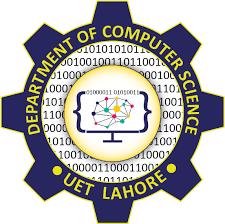
**Asteroids**

****

**Session 2023-2027**

**Submitted by:**

Fajar Shahzad 2023-CS-07

**Supervised by:**

Dr.Awais Hassan

**Course:**

CSC-102L-Programming Fundamentals

Department of Computer Science

**University of Engineering and Technology**

**Lahore Pakistan**

**Table of Contents**

[**Description:** 2](#_heading=h.gjdgxs)

[**Game Character Description: 2**](#_heading=h.8tuqdwsgds6q)

[**SpaceShip(Player): 2**](#_heading=h.yw9ts95iyudt)

[**Asteroids(Enemies):** 3](#_heading=h.ybmmed2ngdo0)

[**Game objects: 3**](#_heading=h.nbn5sxngci1k)

[**Rules and Interactions: 3**](#_heading=h.mtojuoufes70)

[**Goal of Game:** 3](#_heading=h.xdi2trf6iy7p)

[**WireFrames: 4**](#_heading=h.n35gn03rfxjh)

[**Data Structure: 6**](#_heading=h.9yd9bi74tbnr)

[**Function Prototypes: 7**](#_heading=h.syitujv64cz0)

# Description:

Asteroids is a **1979** arcade game where players pilot a spaceship, avoiding asteroids and shooting for points. With basic controls, the game gets progressively harder as more obstacles appear. Survival and point accumulation are the main objectives, and the game ends upon zero health of the player. Its simple yet addictive gameplay makes Asteroids a timeless classic in video game history.As it was a very famous game in past ,I used to play this game with my siblings and I’m addicted to this game.So I decided to make this game with a pinch of my own rules and thrills.

# Game Character Description:

## SpaceShip(Player):

In Asteroids, the player controls a simple spaceship.The spaceship is represented by a basic geometric shape, allowing for easy identification and maneuverability. It can rotate in any direction, thrust forward for movement, and shoot fires to destroy enemies.

## Asteroids(Enemies):

This game has three enemies of different shapes and move in different directions.All these asteroids are irregular shapes that move randomly in he game.All these enemies shot diffrent types of fires at the same time and their aim is to shot the spaceship to reduce the health of spaceship.

* **Enemy 1** continously move up and down and shoot the fire in downward direction
* **Enemy 2** continuously move left and right and shoot the fire in downward direction.
* **Enemy 3** continuously move diagonally and also shoot the fire in downward direction.

# Game objects:

There are two main objects of my game :

* **SpaceShip:**The spaceship is controlled by the plates with move in all direction and shoot the random asteroids.The spaceship also eat the bonus pills to increase the score.
* **Asteroids:**The asteroids are random shapes that move in the game and shoot the fires in diffrent directions to decrease the score of the player and to over the game.

# Rules and Interactions:

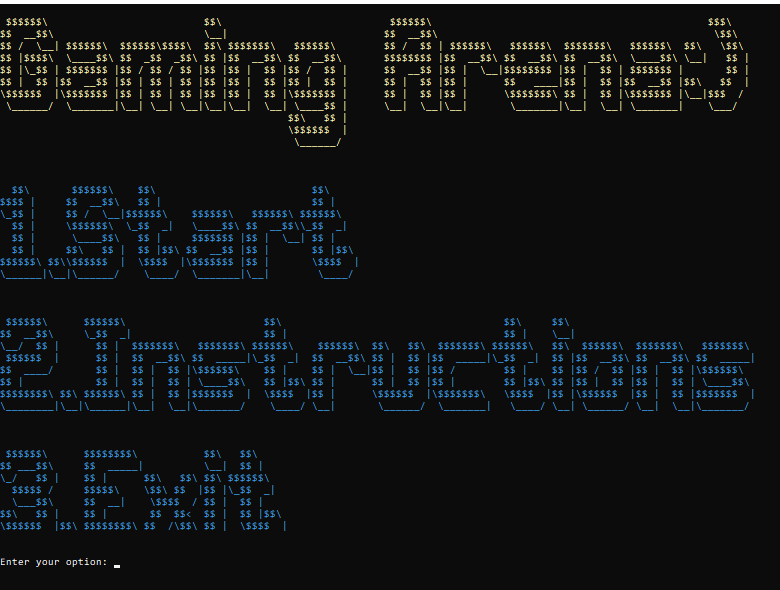
* 1. Press Left Arrow Key <- to move Left the Spaceship
* 2. Press Right Arrow Key -> to move Right the Spaceship
* 3. Press M key to Fire
* 4. Keep Spaceship Away From The Enemies and their Fire
* 5. Save your lives otherwise you'll lose:(
* 6. Press ESC key to End the Game

# Goal of Game:

* **Survival:** Navigate the spaceship through an increasingly hazardous asteroid field without colliding with asteroids or other obstacles. The game challenges players to demonstrate skillful maneuvering to avoid destruction.
* **Score Accumulation:** Destroy asteroids and, optionally, UFOs to earn points. Players aim to achieve the highest score possible by shooting and breaking apart asteroids, with larger asteroids yielding more points.
* **Competition:** Achieving a high score and surpassing previous performances or those of other players adds a competitive and replayable aspect to the game.

