CMPT 214

Assignment 2, Question 3 Solution

Mark Eramian

Assignment 2, Question 3

Solution

Variable `x`: Variable `x` is an integer variable that gets an initial value of 2. A pointer to `x` is passed to `pointer_stuff()` as parameter `c`. Line 8 dereferences `c` and modifies the value it points at to be `a + *b` which is `3 + 5` because `z` was passed to a, and `&y` was passed to `b` which dereferences to `y`'s value of 5. So on line 8, `*c`, which points to `x`, is modified to be 8. The rest of the program does not modify x, so it's final value is 8.

Variable `y`: Variable `y` has an initial value of 5. The address of `y` is passed to `pointer_stuff()` as `b`. Since `*b` is never modified by the function `pointer_stuff()`, the value of `y` is not changed by the function, nor is it modified by `main()`. Thus the final value of `y` remains unchanged at 5.

Variable `z`: The value of `z` is passed to parameter `a` of `pointer_stuff()`. Since `pointer_stuff()` receives the **value** of `z` and not its address, it cannot modify the contents of variable `z`. Since `z` is not modified by anything in `main()` either, its final value remains unchanged at 3.

Variable `w`: The variable `w` is initialized to the address of `x`. Since `w` is a pointer, the value of `w` is the memory address of `x`. The value of `w` never changes from its initial value, it always contains the memory address of `x`.

Note: The correct answer for `w` is that "the value of `w` does not change". This should not be confused with the fact that the value that `w` **points at** does change. Since `pointer_stuff()` changes the value of `*c`, and `c` is a pointer to `x`, and `w` is also a pointer to `x`, the value of `*w` changes, but the value of `w` does **not**.