SS3 - Advanced Programming

Functions, Arrays and Lists | Senior Secondary School 3 | Computer Studies

Functions

A function is a named block of instructions that carries out a particular task. We define it once and call it whenever needed. Functions reduce repeated code and make a program easier to test and correct.

Parts of a function

- Name: identifies the function, such as calculate_total.
- Parameter: a variable written in the function definition to receive an input.
- Argument: the actual value supplied when the function is called.
- Return value: the answer a function sends back to the part of the program that called it.

```
def add_scores(first, second):
    total = first + second
    return total

result = add_scores(12, 8)
print(result)  # 20
```

Here first and second are parameters. The numbers 12 and 8 are arguments. The return statement gives back 20. A function may also print a result, but printing displays it while returning allows another part of the program to use it.

Arrays

An array stores several related values under one name. Each value has a position called an index. In many programming languages, arrays have a fixed size and store values of one data type. For example, an array may hold the test scores of five students: 12, 15, 18, 14 and 17.

Most programming languages start indexing at 0. Therefore the first score is at index 0 and the third score is at index 2. Trying to use an index outside the available positions causes an error. Some languages also provide two dimensional arrays, such as rows and columns in a table.

Lists in Python

A Python list is an ordered, changeable collection. It can grow or shrink and can hold different kinds of values. We use square brackets to create it, and an index to access one item.

```
scores = [12, 15, 18, 14, 17]
print(scores[0])      # 12, the first item
print(scores[2])      # 18, the third item
scores[1] = 16        # change the second item
scores.append(20)     # add an item at the end
print(len(scores))    # 6 items
```

Useful operations include append(value) to add an item, remove(value) to delete the first matching item, len(list) to count items, and list[index] to read or change an item. The last item can also be accessed with index -1.

Using a loop with a list

```
scores = [12, 15, 18]
for score in scores:
    print(score)
print(sum(scores))       # 45
```

The loop visits each score in turn. The built in sum function adds numerical values in the list. The average is sum(scores) / len(scores), provided the list is not empty.

Arrays and lists compared

Feature	General array	Python list
Size	Often fixed when created	Can grow or shrink
Data type	Often one type of value	May contain different types
Access	Usually by index	By index, starting at 0

Note: “Array” is a general programming concept. Python also has specialised array tools; the everyday Python list is the simplest collection for these lessons.

Worked example Student scores

```
def average_score(scores):
    if len(scores) == 0:
        return 0
    return sum(scores) / len(scores)

marks = [60, 75, 65]
print(average_score(marks))  # 66.666...
```

The function receives a list of marks, checks that it is not empty, and returns the average. This combines a function with a list in one useful school task.

Class activity

Write a program that stores five students’ scores in a list. Print the first and last scores, add one new score, and write a function that returns the highest score using max(scores). Run the program and explain each output.

Evaluation questions

- What is a function? Give two reasons for using functions.
- Distinguish between a parameter and an argument.
- What will numbers[1] produce if numbers = [4, 7, 9]?
- Write one Python statement that adds 25 to a list called scores.
- State two differences between a general array and a Python list.