Overall, Asteroids provides a classic arcade experience with simple yet challenging objectives, emphasizing survival, skillful play, and high scores.

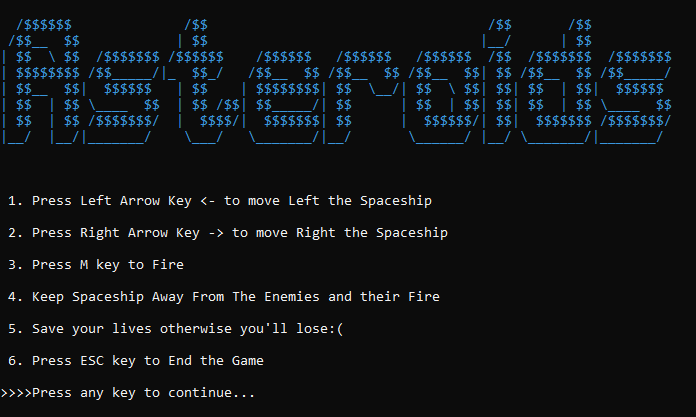
# WireFrames:



**Fig 1.1: Starting Menu**



**Fig 1.2: Game Menu**

****

**Fig 1.3:Instruction Menu**

# Data Structure:

* int pX=27 , pY=27;
* int score=0;
* int health=3;
* int PlayerbulletX = 0;
* int PlayerbulletY = 0;
* bool Playerfired = false;
* int x=85,y=0;
* int lives1=5;
* int lives2=5;
* int lives3=5;
* int eX1=3 , eY1=2;
* bool eY1wallhit=false;
* int eX2=30 , eY2=2;
* bool eX2wallhit=false;
* int eX3=55 , eY3=2;
* bool eX3wallhit=false;
* bool E1fired = false;
* int bullet1X = 0;
* int bullet1Y = 0;
* bool E2fired = false;
* int bullet2X = 0;
* int bullet2Y = 0;
* bool E3fired = false;
* int bullet3X = 0;
* int bullet3Y = 0;

# Function Prototypes:

* void printPlayer();
* void movePlayerRight();
* void movePlayerLeft();
* void movePlayerUp();
* void movePlayerDown();
* void erasePlayer();
* void printPlayerFIRE();
* void movePlayerFIRE();
* void printEnemy1();
* void printEnemy2();
* void printEnemy3();
* void moveEnemy1();
* void moveEnemy2();
* void moveEnemy3();
* void eraseEnemy1();
* void eraseEnemy2();
* void eraseEnemy3();
* void printE1FIRE();
* void printE2FIRE();
* void moveE1FIRE();
* void moveE2FIRE();
* void printE3FIRE();
* void moveE3FIRE();
* string Menu();
* void EndGame();
* void options();
* void welcome();
* void Won();
* void GameOver();
* void reset score();

# Complete Code:

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*LIBRARIES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

#include<iostream>

#include<windows.h>

#include<conio.h>

#include<iomanip>

using namespace std;

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*GOTO FUNCTION\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

void gotoxy(int x, int y);

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*GETCHAR FUNCTION\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

char getCharAtxy(short int x, short int y);

/// PRINT TITLE

void printName();

/// PRINT MAZE

void printBoundary();

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*FUNCTIONS OF PLAYER\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

void printPlayer();

void movePlayerRight();

void movePlayerLeft();

void movePlayerUp();

void movePlayerDown();

void erasePlayer();

void printPlayerFIRE();

void movePlayerFIRE();

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PLAYER'S VARIABLES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

int pX=27 , pY=27;

int score=0;

int health=3;

int PlayerbulletX = 0;

int PlayerbulletY = 0;

bool Playerfired = false;

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ENEMEY'S FUNCTIONS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

void printEnemy1();

void printEnemy2();

void printEnemy3();

void moveEnemy1();

void moveEnemy2();

void moveEnemy3();

void eraseEnemy1();

void eraseEnemy2();

void eraseEnemy3();

void printE1FIRE();

void printE2FIRE();

void moveE1FIRE();

void moveE2FIRE();

void printE3FIRE();

void moveE3FIRE();

string Menu();

void EndGame();

void options();

void welcome();

void Won();

void GameOver();

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ENEMEY'S VARIABLES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

int x=85,y=0;

int lives1=5;

int lives2=5;

int lives3=5;

int eX1=3 , eY1=2;

bool eY1wallhit=false;

int eX2=30 , eY2=2;

bool eX2wallhit=false;

int eX3=55 , eY3=2;

bool eX3wallhit=false;

bool E1fired = false;

int bullet1X = 0;

int bullet1Y = 0;

bool E2fired = false;

int bullet2X = 0;

int bullet2Y = 0;

bool E3fired = false;

int bullet3X = 0;

int bullet3Y = 0;

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Static color setting function for better UI\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

/// COLOR INT VALUES

/// -------------------------------

/// BLACK 0

/// BLUE 1

/// GREEN 2

/// CYAN 3

/// RED 4

/// MAGENTA 5

/// BROWN 6

/// LIGHTGRAY 7

/// DARKGRAY 8

/// LIGHTBLUE 9

/// LIGHTGREEN 10

/// LIGHTCYAN 11

/// LIGHTRED 12

/// LIGHTMAGENTA 13

/// YELLOW 14

/// WHITE 15

string setcolor(unsigned short color);

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DEFINATION OF GOTO AND GETCHAR FUNCTION\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

void gotoxy(int x,int y)

{

COORD coordinates;

coordinates.X=x;

coordinates.Y=y;

SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE),coordinates);

}

char getCharAtxy(short int x, short int y)

{

CHAR\_INFO ci;

COORD xy = {0, 0};

SMALL\_RECT rect = {x, y, x, y};

COORD coordBufSize;

coordBufSize.X = 1;

coordBufSize.Y = 1;

return ReadConsoleOutput(GetStdHandle(STD\_OUTPUT\_HANDLE), &ci, coordBufSize, xy, &rect) ? ci.Char.AsciiChar: ' ';

}

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PRINT TITLE \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

void printName()

{

setcolor(3);

cout<<R"(

/$$$$$$ /$$ /$$ /$$

/$$\_\_ $$ | $$ |\_\_/ | $$

| $$ \ $$ /$$$$$$$ /$$$$$$ /$$$$$$ /$$$$$$ /$$$$$$ /$$ /$$$$$$$ /$$$$$$$

| $$$$$$$$ /$$\_\_\_\_\_/|\_ $$\_/ /$$\_\_ $$ /$$\_\_ $$ /$$\_\_ $$| $$ /$$\_\_ $$ /$$\_\_\_\_\_/

| $$\_\_ $$| $$$$$$ | $$ | $$$$$$$$| $$ \\_\_/| $$ \ $$| $$| $$ | $$| $$$$$$

| $$ | $$ \\_\_\_\_ $$ | $$ /$$| $$\_\_\_\_\_/| $$ | $$ | $$| $$| $$ | $$ \\_\_\_\_ $$

| $$ | $$ /$$$$$$$/ | $$$$/| $$$$$$$| $$ | $$$$$$/| $$| $$$$$$$ /$$$$$$$/

|\_\_/ |\_\_/|\_\_\_\_\_\_\_/ \\_\_\_/ \\_\_\_\_\_\_\_/|\_\_/ \\_\_\_\_\_\_/ |\_\_/ \\_\_\_\_\_\_\_/|\_\_\_\_\_\_\_/

)"<<endl;

setcolor(15);

}

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PRINT MAZE \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

void printBoundary()

{

setcolor(3);

cout<<"####################################################################################"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# $ #"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# $ $ #"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# $ #"<<endl;

cout<<"# #"<<endl;

cout<<"# $ #"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# #"<<endl;

cout<<"# $ #"<<endl;

cout<<"# $ #"<<endl;

cout<<"# #"<<endl;

cout<<"####################################################################################"<<endl;

setcolor(15);

}

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PLAYER FUNCTIONS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

void printPlayer()

{

setcolor(14);

gotoxy(pX,pY);

cout<<" ----- ";

gotoxy(pX,pY+1);

cout<<"\_ \_";

gotoxy(pX,pY+2);

cout<<"| |";

gotoxy(pX,pY+3);

cout<<" \_\_\_\_\_ ";

setcolor(15);

}

void erasePlayer()

{

gotoxy(pX,pY);

cout<<" ";

gotoxy(pX,pY+1);

cout<<" ";

gotoxy(pX,pY+2);

cout<<" ";

gotoxy(pX,pY+3);

cout<<" ";

}

void movePlayerRight()

{

if(getCharAtxy(pX+12,pY)=='$'|| getCharAtxy(pX+12,pY+1)=='$' || getCharAtxy(pX+12,pY+2)=='$'|| getCharAtxy(pX+12,pY+3)=='$')

{

score=score+10;

erasePlayer();

pX=pX+1;

printPlayer();

