PLAN 396
Lecture 11

Dr. Hossen Asiful Mustafa

https://hossenmustafa.buet.ac.bd

Sequence Operators and Functions With Strings

- Remember, a string is a type of sequence
- The function len() returns the length of a sequence (or string)
- The operator “in” test if an element is in a sequence

Indexing Strings

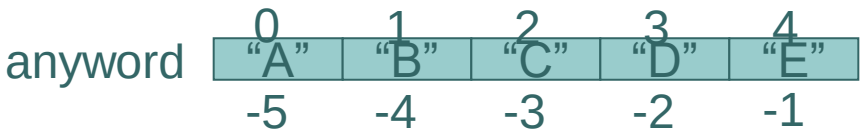
- Looping through a sequence or string character by character is an example of sequential access
- The index operator [] allows for random access of a sequence
  - Syntax:

```
anyword[i]
Index starts at 0
```

Indexing Strings

- An index is a position in the sequence
  - In Python position can be positive or negative
  - anyword[1] == anyword[-4]

- Run-time error if the index is out of range!



Indexing Strings

- Here is a code fragment to access a random sequence/string element

```
anyphrase = “I’d like to have an argument”
high = len(anyphrase)
position = random.randrange(0, high)
print(anyphrase[position])
```

String Immutability

- Sequences are either mutable or immutable
  - Mutable means changeable
  - Immutable means unchangeable

- Strings are immutable sequences

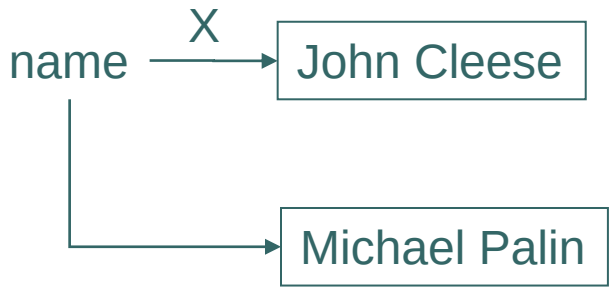
String Immutability

- Consider the following example

```
>>> name = “John Cleese”
>>> print (name)
John Cleese
>>> name = “Michael Palin”
>>> print (name)
Michael Palin
```

- Did we change the string?
  - No, we just assigned a new string to name

String Immutability



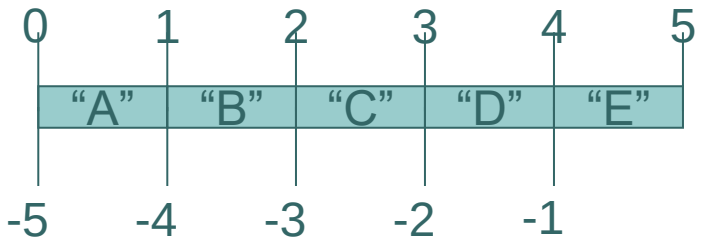

```
#Runtime Error in Python
word = “game”
word[0] = “n”

runtime error!!!!!!!
```

Building A New String

- If your program needs to build a new string it must do so using the string concatenation operator (either + or +=)
  - W1=“123”
  - W2=“abc”
  - W3=W1+W2

Slicing Strings

- Indexing allows access to a single element “slice” from a sequence (string)
- A slice is any consecutive part of a sequence (string)
- Slicing is similar to indexing except you can get a sub-string from the string
  - use two index values
  - anystring[low:high]



Creating Tuples

- A tuple is a sequence that can hold elements of any type
  - Tuples can hold integers, real numbers, strings, lists, dictionaries, and other tuples all at the same time
  - Tuples are immutable

Creating Tuples

- To create a empty tuple

```
anytuple = ()
```

  - an empty tuple is considered to be False

```
if not anytuple
    print “The tuple is not empty”
```

- To create a tuple with elements

```
days = (“Mon” , “Tue”, “Wed”, “Thu”, “Fri”,
  “Sat”, “Sun” )
```

More on Tuples

- To print a tuple

```
print anytuple
```

  - Python will display each element of the tuple surrounded by parenthesis

- To loop through each element in a tuple

```
for item in anytuple
    print item
```

Using Tuples

- Using the in operator with tuples
  - This will test “membership” if an element is in a tuple

```
if “Mon” in days
    print “Monday is a days of the week”
```

Using Tuples

- Remember tuples are immutable
  - days[1] = “MON” will give a runtime error
- Tuples can be constructed or modified by using concatenation (just like strings)
  - Use the + or += operator to add new elements to the end

Using Tuples

- Indexing tuples
  - Just like indexing strings
  - Example:
    - print days[1]
    - print days[4]
- Slicing tuples
  - Just like slicing strings
    - print days[2:3]
    - print days[4:]
    - print days [-4:-1]

Class Assignment

- Write a program named classassignment14.py
- The program should:
  - Define 2 lists with 5 elements in each list
  - Generate a new list with unique combinations of the 2 lists
  - Example
    - L1 = [“A”,”B”,’C”]
    - L2 = [“A”,”B”, “D”]
    - RESULT=[“AA”,”AB”, “AD”, “BA”,”BB”, “BD”,”CA”,’CB”,”CD”, “AC”, “BC”,”CA”,’CB”,”CD”,”DA”,”DB”,”DC”]
    - Print all the lists
    - Use append function to add to list
      - RESULT. append (“xyz”)