****

Data Structure   
Project Report

**Topic：**

**Faculty**

**Program**  Ecosystem

**Name LifeGame**

**Student ID**

**Lecturer**

**Course Code**

**Credit**

**Start/End Date 12.30~2.26**

|  |  |
| --- | --- |
| Teacher’s Comment |  |
| Score |  |
| Remark |  |

**Topic**

**Background**

We always believe that life has its simple intrinsic nature, a series of restricting relationship can produce the uncertain results. So we create the imitating ecosystem, it includes 3 creatures, grass(producer), rats (primary consumer）,and wolves (high consumer). To every creature, there are 12 parameters to decide it. The parameters include: the least amount to give birth to new members, the most amount to make them die, the degree of activity, the age limits, the mature age, the most time for them to being hungry, and so on.

Every life in the system has :

1. reproduce: the random self duplicating when the creature is mature, in the process.
2. die: creatures die when satisfy certain conditions
3. be caught: decrease the feeling of hunger by eating other creatures.
4. competition: the competition between species, when fails, creatures die.
5. movement: move to where there is more food.

**Design Principle**

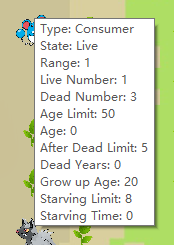
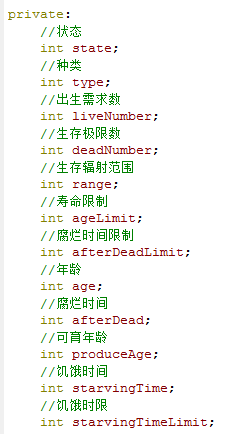
1. **Back-end code:**

The after end code executes the main actions of the cell. It including these parts:

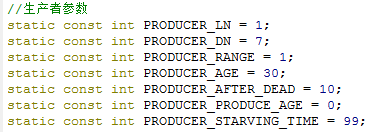
1. Objects with attributes.

The class cell shows this part, in the class, we have 12 private integer member and their get & set functions, we also provide the function of initialization and propagate.

The 12 attributions are state, type, age, age limit, the least burn relations number, the most number in a range with relations, the live range, the grow up age, the have starving time, the longest time for an object that in starving state, the age limit of an object and the time for the corpse become putrid. We add many limits in some get or set functions to make sure the safety of this attributions.



For initialization, we provide some method by give the default value of the parameter.

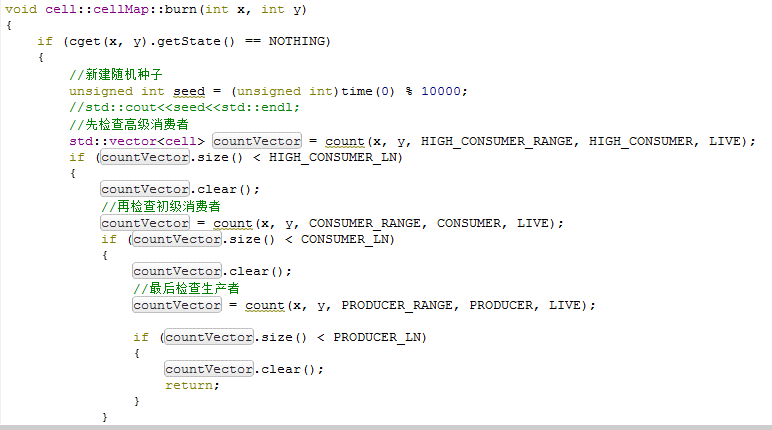


For propagate, we copy most of attributions from the parent but set age, starving time and putrid time to 0.