}

if(getCharAtxy(pX+12,pY)==' '&& getCharAtxy(pX+12,pY+1)==' ' && getCharAtxy(pX+12,pY+2)==' '&& getCharAtxy(pX+12,pY+3)==' ')

{

erasePlayer();

pX=pX+1;

printPlayer();

}

}

void movePlayerLeft()

{

if(getCharAtxy(pX-1,pY)=='$'|| getCharAtxy(pX-1,pY+1)=='$' || getCharAtxy(pX-1,pY+2)=='$' || getCharAtxy(pX-1,pY+3)=='$' )

{

score=score+10;

erasePlayer();

pX=pX-1;

printPlayer();

}

if(getCharAtxy(pX-1,pY)==' '&& getCharAtxy(pX-1,pY+1)==' ' && getCharAtxy(pX-1,pY+2)==' '&& getCharAtxy(pX-1,pY+3)==' ' )

{

erasePlayer();

pX=pX-1;

printPlayer();

}

}

void movePlayerUp()

{

if(getCharAtxy(pX,pY-1)=='$'|| getCharAtxy(pX+1,pY-1)=='$' || getCharAtxy(pX+2,pY-1)=='$' || getCharAtxy(pX+3,pY-1)=='$' || getCharAtxy(pX+4,pY-1)=='$' ||getCharAtxy(pX+5,pY-1)=='$'|| getCharAtxy(pX+6,pY-1)=='$'|| getCharAtxy(pX+7,pY-1)=='$'|| getCharAtxy(pX+8,pY-1)=='$')

{

score=score+10;

erasePlayer();

pY=pY-1;

printPlayer();

}

if(getCharAtxy(pX,pY-1)==' ' && getCharAtxy(pX+1,pY-1)==' '&& getCharAtxy(pX+2,pY-1)==' ' && getCharAtxy(pX+3,pY-1)==' '&& getCharAtxy(pX+4,pY-1==' ')&& getCharAtxy(pX+5,pY-1)==' '&& getCharAtxy(pX+6,pY-1)==' '&& getCharAtxy(pX+7,pY-1)==' '&& getCharAtxy(pX+8,pY-1)==' ')

{

erasePlayer();

pY=pY-1;

printPlayer();

}

}

void movePlayerDown()

{

if(getCharAtxy(pX,pY+4)=='$'|| getCharAtxy(pX+1,pY+4)=='$' || getCharAtxy(pX+2,pY+4)=='$'|| getCharAtxy(pX+3,pY+4)=='$'|| getCharAtxy(pX+4,pY+4)=='$'|| getCharAtxy(pX+5,pY+4)=='$')

{

score=score+10;

erasePlayer();

pY=pY+1;

printPlayer();

}

if(getCharAtxy(pX,pY+4)==' '&& getCharAtxy(pX+1,pY+4)==' ' && getCharAtxy(pX+2,pY+4)==' '&& getCharAtxy(pX+3,pY+4)==' '&& getCharAtxy(pX+4,pY+4)==' '&& getCharAtxy(pX+5,pY+4)==' ')

{

erasePlayer();

pY=pY+1;

printPlayer();

}

}

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ENEMEY'S FUNCTIONS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

void printEnemy1()

{

setcolor(14);

gotoxy(eX1,eY1);

cout<<" ^ ^ ";

gotoxy(eX1,eY1+1);

cout<<" ^ ^ ";

gotoxy(eX1,eY1+2);

cout<<" ^ ";

gotoxy(eX1,eY1+3);

cout<<" ^ ^ ";

gotoxy(eX1,eY1+4);

cout<<" ^ ^ ";

setcolor(15);

}

void eraseEnemy1()

{

gotoxy(eX1,eY1);

cout<<" ";

gotoxy(eX1,eY1+1);

cout<<" ";

gotoxy(eX1,eY1+2);

cout<<" ";

gotoxy(eX1,eY1+3);

cout<<" ";

gotoxy(eX1,eY1+4);

cout<<" ";

}

void printEnemy2()

{

setcolor(14);

gotoxy(eX2,eY2);

cout<<" @@@@@@@@ ";

gotoxy(eX2,eY2+1);

cout<<" @@@@@@@@ ";

gotoxy(eX2,eY2+3);

cout<<" @ % % @ ";

gotoxy(eX2,eY2+4);

cout<<" @ % % @ ";

gotoxy(eX2,eY2+5);

cout<<" @ ";

setcolor(15);

}

void eraseEnemy2()

{

gotoxy(eX2,eY2);

cout<<" ";

gotoxy(eX2,eY2+1);

cout<<" ";

gotoxy(eX2,eY2+3);

cout<<" ";

gotoxy(eX2,eY2+4);

cout<<" ";

gotoxy(eX2,eY2+5);

cout<<" ";

}

void printEnemy3()

