Angela Buob

Anthony Ninthala

Mike Welch

For our snake game program to work, we are going to need numerous methods.

A method that will be needed is one that will create the size of the playing field and creating something like a grid of pixels, so every pixel has a unique address. An extra part to this would be to prompt the user to input a character to change the size of the grid before they begin. For example, they could input ‘S’ for a small playing field, ‘D’ for the default playing field, or a ‘L’ for a large one. If we have time, we could also implement a ‘O’ input variable that would generate a field with obstacles. To start the game, the user would hit ‘ENTER’

One other method that will be needed is one that will create the initial snake that is created when the program is started. This can be done by setting some type of array of a default length. As well as just creating the initial snake, we would need a method that can create the starting point of the game. This can be accomplished with a pointer to a certain spot on the playing grid that will indicate where the head of the snake will be initially printed to when the game starts.

A third method will have to control how the snake moves. The arrow keys can be used to move the snake in the four directions, however, only three of the directions will be able to be used at a time. If all four could be used at all times, then if you hit the arrow key that is the opposite direction of the current movement of the snake, it would cause the snake to ‘kill’ itself since it will touch another part of its body. To go along with this, we would need to force all of the pixels of the snake to follow the exact same path as the head of snake. When the snake moves from going right to going down, the entire snake cannot start going up at once. It will need to take the same path as the head of the snake, and turn up at the same location as when the head did. As well as these, there would need to be a timer that would cause the head of the snake to move one pixel forward in a uniform way, as well as moving at a decent speed. If the user wants to play two player, they should hit ‘W’ before it starts, which would initialize ‘WASD’ as the second set of arrows. When in two player mode, the playing grid will default to the large game grid.

Another method that will be necessary is to have one that will increase the size of the snake. As you play, the snake will move around, trying to get to a highlighted pixel, called an ‘apple’ or ‘pellet’. Once the snake goes through the pixel, the snake will become longer. To do so, we would need to have a function that will add more pixels to the snake somewhere between the head of the snake and the tail. First, the pointer to the tail of the snake will have to be pushed back before new pixels could be added.

Also, a method that can print the pellet to the screen in a random location. We could use the rand() and srand() operators to have the program print out the pellet to a random location. Once the location of the head of the snake is equal to the location of the pellet, it will move the pellet to another random location on the game board. However, every time the pellet changes location, it cannot be printed to the same location as any part of the snake, since it will make it seem like it never printed anything. And as you get to the end of the game, it will become very difficult to be able to tell where it is, as the snake will take up almost all of the game board.

The main thing that will need to be researched how to create a GUI for the program. Without it, the program would be pointless since no human would be able to tell what is going on or what to do.