# Lesson 5 - Lists (Arrays)

|  |  |
| --- | --- |
| **Lesson Outcomes**  In this lesson you will learn:   * how to use basic Lists; * process and manage lists; * how to incorporate Lists into your programs. | **C:\Users\Graham\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\X6CHINOH\MC900441498[1].png** |

## Introduction to Lists

Lists are what we call **DATA STRUCTURES**. A data structure is way of storing data in some ordered fashion so that we may extract and process it easily. The most common type of data structures is an ARRAY, which is implemented in many different languages.

In Python Arrays don’t exist, as instead LISTS are used. Lists are dynamic and allow you store variables in a structured way inside them. Lists also provide many methods to manipulate and extract the stored date. Example, creating a simple list of numbers stored in sequence:



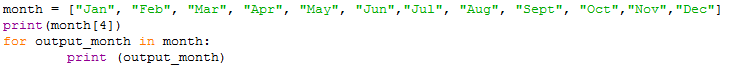
The brackets [] are used to define the List, with each number inserted separated by a comma. The List is assigned to the variable sampleList. Once the list has been created we can process it by referring to the list index number. So, in the above example, by printing sampleList at index position [3] we get the number 4 (the index starts from 0).

Lists can not only hold numbers but also strings or characters:



The above stores all the months in the year, starting with index 0 as “Jan”.

Lists can also be used in for loops to iterate through:



The for loop above goes through each element in the list and outputs that value.

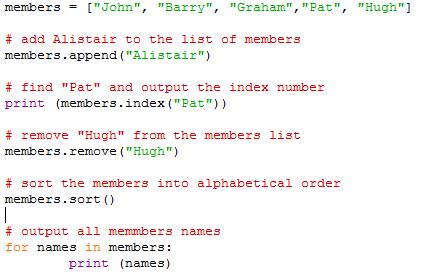
### Tasks

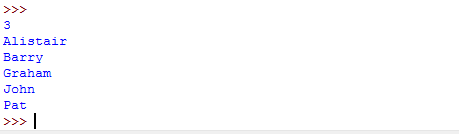
5.1 Write program that stores all the months of the year in a List. The program should prompt the user for a month between 1 and 12, and then output the month name. Include validation so that user will be prompted for a month again if the number entered is outside the range 1 to 12.

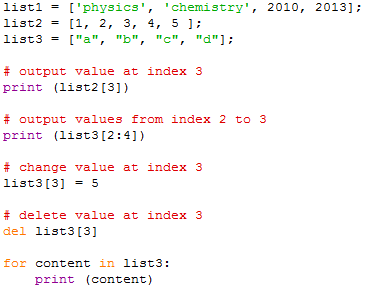
## Processing Lists

One of the most useful things about Python Lists is the common methods provided which allow you to easily process and manipulate the stored data. The following methods are:

* .append (value) adds elements to end of the list
* .count(“x”) counts the number of occurrences of x
* .index(“x”) returns the index of “x” in the list
* .insert(“y”,”x”) inserts “x” at location “y”
* .pop() returns last element then removes it
* .remove(‘X’) finds and removes first ‘x’ from list
* .sort() sorts the list in ascending order.

An example program which uses the above:

The above program, when executed, displays the following:

As well as using the above Lists methods you can also refer to list values through the indexes. For example:

Sometimes you may wish to initialise a list with x number of elements. This can easily be achieved using the for..loop:

The above will add 5 elements defaulting to the value “0”.

More on lists, visit <http://www.tutorialspoint.com/python/python_lists.htm>

### Tasks

5.2 Write a program that reads 6 names into a List. The program must display the names in the same order that they were entered and then in reverse order.

**HINT:** you can use the list method “.reverse”

5.3 We want to simulate throwing a die 30 times and record the scores. If we did this “manually” we would end up with a tally chart. If we use a computer to keep a count of how many times each number was thrown we could use an integer list instead of the tally chart. In general, a die throw will give a score I, and we want to increment the count in the ith element.

TalllyChart[i] = TallyChart[i] + 1;

Write a program to simulate the throwing of a die 30 times. The results of the simulation should be printed as a table of scores (1 to 6) and frequencies.

5.4 **National Lottery Numbers**

We wish to select six random numbers between 1 and 49 with the condition that all numbers are different. One possible strategy, or algorithm, is:

*Initialise a list ready to store six random numbers*

*Generate a random number between 1 and 49*

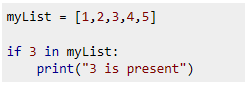
*If the number has been chosen before in the lottery list then generate again.*

*Display this value*

*Add number to the list*

*Repeat the above four steps until six numbers have been selected.*

Write a program to select six unique random numbers between 1 and 49.

 **HINT:** you can check if a number exists in a LIST using the IF statement in the following manner:

5.4 We can declare two lists student and DoB, to store the name of Students and their dates of birth. For example if Fred is born on 22/12/94, then we could store “Fred” in Student [0] and “22/12/94” in DoB[0]. To find a particular student we can use the **“.index”** method to search a name.

Create two lists to store 6 names with their corresponding DOB.

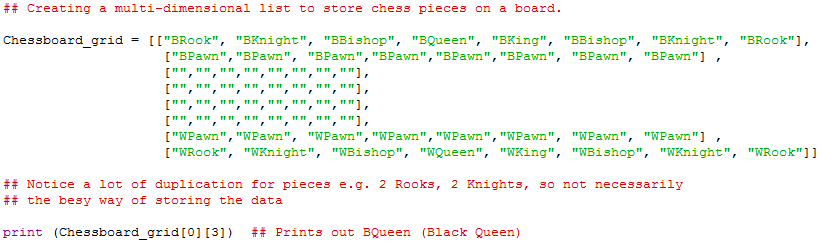
Write a program that asks the user for a NAME to search in the list and return their DOB.

## Multi-Dimensional Lists

Sometimes in programming it is necessary to store data in multi-dimensional lists i.e. lists within lists. For example, if you were writing a chess program you would have to store the chess pieces position in a grid format 8 across by 8 down. For this you would need a 2 dimensional list. For example:

This uses the **FOR** command to create **8 elements by 8 element**s, with the default value of 0. To print out the value of an individual cell across the grid you would run the following command:

You could also setup the chessboard by creating each element in the list manually:



### Tasks

5.5 Change the program you created in task 5.4, so that the student name and DOB are stored in one mult-dimensional list. For example, you can create a list of 2 elements, name and DOB, with each element containing sub-list of 6 names or DOB.