Chapter 10 Lists

Opening Problem

Read one hundred numbers, compute their average, and find out how many numbers are above the average.

Solution

DataAnalysis

Objectives

- To create lists
- To invoke list's append method
- To use the len function/method
- To access list elements using indexed variables
- To traverse elements in a list using a for loop
- To copy contents from one list to another
- To develop and invoke functions with list arguments and return value
- To search elements using the linear

Creating Lists

You may create a list using the following syntax:

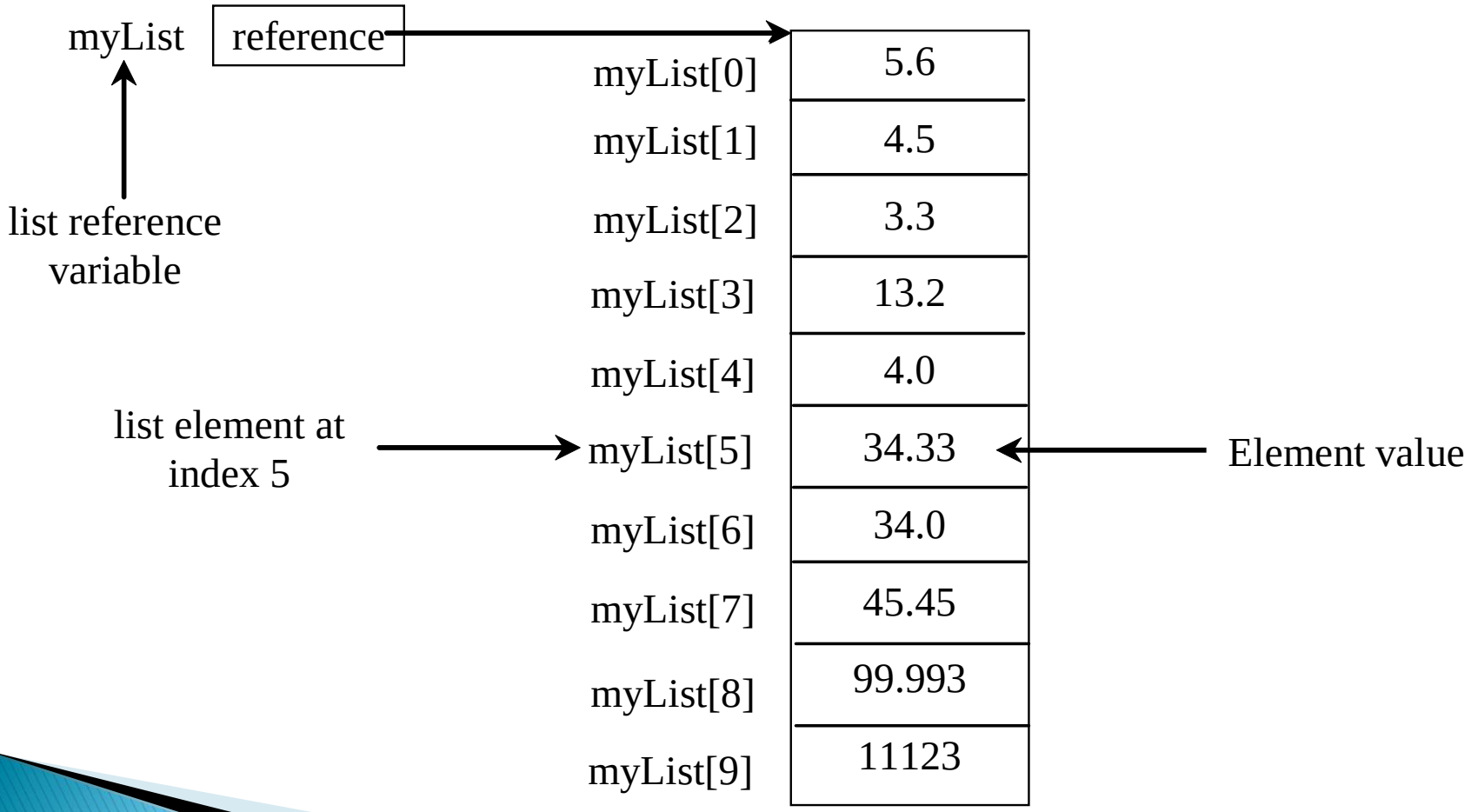
```
list1 = []
list2 = [2, 3, 4]
list3 = ["red", "green"] # Same as list(["red", "green"])
```

Methods (functions) for lists

```
>>> list1 = [2, 3, 4, 1, 32]
>>> len(list1)
5
>>> max(list1)
32
>>> min(list1)
1
>>>
```

