

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии
Департамент цифровых, робототехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №6
дисциплины
«Основы кроссплатформенного программирования»

Выполнил:
Мидов Андемир Исланович
2 курс, группа ИТС-б-о-23-1,
11.03.02 «Инфокоммуникационные
технологии и системы связи», очная
форма обучения

(подпись)

Проверил:
Доцент департамента цифровых,
робототехнических систем и
электроники Воронкин Р.А.

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

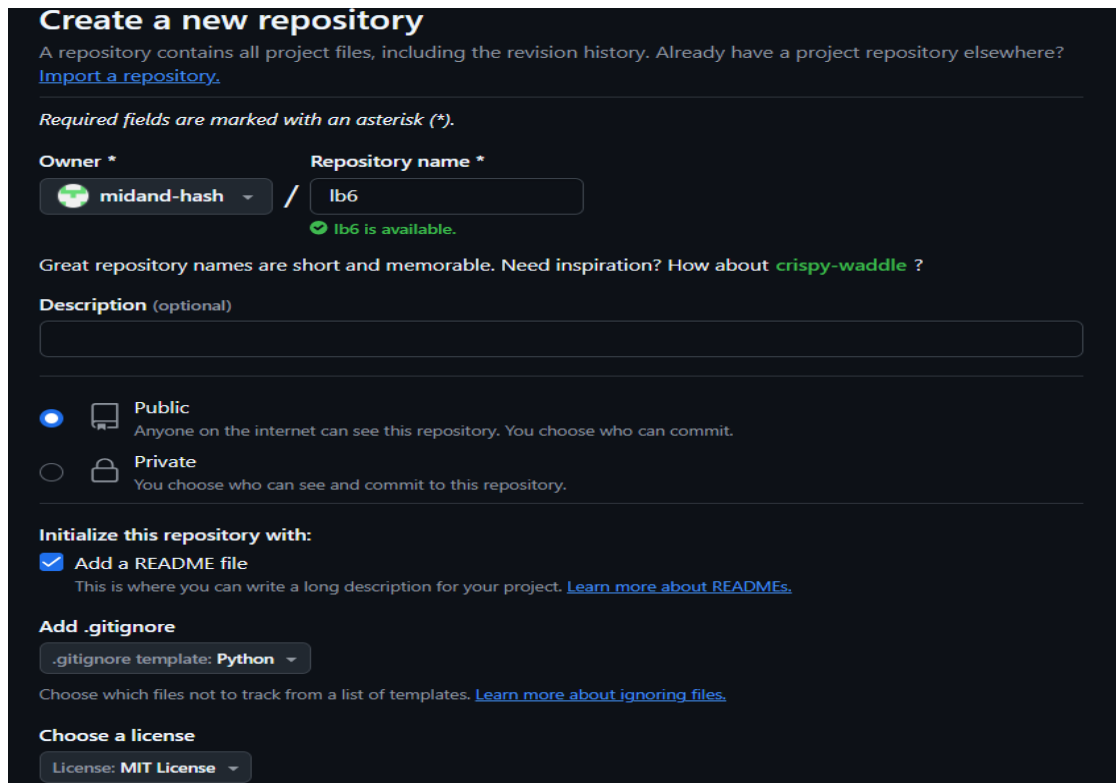
Ставрополь, 2025 г

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.

Порядок выполнения работы:

1. Изучил теоретический материал.
2. Приступил к выполнению заданий.
3. Создание репозитория.



Create a new repository

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

Required fields are marked with an asterisk (*).

Owner * midand-hash / **Repository name *** lb6
✓ lb6 is available.

Great repository names are short and memorable. Need inspiration? How about **crispy-waddle** ?

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:

☒ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: **Python**

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

Choose a license

License: **MIT License**

Рисунок 1. Репозиторий

4. Клонирование репозитория.

```
C:\Users\AND>git clone https://github.com/midand-hash/lb6.git
Cloning into 'lb6'...
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 (from 0)
Receiving objects: 100% (5/5), done.
```

Рисунок 2. Клонирование

5. Создание проекта в папке репозитория.

