

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 «Основы кроссплатформенного программирования»

Выполнил:
Худяков Максим Дмитриевич
1 курс, группа ИТС-б-о-22-1,
11.03.02 «Инфокоммуникационные
технологии и системы связи»,
направленность (профиль)
«Инфокоммуникационные системы и
сети», очная форма обучения

(подпись)

Руководитель практики:
Воронкин Р. А., доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

ОСНОВЫ ЯЗЫКА PYTHON

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

Ход работы:

Задание 1. Создал общедоступный репозиторий на GitHub, в котором использована лицензий MIT и язык программирования Python, также добавил файл .gitignore с необходимыми правилами. Клонировал свой репозиторий на свой компьютер.

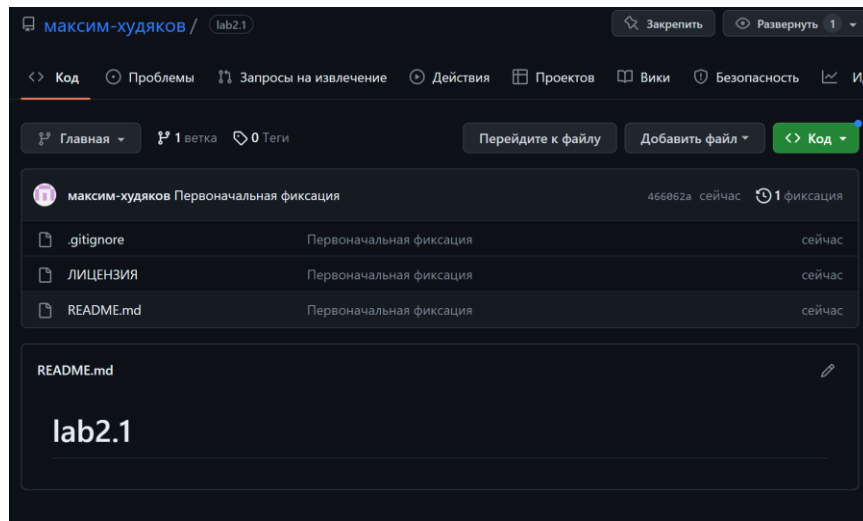


Рисунок 1. Клонирование репозитория

Задание 2. Организовал свой репозиторий в соответствии с моделью ветвления git-flow, появилась новая ветка develop в которой буду выполнять дальнейшие задачи.

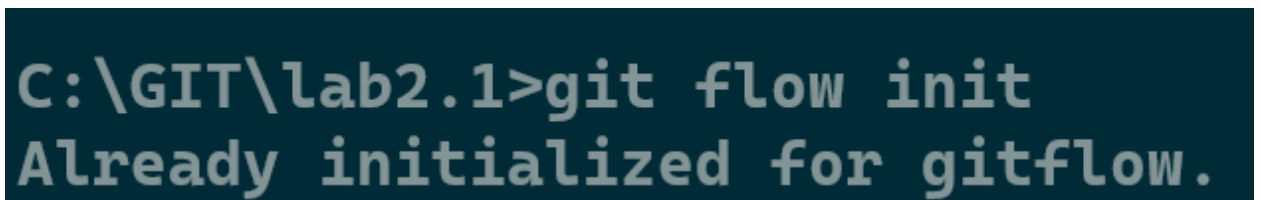


Рисунок 2. Модель ветвления git-flow

Задание 4. Создал проект PyCharm в папке репозитория. Добавил новый файл user.py.