Indexer Operator []

myList = [5.6, 4.5, 3.3, 13.2, 4.0, 34.33, 34.0, 45.45, 99.993, 11123]



Print list using a for loop

```
>>> myList = [2, 3, 1, 5, 9, 7]
>>>
>>> for i in range(len(myList)):
        print("index", i, "is", myList[i])
index 0 is 2
index 1 is 3
index 2 is 1
index 3 is 5
index 4 is 9
index 5 is 7
```

The + Operator

```
>>> list1 = [2, 3]
>>> list2 = [1, 9]
>>> list3 = list1 + list2
>>> list3
[2, 3, 1, 9]
>>>
```

Passing Lists to Functios

```
def printList(lst):
    for element in lst:
        print(element)
```

Invoke the function

```
lst = [3, 1, 2, 6, 4, 2]
printList(lst)
```

Invoke the function

```
printList([3, 1, 2, 6, 4, 2])
```

Anonymous list

Pass By Value

Python uses *pass-by-value* to pass arguments to a function. There are important differences between passing the values of variables of numbers and strings and passing lists.

Immutable objects

Mutable/Changeable objects

Pass By Value (Immutable objects)

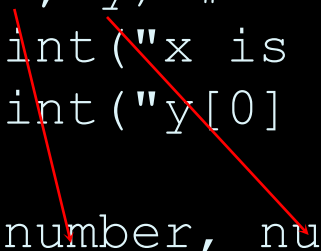
For an argument of a number or a string, the original value of the number and string outside the function is not changed, because numbers and strings are immutable in Python.

Pass By Reference/Sharing (changeable objects)

For an argument of a list, the value of the argument is a reference to a list; this reference value is passed to the function. Semantically, it can be best described as *pass-by-sharing*, i.e., the list in the function is the same as the list being passed. So if you change the list in the function, you will see the change outside the function.

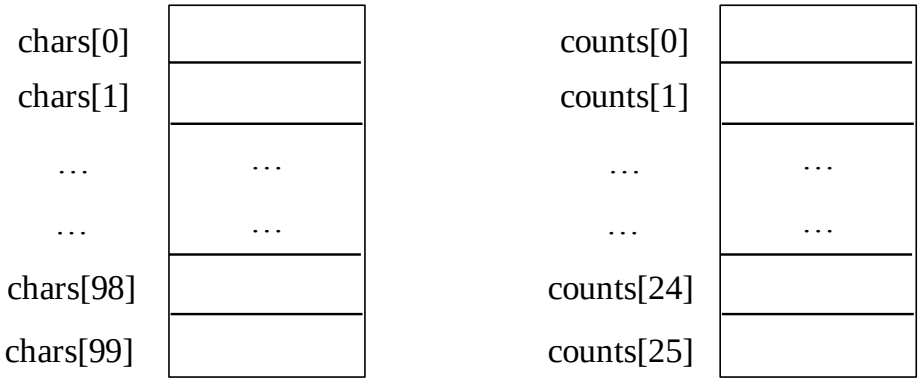
Simple Example

```
def main():
    x = 1 # x represents an int value
    y = [1, 2, 3] # y represents a list
    m(x, y) # Invoke f with arguments x and y
    print("x is " + str(x))
    print("y[0] is " + str(y[0]))

def m(number, numbers):
    number = 1001 # Assign a new value to number
    numbers[0] = 5555 # Assign a new value to numbers[0]

main()
```



Problem: Counting Occurrence of Each Letter

- Generate 100 lowercase letters randomly and assign to a list of characters.
- Count the occurrence of each letter in the list.

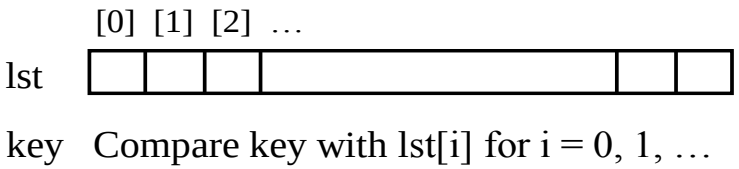


CountLettersInList

Searching Lists

Searching is the process of looking for a specific element in a list; for example, discovering whether a certain score is included in a list of scores. Searching is a common task in computer programming. There are many algorithms and data structures devoted to searching. In this section, one commonly used approach is discussed, *linear search*.

