




INTRODUCTION TO PYTHON ALGORITHMS USING PYTHON

CHARIS CODING CLUB

WWW.CHARISCODINGCLUB.COM

COURSE OUTLINE

- LESSON1: INTRODUCTION TO ALGORITHMS, REFRESHER, GIT
- LESSON2: TACKLING ALGORITHMIC PROBLEMS – EUREKA! CHALLENGE
-  • LESSON3: STRING METHOD AND STRING CHALLENGE
- LESSON4: NEW PYTHON METHODS FOR SCRIPTING
- LESSON5: CHALLENGE 3 AND 4
- LESSON6: FUNCTIONS AND CLASSES
- LESSON7: CHALLENGE 5 AND 6
- LESSON8: SIMPLE GAME
- LESSON9: PROJECT PART 1
- LESSON10: PROJECT PART 2



LESSON 3: STRING METHODS AND STRING CHALLENGE

OBJECTIVES:

TAKE A CLOSER LOOK AT STRINGS

UNDERSTANDING STRING METHODS

BE ABLE TO USE BASIC STRING OPERATIONS METHODS

FROM THE LAST CLASS

- FROM THE LAST CLASS, WE LEARNED WE NOW KNOW HOW TO:
 - UNDERSTAND THE REQUIREMENTS
 - WRITE OUT YOUR THOUGHT PROCESS AND SOLUTION PROCEDURE
 - IMPLEMENT YOUR SOLUTION
 - IMPROVE YOUR SOLUTION.

TODAY WE'RE GOING TO LOOK AT STRINGS

FROM PYTHON BEGINNERS, WE LEARNED THAT:

- String types ("str") represent a string of characters.
- Strings can be concatenated with the + sign.
- Strings have some methods that can be used to modify the string.

Today, we will be looking at some string methods

STRING METHODS

There are many string methods. Here are some of them:

- ISALPHA() TO CHECK IF A STRING CONTAINS ONLY CHARACTERS AND IS NOT EMPTY
- ISALNUM() TO CHECK IF A STRING CONTAINS CHARACTERS OR DIGITS AND IS NOT EMPTY
- ISDECIMAL() TO CHECK IF A STRING CONTAINS DIGITS
- AND IS NOT EMPTY
- LOWER() TO GET A LOWERCASE VERSION OF A STRING
- ISLOWER() TO CHECK IF A STRING IS LOWERCASE
- UPPER() TO GET AN UPPERCASE VERSION OF A STRING
- ISUPPER() TO CHECK IF A STRING IS UPPERCASE
- TITLE() TO GET A CAPITALIZED VERSION OF A STRING

SOME EXAMPLES

Here are some example of how we access the methods in the string class:

```
# We use . Followed by the method name and brackets() to access the methods.
```

```
Word1 = "12345"
```

```
Word2 = "happy algorithms with Python"
```

```
Word = "Afrikan Spirit"
```

```
Word1.isalpha()      # Returns false
```

```
Word2.isalpha()      # Returns true
```

```
Word3.upper()        # Returns "AFRIKAN SPIRIT"
```

```
Word3.isupper()      # Returns false
```

WARM-UP EXERCISE

Write down the code below in your editor.

```
# Assigning four strings to variables

Word1 = "GREAT WOMEN AND MEN"
Word2 = "happy algorithms with Python"
Word3 = "Afrikan Spirit"
Word4 = "12345"

# Your code will start below:
```

Once done, let's move on to the next slide to perform our task.

WARM-UP EXERCISE

From the four strings you've written, write code that will give the following output from the variables in the previous slide

```
"great women and men"  
"Happy Algorithms With Python"  
"AFRIKAN SPIRIT"  
True
```

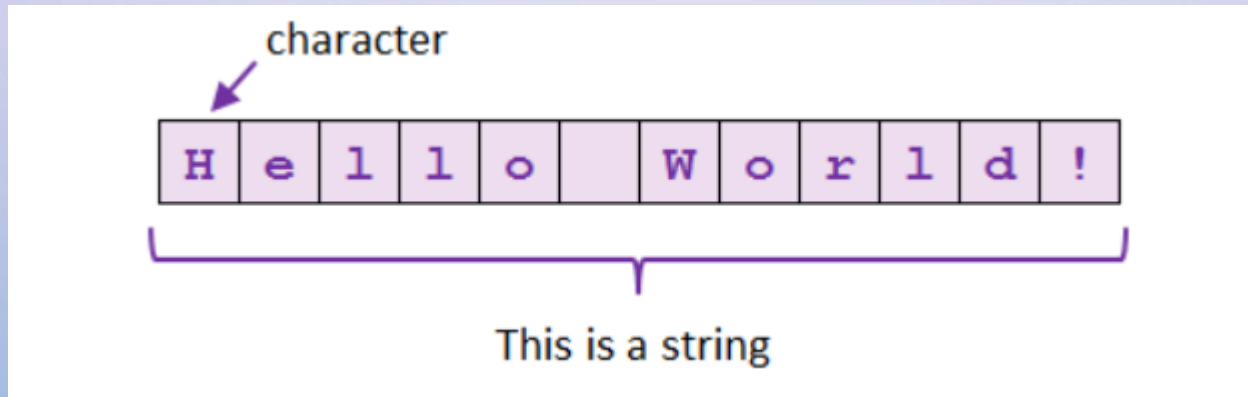
Use the methods learned about in this lesson to achieve the above output.

