МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ

Отчет о лабораторной работе №2.3 по дисциплине основы программной инженерии

Выполнил:
Шальнев Владимир Сергеевич,
2 курс, группа ПИЖ-б-о-20-1,

Проверил:
Доцент кафедры
прикладной математики и
компьютерной безопасности,
Воронкин Р.А.

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2021 г.

ВЫПОЛНЕНИЕ:

```
F:\pythonProject>git clone https://github.com/HAXF13D/laboratory-6
Cloning into 'laboratory-6'...
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 11 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (11/11), done.

F:\pythonProject>cd laboratory-6

F:\pythonProject\laboratory-6>
```

Клонирование репозитория

```
F:\pythonProject\laboratory-6>git checkout develop
Switched to a new branch 'develop'
Branch 'develop' set up to track remote branch 'develop' from 'origin'.
```

Переход на ветку develop

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

if __name__ == '__main__':
    s = input("Введите предложение: ")
    r = s.replace(' ', '_')
    print("Предложение после замены: {r}")
```

Пример 1

```
Введите предложение: мама мыла раму
Предложение после замены: мама_мыла_раму
```

Значение №1

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

if __name__ == '__main__':
    word = input("Введите слово: ")
    idx = len(word) // 2
    if len(word) % 2 == 1:
        # Длина слова нечетная.
        r = word[:idx] + word[idx+1:]
    else:
        # Длина слова четная.
        r = word[:idx-1] + word[idx+1:]
    print(r)
```

Пример 2

```
Введите слово: абвг
аг
```

Значение №1

```
Введите слово: абвгд
абгд
```

Значение №2

```python
import sys

if __name__ == '__main__':
    s = input("Введите предложение: ")
    n = int(input("Введите длину: "))
    # Проверить требуюемую длину.
    if len(s) >= n:
        print(
            "Заданная длина должна быть больше длины предложения",
            file=sys.stderr
        )
        exit(1)
    # Разделить предложение на слова.
    words = s.split(' ')
    # Проверить количество слов в предложении.
    if len(words) < 2:
        print(
            "Предложение должно содержать несколько слов",
            file=sys.stderr
        )
        exit(1)
    # Количество пробелов для добавления.
    delta = n
    for word in words:
        delta -= len(word)
    # Количество пробелов на каждое слово.
    w, r = delta // (len(words) - 1), delta % (len(words) - 1)
    # Сформировать список для хранения слов и пробелов.
    lst = []
    # Пронумеровать все слова в списке и перебрать их.
    for i, word in enumerate(words):
        lst.append(word)
        # Если слово не является последним, добавить пробелы.
        if i < len(words) - 1:
            # Определить количество пробелов.
            width = w
            if r > 0:
                width += 1
                r -= 1
            # Добавить заданное количество пробелов в список.
            if width > 0:
                lst.append(' ' * width)

    # Вывести новое предложение, объединив все элементы списка lst.
    print(''.join(lst))
```

Пример 3

```
Введите предложение: абвгд
Введите длину: 2
Заданная длина должна быть больше длины предложения
```

Значение №1

```
Введите предложение: абв
Введите длину: 4
Предложение должно содержать несколько слов
```

Значение №2

```
Введите предложение: мама мыла
Введите длину: 12
мама    мыла
```

Значение №3

```
F:\pythonProject\laboratory-6>git status
On branch develop
Your branch is up to date with 'origin/develop'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        example1.py
        example2.py
        example3.py

nothing added to commit but untracked files present (use "git add" to track)

F:\pythonProject\laboratory-6>git add .

F:\pythonProject\laboratory-6>git commit -m "Add example task"
[develop 3262a0a] Add example task
 3 files changed, 65 insertions(+)
 create mode 100644 example1.py
 create mode 100644 example2.py
 create mode 100644 example3.py
```

Коммит изменений

```python
# variant 25

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

if __name__ == '__main__':
    text = input("Введите текст: ")
    sent_amount = len(text.split('.')) - 1
    print(sent_amount)
```

Индивидуальное задание 1

```
Введите текст: Дан текст. Определить, сколько в нем предложений.
2
```

Значение №1

```python
# variant 25

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

if __name__ == '__main__':
    word = input("Введите слово четной длины: ")
    if len(word) % 2 != 0:
        print(
            "Слово должно быть четной длины",
            file=sys.stderr
        )
        exit(1)
    n = len(word)
    word = word[n // 2: n][::-1] + word[0: n // 2][::-1]
    print(f"Слово после изменений: {word}")
```

