

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 №6**  
**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

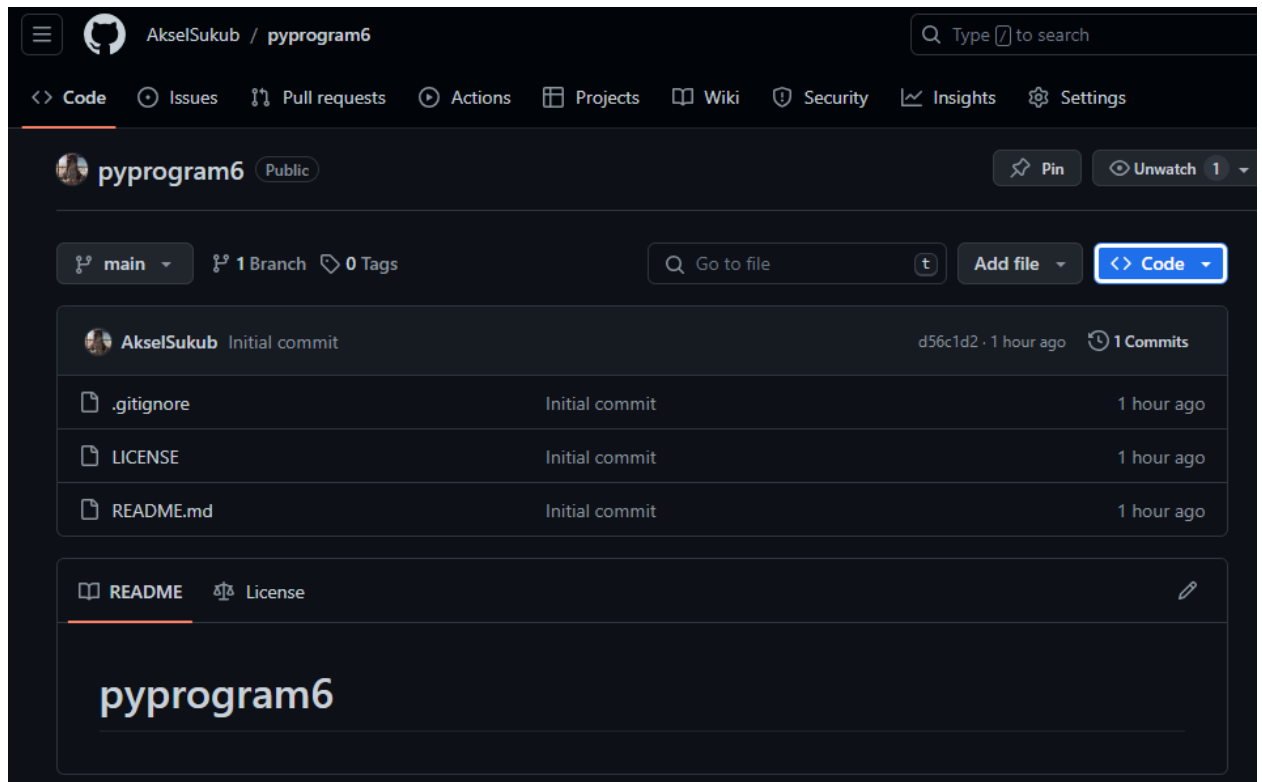


Рисунок 1. Созданный репозиторий

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

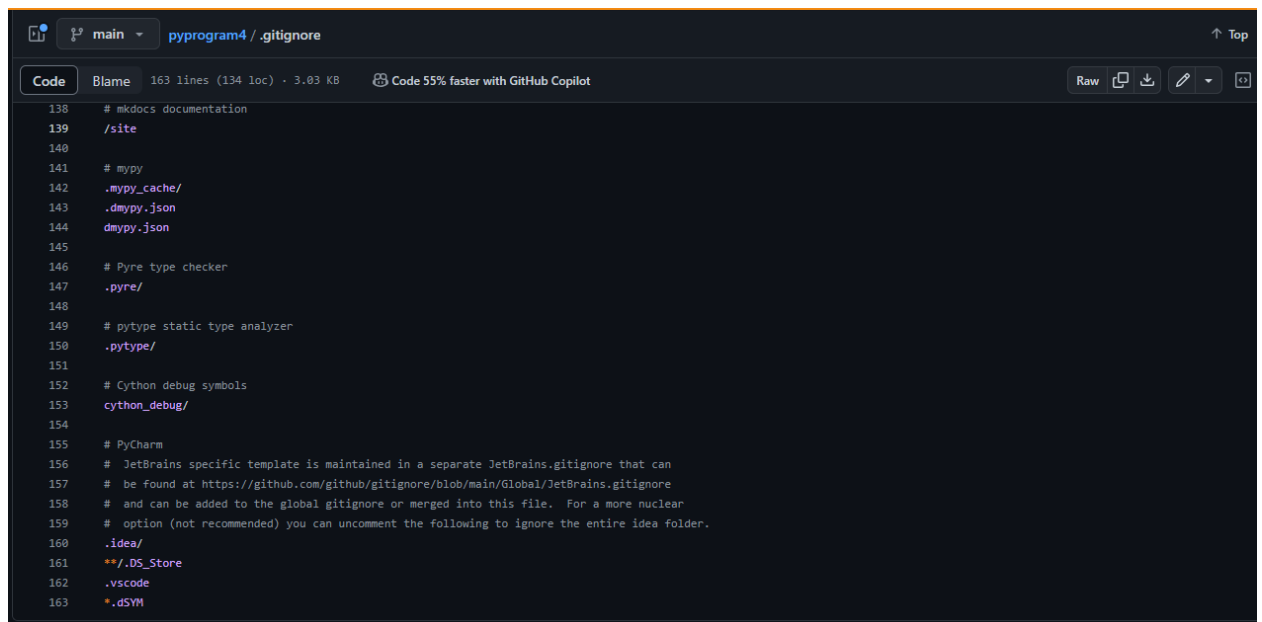


Рисунок 2. Измененный файл .gitignore

3. Выполнил задания

23. Дано предложение. В нем слова разделены одним или несколькими пробелами (символ «-» в предложении отсутствует). Определить количество слов в предложении. Рассмотреть два случая:

- начальные и конечные пробелы в предложении отсутствуют;
- начальные и конечные пробелы в предложении имеются.

```
zadanie1.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  def count_words(sentence):
5      words = sentence.split()
6      return len(words)
7
8  if __name__ == "__main__":
9      input_sentence = input("Введите предложение: ")
10     word_count = count_words(input_sentence)
11     print(f"Количество слов в предложении: {word_count}")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\1\pyprogram6\prog> & "C:/Program Files/Python311/python.exe" zadanie1.py
Введите предложение: Первое предложение для проверки
Количество слов в предложении: 4
PS C:\Users\1\pyprogram6\prog> & "C:/Program Files/Python311/python.exe" zadanie1.py
Введите предложение: Второе предложение с пробелами
Количество слов в предложении: 4
PS C:\Users\1\pyprogram6\prog> 
```

Рисунок 3 . Выполнение 1 задания

23. Дано слово. Поменять местами его m-ю и n-ю буквы.

```
zadanie2.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  def swap_letters(word, m, n):
5      if 0 <= m < len(word) and 0 <= n < len(word):
6          word_list = list(word)
