

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»

Вариант \_\_\_\_

Выполнила:  
Маньшина Дарья Алексеевна  
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., канд. тех. наук,  
доцент кафедры  
инфокоммуникаций

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

Ставрополь, 2023 г.

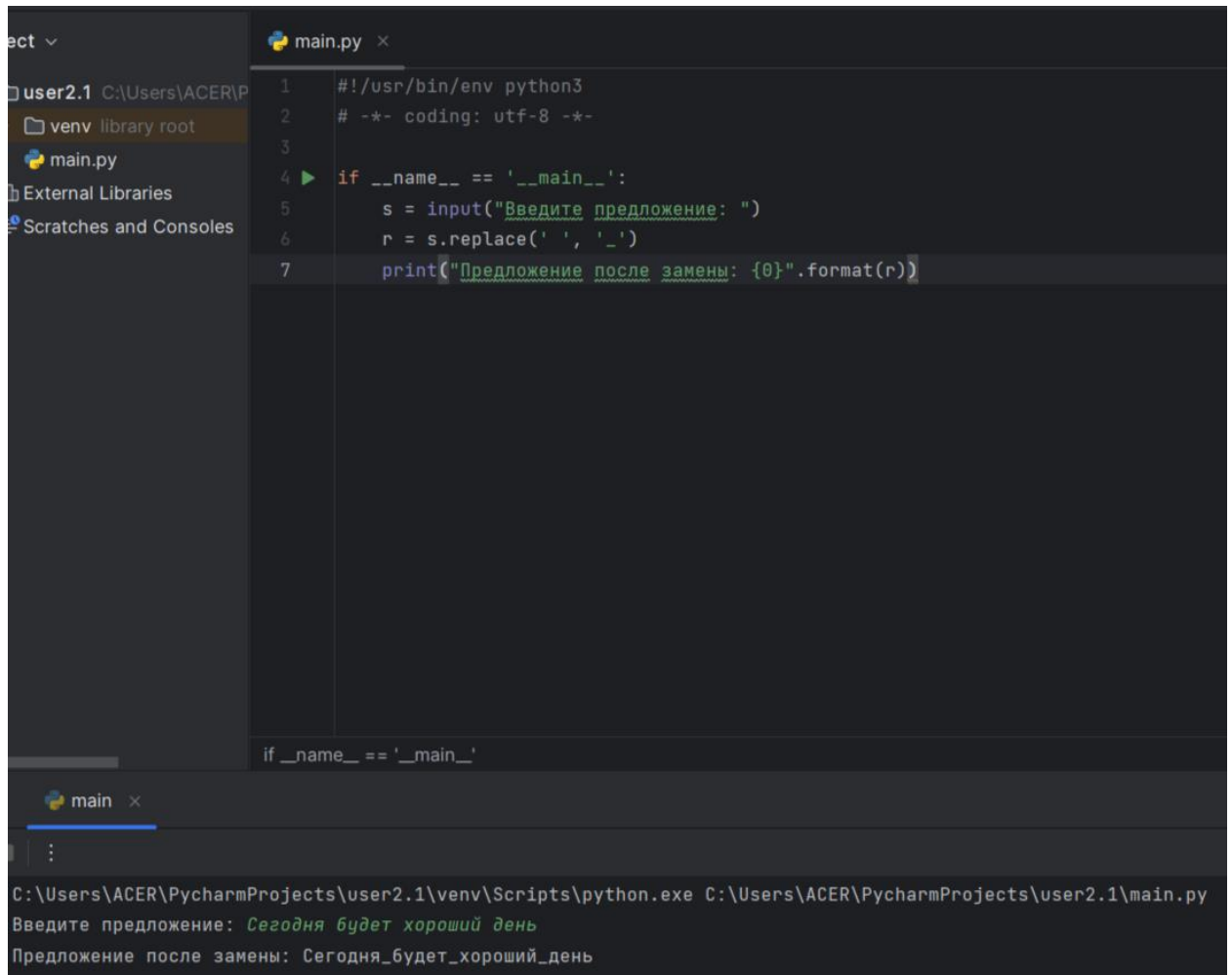
Тема: работа со строками в языке Python

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

Ход работы:

1. Выполним примеры, предложенные в методичке.