Индивидуальное задание 2

```
Введите слово четной длины: qwerty
Слово после изменений: ytrewq
```

Значение №1

```
Введите слово четной длины: wer
Слово должно быть четной длины
```

Значение №2

```python
# variant 25

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

if __name__ == '__main__':
    word = input("Введите слово из 12 символов: ")
    if len(word) != 12:
        print(
            "Длина слова должна быть равна 12",
            file=sys.stderr
        )
        exit(1)
    n = len(word)
    word = word[n // 2: n][::-1] + word[0: n // 2][::-1]
    print(f"Слово после изменений: {word}")
```

Индивидуальное задание 3

```
Введите слово из 12 символов: qwertyuiopas
Слово после изменений: sapoiuytrewq
```

Значение №1

```
Введите слово из 12 символов: qwerty
Длина слова должна быть равна 12
```

Значение №2

```python
# variant 25

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

if __name__ == '__main__':
    sentence = input("Введите предложение: ")
    sentence = sentence.replace(',', '')
    sentence = sentence.replace('.', '')
    sentence = sentence.replace('!', '')
    sentence = sentence.replace('?', '')
    words = sentence.lower().split(' ')
    for i in range(len(words)):
        for j in range(i + 1, len(words)):
            if words[i] == words[j]:
                print(f"Одинаковые слова: {words[i]}")
                exit(1)
    print("Нет одинаковых слов")
```

Задание повышенной сложности

```
Введите предложение: Мирский мирный мир в мире мира мир
Одинаковые слова: мир
```

Значение №1

```
F:\pythonProject\laboratory-6>git status
On branch develop
Your branch is ahead of 'origin/develop' by 1 commit.
  (use "git push" to publish your local commits)

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        hardtask.py
        task1.py
        task2.py
        task3.py

nothing added to commit but untracked files present (use "git add" to track)

F:\pythonProject\laboratory-6>git add .

F:\pythonProject\laboratory-6>git commit -m "Add tasks"
[develop 4ee3f3d] Add tasks
 4 files changed, 61 insertions(+)
 create mode 100644 hardtask.py
 create mode 100644 task1.py
 create mode 100644 task2.py
 create mode 100644 task3.py
```

Коммит изменений

ОТВЕТЫ НА ВОПРОСЫ:

1. Строки в Python - упорядоченные последовательности символов, используемые для хранения и представления текстовой информации.

2. Строки в апострофах и кавычках, экранированные последовательности – служебные символы, “Сырые” строки, строки в тройных апострофах или кавычах.

3. Сложение, умножение. Строковых функций много, вот некоторые из них:

len() – длина строки

str() – изменяет типо объекта на string

4. <название строковой переменной> [число от 0 до длины строки - 1]

5. Если s это строка, выражение формы s[m:n] возвращает часть s , начинающуюся с позиции m , и до позиции n , но не включая позицию:

6. Так их легче представить в памяти.

7. s.istitle()

8. if s1 in s2

9. s.find(s2)

10. len(s)

11. s.count(<char>)

12. Они позволяют проще формировать строки. Пример: s = f”Ваш id = {id}”

13. s.find(<sub>[, <start>[,<end>]])

14.” Ваш id = {}”.format(id)

15. s.isdigit()

16. 'foo.bar.baz.qux'.rsplit(sep='.')

17. s.islower()

18. s[0].islower()

19. Нет, можно только преобразовать число в строку и уже его прибавить.

20. s = s[::-1]

21. ‘-‘.join[<iterable>]

22. Верхний – s.upper(), нижний – s.lower()

23. s[0].upper() s[len(s)-1].upper()

24. s.isupper()

25. В случае, если надо сохранить символы конца стркои.

26. s.replace(“что менять”, “на что менять”)

27. string.endswith(<suffix>[, <start>[, <end>]]) – заканчивается, string.startswith(<suffix>[, <start>[, <end>]]) – начинается.

28. s.isspace()

29. Будет получена копия строки, состоящая из 3 исходных.

30. s.title()

31. string.partition(<sep>) делит строку на основе разделителя. s.partition(<sep>) отделяет от s подстроку длиной от начала до первого вхождения <sep> . Возвращаемое значение представляет собой кортеж из трех частей:

Часть s до <sep>

Разделитель <sep>

Часть s после <sep>

32. Индекс последнего вхождения подстроки в строку.