In-Class Work: Game Of Life

1. Initialization And Printing

Write a program that initializes a 5x5 lattice as follows:

0	0	0	0	0
0	1	1	1	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

Define the size, i.e. 5, as constant in the header of your program, i.e. before main. This will allow us later to change the size of our lattice easily. Then print the lattice on the screen in the format as shown above.

2. Von Neumann Neighbors

2a. Determine Neighbors

Next you determine the von Neumann neighbor lattice sites and their values. To be able to test our program initialize the 5x5 lattice as follows:

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Print out the lattice (same as in 1. but for this lattice). Now read in (from screen) two integers `isite` and `jsite` which specify the site for which you try to determine its von Neumann neighbors. Print out the lattice values for the site `(isite,jsite)` and the lattice values of the right,left, top and bottom sites. Check your result for different values of `(isite,jsite)`. Be careful to use periodic boundary counditions. (Hint: There is an elegant way to take into account the periodic boundary conditions by using the modulo function %.)

2b. Number of Neighbors

You are now ready to determine the number of neighbors (=number of living neighbors). Change your program such that it reads in the initial configuration (5x5 lattice) from the file ¨ kvollmay/sunhome/classes.dir/capstone_s2008.dir/game_of_life.dir/init_5x5_rand.data
The program then prints the lattice and reads in a lattice site `(isite,jsite)`. For this site the program determines the number of living von Neumann neighbors and prints out the result.

2c. Do the same as in 2b. but write a function which has as input the lattice site and returns the number of living neighbors.
Hint: To declare (and similarly to define) the function use for example:
int vonNeumann (const int lattice [LATSIZE][LATSIZE],int isite,int jsite);

In-Class Work: Game Of Life (Newcomers)

1. Initialization And Printing

Write a program that initializes a 5x5 lattice as follows:

0	0	0	0	0
0	1	1	1	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

Define the size, i.e. 5, as constant in the header of your program, i.e. before main. This will allow us later to change the size of our lattice easily. Then print the lattice on the screen in the format as shown above.

2. Von Neumann Neighbors

2a. Determine Neighbors

Next you determine the von Neumann neighbor lattice sites and their values. Copy
~ kvollmay/sunhome/classes.dir/capstone_s2008.dir/game_of_life.dir/game2a_sample.cc
into your working directory. The program initializes and prints a 5x5 lattice as follows:

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

The next task for the game of life is to find neighbors. We use this lattice to be able to test our program. Add to the program that it reads in (from screen) two integers `isite` and `jsite` (each $\in \{0, 1, 2, 3, 4\}$) which specify the site for which you try to determine its von Neumann neighbors. Print out the lattice values for the site `(isite,jsite)` and the lattice values of the right,left, top and bottom sites. Check your result for different values of `(isite,jsite)`. Be careful to use periodic boundary counditions. (Hint: There is an elegant way to take into account the periodic boundary conditions by using the modulo function %. Remember for example $5\%2 = 1$.)

2b. Number of Living Neighbors

You are now ready to determine the number of living neighbors. Start with copying into your working directory the following two files:
~ kvollmay/sunhome/classes.dir/capstone_s2008.dir/game_of_life.dir/init_5x5_rand.data
~ kvollmay/sunhome/classes.dir/capstone_s2008.dir/game_of_life.dir/game2b_sample.cc
The sample program reads in the initial configuration (5x5 lattice) from the file init_5x5_rand.data. The program reads in a lattice site `(isite,jsite)`. Add to the program that it determines the number of living von Neumann neighbors and prints out the result.

3. Discuss the complete flow chart with your neighbors. Make a plan how to complete your program to simulate the game of life. Please get me, when your plan is finished.

In-Class Work: Game Of Life (Advanced)

1. Initialization And Printing

Write a program that initializes a 5x5 lattice as follows:

0	0	0	0	0
0	1	1	1	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

Define the size, i.e. 5, as constant in the header of your program, i.e. before main. This will allow us later to change the size of our lattice easily. Then print the lattice on the screen in the format as shown above.

