Username:	sxw03u
Full Name:	Sheng Wang
Student ID:	4207954

Each section of this document can be extremely brief! Please just add short bullet points and possibly screenshots, not long explanations.

1. Overview

Give a very brief summary (in bullet points) of what your program does. e.g. is it a fast game, a puzzle game, an editor of some kind, a display tool, etc. Basically, you are telling us what it SHOULD do, so that we can judge how well it does what it should.

Tetris is a tile-matching puzzle video game originally designed and programmed by Alexey Pajitnov in theSoviet Union. It was released on June 6, 1984, while he was working for the Dorodnicyn Computing Centre of the Academy of Science of the USSR in Moscow. He derived its name from the Greek numerical prefix tetra- (all of the game's pieces contain four segments) and tennis, Pajitnov's favourite sport.

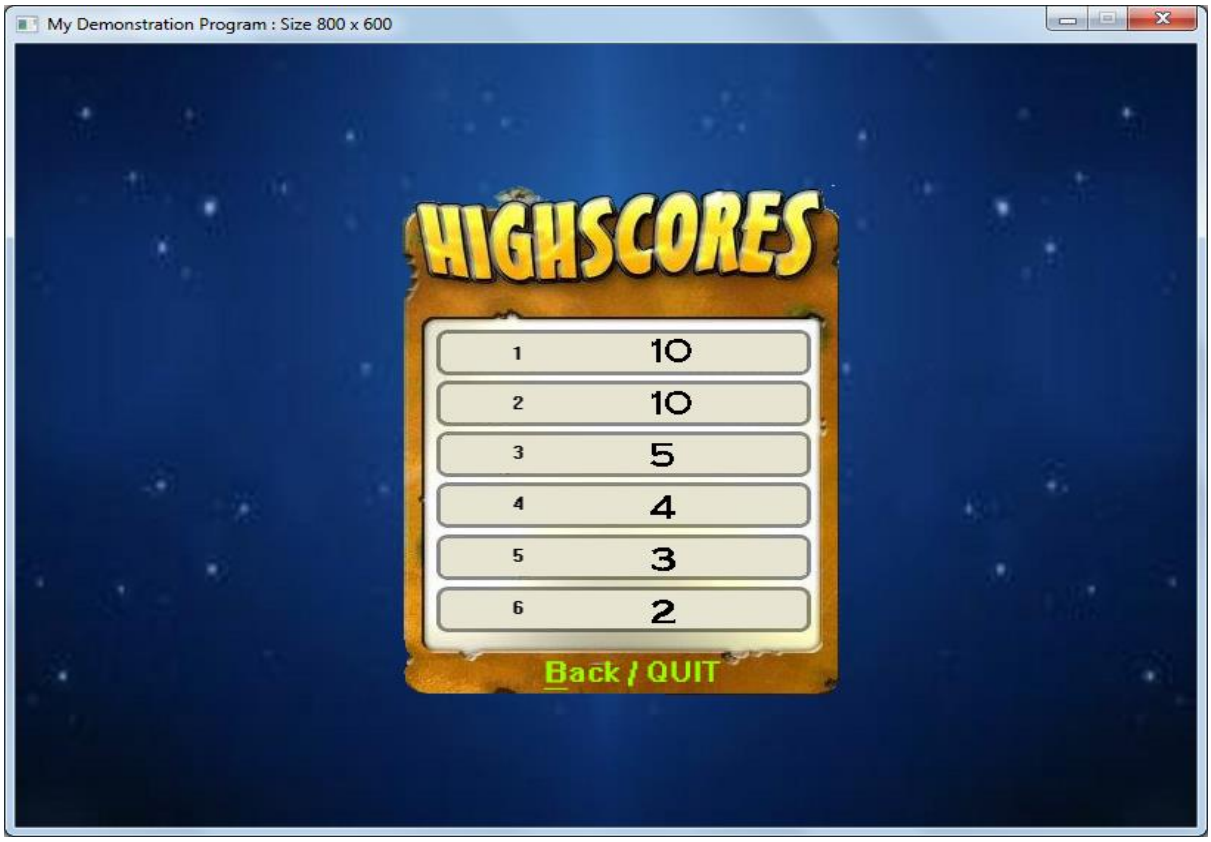
Tetriminos are game pieces shaped like tetrominoes, geometric shapes composed of four square blocks each. A random sequence of Tetriminos falls down the playing field (a rectangular vertical shaft, called the "well" or "matrix"). The objective of the game is to manipulate these Tetriminos, by moving each one sideways and rotating it by 90 degree units, with the aim of creating a horizontal line of ten blocks without gaps. When such a line is created, it disappears, and any block above the deleted line will fall. When a certain number of lines are cleared, the game enters a new level. As the game progresses, each level causes the Tetriminos to fall faster, and the game ends when the stack of Tetriminos reaches the top of the playing field and no new Tetriminos are able to enter.

