

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития  
Кафедра инфокоммуникаций

**«Основы языка Python»**

**ОТЧЕТ**  
**по лабораторной работе №4**  
**дисциплины**  
**«Основы программной инженерии»**

Выполнила:

Кувшин Ирина Анатольевна

2 курс, группа ПИЖ-б-о-21-1,

09.03.04 «Программная инженерия»,

направленность (профиль) «Разработка

и сопровождение программного

обеспечения», очная форма обучения

---

(подпись)

Проверил:

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

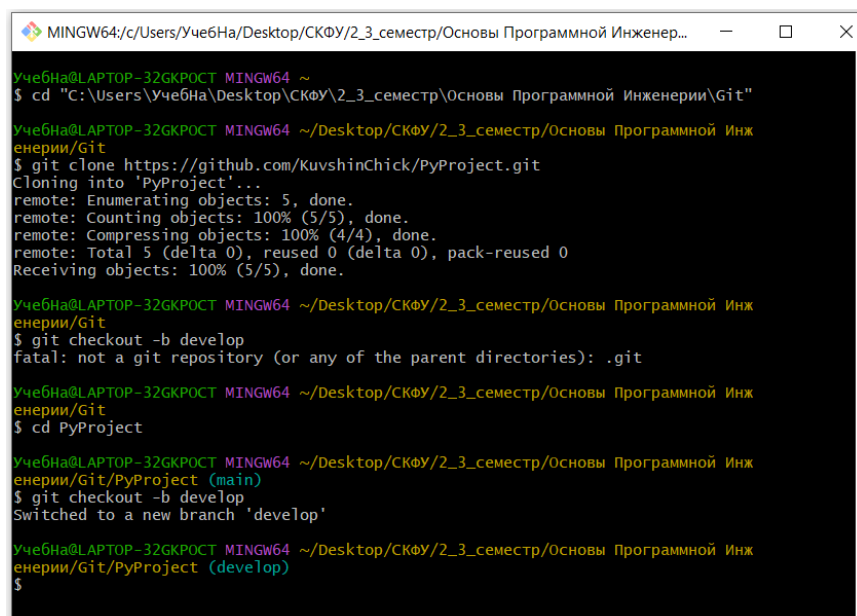
Ставрополь, 2022 г.

**Цель работы:** приобретение навыков программирования разветвляющихся алгоритмов и алгоритмов циклической структуры. Освоить операторы языка Python версии 3.x if , while , for , break и continue , позволяющих реализовывать разветвляющиеся алгоритмы и алгоритмы циклической структуры.

**Ссылка на репозиторий:** <https://github.com/KuvshinChick/TheFirstLab.git>

### 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.
4. Дополните файл .gitignore необходимыми правилами для работы с IDE PyCharm.
5. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.



```
MINGW64:/c:/Users/Учебна/Desktop/СКФУ/2_3_семестр/Основы Программной Инженер...
Учебна@LAPTOP-32GKPOCT MINGW64 ~
$ cd "C:\Users\Учебна\Desktop\СКФУ\2_3_семестр\Основы Программной Инженерии\Git"

Учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы Программной Инж
енерии/Git
$ git clone https://github.com/KuvshinChick/PyProject.git
Cloning into 'PyProject'...
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.

Учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы Программной Инж
енерии/Git
$ git checkout -b develop
fatal: not a git repository (or any of the parent directories): .git

Учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы Программной Инж
енерии/Git
$ cd PyProject

Учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы Программной Инж
енерии/Git/PyProject (main)
$ git checkout -b develop
Switched to a new branch 'develop'

Учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы Программной Инж
енерии/Git/PyProject (develop)
$
```

Рисунок 4.1 – Клонирование репозитория и создание ветки develop

```

Учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы Программной Инж
енерии/Git/PyProject (main)
$ git checkout -b develop
Switched to a new branch 'develop'

Учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы Программной Инж
енерии/Git/PyProject (develop)
$ git push origin develop
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
remote:   https://github.com/KuvshinChick/PyProject/pull/new/develop
remote:
To https://github.com/KuvshinChick/PyProject.git
 * [new branch]      develop -> develop

Учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы Программной Инж
енерии/Git/PyProject (develop)
$

