COMP2152 LAB MANUAL

OPEN SOURCE DEVELOPMENT

This booklet will help the reader understand the concepts, principles, and implementation of the Python programming language. By the end of the booklet, the reader will be able to code comfortably in Python.

Prepared by Ben Blanc

TABLE OF CONTENTS

CHAPTER 4

STRING FUNCTIONS

CHAR ⇔ ASCII VALUE

A string is a series of character.

You can create a string from an ASCII code by:

chr(ASCII_VALUE_INTEGER)

```
output = chr(65)

print(output)
```

You can output the ASCII value of a string by the following code:

ord(CHARACTER_STRING)

```
output = ord('A')

print(output)
```

REPEATING STRINGS

Use the multiplication character (*) as a repetition operator in the print method.

```
print("Hello there! " * 2)

print("You have earned", "*" * 5 , "stars")
```

CHAPTER 4 AT A GLANCE

In this chapter you will learn how to use string functions. You will be introduced to how to:

- Search a string
- Split/slice a string
- Format/align strings
- And more..

STRING AS COLLECTION OF CHARACTERS

A string is an array of characters.

```
str = "Hello"

print( len(str) )
```

The following code will get the first character of a string

```
print( str[0] )
```

The following code will get the last character of a string

```
print( str[ -1 ] )
```

OR

```
print( str[ len(str) - 1 ] )
```

Accessing an index out of range results in an error

```
print( str[10] )
```

This code above produces an IndexError

Attempting to change a character will also result in an error because strings are immutable.

```
str[1] = "A"
```

Code to iterate through the characters of a string would be the following

```
for c in str:
    print( c )

for c in str:
    print("The ASCII value for", c, "is", ord( c ))
```

SLICING STRINGS

You can slice a string by using the braces in after the string variable. String slicing takes one mandatory argment and two aditional, optional arguments:

string_variable[start :]

```
print("The last three characters are", str[2:] )
```

Notice the colon (:) at the end of the braces. If not present, it would only display the third character of the string.

```
print("The last three characters are", str[-3:] )
```

String slicing can also take two arguments.

string_variable[start : end (exclusive)]

```
print("Characters 2-4 are", str[1:4] )
```

Notice the trailing colon (:) character is not needed when two arguments are passed.

String slicing can also take two arguments.

string_variable[start : end (exclusive) : incrimental_value]

```
print("Every other character is", str[0: len(str) :2] )

print("Every second character is", str[1: len(str) :2] )
```

SEARCHING STRINGS

Use the **in** operator to search if a substring is present in a string. The format is

search IN string

It returns a Boolean value, so it should be used in a conditional statement.

```
str = "Hello"

search = "e"

if search in str :
    print(search, "was found in", str)
```

STRING METHODS

There are many operations and processing you can do with the string data type. Below is a chart of some of the most common

Method/Property	Description	Argument(s)	Return	Example

split()	Separates the string into a list of strings	1)String separator. Default separator is the space character
		2) Int max amount of splits. (Optional)
			List	`str = "abc def fed cba"`
				`new_str = str.split()`
				`print(new_str)`
				
				`str = "abc def fed cba"`
				`new_str = str.split(' ',2)`
				`print(new_str)`

title()	Transforms the first character of every word to a capital letter	None	String	`str = "a brave new world"`
				`new_str = str.title()`
				`print(new_str)`

endswith(), startswith()*	Determines if a string ends of starts with a string	1)String search string
		2)Int starting position (optional)
		3)Int ending position (optional)
			Boolean	`str = "Hello"`
				`ans1 = str.endswith("o")`
				`ans2 = str.endswith("o",1,3)`
				`print(ans1)`
				`print(ans2)`
				`ans3 = str.startswith("H")`
				`ans4 = str.startswith("H",1,3)`
				`print(ans3)`
				`print(ans4)`

find()	Returns index position of argument (starting from the beginning at index 0) or -1 if not found	1)String search string
		2)Int starting position (optional)
		3)Int ending position (optional)
			Int	`str = "Hello"`
				`pos1 = str.find("e")`
				`pos2 = str.find("e", 2, 4)`
				`print(pos1)`
				`print(pos2)`

lower()	Converts the string into the lower case	None	String	`str = "Hello"`
				`s_lower = str.lower()`
				`print(s_lower)`

upper()	Converts the string into the upper case	None	String	`str = "Hello"`
				`s_upper = str.upper()`
				`print(s_upper)`

islower(), isupper(), istitle()	Determines if string is all lowercase, all uppercase, or in title format	None	Boolean	`str = "Hello"`
				`print(str.isupper())`
				`print(str.islower())`
				`print(str.istitle())`

Method/Property	Description	Argument(s)	Return	Example

lstrip(), rstrip(), strip()	Trims the whitespace from a string on the left side, right side or both sides of a string	None	String with whitespace removed	`str = "   Hello   "`
				`print(str.lstrip())`
				`print(str.rstrip())`
				`print(str.strip())`

isalpha()	Determines if the string only contains alphabetical characters	None	Boolean	`str = "Hello123"`
				`print(str.isalpha())`
				
				`str = "Hello"`
				`print(str.isalpha())`

isdigit()	Determines if the string only contains numerical characters (negative numbers and decimals will return False)	None	Boolean	`str = "234"`
				`print(str.isdigit())`
				
				`str = "-234"`
				`print(str.isdigit())`
				
				`str = "234.56"`
				`print(str.isdigit())`
				
				`str = "-234.56"`
				`print(str.isdigit())`

isalnum()	Determines if the string only contains alpha-numerical characters			`str = "Hello123"`
				`print(str.isalnum())`
				
				`str = "Hello!"`
				`print(str.isalnum())`