SKY ATTACK v1.0

Console Game Group Project

COURSE:	CO1302 PROGRAMMING FOR ENGINEERS
GROUP:	GROUP 2
YEAR:	2018/2019
DATE OF SUBMISSION:	20/11/2020

INTRODUCTION

This document is to introduce about the game "SKY ATTACK" version 1.0. The game is based on a shooting craft and it shoots enemy crafts coming towards it. This game has been developed using C++ language.

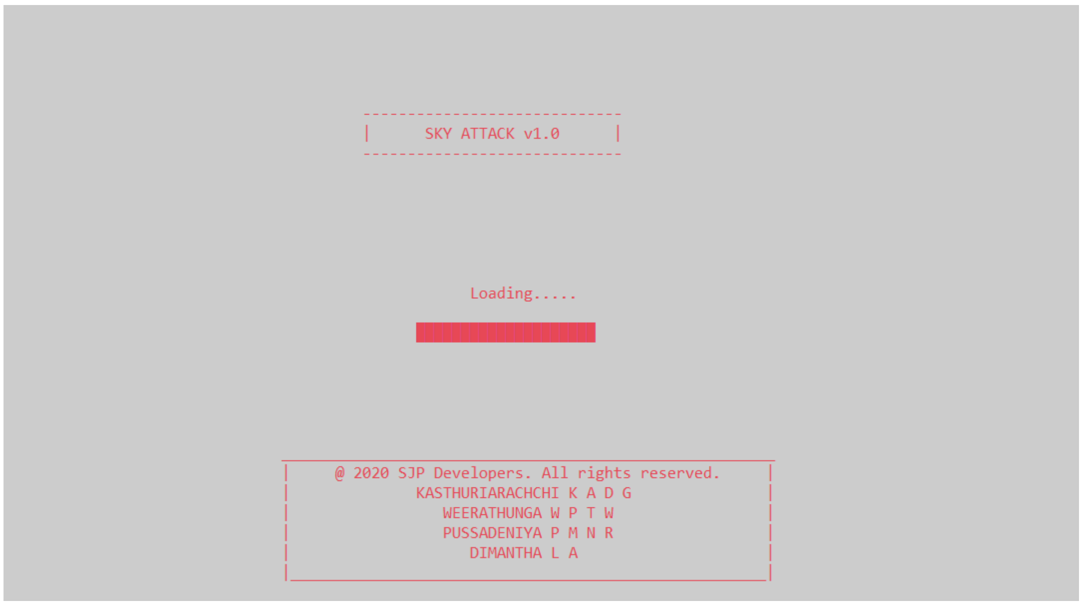
The strength of C++ when it comes to game development is the ability to exactly layout the data-structures that your software will use. When performance real-time systems (such as games) started growing, it was the most commonly supported and most developed programming language. C++ provides the ability to override important performance bottlenecks such as memory allocation. It has the ability to structure and place things exactly where they want in the memory. On top of this it's a flexible programming language that provides a decent development velocity. So, this game has been developed the basic C++ programming.

We have discussed about the functionality of the game, challenges that faced during the development of this game and at the end the improvements that can be done to this game.

FUNCTIONALITY OF THE PROGRAMME

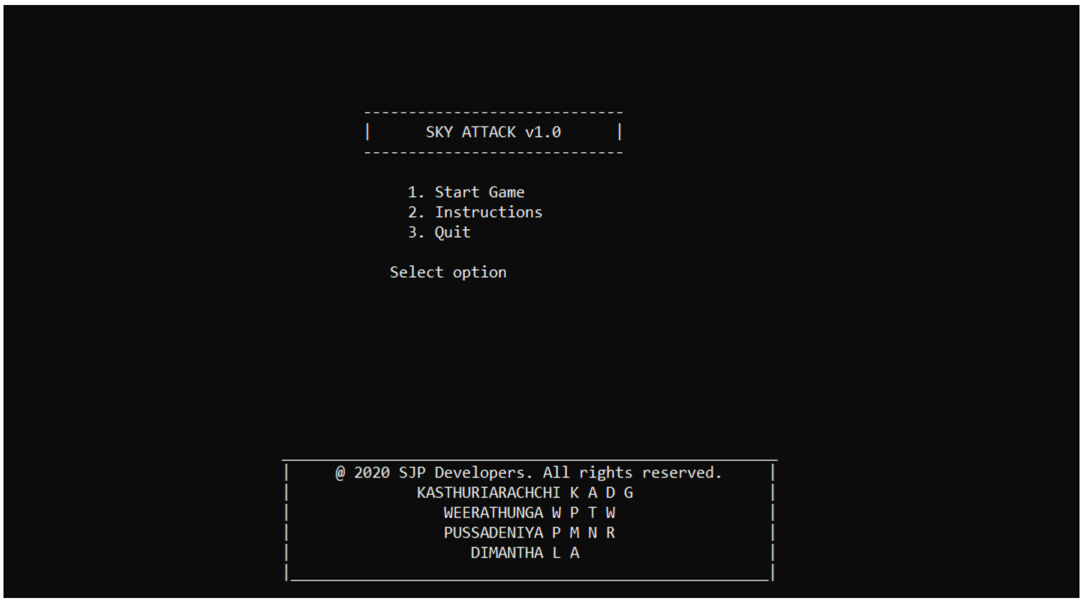
PROCEDURE

The game is starting with a loading screen that has a different color theme.



```
-----------------------------
|      SKY ATTACK v1.0      |
-----------------------------


           Loading.....
      ████████████████████


 _______________________________________________________
|     @ 2020 SJP Developers. All rights reserved.       |
|            KASTHURIARACHCHI K A D G                   |
|               WEERATHUNGA W P T W                     |
|               PUSSADENIYA P M N R                     |
|                  DIMANTHA L A                         |
|_______________________________________________________|
```

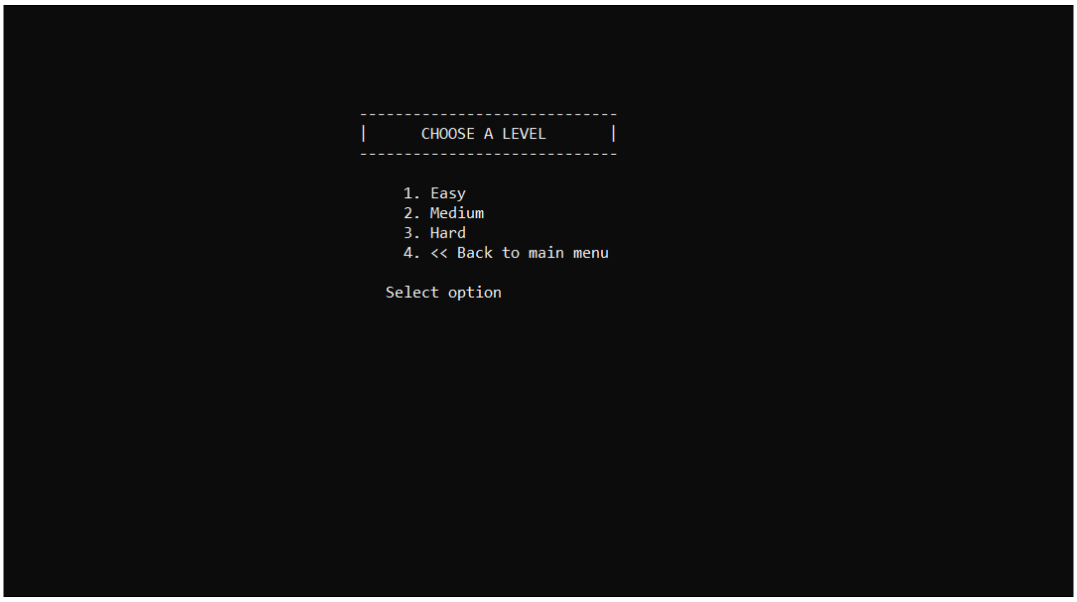
Then the main menu is displayed. Name of the game, developers' names and main options are displayed here.



```
 ----------------------------
|      SKY ATTACK v1.0      |
 ----------------------------

     1. Start Game
     2. Instructions
     3. Quit

   Select option



 _____________________________________________________
|     @ 2020 SJP Developers. All rights reserved.     |
|              KASTHURIARACHCHI K A D G               |
|                 WEERATHUNGA W P T W                 |
|                 PUSSADENIYA P M N R                 |
|                    DIMANTHA L A                     |
|_____________________________________________________|
```

There are 3 options that can choose.

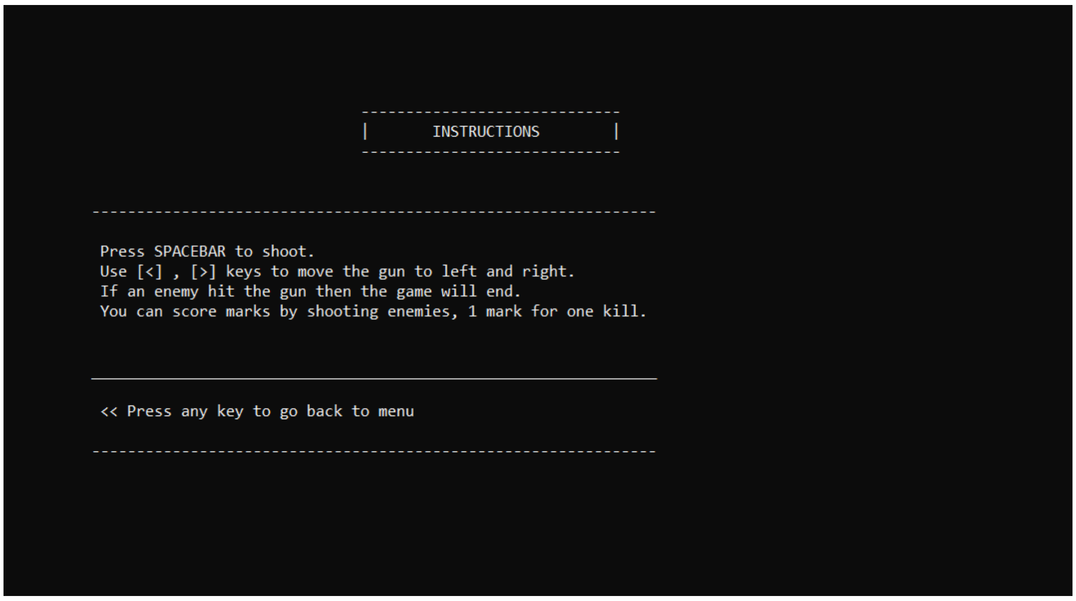
1. Start Game
   When Start game is selected, it displays level selection screen.
   There are 3 levels in the game: easy, medium, hard. The speeds of enemy crafts vary with the levels.



```
 ----------------------------
|      CHOOSE A LEVEL       |
 ----------------------------

     1. Easy
     2. Medium
     3. Hard
     4. << Back to main menu

   Select option
```

In this menu, a level player wants to play can be selected or else can go back to the main menu.

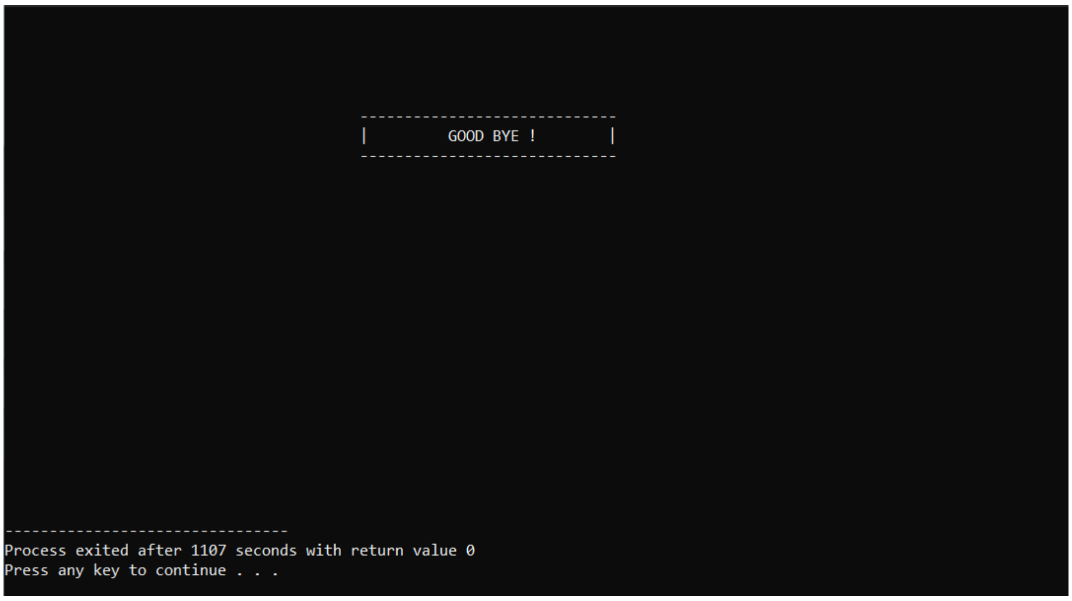
2. Instructions
   In this option, player can see the predefined instructions that needed to know before playing.



```
----------------------------
|       INSTRUCTIONS       |
----------------------------


--------------------------------------------------------------

 Press SPACEBAR to shoot.
 Use [<] , [>] keys to move the gun to left and right.
 If an enemy hit the gun then the game will end.
 You can score marks by shooting enemies, 1 mark for one kill.


______________________________________________________________

 << Press any key to go back to menu

--------------------------------------------------------------
```

   By pressing any key, player can go back to the main menu.

3. Quit
   If player choose to quit, the console will end process.



```
----------------------------
|        GOOD BYE !        |
----------------------------



--------------------------------
Process exited after 1107 seconds with return value 0
Press any key to continue . . .
```

   By pressing any key, player can exit from the console.

