**Arrays Part I**

**An *array* is an ordered list of objects**. You can have an array of students, an array of cars, an array of books. You can even have an array of feelings. An array is ordered, in the sense that you can separate its elements by number: *the third book, the top student, my first impression etc.* The number (1,2,3) is called the **index** of the array.

We have already been working with arrays. A ***string*** is an ***array of letters***. Consider this example:

myName = "Stephanie"

print (myName[0]) #What does this print?

print(myName[3]) #What about this?

Note the square brackets. Square brackets are only reserved for string indexes.

You could print the letters in the string one by one:

myName = "Stephanie"

print(myName[0]) # first letter

print(myName[1]) # second letter...

print(myName[2])

print(myName[3])

print(myName[4]) # you get the idea

This is better done with a loop that counts from 0 to 4. The index in the square brackets is a natural fit for our loop counter, i:

myName = "Stephanie"

for i in range(5):

print(myName[i]) # prints letters 0 to 4

**The len() function**

You’ll notice in the last example, that it does not print all the letters in the name “Stephanie”, since it only prints out the first 5 letters. We could count the number of letters in the name, but there is a better way, using the len() function. The len() function tells us the length of a string.

myName = "Stephanie"

x = len(myName): # find the length of the string

print(x)

What is the length of the string? Now back to our program. We could print all the letters in the string using the len() function like this:

myName = "Stephanie"

x = len(myName): # find the length of the string

for i in range(x): # keep going for the length of the name

print(myName[i]) # prints all the letters

A shorter way to write this is to not use a variable x:

myName = "Stephanie"

for i in range(len(myName)):

print(myName[i]) # prints all the letters

**Arrays of other things**

Arrays are not just for strings. Arrays can be made of virtually anything.

Let’s make an array of friends. First we create an empty array:

myFriends = [] # I have no friends :(

So sad! Let’s change our line and add some friends:

myFriends = ["Jasmine","Hussein"] # I have two friends!

Note the square brackets again. In Python, square brackets tell us that we are working with an array.

Now we can print our list of friends:

print(myFriends)

This will output:

['Jasmine', 'Hussein']

Notice the square brackets. All lists are kept in square brackets. The above list has two **elements**: Jasmine and Hussein.

We can access each element individually by **index**:

print(myFriends[0]) # what does this print?

And…

print(myFriends[1]) # what does this print?

**Adding to a List using append()**

Two friends is not enough! Let's add some friends to our list using the **append()** function:

myFriends.append("Jackie") # welcome!

myFriends.append("Sarah")

print (myFriends) # that's better!

**Len() - The Number of Elements in a List**

Just as with strings, we can print the number of elements in a list using the **len()** function. Add this code to your previous example:

numberOfFriends = len(myFriends)

print ("I have",numberOfFriends , "friends!")

Let’s print them out individually:

print ("They are: ")

print (myFriends[0]) # my first friend

print (myFriends[1]) # my second friend

print (myFriends[2]) # my third friend

print (myFriends[3]) # my fourth friend

Again, notice that the indexing begins at zero.

**Looping Through a List**

Just as with looping through a string, FOR loops are perfect for looping through any kind of list. Let’s use a FOR loop to print out our list of friends:

myFriends = ["Jasmine","Hussein","Jackie","Sarah" ]

numberOfFriends = len(myFriends)

print ("I have",numberOfFriends , "friends! They are: " )

for i in range(numberOfFriends): # loop 4 times

print(myFriends[i]) # print 4 friends

How does this work?

First, I make a list of friends. It has four elements in it.

Next I use the len() function to count the number of elements.

Then I use a for loop that counts from 0 to 3. Notice that the variable i is an integer variable, not a name. The variable i is called an *index* or a *counter*.

Each time the loop prints a friend, and the index goes up by 1 (increments). The print statement prints each friend, beginning with myFriends[0] to myFriends[3].

**The in function**

Python has a very useful keyword called **in**. It tells us if a value is found **in** our array. Here is how it looks:

myFriends = ["Jasmine","Hussein","Jackie","Sarah" ]

name = input("Name a friend: ") # input a name

if name **in** myFriends: # is it in the list?

print(name, "is my friend too!") # yes

else:

print("Maybe", name,"and I could be friends") # no

**Looping Through a List, Part 2**

The **in** keyword gives us a second way of looping through a list:

print ("Here are my friends again: ")

for f **in** myFriends: # for each name in my list

print (f) # print it

In this example, f is not an index like i is. The variable f is an element of the array. In other words, f = “Jasmine” then f = “Hussein” and so on.

To make the above code a little easier to read, we could use a better variable name than “f”:

print ("Here are my friends again: ")

for friend **in** myFriends: # for each name in my list

print (friend) # print it

Just remember that the computer doesn’t know what a **friend** is, it is “Those items in my list called myFriends”. If you instead wrote:

for month **in** myFriends: # for each name in my list

print (month) # print it

It would still print out my friends' names.

**Exercises**

1. Create an empty array called **animals**. Ask the user for an animal. If the animal is in your array, print “got it!”. If the animal is not in your array, print “Need it!” and then add it to your list. Put into a loop. The user breaks out of the loop by entering nothing.
2. Create two arrays:

Days: an array containing 7 days.

Months: an array containing 12 months.

Write a program that creates 10 random dates:

“Tuesday July 4”

“Wednesday February 12”

etc.

Note: Don’t worry about being realistic, i.e. February has 28 days, March has 31 days etc. Just get the basic idea to work.