



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

www.uwtsd.ac.uk

ACCA4020

Python – String Handling

Dr. Seena Joseph

String

- A string is a sequence of characters, enclosed in quotes (single, double, or triple quotes for multi-line strings).
- Example: Greeting="Hello"
- Name='Alice'
- Print (Name + ", " + Greeting)

String

Basic String operations

1. String concatenation

Combine string using the + operator

Sample Code

```
Name='Alice'  
Greeting="Hello"  
  
print(Greeting + "," + Name)
```

Output

Hello,Alice

String

Basic String operations

2. String Repetition

Repeat string multiple times using the * operator

Sample Code

```
Word= 'Python'  
print(Word * 3)
```

Output

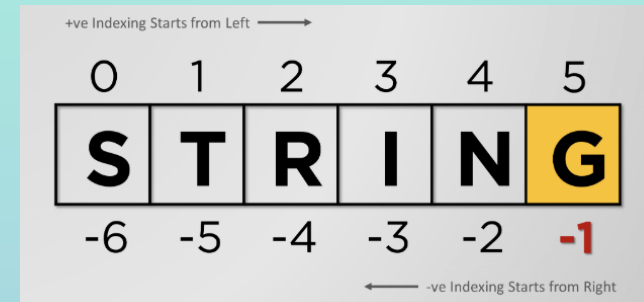
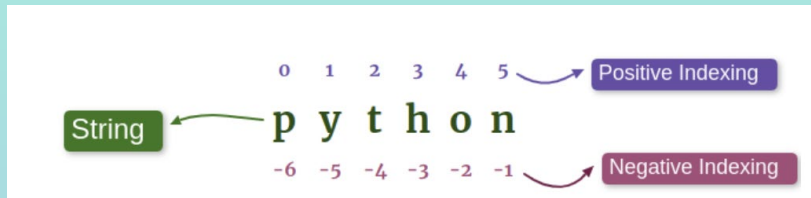
```
PythonPythonPython
```

String

String indexing

Can access individual characters in a string by using indices.

Indexing starts at 0 for the first character, and negative indices can be used to access characters from the end.



Sample Code

```
Word='Python'  
print(Word[0])  
print(Word[-1])  
print(Word[-2])
```

Output

P
n
o

String

- String Methods
- **len()**: returns length of the string

Sample Code

```
text1='Python'  
length1=len(text1)  
text2='I enjoy Python programming'  
length2=len(text2)  
print(f'The length of Text1 is {length1} and the length of Text2 is {length2}')
```

Output

```
The length of Text1 is 6 and the length of Text2 is 26
```

String

- String Methods
- **str.upper()** , **str.lower()**: converts to upper or lowercases

Sample Code

```
word='Python is a good Programming language'  
Upper=word.upper()  
Lower=word.lower()  
print(Upper)  
print(Lower)
```

Output

```
PYTHON IS A GOOD PROGRAMMING LANGUAGE  
python is a good programming language
```

String

String Methods

- **str.replace()**: Replace substrings
- `string.replace(oldstring, newstring, count)`
- **count (optional)**: The number of occurrences to replace. If not provided, it will replace all occurrences.

Sample Code

```
currentText='I like Python programming'  
updatedText=currentText.replace( __old: 'Python', __new: 'Java')  
print(updatedText)
```

```
txt = "one one was a race horse, two two was one too."  
newTxt = txt.replace( __old: "one", __new: "three")  
print(newTxt)
```

```
txt = "one one was a race horse, two two was one too."  
newTxt = txt.replace( __old: "one", __new: "three", __count: 1)  
print(newTxt)
```

Output

```
I like Java programming
```

Output

```
three three was a race horse, two two was three too.
```

Output

```
three one was a race horse, two two was one too.
```

String

- String Methods
- **str.find(), str.index()** : find the position of a string

```
Text='Hello Welcome to the world of Programming'  
pos=Text.find('Hello')  
print(f'The position of Hello is {pos}')
```

Output

```
The position of Hello is 0
```

```
Text='Hello Welcome to the world of Programming'  
pos=Text.find('Welcome')  
print(f'The position of Welcome is {pos}')
```

Output

```
The position of Welcome is 6
```

```
Text='Hello Welcome to the world of Programming'  
sub=input('Enter the string to be found :')  
pos=Text.find(sub)  
print(f'The position of {sub} is {pos}')
```

Output

```
Enter the string to be found :to  
The position of to is 14
```

```
Enter the string to be found :look  
The position of look is -1
```

Activity : Try the index function and understand the difference between find() and index()

String

- String Methods
- **str.strip()** : Removes the spaces at the beginning and at the end of the string

Sample Code

```
txt = "    Python    "  
print(f'{txt} is a good programming language')  
txt1 = txt.strip()  
print(f'{txt1} is a good programming language')
```

Output

```
    Python    is a good programming language  
Python is a good programming language
```

String

- String Slicing

- String[start:stop:step]
- Start: the index where the slice begins
- Stop: the index where the slice ends
- Step: The step value tells how many positions to move each time when extracting characters.

Sample Code

```
sWord=input('Enter a string :')  
slice=sWord[6:9]  
print(slice)
```

```
sWord=input('Enter a string :')  
slice=sWord[1:9]  
print(slice)
```

```
sWord=input('Enter a string :')  
slice=sWord[9:1:-1]  
print(slice)
```

```
sWord=input('Enter a string :')  
reverse=sWord[::-1]  
print(reverse)
```

Output

```
Enter a string :Hello how are you  
how
```

```
Enter a string :Hello how are you  
ello how
```

```
Enter a string :Hello hpw are you  
wph oll
```

```
Enter a string :Python  
nohtyP
```

String

- **in** and **not in** Operators : these are used to check if a substring exist within a string.
- **in** returns True if the substring is found in the main string else returns false
- **not in** returns True if the substring is **not** found in the main string else returns false

Sample Code

```
Sentence=input('Enter a sentence :')
SearchString= input('Enter the string to be searched :')
if SearchString in Sentence:
    print(f'{SearchString} is found')
else:
    print(f'{SearchString} is not found')
```

```
Sentence=input('Enter a sentence :')
SearchString= input('Enter the string to be searched :')
if SearchString not in Sentence:
    print(f'{SearchString} is NOT found')
else:
    print(f'{SearchString} is found')
```

Output

```
Enter a sentence :Welcome to Artificial intelligence
Enter the string to be searched :Artificial
Artificial is found
```

```
Enter a sentence : Welcome to computer programming
Enter the string to be searched :Python
Python is NOT found
```

String

- String Methods
- [Python - String Methods](#)

String

- You can create strings using single or double quotes.
- Concatenate strings using + and repeat using *.
- Access characters in a string with indexing (both positive and negative indices).
- String slicing – extract substring using [start:stop:step]
- String formatting - Use of f-strings for dynamic text formatting
- Python, strings are enriched with lots of functions that allow you to easily manipulate, format, and process them.

Exercise 1

Write Python program to perform the following tasks.

- Input a string and print it in uppercase and lowercase.
- Print the length of the input string
- Check if the input string contain a specific word
- Count the Number of Occurrences of a Specific Character in an input String
- Input a string and print the reverse of the string
- Check if an input string is a Palindrome