# Contra Shooter



Session 2023-2027

**Submitted by:**

Mian Saad Tahir 2023-CS-62

**Supervised by:**

Prof. Dr. Awais Hassan Ma’am Maida Shahid

Department of Computer Science

**University of Engineering and Technology**

**Lahore Pakistan**

[Table of Contents](#_Toc15206)

[[1) Inspiration from original “Contra Shooter”: 3](#_Toc15206)](#_Toc154010479)

[[2) Description of Game: 3](#_Toc15206)](#_Toc154010480)

[[3) Objective of Game: 3](#_Toc15206)](#_Toc154010481)

[[4) Wireframes: 4](#_Toc15206)](#_Toc154010482)

[[5) Prototypes: 5](#_Toc15206)](#_Toc154010483)

[[6) Data Structure: 7](#_Toc15206)](#_Toc154010484)

[[7) Complete Code: 7](#_Toc15206)](#_Toc154010485)

# Inspiration from original “Contra Shooter”:

"Contra" is a classic video game series that originated in the late 1980s and gained widespread popularity for its intense action and challenging gameplay. The first game in the series, titled "Contra," was released by Konami in 1987 for arcade machines. The game was later ported to various home consoles, including the Nintendo Entertainment System (NES).

The background story of the original "Contra" game revolves around two commandos, Bill Rizer and Lance Bean, who are tasked with a mission to thwart the plans of an evil organization known as Red Falcon. This organization, led by an alien entity, seeks to conquer the Earth by unleashing a horde of deadly creatures and advanced weaponry.

The gameplay is known for its side-scrolling, run-and-gun action, where players control one of the commandos as they battle through various levels filled with enemies, obstacles, and powerful bosses. One notable aspect of the game is its cooperative multiplayer mode, allowing two players to team up and tackle the challenges together.

The success of the original "Contra" led to the creation of numerous sequels and spin-offs, each building upon the fast-paced, challenging gameplay that became a hallmark of the series. The Contra series has become a beloved classic in the world of video games, and its impact can still be seen in the gaming industry today.

# Description of Game:

Player has 2 lives. Each life contains 50% health. You can regenarate health by consuming recovery pills. Each pill gives boosts your health by 10%. You can shoot bullets with SPACEBAR key and score points. Score point is incremented by 1 everytime your bullet makes contact with enemy.

There are 3 types of enemies. Each enemy has 100% health. You loose 1 health point if you collide with enemy or if you make contact with enemy's bullets.

# Objective of Game:

Defeat all enemies to win the game

# Wireframes:

**Main Menu:** 

**Instructions:**



**Game Screen**

****

# Prototypes:

strin setcolor(unsigned short color);

voi gotoxy(int x, int y);

void menu();

void box();

void gameOver();

void wonTHEgame();

void pressTOgoBACK();

void produceEnemy1Fire();

void moveEnemy1Fire();

void printEnemy1();

void eraseEnemy1();

void moveEnemy1(string enemy1direction);

string changeDirectionEnemy1(string enemy1direction);

void produceEnemy2Fire();

void moveEnemy2Fire();

void printEnemy2();

void eraseEnemy2();

void moveEnemy2(string enemy2direction);

string changeDirectionEnemy2(string enemy2direction);

void produceEnemy3Fire();

void moveEnemy3Fire();

void printEnemy3();

void eraseEnemy3();

void moveEnemy3(string enemy3direction);

string changeDirectionEnemy3(string enemy3direction);

void printPlayer();

void erasePlayer();

void movePleft();

void movePright();

void movePup();

void movePdown();

void producePlayerFire();

void movePlayerFire();

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

# Data Structure:

Global variable declaration is used for data handling along with gotoxy function for enemy, player and fire movements.

