора `%`.

f-строки - форматирование, которое появилось в Python 3.6. Этот способ похож на форматирование с помощью метода `format()`, но гибче, читабельней и быстрее.

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

Ввод с консоли осуществляется с помощью функции `input()`. Чтобы осуществить ввод значения целочисленной или вещественной переменной, нужно использовать преобразование типов.

Вывод

Исследован процесс установки и изучены базовые возможности языка Python версии 3.x.