```

Рисунок 4.2 – Отправка ветки develop на удаленный сервер

```

Учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы Программн
PyProject (develop)
$ git add .gitignore

Учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы Программн
PyProject (develop)
$ git commit -m "Modified .gitignore"
[develop df3604c] Modified .gitignore
1 file changed, 88 insertions(+)

```

Рисунок 4.3 – Обновление .gitignore

6. Создайте проект PyCharm 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`"

```

Вместо имя , возраст , место\_жительства должны быть данные, введенные пользователем. Примечание: можно писать фразы на русском языке, но если вы планируете стать профессиональным программистом, привыкайте к английскому.

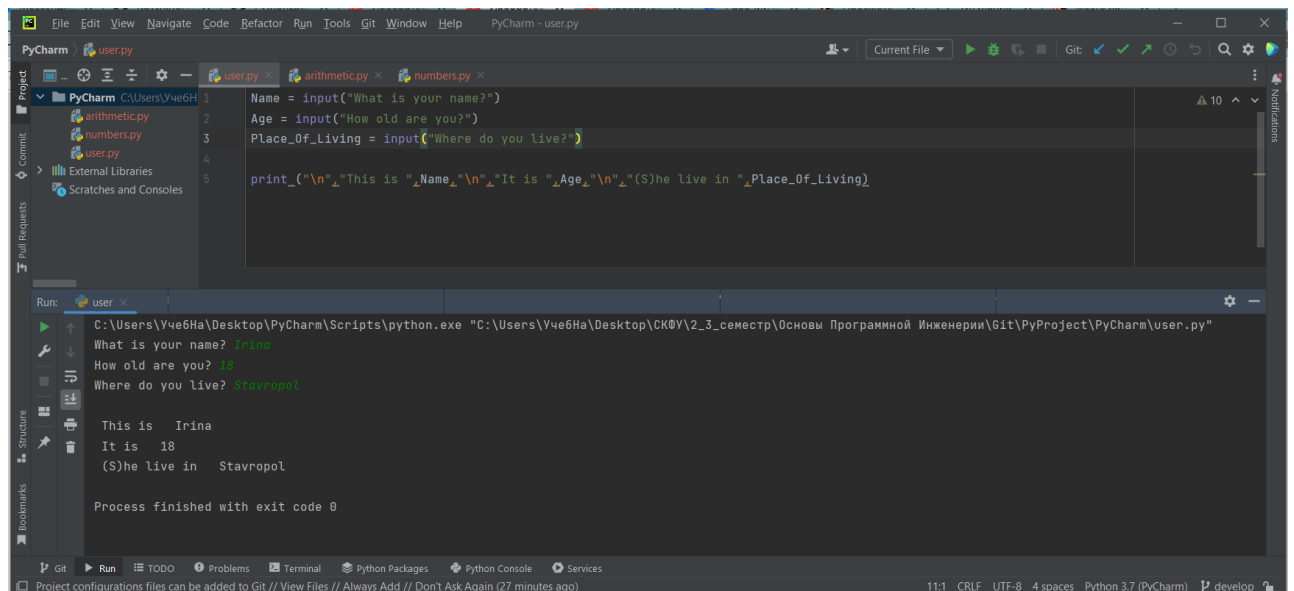


Рисунок 4.4 – Программа user.py

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.

