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 ПРАКТИЧЕСКОЙ РАБОТЕ №4
дисциплины «Программирование на Python»
Вариант 23

Выполнил:
Мотовилов Вадим Борисович
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Информатика и вычислительная
техника», очная форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Роман Александрович
_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Порядок выполнения работы:

1. Создал репозиторий и скопировал его

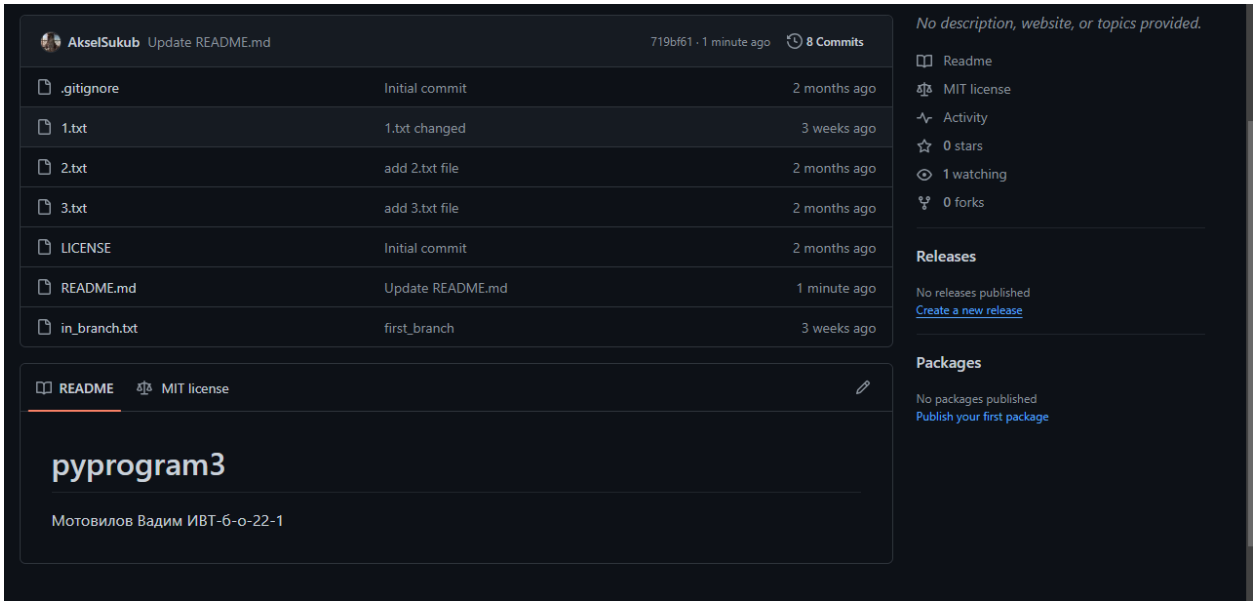

Рисунок 1. Созданный репозиторий

2. Изменил файл .gitignore и README.rm и добавил git flow

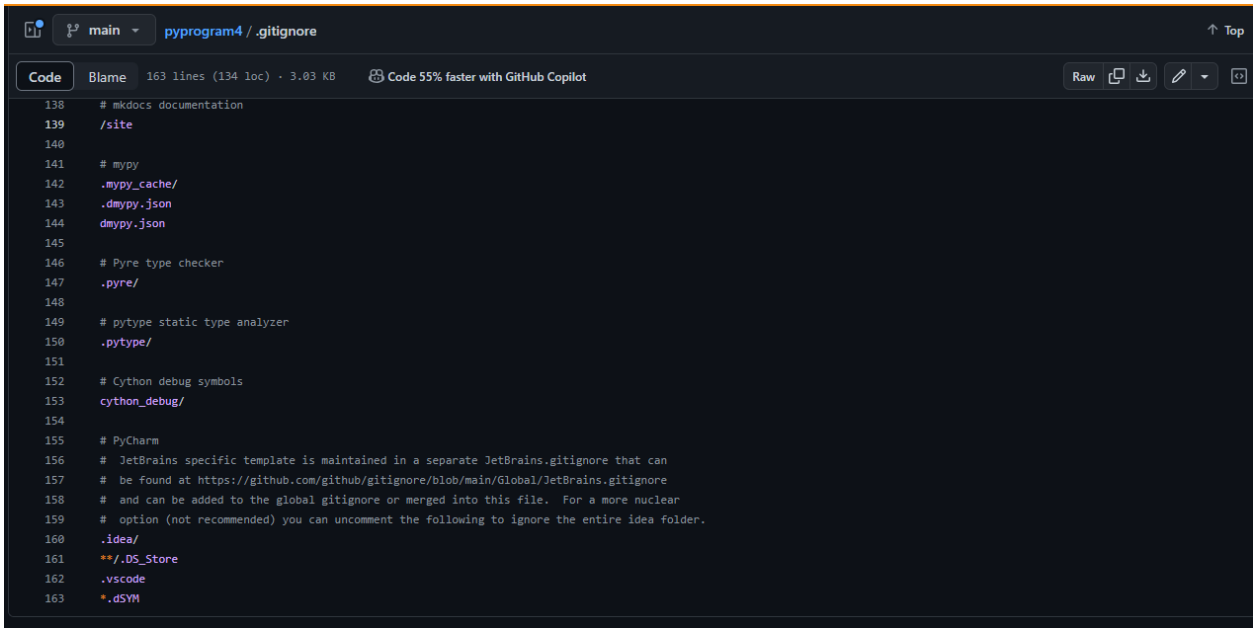

Рисунок 2. Измененный файл .gitignore

3. Выполнил задания

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`"
```

```
Name= input("What it your name? ")
Old= input("How old are you? ")
City= input("Where are you live? ")
print("This is {0} \nIt is {1} \n(S)he live in {2}".format(Name, Old, City))
```

```
What it your name? Vadim
How old are you? 19
Where are you live? Stavropol
This is Vadim
It is 19
(S)he live in Stavropol
PS C:\otkat>
```

Рисунок 3 . Выполнение 1 задания

Напишите программу (файл *arithmetic.py*), которая предлагала бы пользователю решить пример 4 * 100 - 54. Потом выводила бы на экран правильный ответ и ответ пользователя.

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


if __name__ == "__main__":
    print("Решите данное уравнение: 4*100-54; Ваш ответ: ")
    answer = int(input())
    print("Правильный ответ:", 4*100-54)
    print("Ваш ответ: ", answer)
