

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО
ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ**

**Федеральное государственное автономное
образовательное учреждение высшего
образования**

**«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ
УНИВЕРСИТЕТ»**

Кафедра инфокоммуникаций

**«Исследование возможностей Git для
работы локальными репозиториями»**

**Отчет по лабораторной работе № 2.1
по дисциплине «Основы программной инженерии»**

Выполнил студент группы ПИЖ-б-о-21-1
Гребенкин Е.А. _____ «14» сентября 2022г.

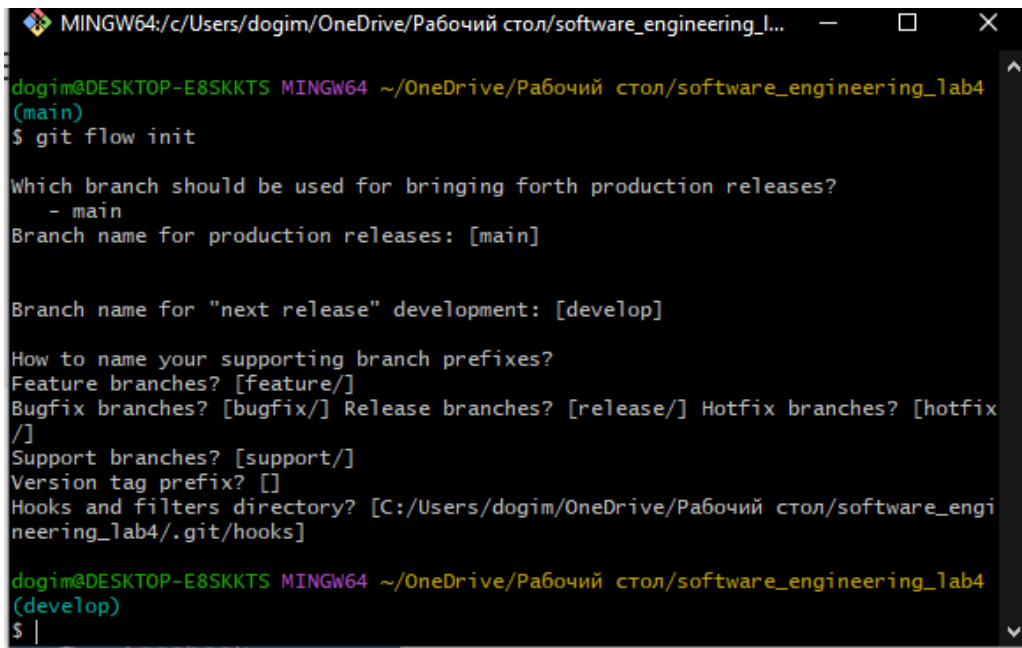
Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверила Воронкин Р.А. _____
(подпись)

МЕТОДИКА И ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

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



```
dogim@DESKTOP-E8SKKTS MINGW64 ~/OneDrive/Рабочий стол/software_engineering_lab4
(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/dogim/OneDrive/Рабочий стол/software_engineering_lab4/.git/hooks]

dogim@DESKTOP-E8SKKTS MINGW64 ~/OneDrive/Рабочий стол/software_engineering_lab4
(develop)
$ |
```

Рисунок 1 – модель ветвления git-flow

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`"
```

Код программы

```
1 name = input("Your name: ")
2 age = input("Your age: ")
3 city = input("Your city: ")
4
5 print(f"This is {name}"
6       f"It is {age}"
7       f"(S)he live in {city}")
8
```

```
Your name: Egor
Your age: 20
Your city: Stav
This is Egor
It is 20
(5)he live in Stav
```

Рисунок 2 – результат выполнения 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.

```
1  print("4 * 100 - 54 = ?")
2
3  answer = int(input("Send the answer: "))
4
5  if answer == 4 * 100 - 54:
6      print("It's right!")
7  else:
8      print(f"It's wrong! Right answer: {4 * 100 - 54}")
9
```

Код программы:

```
4 * 100 - 54 = ?
Send the answer: 321
It's wrong! Right answer: 346
```

Рисунок 3 – результат выполнения 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.

Код программы:

```
1 numbers: list = list(map(int, input("Enter 4 numbers: ").split()))
2
3 print((numbers[0] + numbers[1]) / (numbers[2] + numbers[3]))
4
```

```
Enter 4 numbers: 23 543 23 2
22.64
```

Рисунок 4 – результат выполнения программы

11. Напишите программу (файл individual.py) для решения индивидуального задания. (6 вариант)

6. Даны координаты на плоскости двух точек. Найти расстояние между этими точками.

Код программы:

```
1 from math import sqrt
2
3 x1, y1 = list(map(int, input("Enter x1, y1: ").split()))
4 x2, y2 = list(map(int, input("Enter x2, y2: ").split()))
5
6 print(sqrt((x2 - x1)**2 + (y2 - y1)**2))
7
```