# Complete Code:

#include <iostream>

#include <conio.h>

#include <windows.h>

using namespace std;

string setcolor(unsigned short color);

void gotoxy(int x, int y); // for setting coordinates on screen

void menu(); // prints menu

void box(); // prints game maze

void gameOver(); // lost the game

void wonTHEgame();

void pressTOgoBACK();

void produceEnemy1Fire();

void moveEnemy1Fire();

void printEnemy1(); // prints first enemy

void eraseEnemy1(); // erases first enemy

void moveEnemy1(string enemy1direction); // moves first enemy

string changeDirectionEnemy1(string enemy1direction); // changes directions of first enemy

void produceEnemy2Fire();

void moveEnemy2Fire();

void printEnemy2(); // prints second enemy

void eraseEnemy2(); // erases second enemy

void moveEnemy2(string enemy2direction); // moves second enemy

string changeDirectionEnemy2(string enemy2direction); // changes directions of second enemy

void produceEnemy3Fire();

void moveEnemy3Fire();

void printEnemy3(); // prints third enemy

void eraseEnemy3(); // erases third enemy

void moveEnemy3(string enemy3direction); // moves third enemy

string changeDirectionEnemy3(string enemy3direction); // changes directions of third enemy

void printPlayer(); // prints player

void erasePlayer(); // erases player

void movePleft(); // moves player left

void movePright(); // moves player right

void movePup(); // moves player up

void movePdown(); // moves player down

void producePlayerFire(); // prints player's fire

void movePlayerFire(); // moves player's fire

char getCharAtxy(short int x, short int y); // gets coordinates

bool gameLoop = true; // loop to play the game

int playerHealthX = 93, playerHealthY = 13; // coordinates to display player health

int playerScoreX = 93, playerScoreY = 12; // coordinates to display player score

int playerHealth = 50; // stores player health info

int playerLives = 2; // lives of player

int enemy1X = 27, enemy1Y = 11; // coordinates of enemy1

int enemy2X = 40, enemy2Y = 11; // coordinates of enemy2

int enemy3X = 68, enemy3Y = 16; // coordinates of enemy3

int playerX = 55, playerY = 31; // coordinates of player

int enemyCount = 3; // count of remaining enemies

int enemy1health = 100, enemy2health = 100, enemy3health = 100;

int enemy1healthX = 93, enemy1healthY = 16;

int playerScore = 0;

int erasedEnemy1Count = 0, erasedEnemy2Count = 0, erasedEnemy3Count = 0;

main()

{

system("cls");

string option;

menu();

gotoxy(49, 12);

setcolor(11);

cout << "1. Start Game";

gotoxy(49, 13);

cout << "2. How to Play";

gotoxy(49, 14);

cout << "3. Exit ";

gotoxy(49, 16);

setcolor(2);

cout << "Your Choice > ";

getline(cin >> ws, option);

if (option == "1")

{

string enemy1direction = "down", enemy2direction = "right", enemy3direction = "diagonalUp";

box();

printPlayer();

while (gameLoop)

{

gotoxy(enemy1healthX, enemy1healthY + 4);

setcolor(15);

cout << "Remaining Enemies: " << enemyCount;

gotoxy(enemy1healthX, enemy1healthY);

setcolor(4);

cout << "Enemy 1 Health: " << enemy1health;

gotoxy(enemy1healthX, enemy1healthY + 1);

cout << "Enemy 2 Health: " << enemy2health;

gotoxy(enemy1healthX, enemy1healthY + 2);

cout << "Enemy 3 Health: " << enemy3health;

gotoxy(playerScoreX, playerScoreY);

setcolor(5);

cout << "Score: " << playerScore;

gotoxy(playerHealthX, playerHealthY);

cout << "Player Health: " << playerHealth;

gotoxy(playerHealthX, playerHealthY + 1);

cout << "Remaining Player Lives: " << playerLives;

if (GetAsyncKeyState(VK\_SPACE))

{

producePlayerFire();

}

if (GetAsyncKeyState(VK\_LEFT))

{

movePleft();