2. Main Screenshot(s)

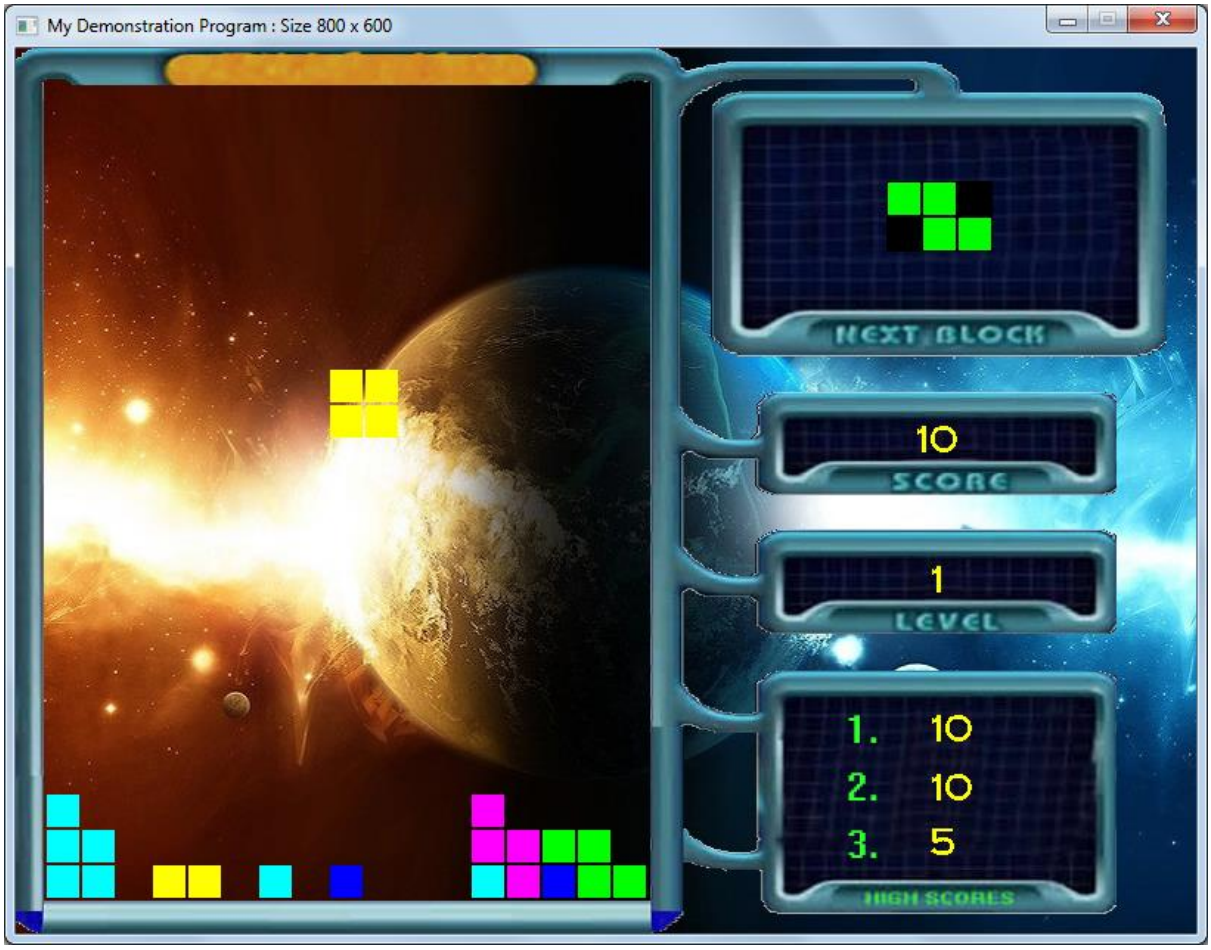
Please include a screenshot of the main game screen, choosing something which illustrates the game in progress. You can include multiple screenshots to illustrate the game if you wish. A couple of other screenshots will also be included later.

