

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 №2.3
дисциплины «Основы кроссплатформенного программирования»

Выполнил:
Хачатрян Владимир Владимирович
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.

Работа со строками в языке Python.

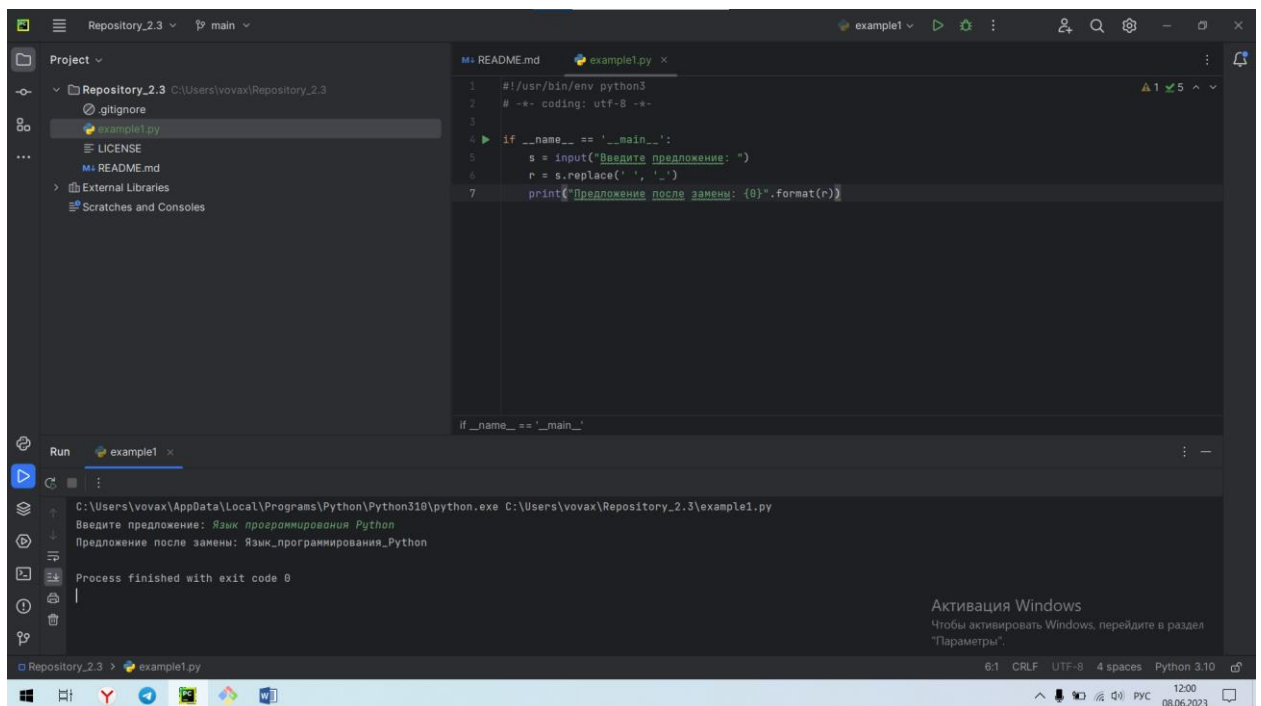
Цель: приобретение навыков по работе со строками при написании программ с помощью языка программирования Python версии 3.x.

Ход работы:

Создал общедоступный репозиторий на GitHub, добавил файл .gitignore с необходимыми правилами. Клонировал репозиторий на свой компьютер.

```
C:\Users\vovax>git clone https://github.com/GitVolodya/Repository_2.3.git
Cloning into 'Repository_2.3'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
```

Пример 1: Дано предложение. Все пробелы в нем заменить символом «_».



```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 if __name__ == '__main__':
5     s = input("Введите предложение: ")
6     r = s.replace(' ', '_')
7     print("Предложение после замены: {}".format(r))
```

Run example1

C:\Users\vovax\AppData\Local\Programs\Python\Python310\python.exe C:\Users\vovax\Repository_2.3\example1.py

