Andrew Plum

CS 120-06

Lab #10

11/3/21

Hypotheses and Explanation for Lab #10

**How much I completed:**

I did all of the regular lab assignment; there was no extra credit.

**Hypotheses:**

* X = 99
* Y = 101
* a[0] = 102
* a[1] = 8
* a[2] = 11
* a[3] = 12
* a[4] = 456

**Correct Output:**

* X = 99
* Y = 101
* a[0] = 102
* a[1] = 8
* a[2] = 11
* a[3] = 12
* a[4] = 456

**Explanation:**

* X = 99
  + X begins the program with it being equal to the value 99, and no changes are made to it throughout the program. Its value does get passed through two functions, but X’s value is not changed because the function use pass by value for it and not pass by reference.
* Y = 101
  + Y begins the program with it being equal to the value 55. Its value is changed when its address is passed through the first function meaning it is using pass by reference. Because the address is passed through with the argument int &b, y is now equal to 101 when b is set to equal 101 in the function because y and b share the same address in memory.
* a[0] = 102
  + a[0] begins the program with it being equal to the value 1. Its value is changed at the line in the program that reads, “a[0] = func(x,y);”. This line of code reads set a[0] equal to the integer value the function func(x,y) returns. The function returns the integer value 102 which means a[0] is now equal to 102.
* a[1] = 8
  + a[1] begins the program with it being equal to the value 2. The value for a[1] is changed at the lines of code in the program that read, “p = a; p++; \*p = 8;”. The first line of code sets the pointer memory address equal to the memory address where array a starts which would be the memory address of a[0]. Then the second line of code makes a jump up one to the memory address at a[1]. The next line of code sets the value at this memory address, which is the memory address a[1], equal to 8. Therefore a[1] is equal to 8.
* a[2] = 11
  + a[2] begins the program with it being equal to the value 3. The value for a[2] is changed at the lines of code in the program that read, “p = &a[3]; p--; \*p = 11;”. The first line of code sets the pointer memory address equal to the memory address of the fourth element in the array a which is a[3]. Then the second line of code makes a jump down one to the memory address at a[2]. The next line of code sets the value at this memory address, which is the memory address a[2], equal to 12. Therefore a[2] is equal to 12.
* a[3] = 12
  + a[3] begins the program with it being equal to the value 4. The value for a[3] is changed at the lines of code in the program that read, “p = &a[3]; \*p = 12;”. The first line of code sets the pointer memory address equal to the memory address of the fourth element in the array a which is a[3]. The next line of code sets the value at this memory address, which is the memory address a[3], equal to 12. Therefore a[3] is equal to 12.
* a[4] = 456
  + a[4] begins the program with it being equal to the value 5. Its value is changed when the memory address of a, which is where the array starts, is passed through the second function meaning it is using pass by reference. Because the address is passed through with the argument int b[], a[4] is now equal to 456 when b[4] is set to equal 456 in the function because a[4] and b[4] share the same address in memory.