1. Game index

2. High Score Board



3. Main interface



4. Paused

5. Game Over

When user finishes the game without creating a new record, it will show this page

6. New Record

If user finishes the game with creating a new record, program will turn to high score board page automatically.

3. Usage

Tell us briefly how to use your program. Do we just run it and it all works or do we need to click on anything on the screen, or press any key to make it do things? i.e. you are telling us how you would like it to be tested.

All instructions are on the game index page.

SPACE --- start game or pause game during the game

ESC --- exit, any time during the game

Direction Arrow --- control movement and rotate

H --- In the game index page, show high score board

B --- in high score board page, back to game index page OR exit when it is in the situation of user creating a new record

4. Known problems

Please mention any problems that you know about with your program. If you don't mention them then we will assume that you do not know about them, and/or that your testing was insufficient.

1. When computer is slow, then the program will be slow, so the control of direction and rotation will be inflexible.
2. Because this program is limited to only use standard library, I can't think of a good seed for rand () function (I used to use system time as seed), so every time the program is executed, blocks are the same.

5. Files

Please provide a list of the files which you added or modified in the following table, along with a brief statement of what they are used for:

Files which I added/are mine:

File name(s)	**Purpose**
MyProjectMain.h	Main program sub-class
MyProjectMain.cpp	Main program sub-class
MyProjectTileManager.h	Main program tile manager
MyProjectTileManager.cpp	Main program tile manager
CombinedBlock.h	Main moving object
CombinedBlock.cpp	Main moving object
Block.h	Component of main moving object

Block.cpp	Component of main moving object

If you had to change any of the base classes, provide details of the changes that you made and why.

Base class files which were modified, and why:

File name(s)	Changes and reasons
BaseEngine.h	
BaseEngine.cpp	
DisplayableObject.h	
DisplayableObject.cpp	
FontManager.h	
FontManager.cpp	
JPGImage.h	
JPGImage.cpp	
TileManager.h	
TileManager.cpp	
MovementPosition.h	
Templates.h	

6. Specific requirements

Consider each of the requirements one at a time and give a brief (bullet-pointed) summary of how you have met the requirement, and why your implementation of it is so good.

If you have failed to implement a feature, or there are known problems with your implementation, then you should include the details here. That way we will know that your testing was not to blame. e.g. if you know something goes wrong under certain circumstances then please say so. Most professional applications have known bugs or problems so this is not a disaster, but you will lose marks for bugs and problems. If you do not mention it here, we have to assume that you thought that the features worked correctly so you will not only lose marks for features which don't work but will also lose marks for not testing it correctly.