Введите предложение: Язык программирования Python

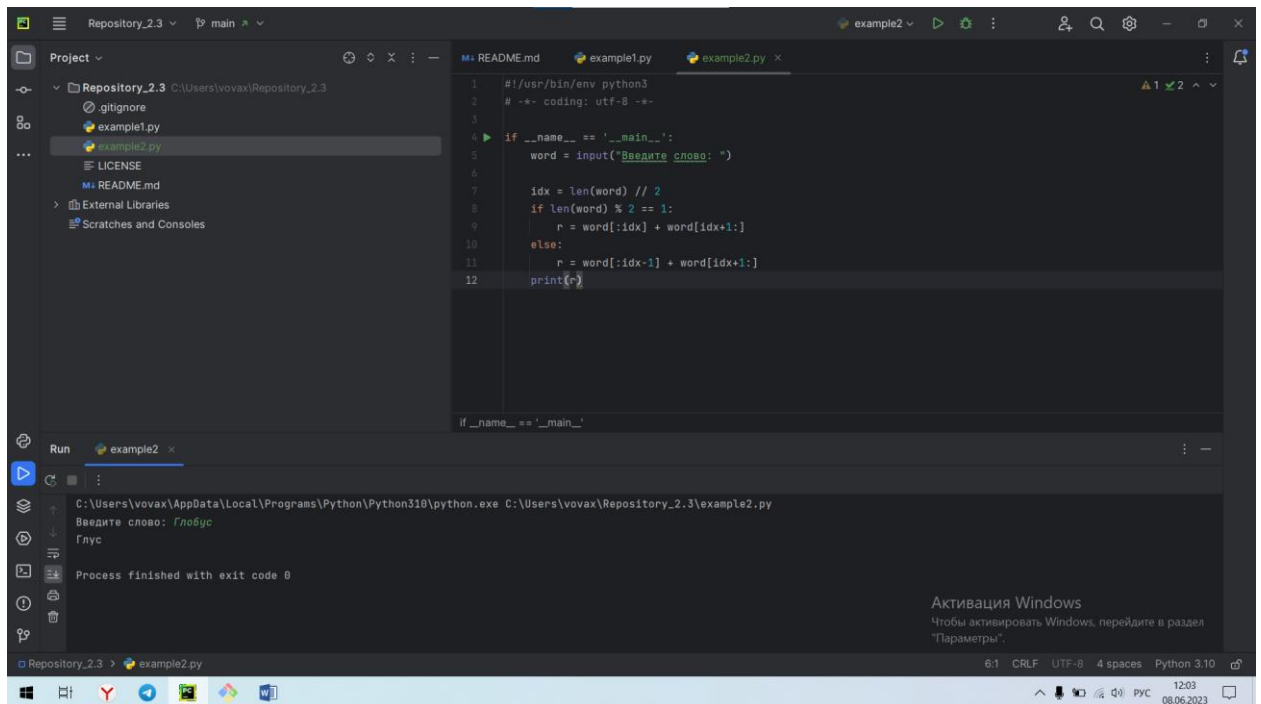
Предложение после замены: Язык_программирования_Python

Process finished with exit code 0

Активация Windows
Чтобы активировать Windows, перейдите в раздел "Параметры".

Repository_2.3 > example1.py 6:1 CRLF UTF-8 4 spaces Python 3.10 12:00 08.06.2023

Пример 2: дано слово. Если его длина нечетная, то удалить среднюю букву, в противном случае – две средние буквы.



```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 if __name__ == '__main__':
5     word = input("Введите слово: ")
6
7     idx = len(word) // 2
8     if len(word) % 2 == 1:
9         r = word[:idx] + word[idx+1:]
10     else:
11         r = word[:idx-1] + word[idx+1:]
12     print(r)
```