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))
```

The screenshot shows a Python IDE with a file named `main.py`. The code in the editor is as follows:

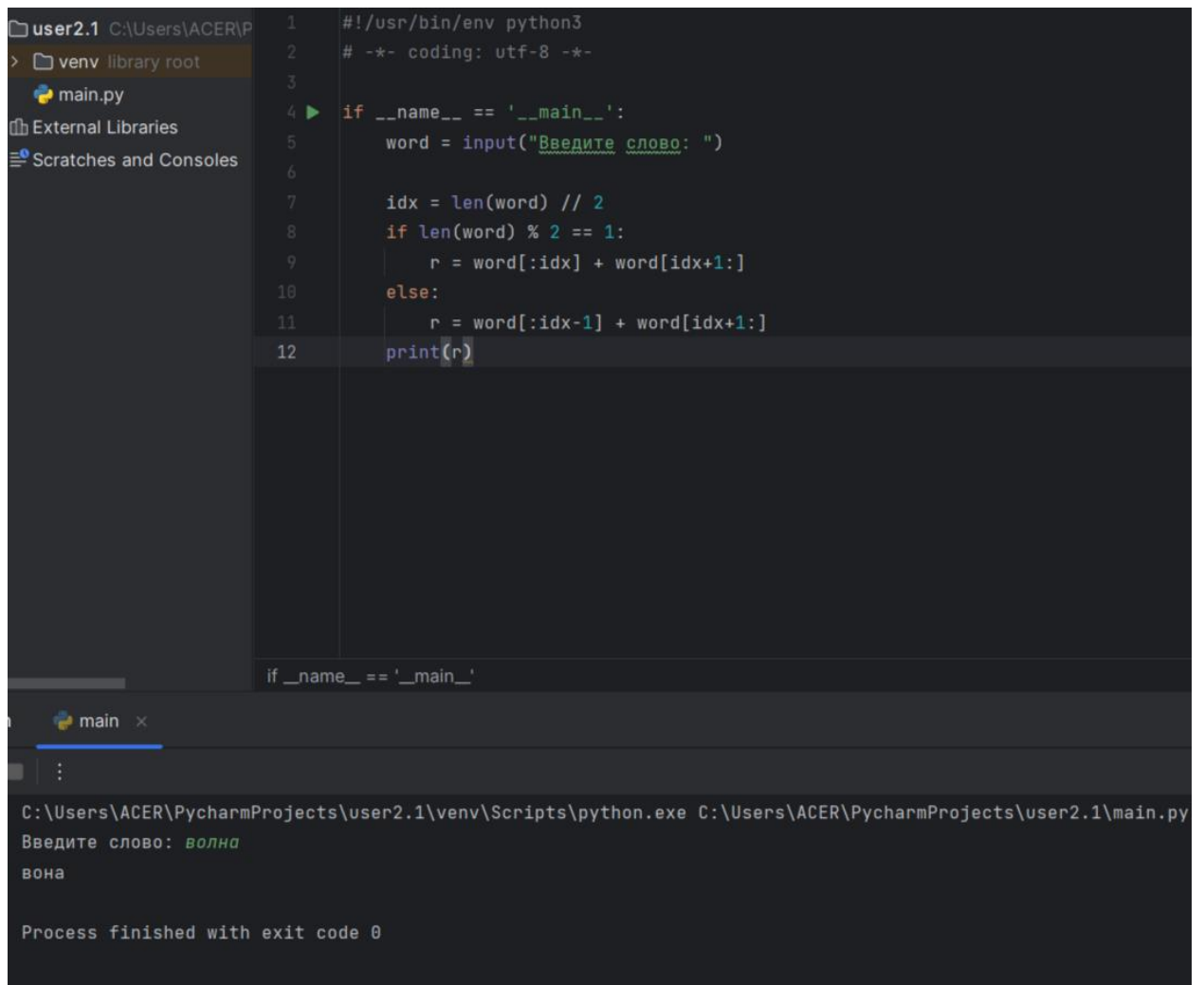
```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  if __name__ == '__main__':
5      s = input("Введите предложение: ")
6      r = s.replace(' ', '_')
7      print("Предложение после замены: {}".format(r))
```

Below the editor, the console output is visible:

