CSCI-UA.0201

Computer System Organization

Homework Assignment 1 - Solutions

1. The compiler will not report an error. But, depending on the compiler, it may give you a warning. The program may or may not crash depending on whether you have overwritten important information in memory.

2. What is passed is the base address of the array.

3. **x = ++a[1];** For this one, a[1] is incremented *before* getting assigned to x. That is, what happens here is a[1]++ followed by x = a[1]. So, x = 2, and a[1] becomes 2.

   y = a[1]++; Here, what happens is: y = a[1] and then a[1]++. So, y = 2 and a[1] becomes 3;

   z = a[x++]; This is equivalent to z = a[x] followed by x++. This means z = z[2] = 15 and x becomes 3.

   The output is: 3 2 15

4. Doing it by hand: It is faster and you have full control.
   Garbage collector: removes the burden from the programmer.

5. a. 502
   b. 502
   c. 502
   d. 10000
   e. 502

6. Nothing is wrong, the declaration is perfectly fine!

7. There is an error in this structure declaration. The structure emp contains a member e of the same type struct emp. At this stage compiler does not know the size of this structure. In the previous problem (#6 above), the compiler knows how big a pointer is.

8. a) 8
   b) 8

9. a) You can pass the address of complex data types.
b) You can make dynamic memory allocation.

10.

a) Because once the function returns, this local variable no longer exists. It disappears. And the address you passed is pointing to some place that will be used later by the compiler for something else.

b) There is nothing wrong with the code. Because the code does not return the address of the local variable x. It returns the address of memory allocated in the heap (remember that malloc allocates memory in the heap), and the bytes allocated in the heap are not reclaimed when the function returns. They are reclaimed with a call to free();

11.

a) Because we start from 0 and not from 1.
b) Because one number from the positive range is used to present the 0.
c) Negative numbers have one extra number than positive numbers.

12.

a) 10101010 = 128 + 32 + 8 + 2 = 170
b) The number is negative so 2s complement (10101010) = 01010110 then the decimal equivalent of that = 2+4+16+64 = 86 … Therefore the final answer is -86
c) 0xAA
d) No, hexadecimal does not depend on whether the number is signed or unsigned because the conversion is done *blindly* for each 4 binary digits to one hexadecimal digit.