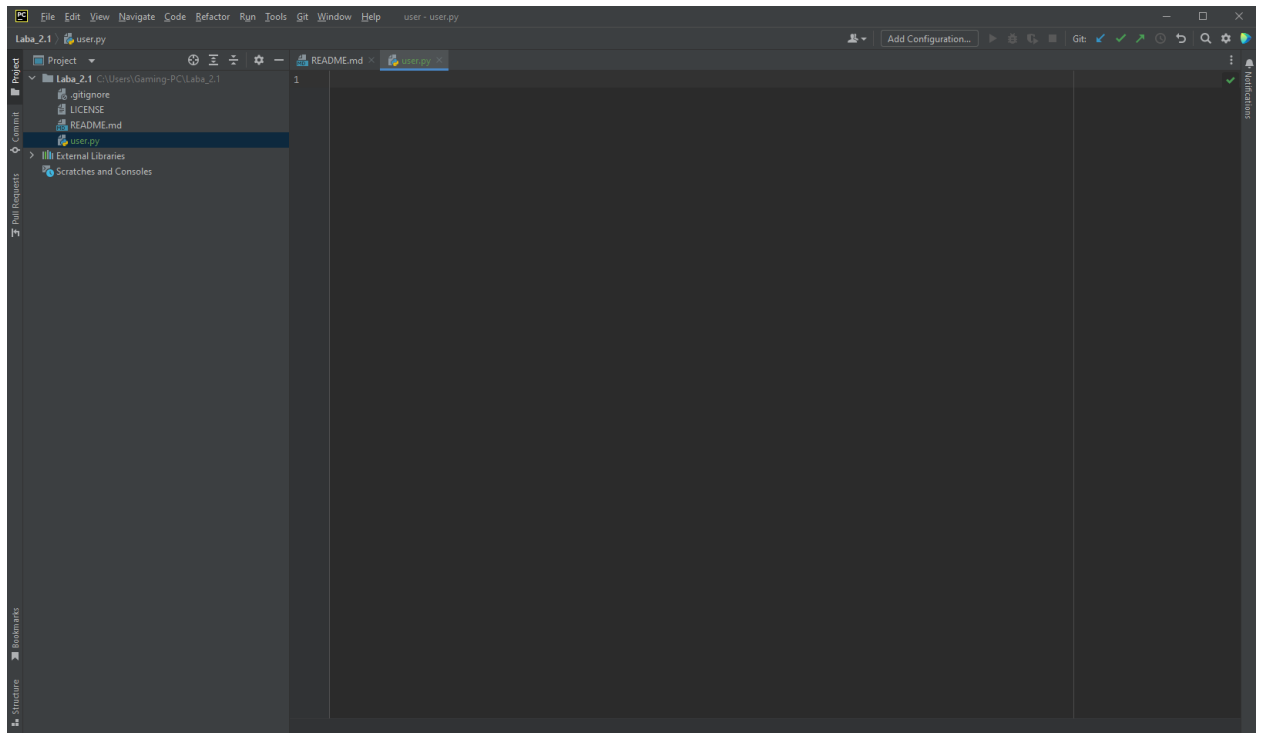


Рисунок 3. Добавил новый файл user.py

Задание 5. Написал программу и проверил на работоспособность в файле user.py. Необходимо, чтобы данная программа запрашивала у пользователя его имя, возраст, где он живет и выводила эти данные, написанные пользователем на экран.