1. Objects’ actions.

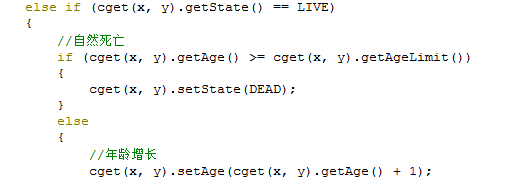
We write some objects’ actions in class cellMap and others in class MainWindow, decide by we release in which part is more convenient. Let me introduce them:

Burn: control the burn of the objects, when the function run, it checks the empty spaces of the map, when it finds an empty space, it checks whether there are some objects nearby or not, if yes, then a new object burn by copy the attribution of a random object in the range. When an object is copy, there is a chance (can be controlled by user) that the evolution happened, when it happened, 6 attribution of the object can be changed in a range randomly. (So when you run the program, at last you always see one type objects become crazy as it happens some very useful change and our micro ecosystem is too easy to hold these objects.).

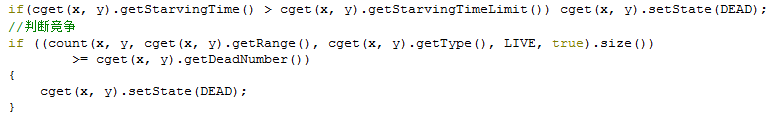


Dead: there are three methods to make an object dead, including dead of age, dead of predation and dead of intraspecific competition.

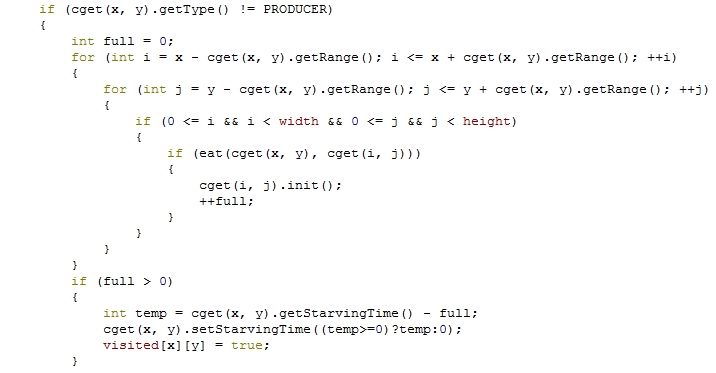
When an object’s age is larger than or equal to its age limit, it will be dead, and its state become a corpse.



When in an object’s range, there are too many relations, it will happen intraspecific competition, and some objects will be dead.



When an object is eaten by other objects, it will disappear.

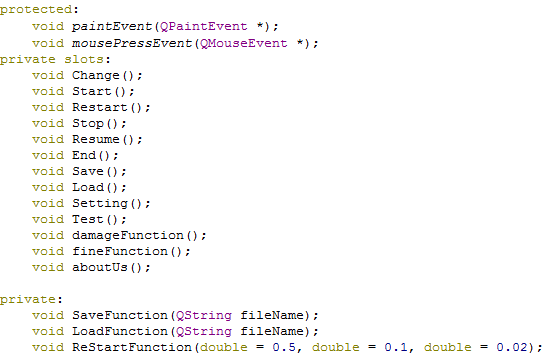


What’s more, after an object dead, it will become a corpse and disappear after some time. At the old version, corpse can be eaten by not only other type objects but also the same types, but my teamer thinks it is too cruel, so I deleted this.

Move: When an objects cannot find foods in the range of its current position, it will move and seek the food by search, the method is similar to a limited BFS, in other topic we will have more clearly explain.

1. The array which hold objects and project them to the GUI with some control-functions.

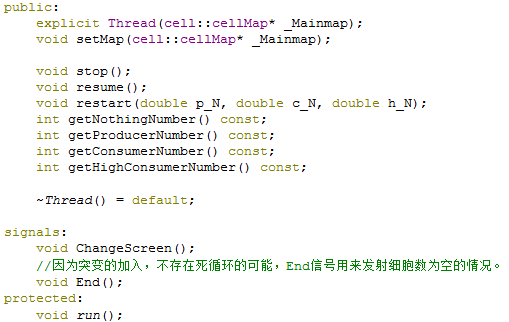
In this part, there are some settings interface for user, including set the speed and the evolution chance of the map, and the GUI will get the array information and paint it by photos.



