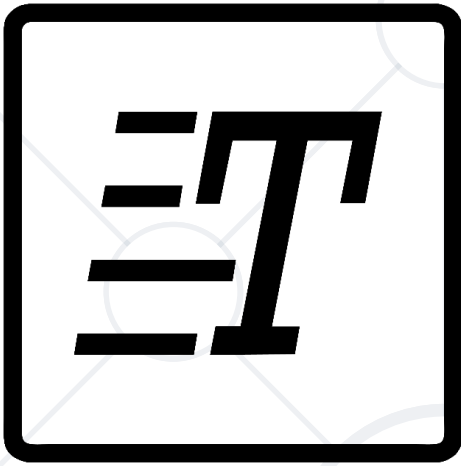


Text Processing



SoftUni Team
Technical Trainers



SoftUni



Software University

<https://softuni.bg>

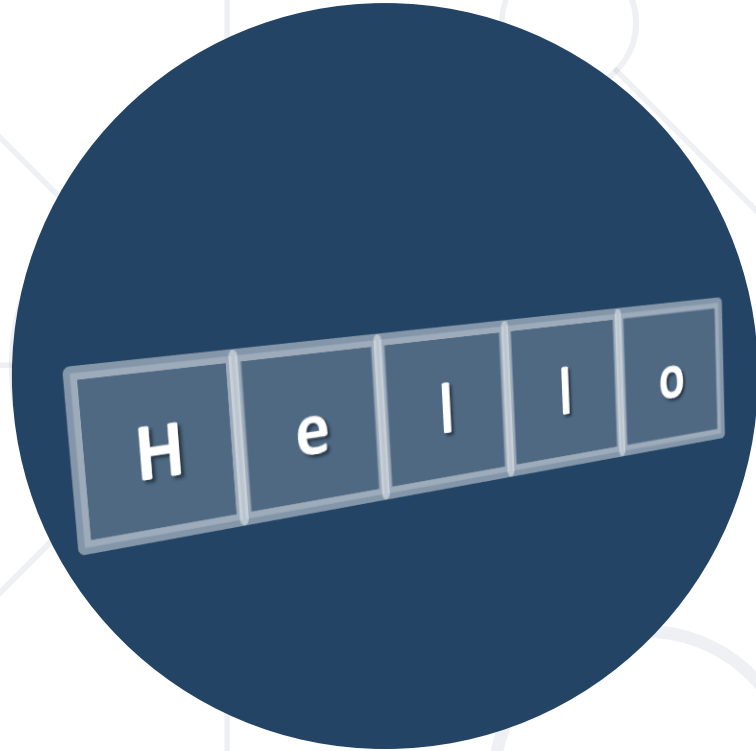
Table of Contents

1. String Definition
2. String Manipulation
3. String Methods



sli.do

#fund-python



String Definition

What is a String?

- A string is a **sequence** of characters
- A character is simply a **symbol**
- Computers do not deal with characters, they deal with **numbers**
- A character is stored and manipulated as a combination of **0's** and **1's**
- In Python, a string is a sequence of **Unicode** characters



- String literals are surrounded by either **single quotation** marks, or **double quotation** marks
- **'hello'** is the same as **"hello"**
- You can display a string literal with the **print()** function

```
print("Hello")  
print('Hello')
```

- Assigning a string to a variable is done with the variable **name** followed by an **equal sign** and the **string**

```
a = "Hello"  
print(a)
```

- Assign a **multiline** string to a variable by using **three quotes**

```
a = """Lorem ipsum dolor sit amet,consectetur adipiscing  
elit, sed do eiusmod tempor incididuntut labore et  
dolore magna aliqua."""  
print(a)
```

Methods str() and split()

- The **str()** function converts the specified value into a string

```
x = str(3.5)  
print(x) # "3.5"
```

- The **split()** method splits a string into a list

```
txt = "hello, my name is Peter, I am 26 years old"  
lst = txt.split(", ")  
print(lst)  
# ["hello", "my name is Peter", "I am 26 years old"]
```


Problem: Reverse Strings

- You will be given series of strings until you receive an "end" command
- Print each pair on separate line in format "**{word} = {reversed word}**"

```
hello  
Softuni  
bottle  
end
```



```
hello = oLleh  
Softuni = inutfoS  
bottle = elttob
```

Solution: Reverse Strings

```
text = input()
while text != "end":
    text_reversed = ""
    for ch in reversed(text):
        text_reversed += ch
    print(text + " = " + text_reversed)
    text = input()
```



String Manipulation

- Use the "+" operator to merge strings

```
str1 = "Hello"  
str2 = "World"  
str3 = str1 + str2  
# HelloWorld
```