```

```
Решите данное уравнение: 4*100-54; Ваш ответ:
345
Правильный ответ: 346
Ваш ответ:  345
```

Рисунок 4 . Выполнение 2 задания

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.

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


if __name__ == "__main__":
    num1 = float(input("Введите первое число: "))
    num2 = float(input("Введите второе число: "))
    num3 = float(input("Введите третье число: "))
    num4 = float(input("Введите четвертое число: "))
    sum1 = num1 + num2
    sum2 = num3 + num4
    result = sum1 / sum2
    print("Результат: {:.2f}".format(result))
```

```
Введите первое число: 12
Введите второе число: 31
Введите третье число: 32
Введите четвертое число: 8
Результат: 1.07
```

Рисунок 5 . Выполнение 3 задания

Даны катеты прямоугольного треугольника. Найти его периметр.

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


if __name__ == "__main__":
    side1 = float(input("Введите длину первой стороны прямоугольника: "))
    side2 = float(input("Введите длину второй стороны прямоугольника: "))
    perimeter = 2 * (side1 + side2)
    diagonal = (side1 + side2) ** 0.5
    print("Периметр прямоугольника: {:.2f}".format(perimeter))
    print("Длина диагонали прямоугольника: {:.2f}".format(diagonal))
```

```
Введите длину первой стороны прямоугольника: 4
Введите длину второй стороны прямоугольника: 3
Периметр прямоугольника: 14.00
Длина диагонали прямоугольника: 2.65
```

Рисунок 6 . Выполнение индивидуального задания

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.