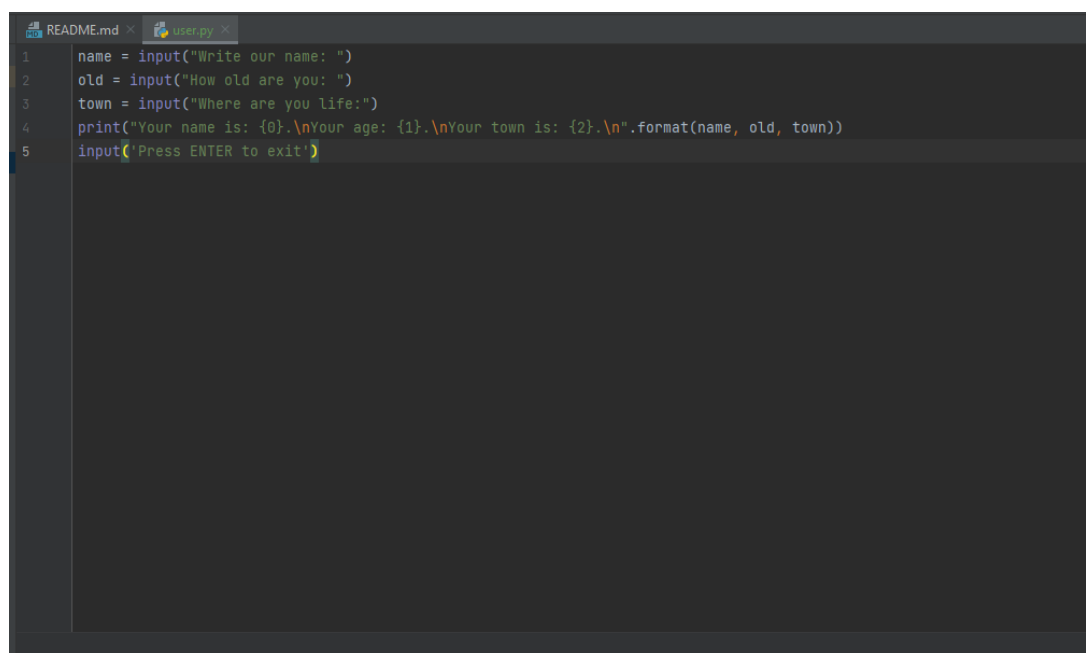


Рисунок 4. Программа user.py

Так как закончил написание программы в ветке feature/user, то необходимо закоммитить изменения и закончить работу с ней с помощью команды в Git: git flow feature finish user. В этом случае ветка feature/user автоматически сольется с веткой develop и удалиться.

```
C:\Users\Gaming-PC\Laba_2.1>git add user.py
C:\Users\Gaming-PC\Laba_2.1>git commit -m "user.py"
[feature/user d0775b4] user.py
1 file changed, 5 insertions(+)
create mode 100644 user.py

C:\Users\Gaming-PC\Laba_2.1>git flow feature finish user
Switched to branch 'develop'
Updating 8044b25..d0775b4
Fast-forward
 user.py | 5 +++++
1 file changed, 5 insertions(+)
create mode 100644 user.py
Deleted branch feature/user (was d0775b4).

Summary of actions:
- The feature branch 'feature/user' was merged into 'develop'
- Feature branch 'feature/user' has been locally deleted
- You are now on branch 'develop'
```

Рисунок 5. git flow feature finish user

Далее необходимо проработать с веткой release/ (т.е. внести изменения если они не обходимы в программе)

```
C:\Users\Gaming-PC\Laba_2.1>git flow release start user
Switched to a new branch 'release/user'

Summary of actions:
- A new branch 'release/user' was created, based on 'develop'
- You are now on branch 'release/user'

Follow-up actions:
- Bump the version number now!
- Start committing last-minute fixes in preparing your release
- When done, run:

    git flow release finish 'user'
```

