Hyperion Development

Task: String Handling

www.hyperiondev.com

Introduction

Welcome to the Strings Handling Task!

You have probably already noticed what an important part strings play in programming. The one of the most crucial concepts to grasp when it comes programming is string handling. We went over this already in the Strings Task, so this task just serves as a brief recap and will involve more advanced programs which use more functions and programming techniques.

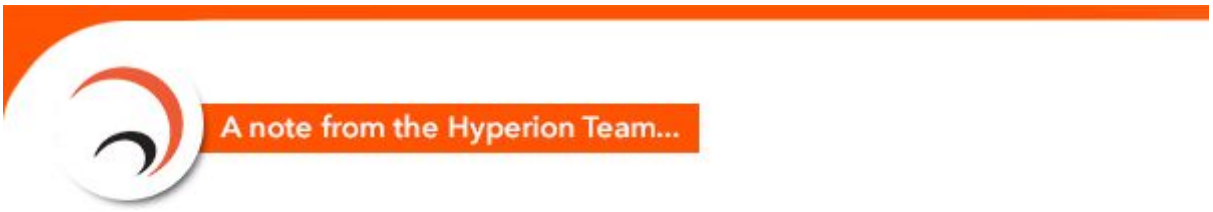
Connect with your mentor

CONNECT

Remember that with our courses - you're not alone! You can contact your mentor to get support on any aspect of your course.

The best way to get help is to login to www.hyperiondev.com/support to start a chat with your mentor. You can also schedule a call or get support via email.

Your mentor is happy to offer you support that is tailored to your individual career or education needs. Do not hesitate to ask a question or for additional support!



The Best Qualification for Programmers.

This article on the Hyperion Hub serves to answer that question we get asked time and time again - 'What degree should I study if I want to become a software developer?'. We previously published an article on the high level-differences between IT and CS. In this article, we'll compare and contrast the employability, salaries, and career paths for IT versus CS qualifications, and answer which qualification is the best for programmers.

Indexing Strings

You can think of the string 'Hello world!' as a list and each character in the string as an item with a corresponding index.

```
'  H  e  l  l  o     w  o  r  l  d  !  '
   0  1  2  3  4  5  6  7  8  9  10 11
```

The space and exclamation point are included in the character count, so 'Hello world!' is 12 characters long, from 'H' at index 0 to '!' at index 11.

```
String = Hello
ch[0] = H
ch[1] = e
ch[2] = l
ch[3] = l
ch[4] = o
```

Remember that if you specify an index, you'll get the character at that position in the string. You can also slice strings by specifying a range from one index to another, the starting index is included and the ending index is not.

Note that slicing a string does not modify the original string. You can capture a slice from one variable in a separate variable. Try typing the following into the interactive shell:

```
newString = 'Hello world!'
fizz = newString[0:5]
print fizz
```

By slicing and storing the resulting substring in another variable, you can have both the whole string and the substring handy for quick, easy access.

String Methods

Once you understand strings and their indexing the next step in the progression is mastering some of the common string methods. These are built in code that perform certain operations on strings. These methods are really useful as they save you time from having to write the code over and over again in order to perform certain operations. The most common of these methods are:

- **s.lower()** and **s.upper()** - these convert a string to either the upper or lowercase version.
- **s.strip()** - removes any white spaces at the beginning or end of a string.
- **s.find('text')** - this will search for a specific text and return its position inside a variable you are searching in.
- **s.replace('oldText', 'newText')** - This will replace any occurrence of the oldText with the newText.
- **s.split('word')** - This breaks down a string into a list of smaller pieces. The string is separated based on what is called a delimiter. This is a string that is given to the method, it can be a string or a character like a word or a comma. If no value is given it will automatically split the string by white spaces.

Escape Character

Python uses the backslash (\) as an escape character. The backslash (\) is used as a marker character to tell the compiler/interpreter that the next character has some special meaning. The backslash together with certain other characters are know as escape sequences

Some useful escape sequences can be found below:

- **\n** - Newline
- **\t** - Tab
- **\s** - Space

The escape character can also be used if you need to include quotation marks within a string. You can put a backslash (\) in front of a quotation mark so that it doesn't terminate the string. You can also but a backslash in front of another backslash if you need to include a backslash in a string.

A note from Masood…

*Hey again, Have you ever wondered where a string variable actually gets it's name from? According to an [article on StackOverflow](): *

The 1971 OED (p. 3097) quotes an 1891 Century Dictionary on a source in the Milwaukee Sentinel of 11 Jan. 1898 (section 3, p. 1) to the effect that this is a compositor's term. Printers would paste up the text that they had generated in a long strip of characters.

- **Masood Gool**, Online Trainer

Instructions

Before you get started we strongly suggest you start using Notepad++ or IDLE to open all text files (.txt) and python files (.py). Do not use the normal Windows notepad as it will be much harder to read.

First read example.py, open it using Notepad++ (Right click the file and select 'Edit with Notepad++') or IDLE.

- example.py should help you understand some simple Python. Every task will have example code to help you get started. Make sure you read all of example.py and try your best to understand.

- You may run example.py to see the output. Feel free to write and run your own example code before doing the Task to become more comfortable with Python.

- You are not required to read the entirety of Additional Reading.pdf, it is purely for extra reference.

Compulsory Task 1

Follow these steps:

- Create a program called "alternative.py" that reads in a sting and makes each alternate character an Uppercase character and each other alternate character a lowercase character.

Compulsory Task 2

Follow these steps:

- Write a Python program called "counting.py" to count the number of characters (character frequency) in a string.
- Store each letter followed by the number of occurrences in a list and print it out.
- Sample String : 'google.com'
- Expected Result : {'o': 3, 'g': 2, '.': 1, 'e': 1, 'l': 1, 'm': 1, 'c': 1}

Compulsory Task 3

Follow these steps:

- Write a program called "seperation.py" which inputs a sentence and displays each word of the sentence on a separate line.

Compulsory Task 4

Follow these steps:

- Write a Python program called "disappear" to strip a set of characters from a string.
- Ask the user to input a string and then ask the user which characters they would like to make disappear.
- For example: The quick brown fox jumps over the lazy dog.
- After stripping a,e,i,o,u
- Th qck brwn fx jmps vr th lzy dg.

Things to look out for:

1. Make sure that you have installed and setup all programs correctly. You have setup **Dropbox** correctly if you are reading this, but **Python or Notepad++** may not be installed correctly.
2. If you are not using Windows, please ask your mentor for alternative instructions.

Give your thoughts..

Hyperion strives to provide internationally-excellent course content that helps you achieve your learning outcomes. Think the content of this task, or this course as a whole, can be improved or think we've done a good job?

Click here to share your thoughts anonymously.