}

if (GetAsyncKeyState(VK\_UP))

{

movePup();

}

if (GetAsyncKeyState(VK\_DOWN))

{

movePdown();

}

if (GetAsyncKeyState(VK\_RIGHT))

{

movePright();

}

if (erasedEnemy1Count < 1)

{

if (enemy1health == 0)

{

enemyCount--;

eraseEnemy1();

erasedEnemy1Count++;

}

}

if (erasedEnemy2Count < 1)

{

if (enemy2health == 0)

{

enemyCount--;

eraseEnemy2();

erasedEnemy2Count++;

}

}

if (erasedEnemy3Count < 1)

{

if (enemy3health == 0)

{

enemyCount--;

eraseEnemy3();

erasedEnemy3Count++;

}

}

if (enemy1health > 0)

{

moveEnemy1(enemy1direction);

if (enemy1Y == 13 || enemy1Y == 18 || enemy1Y == 21)

{

produceEnemy1Fire();

}

enemy1direction = changeDirectionEnemy1(enemy1direction);

}

if (enemy2health > 0)

{

moveEnemy2(enemy2direction);

if (enemy2X == 50 || enemy2X == 40 || enemy2X == 45 || enemy2X == 55 || enemy2X == 58 || enemy2X == 60)

{

produceEnemy2Fire();

}

enemy2direction = changeDirectionEnemy2(enemy2direction);

}

if (enemy3health > 0)

{

moveEnemy3(enemy3direction);

if (enemy3Y == 11 || enemy3Y == 14)

{

produceEnemy3Fire();

}

enemy3direction = changeDirectionEnemy3(enemy3direction);

}

if (playerHealth == -1)

{

playerLives--;

playerHealth = 50;

}

if (playerLives == 0)

{

gameOver();

gameLoop == false;

break;

}

if (enemyCount == 0)

{

wonTHEgame();

gameLoop == false;

break;

}

// whenever health dropped below 10 and 100, it's value on scoreboard started to malfunction so I came up with this solution

if (enemy1health == 90)

{

gotoxy(enemy1healthX, enemy1healthY);

cout << "Enemy 1 health: 90 ";

}

if (enemy2health == 90)

{

gotoxy(enemy1healthX, enemy1healthY + 1);

cout << "Enemy 2 health: 90 ";

}

if (enemy3health == 90)

{

gotoxy(enemy1healthX, enemy1healthY + 2);

cout << "Enemy 3 health: 90 ";

}

if (playerHealth == 9)

{

gotoxy(playerHealthX, playerHealthY);

cout << "Player Health: 9 ";

}

movePlayerFire();

moveEnemy1Fire();

moveEnemy2Fire();

moveEnemy3Fire();

}

}

else if (option == "2")

{

setcolor(1);

gotoxy(44, 12);

cout << "<<< About the Game >>>";

gotoxy(46, 13);

cout << " ";

setcolor(5);

gotoxy(35, 14);

cout << "> Player has 2 lives ";

gotoxy(35, 15);

cout << "> Each life contains 50% health";

gotoxy(35, 16);

cout << "> You can regenarate health by consuming recovery pills";

gotoxy(35, 17);

cout << "> You can shoot bullets with SPACEBAR key and score points";

gotoxy(35, 19);

cout << "> There are 3 types of enemies";

gotoxy(35, 20);

cout << "> Each enemy has 100% health";

gotoxy(35, 21);

cout << "> You loose health if you collide with enemy";

gotoxy(35, 22);

cout << "> You loose health if you make contact with enemy's bullets";

gotoxy(35, 24);

cout << "> Defeat all enemies to win the game";

pressTOgoBACK();

}

else if (option == "3")

{

gameLoop = false;

}

else

{

setcolor(4);

gotoxy(49, 17);

cout << "Invalid choice!";

getch();

main();

}

getch();

}

string changeDirectionEnemy1(string enemy1direction)