{

setcolor(14);

gotoxy(eX3,eY3);

cout<<" \*\*\*\*\*\*\* ";

gotoxy(eX3,eY3+1);

cout<<" \*\*\*\*\* ";

gotoxy(eX3,eY3+2);

cout<<" \*\*\* ";

gotoxy(eX3,eY3+3);

cout<<" \* ";

setcolor(15);

}

void eraseEnemy3()

{

gotoxy(eX3,eY3);

cout<<" ";

gotoxy(eX3,eY3+1);

cout<<" ";

gotoxy(eX3,eY3+2);

cout<<" ";

gotoxy(eX3,eY3+3);

cout<<" ";

}

void moveEnemy1()

{

eraseEnemy1();

if(!eY1wallhit)

eY1=eY1+1;

if(eY1==25)

{

eY1wallhit=true;

}

if(eY1wallhit)

{

eY1=eY1-1;

if(eY1<2)

eY1wallhit=false;

}

printEnemy1();

}

void moveEnemy2()

{

eraseEnemy2();

if(!eX2wallhit)

eX2=eX2+1;

if(eX2==43)

{

eX2wallhit=true;

}

if(eX2wallhit)

{

eX2=eX2-1;

if(eX2<30)

eX2wallhit=false;

}

printEnemy2();

}

void moveEnemy3()

{

eraseEnemy3();

if(!eX3wallhit)

{

eX3=eX3+1;

eY3=eY3+1;

}

if(eX3==65&&eY3==12)

{

eX3wallhit=true;

}

if(eX3wallhit)

{

eX3=eX3-1;

eY3=eY3-1;

if(eX3<56&&eY3<4)

eX3wallhit=false;

}

printEnemy3();

}

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*MAIN FUNCTION \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

main()

{

system("cls");

welcome();

string value = "";

while (value!="3")

{

value=Menu();

}

}

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*FIRING FUNCTIONS \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

void movePlayerFIRE()

{

gotoxy(PlayerbulletX, PlayerbulletY);

cout << " ";

if (getCharAtxy(PlayerbulletX, PlayerbulletY - 1) == ' ')

{

PlayerbulletY = PlayerbulletY-1;

gotoxy(PlayerbulletX, PlayerbulletY);

cout << "!";

}

else if (getCharAtxy(PlayerbulletX,PlayerbulletY - 1) == '\*')

{

lives3--;

PlayerbulletY = PlayerbulletY-1;

gotoxy(PlayerbulletX, PlayerbulletY);

cout << "!";

if(lives3==0)

{

eraseEnemy3();

}

}

else if (getCharAtxy(PlayerbulletX,PlayerbulletY - 1) == '@'||getCharAtxy(PlayerbulletX,PlayerbulletY-1)=='%')

{

PlayerbulletY = PlayerbulletY-1;

gotoxy(PlayerbulletX, PlayerbulletY);

cout << "!";

lives2--;

if(lives2==0)

{

eraseEnemy2();

}

}

else if (getCharAtxy(PlayerbulletX,PlayerbulletY - 1) == '^')

{

lives1--;

PlayerbulletY = PlayerbulletY-1;

gotoxy(PlayerbulletX, PlayerbulletY);

cout << "!";

if(lives1==0)

{

eraseEnemy1();

}

}

else

{

gotoxy(PlayerbulletX, PlayerbulletY);

cout << " ";

Playerfired = false;

}

}

void printPlayerFIRE()

{

PlayerbulletX = pX + 4;

PlayerbulletY = pY - 1;

gotoxy(PlayerbulletX, PlayerbulletY);

cout << "!";

}

void printE1FIRE()

{

if (getCharAtxy(eX1 + 7, eY1 + 7) == ' ')

{

bullet1X = eX1 + 7;

bullet1Y = eY1 + 7;

gotoxy(bullet1X, bullet1Y);

cout << "^";

}

}

void moveE1FIRE()

{

gotoxy(bullet1X, bullet1Y);

cout << " ";

if (getCharAtxy(bullet1X, bullet1Y+1) == ' ')

{

bullet1Y = bullet1Y+1;

gotoxy(bullet1X, bullet1Y);

cout << "^";

}

else if (getCharAtxy(bullet1X,bullet1Y + 1) == '|'||getCharAtxy(bullet1X,bullet1Y + 1)=='-'||getCharAtxy(bullet1X,bullet1Y + 1)=='\_')

{

score--;

bullet1Y= bullet1Y+1;

gotoxy(bullet1X, bullet1Y);

cout << "^";

if(score==0)

{

health--;

}

}

else

{

gotoxy(bullet1X, bullet1Y);

cout << " ";

E1fired = false;

}

}

void printE2FIRE()

{

bullet2X = eX2 + 6;

bullet2Y = eY2 + 6;

gotoxy(bullet2X, bullet2Y);

cout << "%";

