CS 33 Fall 2014 Lab 2

This lab will give you some experience using pointers to look at things and overall problems in dealing with hexadecimal data. It will also reveal how the stack contains automatic variables. It will show you how you can destroy the stack. All of this can be done in .c or .cpp.

This lab has several parts:

Part 1:

Write a memory dump: void dumper(unsigned char *x, int n)
Which prints out memory like this, in hexadecimal.

```
Address       +x00      +x04      +x08      +x0c
7fffb80aaab0  00000000  00000000  004003e3  01234567
7fffb80aaac0  b80aabf8  00007fff  b80aaabc  00007fff
7fffb80aaad0  3a40fb88  00000030  b80aaabc  00007fff
7fffb80aaae0  b80aab00  00007fff  00400537  00000000
7fffb80aaaf0  b80aaafc  00007fff  00000000  01234567
7fffb80aab00  00000000  00000000  3a41ecdd  00000030
7fffb80aab10  00000000  00000000  b80aabe8  00007fff
7fffb80aab20  00000000  00000001  00400514  00000000
```

x is a pointer to somewhere in the stack (the address of any automatic variable), n is the number of 16 byte addresses to print. x could point to anywhere.

Whatever x is, please round it down to a multiple of 16 so that every starting address ends in 0.

The column header is not required.

The first column is the memory address of the first four byte dump. The next four columns are the data at the addresses +0, +4, +8 and +12. In the printout, remember to reverse the order of each four byte chunk to make it read in right endian. Things which are 8 byte things will stay little endian but let's not worry about that.

This lab will give you some experience using pointers to look at things and overall problems in dealing with hexadecimal data. All of this can be done in .c or .cpp.

Part 2:

Write a “spray_paint” function which initializes sz bytes of the input address x to a recognizable token “tok”.

```
void spray_paint( char *x, int sz, char tok, char *t )
 {
  printf( "%p %3d %x %s\n", x,sz,(unsigned char) tok,t )  ;

// you add the rest

  }
```

x is a pointer to somewhere in the stack (the address of any automatic variable), sz is the number of bytes addresses to spray paint. x could point to anywhere. tok is a single character which will initialize x. t is a string used for debugging printout.

spray_paint will also maintain two external pointers: min_ptr and max_ptr to the highest and lowest address spray painted. min_ptr will be initialized to NULL in the template.

Part3:

Modify the template function sub2 to 1) save the entire contents of the memory from min_ptr to max_ptr to an external array. 2) spray_paint the same area to some recognizable character, essentially destroying the entire contents of the stack. Use dumper to print it out. 3) restore the stack from the saved area.

The resulting .c or .cpp program is what you should turn in.

For further education: 1) try running your program by leaving out step 3) above. You should get a segmentation error due to the destroyed stack. 2) try running your program by changing some of the automatic variables, like in main(), sub1() and sub2() (change the array sizes, data types to see how the stack changes). Keep before and after pictures of the output. You will see the effect on the stack of these changes.

The assignment will be due 6PM 9 Nov . You may submit early and often!