

AP (IT) Coursework 2018 Report

Student Name: Yangjiangdai Pan

Student ID: 2294163P

1. Assumption
   * Spec1:
2. A N \* M grid should be generated every 20 milliseconds once the program be executed. It should be implemented by an independent object.
3. Cars represented by different shape (“-” and “o”) should be generated from the North and the West of the grid.
4. Cars should be able to move to the other side with a random constant speed and fall at the end of the grid.
5. Each cell of the grid can only have one car on it, so they will not crash.
6. The grid should be drawn 2000 times. During the 2000 times, cars should keep being generated and moving.
   * Spec2:
7. Cars should be generated from four directions.
8. Cars from one direction can only use half of the grid.
9. There should be a static class to generate a report which includes the max, min, mean passing time for all cars of every generator.

* I have achieved every requirement.

1. Class design
   * There are 11 classes in total:
2. APSpec1:

Main class for level1. The first cars are generated in different cells from this class. Then the grid thread starts.

1. APSpec2:

Similar to APSpec1 except cars are generated from 4 directions and only take half of the grid.

1. Draw:

An independent object to generate the grid.

1. Car:

This is an abstract class of the car generators.

1. CarZeroToX:

A child class of Car class including methods to get the location and the shape of the cars from West to East, avoid crashing, move the car, and reset the car after falling at the end of the grid. It can also record the passing time of cars if it is run by APSpec2.

1. CarYToZero:

The same as CarZeroToX class except it is for cars from North to South.

1. CarXToZero:

The same as CarZeroToX class except it is for cars from East to West.

1. CarZeroToY:

The same as CarZeroToX class except it is for cars from South to North.

1. Grid:

The class to implement most of the functions. The threads of the grid and the cars keep starting in this class until the grid has been drawn 2000 times. Which cars to be shown on grid is decided in this class. The order of checking next cell for cars and starting threads is the most important and difficult part in my opinion. It should be kept strictly by using Boolean and lock/condition.

1. SeperateMain:

A static class including a static integer s. The integer s can be 1 or 2 decided by main classes. It is used by the other class so the car can be generated in the whole grid or half of the grid. Static class should be carefully used to avoid problems.

1. StaticReport:

A static class including methods for each direction to record min/max passing time for each generator, calculate mean passing time with total car numbers and total time.

1. Testing
   * Generator the grid in less times to check if there are problems for generating report, drawing grid and drawing cars (make sure there is enough time for cars of each direction should have at least one car pass through the grid).
   * Test if the methods to avoid crashing is working (the most important thing to be tested):
2. Use “System.out.println(“”);” to test if methods have been reached as plan during coding.
3. Use “Thread.sleep();” to slow down the process so it is easier to witness if the cars can avoid crashing.
4. Look back every grid has been drawn in console to double check.
5. The order of movement and starting thread has been changed several times to avoid crashing with cars both from other directions and the same direction.
   * Test both APSpec1 and APSpec2 after changes have been made, though these two classes are similar.
   * Test report class for several times to see if it is able to overwrite last report.

The number in report should float in a reasonable range.

1. Questions
   * Critique:
2. The system is most likely to be used for a mixture of auto-driven cars and human-driven cars. There is no method in the system to avoid accidents caused by human mistakes. For example, cars may need to break some rules to avoid larger damage.
3. An appointment method for the system can be useful. The users can make an appointment of when and where to go. Then, the system can make a travel plan.
4. The system should be able to change travel plan after considering the current road condition.
   * Extension can be handled:
5. The amount of left fuel in each car can be recorded, so there can a reminder of low level of fuel.
6. There can be methods for cars to turn back or other directions in the middle of the grid, which happens a lot in real world.
7. The start location can be changed to avoid traffic jam.
   * Larger city area:
8. A real city should have much more than one cross. As a result, an array of crosses can help to divide cars at different crosses. This should increase the efficiency of the system.
9. There are not only cars but also pedestrians on the road in real world. There should be rules for avoid pedestrians. A traffic lights method can be added. There can also be method to avoid pedestrians who do not follow traffic rules (like drunk people, which are pretty common in Glasgow), so the car can make a plan to cause minimum damage.
10. Intersections can be in different shapes in real world. As a result, there should be different methods for different intersections. For example, a “Y” shape intersection should be able to make cars from 6 directions check the next cell if it is available and make movements in correct order.
    * Reflection:

I must say this assignment is rather interesting, since it is always a pleasure to test the new tools you just learnt. The biggest challenge is not achieving every function. Instead, I must avoid using the familiar ways to implement the functions. I’ve tried some new things, but I’m not 100% satisfied. I still use too many “if”. If I’m going to do it again, I will try use more “lock”. I’m not sure if it is smart or not to use a static integer to make methods do different things for different classes. Maybe I should put the cars number in main class, so the help methods can just be decided how to work by the number of cars from South and East. Playing with “thread” can be difficult, especially when there are so many of them. Figuring out how to control the order of threads is painful, but I have successfully handled it in the end.

This assignment helps me to understand why we are having software engineering lectures. A good structure can be really helpful. The team project we’ve done also helps a lot. It gives me so much confidence. I don’t afraid of errors anymore after I’ve made so many errors during team project. Just take a breath, and there is always a way to fix it. The most wonderful thing that I have found during coding, however, is the beauty of abstract class. Once you have a plan for it, abstract class can save a lot of time and helps a lot to understand similar classes.