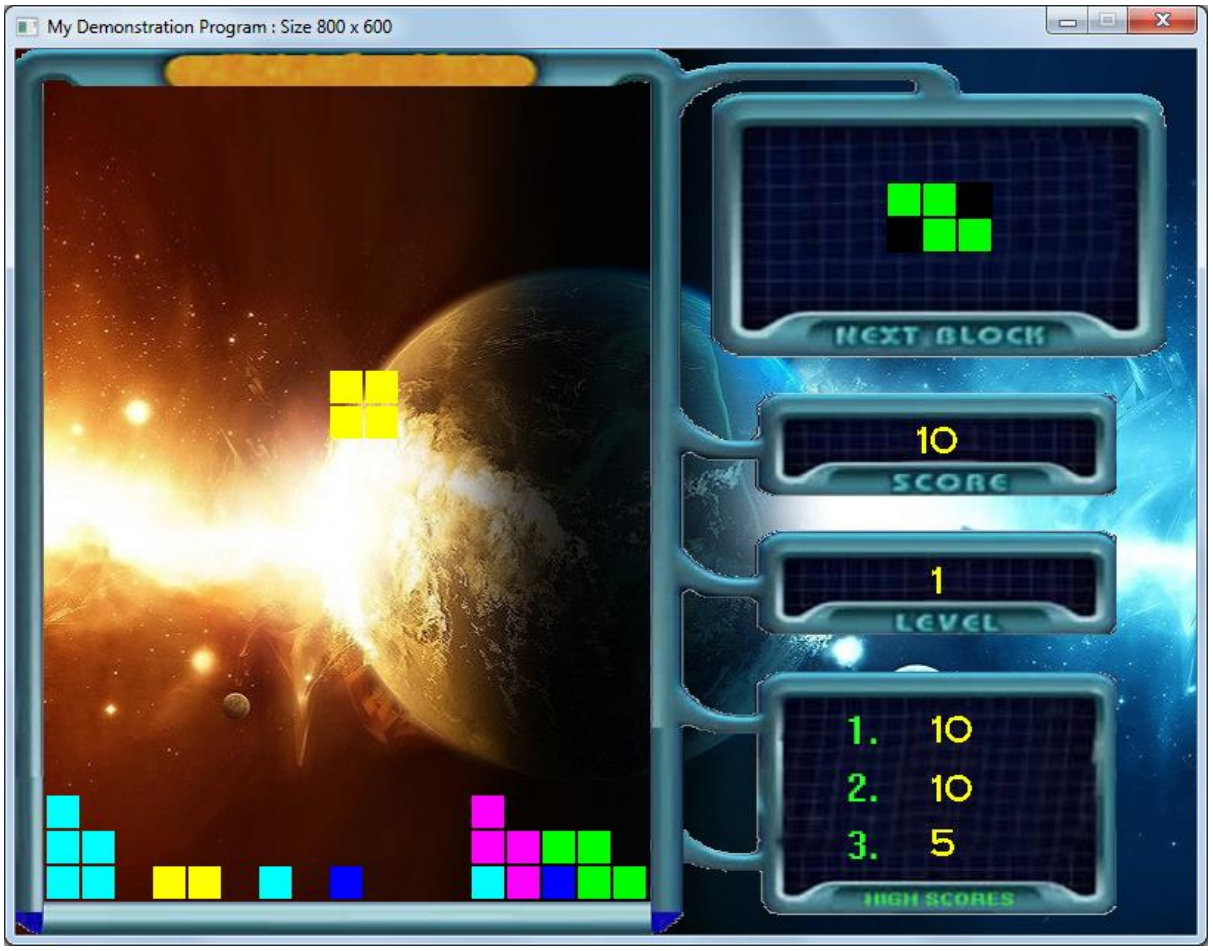
For each requirement that you did implement, you should mention how it has been implemented.

This documentation should be in a format which will allow the person marking the coursework to easily identify the various functions (and data members, where applicable) that you added or modified in order to achieve the functionality. **Please be clear and concise rather than wordy. It will be quicker for you to write and quicker for us to read.**

Note: There is no need to explicitly attempt to use specific C++ features if they are unnecessary. E.g. you should not try to alter your program just so that you can put some exception handling in, to prove that you can do so. However, if using a C++ feature is the most appropriate way to handle a problem, you should consider highlighting that you used it (e.g. casting or exception handling) in the relevant sections for the features, or for the complexity or efficiency marking criteria. Your

knowledge of these features will be tested in the exam, so I see no need for you to also prove this in the coursework.