Рисунок 6. git flow release finish user

Написал программу и проверил на работоспособность в файле arithmetic.py. В новой ветке feature/arithmetic от develop.

```
C:\Users\Gaming-PC\Laba_2.1>git flow feature start arifhmetic
Switched to a new branch 'feature/arifhmetic'

Summary of actions:
- A new branch 'feature/arifhmetic' was created, based on 'develop'
- You are now on branch 'feature/arifhmetic'

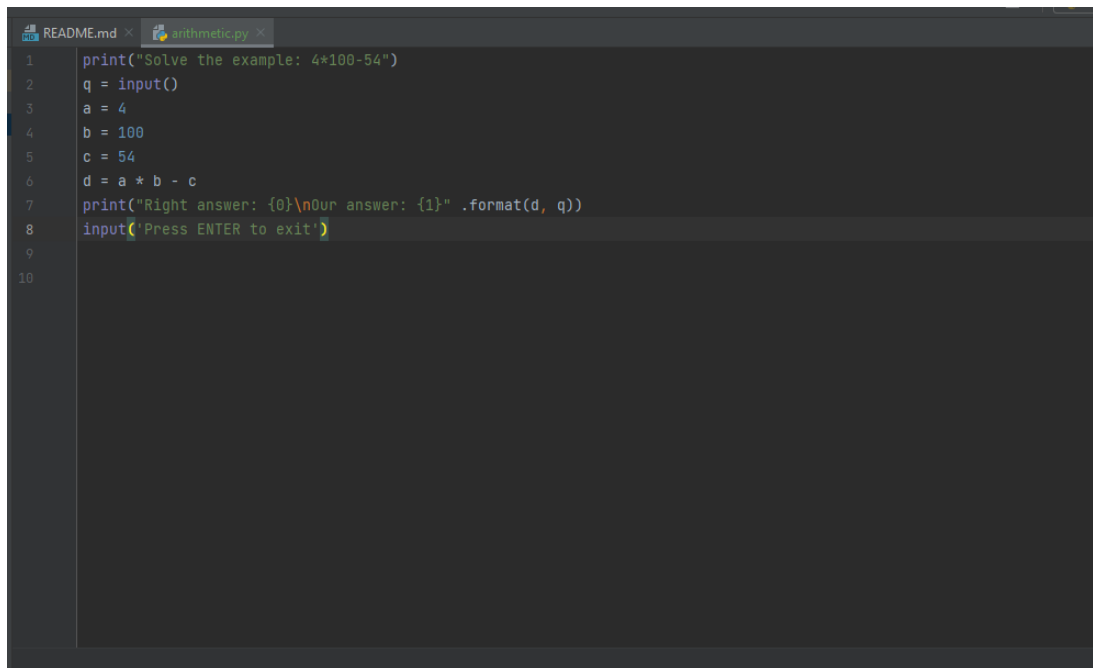
Now, start committing on your feature. When done, use:

    git flow feature finish arifhmetic
```

Рисунок 7. git flow feature/arithmetic

Создал новый файл и написал программу в этой ветке.

Необходимо, чтобы данная программа предлагала бы пользователю решить пример $4 * 100 - 54$. Потом выводила бы на экран правильный ответ и ответ пользователя.

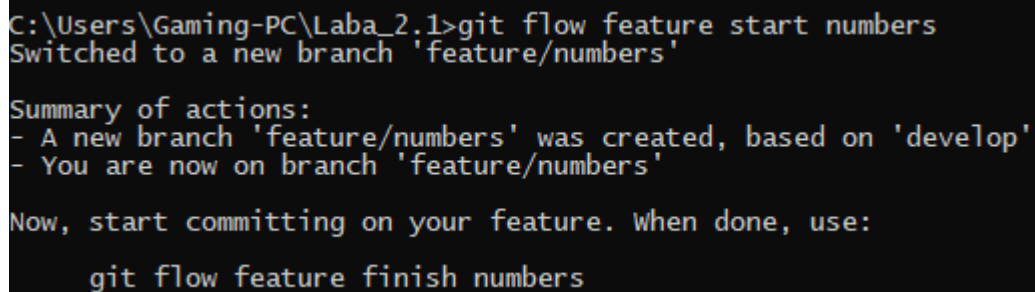


```
1 print("Solve the example: 4*100-54")
2 q = input()
3 a = 4
4 b = 100
5 c = 54
6 d = a * b - c
7 print("Right answer: {0}\nOur answer: {1}" .format(d, q))
8 input('Press ENTER to exit')
9
10
```

Рисунок 8. Программа arithmetic

Далее проделываю в Git те же операции, описанные в задании 4 (добавление изменений, коммит, завершение с веткой feature/arithmetic, создание ветки release/arithmetic, завершение с этой веткой и слияние с веткой main).