}

void moveE2FIRE()

{

gotoxy(bullet2X, bullet2Y);

cout << " ";

if (getCharAtxy(bullet2X, bullet2Y+1) == ' ')

{

bullet2Y = bullet2Y+1;

gotoxy(bullet2X, bullet2Y);

cout << "%";

}

else if (getCharAtxy(bullet2X,bullet2Y+1) == '|'||getCharAtxy(bullet2X,bullet2Y+1)=='-'||getCharAtxy(bullet2X,bullet2Y+1)=='\_')

{

score--;

bullet2Y = bullet2Y+1;

gotoxy(bullet2X, bullet2Y);

cout << "%";

if(score==0)

{

health--;

}

}

else

{

gotoxy(bullet2X, bullet2Y);

cout << " ";

E2fired = false;

}

}

void printE3FIRE()

{

bullet3X = eX3 + 6;

bullet3Y = eY3 + 6;

gotoxy(bullet3X, bullet3Y);

cout << "\*";

}

void moveE3FIRE()

{

gotoxy(bullet3X, bullet3Y);

cout << " ";

if (getCharAtxy(bullet3X, bullet3Y+1) == ' ')

{

bullet3Y = bullet3Y+1;

gotoxy(bullet3X, bullet3Y);

cout << "\*";

}

else if (getCharAtxy(bullet3X,bullet3Y+1) == '|'||getCharAtxy(bullet3X,bullet3Y+1)=='-'||getCharAtxy(bullet3X,bullet3Y+1)=='\_')

{

score--;

bullet3Y = bullet3Y+1;

gotoxy(bullet3X, bullet3Y);

cout << "\*";

if(score==0)

{

health--;

}

}

else

{

gotoxy(bullet3X, bullet3Y);

cout << " ";

E3fired = false;

}

}

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*COLOR SETTING\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

string setcolor(unsigned short color)

{

HANDLE hcon = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleTextAttribute(hcon, color);

return "";

}

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*MAIN MENU \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

string Menu()

{

string option;

while(option!="3")

{

cout<<endl;

options();

again:

cout<<"Enter your option: ";

cin>>option;

if(option<"1"||option>"3")

{

cout<<"Invalid option"<<endl;

goto again;

}

if(option=="1")

{

system("cls");

printBoundary();

printName();

printPlayer();

printEnemy1();

printEnemy2();

while(true)

{

if(GetAsyncKeyState(VK\_LEFT))

{

movePlayerLeft();

}

if(GetAsyncKeyState(VK\_RIGHT))

{

movePlayerRight();

}

if(GetAsyncKeyState(VK\_UP))

{

movePlayerUp();

}

if(GetAsyncKeyState(VK\_DOWN))

{

movePlayerDown();

}

if (GetAsyncKeyState('M'))

{

if (!Playerfired)

{

printPlayerFIRE();

Playerfired = true;

}

if(Playerfired)

{

movePlayerFIRE();

}

}

if(GetAsyncKeyState(VK\_ESCAPE))

{

EndGame();

break;

}

setcolor(14);

if (!E1fired)

{

printE1FIRE();

E1fired = true;

}

if (E1fired)

{

moveE1FIRE();

}

if(lives1==0)

{

eraseEnemy1();

E1fired = true;

}

if (!E2fired)

{

printE2FIRE();

E2fired = true;

}

if (E2fired)

{

moveE2FIRE();

}

if(lives2==0)

{

eraseEnemy2();

E2fired = true;

}

if (!E3fired)

{

printE3FIRE();

E3fired = true;

}

if (E3fired)

{

moveE3FIRE();

}

if(lives3==0)

{

eraseEnemy3();

E3fired = true;

}

gotoxy(x,y);

if(score==0)

cout << "SCORE= " << 0;

else

cout<<"SCORE= "<<score;

gotoxy(x,y+1);

cout<<"LIVES OF ENEMY 1= "<<lives1;

gotoxy(x,y+2);

cout<<"LIVES OF ENEMY 2= "<<lives2;

gotoxy(x,y+3);

cout<<"LIVES OF ENEMY 3= "<<lives3;

gotoxy(x,y+4);

cout<<"HEALTH= "<<health;

setcolor(15);

if (lives1!=0)

moveEnemy1();

Sleep(50);

if (lives2!=0)

moveEnemy2();

Sleep(50);

if (lives3!=0)

moveEnemy3();

Sleep(50);

// if (score!=0)

// movePlayerUp();

// movePlayerDown();

// movePlayerLeft();

// movePlayerRight();

// Sleep(50);

if(lives1==0&&lives2==0&&lives3==0)

{

system("cls");

Won();

Sleep(600);

break;

}

if(health==0)

{

system("cls");

GameOver();

Sleep(600);

break;