1. **Thread:**

We use thread to let user control the micro ecosystem when it is running. Now I will talk about this:

We let the user options leave in the main thread and we set up a new thread to run our map. In the class Thread we run the map by scanning the objects of map one by one, then collision the information of the micro ecosystem and run the functions of objects’ actions. What’s more, when user do some options in the GUI, it will call some functions in the class Thread to control the objects in the map.



1. **GUI:**

GUI include one main window and three control dialog, they are used to relate the options of user and show the micro ecosystem to user.

1. Main Window.



I show you the style of the main window and explain it to you:

‘start’, ‘pause’, ‘resume’ are design for control the thread so that the user can control the process of the micro ecosystem. ‘damage’ button provides a method that killing some of the objects and ‘fine’ button provides a method to randomly set up some new objects.

‘Save’ and ‘Load’ can save map state and you can run it next time, you need to be careful that do not save the save data in a Chinese name or path, which will lead to the load bug.

‘end’ ‘restart’ ‘setting’ and ‘about us’ will call their dialog.

1. Dialog for end.



You can end the ecosystem and see the information of the map.

1. Dialog for setting.



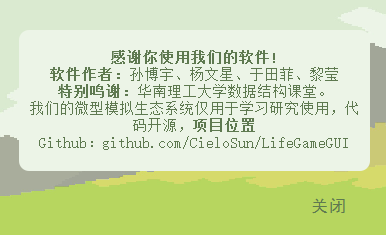
In this part, you can see the information and set the speed or the evolution chance of the micro ecosystem, you can also add some objects in the map.

1. Dialog for reset.



You can set the chance of the objects in the map and restart the micro ecosystem.

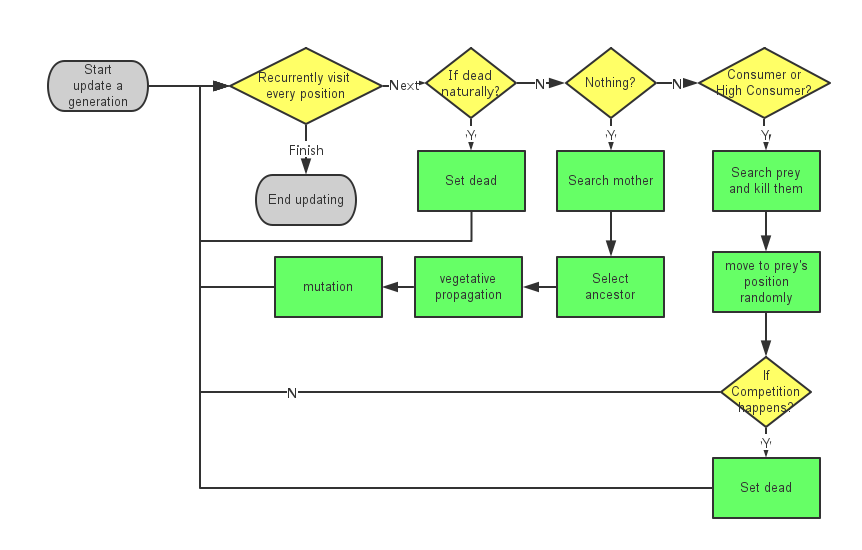
1. Dialog for about us



It will show the about us information.