Run example2 x

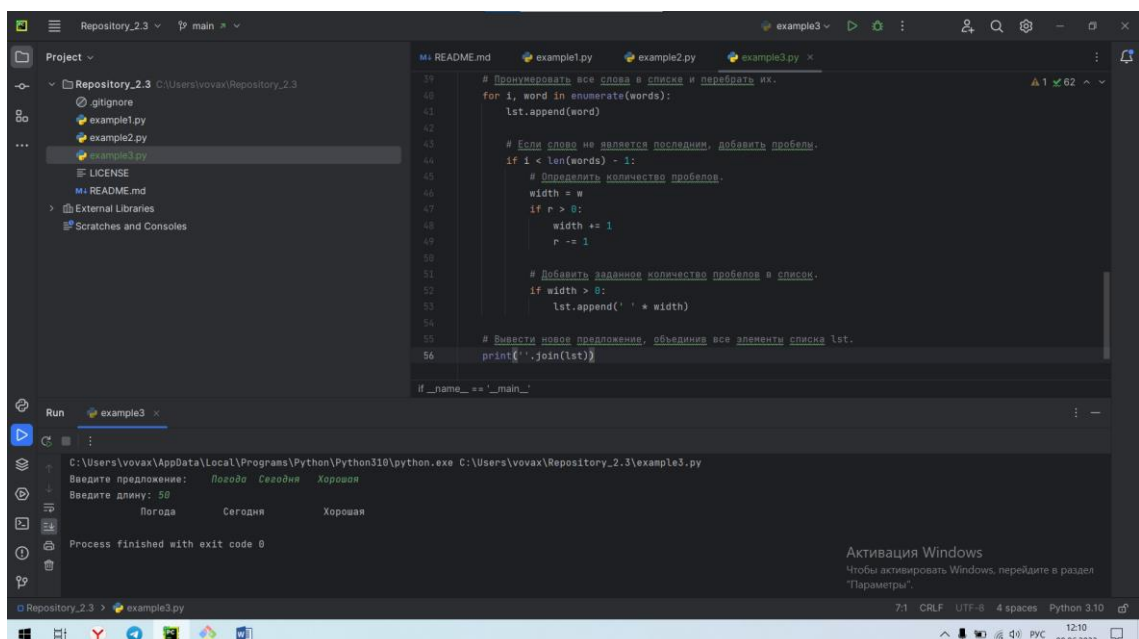
C:\Users\vovax\AppData\Local\Programs\Python\Python310\python.exe C:\Users\vovax\Repository_2.3\example2.py

Введите слово: Глобус

Глус

Process finished with exit code 0

Пример 3: Дана строка текста, в котором нет начальных и конечных пробелов. Необходимо изменить ее так, чтобы длина строки стала равна заданной длине (предполагается, что требуемая длина не меньше исходной). Это следует сделать путем вставки между словами дополнительных пробелов. Количество пробелов между отдельными словами должно отличаться не более чем на 1.



```
37 # Проинтерпретировать все слова в списке и перебрать их.
38 for i, word in enumerate(words):
39     lst.append(word)
40
41 # Если слово не является последним, добавить пробелы.
42 if i < len(words) - 1:
43     # Определить количество пробелов.
44     width = w
45     if r > 0:
46         width += 1
47     r -= 1
48
49 # Добавить заданное количество пробелов в список.
50 if width > 0:
51     lst.append(' ' * width)
52
53 # Вывести новое предложение, объединив все элементы списка lst.
54 print(' '.join(lst))
55
56 if __name__ == '__main__':
```

Run example3 x

C:\Users\vovax\AppData\Local\Programs\Python\Python310\python.exe C:\Users\vovax\Repository_2.3\example3.py