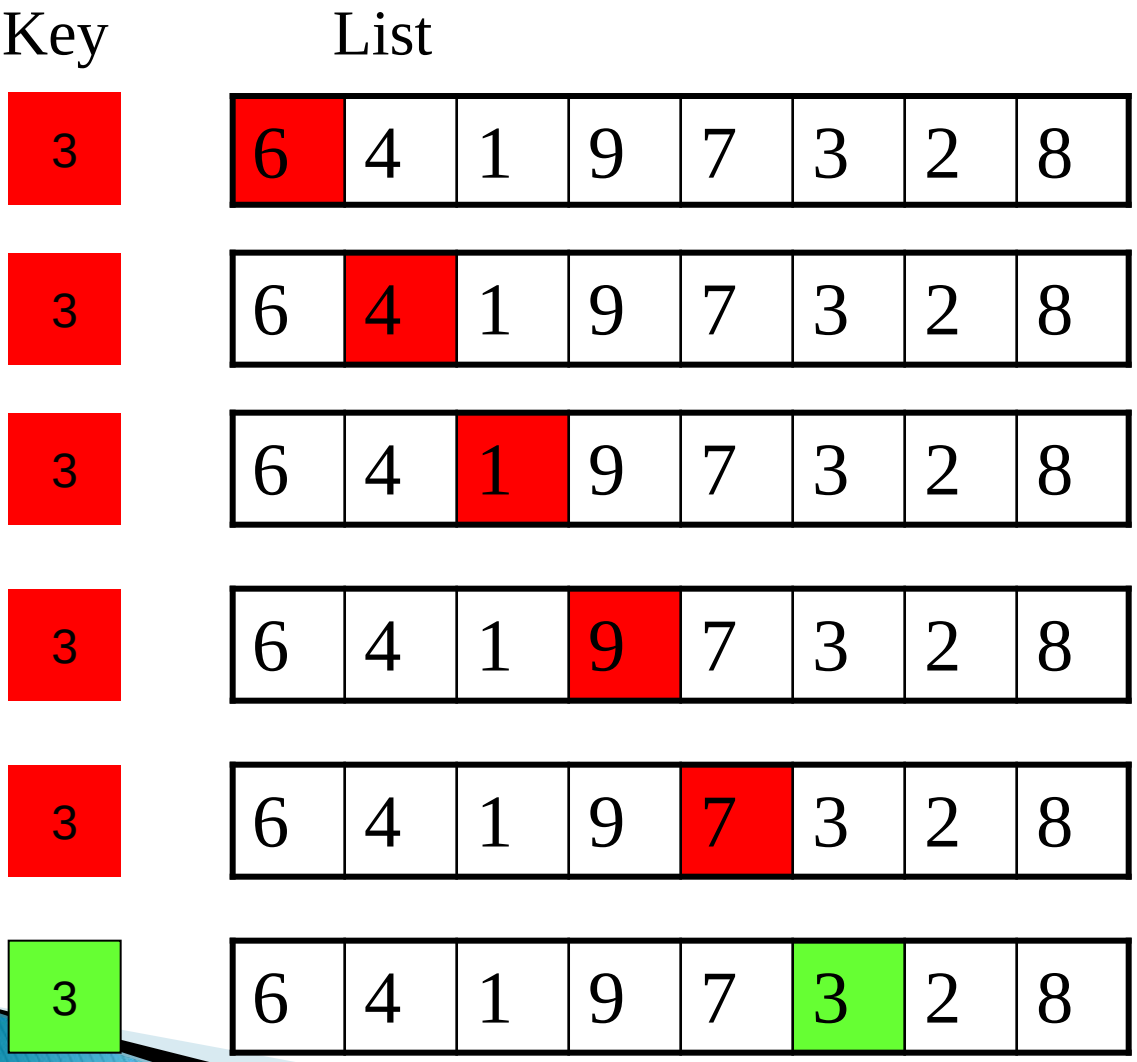
```
# The function for finding a key in the list
def linearSearch(lst, key):
    for i in range(0, len(lst)):
        if key == lst[i]:
            return i

    return -1
```



Linear Search

The linear search approach compares the key element, key, *sequentially* with each element in list. The method continues to do so until the key matches an element in the list or the list is exhausted without a match being found. If a match is made, the linear search returns the index of the element in the list that matches the key. If no match is found, the search returns -1.

Linear Search Animation



Create a list from a file of data (print in reverse when done)

```
infile = open("listdata.py","r")

myList = []

num = eval(infile.readline())

while num != 0:
    myList.append(num)

    num = eval(infile.readline())


infile.close()


# print the list (in reverse) when done

for i in range(len(myList)-1,-1,-1):
    print(myList[i])
```