Задание 6. Создал новую ветку feature/numbers на основании ветки develop и в данной ветке написал программу.



```
C:\Users\Gaming-PC\Laba_2.1>git flow feature start numbers
Switched to a new branch 'feature/numbers'

Summary of actions:
- A new branch 'feature/numbers' was created, based on 'develop'
- You are now on branch 'feature/numbers'

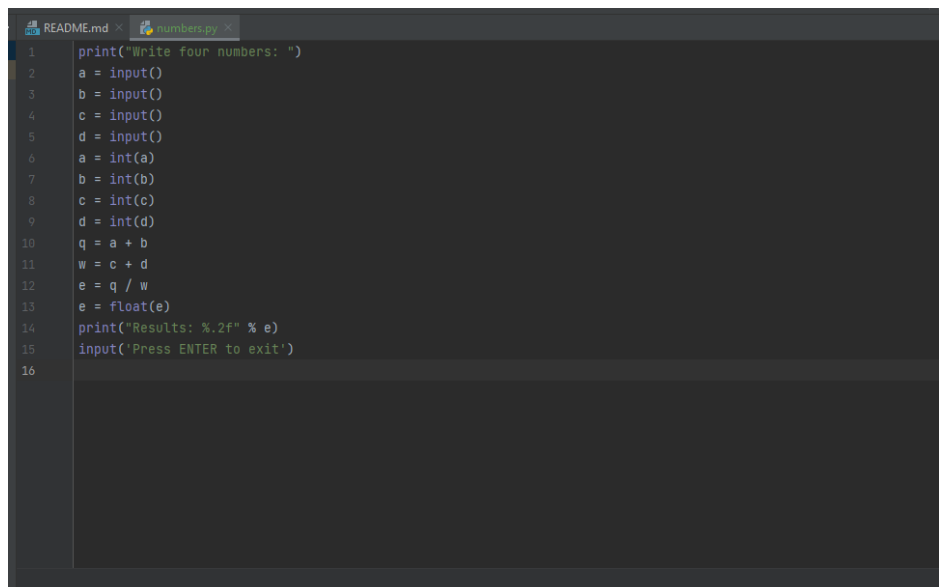
Now, start committing on your feature. When done, use:

    git flow feature finish numbers
```

Рисунок 9. git flow feature/numbers

Создал новый файл numbers.py Написал программу и проверил на работоспособность.

Необходимо, чтобы данная программа запросила у пользователя четыре числа, далее отдельно складывала первые два числа и вторые два числа, разделила первую сумму на вторую и вывела результат на экран (ответ должен содержать 2 цифры после запятой).



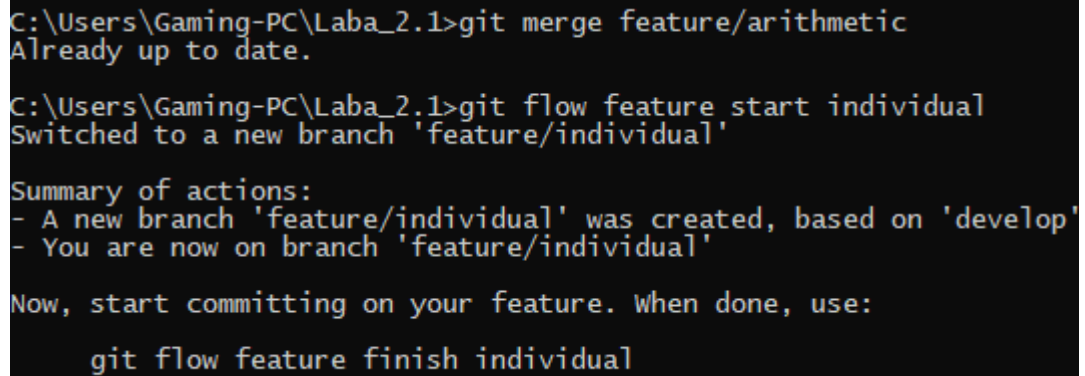
```
1 print("Write four numbers: ")
2 a = input()
3 b = input()
4 c = input()
5 d = input()
6 a = int(a)
7 b = int(b)
8 c = int(c)
9 d = int(d)
10 q = a + b
11 w = c + d
12 e = q / w
13 e = float(e)
14 print("Results: %.2f" % e)
15 input('Press ENTER to exit')
16
```

Рисунок 10. Программа numbers

Далее проделываю в Git те же операции, описанные в задании 4 (добавление изменений, коммит, завершение с веткой feature/arithmetic,

создание ветки release/arithmetic, завершение с этой веткой и слияние с веткой main).