{

if (enemy1Y + 3 == 25)

{

enemy1direction = "up";

}

if (enemy1Y == 11)

{

enemy1direction = "down";

}

return enemy1direction;

}

string changeDirectionEnemy2(string enemy2direction)

{

if (enemy2X + 3 == 66)

{

enemy2direction = "left";

}

if (enemy2X == 40)

{

enemy2direction = "right";

}

return enemy2direction;

}

string changeDirectionEnemy3(string enemy3direction)

{

if (enemy3X == 73 && enemy3Y == 11)

{

enemy3direction = "diagonalDown";

}

if (enemy3X == 68 && enemy3Y == 16)

{

enemy3direction = "diagonalUp";

}

return enemy3direction;

}

void printPlayer()

{

if (playerLives > 0)

{

if (playerLives == 2)

{

setcolor(14);

gotoxy(playerX, playerY);

cout << "<|>";

gotoxy(playerX, playerY + 1);

cout << "|^|";

gotoxy(playerX, playerY + 2);

cout << "|:|";

}

else if (playerLives == 1)

{

setcolor(5);

gotoxy(playerX, playerY);

cout << "<|>";

gotoxy(playerX, playerY + 1);

cout << "|^|";

gotoxy(playerX, playerY + 2);

cout << "|:|";

}

}

}

void erasePlayer()

{

gotoxy(playerX, playerY);

cout << " ";

gotoxy(playerX, playerY + 1);

cout << " ";

gotoxy(playerX, playerY + 2);

cout << " ";

}

void producePlayerFire()

{

if (getCharAtxy(playerX, playerY - 1) == ' ')

{

gotoxy(playerX + 1, playerY - 1);

cout << "!";

}

}

void movePlayerFire()

{

int playerFired = 0;

for (int x = 1; x < 120; x++)

{

for (int y = 1; y < 40; y++)

{

if (playerFired > 0)

{

playerFired = 0;

continue;

}

if (getCharAtxy(x, y) == '!')

{

if (getCharAtxy(x, y - 1) == ' ' || getCharAtxy(x, y - 1) == 'o' || getCharAtxy(x, y - 1) == '+' || getCharAtxy(x, y - 1) == '\*')

{

gotoxy(x, y);

cout << " ";

gotoxy(x, y - 1);

cout << "!";

playerFired++;

}

if (getCharAtxy(x, y - 1) == '|' || getCharAtxy(x, y - 1) == '-')

{

gotoxy(x, y);

cout << " ";

enemy1health -= 10;

playerScore++;

}

if (getCharAtxy(x, y - 1) == '\\' || getCharAtxy(x, y - 1) == '/' || getCharAtxy(x, y - 1) == '~')

{

gotoxy(x, y);

cout << " ";

enemy2health -= 10;

playerScore++;

}

if (getCharAtxy(x, y - 1) == 'T' || getCharAtxy(x, y - 1) == '`')

{

gotoxy(x, y);

cout << " ";

enemy3health -= 10;

playerScore++;

}

if (getCharAtxy(x, y - 1) == '\_' || getCharAtxy(x, y - 1) == 'I' || getCharAtxy(x, y - 1) == '=' || getCharAtxy(x, y - 1) == '0' || getCharAtxy(x, y - 1) == ':')

{

gotoxy(x, y);

cout << " ";

}

}

}

}

}

void movePleft()

{

if ((getCharAtxy(playerX - 1, playerY) == ' ' && getCharAtxy(playerX - 1, playerY + 1) == ' ' && getCharAtxy(playerX - 1, playerY + 2) == ' ') || (getCharAtxy(playerX - 1, playerY) == '0' || getCharAtxy(playerX - 1, playerY + 1) == '0' || getCharAtxy(playerX - 1, playerY + 2) == '0'))

{

erasePlayer();

if (getCharAtxy(playerX - 1, playerY) == '0' || getCharAtxy(playerX - 1, playerY + 1) == '0' || getCharAtxy(playerX - 1, playerY + 2) == '0')