```
C:\Users\ACER\PycharmProjects\user2.1\venv\Scripts\python.exe C:\Users\ACER\PycharmProjects\user2.1\main.py
Введите предложение: Сегодня будет хороший день
Предложение после замены: Сегодня_будет_хороший_день
```

Рисунок 1 – Программа и решение для примера №1

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)
```

main x

C:\Users\ACER\PycharmProjects\user2.1\venv\Scripts\python.exe C:\Users\ACER\PycharmProjects\user2.1\main.py

Введите слово: волна

волна

Process finished with exit code 0

Рисунок 2 – Программа и решение для примера №2

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.

```
4 import sys
5
6 if __name__ == '__main__':
7     s = input("Введите предложение: ")
8     n = int(input("Введите длину: "))
9
10    # Проверить требуемую длину.
11    if len(s) >= n:
12        print(
13            "Заданная длина должна быть больше длины предложения",
14            file=sys.stderr
15        )
16        exit(1)
17
18    # Разделить предложение на слова.
19    words = s.split(' ')
20    # Проверить количество слов в предложении.
21    if len(words) < 2:
22        print(
23            "Предложение должно содержать несколько слов",
24            file=sys.stderr
25        )
26        exit(1)
27
28 if __name__ == '__main__':
```

main x

:

Введите предложение: *сегодня хороший день*

Введите длину: *23*

сегодня хороший день

Рисунок 3 – Программа и решение для примера №3

2. Решим индивидуальные задания. Вариант 15 (по списку группы).

Задание 1. Дано предложение. Определить число букв о в нем.

```
1 a = input("Введите предложение: ")
2 count = a.count('o')
3 print(f"Число букв 'o': {count}")
```

main x

:

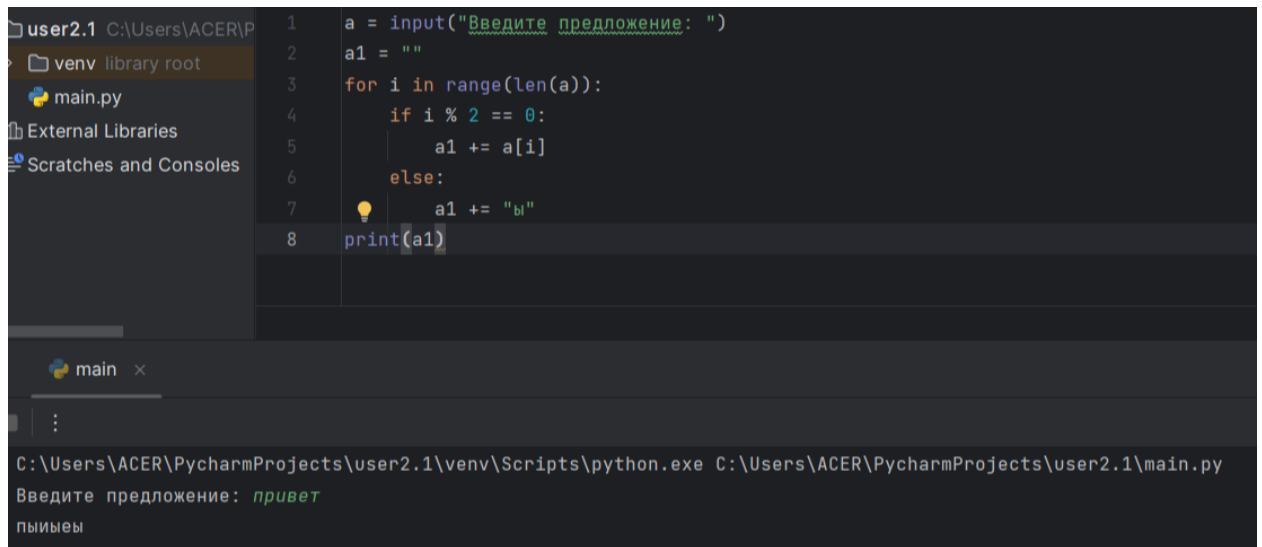
C:\Users\ACER\PycharmProjects\user2.1\venv\Scripts\python.exe C:\Users\ACER\PycharmProjects\user2.1\main.py