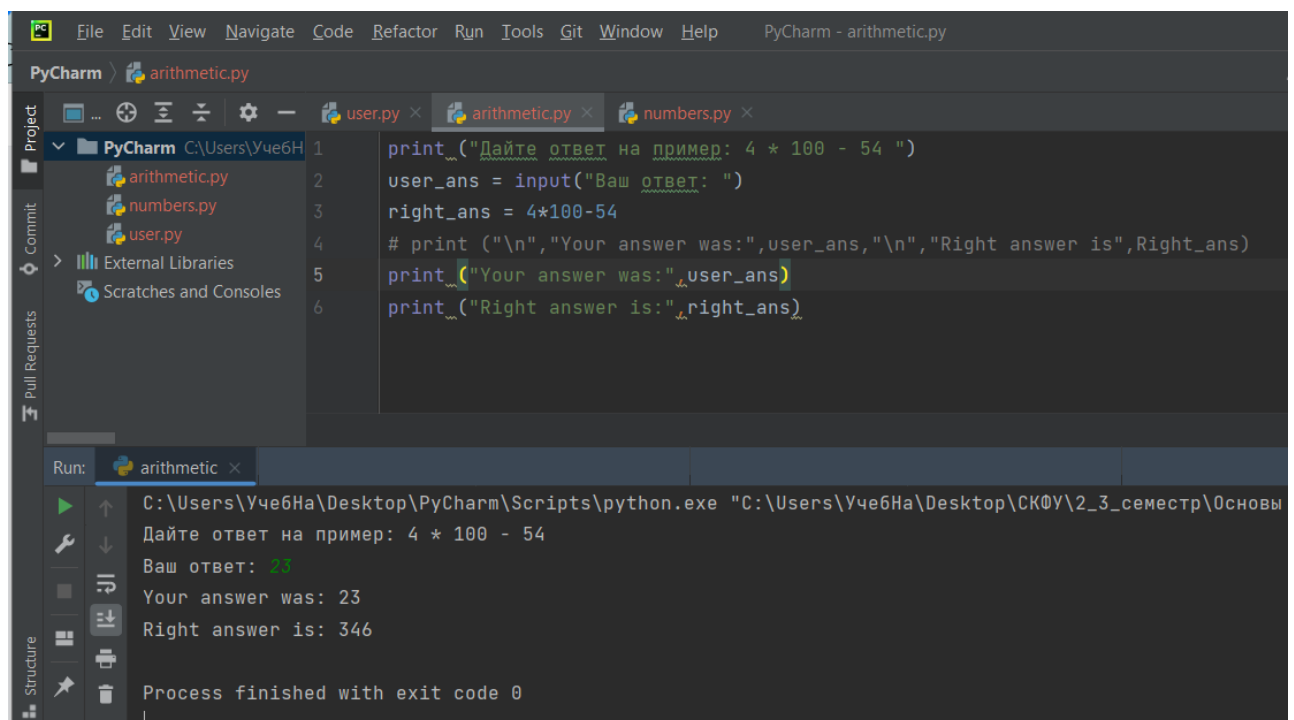


Рисунок 4.5 – Программа arithmetic.py

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.

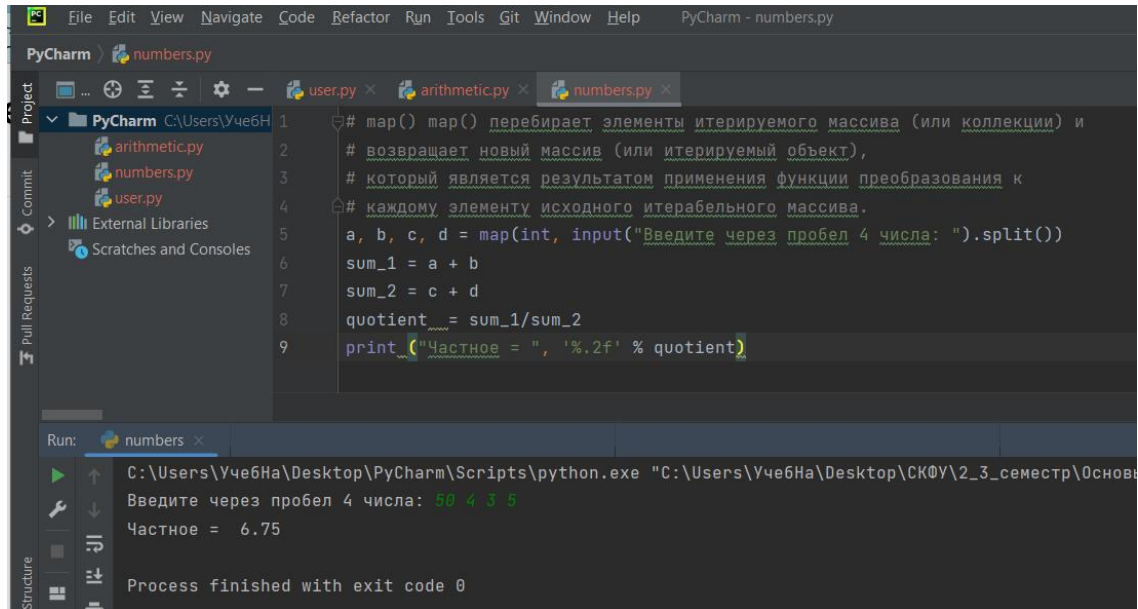


Рисунок 4.6 – Программа numbers.py

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

ИЗ: 15. Два автомобиля едут друг за другом с постоянными скоростями  $V_1$  и  $V_2$  км/ч ( $V_1 > V_2$ ). Определить, какое расстояние будет между ними через 30 мин после того, как первый автомобиль опередил второй на  $S$  км.

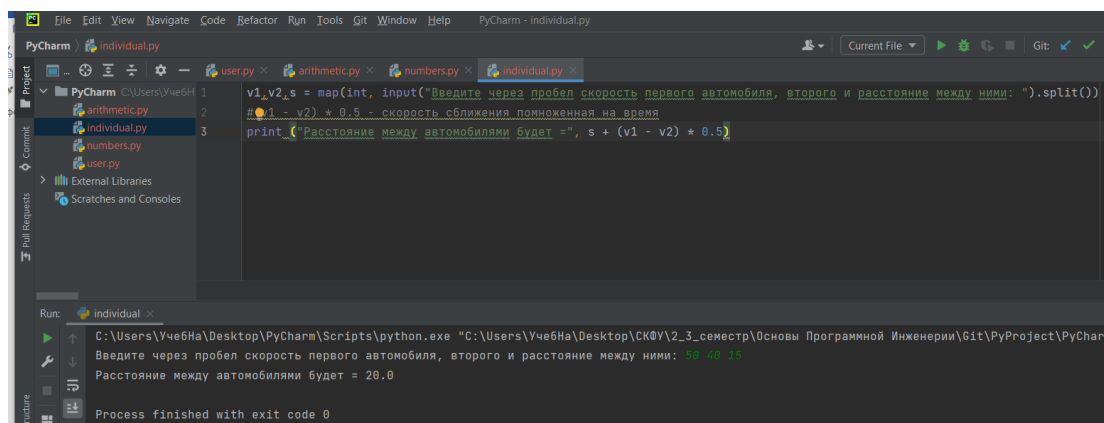


Рисунок 4.7 – Программа individual.py

### Задание повышенной сложности:

7. Часовая стрелка образует угол  $y$  с лучом, проходящим через центр и через точку, соответствующую 12 часам на циферблате,  $0 < y \leq 2\pi$ . Определить значение угла для минутной стрелки, а также количество полных часов и полных минут.