CONTROLS

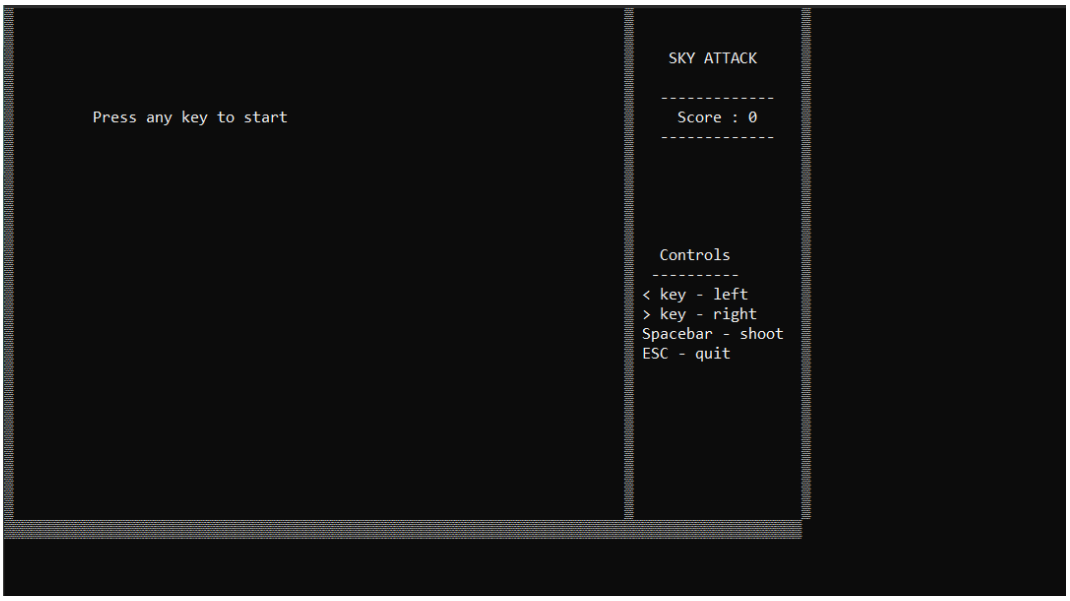
The game is designed to give user the full control of the gameplay. The controls are displayed under the instructions option and displayed in the gameplay interface.

Gameplay controls:

[space] – shooting

[<] – moving shooter to left

[>] – moving shooter to right

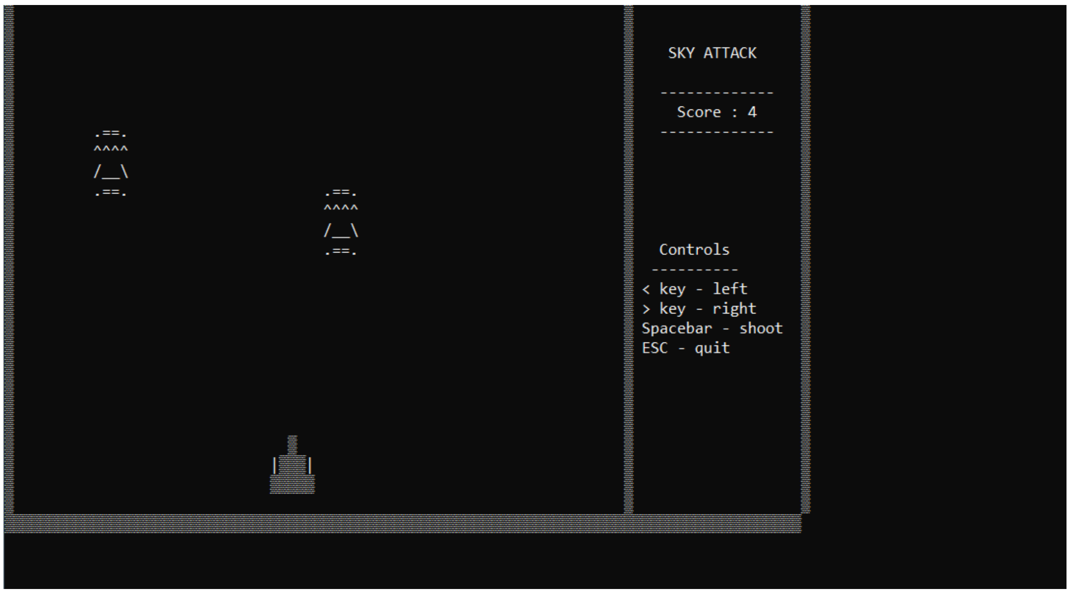


```
Press any key to start

SKY ATTACK
------------
Score : 0
------------

Controls
---------
< key - left
> key - right
Spacebar - shoot
ESC - quit
```

Also, player can quit the game in the gameplay interface by pressing [ESC]. It ends the current game and goes back to the main menu.

HOW SCORE IS CALCULATED

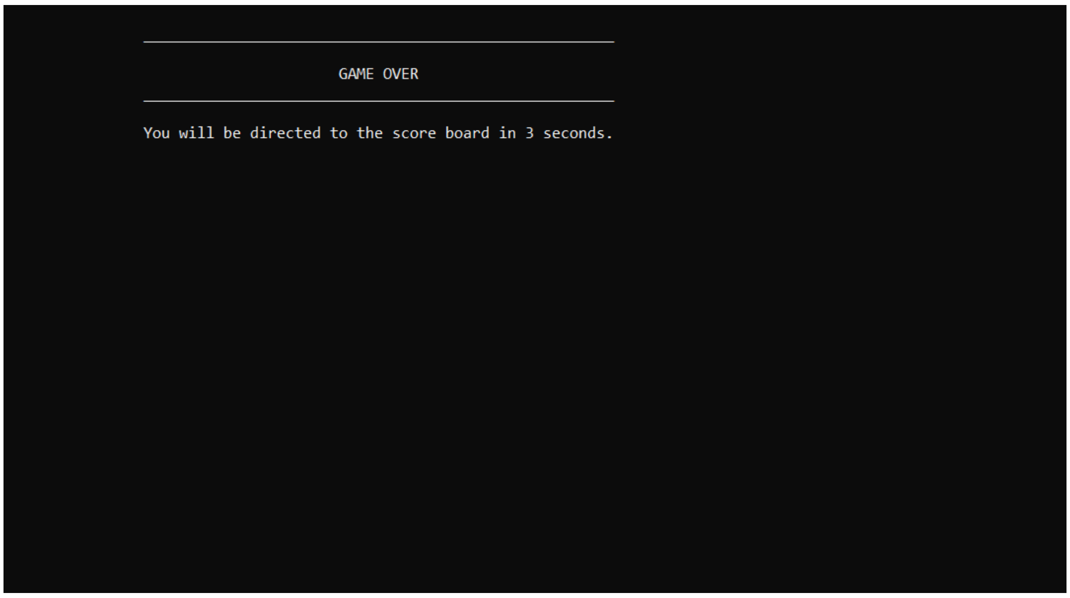
For each enemy killed, one point is added to the score. Score is displayed in the gameplay interface.



```
.==.
^^^^
/__\
.==.
```

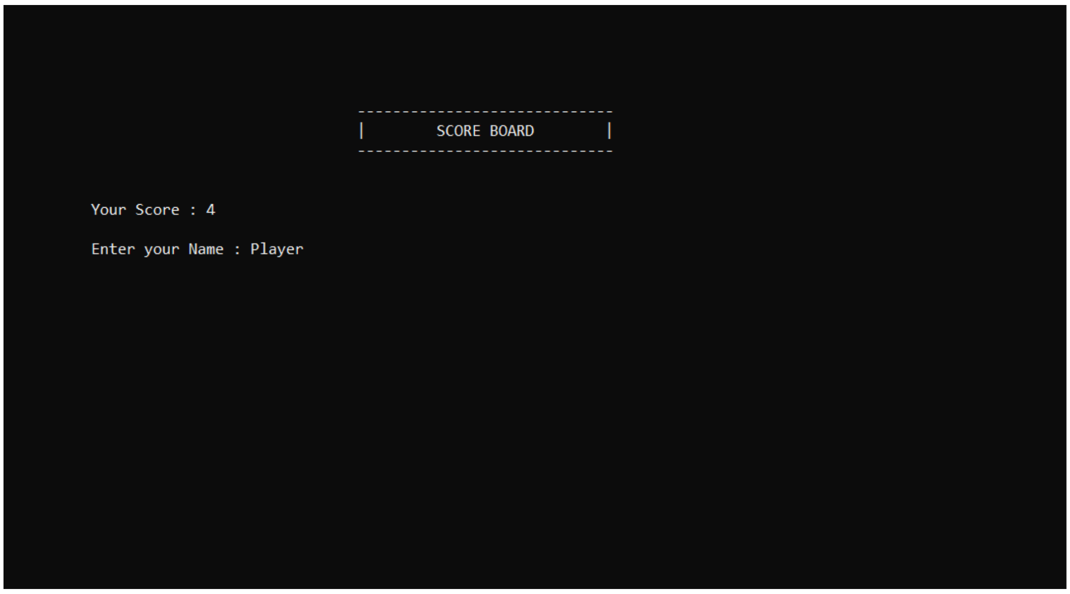
```
SKY ATTACK

-------------
 Score : 4
-------------


 Controls
 ----------
< key - left
> key - right
Spacebar - shoot
ESC - quit
```

If an enemy craft hits the shooter, the game is over. Then the Game Over screen is displayed and there is a timeout of 3 seconds.



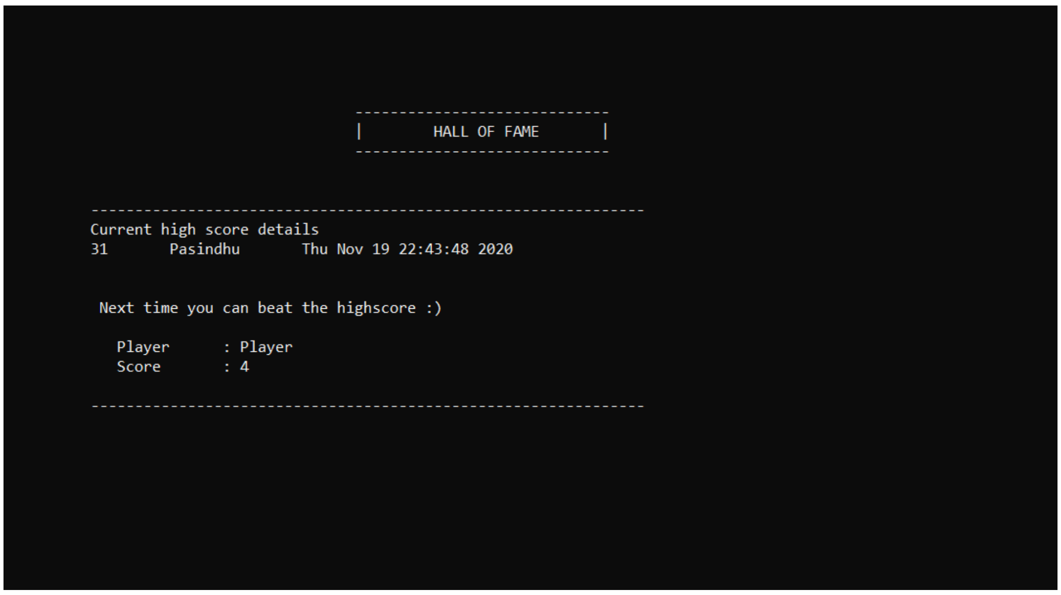
```
___________________________________
              GAME OVER
___________________________________

You will be directed to the score board in 3 seconds.
```

After timeout player is directed to the Score Board. Player has to enter name there.



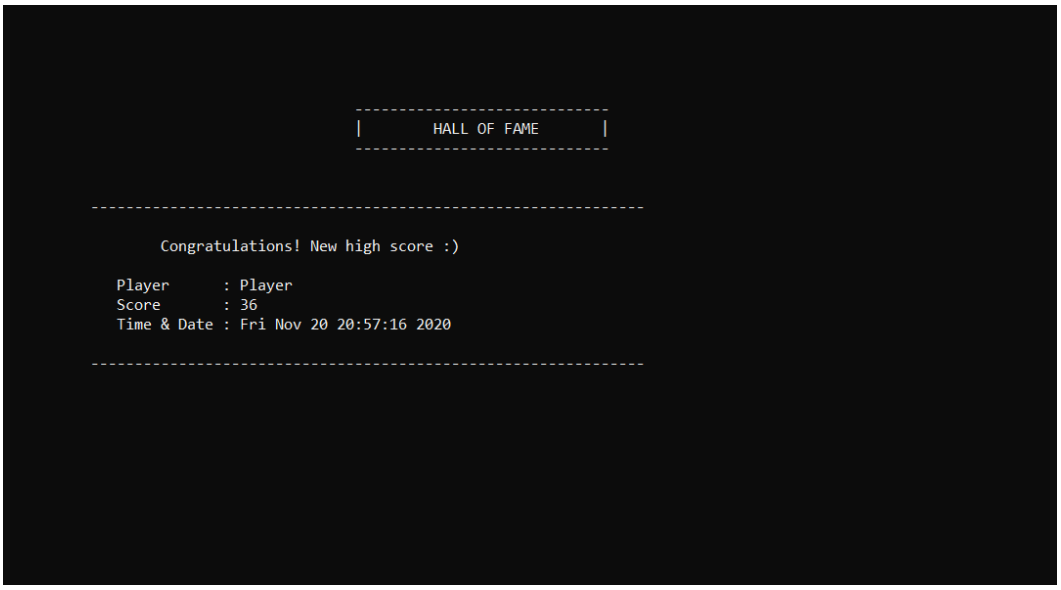
```
                    ----------------------------
                    |        SCORE BOARD       |
                    ----------------------------


   Your Score : 4

   Enter your Name : Player
```

Then the Hall of Flame screen is displayed. The high score and the player score are displayed there.



```
                    ----------------------------
                    |        HALL OF FAME      |
                    ----------------------------


   --------------------------------------------------------------
   Current high score details
   31        Pasindhu       Thu Nov 19 22:43:48 2020


    Next time you can beat the highscore :)

      Player      : Player
      Score       : 4

   --------------------------------------------------------------
```