2. Von Neumann Neighbors

2a. Determine Neighbors

Next you determine the von Neumann neighbor lattice sites and their values. To be able to test our program initialize the 5x5 lattice as follows:

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Print out the lattice (same as in 1. but for this lattice). Now read in (from screen) two integers `isite` and `jsite` (each $\in \{0, 1, 2, 3, 4\}$) which specify the site for which you try to determine its von Neumann neighbors. Print out the lattice values for the site `(isite,jsite)` and the lattice values of the right,left, top and bottom sites. Check your result for different values of `(isite,jsite)`. Be careful to use periodic boundary counditions. (Hint: There is an elegant way to take into account the periodic boundary conditions by using the modulo function %.)

2b. Number of Living Neighbors

You are now ready to determine the number of living neighbors. Change your program such that it reads in the initial configuration (5x5 lattice) from the file
~ kvollmay/sunhome/classes.dir/capstone_s2008.dir/game_of_life.dir/init_5x5_rand.data
(You may want to first copy init_5x5_rand.data into your working directory.) The program then prints the lattice and reads in a lattice site `(isite,jsite)`. For this site the program determines the number of living von Neumann neighbors and prints out the result.

2c. Do the same as in 2b. but write a function which has as input the lattice site and returns the number of living neighbors.
Hint: To declare (and similarly to define) the function use for example:
int vonNeumann (const int lattice [LATSIZE][LATSIZE],int isite,int jsite);

3. Discuss the complete flow chart with your neighbors. Make a plan how to complete your program to simulate the game of life. Please get me, when your plan is finished.

In-Class Work: Game Of Life (Newcomers)

3. Finish Game of Life Program

3a. Use your program (or the solution) for 2b or 2c. Add the main part of the game of life program: the loop over the cells to determine each new cell value. After this loop update all cells at once. (See Flow Chart) Remember that you need for this two two-dimensional arrays. For example if your lattice is called "lattice" then while you determine the new lattice sites determine the neighbors with lattice, but write the new cell values into another array, e.g. "newlattice". Check your program with the result for t=1 in the file
~kvollmay/sunhome/classes.dir/capstone_s2008.dir/game_of_life.dir/game3b_data

3b Add the time loop (see flow chart) and check your result after 10 timesteps with the result in game3b_data.

Solutions:
~kvollmay/sunhome/classes.dir/capstone_s2008.dir/game_of_life.dir/game2a.cc etc.

In-Class Work: Game Of Life (Advanced)

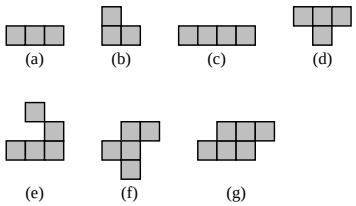

Fig.1: Initial Configurations to be used with Moore neighborhood (see 5b.)

3a,b Work through Newcomer-Page (backpage) first.

4. Graphics

4a Let us see how you can watch a movie of the game of life. Use your program from 3b. and change the output such that it looks on the screen like the file
~kvollmay/sunhome/classes.dir/capstone_s2008.dir/game_of_life.dir/game4_data
So add at the beginning of each configuration two lines, the first with "`#pause 2`" and the second with "`#string time = timevalue`". Add after each configuration a blank line. Then look at the resulting output. In case you print on the screen then use:
executablefilename | `DynamicLattice -nx 5 -ny 5 -matrix`
or in case your output is in a file "fileout":
`DynamicLattice -nx 5 -ny 5 -matrix < fileout`
Change the number "2" to learn about its effect. The "5" is specifying the size of the lattice in the horizontal and vertical direction.

4b Now let's watch a movie of a 10x10 lattice. Read in the initial configuration
~kvollmay/sunhome/classes.dir/capstone_s2008.dir/game_of_life.dir/init_10x10_rand.data
and change in your program the lattice size from 5 to 10. Rerun your program and adjust the command of Dynamic Lattice accordingly.