Введите предложение: Погода Сегодня Хорошая

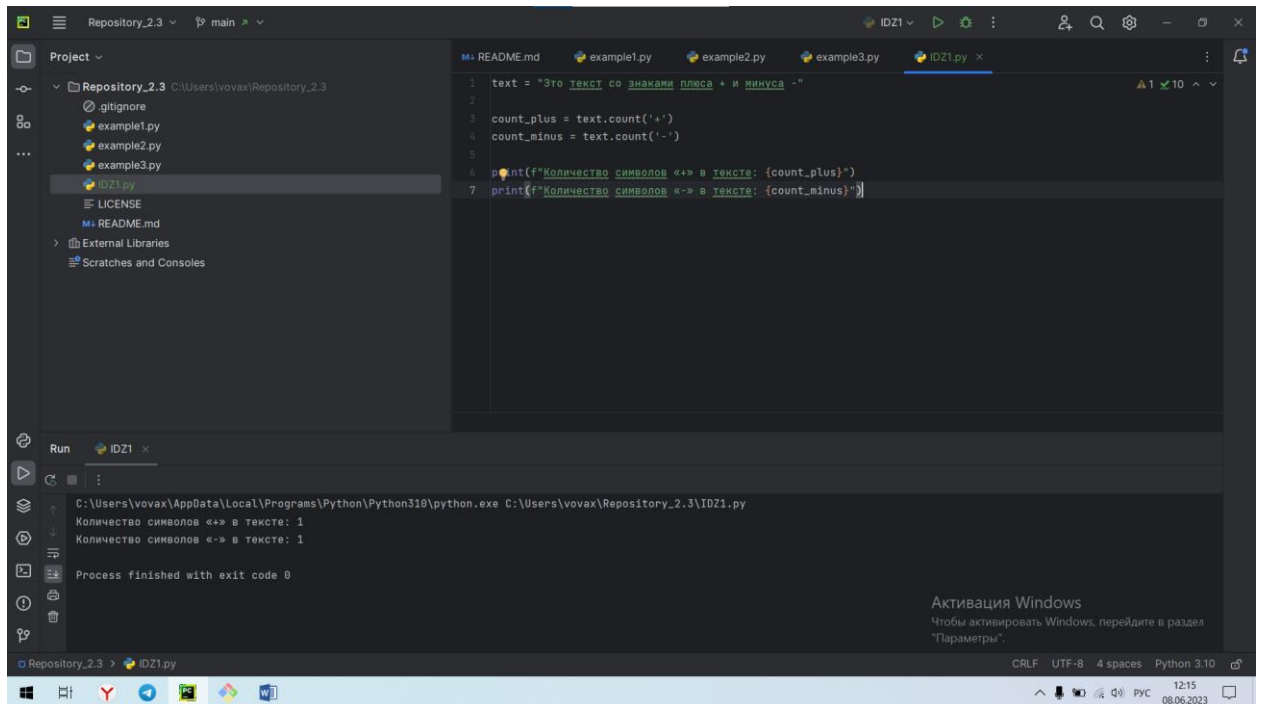
Введите длину: 50

Погода Сегодня Хорошая

Process finished with exit code 0

Индивидуальное задание 1

Условие: Дан текст. Подсчитать общее число вхождений в него СИМВОЛОВ «+» и «-».



The screenshot shows a Python IDE with a project named 'Repository_2.3'. The file explorer on the left shows files: .gitignore, example1.py, example2.py, example3.py, IDZ1.py, LICENSE, README.md, External Libraries, and Scratches and Consoles. The main editor displays the code for IDZ1.py:

```
1 text = "Это текст со знаками плюса + и минуса -"
2
3 count_plus = text.count('+')
4 count_minus = text.count('-')
5
6 print(f"Количество символов «+» в тексте: {count_plus}")
7 print(f"Количество символов «-» в тексте: {count_minus}")
```

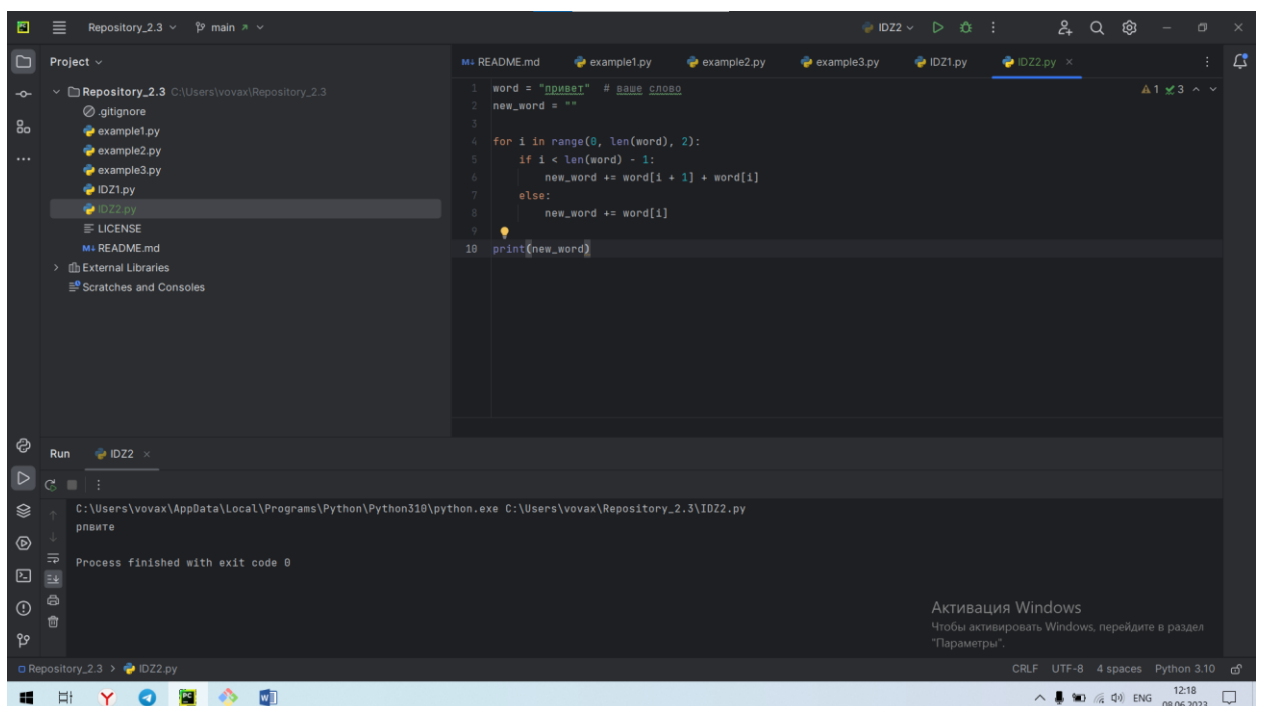
The Run window at the bottom shows the execution output:

```
C:\Users\vovax\AppData\Local\Programs\Python\Python310\python.exe C:\Users\vovax\Repository_2.3\IDZ1.py
Количество символов «+» в тексте: 1
Количество символов «-» в тексте: 1
Process finished with exit code 0
```