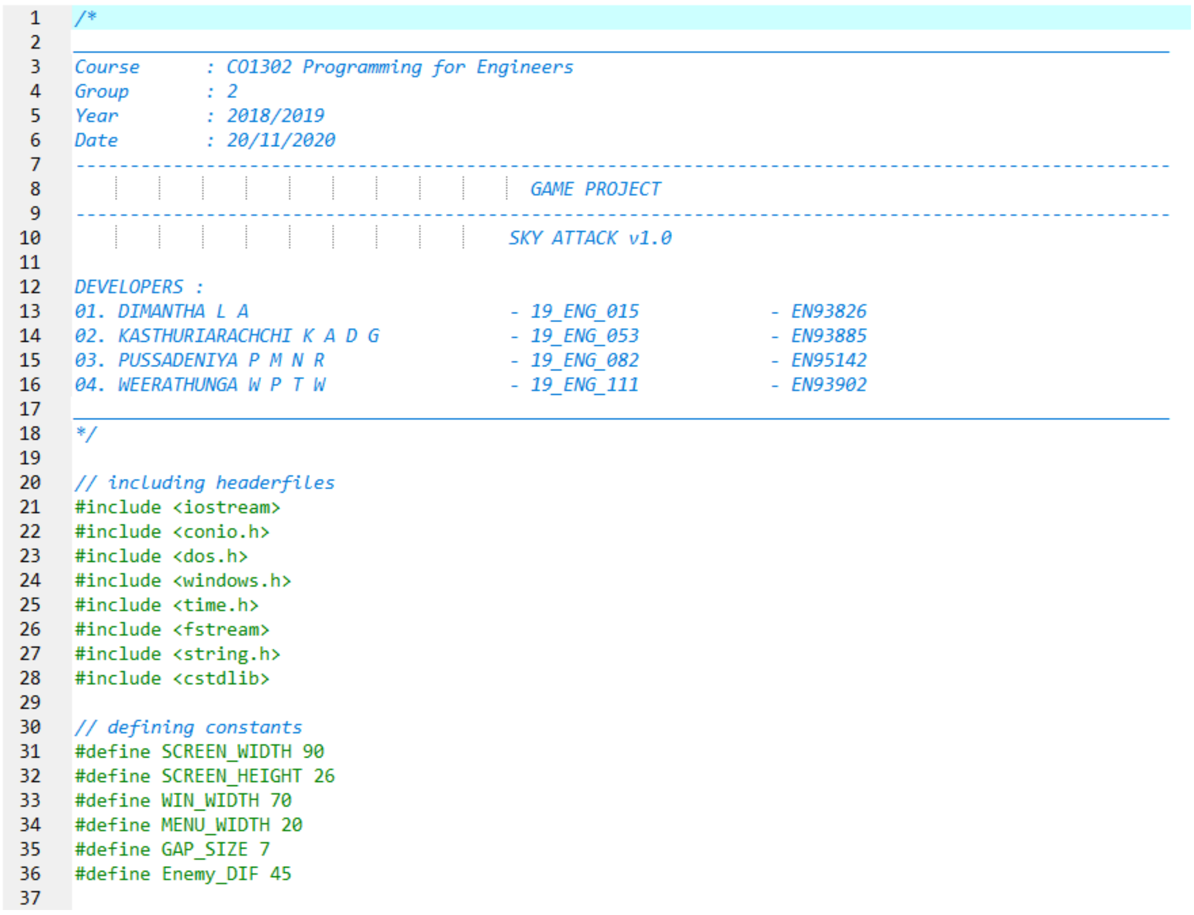
If the player beats high score, a congratulations screen is displayed.



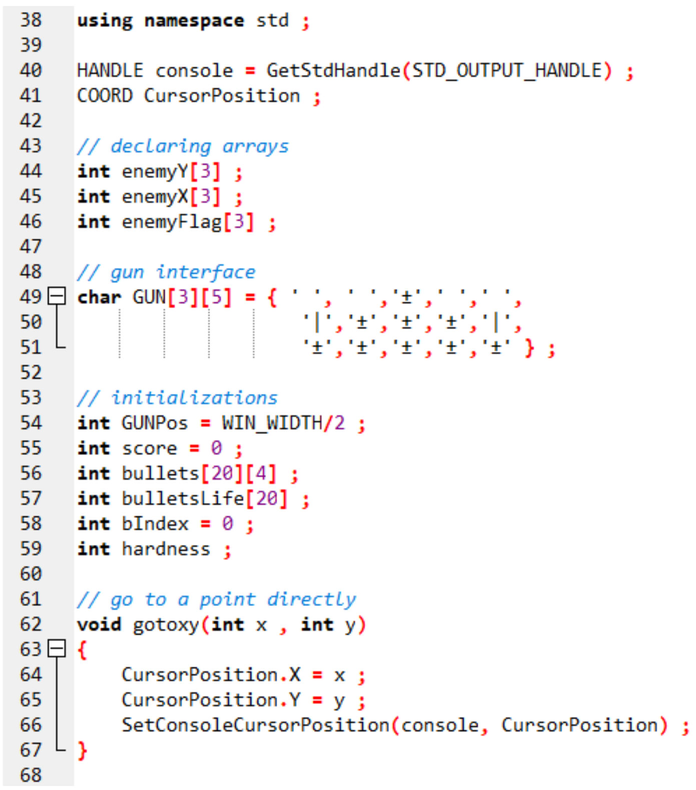
```
                                        ----------------------------
                                        |        HALL OF FAME      |
                                        ----------------------------


          --------------------------------------------------------------

                  Congratulations! New high score :)

             Player      : Player
             Score       : 36
             Time & Date : Fri Nov 20 20:57:16 2020

          --------------------------------------------------------------
```

The Hall of Flame screen is displayed for 7 seconds and then returned to the main menu.

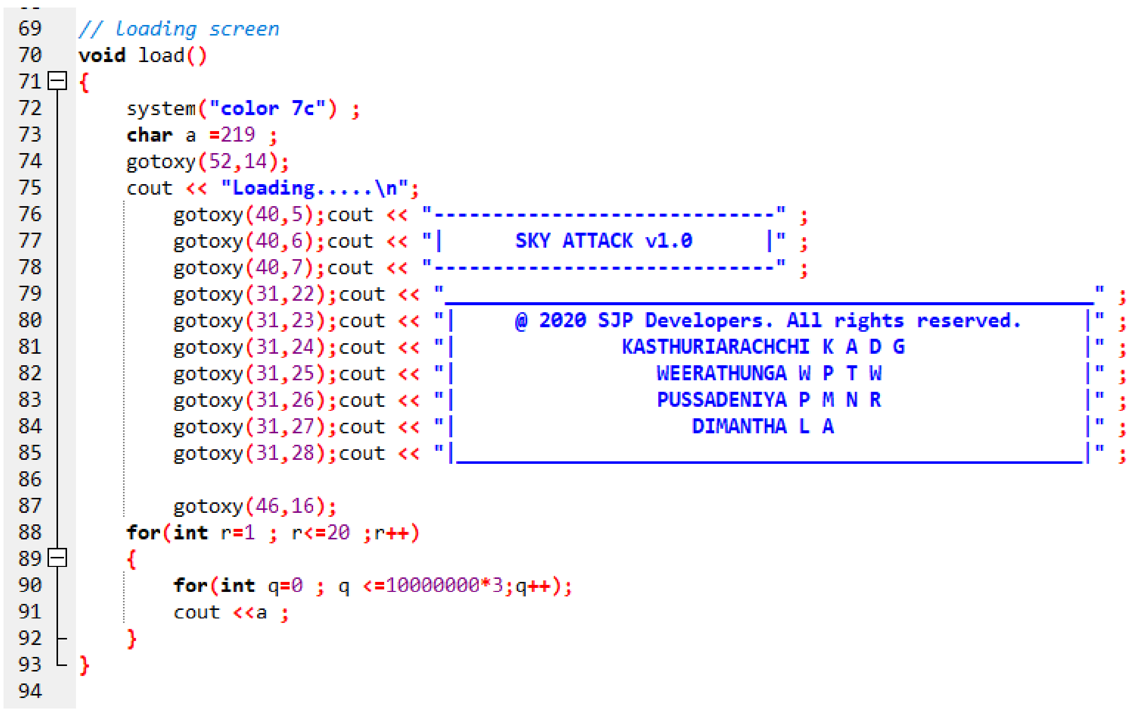
C++ CODE



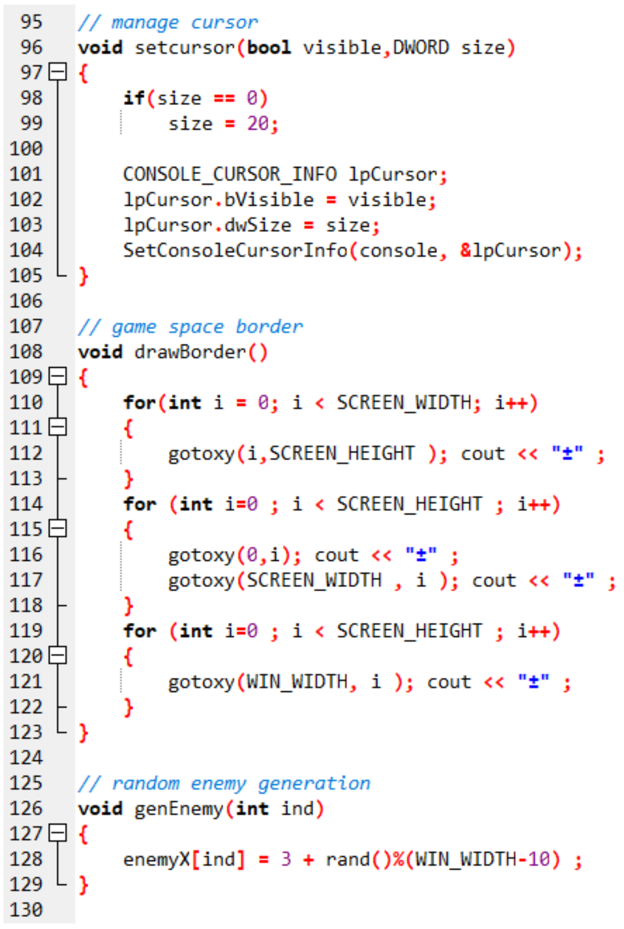
```cpp
/*
_____________________________________________________________________________________________
Course       : CO1302 Programming for Engineers
Group        : 2
Year         : 2018/2019
Date         : 20/11/2020
---------------------------------------------------------------------------------------------
                                        GAME PROJECT
---------------------------------------------------------------------------------------------
                                      SKY ATTACK v1.0

DEVELOPERS :
01. DIMANTHA L A                        - 19_ENG_015             - EN93826
02. KASTHURIARACHCHI K A D G            - 19_ENG_053             - EN93885
03. PUSSADENIYA P M N R                 - 19_ENG_082             - EN95142
04. WEERATHUNGA W P T W                 - 19_ENG_111             - EN93902
_____________________________________________________________________________________________
*/

// including headerfiles
#include <iostream>
#include <conio.h>
#include <dos.h>
#include <windows.h>
#include <time.h>
#include <fstream>
#include <string.h>
#include <cstdlib>

// defining constants
#define SCREEN_WIDTH 90
#define SCREEN_HEIGHT 26
#define WIN_WIDTH 70
#define MENU_WIDTH 20
#define GAP_SIZE 7
#define Enemy_DIF 45

```



```cpp
using namespace std ;

HANDLE console = GetStdHandle(STD_OUTPUT_HANDLE) ;
COORD CursorPosition ;

// declaring arrays
int enemyY[3] ;
int enemyX[3] ;
int enemyFlag[3] ;

// gun interface
char GUN[3][5] = { ' ', ' ','±',' ',' ',
                   '|','±','±','±','|',
                   '±','±','±','±','±' } ;

// initializations
int GUNPos = WIN_WIDTH/2 ;
int score = 0 ;
int bullets[20][4] ;
int bulletsLife[20] ;
int bIndex = 0 ;
int hardness ;

// go to a point directly
void gotoxy(int x , int y)
{
    CursorPosition.X = x ;
    CursorPosition.Y = y ;
    SetConsoleCursorPosition(console, CursorPosition) ;
}

```



```
// Loading screen
void load()
{
    system("color 7c") ;
    char a =219 ;
    gotoxy(52,14);
    cout << "Loading.....\n";
        gotoxy(40,5);cout << "-----------------------------" ;
        gotoxy(40,6);cout << "|       SKY ATTACK v1.0       |" ;
        gotoxy(40,7);cout << "-----------------------------" ;
        gotoxy(31,22);cout << "_______________________________________________________" ;
        gotoxy(31,23);cout << "|     @ 2020 SJP Developers. All rights reserved.    |" ;
        gotoxy(31,24);cout << "|              KASTHURIARACHCHI K A D G              |" ;
        gotoxy(31,25);cout << "|                 WEERATHUNGA W P T W                |" ;
        gotoxy(31,26);cout << "|                 PUSSADENIYA P M N R                |" ;
        gotoxy(31,27);cout << "|                    DIMANTHA L A                    |" ;
        gotoxy(31,28);cout << "|____________________________________________________|" ;

        gotoxy(46,16);
    for(int r=1 ; r<=20 ;r++)
    {
        for(int q=0 ; q <=10000000*3;q++);
        cout <<a ;
    }
}
```

