TCSS 142 – Introduction to Programming

Autumn 2014
Day 13

Day 13 Overview

- Programming Assignment 1

- Strings
  - indexes
  - slicing
  - repetition

Programming A 1

- Questions?

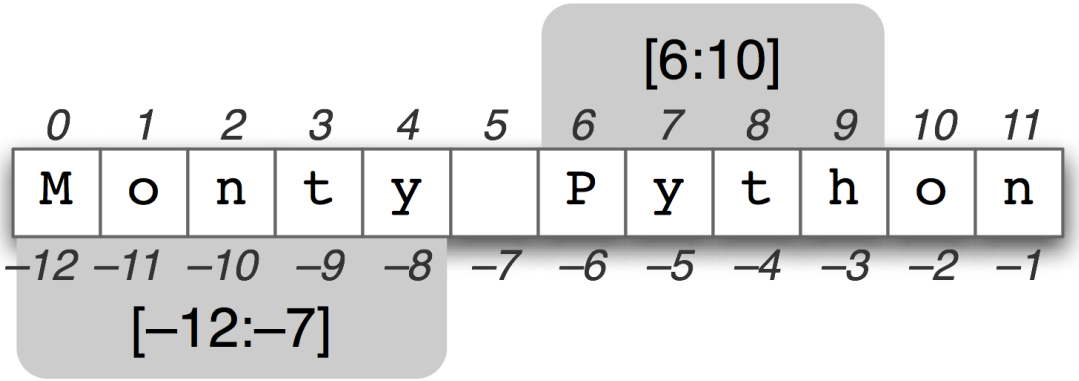
Strings Summary

- What have we learned about strings so far?

More on Strings

- Characters of a string are numbered with *indexes*:

```
phrase = "Monty Python"
```



  - First character's index : 0 or -length
  - Last character's index : length-1 or -1

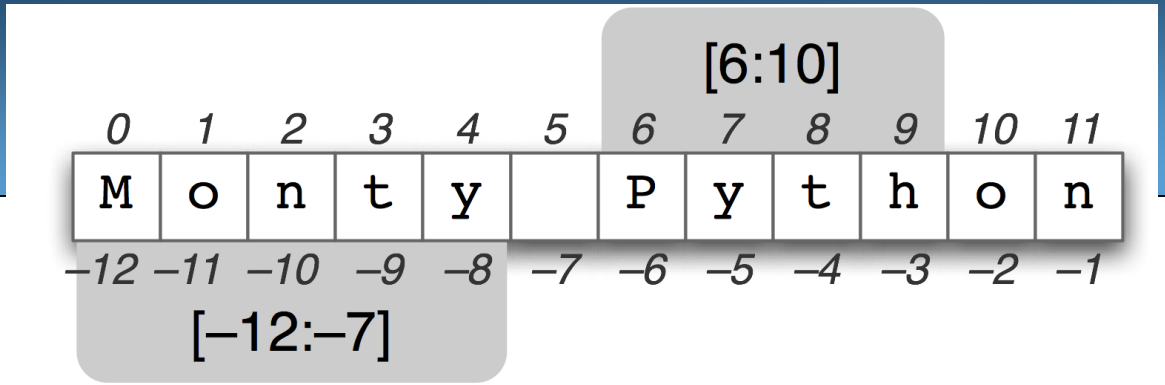
Index

- When we want to get a particular character from a string we use [ ] operator.

```
>>> phrase = "Monty Python"
>>> first1 = phrase[0]
>>> first2 = phrase[-??]
>>> whatAmI = "Monty Python"[6]
>>> print (phrase[20])
>>> len(phrase)
>>> phrase[0] = 'm'
```

Slicing

- Slicing selects a range of characters from a string and returns them as a string
  - uses `[:]` operator
  - format `stringVar[start:end]`
  - `start` – inclusive, `end` – exclusive
  - if start missing, then Python uses a zero, e.g. [: 10] means extract `0 to 9`
  - if end missing, uses string length, e.g. `[5 : ]` means extract `5 to length-1`




```
>>> phrase[0:5]
??
>>> word = phrase[6:12]
>>> word
??
>>> phrase
>>> phrase[:5]
???
>>> phrase[6:]
???
>>> phrase[:]
???
>>> for i in range(len(phrase)+1):
        print(phrase[0:i])
```

