

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Кафедра инфокоммуникаций

Основы кроссплатформенного программирования

Отчет по лабораторной работе №2.3

Тема: «Работа со строками в языке Python»

Выполнил студент группы

ИВТ-б-о-21-1

Богадунов В.И. « » _____ 20__ г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил доцент

Кафедры инфокоммуникаций, старший
преподаватель

Воронкин Р.А.

(подпись)

Ставрополь 2022

1. Создал репозиторий в GitHub, дополнил правила в .gitignore для работы с IDE PyCharm с ЯП Python, выбрал лицензию MIT, клонировал его на компьютер и организовал в соответствии с моделью ветвления git-flow.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner *



ItsMyLife1337 ▾

Repository name *

ForPrograms2.3 ✓

Great repository names are short and memorable. Need inspiration? How about [legendary-winner?](#)

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☐ Add a README file

This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: Python ▾

Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

License: MIT License ▾

You are creating a public repository in your personal account.

Рисунок 1.1 – Создание репозитория

```
275 lines (219 sloc) | 5.56 KB
1 # Created by https://www.toptal.com/developers/gitignore/api/python,pycharm
2 # Edit at https://www.toptal.com/developers/gitignore?templates=python,pycharm
3
4 ### PyCharm ###
5 # Covers JetBrains IDEs: IntelliJ, RubyMine, PhpStorm, AppCode, PyCharm, CLion, Android Studio, WebStorm and Rider
6 # Reference: https://intellij-support.jetbrains.com/zh-cn-us/articles/206544839
7
8 # User-specific stuff
9 .idea/**/workspace.xml
10 .idea/**/tasks.xml
11 .idea/**/usage.statistics.xml
12 .idea/**/dictionaries
13 .idea/**/shelf
14
15 # AWS User-specific
16 .idea/**/aws.xml
17
18 # Generated files
19 .idea/**/contentModel.xml
20
21 # Sensitive or high-churn files
22 .idea/**/dataSources/
23 .idea/**/dataSources.ids
```

Рисунок 1.2 – Изменённый файл .gitignore

```

C:\Users\Admin>cd /d c:\users\admin\desktop\git

c:\Users\Admin\Desktop\git>git clone https://github.com/ItsMyLife1337/ForPrograms2.3.git
Cloning into 'ForPrograms2.3'...
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 7 (delta 1), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (7/7), 4.37 KiB | 447.00 KiB/s, done.
Resolving deltas: 100% (1/1), done.

c:\Users\Admin\Desktop\git>

```

Рисунок 1.3 – Клонирование репозитория на компьютер

```

c:\Users\Admin\Desktop\git>cd /d c:\users\admin\desktop\git\forprograms2.3

c:\Users\Admin\Desktop\git\ForPrograms2.3>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/Admin/Desktop/git/ForPrograms2.3/.git/hooks]

```

Рисунок 1.4 – Организация репозитория в соответствии с моделью ветвления git-flow

2. Создал проект PyCharm в папке репозитория, проработал примеры ЛР.