The status bar at the bottom indicates the file is in CRLF, UTF-8, 4 spaces, Python 3.10 format.

Индивидуальное задание 2

Условие: Дано слово из четного числа букв. Поменять местами первую букву со второй, третью – с четвертой и т. д.



The screenshot shows the same Python IDE with the file explorer showing the same files. The main editor displays the code for IDZ2.py:

```
1 word = "привет" # ваше слово
2 new_word = ""
3
4 for i in range(0, len(word), 2):
5     if i < len(word) - 1:
6         new_word += word[i + 1] + word[i]
7     else:
8         new_word += word[i]
9
10 print(new_word)
```

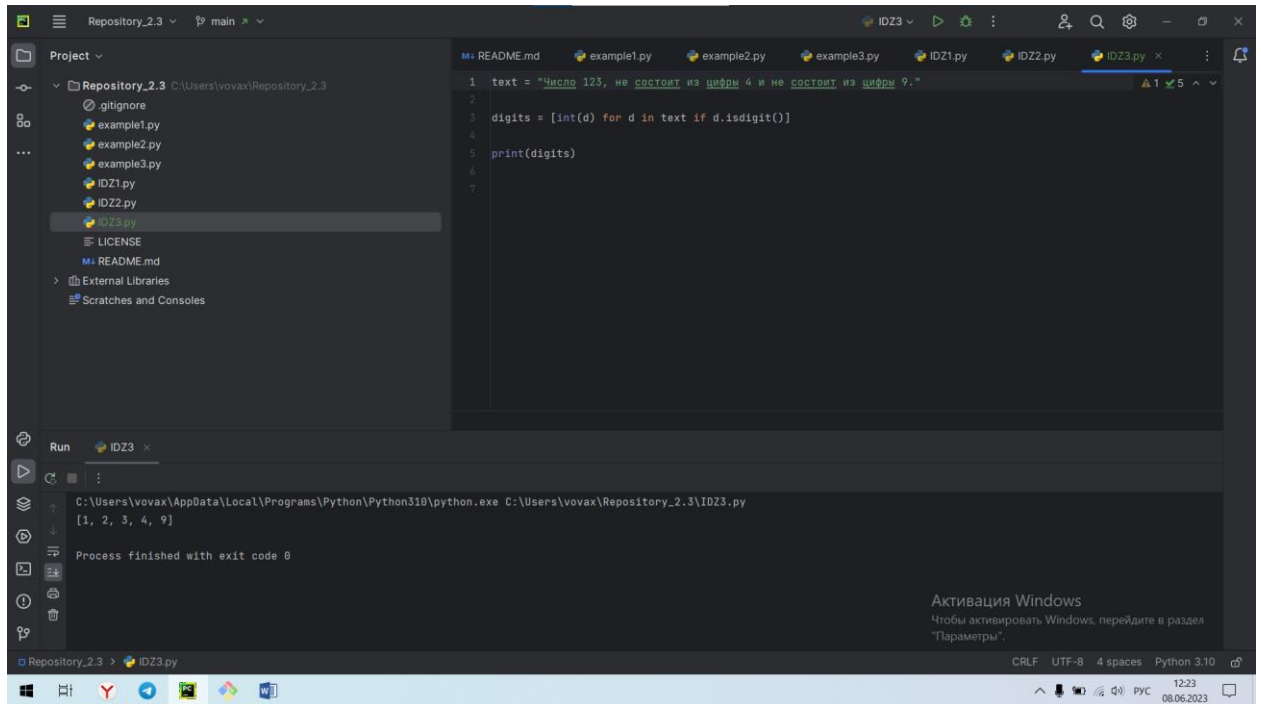
The Run window at the bottom shows the execution output:

```
C:\Users\vovax\AppData\Local\Programs\Python\Python310\python.exe C:\Users\vovax\Repository_2.3\IDZ2.py
рпвите
Process finished with exit code 0
```