Задание 7. Создал новую ветку feature/individual на основании ветки develop.



```
C:\Users\Gaming-PC\Laba_2.1>git merge feature/arithmetic
Already up to date.

C:\Users\Gaming-PC\Laba_2.1>git flow feature start individual
Switched to a new branch 'feature/individual'

Summary of actions:
- A new branch 'feature/individual' was created, based on 'develop'
- You are now on branch 'feature/individual'

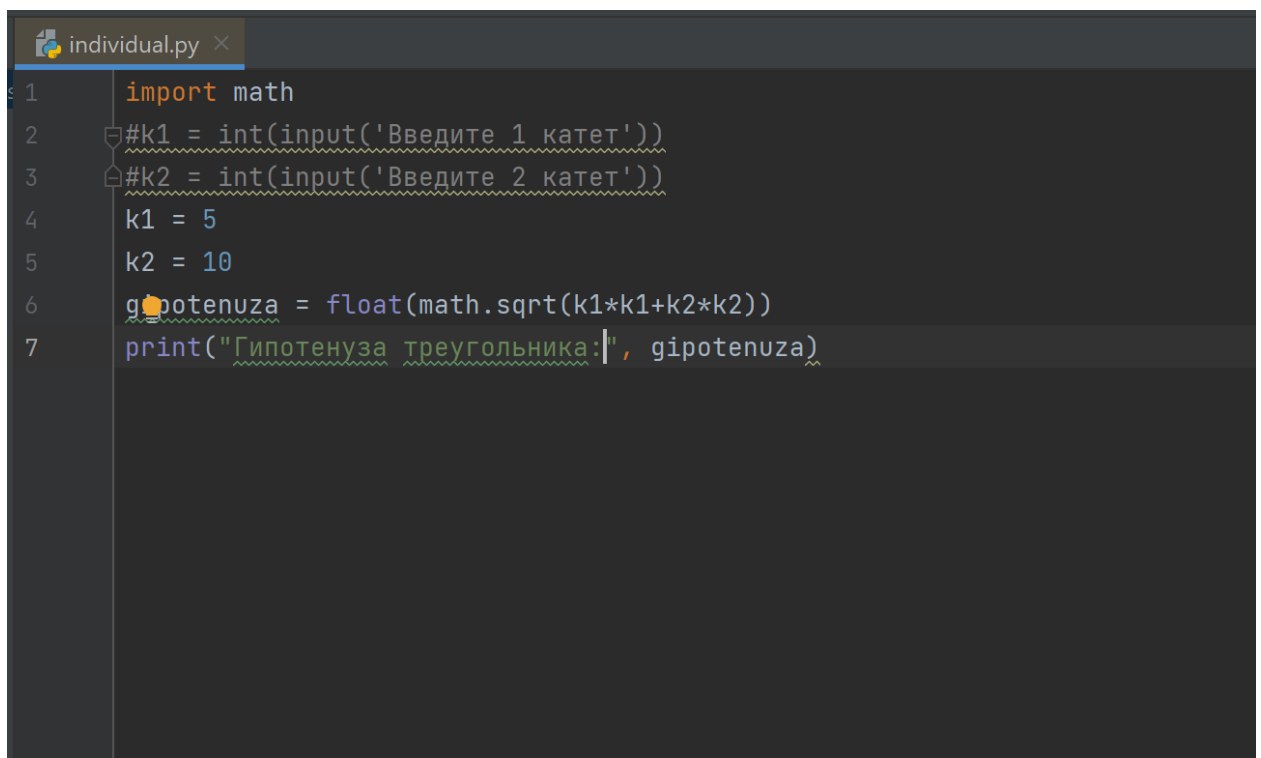
Now, start committing on your feature. When done, use:

    git flow feature finish individual
```

Рисунок 11. git flow feature/individual

Сделал индивидуальное задание в ветке feature/individual (вариант 21)

21. Даны катеты прямоугольного треугольника. Найти его гипотенузу.



```
individual.py x
1 import math
2 #k1 = int(input('Введите 1 катет'))
3 #k2 = int(input('Введите 2 катет'))
4 k1 = 5
5 k2 = 10
6 gipotenuza = float(math.sqrt(k1*k1+k2*k2))
7 print("Гипотенуза треугольника:", gipotenuza)
```

Рисунок 12. Программа individual

Задание 8. Создал новую ветку feature/hardtask

```
C:\Users\Gaming-PC\Lab2_1>git flow feature start hardtask
Switched to a new branch 'feature/hardtask'

Summary of actions:
- A new branch 'feature/hardtask' was created, based on 'develop'
- You are now on branch 'feature/hardtask'

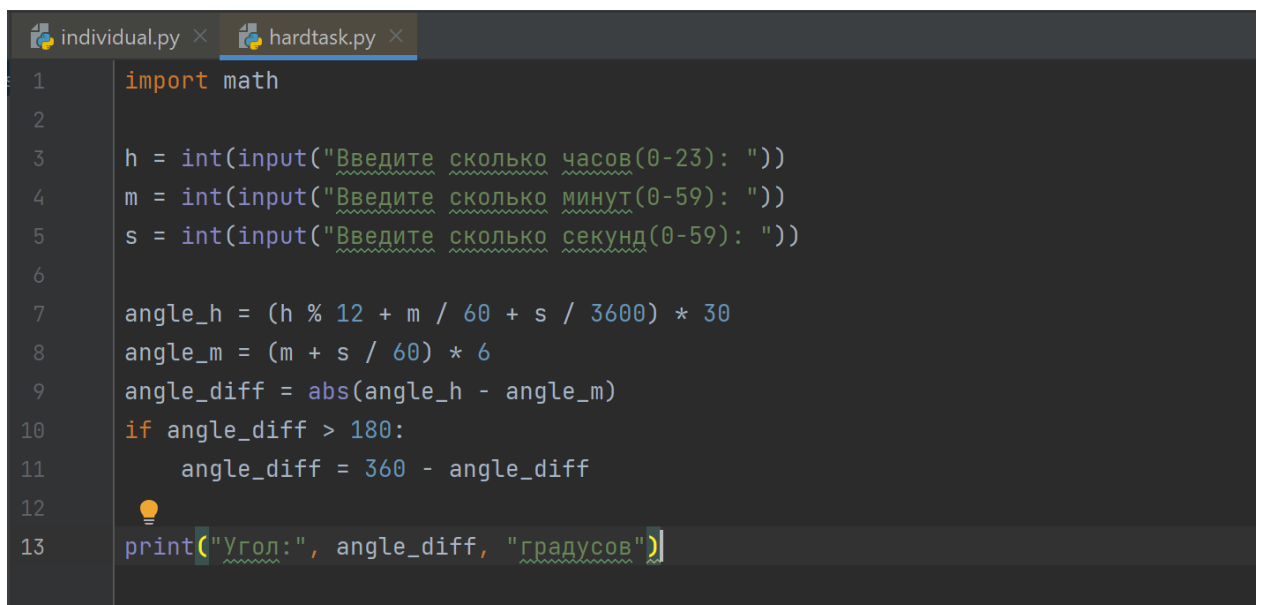
Now, start committing on your feature. When done, use:

    git flow feature finish hardtask
```

Рисунок 13. git flow feature/hardtask

Создал новый файл в этой ветке под названием hardtask. На основании новой созданной ветки сделал задание усложненного типа, вариант 5.

