59 6f 75 20 6e 65 76 65 72 20 66 61 69 6c 20 75 6e 74 69 6c 20 79 6f 75 20 73 74 6f 70 20 74 72 79 69 6e 67

A Level Computer Science

Preparation – Bridging Lesson 3

Objectives

- Python Programming part 3
  - Lists and Arrays
  - Other topic areas (string manipulation, file handling, subroutines)
  - Exercises

- Useful Links - Python
  - Lists
  - Strings
  - Functions
  - File Handling

Lists and Arrays

- Lists and arrays are very useful data structures in programming that are used to store many different values within one single structure.
- This means that many related values can be accessed using *one variable name*.
- An example is as follows, which demonstrates using a list to hold a student's exam scores:
- results = [77, 87, 66, 56, 81]
- Here, we could access a particular exam score using the variable results and the index of the particular result (or element that we want to retrieve), such as print(result[1]) would print out 87 (note that the indexing starts at 0)
- Learning Point – lists and arrays are similar (we will not investigate the differences here – but you could research this topic). As Python only has lists, we will refer to lists for the remainder of this lesson (we will look at arrays when we use Java).

Lists and Arrays

- A list (in Python) can contain different data types, such as:
- scores = [“Tom”, 87, 66, “Sam”, 56, 81]
- These values could be printed out using a for loop as follows:
- for i in range (0, len(scores)):
- print(scores[i])
- This would print out each element of the above list on separate lines.
- Learning Point – the len keyword will return the number of items in a list (or the number of characters in a string).
- An empty list could be created using a statement such as myList = []
- We could add elements to a list by using the append command, myList.append(“Alice”)

Lists and Arrays

- Exercises
- Exercise 1 – create a program that starts with an empty list called names. Use a suitable iteration to ask the user to enter five integer values. When the user has entered the values, add code to calculate the sum of the integers in the list, and print this off in a suitable manner.
- Exercise 2 – create a program that will ask a user to enter numbers which will be added to a list. The process is to stop when the user enters -1. Add further code to print the contents of the list out.
- Exercise 3 – ask the user to enter to enter six words. These words should be stored in a list. When all words have be added, the program should compute the longest word and print it out (if two or more words are the same length, the program should print out the first of those words).

Other Techniques

- Further Studies
- In this series of lessons, we have briefly covered some of the fundamentals of Python programming. Many of these constructs and techniques are very important across all programming languages, so should be reviewed and you are encouraged to use websites such as https://www.w3schools.com/python/default.asp in order to get a better grasp of Python before the course starts.
- Suggested further study:
  - String Manipulation – https://www.w3schools.com/python/python_strings.asp
  - Functions – https://www.w3schools.com/python/python_functions.asp
  - File Handling – https://www.w3schools.com/python/python_file_handling.asp

Exercises

- Further Studies
- **The following exercises should be completed before you start the A Level course. Note that you will be expected to have completed these exercises prior to the first lesson.**
- Task 1 – create a program that reads a series of integers from a text file into a list. The text file should be called **numbers.txt** and the file should contain the numbers 34, 12, 45, 77, 4, 100 (on separate lines). The program should write all the even numbers into a new text file called **even.txt**, and all odd numbers into a new text file called **odd.txt**. Note – you will need to create the initial file called **numbers.txt** and type the numbers into it.
- Task 2 – create a program that asks the user for words. These words will be stored in a list. The user is continually asked for words until a word is entered that is less than 4 characters long – this will stop the process, and that word will not be added to the list. When this process is complete, the program should write any words that are over 6 characters long or words that start with the letter 'a' into a new file called **longer.txt**.