Descriptions

Approximation of pi program

The design of this program starts with first translating the pascal to the c++ code. Seeds are hard-coded in the "main.cpp" file since professor Vlad is asking for a specific set of seeds.

Next, since we are measuring shots landing in the circle, and not landing in the circle, we must keep track of the distances from the centre point, and the point in which the shot landed. This is done in my program by taking the simple pythagorean's theorem a^2 + b^2 = c^2 , where a is the difference between the X value of the shot that was attempted, and the X value of the centre shot and b is the difference between the Y value of the shot that was attempted, and the Y value of the centre shot. Given this, if the
distance is less than 0.5, then we know the shot landed in the circle, otherwise it did not. Shots are generated by utilizing the number generator for a random X and Y value for every new shot.

So the approximation is computed with the given formula that professor gave to us after every 1000 shots and printed into the cmd.

Pairing Primes program

The design of this program follows very closely with the pseudocode that was provided by Professor. I first initialize a boolean array of length 30000, and following the pseudocode, I start with the index of 2 and set that to true. Next, I push that value into a stack for later use to determine pairs.

Next, number of primes will be counted for the next array for the prime array that I will use to
store all of the prime numbers. The factors of the prime number that I found will now be passed as a parameter to increment the
boolean array and set "true" to all of the values we know are not prime. Continue the loop until completed the length of the array.

The stack is now containing all of the prime values from 2 to 30,000. Now, I store this into an array to make for ease of computing the pairs in which their difference is 2. I loop through the array comparing every value against each other. If their difference is 2,
then print the pair.

Every effort was made to conserve as much of the space as possible, so that every prime number and every pair is able to fit on the command prompt.

Simulation of Life program

The number generator is reused from part one of the assignment. A char 2D array is declared to store the boards, where a nested loop is used to determine randomly if a cell is dead or alive with the generator provided, and 3 seeds the user inputs. I used the logic directly from the assignment page in determining if a cell is dead or alive in the next generation, and then that cell is set
onto a temporary array, so that all of the cells are determined to be dead or alive for next generation without affecting the current generation. Generations can be generated for as many generations as the user likes by inputting "1" into the program.