Введите предложение: *Сегодня будет хороший день*

Число букв 'o': 3

Рисунок 4 – Программа и решение для индивидуального задания №1

Задание 2. Дано предложение. Все его символы, стоящие на четных местах, заменить буквой ы.



```
1 a = input("Введите предложение: ")
2 a1 = ""
3 for i in range(len(a)):
4     if i % 2 == 0:
5         a1 += a[i]
6     else:
7         a1 += "ы"
8 print(a1)
```

main x

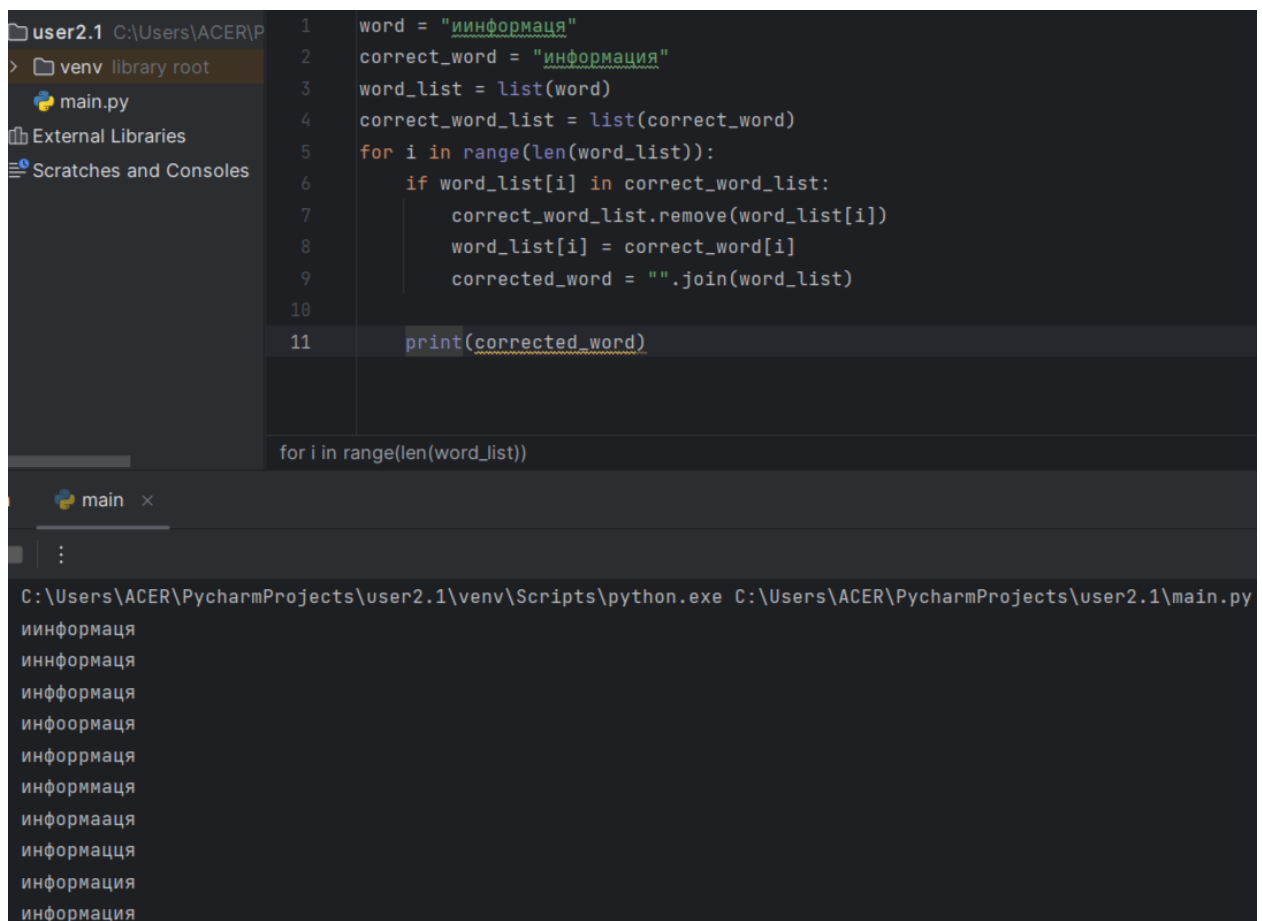
C:\Users\ACER\PycharmProjects\user2.1\venv\Scripts\python.exe C:\Users\ACER\PycharmProjects\user2.1\main.py