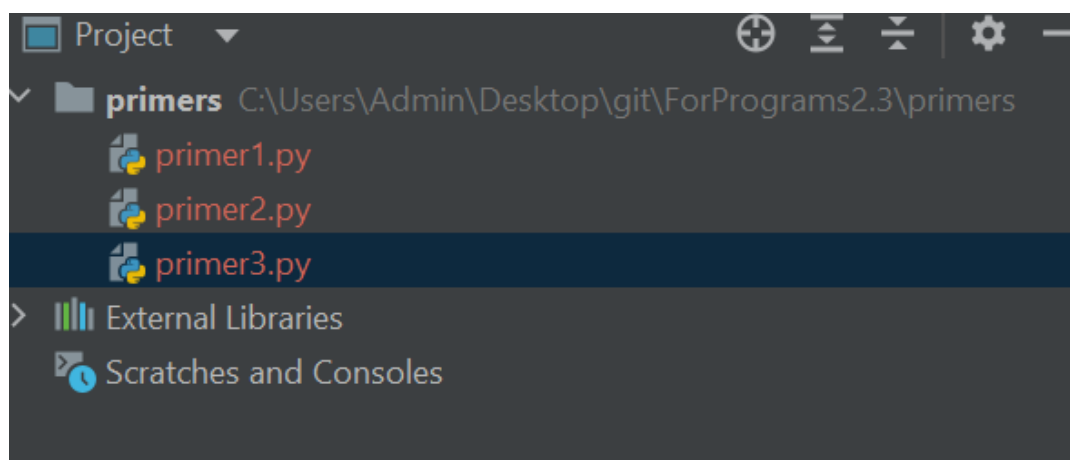


Рисунок 2.1 – Созданные проекты

```

C:\Users\Admin\AppData\Local\Programs\Python\Python39\p
Введите предложение: ahyhyahyahu hyahyahaha hyahya
Предложение после замены: ahyhyahyahu_hyahyahaha_hyahya

Process finished with exit code 0

```

Рисунок 2.2 – Результат выполнения примера №1

```

primer2 x
C:\Users\Admin\AppData\L
Введите слово: lokek
loek

```

Рисунок 2.3 – Результат выполнения примера №2

```

primer3 x
C:\Users\Admin\AppData\Local\Programs\Python\F
Введите предложение: ahuha hahy ahyhy ha
Введите длину: 26
ahuha hahy ahyhy ha

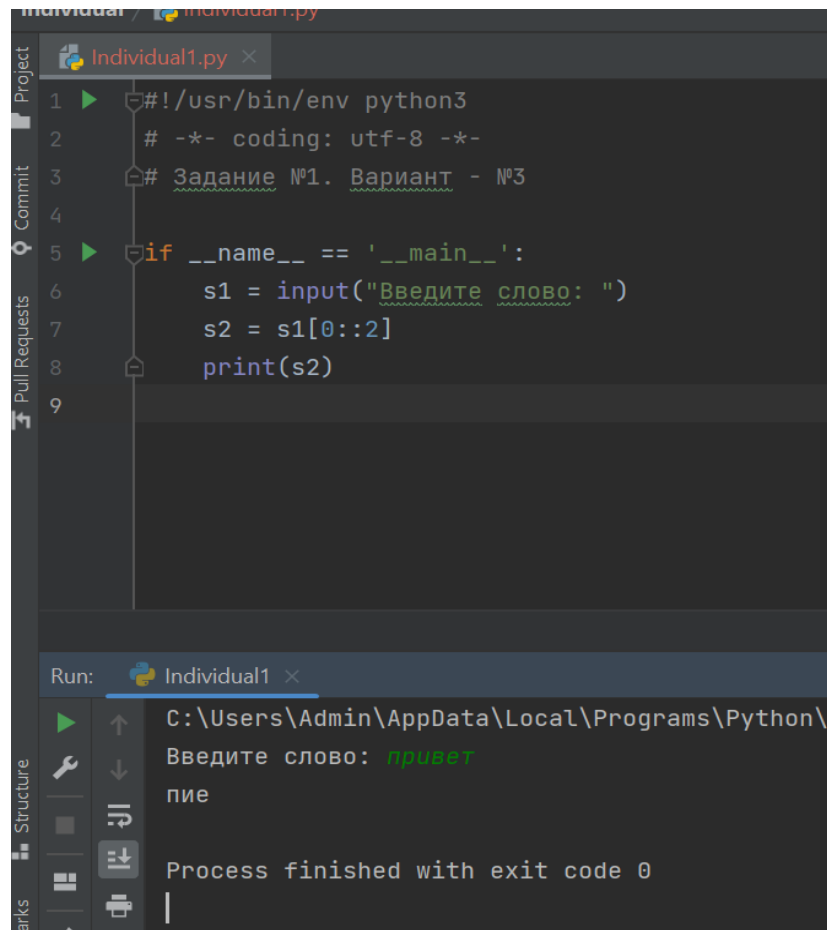
Process finished with exit code 0

```

Рисунок 2.4 – Результат выполнения примера №3

3. Выполнил 3 индивидуальных задания и 1 усложнённое. Вариант – №3.