Repetition

- To repeat a part of a string, use *

```
>>> 'bon' * 2
bonbon
>>> candy = ("bon" * 2) + 's'
>>> candy
bonbons
```

in and not in

- There are two more string operators
  `in` – determines if one strings contains another
  `not in` – determines if one string does not contain another

```
>>> 'bc' in "Monty Python"
False
>>> 'bc' in phrase
False
>>> 'bc' not in phrase
True
>>> for ch in phrase:
      print(ch)
```

Exercise

- Given the following code, what indexes must be used instead of `a` and `b` to produce the new string with the value `SCORE` ? What indexes to produce `fouryears` ?

```
quote = "Four score and seven years ago"
expr1 = quote[a : b].upper()          #  "SCORE"
expr2 = quote.lower()[a : b] + quote[a : b]
          # "fouryears"
```

Exercise

At the command prompt, declare these variables

```
str1 = "Frodo Baggins"
str2 = "Gandalf the GRAY"
```

And then evaluate the following expressions:

```
len(str1)
str1[7]
str1.find('B')
str1.lower().find('B')
print(str1)
str1[ : 4]
str1 = str1.upper()
print(str1)
```

```
str2[0]
str2.upper()
str2[3:14]
str2[12:]
str2[12:20]
print(str2)
```

Gangsta Name (again)

- Let's redo `gangstaname.py` now that we know slicing
  - step 1 – download and examine
  - step 2 – save as `gangstaslice.py` and redo

Transposition Cipher

	0123..
Original:	It was a dark and stormy night
Even/Odd:	It_was_a_dark_and_stormy_night
Message:	twsandr_n_tryngtI_a__akadsom_ih

Transposition Decryption

- Algorithm
  - Split the message into half
  - Place the first half characters into odd positions of a new string
  - Place the second half characters into even positions of a new string
  - What if the original is of odd/even length?

- Let's add a function to `transposeFunctions.py` that does the decryption

Typical String Ops

- Remove duplicate letters

```
def removeDupes(myString):
    newStr = ""
    for ch in myString:
        if ch not in newStr:
            newStr = newStr + ch
    return newStr
```

Typical String Ops

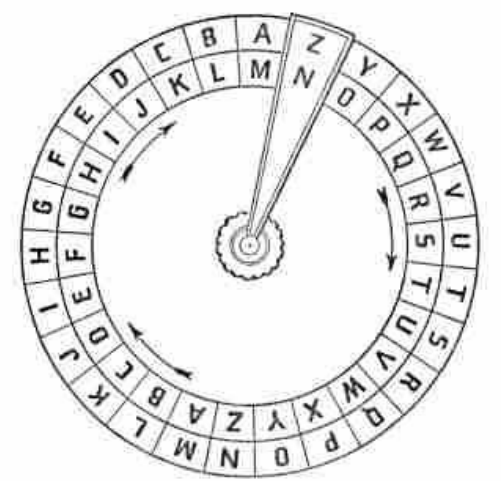
- Remove certain letters (as specified in the second param) from the first string

```
def removeMatches(myString, removeString):
    newStr = ""
    for ch in myString:
        if ch not in removeString:
            newStr = newStr + ch
    return newStr
```

Substitution Cipher

Substitutes one letter for another – uses a key (align two alphabets and shift one of them)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M



CODE WHEEL FOR REVERSE CODES

Encryption Algorithm

- Find the position of the letter in the alphabet
- Use the position from the prior step to find the new letter in the key
- Append the key letter to the string representing a cipher

Substitution Cipher

```
>>> alphabet = "abcdefghijklmnopqrstuvwxyz"
>>> key = "nopqrstuvwxyzabcdefghijklm"
>>> i = alphabet.index('d')
>>> cipher = ""
>>> cipher = cipher + key[i]
>>> i = alphabet.index('o')
>>> cipher = cipher + key[i]
>>> i = alphabet.index('g')
>>> cipher = cipher + key[i]
>>> cipher
```

Code

- Let's create a new program called `Caeser.py` and the encryption function

- Add interactive input

Exercise

- Add decryption function

- Write a program `capitalizer.py` that asks for a sentence and then changes the first character of each sentence to uppercase. For example, if

```
hello. my name is joe. what is your name?
```

is entered, the program produces

```
Hello. My name is joe. What is your name?
```

Make your program modular. Hint: check the Python library for a method that will make your job easier.

Last Slide ☺

- No class next Tuesday, Nov. 11.
- Read chapter 6.1 – 6.3 and 7.1 – 7.7 and complete the quiz by the next class meeting on Thursday.
- Class ends at 17:10