5. Moore and Patterns

5a. Use your program of 4., that means start with the 5x5 lattice of file
~kvollmay/sunhome/classes.dir/capstone_s2008.dir/game_of_life.dir/init_5x5_rand.data
and do 10 timesteps. Instead of the von Neumann neighbors use Moore neighbors (add another function). Check your result with the file
~kvollmay/sunhome/classes.dir/capstone_s2008.dir/game_of_life.dir/game5a_data

5b. Now start with different initial configurations. Use a 30x30 lattice with all cells dead but approximately in the middle of the lattice the pattern of Fig.1a of alive cells. Using Dynamic Lattice watch the pattern how it changes with time. Repeat this for the patterns b - g. (Use the Moore neighborhood.)

In-Class Work: Game Of Life (Newcomers)

3. Finish Game of Life Program

3b. Use your program from the in-class work 3a. or the solution program
~kvollmay/sunhome/classes.dir/capstone_s2008.dir/game_of_life.dir/game3a.cc
i.e. your program which includes the game of life rules but does not yet have the time loop. Add the time loop (see flow chart) and check your result after 10 timesteps with the result in
~kvollmay/sunhome/classes.dir/capstone_s2008.dir/game_of_life.dir/game3b_data

4. Movie

4a Let us see how you can watch a movie of the game of life. Use your program from 3b. and change the output such that it looks on the screen like the file
~kvollmay/sunhome/classes.dir/capstone_s2008.dir/game_of_life.dir/game4_data
So add at the beginning of each configuration two lines, the first with "`#pause 2`" and the second with "`#string time = timevalue`". Add after each configuration a blank line. Then look at the resulting output. In case you print on the screen then use:
executablefilename | `DynamicLattice -nx 5 -ny 5 -matrix`
or in case your output is in a file "fileout":
`DynamicLattice -nx 5 -ny 5 -matrix < fileout`
Change the number "2" to learn about its effect. The "5" is specifying the size of the lattice in the horizontal and vertical direction.

4b Now let's watch a movie of a 10x10 lattice. Read in the initial configuration
~kvollmay/sunhome/classes.dir/capstone_s2008.dir/game_of_life.dir/init_10x10_rand.data
and change in your program the lattice size from 5 to 10. Rerun your program and adjust the command of Dynamic Lattice accordingly.

In-Class Work: Game Of Life (Advanced)

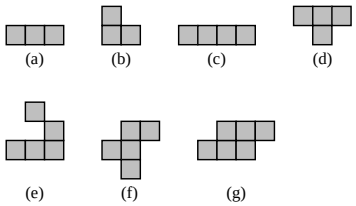

Fig.1: Initial Configurations to be used with Moore neighborhood (see 5b.)

4. Work through Newcomer-Page (backpage) first.

5. Moore and Patterns

5a. Use your program of 4., that means start with the 5x5 lattice of file
~kvollmay/sunhome/classes.dir/capstone_s2008.dir/game_of_life.dir/init_5x5_rand.data
and do 10 timesteps. Instead of the von Neumann neighbors use Moore neighbors (add another function). Check your result with the file
~kvollmay/sunhome/classes.dir/capstone_s2008.dir/game_of_life.dir/game5a_data

5b. Now start with different initial configurations. Use a 30x30 lattice with all cells dead but approximately in the middle of the lattice the pattern of Fig.1a of alive cells. Using Dynamic Lattice watch the pattern how it changes with time. Repeat this for the patterns b - g. (Use the Moore neighborhood.)

6. Parity Rules

Before you work on this program, please let me know, so that I can explain some background. Next we change the rules of the game of life. Use the following "Parity Rule": For a given site count how many living von Neumann neighbors the site has, add to the number of living sites the number of the lattice value at the site itself. If the resulting number is even, then the new site value is 0. If the resulting number is odd, then the new site value is 1.
Use a large lattice (e.g. 100x100) and start with all cells dead but a square (2x2 or larger) of living cells in the middle. Run your program and look at it with DynamicLattice.

7. Population Growth

Besides watching some cool movies of the game of life let us analyze the simulations differently. To do so let us go back to the original motivation of the game of life rules. The lattice values represent people being alive or dead. Change your program so that it does not print the lattice but instead it writes on screen for each time step the time t and the number of all living cells on the whole lattice N. You can look at the result $N(t)$ by running your executable :
$executablefilename$ | `xgraph -m`