Задание №1. В - 3. Дано слово S1. Получить слово S2, образованное нечетными буквами слова S1.



The screenshot shows an IDE window with a file named 'Individual1.py'. The code is as follows:

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 # Задание №1. Вариант - №3
4
5 if __name__ == '__main__':
6     s1 = input("Введите слово: ")
7     s2 = s1[0::2]
8     print(s2)
9
```

Below the code editor, the 'Run' console shows the execution path and output:

```
Run: C:\Users\Admin\AppData\Local\Programs\Python\Python38\python.exe
      Введите слово: привет
      пие
      Process finished with exit code 0
```

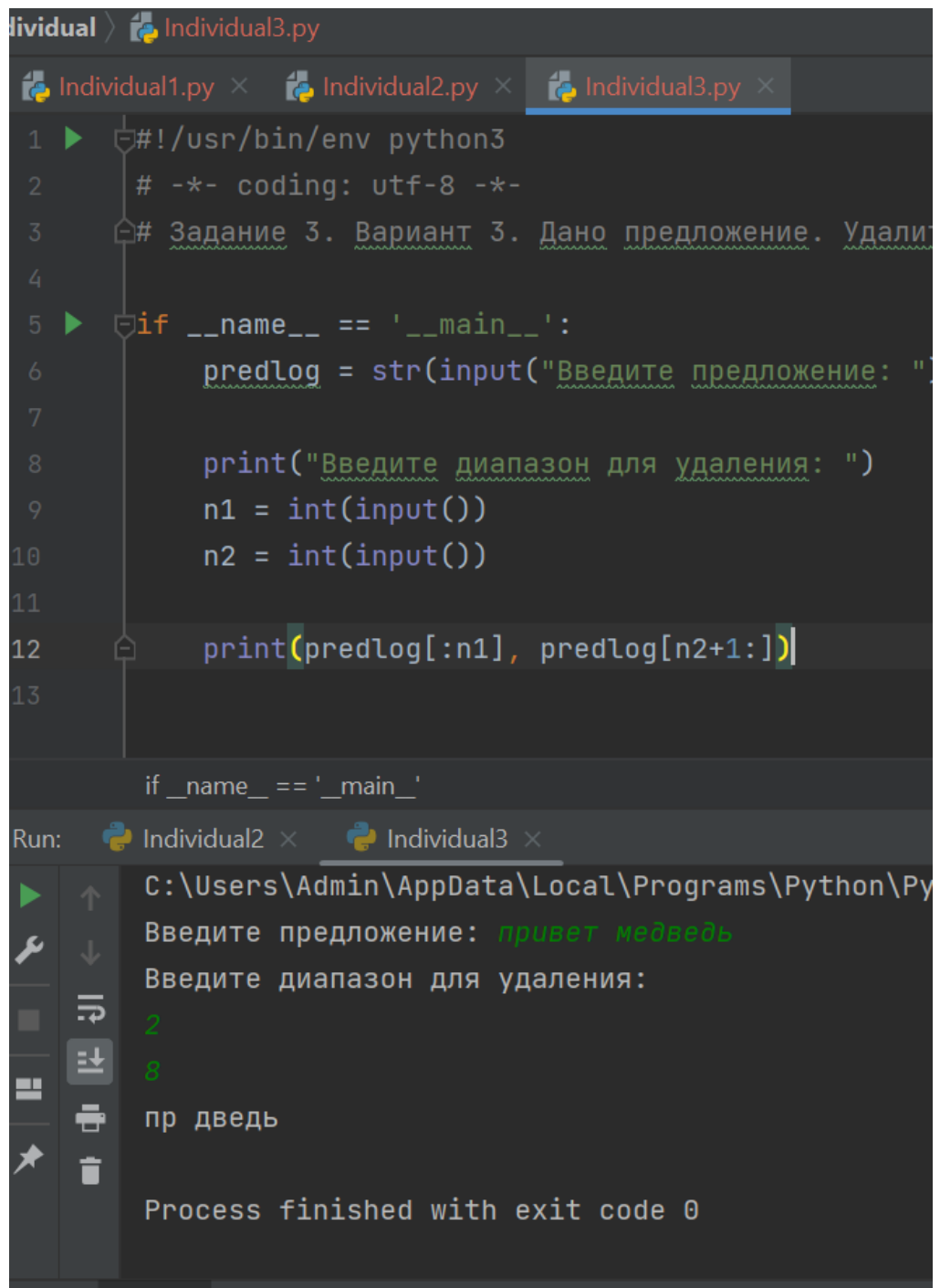
Рисунок 3.1 – Выполненное индивидуальное задание №1

Задание №2. В – 3. Дан текст. Определить количество букв и в первом предложении. Рассмотреть два случая: известно, что буквы и в этом предложении есть; букв и в тексте может не быть.