STRING SLICING IN PYTHON

Let's examine strings a bit more closely.

In computer science, a string is a collection of characters or a piece of text.

It is quite like a list, with each member of the list being a character.



String Length

The length of a string represents the number of characters it contains. The `len()` function is used to check the length of a string in Python.

```
password = input("Enter your password:")  
  
if len(password) < 8:  
    print("Your password is not long enough!")
```

String Slicing

On occasions, you will want your program to **extract a few characters from a string**. This is called **string slicing**.

You may for instance extract the first 5 characters of a phone number to retrieve the area code. Or you may need to extract the last 4 characters of a date to extract the year part.

Extracting a single character at a given position:

You can retrieve a character at a given position as follows:

```
string[position]
```

String Slicing

In the example below firstCharacter and lastCharacter will hold letters “A” and “Z” respectively:

```
alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"  
firstCharacter = alphabet[0]  
lastCharacter = alphabet[25]
```

Remember counting starts from zero.

Similar to lists, right?

String Slicing

Extracting a substring (multiple characters) from a string:

To slice a string in Python you have to think about the starting position and the ending position and use the following syntax:

string[start:end]

String Slicing

This syntax will extract all the characters from position *start* up to **but not including** position *end*.

If *start* is left empty, the substring will start from position 0.

If *end* is left empty, the substring will end at the last character of the string.

A **negative** *start* value or *end* value is used to identify a position (number of characters) from the end of the string. This can be useful to extract the last few characters of a string.

String Slicing Examples

Slicing the alphabet:

```
alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
print("The first 5 letters of the alphabet are: " + alphabet[:5])
print("The last 5 letters of the alphabet are: " + alphabet[-5:])
print("The letters in between are: " + alphabet[5:-5])
```

Slicing a date:

```
date = "05 January 2019"
day = date[:2] #First 2 characters
year = date[-4:] #Last 4 characters
month = date[3:-5] #The remaining characters in the middle
```

Locating if a substring is in a string (and retrieving it's position) using "in"

You can find if a substring is in a string using the keyword *in*

```
date = "05 January 2019"  
  
"January" in date    #returns true  
"February" in date   #returns false
```

Slicing a date example:

```
dateOfBirth = "15 January 2000"  
  
if "January" in dateOfBirth:  
    print("You were born in January!")
```

Locating if a substring is in a string (and retrieving it's position) using "find()"

You can also find if a substring is in a string using the keyword *find()*

```
alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"  
pos = alphabet.find("C")  
print("Letter C is at position: " + str(pos))
```

This script would tell us that letter "C" is at position 2. (Even though it is the third letter of the alphabet). That's because the first letter in a string is always at position 0, not 1!

Note that if the substring is **not found** in the string, the *find()* function will return the value **-1**

Try running the code below to see the output.

```
alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

if "@" in alphabet:
    print("Letter @ is in the alphabet!!!")
else:
    print("Letter @ is not in the alphabet!!!")

posAt = alphabet.find("@")
print("The position of @ in the alphabet is: " + str(posAt))
```

Now, Include the @ in the alphabet and see the result.

Python Challenge: Username Validation

Your challenge is write a program to ask the user to enter their school username in the following format:

- Year Group Using two digits (e.g. “07” for year 7, “11” for year 11, “00” for staff members)
- 1 character for their initial (first letter of their firstname)
- The user’s lastname
- A 2-digit code: “_S” for students, “_T” for teachers, “_A” for admin staff.

Python Challenge: Username Validation

For instance the following usernames are valid usernames:

07jFox_S
09kJohnson_S
11rTaylor_S
00pJones_T
00jOliver_A

Your Username Validation program should read the username and decide:

- If the username is less than 6 characters long the program should ask the user to enter a valid username.
- If the username does not contain the character “_” it should also ask the user to enter a valid username.
- If the username is valid, the program should decide if the user is a member of staff or a student.
- If they are a student the programme should find out their year group.
- The program should also display the initial of the student as well as their lastname.
- Finally the program should display whether the user is a Student, a Teacher or an Admin member of staff.

QUESTION AND ANSWER TIME!

- ANY QUESTION?

GOOD LUCK WITH THE ASSIGNMENT!

SEND IN YOUR ANSWERS ON OR BEFORE **THURSDAY** NEXT WEEK