64010899 สิระติ หิรัญธานี

```
#include<iostream>
#include<windows.h>
#include<time.h>
#include<conio.h>
using namespace std;

//Setting System
char keyInput = ' ';
int bulletsNow = 0;
size_t score = 0;
const size_t numSpawn = 20;
int spawnShipX = 38;
int spawnShipY = 29;
enum KEY {LEFT = 97, RIGHT = 100, STOP = 115, SHOOT = 32};

//Object
struct SHIP
{
	enum STATUS { LEFT, RIGHT, STOP };
	STATUS status = STOP;
	int x = spawnShipX;
	int y = spawnShipY;
	string draw = "<-0->";
}ship;

struct BULLETS
{
	enum SHOOT { OPEN, CLOSE };
	SHOOT status = CLOSE;
	int x = ship.x + 2;
	int y = ship.y - 1;
	string draw = "^";
}bullets[5];

struct ENEMY
{
	int x;
	int y;
	string draw = "*";
}enemy[numSpawn];

//System Function
void setcursor(bool visible)
{
	HANDLE console = GetStdHandle(STD_OUTPUT_HANDLE);
	CONSOLE_CURSOR_INFO lpCursor;
	lpCursor.bVisible = visible;
	lpCursor.dwSize = 20;
	SetConsoleCursorInfo(console, &lpCursor);
}
void setcolor(int fg, int bg)
{
	HANDLE hConsole = GetStdHandle(STD_OUTPUT_HANDLE);
	SetConsoleTextAttribute(hConsole, bg * 16 + fg);
}

//Game Function
```

```
void gotoxy(int x, int y)
{
	COORD c ={ x, y };
	SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), c);
}
char cursor(int x, int y)
{
	HANDLE hStd =GetStdHandle(STD_OUTPUT_HANDLE);
	char buf[2];
	COORD c ={ x,y };
	DWORD num_read;
	if(!ReadConsoleOutputCharacter(hStd, (LPTSTR)buf, 1, c, (LPDWORD)&num_read))
		return '\0';
	else
		return buf[0];
}
void scores()
{
	gotoxy(85, 2);
	setcolor(7, 0);
	cout <<"Score :"<< score;
}
void movement(char keyInput)
{
	switch(keyInput)
	{
	case KEY::LEFT:
		ship.status =ship.LEFT;
		break;
	case KEY::RIGHT:
		ship.status =ship.RIGHT;
		break;
	case KEY::STOP:
		ship.status =ship.STOP;
		break;
	case KEY::SHOOT:
		if(bullets[bulletsNow].status ==bullets[bulletsNow].CLOSE)
		{
			bullets[bulletsNow].x =ship.x +2;
			bullets[bulletsNow].y =ship.y -1;
			bullets[bulletsNow].status =bullets[bulletsNow].OPEN;
			bulletsNow++;
			Beep(700, 25);
		}
		break;
	default:
		break;
	}
}

//Ship Function
void erase_ship(int x, int y)
{
	setcolor(2, 0);
	gotoxy(x, y);
	cout <<"  ";
}
```

```
void draw_ship(int x, int y)
{
	setcolor(2, 4);
	gotoxy(x, y);
	cout << ship.draw;
}
void autoLeft()
{
	erase_ship(ship.x, ship.y);
	draw_ship(--ship.x, ship.y);
}
void autoRight()
{
	erase_ship(ship.x, ship.y);
	draw_ship(++ship.x, ship.y);
}

//Shoot Function
void erase_shoot(int x, int y)
{
	setcolor(0, 0);
	gotoxy(x, y);
	cout <<" ";
}
void shoot(int x, int y)
{
	setcolor(2, 0);
	gotoxy(x, y);
	cout << bullets->draw;
}

// Enemy Function
void draw_enemy(int x, int y)
{
	setcolor(7, 0);
	gotoxy(x, y);
	cout << enemy->draw;
}
void initEnemy()
{
	for (int i = 0; i < numSpawn; i++)
	{
		enemy[i].x = rand() % 69 + 10;
		enemy[i].y = rand() % 4 + 2;
		draw_enemy(enemy[i].x, enemy[i].y);
	}
}
void erase_enemy(int x, int y)
{
	setcolor(0, 0);
	gotoxy(x, y);
	cout <<" ";
}
//Collision
void checkCollision(int count)
{
	if (cursor(bullets[count].x, bullets[count].y - 1) == '*')
```

```
        {
                score++;
                scores();
                erase_enemy(bullets[count].x, bullets[count].y - 1);
                erase_shoot(bullets[count].x, bullets[count].y);
                bullets[count].status = bullets[count].CLOSE;
                for (int j = 0; j < numSpawn; j++)
                {
                        if (enemy[j].x == bullets[count].x && enemy[j].y == bullets[count].y - 1)
                        {
                                enemy[j].x = rand() % 69 + 10;
                                enemy[j].y = rand() % 4 + 2;
                                draw_enemy(enemy[j].x, enemy[j].y);
                        }
                }
                Beep(1000, 25);
        }
}
void shootCheck(int count)
{
        if (bullets[count].status == bullets[count].OPEN)
        {
                erase_shoot(bullets[count].x, bullets[count].y);
                if (bullets[count].y > 0)
                {
                        shoot(bullets[count].x, --bullets[count].y);
                        checkCollision(count);
                }
                else
                {
                        bullets[count].status = bullets[count].CLOSE;
                }
        }
}
int main() {
        srand(time(NULL));
        setcursor(0);
        draw_ship(ship.x, ship.y);
        initEnemy();
        scores();
        do {
                bulletsNow %= 5;
                if (_kbhit())
                {
                        keyInput = _getch();
                        movement(keyInput);
                        fflush(stdin);
                }
                if (ship.status == ship.LEFT && ship.x > 0) autoLeft();
                if (ship.status == ship.RIGHT && ship.x < 81) autoRight();
                for (int i = 0; i < 5; i++) shootCheck(i);
                Sleep(60);
        } while (keyInput != 'x');
        return 0;
}
```