```
user.py x arithmetic.py x numbers.py x individual.py x individual_increasedLevel.py x
1 y = float(input("Введите угол 0<y<2П без PI: "))
2 print("Вы ввели:", y, "PI")
3 # Умножение на PI (3.14)
4 y = y*3.14
5 # Вычисление угла часовой стрелки в градусах (из радиан)
6 y=y*180.0/3.14
7 print("Угол часовой стрелки: ", y)
8 # Вычисление угла минутной стрелки (%30 - отделить целые часы, *12 - угол минутной стрелки)
9 # Тк угловая скорость минутной стрелки в 12 раз больше, чем часовой
10 x = y%30*12
11 print("Угол минутной стрелки: ", x)
12 # Кол-во полных часов = угол часовой /30
13 # Тк одному полному часу соответствуют 30 градусов
14 hour = y/30
15 print("Полных часов: ", hour)
16 # 1градус часовой соответствует двум минутам
17 # Тк одному полному часу соответствуют 30 градусов
18 min = y%30*2
19 print("Полных минут: ", min)
```

Рисунок 4.8 – Код программы

Run: individual\_increasedLevel x

C:\Users\УчебНа\Desktop\PyCharm\Scripts\python.exe "C:\Users\УчебНа\Desktop\PyCharm\Scripts\python.exe "C:\Users\УчебНа\Desktop\PyCharm\Scripts\python.exe "C:\Users\УчебНа\Desktop\PyCharm\Scripts\python.exe"

Введите угол 0<y<2П без PI: 0.25

Вы ввели: 0.25 PI

Угол часовой стрелки: 45.0

Угол минутной стрелки: 180.0

Полных часов: 1.5

Полных часов: 30.0

Process finished with exit code 0

individual\_increasedLevel x

C:\Users\УчебНа\Desktop\PyCharm\Scripts\python.exe "C:\Users\УчебНа\Desktop\PyCharm\Scripts\python.exe "C:\Users\УчебНа\Desktop\PyCharm\Scripts\python.exe "C:\Users\УчебНа\Desktop\PyCharm\Scripts\python.exe"

Введите угол 0<y<2П без PI: 0.9

Вы ввели: 0.9 PI

Угол часовой стрелки: 162.0

Угол минутной стрелки: 144.0

Полных часов: 5.4

Полных минут: 24.0

Process finished with exit code 0

Рисунок 4.9 – Результат программы

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

13. Выполните слияние ветки для разработки с веткой master.

14. Отправьте сделанные изменения на сервер GitHub.

