Midterm Question 1

Victor Harris

CIS-104 Introduction to Programming Logic

The output of this program is:

10

11

Final Report: 12 15 18

Although the output is relatively simple, there is a lot more under the hood than is immediately obvious. However, the dissection of this output can be made equally as simple if we look at the process step by step. Before we break it down line by line, though, there are a few things that we can quickly identify that give us a general idea of what we can expect as an output. For example, by looking at this program we can easily identify that there is one function that results in an output: ‘print ()’. So, with that in mind, we know that there are 2 lines that can potentially result in output (lines 6 and 8), and therefore know that we will have lines of output equal to the number of times those lines of code are executed. Being that line 6 is nested inside of a loop, we know that we should see some number (in this case 2) of lines printed to the console, along a final line printed by the print function in line 8. For line 8, we know that it will run only once because it is written as a process and not part of a subroutine. So, by identifying these 2 lines of code, we can determine that regardless of variables and their values, we know that the output of this program should follow this general format:

## Some number of printed values, each printed on their own line.

## A final output of ‘Final Report: ‘along with 3 other values.

Knowing this, we can more accurately determine that ‘some number’, in this case, is 2, because the value of ‘counter’ is defined as 0 before entering the loop, and the decision that is evaluated against it (counter < 3) will run twice before failing based on the process in line 4, which increments the value of counter prior to it being evaluated. So, with this in mind we know that the output will be:

## Printed value of ‘n’

## Printed value of ‘n’

## A final output of ‘Final Report: ‘along with 3 other values.

Now, it is simply a matter of determining what the values will be at the time of the lines being printed. For the first and second lines, this can be done by determining which iteration of the for loop we are on at the time of printing. Being that we already know that the test on line 5 will only pass twice (in its first two iterations), we can determine that the value of ‘n’ during those iterations will equal the first two values stored in ‘list’; 10 and 11. For line 8, the printed values reference list by pulling values stored at specific indexes, which in this case are the 3rd, 6th, and 8th values in this list, or 12, 15, and 18, respectively. This gives us the resulting output:

10

11

Final Report: 12 15 18

Even with the output determined, though, there are other factors at work in this program. Each line is critical in the result, and even the smallest tweak could result in a completely different final product. Let us break down what each line does and how it affects the whole program:

Line 1: Declares a list object which is named (ambiguously) ‘list. This object serves as a container for values that the rest of this program will reference. The object is also defined in this line as a container of 10 values. The values contained in this list will be interpreted as data-type integers.

Line 2: Declares a variable ‘counter’ and defines it as an (interpreted) integer value of 0.

Line 3: Begins a for loop that will sequence through each value contained in ‘list’.

Line 4: Increments the value of counter’ by one for each iteration of the loop in which it is nested.

Line 5: Tests the value of ‘counter’ as it compares to 3 and determines whether or not the program should execute line 6.

Line 6: Prints the current value of ‘n’ as a string, which in this case is an integer value from ‘list’, which value is dependent on which iteration the loop is currently executing.

Line 7: White space which separates the loop beginning on line 3.

Line 8: Prints four string values consisting of the static ‘Final Report: ‘, with the integer values (converted to string) at indexes 2, 5, and 7 of ‘list’, with the default separator of ‘ ‘.