A list of member function :

For Piece, most of the untrivial function:

The shape function is a pointer function that returns a pointer to a pointer. Shape function is a 3D array and the returned pointer to pointer points to it. It goes Orientation, x-position, y-position.

I put it here because it is a property of the piece. I made the function pure virtual because each piece has a shape and all the shapes are different.

I also made a pure virtual function called getType, which is useful when I use it in the game. I made the function in the piece class because type is a property of piece. (type=0 for normal pieces, type =1 or 2 or 3 for special pieces).

There’s also destructor in each piece’s inherited class.

For Tank, most of the untrivial functions:

There’s the constructor which add the @ sign to the edges of the tank.

The tank class has a private variable call m\_tank which is a 2D char array that contains the characters.

There’s also a display class called tank::display that put the m\_tank elements onto the screen.

There’s also a input function that let me put characters into the m\_tank from another class.

There’s also a access function that let functions from another class to access the elements in m\_tank.

For Game, most of the untrivial functions:

The constructor initializes some of the member variable’s value.

Some of the variables are the level, score and rows changed.

The “clearscreen” clear the screen.

The “displaystatus” access private members such as levels or scores to display the status.

There’s a genpiece function which generate a pointer to each pieces, we want the type of the piece and the pointers generated by the shape() function in piece class.

There are two displaypiece functions. The “redisplaypiece” is for to display the piece after the second time. The displaypiece is for the first time.

In the displaypiece function it takes a pointer to the shape function and takes variables posx posy and orientation. posx and posy is the position where we want to display it. The default orientation is zero and if we want to change, we add piece pointer to 16, which goes to the next orientation. Every time we make change to the screen, we make change to the m\_tank as well. We make true ‘#’ and false space.

Another function called todollar which loops through all elements in the shape (which is a private member in the Game class of current piece). The doolar function is called after the piece reaches the end (can’t move) .If there’s “#” we make it the dollar sign. Also in the dollar sign, it takes into account of the special type, if it’s a foam, we call an recursive function to fill the space. If it’s vapor, we delete the upper four and lower four pieces, if ‘$’ or ‘\*’.

Another function called canmove take orientation, posx and posy to check whether the piece can move forward or not.

Another function called canRotate takes orientation, posx, and posy to check whether the piece can keep rotating. If above functions satisfies, we let it move in the playone level function.

There’s also undisplay function when we want to move the piece downward, we undiplay it and then display it in the next position.

The playonelever do most of the work:

Display status, set variable back, display next piece, takes keyboard input, move the piece. And change status (like levels, score and such).

There’s a check if to remove function, (whether one or more rows are filled with \* or $), and there’s another function to remove them.

There’s also a recursion function to fill the space in foam case.

In crazypiece case, if crazypiece type, the swich ,keyboard input variables get swiched.

Some of stuff that doesn’t quite work:

The foam bomb occasionally doesn’t work really well, it doesn’t eliminate rows sometimes. But I called ifRemove everytime the function toDollar finishes so that weird.

Occasionally the shape of piece in Next Piece display get messed up (maybe I fixed it, not sure).

The display is weird because I can’t get printf to work, so I made a algorithm to calculate the digits which is related to the space between.

Design decisions:

The piece classes doesn’t have a lot of functions which kind of upsets me.

I made the shape in piece class a function rather than a data member because it’s easier to access(I got this idea from a stackexchange post though).

I made the remove function in the game class because it’s easier to process it sinec a lot of data members is in game which is fact it should be in the tank.

I only used tank for checking variable exist or not purpose which is kind of weird.