{

playerHealth = playerHealth + 10;

}

playerX--;

printPlayer();

}

}

void movePright()

{

if ((getCharAtxy(playerX + 3, playerY) == ' ' && getCharAtxy(playerX + 3, playerY + 1) == ' ' && getCharAtxy(playerX + 3, playerY + 2) == ' ') || (getCharAtxy(playerX + 3, playerY) == '0' || getCharAtxy(playerX + 3, playerY + 1) == '0' || getCharAtxy(playerX + 3, playerY + 2) == '0'))

{

erasePlayer();

if (getCharAtxy(playerX + 3, playerY) == '0' || getCharAtxy(playerX + 3, playerY + 1) == '0' || getCharAtxy(playerX + 3, playerY + 2) == '0')

{

playerHealth = playerHealth + 10;

}

playerX++;

printPlayer();

}

}

void movePup()

{

if ((getCharAtxy(playerX, playerY - 1) == ' ' && getCharAtxy(playerX + 1, playerY - 1) == ' ' && getCharAtxy(playerX + 2, playerY - 1) == ' ') || (getCharAtxy(playerX, playerY - 1) == '0' || getCharAtxy(playerX + 1, playerY - 1) == '0' || getCharAtxy(playerX + 2, playerY - 1) == '0'))

{

erasePlayer();

if (getCharAtxy(playerX, playerY - 1) == '0' || getCharAtxy(playerX + 1, playerY - 1) == '0' || getCharAtxy(playerX + 2, playerY - 1) == '0')

{

playerHealth = playerHealth + 10;

}

playerY--;

printPlayer();

}

}

void movePdown()

{

if ((getCharAtxy(playerX, playerY + 3) == ' ' && getCharAtxy(playerX + 1, playerY + 3) == ' ' && getCharAtxy(playerX + 2, playerY + 3) == ' ') || (getCharAtxy(playerX, playerY + 3) == '0' || getCharAtxy(playerX + 1, playerY + 3) == '0' || getCharAtxy(playerX + 2, playerY + 3) == '0'))

{

erasePlayer();

if (getCharAtxy(playerX, playerY + 3) == '0' || getCharAtxy(playerX + 1, playerY + 3) == '0' || getCharAtxy(playerX + 2, playerY + 3) == '0')

{

playerHealth = playerHealth + 10;

}

playerY++;

printPlayer();

}

}

void printEnemy1()

{

setcolor(3);

gotoxy(enemy1X, enemy1Y);

cout << "---";

gotoxy(enemy1X, enemy1Y + 1);

cout << "(1)";

gotoxy(enemy1X, enemy1Y + 2);

cout << "-|-";

}

void eraseEnemy1()

{

gotoxy(enemy1X, enemy1Y);

cout << " ";

gotoxy(enemy1X, enemy1Y + 1);

cout << " ";

gotoxy(enemy1X, enemy1Y + 2);

cout << " ";

}

void moveEnemy1(string enemy1direction)

{

eraseEnemy1();

if (enemy1direction == "down")

{

if (getCharAtxy(enemy1X, enemy1Y + 3) == ' ' && getCharAtxy(enemy1X + 1, enemy1Y + 3) == ' ' && getCharAtxy(enemy1X + 2, enemy1Y + 3) == ' ')

{

enemy1Y += 1;

}

if ((getCharAtxy(enemy1X, enemy1Y + 3) == '<' || getCharAtxy(enemy1X + 1, enemy1Y + 3) == '<' || getCharAtxy(enemy1X + 2, enemy1Y + 3) == '<') || (getCharAtxy(enemy1X, enemy1Y + 3) == '|' || getCharAtxy(enemy1X + 1, enemy1Y + 3) == '|' || getCharAtxy(enemy1X + 2, enemy1Y + 3) == '|') || (getCharAtxy(enemy1X, enemy1Y + 3) == '>' || getCharAtxy(enemy1X + 1, enemy1Y + 3) == '>' || getCharAtxy(enemy1X + 2, enemy1Y + 3) == '>'))