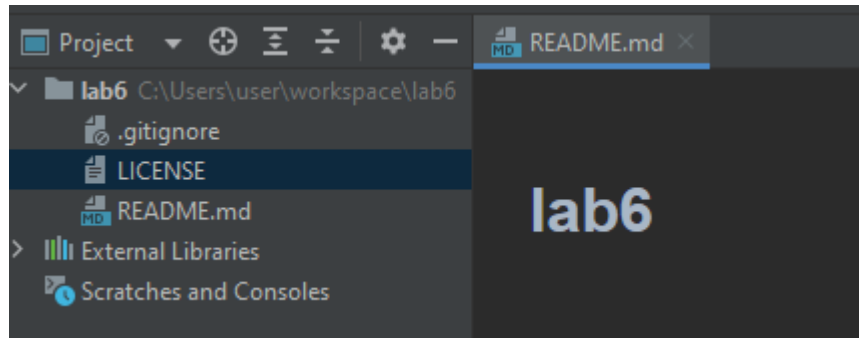


Рисунок 3. Проект

6. Приступил к выполнению примеров.

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

```
Введите предложение: Hello world  
Предложение после замены: Hello_world
```

Рисунок 4. Пример 1.

```
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)
```

```
Введите слово: Привет мир  
Прив мир
```

Рисунок 5. Пример 2.

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

```
import sys

if __name__ == "__main__":
    s = input("Введите предложение: ")
    n = int(input("Введите длину: "))

    # Проверить требуемую 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 - len(s)

    for word in words:
        delta -= len(word)

    # Количество пробелов на каждое слово.
    w, r = divmod(delta, len(words) - 1)

    # Сформировать список для хранения слов и пробелов.
    lst = []

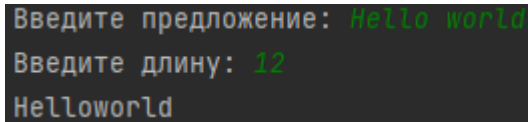
    # Пронумеровать все слова в списке и перебрать их.
    for i, word in enumerate(words):
        lst.append(word)

        # 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)

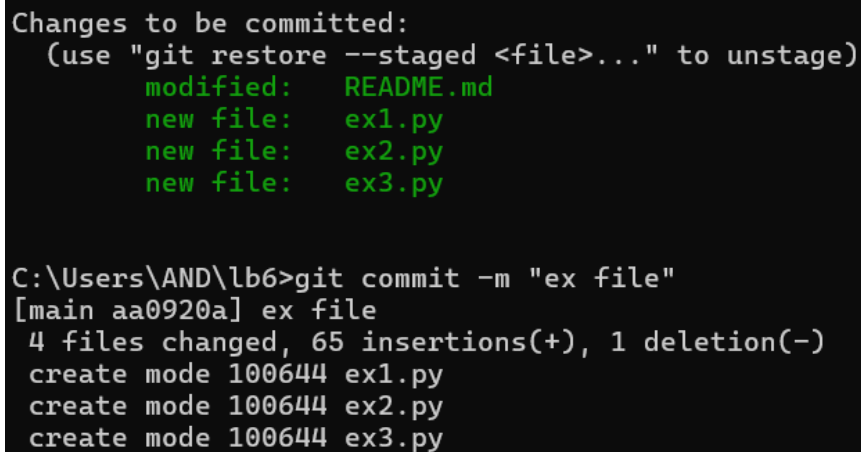
# Вывести новое предложение, объединив все элементы списка lst.
print(".".join(lst))
```



```
Введите предложение: Hello world
Введите длину: 12
Helloworld
```

Рисунок 6. Пример 3.

7. Зафиксировал изменения в репозитории.



```
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   README.md
    new file:   ex1.py
    new file:   ex2.py
    new file:   ex3.py

C:\Users\AND\lb6>git commit -m "ex file"
[main aa0920a] ex file
 4 files changed, 65 insertions(+), 1 deletion(-)
 create mode 100644 ex1.py
 create mode 100644 ex2.py
 create mode 100644 ex3.py
```

Рисунок 7. Изменения

8. Привел скриншоты результатов каждой из программы.

9. Приступил к выполнению индивидуального задания.

```
sentence = input("Enter a sentence: ")
```

```
matches = [word for word in sentence.split() if "нн" in word]
```

```
if matches:
```

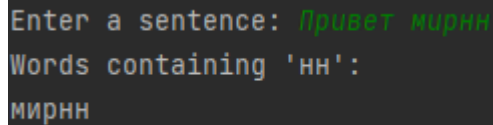
```
    print("Words containing 'нн':")
```

```
    for word in matches:
```

```
        print(word)
```

```
else:
```

```
    print("No occurrences of 'нн' found.")
```



```
Enter a sentence: Привет мирнн
Words containing 'нн':
мирнн
```