Введите предложение: **привет**

пыивыы

Рисунок 5 – Программа и решение для индивидуального задания №2

Задание 3. Дано ошибочно написанное слово иинформация. Путем перемещения его букв получить слово информация.



```
1 word = "иинформация"
2 correct_word = "информация"
3 word_list = list(word)
4 correct_word_list = list(correct_word)
5 for i in range(len(word_list)):
6     if word_list[i] in correct_word_list:
7         correct_word_list.remove(word_list[i])
8         word_list[i] = correct_word[i]
9         corrected_word = "".join(word_list)
10
11 print(corrected_word)
```

main x

C:\Users\ACER\PycharmProjects\user2.1\venv\Scripts\python.exe C:\Users\ACER\PycharmProjects\user2.1\main.py

иинформация  
иинформация  
иинформация  
инфоормация  
информмаця  
информаация  
информация  
информация  
информация  
информация

Рисунок 6 – Программа и решение для индивидуального задания №3

3. Клонирование репозитория. Организация репозитория с моделью ветвления git flow.

```
C:\Users\ACER>git clone https://github.com/Dash-A1/2.3.git
Cloning into '2.3'...
remote: Enumerating objects: 30, done.
remote: Counting objects: 100% (30/30), done.
remote: Compressing objects: 100% (27/27), done.
remote: Total 30 (delta 8), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (30/30), 9.67 KiB | 1.61 MiB/s, done.
Resolving deltas: 100% (8/8), done.
```

Рисунок 7 – Клонирование

```
C:\Users\ACER>git flow init
No branches exist yet. Base branches must be created now.
Branch name for production releases: [master]
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/ACER/.git/hooks]
```

Рисунок 8 – Использование git flow init

```
C:\Users\ACER>git checkout master
Switched to branch 'master'

C:\Users\ACER>git branch
  develop
* master

C:\Users\ACER>git merge develop
Already up to date.
```

Рисунок 9 – Использование git checkout

Ответы на контрольные вопросы

1. Что такое строки в языке Python?

**Строки в Python** - упорядоченные неизменяемые 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?

Работа со строками в Python очень удобна. Существует несколько литералов строк: строки в апострофах и в кавычках; экранированные последовательности - служебные символы

```
S = 'spam"s'
```

```
S = "spam's"
```

Строки в апострофах и в кавычках - одно и то же. Причина наличия двух вариантов в том, чтобы позволить вставлять в литералы строк символы кавычек или апострофов, не используя экранирование.

Экранированные последовательности позволяют вставить символы, которые сложно ввести с клавиатуры.

3. Какие операции и функции существуют для строк?

`S.split(символ)` - разбиение строки по разделителю

`S.isdigit()` - состоит ли строка из цифр

`S.isalpha()` - состоит ли строка из букв

`S.isalnum()` - состоит ли строка из цифр или букв

4. Как осуществляется индексирование строк?

Доступ к отдельным символам в строке можно получить, указав имя строки, за которым следует число в квадратных скобках (`[]`). Индексация строк в Python основана на нуле: первый символ в строке имеет индекс 0, следующий-индекс 1 и так далее.