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


if __name__ == "__main__":
    y = float(input("Введите угол Y: "))
    print("Значение угла для минутной стрелки: ", y % 30 * 12)
    print("Количество полных часов: ", y // 30)
    print("Количество полных минут: ", y % 30 * 2)
```

```
Введите угол Y: 345
Значение угла для минутной стрелки:  180.0
Количество полных часов:  11.0
Количество полных минут:  30.0
```

Рисунок 6 . Выполнение усложненного индивидуального задания

Ответы на контрольные вопросы:

1. Для чего нужны диаграммы деятельности UML? Позволяет наглядно визуализировать алгоритм программы.

2. Что такое состояние действия и состояние деятельности? Состояние действия - частный вид состояния деятельности, а конкретнее – такое состояние, которое не может быть подвергнуто дальнейшей декомпозиции. Состояние деятельности можно представлять себе как составное состояние, поток управления которого включает только другие состояния деятельности и действий.

3. Какие нотации существуют для обозначения переходов и ветвлений в диаграммах деятельности? Переходы, ветвление, алгоритм разветвляющейся структуры, алгоритм циклической структуры.

4. Какой алгоритм является алгоритмом разветвляющейся структуры? Алгоритм разветвляющейся структуры - это алгоритм, в котором вычислительный процесс осуществляется либо по одной, либо по другой ветви, в зависимости от выполнения некоторого условия.

5. Чем отличается разветвляющийся алгоритм от линейного? Линейный алгоритм - алгоритм, все этапы которого выполняются однократно и строго последовательно. Разветвляющийся алгоритм - алгоритм, содержащий хотя бы одно условие, в результате проверки которого ЭВМ обеспечивает переход на один из нескольких возможных шагов.

6. Что такое условный оператор? Какие существуют его формы? Оператор, конструкция языка программирования, обеспечивающая выполнение определённой команды (набора команд) только при условии истинности некоторого логического выражения, либо выполнение одной из нескольких команд. Условный оператор имеет полную и краткую формы.

7. Какие операторы сравнения используются в Python? If, elif, else

8. Что называется простым условием? Приведите примеры. Простым условием называется выражение, составленное из двух арифметических выражений или двух текстовых величин. Пример: a == b

9. Что такое составное условие? Приведите примеры. Составное условие – логическое выражение, содержащее несколько простых условий объединенных логическими операциями. Это операции not, and, or. Пример: (a == b or a == c)

10. Какие логические операторы допускаются при составлении сложных условий? not, and, or.

11. Может ли оператор ветвления содержать внутри себя другие ветвления? Может.

12. Какой алгоритм является алгоритмом циклической структуры? Циклический алгоритм — это вид алгоритма, в процессе выполнения которого одно или несколько действий нужно повторить.

13. Типы циклов в языке Python. В Python есть 2 типа циклов: - цикл while, - цикл for.

14. Назовите назначение и способы применения функции range. Функция range генерирует серию целых чисел, от значения start до stop, указанного пользователем. Мы можем использовать его для цикла for и обходить весь диапазон как список.

15. Как с помощью функции range организовать перебор значений от 15 до 0 с шагом 2? range(15, 0, 2)

16. Могут ли быть циклы вложенными? Могут.

17. Как образуется бесконечный цикл и как выйти из него? Бесконечный цикл в программировании — цикл, написанный таким образом, что условие выхода из него никогда не выполняется.

18. Для чего нужен оператор break? Используется для выхода из цикла.

19. Где употребляется оператор continue и для чего он используется? Оператор continue используется только в циклах. В операторах for , while , do while , оператор continue выполняет пропуск оставшейся части кода тела цикла и переходит к следующей итерации цикла.

20. Для чего нужны стандартные потоки stdout и stderr? Ввод и вывод распределяется между тремя стандартными потоками: stdin — стандартный ввод (клавиатура), stdout — стандартный вывод (экран), stderr — стандартная ошибка (вывод ошибок на экран)

21. Как в Python организовать вывод в стандартный поток stderr? Указать в print(…, file=sys.stderr).

22. Каково назначение функции exit? Функция exit() модуля sys - выход из Python. Вывод: исследовали процесс установки и базовые возможности языка Python версии 3.