Lecture 16 – the split() method, updating the elements of lists, lists are mutable objects

Recap
From lecture 14
- we can iterate through the elements of a list (visit each element) using a for...in loop
- calculations can be done using the values in the elements of a list

```python
def start_with_vowel_count(a_list):
    vowels = "aeiouAEIOU"
    count = 0
    for word in a_list:
        if word[0] in vowels:
            count += 1
    return count

def main():
    my_list = ['Nobody', 'goes', 'to', 'that', 'restaurant', 'because', 'it', 'is', 'too', 'crowded']
    vowel_starters = start_with_vowel_count(my_list)
    print("Start with a vowel", vowel_starters)
main()
```

Learning outcomes
At the end of this lecture, students should be able to:
- use the index number to access individual elements of a list
- make changes to the elements of a list
- copy the values of a list
- use the split() method on a string to obtain a list of string objects
- lists are mutable objects

Accessing elements from the end of a list
A negative index value can be used to access the elements from the end of a list.

```python
my_list = [10, 20, 30, 40, 50]
print(my_list[-4])
my_list[-3] = my_list[-1] + my_list[-2]
print(my_list[-3], my_list[1], my_list[-5])
```

```
20 90 20, 10
```
Why does the following not work as intended?
In the following for...in loop, each element of the list is accessed but ...

```python
def main():
    a_list = [10, 8, 6, 4, 7]
    print("1.", a_list)
    for number in a_list:
        number = number * 2
        print(number, end=" ")
    print()  
    print("2.", a_list)
main()  
1. [10, 8, 6, 4, 7]  
20 16 12 8 14  
2. [10, 8, 6, 4, 7]  
Note that in the above example, the values of the elements in the list have not changed in any way.
```

What if the intention was to update the element values in the list?

```python
def main():
    a_list = [10, 8, 6, 4, 7]
    print("1.", a_list)
    number_of_elements = len(a_list)
    for index in range(number_of_elements):
        a_list[index] = a_list[index] * 2
    print("2.", a_list)
main()  
1. [10, 8, 6, 4, 7]  
2. [20, 16, 12, 8, 14]  
```

Changing a value at an index location updates the element of the list.

Give the output

```python
def main():
    my_list = [10, 8, 6, 4, 7]
    for index in range(len(my_list)):
        print(index, my_list[index] * 2)
main()  
```

The string method, split()

The string method, split(), separates a single string into a list of the parts of the string (the tokens) using the separator defined (inside the parentheses). Each element of the resulting list is a string object. This method can be applied to any string object.

If no separator is defined (as in the code below), whitespace is the default separator, e.g.,

```python
def main():
    phrase = 'The best cure for insomnia is to get a lot of sleep'
    words_list = phrase.split()
    print(words_list[0], words_list[4], words_list[7])
main()  
The insomnia get
```
Complete the main() function
Complete the code in the main() function which adds 1 to each list element in the list which has an odd value.

```python
import random
def main():
a_list = []
for index in range(10):
a_list = a_list + [random.randrange(1, 100)]
print("1.", a_list)
print("2.", a_list)
main()
```

```
1. [69, 98, 7, 92, 13, 9, 27, 36, 96, 46]
2. [70, 98, 8, 92, 14, 10, 28, 36, 96, 46]
```

Complete the main() function
Complete the code in the main() function which changes the elements starting from index 1 so that each element is the accumulative total of the previous elements (i.e., element 1 is the sum of the element 0 and element 1, element 2 is the sum of element 1 and element 2, etc.).

```python
import random
def main():
a_list = []
for num in range(10):
a_list = a_list + [random.randrange(1, 10)]
print("1.", a_list)
print("2.", a_list)
main()
```

```
1. [6, 8, 9, 5, 6, 3, 6, 4, 5, 6]
2. [8, 9, 18, 23, 29, 32, 38, 42, 47, 53]
```

Complete the main() function
Complete the code in the main() function which changes each string element of the list into an integer.

```python
import random
def main():
a_list = ["6", "7", "5", "3", "8", "1", "9", "2", "8"]
print("1.", a_list)
print("2.", a_list)
main()
```

```
1. ['6', '7', '5', '3', '8', '1', '9', '2', '8']
2. [6, 7, 5, 3, 8, 1, 9, 2, 8]
```

The split() method - Example

```python
def main():
prompt = "Enter a line of numbers: 
line_of_nums = input(prompt)
list_of_nums = line_of_nums.split()
for index in range(len(list_of_nums)):
    list_of_nums[index] = int(list_of_nums[index])
total = 0
for number in list_of_nums:
total = total + number
print("Total:", total)
main()
```

```
Enter a line of numbers: 4 6 12 13 9
Total: 44
Enter a line of numbers: 5 -3 6 8 1
Total: 17
```

Note that split() function breaks a string up into a list of strings.
Assigning a list object to a variable

Python lists are objects. When an object is assigned to a variable, the reference (i.e., the address) is copied and stored in the variable.

```python
list1 = [1, 2, 3]
list2 = list1
list3 = [1, 2, 3]

print(list1)
print(list2)
print(list3)


print()
print(list1)
print(list2)
print(list3)
```

Summary

In a Python program:
- a `for ... in` loop can be used to access each individual element of a list
- a `for ... in range()` loop can be used to make changes to individual element of a list
- a list is an object. Assigning a list to a variable makes a copy of the reference (not a copy of the list).
- lists are mutable objects
- we use the `split()` method to break a string into a list of strings. The default separator for the `split()` method is whitespace.

Examples of Python features used in this lecture

```python
def change_list(a_list):
    number_of_elements = len(a_list)
    for i in range(number_of_elements):
        a_list[i] = a_list[i] * 2

def use_lists(list1, list2):
    list3 = []
    for index in range(len(list1)):
        list3 = list3 + [list1[index] + list2[index]]

    return list3

def split_message(message):
    words = message.split()
    print(words[2], words[0])
```