7          word_list[m], word_list[n] = word_list[n], word_list[m]
8          return ''.join(word_list)
9      else:
10         print("Ошибка: Неверные значения m и n.")
11         return word
12
13 if __name__ == "__main__":
14     input_word = input("Введите слово: ")
15     index_m = int(input("Введите индекс m: "))
16     index_n = int(input("Введите индекс n: "))
17
18     result_word = swap_letters(input_word, index_m, index_n)
19     print(f"Слово после замены: {result_word}")
```

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

```
PS C:\Users\1\pyprogram6\prog> & "C:/Program Files/Python311/python.exe" c:/User
Введите слово: Слово
Введите индекс m: 2
Введите индекс n: 3
Слово после замены: Слвоо
```

Рисунок 4 . Выполнение 2 задания

Дано ошибочно написанное слово килбайот. Путем перемещения его букв получить слово килобайт.

```
zadanie3.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  def fix_spelling(word):
5      if "й" in word and "б" in word and "ай" in word and "т" in word:
6          # Перемещаем буквы для получения "килобайт"
7          corrected_word = word.replace("й", "").replace("б", "", 1).replace("ай", "й").replace("т", "байт")
8          return corrected_word
9      else:
10         print("Ошибка: в слове отсутствуют необходимые буквы.")
11         return word
12
13 if __name__ == "__main__":
14     misspelled_word = "килбайот"
15     corrected_word = fix_spelling(misspelled_word)
16     print(f"Исправленное слово: {corrected_word}")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Исправленное слово: килобайт

Рисунок 5 . Выполнение 3 задания

Дано предложение. Напечатать все слова, которые встречаются в нем по одному разу.