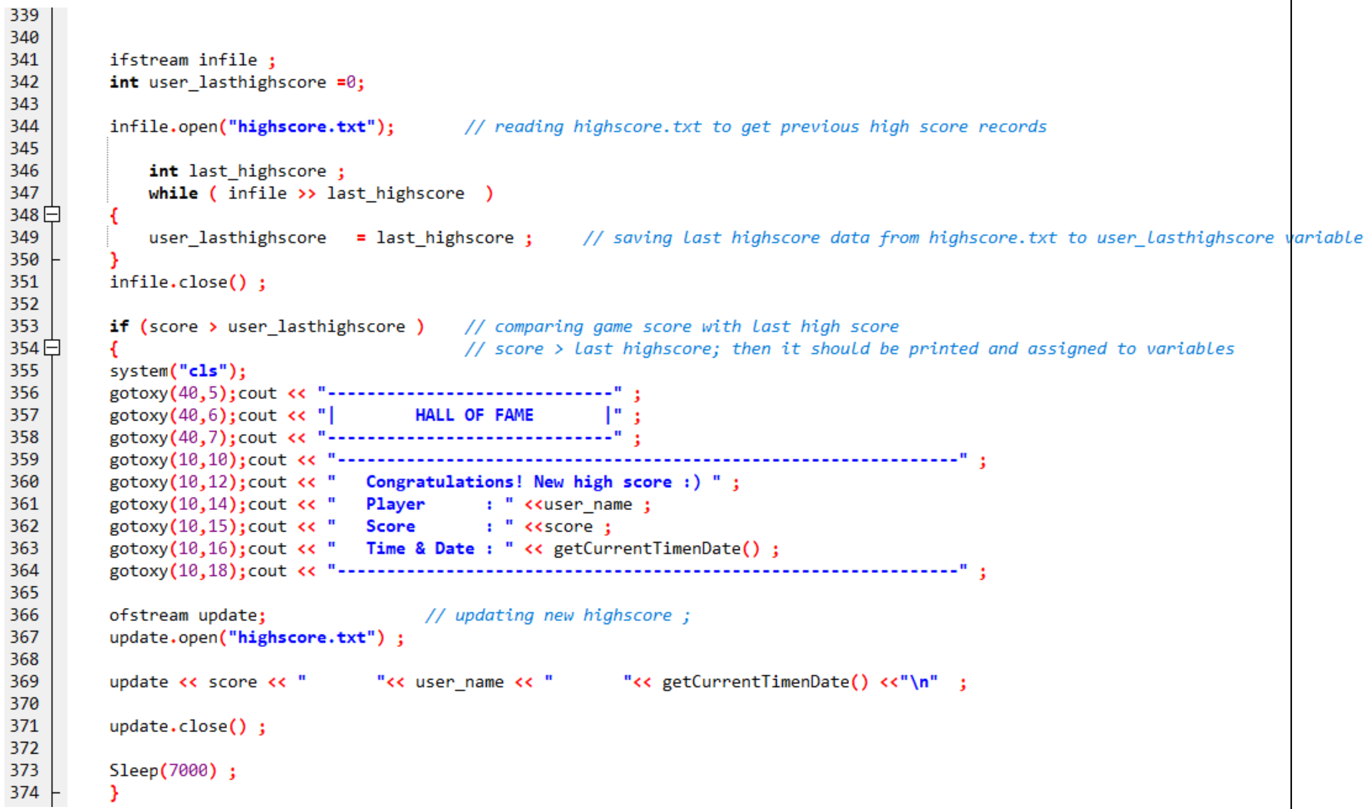


```
// manage cursor
void setcursor(bool visible,DWORD size)
{
    if(size == 0)
        size = 20;

    CONSOLE_CURSOR_INFO lpCursor;
    lpCursor.bVisible = visible;
    lpCursor.dwSize = size;
    SetConsoleCursorInfo(console, &lpCursor);
}

// game space border
void drawBorder()
{
    for(int i = 0; i < SCREEN_WIDTH; i++)
    {
        gotoxy(i,SCREEN_HEIGHT ); cout << "±" ;
    }
    for (int i=0 ; i < SCREEN_HEIGHT ; i++)
    {
        gotoxy(0,i); cout << "±" ;
        gotoxy(SCREEN_WIDTH , i ); cout << "±" ;
    }
    for (int i=0 ; i < SCREEN_HEIGHT ; i++)
    {
        gotoxy(WIN_WIDTH, i ); cout << "±" ;
    }
}

// random enemy generation
void genEnemy(int ind)
{
    enemyX[ind] = 3 + rand()%(WIN_WIDTH-10) ;
}
```

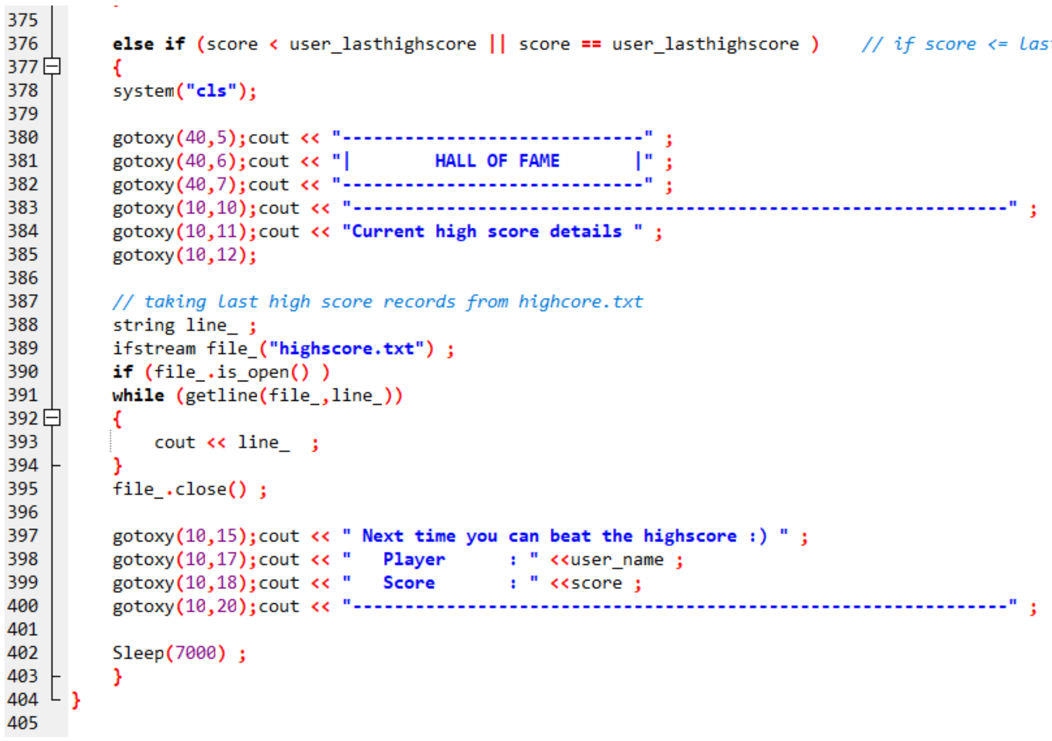
```cpp
// enemy shape
void drawEnemy(int ind)
{
    if (enemyFlag[ind] == true )
    {
        gotoxy(enemyX[ind], enemyY[ind]) ;      cout << ".==." ;
        gotoxy(enemyX[ind], enemyY[ind]+1) ;    cout << "^^^^" ;
        gotoxy(enemyX[ind], enemyY[ind]+2) ;    cout << "/__\\" ;
        gotoxy(enemyX[ind], enemyY[ind]+3) ;    cout << ".==." ;
    }
}

// erasing enemies
void eraseEnemy(int ind)
{
    if(enemyFlag[ind] == true)
    {
    gotoxy(enemyX[ind], enemyY[ind])   ; cout << "    " ;
    gotoxy(enemyX[ind], enemyY[ind]+1) ; cout << "    " ;
    gotoxy(enemyX[ind], enemyY[ind]+2) ; cout << "    " ;
    gotoxy(enemyX[ind], enemyY[ind]+3) ; cout << "    " ;
    }
}

// resetting enemies
void resetEnemy (int ind)
{
    eraseEnemy(ind);
    enemyY[ind] =4 ;
    genEnemy(ind);
}

// bullet generation
void genBullet()
{
    bullets[bIndex][0] = 22 ;
    bullets[bIndex][1] = GUNPos   ;
    bullets[bIndex][2] = 22 ;
    bullets[bIndex][3] = GUNPos +4 ;
    bIndex++ ;

    if( bIndex == 20 )
    {
        bIndex = 0 ;
    }
}

// moving bullet
void moveBullet()
{
    for(int i=0 ; i<20 ; i++)
    {
        if(bullets[i][0] > 2 )
        {
            bullets[i][0]--;
        } else {
            bullets[i][0] = 0 ;
        }

        if(bullets[i][2] > 2)
        {
            bullets[i][2]--;
        } else {
            bullets[i][2] = 0 ;
        }
    }
}
```

```cpp
// bullet shape
void drawBullets()
{
    for(int i = 0 ; i <20 ; i++)
    {
        if (bullets[i][0] >1)
        {
        gotoxy(bullets[i][1],bullets[i][0]);cout << "^" ;
        gotoxy(bullets[i][3],bullets[i][2]);cout << "^" ;
        }
    }
}

// erasing bullets
void eraseBullets()
{
    for(int i = 0; i<20; i++)
    {
        if(bullets[i][0] >= 1)
        {
        gotoxy(bullets[i][1],bullets[i][0]);cout << " ";
        gotoxy(bullets[i][3],bullets[i][2]);cout << " ";
        }
    }
}

void eraseBullet(int i)
{
        gotoxy(bullets[i][1],bullets[i][0]);cout << " ";
        gotoxy(bullets[i][3],bullets[i][2]);cout << " ";

}

// gun generation
void drawGUN()
{
    for(int i=0; i < 3; i++)
    {
        for(int j = 0 ; j<5 ; j++)
        {
            gotoxy(j+GUNPos, i+22); cout << GUN[i][j];
        }
    }
}

// erasing gun
void eraseGUN()
{
    for(int i=0; i<3; i++)
    {
        for(int j = 0; j<5; j++)
        {
            gotoxy(j+GUNPos, i+22); cout<<" ";
        }
    }
}

// checking gun and enemy collision
int collision()
{
    if (enemyY[0]+4 >= 23)
    {
        if(enemyX[0] + 4 - GUNPos >= 0 && enemyX[0] + 4 - GUNPos <8)
        {
            return 1 ;
        }
    }
    return 0 ;
}
```

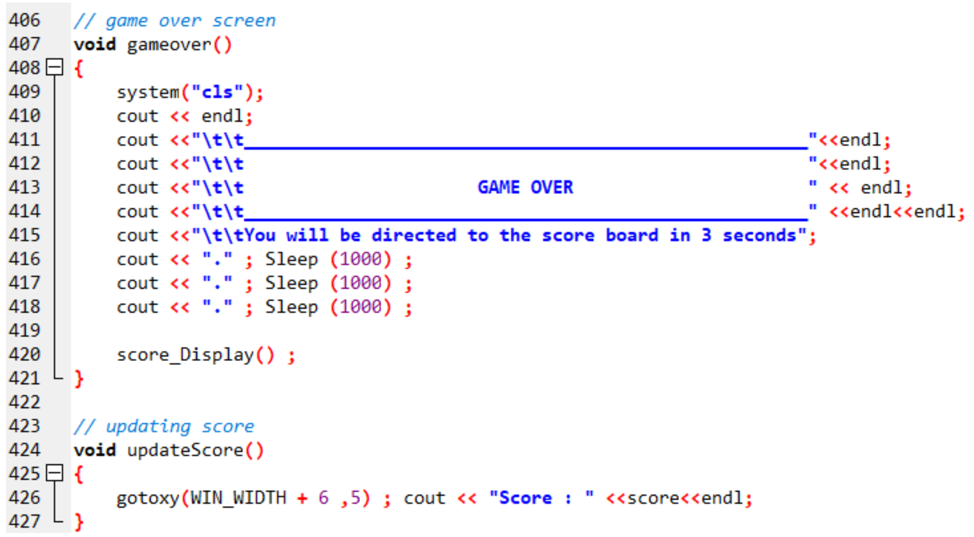
```
// checking bullet and enemy collision
int bulletHit()
{
    for(int i=0; i<20;i++)
    {
        for (int j=0; j <4; j+=2)
        {
            if( bullets[i][j] != 0 )
            {
                if (bullets[i][j] >= enemyY[0] && bullets[i][j] <= enemyY[0]+4 )
                {
                    if(bullets[i][j+1] >= enemyX[0] && bullets[i][j+1] <= enemyX[0]+4)
                    {
                        eraseBullet(i);
                        bullets[i][j] = 0 ;
                        resetEnemy(0) ;
                        return 1 ;
                    }
                }
                if (bullets[i][j] >= enemyY[1] && bullets[i][j] <= enemyY[1]+4 )
                {
                    if(bullets[i][j+1] >= enemyX[1] && bullets[i][j+1] <= enemyX[1]+4)
                    {
                        eraseBullet(i);
                        resetEnemy(1);
                        bullets[i][j] = 0 ;
                        return 1 ;
                    }
                }
            }
        }
    }
    return 0;
}

// generate current time and date
string getCurrentTimenDate()
{
    //taking current date and time to save records in record.txt ;
    //declaring argumnet for time()
    time_t tt;
    //declaring variable to store return value of local time() ;
    struct tm*ti ;
    //applyig time()
    time(&tt);
    //using local time()
    ti= localtime(&tt) ;

    return asctime(ti) ;
}

// displaying scores
void score_Display()
{
    system("cls");
        gotoxy(40,5);cout << "------------------------------" ;
        gotoxy(40,6);cout << "|         SCORE BOARD        |" ;
        gotoxy(40,7);cout << "------------------------------" ;
        gotoxy(10,10);cout << "Your Score : " << score ;
        gotoxy(10,12);cout << "Enter your Name : "  ;

    char user_name[30] ;
    cin >> user_name ;

    ofstream outFile;
    outFile.open("score.txt") ;     // save game details(score, player name, time&date) to score.txt

    outFile << score << "       "<< user_name << "       "<< getCurrentTimenDate() <<"\n"  ;

    outFile.close() ;
```