- The "*" operator repeats the string

```
str1 = "red"  
print(str1 * 3)  
# redredred
```

- Formatting with the "%" operator

```
x = 'apples'  
y = 'lemons'  
z = "In the basket are %s and %s" % (x, y)
```

- Formatting with the "{}" operators

```
x = 'apples'  
y = 'lemons'  
z = "In the basket are {} and {}".format(x, y)
```

- Python 3 introduced new and simple way for string formatting called "**f-String**"
- Since Python 3 came out, it is the most used way for string formatting

```
x = 'apples'  
y = 'lemons'  
z = f"In the basket are {x} and {y}"  
# In the basket are apples and Lemons
```

- Python offers many ways to substring a string
- It is often called "**slicing**"
- Slicing can also be used with **lists**
- It is equivalent to the **slice()** method

```
text = "My name is Peter"  
name = text[-5:]  
# same as text[11:] or text[slice(-5, 16, 1)]
```

Problem: Repeat Strings

- Write a program that reads an array of strings
- Each string is repeated **N** times, where N is the **length** of the string
- Print the concatenated string

`hi abc add` → `hihiabcbcabcbaddaddadd`

Solution: Repeat Strings

```
strings = input().split(" ")
result = ""
for word in strings:
    length = len(word)
    result += word * length
print(result)
```



String Methods

- Check if a character is a digit: **isdigit()**

```
'1'.isdigit() # True  
'p'.isdigit() # False
```

- Check if a character is upper/lower case: **isupper()**
and **islower()**

```
'P'.isupper() # True  
'P'.islower() # False  
'u'.islower() # True
```

- Convert to upper/lower case: **upper()** and **lower()**

```
"hello".upper() # "HELLO"
```

```
"HeLlO".lower() # "hello"
```

- Remove white spaces in start/end or both: **strip()**, **rstrip()**, **lstrip()**

```
" hello ".lstrip() # "hello "
```

```
" hello ".rstrip() # " hello"
```

```
" hello ".strip() # "hello"
```

- You can use the **replace()** method to replace all occurrences of a specified phrase with another specified phrase

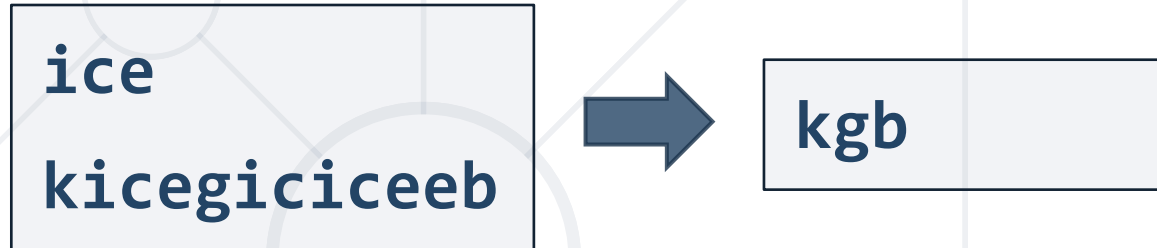
```
txt = "I like bananas"  
print(txt.replace("bananas", "apples")) # I like apples
```

- If you only want to replace a certain number of phrases, add **count**

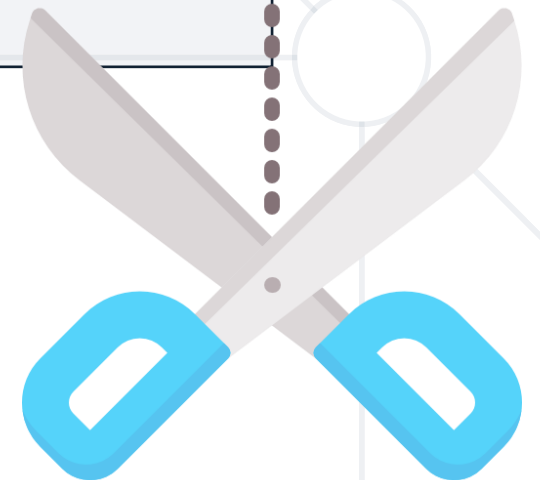
```
txt = "I like bananas bananas bananas"  
x = txt.replace("bananas", "apples", 2)  
print(x)
```

Problem: Substring

- You will receive **two strings**
- Write a program that removes all of the **occurrences of the first string in the second** until there is no match
- At the end print the **remaining** string



```
first = input()
second = input()
while first in second:
    second = second.replace(first, "")
print(second)
```