```
zadanieps.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  def print_unique_words(sentence):
5      words = sentence.split()
6      word_count = {}
7
8      for word in words:
9          # Приводим слово к нижнему регистру, чтобы исключить различие в регистре
10         word_lower = word.lower()
11
12         # Обновляем счетчик для каждого слова
13         word_count[word_lower] = word_count.get(word_lower, 0) + 1
14
15         # Выводим слова, которые встречаются только один раз
16         unique_words = [word for word, count in word_count.items() if count == 1]
17         print("Слова, встречающиеся по одному разу:", unique_words)
18
19 # Ввод предложения с клавиатуры
20 user_sentence = input("Введите предложение: ")
21 print_unique_words(user_sentence)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\1\pyprogram6\prog> & "C:/Program Files/Python311/python.exe" c:/Users/1/pyprogram6/prog/zadanieps.py
Введите предложение: Предложение проверки предложение которое предложение
Слова, встречающиеся по одному разу: ['проверки', 'которое']
PS C:\Users\1\pyprogram6\prog> █
```

Рисунок 6 . Выполнение усложненного индивидуального задания

Ответы на контрольные вопросы:

1. **Что такое строки в языке Python?**

Строки в Python представляют собой последовательность символов и используются для хранения текстовой информации.

2. **Какие существуют способы задания строковых литералов в языке Python?**

Строковые литералы можно задавать с использованием одинарных (' '), двойных (" "), или тройных ("\""\" или \"\" \"\") кавычек.

3. **Какие операции и функции существуют для строк?**

Существуют различные операции, такие как конкатенация (+), умножение (\*), методы для изменения регистра (upper(), lower()), получения длины (len()), извлечения подстрок (substring()), и многие другие.

#### **4. Как осуществляется индексирование строк?**

Строки индексируются, начиная с 0. Доступ к отдельным символам осуществляется с использованием квадратных скобок, например, `my_string[0]` вернет первый символ строки.

#### **5. Как осуществляется работа со срезами для строк?**

Срезы в Python создаются с использованием оператора `:`. Например, `my_string[1:4]` вернет подстроку, начиная со второго символа до четвёртого (не включая).

#### **6. Почему строки Python относятся к неизменяемому типу данных?**

Строки являются неизменяемыми, потому что после создания строки её нельзя изменить напрямую. Операции над строками создают новые строки.

#### **7. Как проверить то, что каждое слово в строке начинается с заглавной буквы?**

Можно воспользоваться методом `istitle()`.

#### **8. Как проверить строку на вхождение в неё другой строки?**

Используйте оператор `in`: `substring in my_string`.

#### **9. Как найти индекс первого вхождения подстроки в строку?**

Используйте метод `find()`.

**10. Как подсчитать количество символов в строке?**

Используйте функцию `len()`.

**11. Как подсчитать то, сколько раз определённый символ встречается в строке?**

Воспользуйтесь методом `count()`.

**12. Что такое f-строки и как ими пользоваться?**

F-строки представляют собой строковые литералы, начинающиеся с буквы 'f' или 'F'. Они позволяют встраивать значения переменных и выражений в строку.

**13. Как найти подстроку в заданной части строки?**

Используйте метод `find()` с указанием диапазона индексов.

**14. Как вставить содержимое переменной в строку, воспользовавшись методом `format()`?**

Пример: `"Привет, {}".format(name)`.

**15. Как узнать о том, что в строке содержатся только цифры?**

Используйте методы `isdigit()` или `isnumeric()`.

**16. Как разделить строку по заданному символу?**

Используйте метод `split()`.

**17. Как проверить строку на то, что она составлена только из строчных букв?**

Используйте метод `islower()`.



**18. Как проверить то, что строка начинается со строчной буквы?**

Используйте метод `str.startswith()`.

**19. Можно ли в Python прибавить целое число к строке?**

Нет, так как строки неизменяемы. Операции над строками создают новые строки.

**20. Как «перевернуть» строку?**

Используйте срезы: `reversed_string = my_string[::-1]`.

**21. Как объединить список строк в одну строку, элементы которой разделены дефисами?**

Используйте метод `join()`: `result = "-".join(my_list)`.

**22. Как привести всю строку к верхнему или нижнему регистру?**

Используйте методы `upper()` или `lower()`.

**23. Как преобразовать первый и последний символы строки к верхнему регистру?**

Используйте методы `capitalize()` и `title()`.

**24. Как проверить строку на то, что она составлена только из прописных букв?**

Используйте метод `isupper()`.

**25. В какой ситуации вы воспользовались бы методом `splitlines()`?**

Метод `splitlines()` используется для разделения строки по символам новой строки (`\n`) и возвращения списка строк.

**26. Как в заданной строке заменить на что-либо все вхождения некоей подстроки?**

Используйте метод `replace()`.

**27. Как проверить то, что строка начинается с заданной последовательности символов, или заканчивается заданной последовательностью символов?**

Используйте методы `startswith()` и `endswith()`.

**28. Как узнать о том, что строка включает в себя только пробелы?**

Используйте метод `isspace()`.