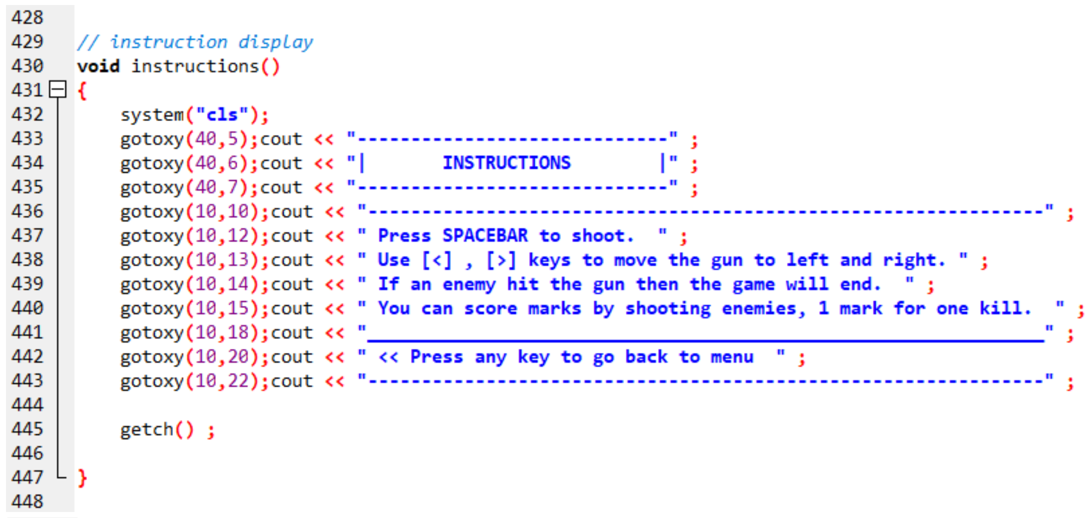
```cpp


    ifstream infile ;
    int user_lasthighscore =0;

    infile.open("highscore.txt");       // reading highscore.txt to get previous high score records

        int last_highscore ;
        while ( infile >> last_highscore  )
    {
        user_lasthighscore   = last_highscore ;     // saving last highscore data from highscore.txt to user_lasthighscore variable
    }
    infile.close() ;

    if (score > user_lasthighscore )    // comparing game score with last high score
    {                                   // score > last highscore; then it should be printed and assigned to variables
    system("cls");
    gotoxy(40,5);cout << "-----------------------------" ;
    gotoxy(40,6);cout << "|        HALL OF FAME        |" ;
    gotoxy(40,7);cout << "-----------------------------" ;
    gotoxy(10,10);cout << "--------------------------------------------------------------" ;
    gotoxy(10,12);cout << "   Congratulations! New high score :) " ;
    gotoxy(10,14);cout << "   Player       : " <<user_name ;
    gotoxy(10,15);cout << "   Score        : " <<score ;
    gotoxy(10,16);cout << "   Time & Date : " << getCurrentTimenDate() ;
    gotoxy(10,18);cout << "--------------------------------------------------------------" ;

    ofstream update;                // updating new highscore ;
    update.open("highscore.txt") ;

    update << score << "       "<< user_name << "       "<< getCurrentTimenDate() <<"\n"  ;

    update.close() ;

    Sleep(7000) ;
    }
```



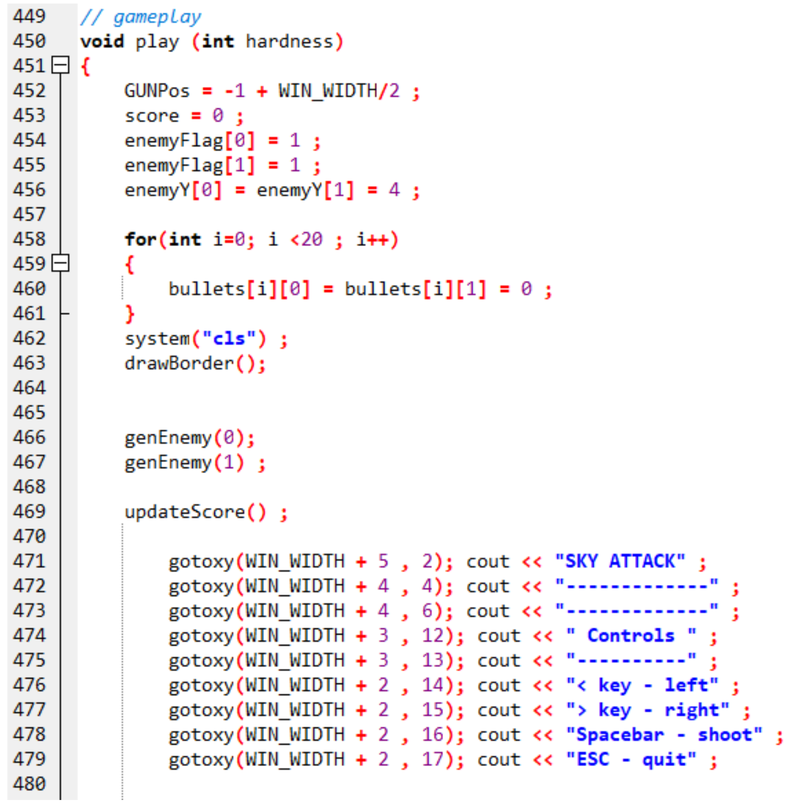
```cpp

    else if (score < user_lasthighscore || score == user_lasthighscore )    // if score <= last high score
    {
    system("cls");

    gotoxy(40,5);cout << "-----------------------------" ;
    gotoxy(40,6);cout << "|        HALL OF FAME        |" ;
    gotoxy(40,7);cout << "-----------------------------" ;
    gotoxy(10,10);cout << "--------------------------------------------------------------" ;
    gotoxy(10,11);cout << "Current high score details " ;
    gotoxy(10,12);

    // taking last high score records from highcore.txt
    string line_ ;
    ifstream file_("highscore.txt") ;
    if (file_.is_open() )
    while (getline(file_,line_))
    {
        cout << line_  ;
    }
    file_.close() ;

    gotoxy(10,15);cout << " Next time you can beat the highscore :) " ;
    gotoxy(10,17);cout << "  Player       : " <<user_name ;
    gotoxy(10,18);cout << "  Score        : " <<score ;
    gotoxy(10,20);cout << "--------------------------------------------------------------" ;

    Sleep(7000) ;
    }
}

```



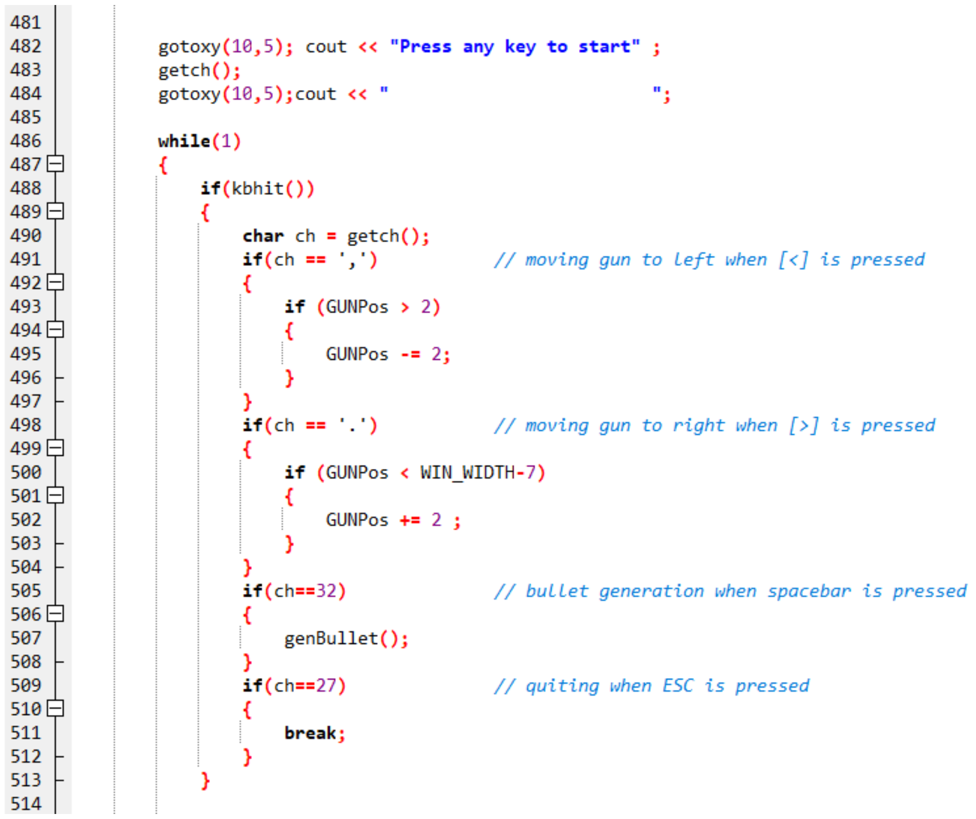
```cpp
// game over screen
void gameover()
{
    system("cls");
    cout << endl;
    cout <<"\t\t______________________________________________________"<<endl;
    cout <<"\t\t                                                      "<<endl;
    cout <<"\t\t                      GAME OVER                       " << endl;
    cout <<"\t\t______________________________________________________" <<endl<<endl;
    cout <<"\t\tYou will be directed to the score board in 3 seconds";
    cout << "." ; Sleep (1000) ;
    cout << "." ; Sleep (1000) ;
    cout << "." ; Sleep (1000) ;

    score_Display() ;
}

// updating score
void updateScore()
{
    gotoxy(WIN_WIDTH + 6 ,5) ; cout << "Score : " <<score<<endl;
}
```



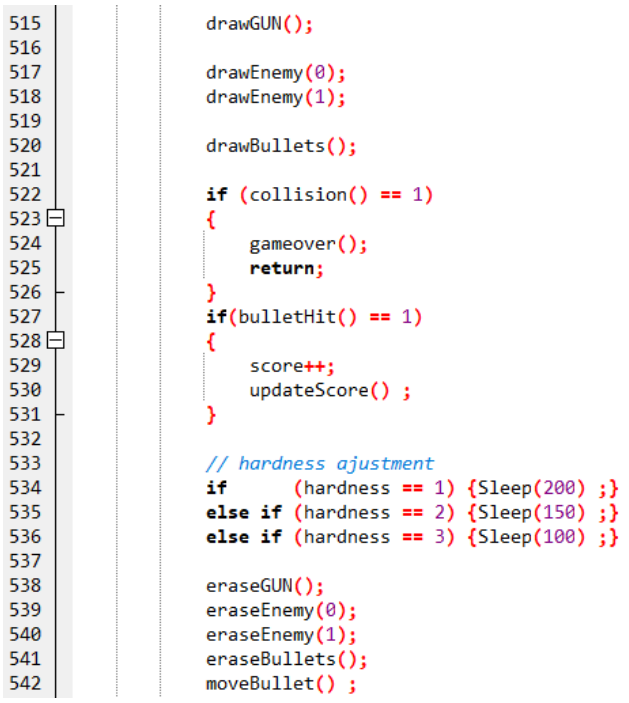
```cpp

// instruction display
void instructions()
{
    system("cls");
    gotoxy(40,5);cout << "------------------------------" ;
    gotoxy(40,6);cout << "|        INSTRUCTIONS        |" ;
    gotoxy(40,7);cout << "------------------------------" ;
    gotoxy(10,10);cout << "-----------------------------------------------------------------" ;
    gotoxy(10,12);cout << " Press SPACEBAR to shoot.  " ;
    gotoxy(10,13);cout << " Use [<] , [>] keys to move the gun to left and right. " ;
    gotoxy(10,14);cout << " If an enemy hit the gun then the game will end.  " ;
    gotoxy(10,15);cout << " You can score marks by shooting enemies, 1 mark for one kill.  " ;
    gotoxy(10,18);cout << "_________________________________________________________________" ;
    gotoxy(10,20);cout << " << Press any key to go back to menu  " ;
    gotoxy(10,22);cout << "-----------------------------------------------------------------" ;

    getch() ;

}

```



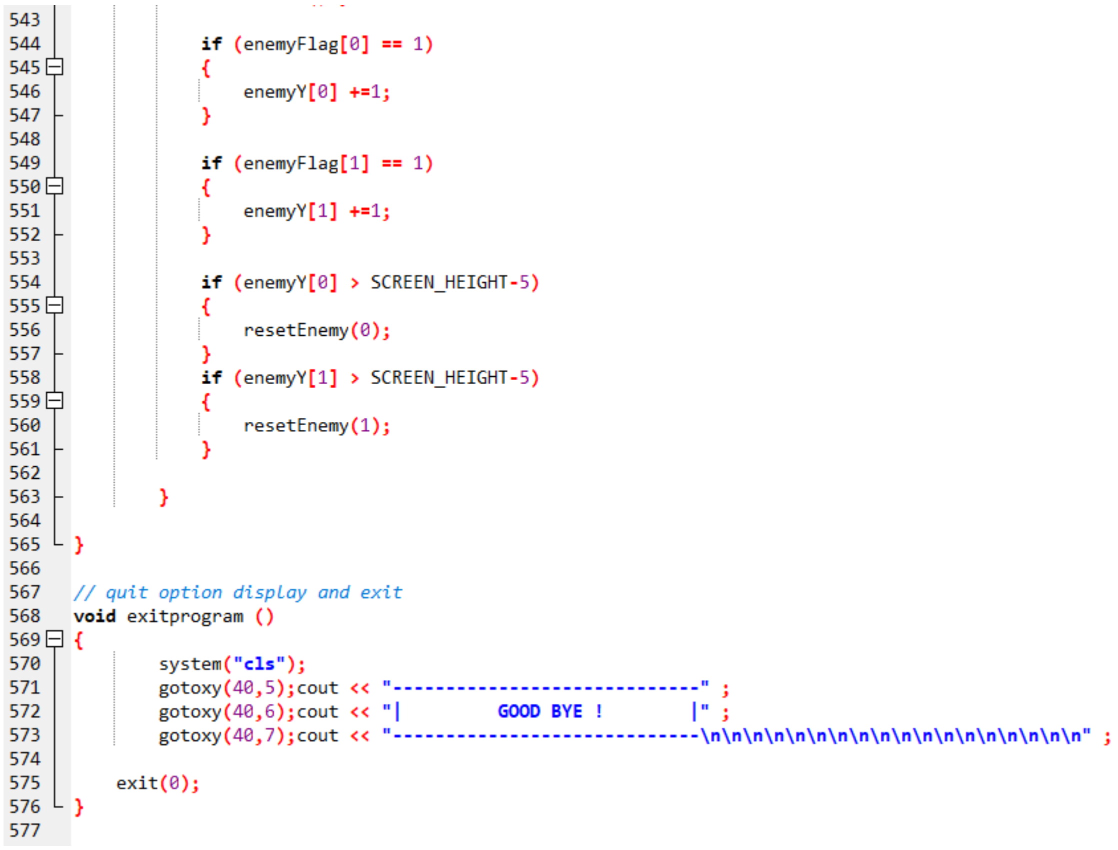
```cpp
// gameplay
void play (int hardness)
{
    GUNPos = -1 + WIN_WIDTH/2 ;
    score = 0 ;
    enemyFlag[0] = 1 ;
    enemyFlag[1] = 1 ;
    enemyY[0] = enemyY[1] = 4 ;

    for(int i=0; i <20 ; i++)
    {
        bullets[i][0] = bullets[i][1] = 0 ;
    }
    system("cls") ;
    drawBorder();


    genEnemy(0);
    genEnemy(1) ;

    updateScore() ;

        gotoxy(WIN_WIDTH + 5 , 2); cout << "SKY ATTACK" ;
        gotoxy(WIN_WIDTH + 4 , 4); cout << "-------------" ;
        gotoxy(WIN_WIDTH + 4 , 6); cout << "-------------" ;
        gotoxy(WIN_WIDTH + 3 , 12); cout << " Controls " ;
        gotoxy(WIN_WIDTH + 3 , 13); cout << "----------" ;
        gotoxy(WIN_WIDTH + 2 , 14); cout << "< key - left" ;
        gotoxy(WIN_WIDTH + 2 , 15); cout << "> key - right" ;
        gotoxy(WIN_WIDTH + 2 , 16); cout << "Spacebar - shoot" ;
        gotoxy(WIN_WIDTH + 2 , 17); cout << "ESC - quit" ;

