String Manipulation

A string is the technical name for a series of characters, or a piece of text. A string variable, therefore, is a variable which stores a word, words, or series of characters from the keyboard. These are usually stored as ASCII. In most programming languages we indicate a piece of data is a string by enclosing it in single or double quotation marks.

Question 1 - In a computer program, why does “8” not equal 8?

Programming languages have various built in functions that allow us to manipulate strings.

Concatenation

The simplest string manipulation function is concatenation - it simply sticks two strings together. Usually a single character is used to concatenate.

In Python, we just use the + symbol to concatenate:

```
Firstname = ‘Bob’
Lastname = ‘Smith’
Fullname = Firstname + “ “ + Lastname
```

In Pseudocode the & symbol is used:

```
SET Firstname TO ‘Bob’
SET Lastname TO ‘Smith’
SET Fullname TO Firstname & “ “ & Lastname
```

Length

It is often useful to find out how long a string is. The Length function does this.

In Python:

```
StrLength = len(Fullname)
```

In Pseduocode:

```
StrLength = LENGTH(Fullname)
```

Indexes of a string

If we wish to refer to a single letter or character within a string we can do so through its index number. This works the same as the index of an array - the first character is 0, the second

character 1, and so on. So, for example, in the string ‘London’, L has a n index of 0, d has an index of 4.

String traversal

To traverse is to move across, or through. When we talk about string traversal we are talking about moving through a string letter by letter. We can use these techniques to search a string to see if it contains a particular character.

If we wished to print each character of a string one by one, we could do this as follows. In Python:

```
stringLength= len(myString)
for i in range (0,stringLength):
        Character = myString(i)
        print(Character)
```

Or in Pseudocode:

```
stringLength= LENGTH(myString)
FOR index FROM 0 TO stringLength-1 DO
        SET Character TO myString(index)
        SEND Character TO DISPLAY
NEXT FOR
```

Alternatively, we can use a For Loop to move directly through a string, python makes this particularly easy. The following code would do the same as the above -

```
for char in myString:
        print(char)
```

This would automatically work through one character after another.

In Pseudocode, this could be written using the ‘for each’ structure:

```
FOR EACH char FROM myString DO
        SEND char TO DISPLAY
END FOR
```

These techniques are easier, but less clear. They also don’t keep an index value as they look, which might be useful for some algorithms.

Question 2 - How would you use the above technique to search an email address to check it contains the @ symbol?

Cutting up a String

In Python you can use the index to return substrings. By placing two numbers, separated by a colon, in square brackets you can specify which characters from a string you wish to return. E.g.

```
Str = “Ealing”
print(Str[2:4])
```

This would print li

```
Str = “Ealing”
print(Str[0:3])
```

This would print Eal

You don’t actually need to include the 0 in this case - a blank is taken to mean ‘from the beginning’ or ‘to the end’, e.g.

```
Str = “Ealing”
print(Str[:4])
```

This would print Eali

```
Str = “Ealing”
print(Str[3:])
```

This would print ing

You can also specify a ‘step count’, just as you can in a for loop. E.g.

```
Str = “Ealing”
print(Str[1:6:2])
```

This would print aig

The most useful thing to do with this is reverse the order of a string. E.g.

```
Str = “Ealing”
print(Str[6:1:-1])
```

This would print gnil

Question 3 - what would the following code print:

```
Str = “That is not dead that can eternal lie”
print(Str[4:12])
```

Question 4 - what would the following code print:

```
Str = “That is not dead that can eternal lie”
print(Str[3:16:3])
```

Question 5 - complete the code so that it will print out “ed ton si”

```
Str = “That is not dead that can eternal lie”
print(Str[])
```

Other String functions

The specific string functions available vary greatly from one programming language to another. Python has a lot of these (here's a list - https://www.w3schools.com/python/python_ref_string.asp). The most commonly used include:

upper() - Converts a string into upper case
lower() - Converts a string into lower case
isalpha() - Returns True if all characters in the string are in the alphabet
split() - Splits the string at the specified separator, and returns a list (i.e. an array)
index() - Searches the string for a specified value and returns the index of the position of where it was found
find() - Searches the string for a specified value and returns the position of where it was found