Please fill this in with your expected mark and a summary of what you did for each category. It doesn’t need a lot of detail, but does need enough information to determine what was done and not done.

Please fill in the **Mark** column with an idea of what mark you expect, so we can easily check.

Please both submit this to moodle and provide it to your marker at the demo (by posting into the chat for the one-to-one teams meeting where you share your screen to do the demo).

Please delete all text with yellow background and replace my example comments by your own.

Name: Ning ZHU. Student id: 20215673

C++ Programming, Coursework Part 2, mark sheet

Please include one or more screenshots here, to impress us

|  |  |  |
| --- | --- | --- |
| Mark | Requirement | **What you did and where it is in the code (file/lines)**  **You should fill in this boxes in this column** |
| 3 | **1. Handling program states**  1=startup/pause/running  2=5 states, etc  3=state pattern | Use the state pattern, and includes 7 states:  Initial state, guidance state, game state, win state, lose state, upload state and history state. |
| 3 | **2. Save and load some non-trivial data**  1=one value save AND load  2=more complex data  3=save/load state | How to save: In the game state, press “ESC” to show the pause menu, and click the load button.  How to load: In the initial state, click the load button.  Everything of the game will be perfectly loaded as the moment when they are saved, except the appearance of the bullet object, cause i use the rand() to set the color of the bullet in the virtDraw() function of BullletObject class. |
| 1 | **3. Use appropriate sub-classing with automated objects**  Intermediate class with functionality.  At least three different subclasses. | I create a TemplateObject class for all of my object classes: UserObject, BulletObject and BaseEnemyObject class directly inherits from it.  It provides the basic thing like holding an image container, the method used to get the pixel map of the image.... |
| 2 | **4. Creating new displayable objects during operation**  1 = Created or made visible  2 = destroy properly as well | Pretty easy, the bullets are the most obvious one.  When playing the game, press the “S” to shoot(create the bullet), the bullet will be destroyed when it hits the tile, out of the screen or hits the enemy object. |
| 3-4 | **5. Complex intelligence on an automated moving object**  1-2 marks then justify to the right ->  3 or 4 marks needs good documentation explanation later.  4 marks also needs video! | See the class BossEnemyObject  The target of the game is to pass the game as soon as possible, in the smaller time with higher points. In that case I set that avoiding bullets with higher priority than killing the user.  How to chase the user: use BFS to find the shortest path to the user, if there is somebody use A\*, may  be I just can get 3.  How to avoid the bullets: according to the current moving direction of object itself and the bullets object, predict whether it will be attacked by the bullets in three frame, if so, avoid it. |
| 2 | **6. Non-trivial pixel-perfect collision detection**  Improved=better than the class I gave  Complex irregular shapes is meant to be hard – to challenge the best students, e.g. check each pixel in shapes for intersection. | All collisions are handled using pixel by pixel check.  Every objects in the game has its own changing pixel mapping, depends on the state of that object.  See the file MyCollisionDetector.h to check my advanced collision detection. There are two methods, one is to check the collision, another gives the best safe distance to avoid the collision |
| 1 | **7. Implement a scrolling background by manipulating the way that the background image is drawn** | See win and lose state, also the game state and the guidance state, I will not explain it here. |
| 1 | **8. Have an animated or changing background by utilising multiple images** | I have six drawing surfaces in the initial state, generally to say, these additional surfaces are the pixel containers for images.  Background surface will copy pixels from these additional surfaces to simulate the animation. |
| 2 | **9. Correctly implement scrolling and zooming of the foreground, allowing the user to scroll around using keys and/or mouse**  1 mark – one of them  2 marks – both using filterpoints | I create my own filter point subclass which combines both class FilterPointsScaling and FilterPointsTranslation, and applies it in the guidance state.  Use mouse wheel to zooming, and use keys to scrolling. |
| 2 | **10. Animate moving objects**  1 = 1 object, some understanding  2= all objects, smooth and impressive | See the game state.  Every object has more than 4 images, the animation is smooth and impressive |
| 1 | **11. Image rotation/manipulation using the ImagePixelMapping object**  Your own new subclass! | Create my own subclass, which support partial x and y shift, you can see it in the history state,  but I choose the image casually. |
| 2 | **12. Interesting and impressive tile manager usage**  5+ tile types  Appropriate/different pictures  One or more images  2 marks: impressive and changes | More than 5 tiles. Each level has different tiles map. When all enemies in the current level are dead, the portal tile will appear, and only when user stands on the portal tile, the prompt tile will appear. |
| 1 | **13. Allow user to enter text which appears on the graphical display** | See the upload state.  Accepts 26 letters and numbers.  Press backspace to delete the last char and enter to submit the form. |
| 1 | **14. Show your understanding of templates, operator overloading or smart pointers**  Used in appropriate way**.** | Operator overloading:  “<<” see all enemy object class.  “<” see GameRecord.h. |
| 1 | **15. Additional complexity, pre-agreed in advance with Jason (max 1 mark).**  In general this is for advanced things which don’t fit other criteria. | I have not connected with Jason, actually Jason did not reply me on teams :(  But i think my work does show I am the best student this year in this module.  I simulate a 2D physical engine on my UserObject, I think if you have played the game you will find it perfectly implemented.  Rather than use one object container, I use three object containers to save user, enemy and bullet object, which provides higher efficiency to handle the event.  Game info status bar, many useful util classes... |
| 3 | **16. Impact/impression/WOW factor!**  2 or 3 marks needs explanation in document.  3 marks needs a short video too. | See the screen shots and the video. |
|  |  |  |
| Never | Your program crashes on exit or has a clear memory leak. (Lose 10% of your mark.) | |
| Never | Your program crashes at least once during its operation. (Lose 20% of your mark.) | |
| Never | Your program crashes multiple times. (Lose 30% of your mark.) | |
| Never | Your program crashes frequently. (Lose 40% of your mark.) | |
| Never | Your program has some odd/unexpected behaviour/errors. (Lose 10% of your mark.) | |
| Never | Your program has a lot of unexpected behaviour/errors. (Lose 20% of your mark.) | |
| Never | Your program crashes on exit or has a clear memory leak. (Lose 10% of your mark.) | |

# Further documentation section:

## Complex intelligence on an automated moving object

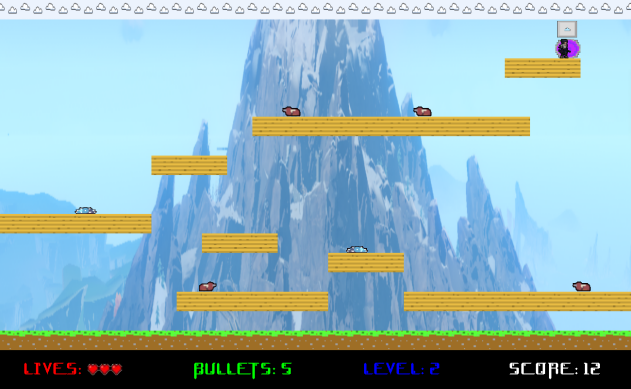
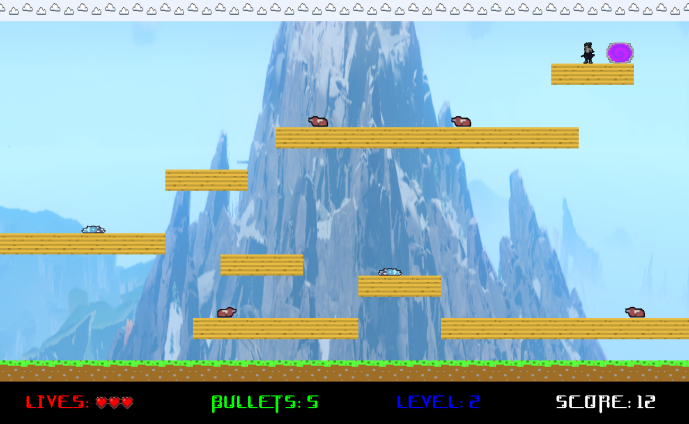
See my descriptions above and my video, or play the game to see whether you can pass the game(it is not easy kill the final boss!!!)

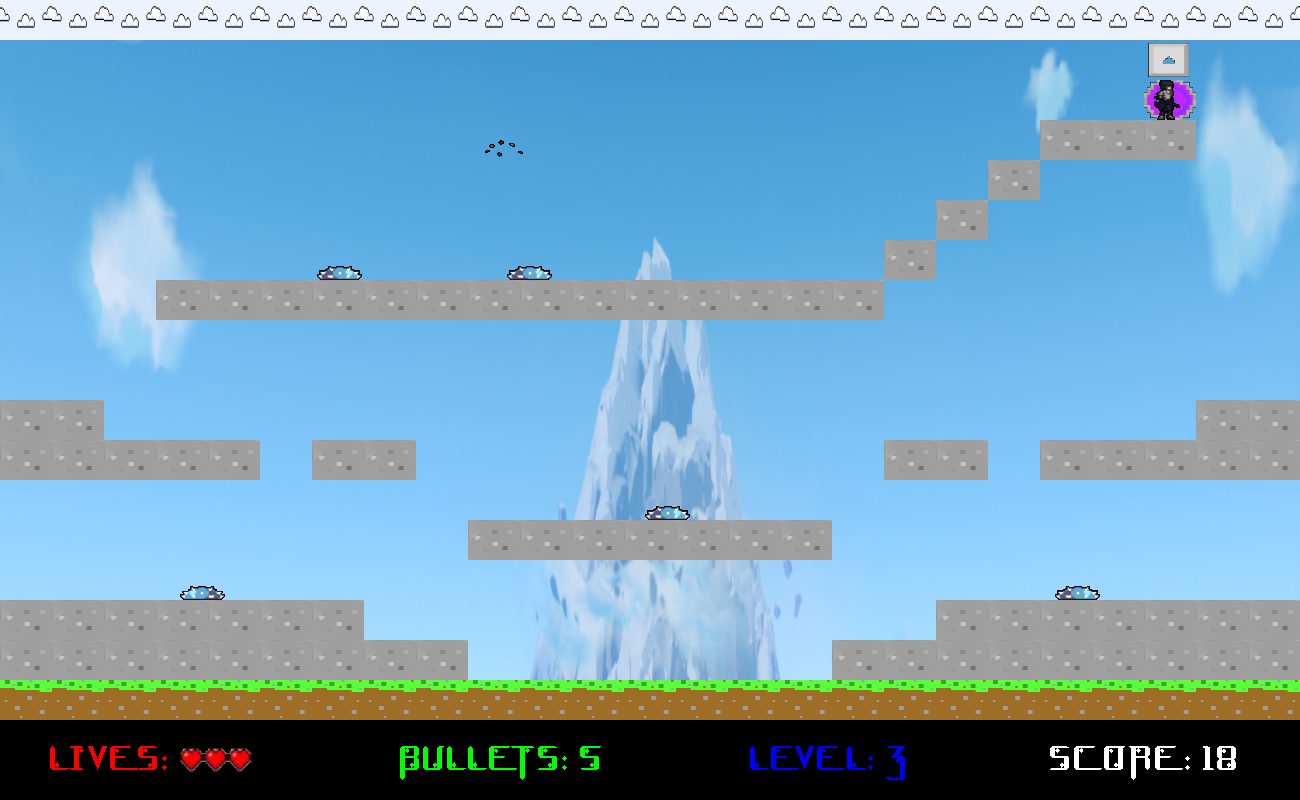
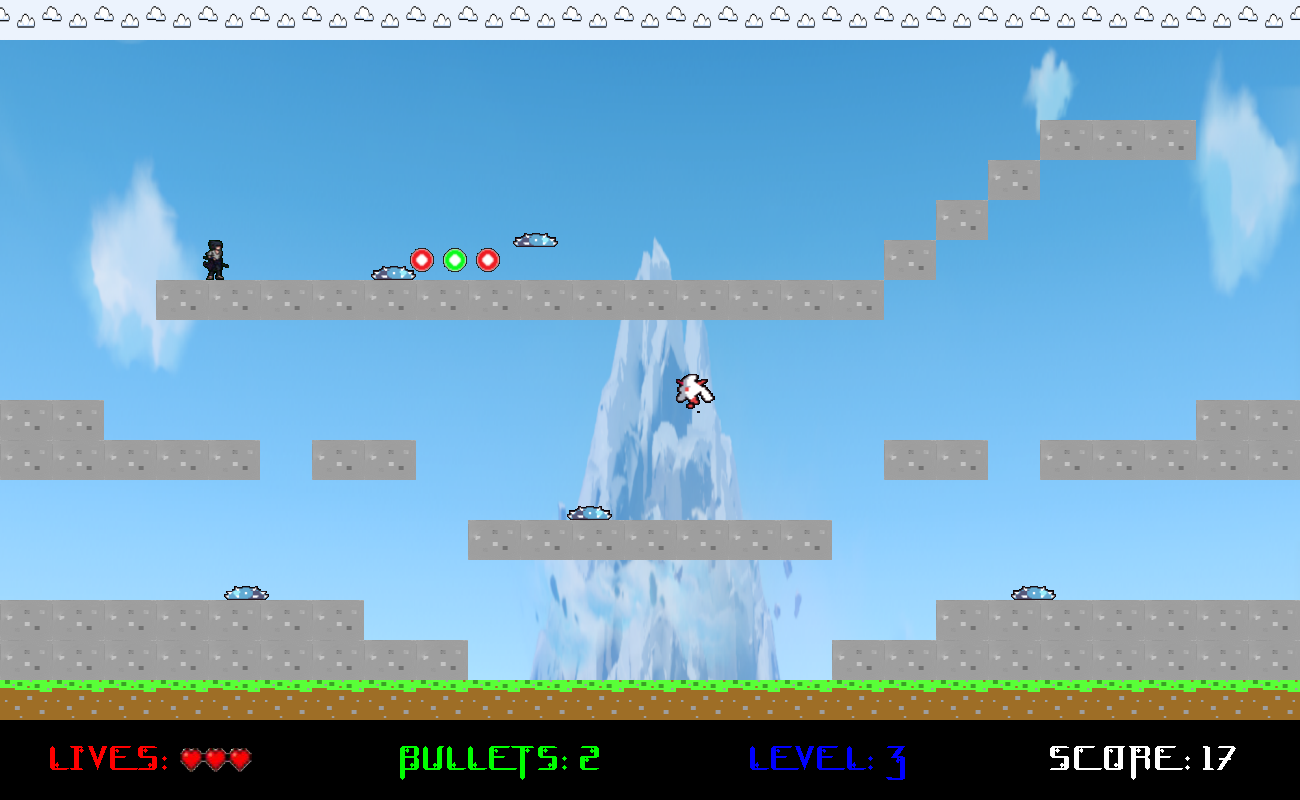
Video(***AI object.mp4***) will show how difficult to beat the final boss.