**Model Discussion**

# 



1. **Particular points:**
2. If it is nothing, A new living may be burned. It can be producer, consumer or high consumer. It depends on number of livings around it. It won’t be burned if the number is too high or too low.
3. For consumer and high consumer, it has a random life expectancy and random limit for starvation. When it reach one of its limit, it will die.
4. Only consumer will eat producer and only high consumer will eat consumer.
5. For consumer and high consumer, it will eat all its prey in its releam. It can reduce its starvation.
6. For consumer and high consumer, if there are too many livings of the same type nearby, they will die from competition.
7. “Favourable weather” and “Disasters” will reproduce/kill livings.
8. The system will terminate if all livings die out.
9. The succession completely depends on seeds if there is no manual intervention.
10. **Complexity Analysis:**

For m\*n maps

1. O(mn) for map generation, saving and map update.
2. O(mn) for map update, an constant time for range searching, depending on build-in parameters.
3. **Data Structure :**
4. 2-dimensional array to store all the livings.
5. Linked list for temporary storing consumers that will be updated. This can avoid new data overwriting old data before the scanning is finished.
6. **Demonstration**

Click the start button, the ecosystem begins running! As you see, the background is a grass land with green grass, blue rats and gray wolf. Now you are the god of them, you can control the starting, continuing and ending of the whole ecosystem using the buttons on the right. Of course you can control the state of the whole system --is there disasters or it just being good all the time, or you can add up the amount of a certain kind of feature in the system. When the system run, their life begins, considering the capability of the ecosystem and the needs of the species, only when the amount of the species is large enough and smaller than the limit amount can it stay alive. The alive ones grows up, and when its mature enough (the appearance of it changes), it can reproduce the next generation (with certain rate of variation, but the appearance has no change, only the attributes). But bad things may happen to them, they might come across hunger, and if it is severe enough (measured by time), they will be hungry to death, or they will be caught and eaten by their predators. And if they are lucky enough, they can be alive till their end of life. After they die, their body will be broken down by the decomposer or disappear of other reasons. (the features mentioned above are given in advance, you can click then to see their states at the moment).

Also, you can save this game data in your computer, be careful, the path you save the data have to only contain English. Then, the next time you want to begin this game in the same place you only need to open the file. Besides, there is some information about us and exit button.

**Conclusion**

1.In the processing of completing LifeGame program, we had a clear and efficient division of labor and always helped each other.At last,both us learn some things new and our efforts is rewarded.

2.We build a ecosystem using Cellular Automaton Model and Thread which real described the relationship and behavior between species successfully. Also, the user can change the state of this ecosystem as they want and get the information of organisms.

**Individual Report**

CieloSun：

I am the leader of the team, in this work, I am the manager and the main coder. I design the app’s functions. Then I write the basic functions of the program and design a part of the basic frame logic of GUI.

When I code for the program, the first difficult is that we never work in 4 different place, which means we cannot work together, so I study to use the github to manage our program, now it become my daily tool.

Then we never used thread, at first we try to study the thread part of C++ 11, and my teamer Wenxing Yang write nearly 1000 line codes, but we failed, we at last have to give up for it is too difficult that many bugs happen and we do not know how to do. I decide to study the thread method of Qt and this time, I success. (That is why my code amount become very large in github, but my teamers’ code looks few)

Next, I start to understand some method to debug the GUI app, I have never due with so much bugs when I develop other little GUI program before. As it is not a small program, you cannot due with these problems as the same as console app.

By this work, I study many technology of GUI, understand the version control technology and understand concurrency in action. Maybe in my part there is few data structure, but I learnt much things in the reality developing.

The other challenge is that I studied the way to cooperate with others, I always do a work alone before even if it is not a one-person work as I am not good at cooperating with others. This time, for the work is a little large, I cannot do it only by myself, which means I have to cooperate with others. I think in future day**s, most of time you have no chance to code something by only yourself, cooperating with other people is a very important ability.**

Jim9606

The main idea of this project is brought up by me first. I am responsible for the core utility (compared to user interface) and debugging.

At first, I want to implement Conway's Game of Life. This is a Cellular automaton. It rules is very simple: each cell will interactive with 8 cells next to it. Its state at the next generation depends on the number of living neighbors. If there are less than two or more than 3, it will die. Secondly, if there are actually 3 living neighbors, it will bring back to life.