Please include a screenshot of the main game screen, choosing something which illustrates the game in progress. You can include multiple screenshots to illustrate the game if you wish. A couple of other screenshots will also be included later.



Draw an appropriate background.

What I did:

e.g. game background , game window

Files/methods changed to do this:

MyProjectMain -> setBackgroundBuffer

Screenshot:

See before

Have moving objects.

What I did:

Moving blocks

Files/methods changed to do this:

CombinedBlock

Block

Screenshot:

See before

Have interaction between the moving objects and the environment.

What I did:

Linking a line can add score and level

Files/methods changed to do this:

CombinedBlock -> DoUpdate

Provide player interaction.

What I did:

User can control blocks

Files/methods changed to do this:

MyProjectMain -> KeyDown

Provide AI-controlled objects.

What I did:

When blocks touch the bottom of game window or other tiles which have been drawn on background already, program will draw tiles as the same shape and colour as blocks on background

When tiles link a line , program will undrawn this line of tiles

Files/methods changed to do this:

MyProjectMain

CombinedBlock

Load information from files.

What I did:

High score board will load data from file and finally will write data to the file

Files/methods changed to do this:

MyProjectMain -> setTileData

Save information to files.

What I did:

High score board will load data from file and finally will write data to the file

Files/methods changed to do this:

See before

Display status information on the screen.

What I did:

Paused state , game over state , new record state

Files/methods changed to do this:

MyProjectMain -> setBackgroundBuffer

Screenshot:


My Demonstration Program : Size 800 x 600
Game Over
Press ESC Key To Exit ...

Support different states.

What I did:

This program have 6 states : stateInit , stateMain , statePaused , highScore , gameOver, stop.

Files/methods changed to do this:

MyProjectMain->GameAction

Screenshot:

See before

7. Marking criteria

Since you know the marking criteria, you may want to make some comments about some of them, to point out something which you would like us to take into account in the marking. E.g. is there a requirement which you think was particularly well implemented?

Complexity (VERY IMPORTANT)

What do you think was particularly complex about your program?

1. You have to use a group of rectangles (objects) which is never seen in demo, and let main object pass different values to them many times.

2. You have to implement a way to let object interact with tiles on background, so that you can only create one block object instead of creating infinite blocks.
3. Program will execute codes quickly and repeatedly, however, for Tetris, you have to slow down the execution of relevant codes like rotating blocks, otherwise, it will rotate, maybe 360 degree, in just one press and look like program working wrong.

Compilation and Reliability

Do you know of any issues with compiling your program?

No

Do you know of any issues with running your program? Does it crash or hang under certain circumstances as far as you know?

No

Clarity of code, or this documentation

Do you want to explain anything about your code style of structure? Maybe you used a consistent naming style which we may not have seen before or maybe you have reasons for what seems to be an odd style? If so then here is your chance to comment.

No

Efficiency

What part(s) of your program do you think are especially efficient or inefficient?

Because blocks have many shapes and can switch, change shapes, it may be inefficient.

In addition, I'm not sure whether using draw function provided by framework is more efficient than using pictures, if yes, I think using a little too much picture may be inefficient.

Appearance

Is there anything especially good or bad about the appearance of your program, that you want to draw our attention to? Please feel free to add screenshots to support your arguments if you wish.

As you can see before, the appearance is pretty cool. No need to explain more.

8. Additional information

In this section you should make any comments which are useful but did not fit the previous sections.

E.g. you may wish to make a comment about additional features which you added which you would like to have considered in marking, or a justification of why your implementation of something was particularly good.

Alternatively, you may wish to explain any unusual behaviour or problems that your program has, or anything else not mentioned in the previous sections.

Please do NOT mention anything which was already mentioned elsewhere.