# Lab 3

Problem: Program the “Game of Life”



# Source File:

/\*Problem: Program the "Game of Life"

Author: Andrew Naro

January 27th, 2014

Inputs: None

Outputs: Game of Life

\*/

#include <iostream>

#include <ctime>

using namespace std;

void initializeBoard(char currentGeneration[][20]);

void newGeneration(char currentGeneration[][20],char nextGeneration[][20]);

int neighbors(char currentGeneration[][20],int i, int j);

void printBoard(char currentGeneration[][20]);

int main()

{

/\*

Passes currentGeneration to initializeBoard to be filled

While the user wants to continue, creates new generations and prints them out

\*/

char currentGeneration[20][20];

char nextGeneration[20][20];

int repeat=1;

cout<<"Welcome to the Game of Life!\n\n";

initializeBoard(currentGeneration);

do{

newGeneration(currentGeneration,nextGeneration);

cout<<endl;

printBoard(currentGeneration); //the 'first' generation is generated from the random board to give a more stable starting point

cout<<"\n\nEnter 1 to continue or any other key to quit: ";

cin>>repeat;

}while(repeat==1);

return 0;

}

void initializeBoard(char currentGeneration[][20])

{

/\*

Takes currentGeneration and fills the inside of the perimeter randomly with cells

\*/

srand(time(0));

int filler=0;

for(int i=1;i<19;i++)

{

for(int j=1;j<19;j++)

{

filler=rand()%8;

if(filler==0)

{

currentGeneration[i][j]='\*'; //stars are cells

}

else

{

currentGeneration[i][j]='.'; //dots are non-cells

}

}

}

//building the frame of the board which will be hidden from the user

for(int f=0;f<20;f++)

{

for(int g=0;g<20;g++)

{

if(f==0||f==19)

currentGeneration[f][g]='.';

else

{

currentGeneration[f][g]='.';

currentGeneration[f][19]='.';

break;

}

}

}

}

void newGeneration(char currentGeneration[][20],char nextGeneration[][20])

{

/\*

Takes currentGeneration and builds the nextGeneration with it

Sends each individual coordinate to neighbors() to determine quantity of neighbors, effecting births and deaths

After nextGeneration is built, currentGeneration becomes the nextGeneration

\*/

int neighborcount=0;

for(int i=1;i<19;i++)

{

for(int j=1;j<19;j++)

{

if(currentGeneration[i][j]=='.')

{

if(neighbors(currentGeneration,i,j)==3)

nextGeneration[i][j]='\*';

else

nextGeneration[i][j]='.';

}

if(currentGeneration[i][j]=='\*')

{

neighborcount=neighbors(currentGeneration,i,j);

if(neighborcount>1&&neighborcount<=3)

nextGeneration[i][j]='\*';

else

nextGeneration[i][j]='.';

}

}

}

for(int i=1;i<19;i++)

for(int j=1;j<19;j++)

currentGeneration[i][j]=nextGeneration[i][j];

}

int neighbors(char currentGeneration[][20],int i, int j)

{

/\*

Counts the number of neighbors at each coordinate

\*/

//Neighbors are i-1=>i+1 x j-1=>j+1, but not i x j

int neighbors=0;

for(int n=-1;n<2;n++)

{

for(int m=-1;m<2;m++)

{

if(n==0&&m==0)

m++;

if(currentGeneration[i+n][j+m]=='\*')

neighbors++;

}

}

//cout<<i<<"x"<<j<<": "<<neighbors<<"\t";

return neighbors;

}

void printBoard(char currentGeneration[][20])

{

/\*

Prints out currentGeneration

\*/

for(int i=1;i<19;i++)

{

for(int j=1;j<19;j++)

{

cout<<currentGeneration[i][j];

}

cout<<endl;

}

}

# Sample Run:

Welcome to the Game of Life!

..................

..................

..................

..................

..................

..................

..................

..................

..................

.\*\*............\*..

.\*..............\*.

..........\*\*......

.........\*\*.......

..................

...\*..............

..................

..................

..................

Enter 1 to continue or any other key to quit: 1

..................

..................

..................

..................

..................

..................

..................

..................

..................

.\*\*...............

.\*\*...............

.........\*\*\*......

.........\*\*\*......

..................

..................

..................

..................

..................

Enter 1 to continue or any other key to quit: 1

..................

..................

..................

..................

..................

..................

..................

..................

..................

.\*\*...............

.\*\*.......\*.......

.........\*.\*......

.........\*.\*......

..........\*.......

..................

..................

..................

..................

Enter 1 to continue or any other key to quit: