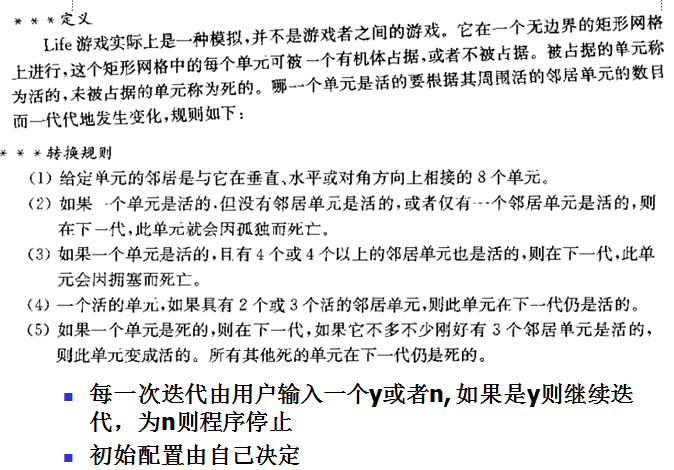
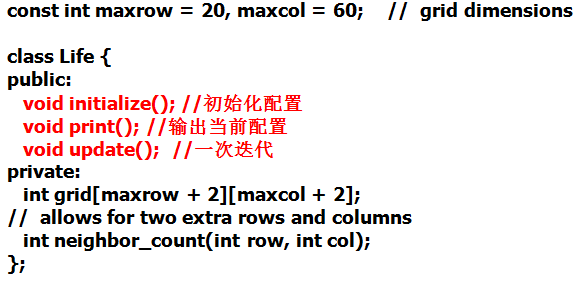
**WEEK2 Life Problem**

郁思敏 10153901225

**question：**

****

****

**codes:**

**1.life.h**

#ifndef LIFE\_H\_INCLUDED

#define LIFE\_H\_INCLUDED

const int maxrow = 20, maxcol = 60;

class Life

{

public:

void initialize(); //initialize configuration

void print(); //print current configuration

void print\_neighbor(); //print the number of neighbor

void update(); //one iteration

private:

int grid[maxrow+2][maxcol+2]; //record if it is live or die

int neighbor\_count(int row, int col); //calculate its neighbor\_count

};

#endif // LIFE\_H\_INCLUDED

**2.life.cpp**

#include <iostream>

#include<stdlib.h>

#include "life.h"

using namespace std;

int in\_row=0, in\_col=0; //the input row and column

void Life::initialize() //initialize configuration

{

int cnt, i, j, x, y;

for(i=0; i<maxrow; i++)

{

for(j=0; j<maxcol; j++)

{

Life::grid[i][j]=0;

}

}

cout << "enter the row of your array:(row<=20)" <<endl;

cin >> in\_row;

cout << "enter the column of your array:(col<=60)" <<endl;

cin >> in\_col;

cout << "enter the total number of life:" << endl;

cin >> cnt;

cout << "enter the (x,y) of life:" << endl;

for (i=0; i<cnt; i++)

{

cin >> x;

cin >> y;

Life::grid[x][y]=1;

}

}

void Life::print() //print current configuration

{

int i, j;

for(i=0; i<in\_row; i++)

{

for(j=0; j<in\_col; j++)

{

cout << Life::grid[i][j] << " ";

}

cout << endl;

}

}

int Life::neighbor\_count(int row, int col) //calculate its neighbor\_count

{

int ncount=0;

int i, j;

for (i=((row-1)>=0?(row-1):row); i<=((row+1)<in\_row?(row+1):row); i++)

{

for (j=((col-1)>=0?(col-1):col); j<=((col+1)<in\_col?(col+1):col); j++)

{

if(Life::grid[i][j]==1)

{

ncount++;

}

}

}

ncount-=Life::grid[row][col];

return ncount;

}

void Life::print\_neighbor() //print the number of neighbor

{

int i, j;

for(i=0; i<in\_row; i++)

{

for(j=0; j<in\_col; j++)

{

cout << Life::neighbor\_count(i,j) << " ";

}

cout << endl;

}

}

void Life::update() //one iteration

{

int i, j;

int sum, self;

int backup\_array[in\_row][in\_col]; //backup pre\_neighbor\_count

for(i=0; i<in\_row; i++)

{

for(j=0; j<in\_col; j++)

{

backup\_array[i][j]=Life::neighbor\_count(i,j);

}

}

for(i=0; i<in\_row; i++)

{

for(j=0; j<in\_col; j++)

{

sum=backup\_array[i][j]; //pre\_neighbor\_count

self=Life::grid[i][j]; //1:life 0:die

if(sum>=4 || sum<=1)

Life::grid[i][j]=0;

if(sum==3)

Life::grid[i][j]=1;

if (sum==2 && self==0)

Life::grid[i][j]=0;

}

}

}

**3.main.cpp**

#include <iostream>

#include<stdlib.h>

#include "life.h"

#include "life.cpp"

using namespace std;

int main()

{

char ch;

Life life1;

life1.initialize();

cout << "print:" << endl;

life1.print();

cout << "next:y/n" << endl;

cin >> ch;

while(ch=='y'||'Y') //if 'y' or 'Y', enter next iteration

{

cout << "neighbor\_count:" <<endl;

life1.print\_neighbor(); //print neighbor\_count

life1.update(); //update

cout << "print:" << endl;

life1.print(); //print life statement after one iteration

cout << "next:y/n" << endl; //ask whether need next iteration

cin >> ch;

if(ch!='y' && ch!='Y')

exit(0);

}

return 0;

}