```
Enter x1, y1: 45 564
Enter x2, y2: 5 45
520.5391435809606
```

Рисунок 5 – результат выполнения программы

Вопросы для защиты работы

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

- Для установки интерпретатора Python первое, что нужно сделать – это скачать дистрибутив.
- Запустить скачанный установочный файл.
- Выбрать способ установки.
- Отметить необходимые опции установки (доступно при выборе Customize installation).
- Выберете место установки (доступно при выборе Customize installation).

При установке для Linux, в случае ошибки необходимо либо собрать

Python из исходников либо взять из репозитория. Для установки из репозитория в Ubuntu воспользуйтесь командой «`sudo apt-get install python3`»

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

Anaconda является дистрибутивом языков программирования таких как Python и R для науки о данных и машинного обучения, а Python — это язык программирования высокого уровня общего назначения.

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, после чего запустится веб-сервер и среда разработки в браузере.

Создайте ноутбук для разработки, для этого нажмите на кнопку New (в правом углу окна) и в появившемся списке выберете Python. В результате будет создана новая страница в браузере с ноутбуком. Введите в первой ячейке команду «print("Hello, World!")» и нажмите Alt+Enter на клавиатуре. Ниже ячейки должна появиться соответствующая надпись.

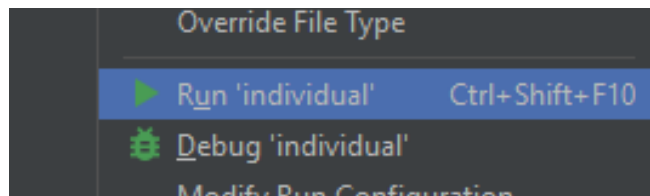
4. Как задать используемый интерпретатор языка Python в IDE PyCharm?

При создании нового проекта в PyCharm есть возможность выбрать путь до проекта и интерпретатор.

5. Как осуществить запуск программы с помощью IDE

PyCharm? Правой кнопкой в любом месте или по файлу слева и выбрать из

появившегося меню пункт «Run»



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

Интерактивный режим позволяет вводить команды, которые сразу же будут обрабатываться, это можно использовать в качестве калькулятора.

Пакетный режим позволяет запустить файл с исходным кодом.

7. Почему язык программирования Python называется языком динамической типизации?

Это означает, что одна и та же переменная в разное время может ссылаться на данные разного типа.

8. Какие существуют основные типы в языке программирования Python?

1. None (неопределенное значение переменной)

2. Логические переменные (Boolean Type)

3. Числа (Numeric Type)

3.1. int – целое число

3.2. float – число с плавающей точкой

3.3. complex – комплексное число

4. Списки
(Sequence Type)

4.1. list – список

4.2. tuple – кортеж

4.3. range –
диапазон

5. Строки (Text Sequence Type)

6. Бинарные списки (Binary Sequence Types)

6.1. bytes – байты

6.2. bytearray – массивы байт

6.3. memoryview – специальные объекты для
доступа к внутренним данным объекта через
protocol buffer

7. Множества (Set Types)

7.1. set – множество

7.2. frozenset – неизменяемое множество

8. Словари (Mapping Types)

8.1. dict – словарь

9. Как создаются объекты в памяти? Каково их устройство? В чем заключается процесс объявления новых переменных и работа операции присваивания?

При создании переменной, вначале создается объект, который имеет уникальный идентификатор, тип и значение, после этого переменная может ссылаться на созданный объект.

Создание переменных и объектов в Python происходит с помощью оператора присваивания («=»). Записывается имя переменной, оператор и значение, например, «a = 5». Множественное

присваивание (позиционное присваивание) в Python реализуется следующим образом: « a, b, c = 5, 3, 1 »

10. Как получить список ключевых слов в Python?

Нужно подключить модуль keyword и воспользоваться командой keyword.kwlist.

11. Каково назначение функций id() и type()?

id() – возвращает уникальный идентификатор объекта в программе. type() – возвращает тип переменной.

12. Что такое изменяемые и неизменяемые типы в Python.

Изменяемый объект можно изменить после его создания, а неизменяемый – нет. Во втором случае фактически мы не изменяем значение переменной, а создаем другой объект с тем же именем и присваиваем ему другое значение. Мы связываем имя переменной с новым значением. Теперь, при ее вызове, он будет выводить новое значение и ссылаться на новое местоположение.

13. Чем отличаются операции деления и целочисленного деления? Во втором случае не учитывается остаток от деления.

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.

15. Каково назначение и основные функции библиотеки

(модуля) `math`? По аналогии с модулем `math` изучите самостоятельно назначение и основные функции модуля `cmath`.

Библиотека `math` содержит большое количество часто используемых математических функций, такие как округление, логарифмы, факториал, модуль числа, экспонента (e^x), степень, квадратный корень, синус/косинус, числа π и e и т.д.

Модуль `cmath` – предоставляет функции для работы с комплексными числами. Из отличных функций можно выделить преобразование к полярным координатам и преобразование из полярных координат.

`cmath.isfinite(x)` - True, если действительная и мнимая части конечны.

`cmath.isinf(x)` - True, если либо действительная, либо мнимая часть бесконечна.

`cmath.isnan(x)` - True, если либо действительная, либо мнимая часть NaN.

16. Каково назначение именованных параметров `sep` и `end` в функции `print()`? `sep()` устанавливает отличный от пробела разделитель строк.

`End()` указывает, что делать, после вывода строки (по умолчанию стоит переход на новую строку)

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 выводимые строки.

Рассмотрим пример:

```
print("This is a {0}. It's {1}.".format("ball", "red"))
```

В строке в фигурных скобках указаны номера данных, которые будут сюда подставлены. Далее к строке применяется метод `format()`. В его скобках указываются сами данные (можно использовать переменные). На нулевом месте подставится первый аргумент метода `format()`, на место с номером 1 – второй и т. д.

Форматирование также может осуществляться в старом (Си-) стиле. Он схож с тем, как происходит вывод на экран в языке C. Пример:

```
print("It's %s, %d. Level: %f" % (pupil, old, grade))
```

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.

F-Строки являются упрощенной версией метода `format()`. F-strings являются строковыми литералами с «f» в начале и фигурные скобки, содержащие выражения, которые в дальнейшем будут заменены своими значениями. Пример:

```
print(f"Hello, {name}. You are {2*8}.") //name – переменная
```

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

С помощью функции `input()` можно получить вводимые с консоли данные. Однако они будут принадлежать к строковому типу, чтоб получить число нужно использовать функции преобразования типов:

```
c = float(input("Enter temperature (Celsius) : "))
```