The status bar at the bottom indicates the file is in CRLF, UTF-8, 4 spaces, Python 3.10 format.

Индивидуальное задание 3

Условие: Дан текст. Напечатать все имеющиеся в нём цифры.



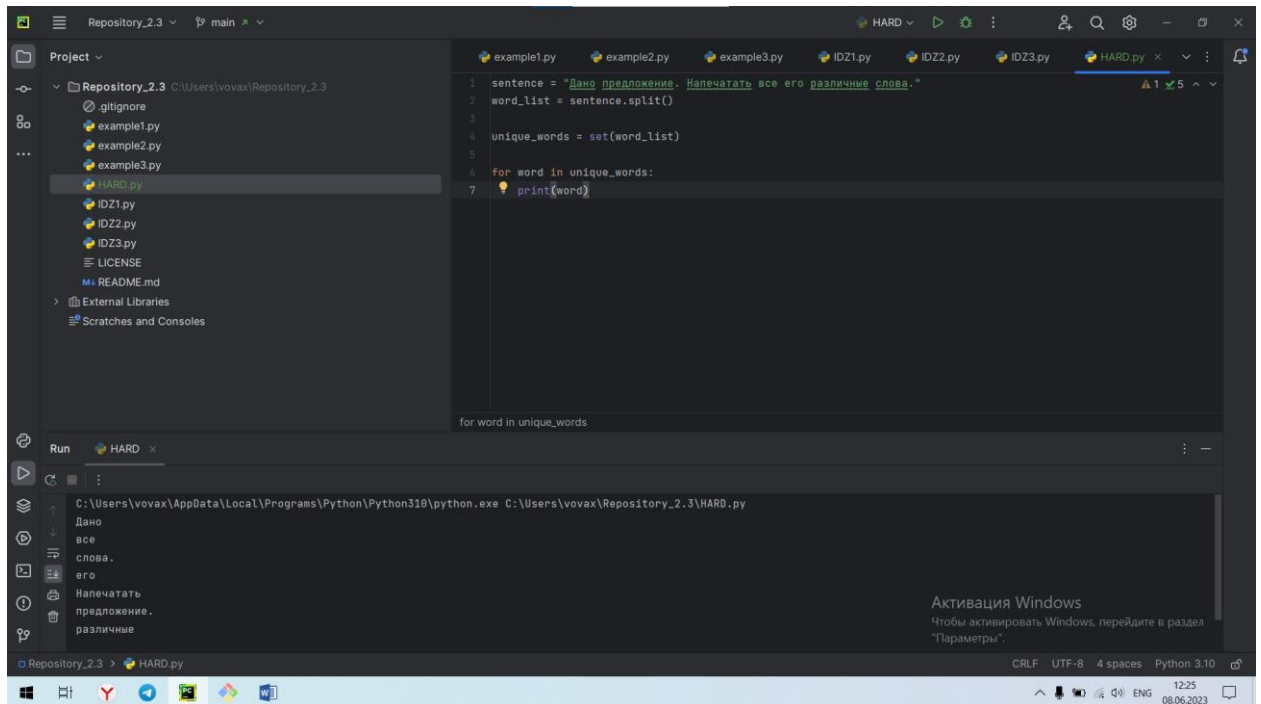
The screenshot shows the Visual Studio Code editor with a project named 'Repository_2.3'. The file explorer on the left shows several Python files, including 'IDZ3.py' which is selected. The editor displays the following code in 'IDZ3.py':

```
1 text = "Число 123, не состоит из цифр 4 и не состоит из цифр 9."
2
3 digits = [int(d) for d in text if d.isdigit()]
4
5 print(digits)
6
7
```

The Run and Debug console at the bottom shows the execution of the script, resulting in the output: [1, 2, 3, 4, 9]. The process finished with exit code 0.