}

}

}

if(option=="2")

{

system("cls");

printName();

cout << " 1. Press Left Arrow Key <- to move Left the Spaceship" << endl;

cout<<endl;

cout << " 2. Press Right Arrow Key -> to move Right the Spaceship" << endl;

cout<<endl;

cout << " 3. Press M key to Fire " << endl;

cout<<endl;

cout << " 4. Keep Spaceship Away From The Enemies and their Fire " << endl;

cout<<endl;

cout << " 5. Save your lives otherwise you'll lose:("<<endl;

cout<<endl;

cout << " 6. Press ESC key to End the Game "<<endl;

cout<<endl;

cout << ">>>>Press any key to continue...";

getch();

}

system("cls");

}

return option;

}

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*CLEARING SYSTEM\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

void EndGame()

{

system("cls");

}

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PRINT WELCOME\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

void welcome()

{

setcolor(14);

cout<<R"(

$$$$$$\ $$\ $$$$$$\ $$$\

$$ \_\_$$\ \\_\_| $$ \_\_$$\ \$$\

$$ / \\_\_| $$$$$$\ $$$$$$\$$$$\ $$\ $$$$$$$\ $$$$$$\ $$ / $$ | $$$$$$\ $$$$$$\ $$$$$$$\ $$$$$$\ $$\ \$$\

$$ |$$$$\ \\_\_\_\_$$\ $$ \_$$ \_$$\ $$ |$$ \_\_$$\ $$ \_\_$$\ $$$$$$$$ |$$ \_\_$$\ $$ \_\_$$\ $$ \_\_$$\ \\_\_\_\_$$\ \\_\_| $$ |

$$ |\\_$$ | $$$$$$$ |$$ / $$ / $$ |$$ |$$ | $$ |$$ / $$ | $$ \_\_$$ |$$ | \\_\_|$$$$$$$$ |$$ | $$ | $$$$$$$ | $$ |

$$ | $$ |$$ \_\_$$ |$$ | $$ | $$ |$$ |$$ | $$ |$$ | $$ | $$ | $$ |$$ | $$ \_\_\_\_|$$ | $$ |$$ \_\_$$ |$$\ $$ |

\$$$$$$ |\$$$$$$$ |$$ | $$ | $$ |$$ |$$ | $$ |\$$$$$$$ | $$ | $$ |$$ | \$$$$$$$\ $$ | $$ |\$$$$$$$ |\\_\_|$$$ /

\\_\_\_\_\_\_/ \\_\_\_\_\_\_\_|\\_\_| \\_\_| \\_\_|\\_\_|\\_\_| \\_\_| \\_\_\_\_$$ | \\_\_| \\_\_|\\_\_| \\_\_\_\_\_\_\_|\\_\_| \\_\_| \\_\_\_\_\_\_\_| \\_\_\_/

$$\ $$ |

\$$$$$$ |

\\_\_\_\_\_\_/

)"<<endl;

setcolor(15);

}

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PRINT OPTIONS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

void options()

{

setcolor(3);

cout<<R"(

$$\ $$$$$$\ $$\ $$\

$$$$ | $$ \_\_$$\ $$ | $$ |

\\_$$ | $$ / \\_\_|$$$$$$\ $$$$$$\ $$$$$$\ $$$$$$\

$$ | \$$$$$$\ \\_$$ \_| \\_\_\_\_$$\ $$ \_\_$$\\\_$$ \_|

$$ | \\_\_\_\_$$\ $$ | $$$$$$$ |$$ | \\_\_| $$ |

$$ | $$\ $$ | $$ |$$\ $$ \_\_$$ |$$ | $$ |$$\

$$$$$$\ $$\\$$$$$$ | \$$$$ |\$$$$$$$ |$$ | \$$$$ |

\\_\_\_\_\_\_|\\_\_|\\_\_\_\_\_\_/ \\_\_\_\_/ \\_\_\_\_\_\_\_|\\_\_| \\_\_\_\_/

$$$$$$\ $$$$$$\ $$\ $$\ $$\

$$ \_\_$$\ \\_$$ \_| $$ | $$ | \\_\_|

\\_\_/ $$ | $$ | $$$$$$$\ $$$$$$$\ $$$$$$\ $$$$$$\ $$\ $$\ $$$$$$$\ $$$$$$\ $$\ $$$$$$\ $$$$$$$\ $$$$$$$\

$$$$$$ | $$ | $$ \_\_$$\ $$ \_\_\_\_\_|\\_$$ \_| $$ \_\_$$\ $$ | $$ |$$ \_\_\_\_\_|\\_$$ \_| $$ |$$ \_\_$$\ $$ \_\_$$\ $$ \_\_\_\_\_|

$$ \_\_\_\_/ $$ | $$ | $$ |\$$$$$$\ $$ | $$ | \\_\_|$$ | $$ |$$ / $$ | $$ |$$ / $$ |$$ | $$ |\$$$$$$\

$$ | $$ | $$ | $$ | \\_\_\_\_$$\ $$ |$$\ $$ | $$ | $$ |$$ | $$ |$$\ $$ |$$ | $$ |$$ | $$ | \\_\_\_\_$$\

$$$$$$$$\ $$\ $$$$$$\ $$ | $$ |$$$$$$$ | \$$$$ |$$ | \$$$$$$ |\$$$$$$$\ \$$$$ |$$ |\$$$$$$ |$$ | $$ |$$$$$$$ |