```



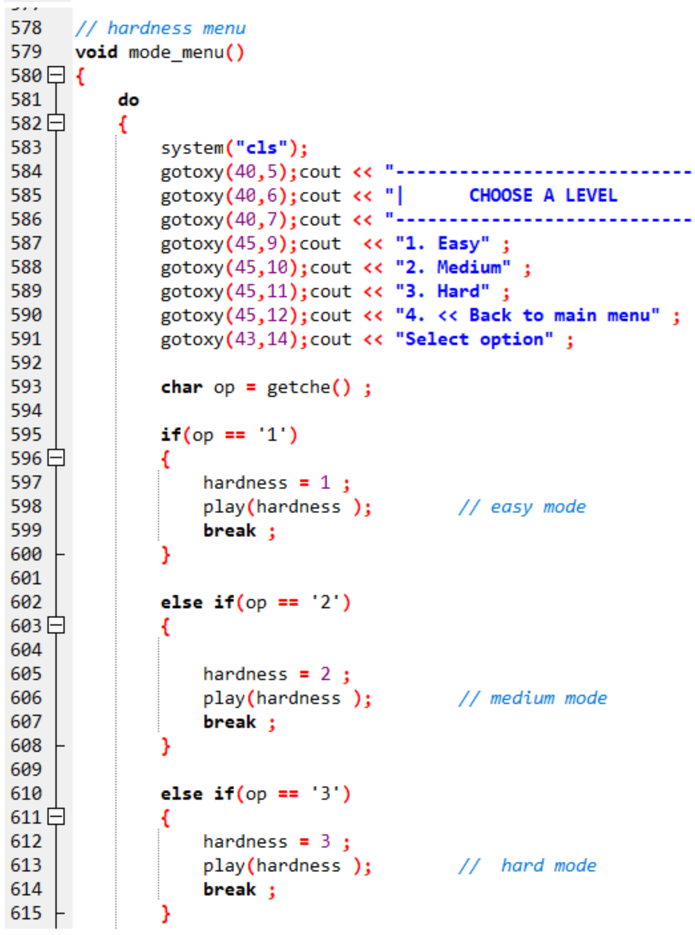
```cpp

gotoxy(10,5); cout << "Press any key to start" ;
getch();
gotoxy(10,5);cout << "                         ";

while(1)
{
    if(kbhit())
    {
        char ch = getch();
        if(ch == ',')          // moving gun to left when [<] is pressed
        {
            if (GUNPos > 2)
            {
                GUNPos -= 2;
            }
        }
        if(ch == '.')          // moving gun to right when [>] is pressed
        {
            if (GUNPos < WIN_WIDTH-7)
            {
                GUNPos += 2 ;
            }
        }
        if(ch==32)             // bullet generation when spacebar is pressed
        {
            genBullet();
        }
        if(ch==27)             // quiting when ESC is pressed
        {
            break;
        }
    }

```



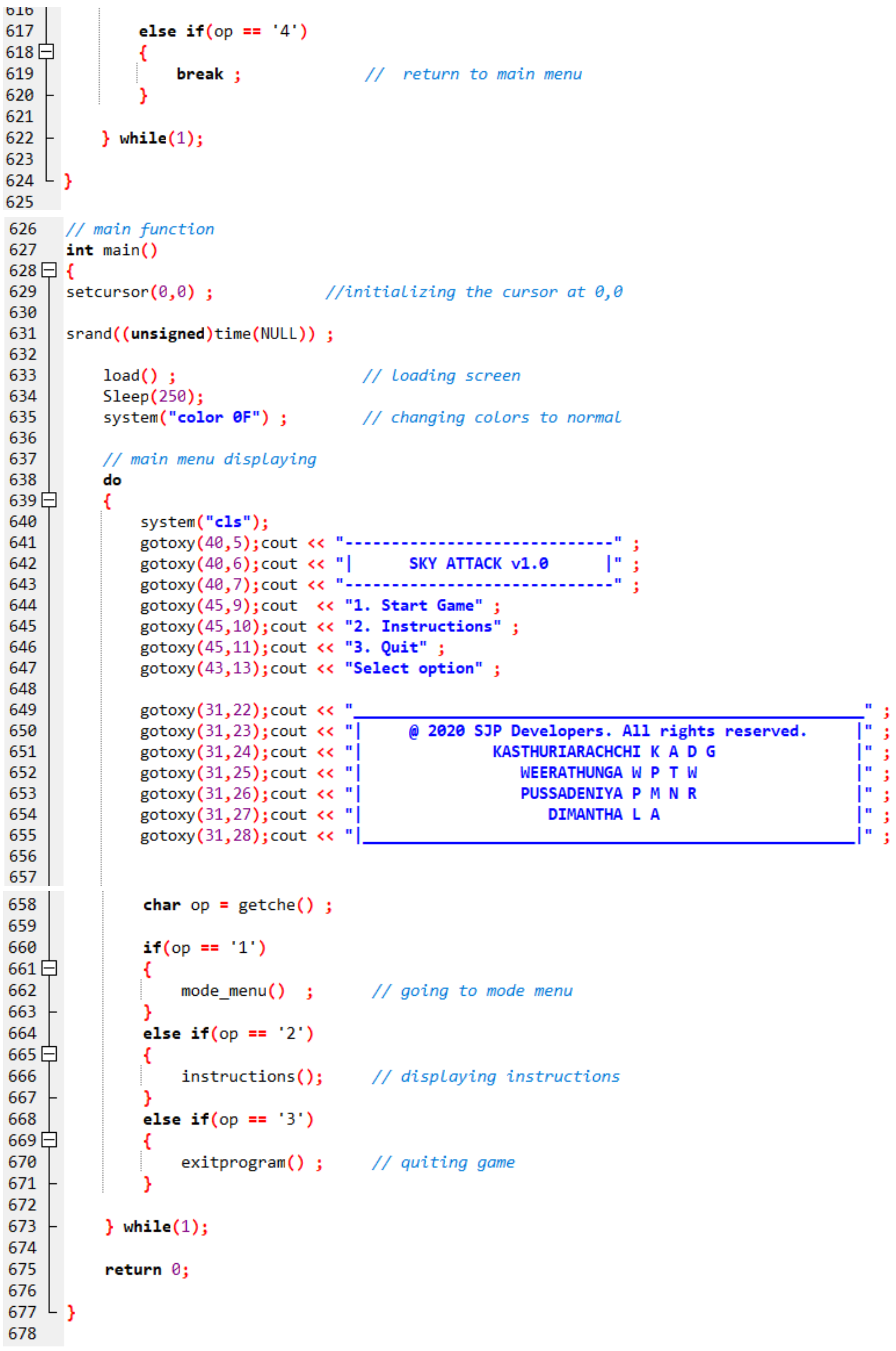
```cpp
    drawGUN();

    drawEnemy(0);
    drawEnemy(1);

    drawBullets();

    if (collision() == 1)
    {
        gameover();
        return;
    }
    if(bulletHit() == 1)
    {
        score++;
        updateScore() ;
    }

    // hardness ajustment
    if      (hardness == 1) {Sleep(200) ;}
    else if (hardness == 2) {Sleep(150) ;}
    else if (hardness == 3) {Sleep(100) ;}

    eraseGUN();
    eraseEnemy(0);
    eraseEnemy(1);
    eraseBullets();
    moveBullet() ;
```



```cpp

            if (enemyFlag[0] == 1)
            {
                enemyY[0] +=1;
            }

            if (enemyFlag[1] == 1)
            {
                enemyY[1] +=1;
            }

            if (enemyY[0] > SCREEN_HEIGHT-5)
            {
                resetEnemy(0);
            }
            if (enemyY[1] > SCREEN_HEIGHT-5)
            {
                resetEnemy(1);
            }

        }

}

// quit option display and exit
void exitprogram ()
{
        system("cls");
        gotoxy(40,5);cout << "-----------------------------" ;
        gotoxy(40,6);cout << "|         GOOD BYE !        |" ;
        gotoxy(40,7);cout << "-----------------------------\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n" ;

    exit(0);
}

```



```cpp
// hardness menu
void mode_menu()
{
    do
    {
        system("cls");
        gotoxy(40,5);cout << "-----------------------------" ;
        gotoxy(40,6);cout << "|      CHOOSE A LEVEL       |" ;
        gotoxy(40,7);cout << "-----------------------------" ;
        gotoxy(45,9);cout  << "1. Easy" ;
        gotoxy(45,10);cout << "2. Medium" ;
        gotoxy(45,11);cout << "3. Hard" ;
        gotoxy(45,12);cout << "4. << Back to main menu" ;
        gotoxy(43,14);cout << "Select option" ;

        char op = getche() ;

        if(op == '1')
        {
            hardness = 1 ;
            play(hardness );        // easy mode
            break ;
        }

        else if(op == '2')
        {

            hardness = 2 ;
            play(hardness );        // medium mode
            break ;
        }

        else if(op == '3')
        {
            hardness = 3 ;
            play(hardness );        //  hard mode
            break ;
        }
```



```cpp
        else if(op == '4')
        {
            break ;              //  return to main menu
        }

    } while(1);

}

// main function
int main()
{
setcursor(0,0) ;           //initializing the cursor at 0,0

srand((unsigned)time(NULL)) ;

    load() ;                     // loading screen
    Sleep(250);
    system("color 0F") ;         // changing colors to normal

    // main menu displaying
    do
    {
        system("cls");
        gotoxy(40,5);cout << "-----------------------------" ;
        gotoxy(40,6);cout << "|      SKY ATTACK v1.0      |" ;
        gotoxy(40,7);cout << "-----------------------------" ;
        gotoxy(45,9);cout  << "1. Start Game" ;
        gotoxy(45,10);cout << "2. Instructions" ;
        gotoxy(45,11);cout << "3. Quit" ;
        gotoxy(43,13);cout << "Select option" ;

        gotoxy(31,22);cout << "_____________________________________________________" ;
        gotoxy(31,23);cout << "|    @ 2020 SJP Developers. All rights reserved.     |" ;
        gotoxy(31,24);cout << "|             KASTHURIARACHCHI K A D G               |" ;
        gotoxy(31,25);cout << "|               WEERATHUNGA W P T W                  |" ;
        gotoxy(31,26);cout << "|               PUSSADENIYA P M N R                  |" ;
        gotoxy(31,27);cout << "|                  DIMANTHA L A                      |" ;
        gotoxy(31,28);cout << "|___________________________________________________|" ;


        char op = getche() ;

        if(op == '1')
        {
            mode_menu()  ;      // going to mode menu
        }
        else if(op == '2')
        {
            instructions();     // displaying instructions
        }
        else if(op == '3')
        {
            exitprogram() ;     // quiting game
        }

    } while(1);

    return 0;

}
```

CHALLENGES

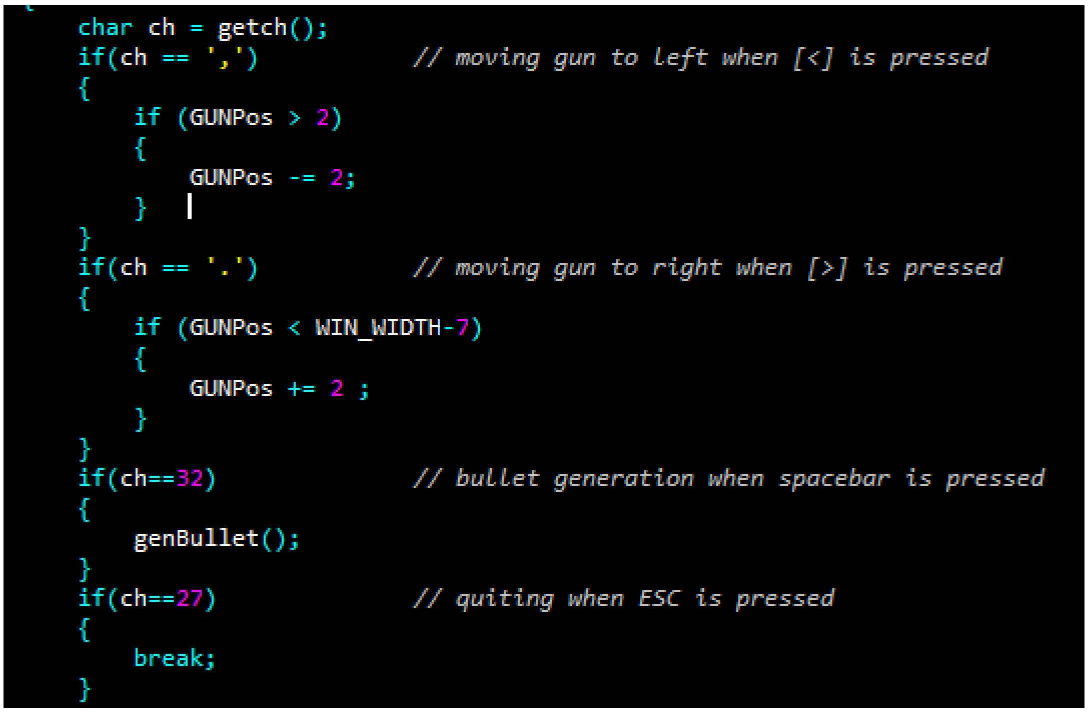
There were some challenges we found while preparing this game project at different stages like designing game and coding game. Some are mentioned below.