```
Учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы Программной Инженерии/Git/PyProject (develop)
$ git checkout -b feature
Switched to a new branch 'feature'

Учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы Программной Инженерии/Git/PyProject (feature)
$ git branch
  develop
* feature
  main
```

Рисунок 4.10 – Создание ветки feature

```
Учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы Программной Инженерии/Git/PyProject (feature)
$ git checkout develop
Switched to branch 'develop'

Учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы Программной Инженерии/Git/PyProject (develop)
$ git merge feature
Updating df3604c..4a27db2
Fast-forward
 PyCharm/.idea/.gitignore      | 3 +++
 PyCharm/.idea/PyCharm.iml     | 8 ++++++++
 .../.idea/inspectionProfiles/profiles_settings.xml | 6 ++++++
 PyCharm/.idea/misc.xml       | 4 ++++
 PyCharm/.idea/modules.xml     | 8 ++++++++
 1 file changed, 29 insertions(+), 0 deletions(-)
```

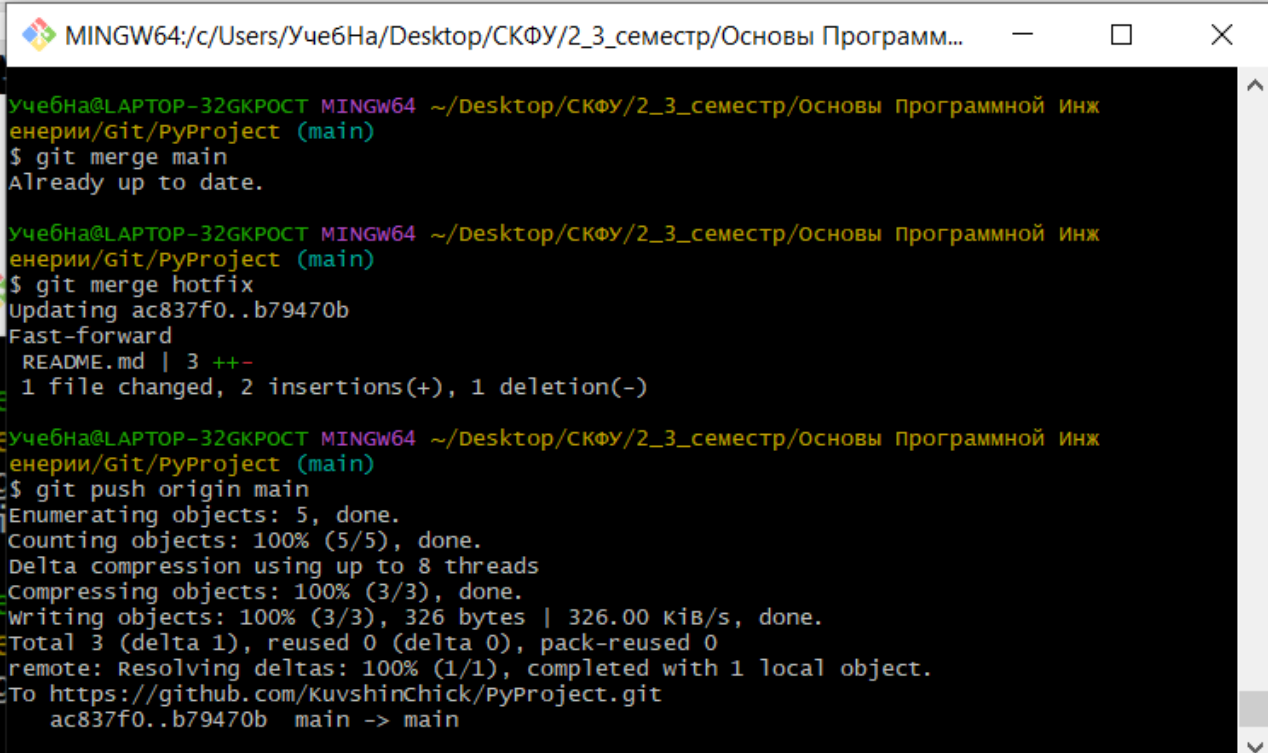
Рисунок 4.11 – Слияние веток