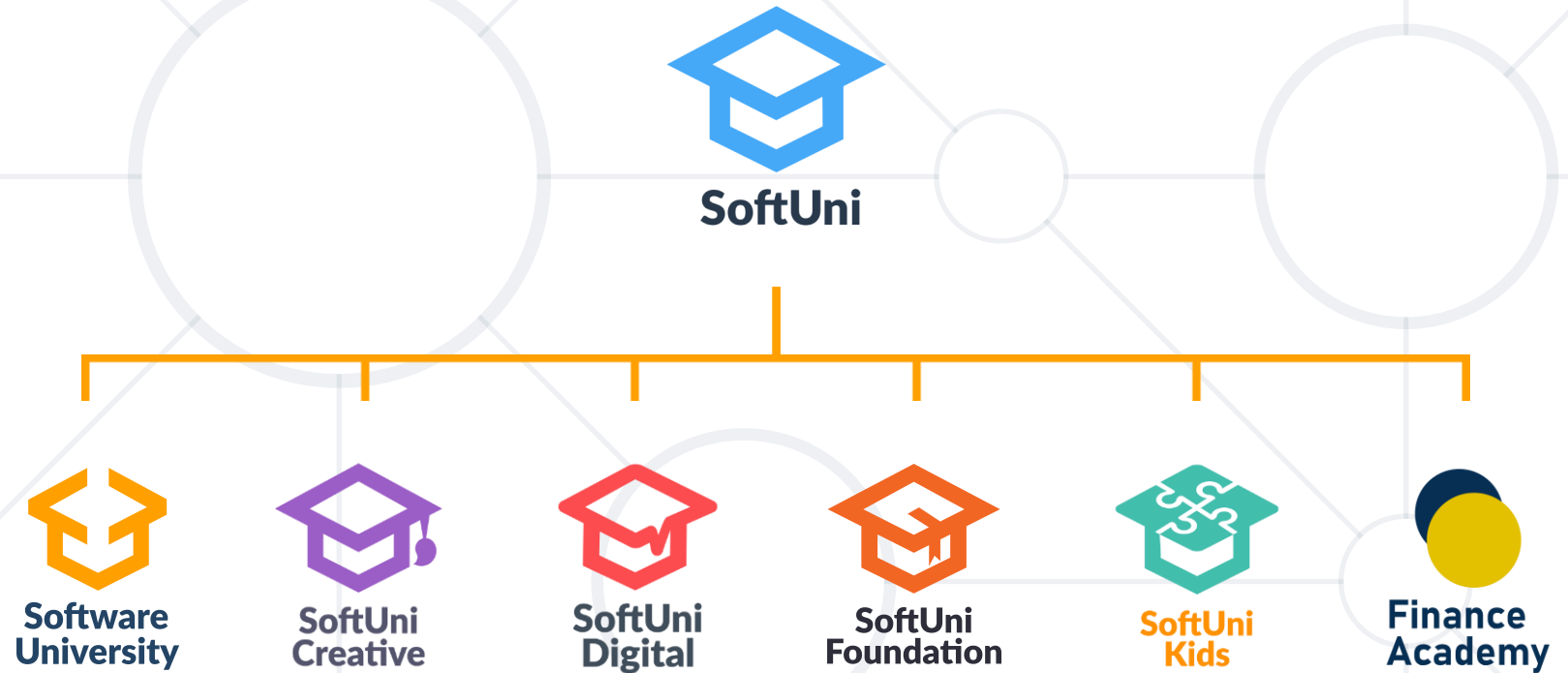


Live Exercises

- A string is a **sequence** of characters
- A character is simply a **symbol**
- We can use "+" operator to merge strings
- We can use "*" operator to repeat strings
- There are many methods we can use to manipulate strings: **upper()**, **lower()**, **split()**, etc.



Questions?



SoftUni Diamond Partners

**SUPER
HOSTING
.BG**



**Coca-Cola HBC
Bulgaria**

 **Flutter**TM
International

INDEAVR
Serving the high achievers



AMBITIONED

 **DRAFT
KINGS**



BOSCH

 **Postbank**
Решения за твоето утре

 **PHAR
VISION**



SmartIT

DXC
TECHNOLOGY

createX

- Software University – High-Quality Education, Profession and Job for Software Developers

- softuni.bg, softuni.org

- Software University Foundation

- softuni.foundation

- Software University @ Facebook

- facebook.com/SoftwareUniversity

- Software University Forums

- forum.softuni.bg



- This course (slides, examples, demos, exercises, homework, documents, videos, and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – <https://softuni.org>
- © Software University – <https://softuni.bg>