1. Changing colors of console background and text.
    - Many programmers have used some easy and special libraries which are specially made to change colors of text and background.
    - Because there were some limits in the game project other libraries which are does not come with C++, we had to do that with the help of standard libraries. therefore to overcome that issue, we included <stdlib.h> header file and used system() function.
    - The syntax is; color <BG_BIT><FG_BIT>
      Where BG_BIT is the bit corresponding to the background color, and FG_BIT the text color.
      The colors are as follows.

background	Foreground
0 – black	8 – gray
1 – blue	9 – light blue
2 – green	A – light green
3 – aqua	B – light aqua
4 – red	C – light red
5 – purple	D – light purple
6 – yellow	E – light yellow
7 – white	F – bright white

Example: system(“color 7c”)

2. Using function keys
    - We needed to use spacebar for shooting and arrow keys for moving but unfortunately, giving inputs in that ways caused to stuck the inputs. therefore, like that way the player could not control the game efficiently and smoothly. To overcome that we decided to use [<], [>] keys to move the gun.

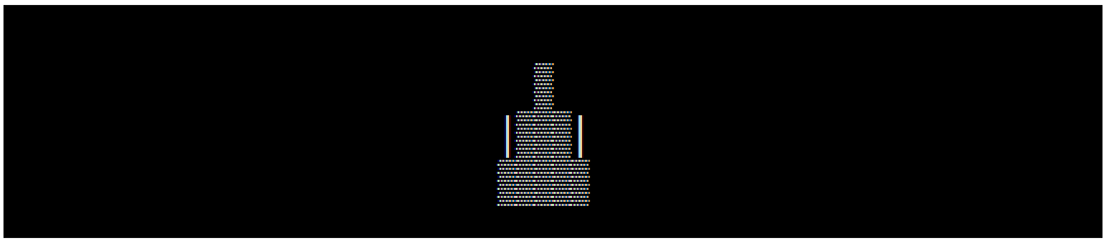


```
    char ch = getch();
    if(ch == ',')            // moving gun to left when [<] is pressed
    {
        if (GUNPos > 2)
        {
            GUNPos -= 2;
        }
    }
    if(ch == '.')            // moving gun to right when [>] is pressed
    {
        if (GUNPos < WIN_WIDTH-7)
        {
            GUNPos += 2 ;
        }
    }
    if(ch==32)               // bullet generation when spacebar is pressed
    {
        genBullet();
    }
    if(ch==27)               // quiting when ESC is pressed
    {
        break;
    }
```

As the above code we removed the ASCII numbers for arrow keys '，' and '．' were replaced.

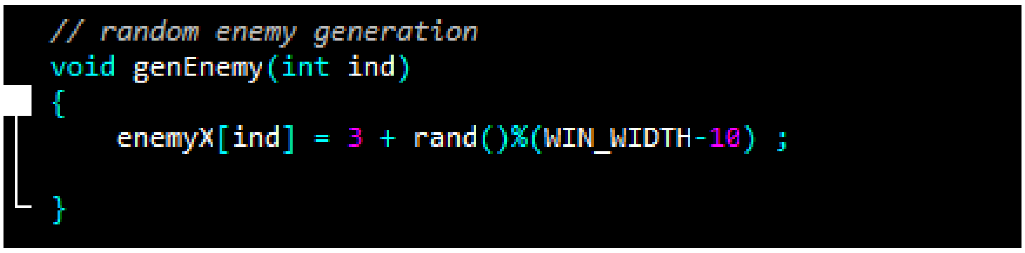
3. The rotating gun

- Here we needed to design game with a rotating gun which indicate the direction but with the standard libraries building that thing was very for us. Therefore, instead of that we decided to make floating gun which is able to move only in horizontal direction.



4. Generating enemies randomly within screen

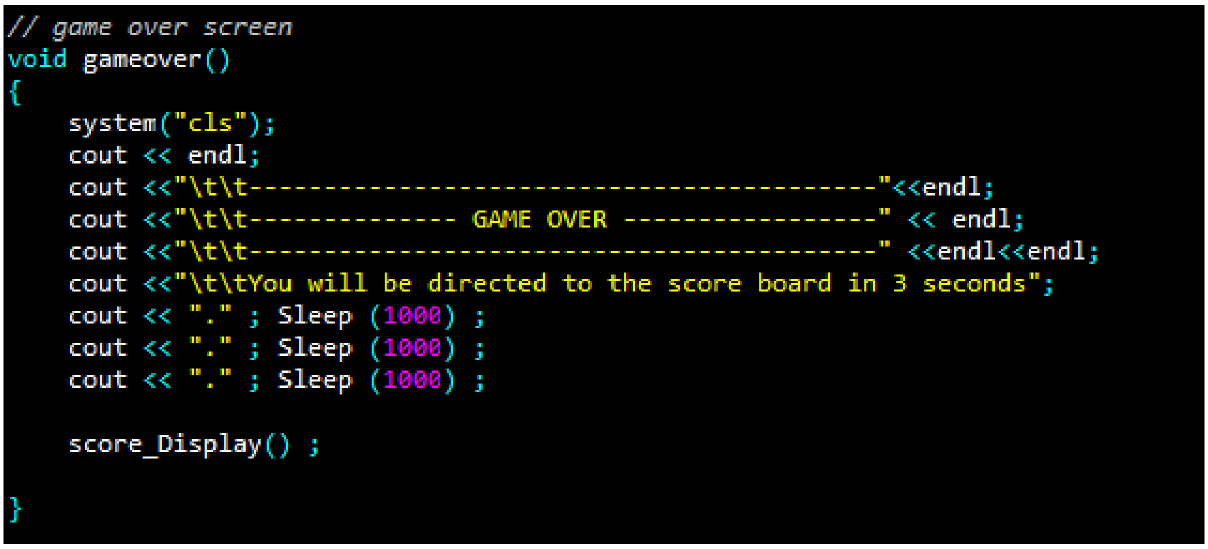
- To randomly do something we need rand() and srand() functions therefore we included <cstdlib> and <time.h> standard header files. Then those functions were included on functions which are used to generate enemies.



```
// random enemy generation
void genEnemy(int ind)
{
    enemyX[ind] = 3 + rand()%(WIN_WIDTH-10) ;
}
```

5. delay a function

- In this game, sometimes we needed to show things for a limited time. As an example, we needed to display ‘gameover’ message for 3 seconds. to do that we included <windows.h> (as operating system) header file then used the Sleep() function.
- Syntax; Sleep (time in milliseconds)



```
// game over screen
void gameover()
{
    system("cls");
    cout << endl;
    cout <<"\t\t------------------------------------------"<<endl;
    cout <<"\t\t-------------- GAME OVER -----------------" << endl;
    cout <<"\t\t------------------------------------------" <<endl<<endl;
    cout <<"\t\tYou will be directed to the score board in 3 seconds";
    cout << "." ; Sleep (1000) ;
    cout << "." ; Sleep (1000) ;
    cout << "." ; Sleep (1000) ;

    score_Display() ;

}
```

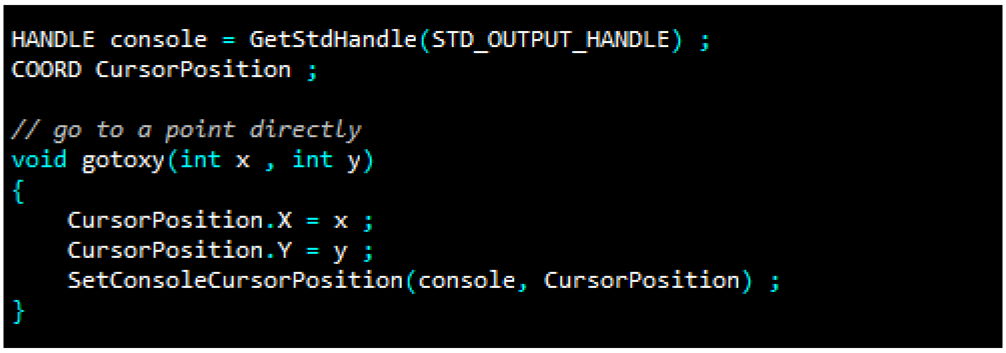
- Sometimes when game is over ,still players inputs can be here, that means when player hit 3 times space bar but when 2nd time player hit the space bar ,if the game is over already the 3rd input which is given by pressing spacebar also going through the console inputs. Then if there is a function like getche() ; in ‘gameover’ screen with the help of that 3rd spacebar input is disappears quickly and player won’t see it . to overcome that issue, we can put Sleep() function there and can wait some time before go to next screen.

6. clear the screen

- Sometimes game screen should be clearer, when C++ compiler print something to screen or whatever, we can see the history of the command outputs. But by making console screen clear it helps to identify things easily especially in games.
- To overcome that issue <cstdlib> header file was included and used system("CLS") function. It will clear the screen.

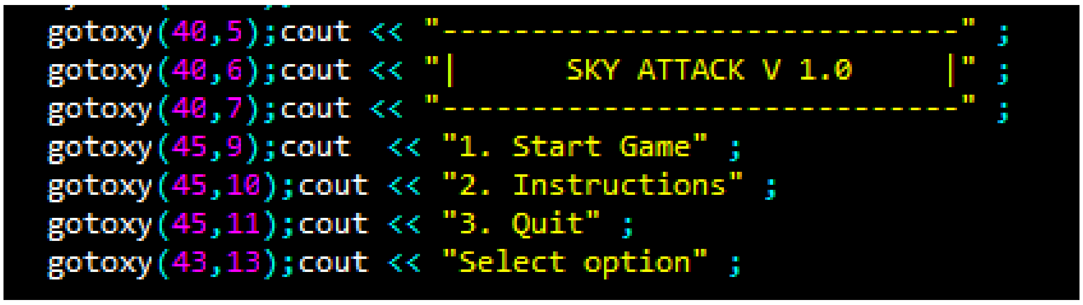
7. move cursor to a specific point in the screen and initializing the cursor

- Here we have decided the length and height of the screen to locate a specific point the cursor we needed a way, therefore we use COORD structure in windows console to manipulate coordinate system. Then we used:
  gotoxy(x coordinate ,y cordinate) function to locate cursor.



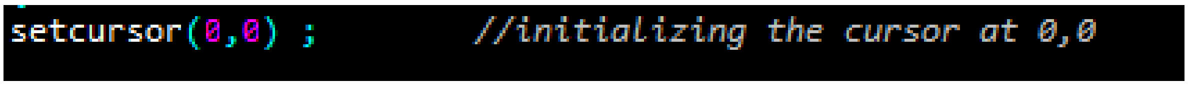
```
HANDLE console = GetStdHandle(STD_OUTPUT_HANDLE) ;
COORD CursorPosition ;

// go to a point directly
void gotoxy(int x , int y)
{
    CursorPosition.X = x ;
    CursorPosition.Y = y ;
    SetConsoleCursorPosition(console, CursorPosition) ;
}
```

Example:



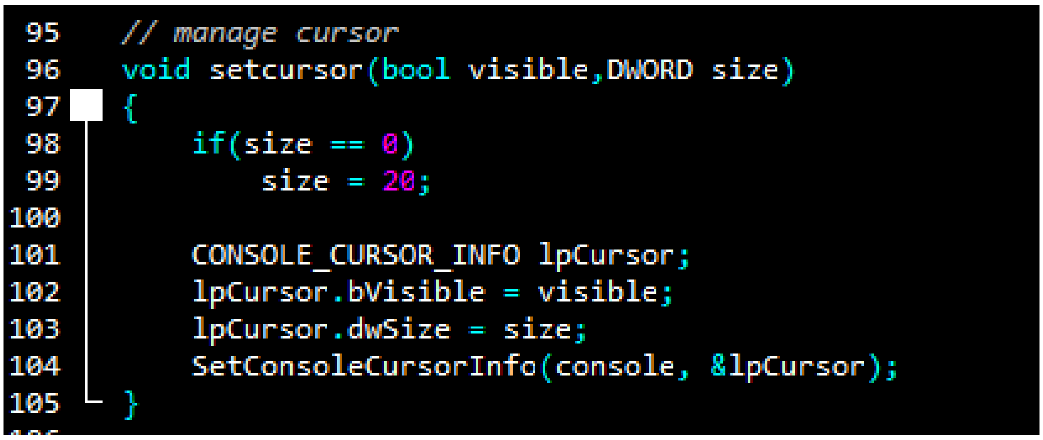
```
gotoxy(40,5);cout << "------------------------------" ;
gotoxy(40,6);cout << "|       SKY ATTACK V 1.0     |" ;
gotoxy(40,7);cout << "------------------------------" ;
gotoxy(45,9);cout  << "1. Start Game" ;
gotoxy(45,10);cout << "2. Instructions" ;
gotoxy(45,11);cout << "3. Quit" ;
gotoxy(43,13);cout << "Select option" ;
```

- to initialize the cursor we made a user defined function called setcursor(),



```
setcursor(0,0) ;        //initializing the cursor at 0,0
```