```
MINGW64:/c:/Users/Учебна/Desktop/СКФУ/2_3_семестр/Основы Программной Инженерии/Git/PyProject
Учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы Программной Инженерии/Git/PyProject (main)
$ git checkout -b hotfix
Switched to a new branch 'hotfix'

Учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы Программной Инженерии/Git/PyProject (hotfix)
$ git add README.md

Учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы Программной Инженерии/Git/PyProject (hotfix)
$ git commit -m "+README"
[hotfix b79470b] +README
 1 file changed, 2 insertions(+), 1 deletion(-)
```

Рисунок 4.12 – Создание ветки hotfix и изменение файла README

A screenshot of a terminal window with a black background and white text. The window title is 'MINGW64:/c/Users/УчебНа/Desktop/СКФУ/2\_3\_семестр/Основы Программ...'. The terminal shows a user performing a 'git merge main' which is successful. Then, they perform 'git merge hotfix', which updates the main branch with a fast-forward merge of a hotfix. The merge shows changes to README.md with 3 additions and 1 deletion. Finally, they run 'git push origin main', which successfully pushes the changes to the remote repository at https://github.com/kuvshinchick/PyProject.git.

```
MINGW64:/c/Users/УчебНа/Desktop/СКФУ/2_3_семестр/Основы Программ...
учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы программной инж
нерии/Git/PyProject (main)
$ git merge main
Already up to date.

учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы программной инж
нерии/Git/PyProject (main)
$ git merge hotfix
Updating ac837f0..b79470b
Fast-forward
 README.md | 3 ++-
 1 file changed, 2 insertions(+), 1 deletion(-)

учебна@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_3_семестр/Основы программной инж
нерии/Git/PyProject (main)
$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 326 bytes | 326.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/kuvshinchick/PyProject.git
ac837f0..b79470b main -> main
```

Рисунок 4.13 – Слияние веток и отправка в удаленный реп

### Контрольные вопросы:

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

Windows:

- Загрузить дистрибутив (с расширением exe/ с расширением zip)
- Запустить скачанный установочный файл.
- Выбрать способ установки
- Отметить необходимые опции установки
- Выбрать место установки

Linux:

- взять из репозитория при помощи команды `$ sudo apt-get install python3`

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

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

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

При создании нового проекта нужно будет указать путь до него и интерпретатор.

### 5. Как осуществить запуск программы с помощью IDE PyCharm?

После создания нового проекта нужно добавить python файл в проект и написать простую программу, ПКМ по рабочей области, нажать пункт «Run (имя файла)»

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

Интерактивный режим – непосредственное выполнение команд одна за другой в консоли. Пакетный режим – запуск программы из файла.

### 7. Почему язык программирования Python называется языком динамической типизации?

Также языки бывают с динамической и статической типизацией. В первом случае тип переменной определяется непосредственно при выполнении программы, во втором – на этапе компиляции.

### 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)

1. int – целое число
2. float – число с плавающей точкой
3. complex – комплексное число

## 4. Списки (Sequence Type)

1. list – список
2. tuple – кортеж
3. range – диапазон

## 5. Строки (Text Sequence Type )

1. str

## 6. Бинарные списки (Binary Sequence Types)

1. bytes – байты
2. bytearray – массивы байт
3. memoryview – специальные объекты для доступа к внутренним

данным объекта через protocol buffer

## 7. Множества (Set Types)

1. set – множество
2. frozenset – неизменяемое множество

## 8. Словари (Mapping Types)

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. Например строка