5. Даны целые числа h, m, s ($0 < h \leq 23, 0 \leq m \leq 59, 0 \leq s \leq 59$), 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.



```
1  import math
2
3  h = int(input("Введите сколько часов(0-23): "))
4  m = int(input("Введите сколько минут(0-59): "))
5  s = int(input("Введите сколько секунд(0-59): "))
6
7  angle_h = (h % 12 + m / 60 + s / 3600) * 30
8  angle_m = (m + s / 60) * 6
9  angle_diff = abs(angle_h - angle_m)
10 if angle_diff > 180:
11     angle_diff = 360 - angle_diff
12
13 print("Угол:", angle_diff, "градусов")
```

Рисунок 14. Программа hardtask

Ссылка на репозиторий: <https://github.com/maxim-hoodyakov/lab2.1>

Ответы на контрольные вопросы:

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

Для начала скачиваем дистрибутив по ссылке. Запускаем скачанный файл. Выбираем способ установки (install now или customize installation).

Отмечаем необходимые опции для установки. Выбираем место установки (доступно при выборе Customize installation).

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

Anaconda наиболее известна как дистрибутив Python со встроенным в него пакетным менеджером conda. Она позволяет изолировать окружение проекта от системной версии Python, который критически необходим для работы системы. Использование `sudo pip` считается плохой практикой. Также conda позволяет без проблем переносить окружение с одной машины на другую. Кроме того, если вы что-то сломаете, то с Anaconda вы всегда сможете откатиться на более старую версию окружения.

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?

Запускаем программу. И он не содержит ни одного файла с текстом программы. Создадим его. Нажимаем правую кнопку мыши, выбираем «New» -> «Python File». Вводим имя файла, например, `ex1` и этот файл автоматически добавляется в наш проект. Здесь мы можем писать наши программы, на Питоне.

5. Как осуществить запуск программы с помощью IDE PyCharm? Здесь при первом запуске необходимо создать новый проект. Нажимаем «Create New Project». В поле «Location» указывается расположение проекта и его имя.

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

Его суть в удобстве использования языка Питон, а именно его внешний вид и простота, сопровождаемая автоматикой действий, позволяющей пользователям работать с данными программами.

7. Почему язык программирования Python называется языком динамической типизации?

Потому что в Питоне 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)

int – целое число

float – число с плавающей точкой

complex – комплексное число

4. Списки (Sequence Type)

list – список

tuple – кортеж

range – диапазон Строки (Text Sequence Type)

str Бинарные списки (Binary Sequence Types)

bytes – байты

bytearray – массивы байт

memoryview – специальные объекты для доступа к внутренним данным

Объекта через protocol buffer Множества (Set Types)

set – множество

frozenset – неизменяемое множество Словари (Mapping Types)

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` 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. 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.

13. Чем отличаются операции деления и целочисленного деления?

Целочисленное деление (`div`) отличается от обычной операции деления тем, что возвращает целую часть частного, дробная часть отбрасывается.

Перед выполнением операции оба операнда округляются до целых значений.

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 $a + bj$.

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.

16. Каково назначение именованных параметров `sep` и `end` в функции `print()`?

`sep` – с помощью этого параметра вы можете указать разделитель строк. А по умолчанию в качестве разделителя используется пробел. `end` – этот параметр позволяет указать, что нужно добавить после последней строки. По умолчанию добавляется управляющий символ `'\n'` (перевод 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()`, который определен у строк, позволяет форматировать строку, вставляя в нее на место плейсхолдеров определенные значения. Для вставки в строку используются специальные параметры, которые обрамляются фигурными скобками (`{}`).

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

Если необходимо вести целочисленное значение можно использовать код: `A = input()` `A = int()`

Если необходимо вести вещественное значение можно использовать код: `A = input()` `A = float()`

Для ввода нужно нажать Enter после завершения набора текста. Обычно Enter добавляет символ новой строки (`\n`), но не в этом случае. Введенная строка просто будет передана приложению.

Вывод: исследовал процесс установки и базовые возможности языка Python3.