Python Data Types: Lists

Computer Science ICS20

Creating Lists

Lists are data structures available in Python in order to deal efficiently with sets of data. A list can contain strings, integers, decimal numbers, logic values, etc. Lists can be created by using square brackets. For example, to create a list called myList:

```
myList = []                  #empty list
myList2 = [2, 4, 6, 1, 0]    #list with initial values
```

Lists can also be created using a combination of the list() and the range() functions:

```
myList3 = list(range(5))     #creates a list: [0,1,2,3,4]
```

Adding Members to a List

Adding members to a list can be done by using the append() function:

```
myList.append(“David”)
myList.append(“Lisa”)
myList.append(“Bob”)
```

Printing a List

A list can be displayed by using the print() function. For the above list, the command **print(myList)** would print the following:

```
[“David”, “Lisa”, “Bob”]
```

Accessing Members of a List

We access individual members of a list in much the same way as we do with strings; by using index values in square brackets. For example, if we want to print the first item in myList:

```
print(myList[0])
```

which would print:

```
David
```

Removing Members from a List

Removing items from a list can be done using the remove() function:

```
myList.remove(“Lisa”)
```

Lists and Operators

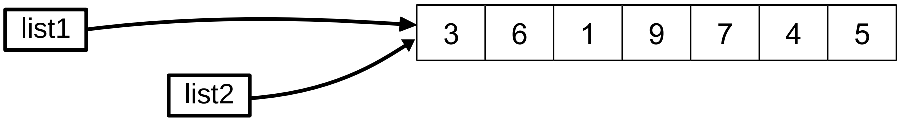
There are some operators that work with lists. For example the **assignment operator** makes a copy of the list name while still referring to the same list. As an example:

```
list1 = [3, 6, 1, 9, 7, 4]
list2 = list1
print(list1)
list1.append(5)
print(list2)
```

The above code would print :

```
[3, 6, 1, 9, 7, 4, 5]
```

Whatever changes are made in list1 also occur in list2, because they both refer to the same list! The following image shows this concept:



In this situation, **list1** and **list2** are called **references** that both point to the same list.

The **+ operator** can also be used for **list concatenation**:

```
list1 = [1, 2, 3]
list2 = [11, 6, 23]
list3 = list1 + list2
```

The above code would print [1,2,3,11,6,23].

The *** operator** can be used for **list repetition**:

```
list1 = [1, 4]
list2 = list1 * 4
print(list2)
```

The above code would print [1,4,1,4,1,4,1,4].

List Functions

There are also many functions that can operate on lists. Here are some of them:

Function Name	**Description**
len(list)	Returns the number of items in the list
sum()	Returns the sum of the list
max()	Returns the greatest item in the list
min()	Returns the smallest item in the list
list.append(item)	Adds the item to the end of the list.
list.clear()	Removes all items from the list.

list.count(item)	Returns the total number of matching items specified by the argument.
list.index(item)	Returns the index of the specified item if found in the list, otherwise returns error.
list.insert(index, item)	Inserts a new item at the specified index location.
list.reverse()	Reverses all the items in the list.
list.sort(reverse=False/True)	Sorts the list in ascending or descending order.
list.extend(list2)	Adds the items of another list, list2, to the end of the list.
list.pop(index)	Returns and removes the item at the given index from the list.
list.copy()	Returns a separate copy of the list.

Exercises

1. Write a Python program that repeatedly asks the user for 5 names and then prints them out in reverse order.
2. Write a Python program that creates a list of the first 100 natural numbers and outputs the their average.
3. Write a Python program that determines if a given list is in sorted order.
4. Write a Python program that removes any duplicates from a given list.
5. Write a Python program that merges two lists as follows:
   If list1 = [1, 2, 3, 4, 5, 6] and list2 = [7, 8, 9, 10] then the merger of the two lists would be
   [1, 7, 2, 8, 3, 9, 4, 10, 5, 6]
6. Write a Python program that finds the number of occurrences of a number in a given list.
7. Given a list of integers and a starting location, use the value at that location as the position of the next location and so on. If you end up on the last item the program quits and prints “You win!”. If you end up outside of the list, the program prints “You lose!” For example (2, 5, 3, -1, 4, -4, 2, -3) and a starting location of 1 leads to the sequence: 2, 3, -4, 5, 2 ....You lose!
8. A detector in a road sends a signal every second. 1=no vehicle, 2=vehicle, 0=end of survey. Given a series of these data elements in a list, calculate the length of the survey, the number of vehicles, the longest interval with no vehicles, and the average # of vehicles per minute.
9. Write a Python program that finds the peak daily air pollution counts stored in a list of numbers. A peak number is defined as a number which is greater than the number before and after. The output will be a list of all peaks, as well as the largest peak.
   Example: input list [2, 4, 5, 2 , 34, 23, 12, 29, 22, 17, 5]
   output: peaks = [5, 34, 29], max peak = 34
10. Write a Python program that determines if a number is nasty within a given list of numbers. A nasty number has at least one pair of factors such that the difference of the pair = the sum of another pair. For example, if we have a list:

    [1, 2, 3, 6, 8, 4, 12, 15, 7, 24, 32]

    then 6 is nasty because 6-1 = 2+3, 24 is nasty because 12-2 = 4+6