## Impact/impression/wow factor:

1. Screen shots for my three levels:





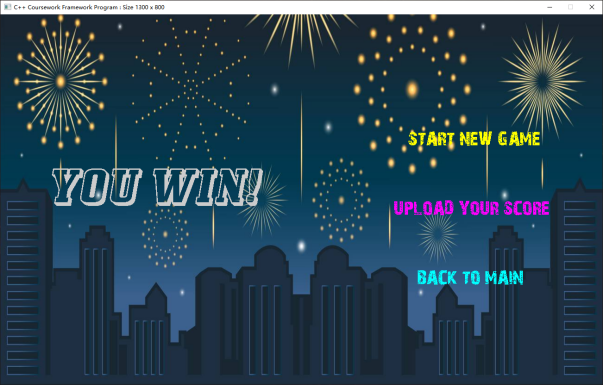
Level3:

Initial surface animation:

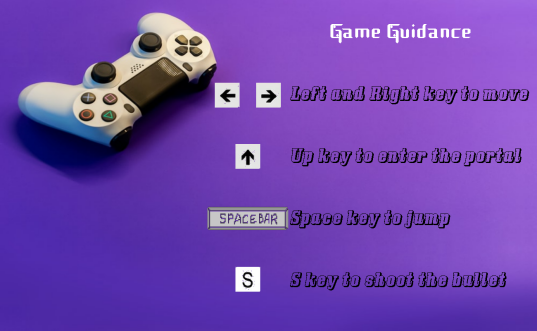




Win and lose state:



Guidance, upload and history state:



See video ***game.mp4.***