CPSC 231 Tutorial

Strings

What is a string?

- A string is simply a sequence of characters, for example:
  - hello
  - asdf
- You have already used strings in python, something like this:
  - str1 = “this is a string”
  - print(str1)

string1.py

```
index = int(input("Enter index value (0-12): "))
name = "Homer Simpson"
print(name)

if (index >= 0) and (index <= 12):
 print(name[index])
else:
 print("%d is outside the bounds of 0 - 12" %index)
```

- We can access single characters in a string using and **index** and **square brackets [ ]**
- The lowest index value we can use is 0, str[0] would be the first character in str

Immutability of Strings

- In python strings are said to be **immutable**
- This means that once a string is created it cannot be modified
- Variables can reference strings :
  - var = “hello”
  - var = var + “, how are you?” #new string created
- When we apply concatenation a **new** string is created and the variable would now reference this **new** string, the previous string has not been modified in memory

string2.py

```
hero = "Bart" # Creates string and reference
temp = hero # Two references to same string
hero = hero + " Simpson" # Concat creates new string, old string
still exists

print(temp)
print(hero)

hero[1] = 'A' # Error: strings are immutable
hero[2] = 'R'
hero[3] = 'T'
```

String Concatenation

- String concatenation uses the ‘+’ operator and is essentially like “gluing” two strings together

```
str1 = "hello"
str2 = ", how are you?"

str3 = str1 + str2     #str3 = "hello, how are you?"
```

String Slicing

- Format: [<start> : <end>]

```
var = "hello world"
#str[start:end]
print(var[0])
print(var[1:5])
print(var[2:])
print(var[:5])
#output
"""
h
ello
llo world
hello
"""
```

String Splitting

- str.split([sep [,maxsplit]])
- returns a list of words in str that have been separated by sep
  - When sep is not specified the default value is a space and all spaces will be stripped
- If maxsplit is given, at most maxsplit splits are done meaning you will have a list of at most maxsplit + 1 elements
  - If maxsplit = -1 or is not specified then there is not limits on number of splits

strings3.py

```
name = "James"
name = name + "T. Kirk"

print(name)
print(name[0:5])
print(name[2:])
print(name[:3])

first, second = name.split('.')
print(second,first)
```

string4.py (review)

```
a = 12
b = '3'
x = a + int(b) # Two numbers: okay (fast)
print(x)

y = str(a) + b # Two strings: okay (fast)
print(y)

x = b + 1 # Mixed types not okay
```

String Iteration

- We can iterate through characters in a string using a for loop:

```
sentence = "this is a sentence"

for aChar in sentence:
    print(aChar)
```

Character Comparison

- If we want to check if a particular character in a string is equal to some letter (upper or lower case) instead of checking for each case we can first convert to uppercase and check only once

```
sentence = "this is a sentence"

for aChar in sentence:
    aChar = aChar.uuper()
    if aChar == "E":
        print("Character is an e!")
```

Excercise

- Write a program that prompts user for a sentence and then counts the number of vowels in that sentence and outputs results to screen
- Vowels are a, e, i, o, and u

Solution

```
count = 0
sentence = input("Enter a sentence: ")

for aChar in sentence:
 aChar = aChar.upper()
 if (aChar in ('A','E','I','O','U')):
    count = count + 1

print("Number vowels in sentence %d" %count)
```

strings.py

```
aString = "mErrY MEn!"
upper = aString.upper()
lower = aString.lower()
print("Original: %s\tMeek: %s\tShouting: %s" %(aString,lower,upper))

emphasized = aString.replace("!","!!!")
emphasized = emphasized.replace("mErrY","merry :-) :D =)")
print("Original: %s\tWith emphasis: %s" %(aString,emphasized))

subString = aString[1:3]
print("Original: %s\tStripped: %s?" %(aString,subString))
```

strings.py (cont)

```
# New string full of trash
aString = "    xy z. "
print("Original: '%s'\tLeft: '%s'\tRight: '%s'"
      %(aString,aString.lstrip(),aString.rstrip()))

# Changing tabs to spaces
aString = "\ta\t\tb\t\t\tc"
print("""
0123456789012345679012345679012345679012345679012345679""")
print("Original=%s" %(aString))
print("2 spaces=%s" %(aString.expandtabs(2)))
print("8 spaces=%s" %(aString.expandtabs(8)))
```