Задание повышенной сложности

Условие: Дано предложение. Напечатать все его различные слова.



The screenshot shows the Visual Studio Code editor with the same project. The file explorer shows 'HARD.py' selected. The editor displays the following code in 'HARD.py':

```
1 sentence = "Дано предложение. Напечатать все его различные слова."
2 word_list = sentence.split()
3
4 unique_words = set(word_list)
5
6 for word in unique_words:
7     print(word)
```

The Run and Debug console shows the output of the script: Дано, предложение, Напечатать, все, его, слова, различные. The process finished with exit code 0.

Ссылка на репозиторий: https://github.com/GitVolodya/Repository_2.3.git

Ответы на контрольные вопросы:

1. *Что такое строки в языке Python?*

Строки в языке Python являются типом данных, специально предназначенным для обработки текстовой информации. Строка может содержать произвольно длинный текст (ограниченный имеющейся памятью). В новых версиях Python имеются два типа строк: обычные строки (последовательность байтов) и Unicode-строки (последовательность символов).

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

Литералы строк позволяют интерпретатору Python убедиться, что перед ним действительно находится строка. Такой подход называется "утиной" типизацией – если что-то плавает как утка, крикает как утка и откладывает яйца как утка, то скорее всего это действительно утка. То же самое и с литералами строк – если что-то соответствует литералам строк, то это можно считать строкой.

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

Для работы со строками существуют следующие стандартные процедуры и функции: `d:=copy (a, x, y)` функция, возвращает копию из `y` (`integer`) знаков части строки `a` (`string`), начиная с позиции `x` (`integer`). Формат: `d:=copy (a, x, y)`. Результат записывается в переменную `d`, например: `a='бегемот'; d:=copy (a,5,3); writeln (d);` результат - `мот` `delete (a, x, y)` процедура, удаляет `y` (`integer`) знаков части строки `a` (`string`), начиная с позиции `x` (`integer`).

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. Так же следует понимать, что этот самый доступ, основан на смещении, т.е. расстоянии символов от левого или правого края строки. Данное расстояние измеряется

целыми числами и, по сути, определяет номер позиции символов в строке – их индекс.

5. Как осуществляется работа со срезами для строк?

Есть три формы срезов: Самая простая форма среза - взятие одного символа строки - `S[i]`, где `S` строка, `i` - индекс.

Второй тип - срез с двумя параметрами. Т.е. `S[a:b]` возвращает подстроку, начиная с символа с индексом `a` до символа с индексом `b`, не включая его. Если опустить второй параметр (но поставить двоеточие), то срез берется до конца строки. Срез с тремя параметрами - `S[a:b:d]`.

Третий параметр задает шаг (как в случае с функцией `range`), то есть будут взяты символы с индексами `a`, `a + d`, `a + 2 * d` и т. д. Например, при задании значения третьего параметра, равному 2, в срез попадет каждый второй символ.

6. Почему строки Python относятся к неизменяемому типу данных?

Python обрабатывает изменяемые и неизменяемые объекты по-разному. Доступ к неизменяемым объектам осуществляется быстрее, чем к изменяемым.

Изменяемые объекты отлично подойдут, если вам нужно менять размер объектов, например, `list`, `dict` и т.д.

Неизменяемые значения используются, когда вам нужно убедиться, что созданный вами объект никогда не будет меняться. Неизменяемые объект принципиально дорого менять, поскольку для этого требуется создать копию, а изменяемые менять легко.

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

Метод `str.istitle()` возвращает `True`, если каждое слово в строке `str` начинается с заглавной буквы и в ней есть хотя бы один символ в верхнем регистре. Возвращает `False` в противном случае. Например, заглавные буквы могут следовать только за непрописанными символами, а строчные - только за прописными.

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

Использование `find()` для проверки наличия в строке другой подстроки. Мы также можем использовать функцию `string find()`, чтобы проверить, содержит ли строка подстроку или нет. Эта функция возвращает первую позицию индекса, в которой найдена подстрока, иначе возвращает `-1`.

9. Как найти индекс первого вхождения подстроки в строку?