At this project, I take some new rules and new types of cells to the system. The range of each living will be larger but not covering the whole system. I want to test if I can run stability with manual intervention.

I may want to try another programming language to build this project, but I give up and turn back to C++ because I may cause too much time to learn a new language in a winter holiday. Also, we have a bit of experience on GUI library and toolchain on C++. It is not good to give up it.

On our first meeting, I supposed that we should use version control. Compared to High-level Language Programming and Designing’s project, this is a teamwork. If we only use QQ or other IMs to transfer the project’s source archive, it will be difficult to find to modification and it spends much time to configure IDE. Second, when we want to discard a new feature with serious problems, it is not easy to find previous tested version. As a result, we decided to use GitHub to store our source code. It is better way to manage codes, although everyone needs some time to learn how to use Git on Windows and how to create feature branch, fix merge conflicts and rebase.

We don’t have enough experience at such type of projects. I don’t know how much time we have to programming and debugging. I don’t know if it is better to rewrite a new feature than speed unknown time to fix a bug program. What’s more, I may have other urgent matters although I had planned to take this project during my holiday.

To solve this program, I supposed that we should take a usable version with only part of our target features and think about expansibility when designing classes and application interfaces. For example, our first version contains only one species. There is no breeding, hunting, and data statistics. However, we left some attributes and interfaces for new features like starving and age. We try to separate user interaction and core utility. For early stage, use command line to test the program is more convenient.

We have some unfinished features because of time, such as self-optimized global parameters. It can tweak the rules according to the statistics. I also wants to allow users to generate an particular system with the same evolution road. It will use a pseudo-random generator with user-defined seed. It is also a good way for ourselves to regenerate a special system which will take out bugs.

During this project, I have learned how to take a development project with teammates’ communication. I also know how to review others’ code when it is in another programming style, which has been difficult for us to get its point. Also, I know the advantage to use version control tools to avoid problems in software development.

Chelsea

- Which part you are responsible

My part is mainly connecting the front end system with the backstage . supporter and help to build the interface.

- What are the difficulties?

I find there so many bugs when i call the dialogue. When I try to change the size of the window dynamically, so that it will be more elegant, but many problems came during it,and I just solved it by surfing the internet for answers.And because we are using github and qt language to complete the project, and I had never used them before ,so I read the pdf and learned how to use them,

- What you have learnt

I learned how to use the github, how to use it to managing teamwork coding and how to pull codes into the program. And I think our group did very well in sharing out the work and cooperating with one another, in this process, I knew how to think from other’s side and it will be better if you can express your idea to your partner.

- Any special challenges

In fact, learning how to use github and QT is a little bit challenging for me, and moreover, there are bugs, so being patient is also a requirement for me.

Britney:

Which part you are responsible？

my part is to achieve the function of buttons and the design of interface.

What are the difficulties?

I am a novice at QT and GitHub. So when I dealt with some easy problems I always got in trouble and made the solution complicated. For example, when I changed the background of mainwindow, I had many choices but most of them could not work or it made the program bug. Through hard, I have found an appropriate way. Unfortunately，I got into another problem ,that is，it influenced the background of the children windows (dialogs) but I did not want to use the same background to the dialogs. Then I search it on the internet and consult qt documentation over and over again. At last, I found it’s easy to deal with, that is, there is inheritance relationship between father windows and children windows, if I want to use different background, I only can choose a special way which ensured that it won’t make the background overlay.

What you have learnt?

Actually, I do not think I learnt much about algorithm. But I did learn some things in the process of doing this program. I learnt how to use GitHub to build a program for teamwork and how to fetch code from GitHub or pull your code to it to make teamwork coding efficient. Also, I learnt how to make a beautiful interface rather than a black box.

Any special challenges?

I don’t think I met any special challenges. But learning something new and using it immediately is a challenge, then I met one.