CISC 3110

Project 2: Snake (part 1)

In this project, you'll get a bit of experience working with non-standard libraries, and with the difference between object-oriented design and "pre-object-oriented" design.

The Game

You are probably familiar with the old-school game called snake. If not, you can play it online at playsnake.org (the control keys are the arrow keys). You can also play it on your Linux/vagrant machine. To do that, run the command

`$ sudo apt install bsdgames`   (If you're asked for a password, use `vagrant`.)

Then run the command

```
$ worm
```

You can always say

```
$ man worm
```

to learn more about the game.

But for this project, we're going to adapt someone's implementation of the game.

Getting Started

Like most applications that use the Linux terminal screen in interesting ways, the code we'll be using is based on the `ncurses` library. (Don't worry; I'm not going to ask you to write code that uses this directly, but you need to understand essentially what's going on.) To be able to use `ncurses` in our code, we have to install the library:

`$ sudo apt-get install ncurses-dev`   (Again, the password is `vagrant`.)

Now, about the code. There's a (mostly) functioning implementation of the game on stackoverflow. Look at the code and the comments. I've made a *very* slightly modified version of the code available at https://bc-cisc3110-f16.github.io/snake_c.cpp. (Essentially, my modifications take C code and make it enough like C++ that g++ will compile it.) Download this code and compile and run it in vagrant. To compile, you need to tell the linker to use the `ncurses` library:

```
$ g++ snake_c.cpp -lncurses
```

Now run it—it works, more or less. See if you can relate the comments about bugs on stackoverflow with the behavior of the running code.

What's Next

Next week, I will give you a slightly revised version of this code—mostly, improving on some of the ncurses code. Then I will ask you to "translate" this code into object-oriented C++ code. Instead of structs and functions, we'll have full classes (`Snake`, `Position`, `Food`, maybe others). And I'll ask you to fix some of the other problems and add a little bit of behavior.