5. Как осуществляется работа со срезами для строк?

**Срез (slice)** — извлечение из данной строки одного символа или некоторого фрагмента (подстроки).

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.

Если же номер символа в срезе строки `S` больше либо равен `len(S)`, или меньше, чем `-len(S)`, то при обращении к этому символу строки произойдет ошибка **`IndexError: string index out of range`**.

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.

Если задать срез с тремя параметрами `S[a:b:d]`, то 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.

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

Кортежи относятся к неизменяемому типу данных и, следовательно, не могут быть изменены после их создания в Python. Это потому, что они поддерживают те же последовательности операций, что и строки. А мы все знаем, что строки являются неизменяемыми объектами.

## 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.

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

Метод `includes()` проверяет, содержит ли строка заданную подстроку, и возвращает, соответственно `true` или `false`.



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

Чтобы найти позицию первого вхождения строки, вы можете использовать метод `string.find()` в Python. Где, `string` – это строка, в которой нужно найти индекс первого вхождения подстроки.

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

В Python есть встроенная функция `len()`, которая позволяет узнать длину строки, т. е. количество символов в ней. Функция `len()` принимает аргументом строку и возвращает целое число, равное количеству символов в этой строке.

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

При помощи метода `count()`, который возвращает количество вхождений в строку заданного символа.

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

"F-строки" обеспечивают краткий, читаемый способ включения значения выражений Python внутри строк. В исходном коде Python форматированная строка или по другому `f-string` - это буквальная строка с префиксом `'f'` или `'F'`, которая содержит выражения внутри фигурных скобок `{}`. Выражения заменяются их значениями.

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

Метод `indexOf` осуществляет поиск подстроки в строке. В первом параметре указываем искомую подстроку в нужном нам регистре (большие буквы или маленькие). Метод вернет позицию первого совпадения, а если оно не найдено, то вернет `-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. Используя этот метод, учитывайте то, что знаки препинания он цифрами не считает.

16. Как разделить строку по заданному символу?

Чтобы разделить строку на символ-разделитель, используйте `IndexOf` метод или `IndexOfAny` для поиска символа-разделителя в строке. Чтобы разделить строку в строке разделителя, используйте `IndexOf` метод или `IndexOfAny` для поиска первого символа строки разделителя.

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?

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.

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

Чтобы конвертировать число в строку, используйте встроенную функцию `str()` . Эта функция в качестве входящих данных принимает число (или другой тип данных), а в результате выдает строку.

## 20. Как «перевернуть» строку?

Строка поворачивается путем обратной итерации при помощи метода `reversed()`. После этого происходит объединение символов в новую строку при помощи метода `join()`.

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

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

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

Чтобы преобразовать `String` в верхний регистр, вы можете использовать метод `upper()`. Функция `String.lower()` возвращает строку, в которой все символы этой строки преобразованы в нижний регистр.

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

Вы можете использовать `str.title()` чтобы получить версию строки в заглавном регистре. Это преобразует первый символ каждого слова в строке в верхний регистр, а остальные символы в нижний регистр.

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

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?

Можно прибегнуть, соответственно, к методам `startswith()` и `endswith()`

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?

Метод `str.title()` возвращает копию строки `str`, в которой у каждого слова в строке, первый символ имеет верхний регистр, а остальные символы слова переводятся в нижний регистр. Другими словами, метод вернет копию строки, в которой все слова начинаются с заглавной буквы.

31. Как пользоваться методом `partition()`?

Метод `partition` в Python принимает разделитель параметров строки, который разделяет строку при первом ее появлении.

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

Метод `str.rfind()` возвращает индекс последнего совпадения подстроки `sub` в строке `str`, где подстрока или символ `sub` находится в пределах среза `str[start:end]`. Другими словами, находит и возвращает индекс по которому обнаруживается конец указанной подстроки в исходной строке `str`.

Вывод: в ходе лабораторной работы приобрела навыки по работе со строками при написании программ с помощью языка программирования Python версии 3.x.