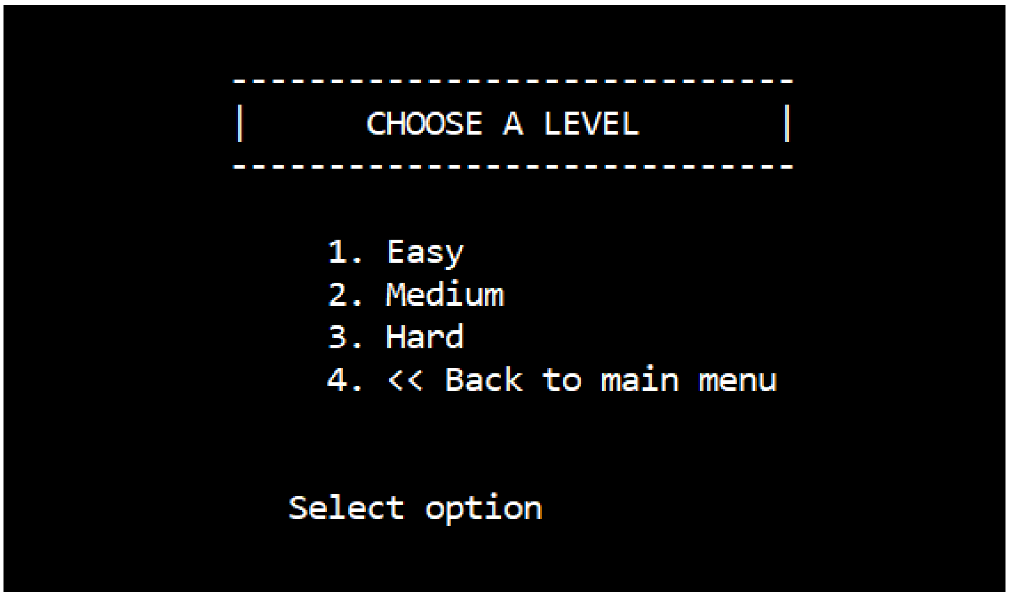
- Function syntax:



```
// manage cursor
void setcursor(bool visible,DWORD size)
{
    if(size == 0)
        size = 20;

    CONSOLE_CURSOR_INFO lpCursor;
    lpCursor.bVisible = visible;
    lpCursor.dwSize = size;
    SetConsoleCursorInfo(console, &lpCursor);
}
```

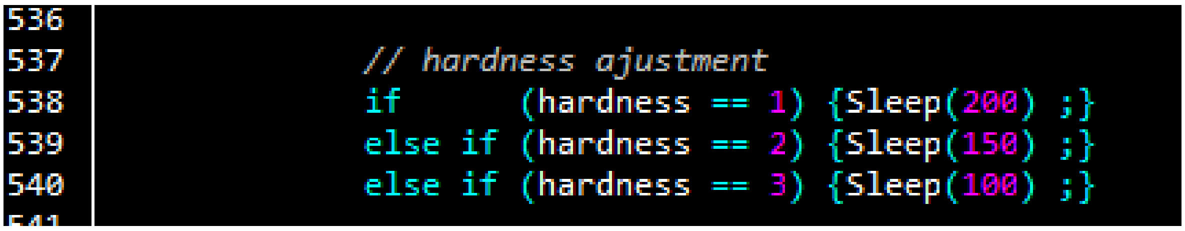
8. game difficulty

- In this game we needed to make some difficulty levels like ‘easy’, ‘medium’ and ‘hard,



```
 -----------------------------
 |       CHOOSE  A  LEVEL       |
 -----------------------------

      1. Easy
      2. Medium
      3. Hard
      4. << Back to main menu


    Select option
```

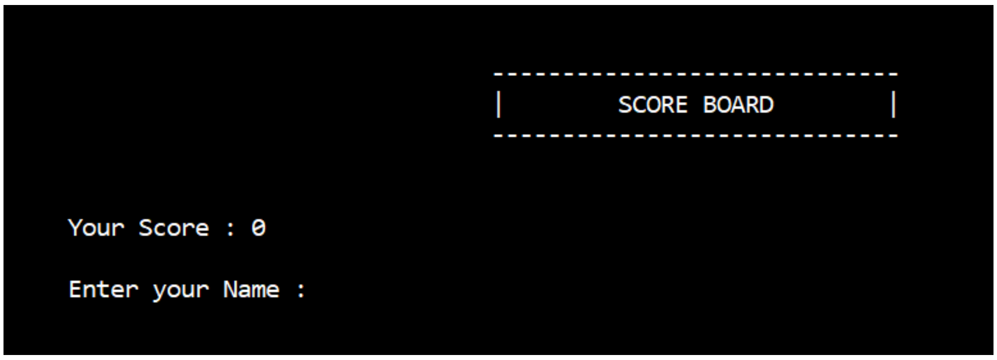
We have adjusted the Sleep() function and it increases the gun moving speed and the random enemies travelling speed.



```
        // hardness ajustment
        if      (hardness == 1) {Sleep(200) ;}
        else if (hardness == 2) {Sleep(150) ;}
        else if (hardness == 3) {Sleep(100) ;}
```

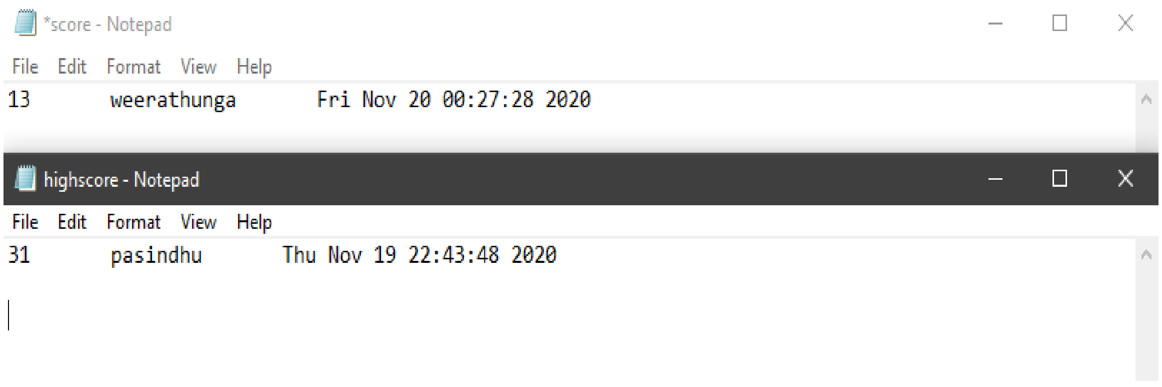
9. Making a Hall of Fame according to high score

- We needed to make game more engaging by making a Hall of fame. It shows the highest score, taken in the game with the player name and the date and time.



```
                              -----------------------------
                              |        SCORE BOARD        |
                              -----------------------------


   Your Score : 0

   Enter your Name :
```

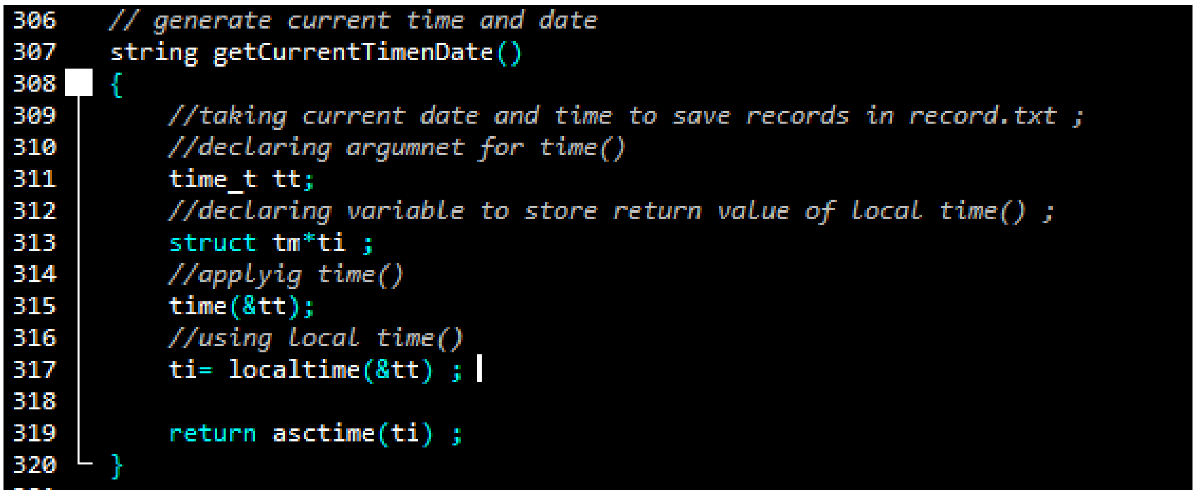
- It was little bit hard and we have done some very simple file handling technique with the help of <fstream> standard header file.
- player can input the name after playing then if the player has scored more than the high score it will be recorded in a separate text file. there are two text files coming with the game cpp file and they are **score.txt** and **highscore.txt.** they keep the records of our scores. Score.txt keeps the current player score and highscore.txt holds the highest score ever like below.



```
13      weerathunga     Fri Nov 20 00:27:28 2020
```

```
31      pasindhu      Thu Nov 19 22:43:48 2020
```

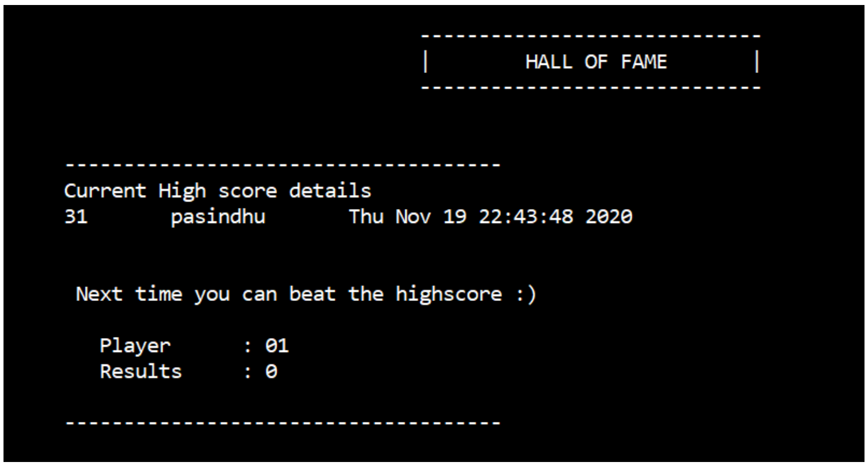
- taking current time and date also a challenge and with the help of <time.h> header file we made a function.



```
// generate current time and date
string getCurrentTimenDate()
{
    //taking current date and time to save records in record.txt ;
    //declaring argumnet for time()
    time_t tt;
    //declaring variable to store return value of local time() ;
    struct tm*ti ;
    //applyig time()
    time(&tt);
    //using local time()
    ti= localtime(&tt) ;

    return asctime(ti) ;
}
```

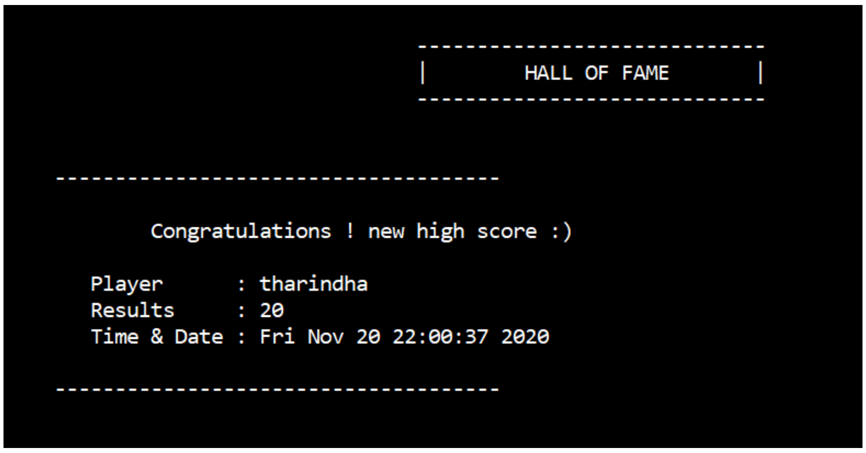
- with the help of ifstream() and ofstream() function we able to read from a text file and write to a text file.

When player could not beat the high score:



```
                                   ----------------------------
                                   |        HALL OF FAME       |
                                   ----------------------------


     ------------------------------------
     Current High score details
     31       pasindhu       Thu Nov 19 22:43:48 2020


      Next time you can beat the highscore :)

        Player      : 01
        Results     : 0

     ------------------------------------
```

When player could beat the high score:



```
                                   -----------------------------
                                   |        HALL OF FAME        |
                                   -----------------------------


    -------------------------------------

            Congratulations ! new high score :)

       Player      : tharindha
       Results     : 20
       Time & Date : Fri Nov 20 22:00:37 2020

    -------------------------------------
```

IMPROVEMENTS

- In this game, when we press ESC the game will disappear and comes to the main menu. But normally the escape function does not quit the game, but it pauses the game till we resume. Therefore, we can develop this game to that state. Then the user is able to pause the game for a while without exiting from it.

- If the game is at the same difficulty level player can gets bored when playing for a long time. So, we can develop this game to become more difficult or harder when a player scoring continuously. Then a player would not go for a higher score easily and it will become more adventurous.

- It is better if we can design the shooting machine as which is able to rotate and spread the bullets.

- Our shooting machine can be modified as a machine that having a health level and when it gets shot the health level gradually decreases. And also, we can drop a medical pack from time to time which we can catch and increase its health.

- Now the game is displayed in a defined area. We can improve this to full screen display, and it is more comfortable to the player.

- Game should have different firing and exploding sounds.

- Graphics can be seen in 2D or 3D with higher bitrate details colorfully.

- Ability to save player profiles. It is easy to play again without typing the name again and again.

CONTRIBUTIONS

Index Number	Name	Contribution
19/ENG/015	DIMANTHA L A	25%
19/ENG/053	KASTHURIARACHCHI K A D G	25%
19/ENG/082	PUSSADENIYA P M N R	25%
19/ENG/111	WEERATHUNGA W P T W	25%