

CISC181H Spring 2009 Lab03

- Write a program for each of the following problems. Be sure to save every separate program. All programs must be properly commented and indented (see Assignment Standards on the class website). Ask your TA for guidance.
- Name each program lab02.n.cc, where n is the number in the list below. For example, the name of the file for the first will be lab02.1.cc. Put the files in your lab02 directory (see Anderson for instructions on how to make a directory).
- Do not be afraid to write multiple functions for a single problem. Short, simple functions are much more likely to work, and tend to be easier to write.
- C++ has a type called “string”. For a lot of good reasons, we aren’t using it yet; we are using the C-strings we discussed in class.

Programs

1. Declare an empty 2-d array of char with 5 rows. Loop through it and fill it with words from the user (use the insertion operator and cin to read a whole word into a row). Have the third word be 2 chars longer than the available space. Print each word on a separate line¹. Explain what you see by drawing a diagram of how the array looks in memory.
2. Repeat the program for problem 1, but using an array of char pointers. Declare an array of five char pointers. As the user enters each word, find out how much space you will need to store it (use a cstring function!). Then allocate the correct amount of space required (careful, don’t be off by one!) for just that one word. If I had a `char* cp`, I could make it point to dynamically allocated space for 7 characters by saying:

```
cp = new char[7];
```

Compare the behavior of this array to the first problem. Answer: Do you get the same behavior when you make the third input word too long? Why not?

3. Repeat the previous problem², but this time also dynamically allocate the array of char pointers. It will help to think of this: the name of an array of char is a pointer to char. The name of an array of pointers to char is, then, a pointer to a pointer to char. In this case, that’s type `char**`.
4. Code (in teams) the tic tac toe program we discuss in class Wednesday. Testing code will be due Sunday night, and **all** code will be due with this lab Tuesday night.

You should have a total of 4 programs named lab03.1.cc to lab03.4.cc, plus your game code and game test code. Make a single script file (see lab00 for the instructions) where you cat, compile, and run lab code in its final form. For the game, the TA will run your submitted testing and game code during the next lab.

¹Note: just because this “works” doesn’t mean it is ok. This is an error. Any time we write past the end of an array it is an error.

²No, you may not just jump to this one, do each version separately.

Submit all 4 program files *and* your script on WebCT by midnight before your next lab. Give the paper version of the complete script file **only** on paper to your TA at the **beginning** of your next lab. Note: Cat, compile, and run each program in order - do *not* cat all programs, then compile, etc.