```
b = 5
```

объявляет переменную b и присваивает ей значение 5

Целочисленное значение 5 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 (об этом чуть позже).

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. При инициализации переменной, на уровне интерпретатора, происходит следующее: создается целочисленный объект 5 (можно представить, что в этот момент создается ячейка и 5 кладется в эту ячейку); данный объект имеет некоторый идентификатор, значение: 5, и тип: целое число; посредством оператора “=” создается ссылка между переменной b и целочисленным объектом 5 (переменная b ссылается на объект 5).

Имя переменной не должно совпадать с ключевыми словами интерпретатора Python.

#### 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.

```
>>> import keyword
>>> print("Python keywords: ", 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().

```
>>> a = 4
>>> b = 5
>>> id(a)
1829984576
>>> id(b)
1829984592
>>> a = b
>>> id(a)
1829984592
```

Как видно из примера, идентификатор – это некоторое целочисленное значение, посредством которого уникально адресуется объект. Изначально переменная `a` ссылается на объект 4 с идентификатором 1829984576, переменная `b` – на объект с `id = 1829984592`. После выполнения операции присваивания `a = b`, переменная `a` стала ссылаться на тот же объект, что и `b`.

Функция `type()` возвращает тип конкретного объекта/переменной

## 12. Что такое изменяемые и неизменяемые типы в Python.

В Python существуют изменяемые и неизменяемые типы. 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.

## 13. Чем отличаются операции деления и целочисленного деления?

- Целочисленное деление возвращает целую часть от деления, тип данных `int`
- Деление возвращает `float`

## 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  $a + bj$ .

```
>>> z = 1 + 2j
>>> print(z)
(1+2j)
>>> x = complex(3, 2)
>>> print(x)
(3+2j)
```

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.

Для работы с данным модулем его предварительно нужно импортировать.

```
>>> import math
```

Отличие модуля `cmath` от `math` заключается в том, что модуль `cmath` работает с комплексными числами, а модуль `math` работает с математическими операциями

16. Каково назначение именованных параметров `sep` и `end` в функции `print()`? Через параметр «`sep`» можно указать отличный от пробела разделитель строк.

```
>>> print("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun", sep="-")
Mon-Tue-Wed-Thu-Fri-Sat-Sun
>>> print(1, 2, 3, sep="//")
1//2//3
```

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.

```
>>> print(10, end="")
10>>>
```

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. Рассмотрим пример:

```
>>> pupil = "Ben"
>>> old = 16
>>> grade = 9.2
>>> print("It's %s, %d. Level: %f" % (pupil, old, grade))
It's Ben, 16. Level: 9.200000
```

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. Если бы требовалось подставить три строки, то во всех случаях использовалось бы сочетание `%s`.

f-строки. Форматирование, которое появилось в Python 3.6 ([PEP 498](#)). Этот способ похож на форматирование с помощью метода `format()`, но гибче, читабельней и быстрее.

```
>>> name = "Дмитрий"
>>> age = 25
>>> print(f"Меня зовут {name} Мне {age} лет.")
>>> Меня зовут Дмитрий. Мне 25 лет.
```

f-строки делают очень простую вещь — они берут значения переменных, которые есть в текущей области видимости, и подставляют их в строку. В самой строке вам лишь нужно указать имя этой переменной в фигурных скобках.

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

Input() возвращает строковое значение. Чтобы получить число, нужно использовать функции преобразования типов.

Целочисленный ввод: `int(input())`

Вещественный ввод: `float(input())`