```
Individual1.py x Individual2.py x
4 Дан текст. Определить количество букв и в первом предл
5 -известно, что буквы и в этом предложении есть;
6 -букв и в тексте может не быть.
7 """
8
9 if __name__ == '__main__':
10     predlog = input("Введите предложение: ")
11     col = 0
12     i = 1
13
14     for i in range(len(predlog)):
15         if predlog[i] == 'и':
16             col = col + 1
17         i = i + 1
18
19 if __name__ == '__main__':
20     pass
21
22 un: Individual2 x
23 C:\Users\Admin\AppData\Local\Programs\Python\Python39
24 Введите предложение: Привет ива иващенко ива
25 Букв 'и' в предложении: 4
26
27 Process finished with exit code 0
28 |
```

Рисунок 3.2 – Выполненное индивидуальное задание №2

Задание №3. В – 3. Дано предложение. Удалить из него все символы с n1-го по n2-й ($n1 \leq n2$).



```
Individual > Individual3.py
Individual1.py x Individual2.py x Individual3.py x
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 # Задание 3. Вариант 3. Дано предложение. Удали
4
5 if __name__ == '__main__':
6     predlog = str(input("Введите предложение: "))
7
8     print("Введите диапазон для удаления: ")
9     n1 = int(input())
10    n2 = int(input())
11
12    print(predlog[:n1], predlog[n2+1:])
13
if __name__ == '__main__'
```

Run: Individual2 x Individual3 x

C:\Users\Admin\AppData\Local\Programs\Python\Py

Введите предложение: *привет медведь*

Введите диапазон для удаления:

2

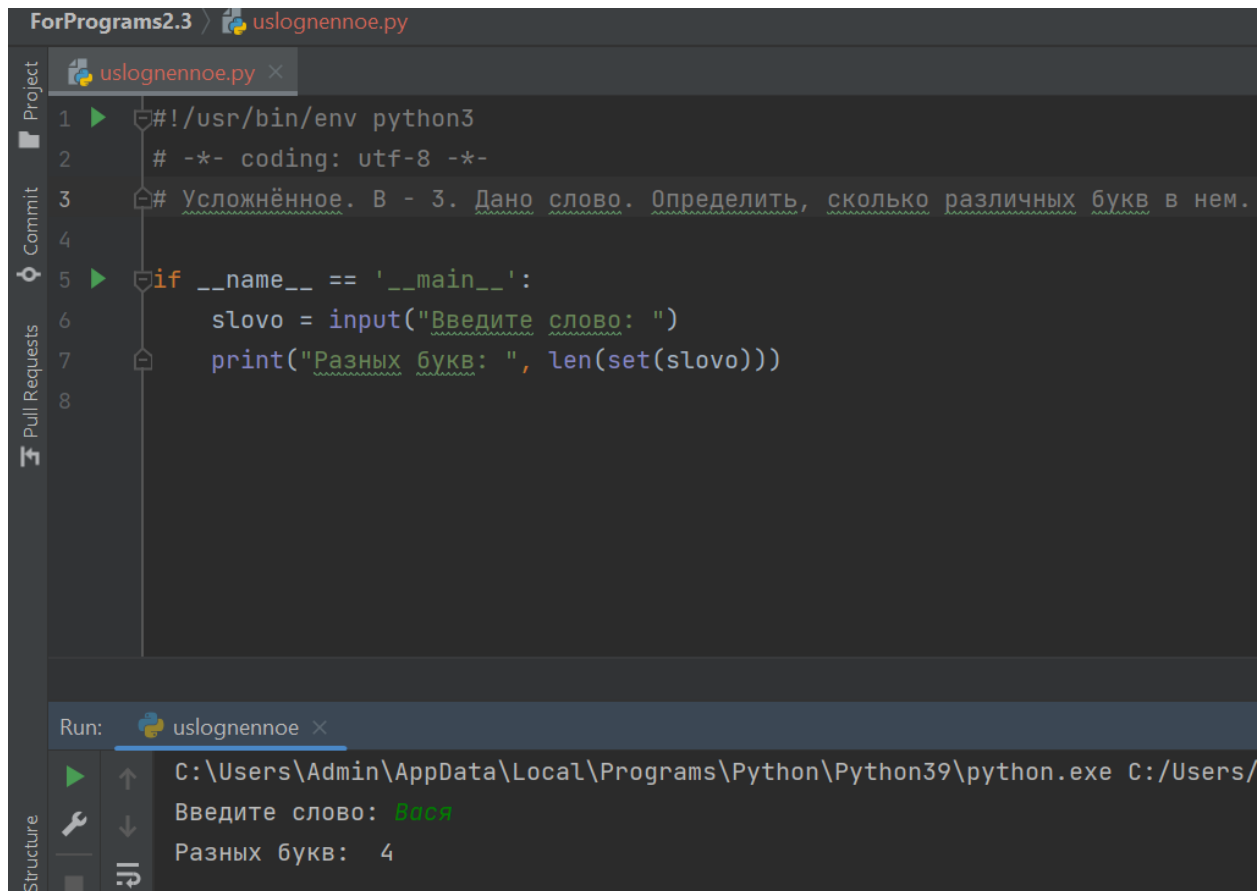
8

пр дведь

Process finished with exit code 0

Рисунок 3.3 – Индивидуальное задание №3

Задание повышенной сложности. В – 3. Дано слово. Определить, сколько различных букв в нем.



The screenshot shows an IDE window titled 'ForPrograms2.3' with a file named 'uslognennoe.py'. The code in the editor is as follows:

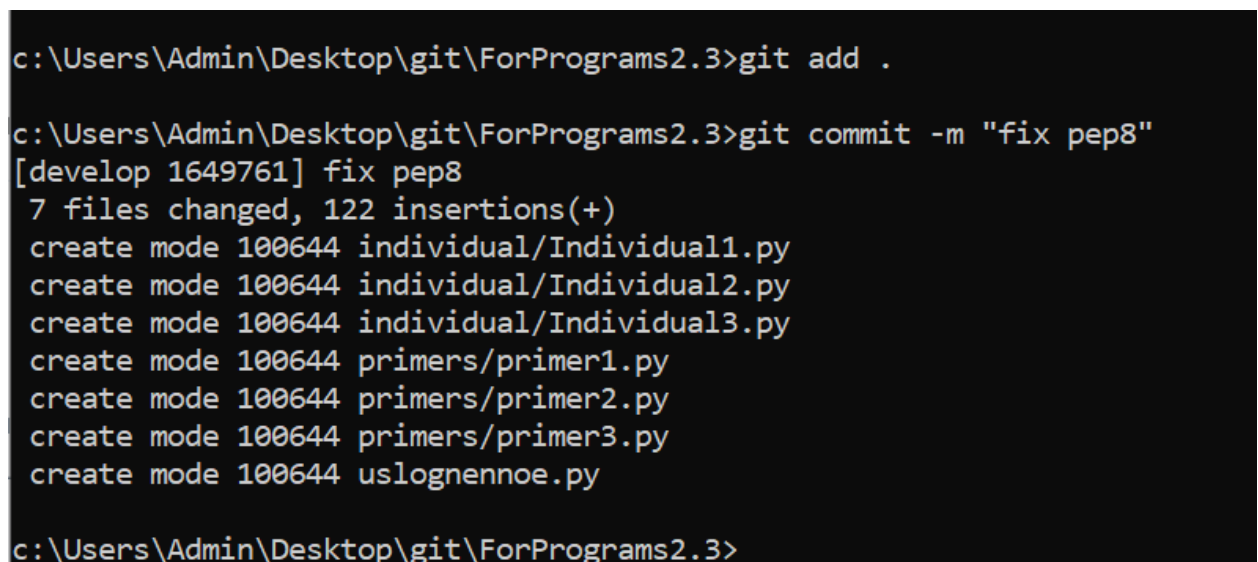
```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 # Усложнённое. В - 3. Дано слово. Определить, сколько различных букв в нем.
4
5 if __name__ == '__main__':
6     slovo = input("Введите слово: ")
7     print("Разных букв: ", len(set(slovo)))
8
```

Below the editor, the 'Run' panel shows the execution command and output:

```
Run: python3 uslognennoe.py
C:\Users\Admin\AppData\Local\Programs\Python\Python39\python.exe C:/Users/
Введите слово: Вася
Разных букв: 4
```

Рисунок 3.4 – Задание повышенной сложности В – 3

4. Сделал коммит всех изменений, выполнил слияние с веткой main.



The screenshot shows a terminal window with the following commands and output:

```
c:\Users\Admin\Desktop\git\ForPrograms2.3>git add .
c:\Users\Admin\Desktop\git\ForPrograms2.3>git commit -m "fix pep8"
[develop 1649761] fix pep8
7 files changed, 122 insertions(+)
create mode 100644 individual/Individual1.py
create mode 100644 individual/Individual2.py
create mode 100644 individual/Individual3.py
create mode 100644 primers/primer1.py
create mode 100644 primers/primer2.py
create mode 100644 primers/primer3.py
create mode 100644 uslognennoe.py
c:\Users\Admin\Desktop\git\ForPrograms2.3>
```

Рисунок 4.1 – Коммит всех изменений


```

c:\Users\Admin\Desktop\git\ForPrograms2.3>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

c:\Users\Admin\Desktop\git\ForPrograms2.3>git merge develop
Updating df90810..1649761
Fast-forward
 individual/Individual1.py | 8 ++++++
 individual/Individual2.py | 19 ++++++
 individual/Individual3.py | 12 ++++++
 primers/primer1.py       | 7 +++++
 primers/primer2.py       | 13 ++++++
 primers/primer3.py       | 56 ++++++
 uslognennoe.py           | 7 +++++
 7 files changed, 122 insertions(+)
 create mode 100644 individual/Individual1.py
 create mode 100644 individual/Individual2.py
 create mode 100644 individual/Individual3.py
 create mode 100644 primers/primer1.py
 create mode 100644 primers/primer2.py
 create mode 100644 primers/primer3.py
 create mode 100644 uslognennoe.py

c:\Users\Admin\Desktop\git\ForPrograms2.3>

```

Рисунок 4.2 – Слил ветку develop с веткой main