Для поиска подстроки в строке в Python применяется метод `find()`, который возвращает индекс первого вхождения подстроки в строку и имеет три формы:

`find(str)` : поиск подстроки `str` ведется с начала строки до ее конца

`find(str, start)` : параметр `start` задает начальный индекс, с которого будет производиться поиск

`find(str, start, end)` : параметр `end` задает конечный индекс, до которого будет идти поиск

10. Как подсчитать количество символов в строке?

Метод `len ()` подсчитывает общее количество символов в строке. Если нам нужно подсчитать, сколько раз в строке встречается определенный символ или последовательность символов, мы можем использовать метод `str.count ()`.

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

Давайте возьмем нашу строку `ss = "Sammy Shark!"` и подсчитаем, сколько раз в ней встречается символ "a": `print (ss. count (" a"))` Output 2. Мы можем поискать и другой символ: `print (ss. count (" s"))` Output 0.

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

F-строчный метод. Это новый механизм форматирования строк, представленный PEP 498. Он также известен как интерполяция литеральной строки или, чаще, как F-строки (символ `f`, предшествующий строковому литералу). Основная задача этого механизма — облегчить интерполяцию. Когда мы добавляем строке букву 'F', строка сама становится f-строкой. F-строка может быть отформатирована почти так же, как метод `str.format ()`.

13. Как найти подстроку в заданной части строки?

Для поиска подстроки в строке Python, используется специальный метод `find()`. Метод `find()` как и большинство остальных методов, работает довольно просто. Если искомый элемент найден в строке, он вернет нам индекс первого вхождения, если не найден он вернет нам `-1`.

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

Метод `format()` позволяет добиваться результатов, сходных с теми, которые можно получить, применяя f-строки.

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

Существует метод `isnumeric()`, который возвращает `True` в том случае, если все символы, входящие в строку, являются цифрами. Знаки препинания он цифрами не считает.

16. Как разделить строку по заданному символу?

Здесь нам поможет метод `split()` который разбивает строку по заданному символу или по нескольким символам.

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

Метод `islower()` возвращает `True` только в том случае, если строка составлена исключительно из строчных букв.

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

Сделать это можно, вызвав вышеописанный метод `islower()` для первого символа строки.

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

В некоторых языках это возможно, но Python при попытке выполнения подобной операции будет выдана ошибка `TypeError`.

20. Как «перевернуть» строку?

Для того чтобы «перевернуть» строку, её можно разбить, представив в виде списка символов, «перевернуть» список, и, объединив его элементы, сформировать новую строку.

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

Метод `join()` умеет объединять элементы списков в строки, разделяя отдельные строки с использованием заданного символа.

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

Для решения этих задач можно воспользоваться методами `upper()` и `lower()`, которые, соответственно, приводят все символы строк к верхнему и нижнему регистрам.

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

Тут, как и в одном из предыдущих примеров, мы будем обращаться к символам строки по индексам. Строки в Python иммутабельны, поэтому мы будем заниматься сборкой новой строки на основе существующей.

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

Имеется метод `isupper()`, который похож на уже рассмотренный `islower()`. Но `isupper()` возвращает `True` только в том случае, если вся строка состоит из прописных букв.

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

Метод `splitlines()` разделяет строки по символам разрыва строки.

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

Если обойтись без экспорта модуля, позволяющего работать с регулярными выражениями, то для решения этой задачи можно воспользоваться методом `replace()`.

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

В состав алфавитно-цифровых символов входят буквы и цифры. Для ответа на этот вопрос можно воспользоваться методом `isalnum()`.

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

Есть метод `isspace()` который возвращает `True` только в том случае, если строка состоит исключительно из пробелов.

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

Будет создана новая строка, представляющая собой исходную строку, повторённую три раза.

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

Существует метод `title()`, приводящий к верхнему регистру первую букву каждого слова в строке.

31. *Как пользоваться методом `partition()` ?*

Метод `partition()` разбивает строку по заданной подстроке. После этого результат возвращается в виде кортежа. При этом подстрока, по которой осуществлялась разбивка, тоже входит в кортеж.

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

Метод `rfind()` похож на метод `find()`, но он, в отличие от `find()`, просматривает строку не слева направо, а справа налево, возвращая индекс первого найденного вхождения искомой подстроки.

Вывод: в ходе работы я приобрел навыки по работе со строками при написании программ с помощью языка программирования Python3.