\\_\_\_\_\_\_\_\_|\\_\_|\\_\_\_\_\_\_|\\_\_| \\_\_|\\_\_\_\_\_\_\_/ \\_\_\_\_/ \\_\_| \\_\_\_\_\_\_/ \\_\_\_\_\_\_\_| \\_\_\_\_/ \\_\_| \\_\_\_\_\_\_/ \\_\_| \\_\_|\\_\_\_\_\_\_\_/

$$$$$$\ $$$$$$$$\ $$\ $$\

$$ \_\_\_$$\ $$ \_\_\_\_\_| \\_\_| $$ |

\\_/ $$ | $$ | $$\ $$\ $$\ $$$$$$\

$$$$$ / $$$$$\ \$$\ $$ |$$ |\\_$$ \_|

\\_\_\_$$\ $$ \_\_| \$$$$ / $$ | $$ |

$$\ $$ | $$ | $$ $$< $$ | $$ |$$\

\$$$$$$ |$$\ $$$$$$$$\ $$ /\$$\ $$ | \$$$$ |

\\_\_\_\_\_\_/ \\_\_|\\_\_\_\_\_\_\_\_|\\_\_/ \\_\_|\\_\_| \\_\_\_\_/

)"<<endl;

setcolor(15);

}

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PRINT WINING STATE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

void Won()

{

setcolor(3);

cout<<R"(

$$\ $$\ $$\ $$\ $$$\

\$$\ $$ | $$ | $\ $$ | \$$\

\$$\ $$ /$$$$$$\ $$\ $$\ $$ |$$$\ $$ | $$$$$$\ $$$$$$$\ $$\ \$$\

\$$$$ /$$ \_\_$$\ $$ | $$ | $$ $$ $$\$$ |$$ \_\_$$\ $$ \_\_$$\ \\_\_| $$ |

\$$ / $$ / $$ |$$ | $$ | $$$$ \_$$$$ |$$ / $$ |$$ | $$ | $$ |

$$ | $$ | $$ |$$ | $$ | $$$ / \$$$ |$$ | $$ |$$ | $$ |$$\ $$ |

$$ | \$$$$$$ |\$$$$$$ | $$ / \$$ |\$$$$$$ |$$ | $$ |\\_\_|$$$ /

\\_\_| \\_\_\_\_\_\_/ \\_\_\_\_\_\_/ \\_\_/ \\_\_| \\_\_\_\_\_\_/ \\_\_| \\_\_| \\_\_\_/

)"<<endl;

setcolor(15);

}

///\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PRINT LOSING STATE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*///

void GameOver()

{

setcolor(14);

cout<<R"(

$$$$$$\ $$$$$$\ $$$\

$$ \_\_$$\ $$ \_\_$$\ $$ \_|

$$ / \\_\_| $$$$$$\ $$$$$$\$$$$\ $$$$$$\ $$ / $$ |$$\ $$\ $$$$$$\ $$$$$$\ $$\ $$ /

$$ |$$$$\ \\_\_\_\_$$\ $$ \_$$ \_$$\ $$ \_\_$$\ $$ | $$ |\$$\ $$ |$$ \_\_$$\ $$ \_\_$$\\\_\_|$$ |

$$ |\\_$$ | $$$$$$$ |$$ / $$ / $$ |$$$$$$$$ |$$ | $$ | \$$\$$ / $$$$$$$$ |$$ | \\_\_| $$ |

$$ | $$ |$$ \_\_$$ |$$ | $$ | $$ |$$ \_\_\_\_|$$ | $$ | \$$$ / $$ \_\_\_\_|$$ | $$\ \$$\

\$$$$$$ |\$$$$$$$ |$$ | $$ | $$ |\$$$$$$$\ $$$$$$ | \$ / \$$$$$$$\ $$ | \\_\_| \$$$\

\\_\_\_\_\_\_/ \\_\_\_\_\_\_\_|\\_\_| \\_\_| \\_\_| \\_\_\_\_\_\_\_| \\_\_\_\_\_\_/ \\_/ \\_\_\_\_\_\_\_|\\_\_| \\_\_\_|

)"<<endl;

setcolor(15);

}