```

c:\Users\Admin\Desktop\git\ForPrograms2.3>git push
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 4 threads
Compressing objects: 100% (11/11), done.
Writing objects: 100% (11/11), 2.79 KiB | 476.00 KiB/s, done.
Total 11 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/ItsMyLife1337/ForPrograms2.3.git
    df90810..1649761  main -> main

c:\Users\Admin\Desktop\git\ForPrograms2.3>

```

Рисунок 4.3 – Отправил изменения на удалённый репозиторий


main		1 branch	0 tags	Go to file	Add file	Code
<div>  <div>ItsMyLife1337 fix pep8</div> <div>1649761 6 minutes ago 3 commits</div> </div>						
individual		fix pep8	6 minutes ago			
primers		fix pep8	6 minutes ago			
.gitignore		Update .gitignore	6 hours ago			
LICENSE		Initial commit	6 hours ago			
uslognennoe.py		fix pep8	6 minutes ago			

Рисунок 4.4 – Зафиксировал изменения

Ответы на контрольные вопросы:

1. Что такое строки в языке Python?

Строки в Python - упорядоченные последовательности символов, используемые для хранения и представления текстовой информации, поэтому с помощью строк можно работать со всем, что может быть представлено в текстовой форме.

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

Строки в апострофах и в кавычках, экранированные последовательности, "сырые" строки, строки в тройных апострофах или кавычках

3. Какие операции и функции существуют для строк?

Сложение, дублирование, длина строки, извлечение среза и т. д.

4. Как осуществляется индексирование строк?

Доступ к символам в строках основан на операции индексирования – после строки или имени переменной, ссылающейся на строку, в квадратных скобках указываются номера позиций необходимых символов.

5. Как осуществляется работа со срезами для строк?

Есть три формы срезов. Самая простая форма среза: взятие одного символа строки, а именно, `S[i]` — это срез, состоящий из одного символа, который имеет номер `i`, при этом считая, что нумерация начинается с числа 0.

То есть если `S = 'Hello'`, то `S[0]=='H'`, `S[1]=='e'`, `S[2]=='l'`, `S[3]=='l'`, `S[4]=='o'`.

Если указать отрицательное значение индекса, то номер будет отсчитываться с конца, начиная с номера -1.

Срез с двумя параметрами: `S[a:b]` возвращает подстроку из `b-a` символов, начиная с символа с индексом `a`, то есть до символа с индексом `b`, не включая его.

6. Почему строки Python относятся к неизменяемому типу данных?

Строки — один из типов данных, которые Python считает неизменяемыми, что означает невозможность их изменять. Python дает возможность изменять (заменять и перезаписывать) строки.

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

```
string.istitle()
```

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

```
string.find()
```

9. Как найти индекс первого вхождения подстроки в строку?

```
s.partition(<sep>)
```

10. Как подсчитать количество символов в строке?

```
len(s)
```

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

```
s.count(<sub>)
```

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

Эти строки улучшают читаемость кода, а также работают быстрее чем другие способы форматирования. F-строки задаются с помощью литерала «f» перед кавычками. Пример: `print(f"Меня зовут {name} Мне {age} лет.")`

13. Как найти подстроку в заданной части строки?

`s.find(значение, начало, конец)`

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

`print('{}'.format(s))`

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

`s.isdigit()`

16. Как разделить строку по заданному символу?

`str.split()`

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

`s.isalpha()`

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

`s.istitle()`

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

Нет

20. Как «перевернуть» строку?

`s.reverse()`

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

`str.split('-')`

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

`s.upper()`

`s.lower`

23. Как преобразовать первый символ строки к верхнему регистру?

`s.capitalize()`

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

`s.isupper()`

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

`s.splitlines()` делит `s` на строки и возвращает их в списке. Любой из следующих символов или последовательностей символов считается границей строки.

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

`s.replace(old, new)`

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

`str.startswith()` и `str.endswith()`

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

`s.isspace()`

29. Что случится, если умножить некую строку на 3?

`Asd*3 = AsdAsdAsd`

30. Как привести к верхнему регистру первый символ каждого слова в строке?

`s.title()`

31. Как пользоваться методом `partition()`?

Метод `partition()` разбивает строку при первом появлении строки аргумента и возвращает кортеж, содержащий часть перед разделителем, строку аргумента и часть после разделителя.

32. В каких ситуациях пользуются методом `rfind()`?

`s.rfind(<sub>)` возвращает индекс последнего вхождения подстроки `<sub>` в `s`, который соответствует началу `<sub>`.

Вывод: научился работать со строками в python.