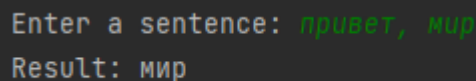
Рисунок 8. Задание 1

```
sentence = input("Enter a sentence: ")

first_comma = sentence.find(',')
second_comma = sentence.find(',', first_comma + 1)

if first_comma != -1 and second_comma != -1:
    result = sentence[first_comma + 1:second_comma]
elif first_comma != -1:
    result = sentence[first_comma + 1:]
else:
    result = "No commas found in the sentence."

print(f"Result: {result.strip()}")
```



```
Enter a sentence: привет, мир
Result: мир
```

Рисунок 9. Задание 2

```
correct_word1 = "процессор".replace("pp", "p").replace("с", "cc") #
Исправление слова "процессор"
correct_phrase1 = "текстовый файл".replace("ее", "е").replace("ый файл", "ый
файл") # Исправление фразы "текстовый файл"
correct_phrase2 = "програма и алгоритм".replace("мм", "м").replace("лл",
"л").replace("програма", "программа") # Исправление фразы "програма и
алгоритм"
correct_phrase3 = "процессор и паммать".replace("с", "cc").replace("мм", "м") #
Исправление фразы "процессор и паммать"

print(f"Исправленное слово: {correct_word1}")
print(f"Исправленная фраза 1: {correct_phrase1}")
print(f"Исправленная фраза 2: {correct_phrase2}")
print(f"Исправленная фраза 3: {correct_phrase3}")
```

```
Исправленное слово: процессор
Исправленная фраза 1: текстовый файл
Исправленная фраза 2: программа и алгоритм
Исправленная фраза 3: процессор и память
```

Рисунок 10. Задание 3

10. Зафиксировал изменения в репозитории.

```
C:\Users\AND\lb6>git add .
C:\Users\AND\lb6>git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   i1.py
        new file:   i2.py
        new file:   i3.py

C:\Users\AND\lb6>git commit -m "ind file"
[main 035152e] ind file
3 files changed, 33 insertions(+)
create mode 100644 i1.py
create mode 100644 i2.py
create mode 100644 i3.py
C:\Users\AND\lb6>git push
```

Рисунок 11. Изменения

Ответы на контрольные вопросы:

1. Что такое строки в языке Python?

Строки в Python — строки в языке Python это последовательности символов, используемые для хранения текстовой информации. Они могут содержать буквы, цифры, пробелы и специальные символы.

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

Строки можно задавать с помощью одинарных, двойных, тройных одинарных и тройных двойных кавычек.

3. Какие операции и функции существуют для строк?

Операции включают конкатенацию, повторение, а также функции, такие как upper, lower, replace, split.

4. Как осуществляется индексирование строк?

Индексирование строк осуществляется с помощью квадратных скобок. Индексы начинаются с 0 для первого символа, -1 для последнего и тд.

5. Как осуществляется работа со срезами для строк?

Срезы позволяют извлекать подстроки. Синтаксис: s - start - начальный индекс, а end - конечный индекс.

6. Почему строки Python относятся к неизменяемому типу данных?

Строки являются неизменяемыми, потому что после создания их содержимое нельзя изменить. Любые операции, которые изменяют строку, создают новую строку.

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

istitle, s.istitle.

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

Оператор in: substring in s.

9. Как найти индекс первого вхождения подстроки в строку?

Через find: s.find.

10. Как подсчитать количество символов в строке?

Через функцию len: len(s).

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

Через метод count.

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

Форматированные строки позволяют вставлять выражения в строку.

13. Как найти подстроку в заданной части строки?

Через метод find(start, end)

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

Через format: "Пример, {}".format(name).

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

Через `isdigit`

16. Как разделить строку по заданному символу?

Через метод `split()`.

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

Через метод `islower`: `s.islower`.

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

Через `islower` для первого символа.

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

Нет. Строки и числа нельзя складывать напрямую.

20. Как «перевернуть» строку?

Через срезы

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

через `join`

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

через методы `upper` и `lower`

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

Через `upper`

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

Через метод `isupper`

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

Метод `splitlines()` полезен для разделения строки на строки по символам новой строки.

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

Через метод `replace`.

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

Через методы `startswith()` и `endswith()`

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

Используйте метод `isspace: s.isspace`.

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

Строка будет повторена трижды.

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

Через метод `title: s.title`.

31. Как пользоваться методом `partition()`?

Метод `partition()` разделяет строку на три части: до, разделитель и после.

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

Метод `rfind()` используется для поиска последнего вхождения подстроки в строке

Вывод: в ходе работы исследовал базовые возможности системы контроля версий Git для работы с локальными репозиториями.

Ссылка на репозиторий: <https://github.com/midand-hash/lb6.git>