{

playerHealth = playerHealth - 1;

}

if ((getCharAtxy(enemy1X + 3, enemy1Y) == '<' || getCharAtxy(enemy1X + 3, enemy1Y + 1) == '<' || getCharAtxy(enemy1X + 3, enemy1Y + 2) == '<') || (getCharAtxy(enemy1X + 3, enemy1Y) == '|' || getCharAtxy(enemy1X + 3, enemy1Y + 1) == '|' || getCharAtxy(enemy1X + 3, enemy1Y + 2) == '|'))

{

playerHealth = playerHealth - 1;

}

if ((getCharAtxy(enemy1X - 1, enemy1Y) == '<' || getCharAtxy(enemy1X - 1, enemy1Y + 1) == '<' || getCharAtxy(enemy1X - 1, enemy1Y + 2) == '<') || (getCharAtxy(enemy1X - 1, enemy1Y) == '|' || getCharAtxy(enemy1X - 1, enemy1Y + 1) == '|' || getCharAtxy(enemy1X - 1, enemy1Y + 2) == '|'))

{

playerHealth = playerHealth - 1;

}

}

if (enemy1direction == "up")

{

if (getCharAtxy(enemy1X, enemy1Y - 1) == ' ' && getCharAtxy(enemy1X + 1, enemy1Y - 1) == ' ' && getCharAtxy(enemy1X + 2, enemy1Y - 1) == ' ')

{

enemy1Y -= 1;

}

if ((getCharAtxy(enemy1X, enemy1Y - 1) == '|' || getCharAtxy(enemy1X + 1, enemy1Y - 1) == '|' || getCharAtxy(enemy1X + 2, enemy1Y - 1) == '|') || (getCharAtxy(enemy1X, enemy1Y - 1) == ':' || getCharAtxy(enemy1X + 1, enemy1Y - 1) == ':' || getCharAtxy(enemy1X + 2, enemy1Y - 1) == ':'))

{

playerHealth -= 1;

}

if ((getCharAtxy(enemy1X + 3, enemy1Y) == '<' || getCharAtxy(enemy1X + 3, enemy1Y + 1) == '<' || getCharAtxy(enemy1X + 3, enemy1Y + 2) == '<') || (getCharAtxy(enemy1X + 3, enemy1Y) == '|' || getCharAtxy(enemy1X + 3, enemy1Y + 1) == '|' || getCharAtxy(enemy1X + 3, enemy1Y + 2) == '|'))

{

playerHealth -= 1;

}

if ((getCharAtxy(enemy1X - 1, enemy1Y) == '<' || getCharAtxy(enemy1X - 1, enemy1Y + 1) == '<' || getCharAtxy(enemy1X - 1, enemy1Y + 2) == '<') || (getCharAtxy(enemy1X - 1, enemy1Y) == '|' || getCharAtxy(enemy1X - 1, enemy1Y + 1) == '|' || getCharAtxy(enemy1X - 1, enemy1Y + 2) == '|'))

{

playerHealth -= 1;

}

}

printEnemy1();

}

void produceEnemy1Fire()

{

if (getCharAtxy(enemy1X + 1, enemy1Y + 3) == ' ')

{

setcolor(3);

gotoxy(enemy1X + 1, enemy1Y + 4);

cout << "o";

}

}

void moveEnemy1Fire()

{

int enemy1Fired = 0;

for (int x = 1; x < 70; x++)

{

for (int y = 10; y < 40; y++)

{

if (enemy1Fired > 0)

{

enemy1Fired = 0;

continue;

}

if (getCharAtxy(x, y) == 'o')

{

if (getCharAtxy(x, y + 1) == ' ')

{

gotoxy(x, y);

cout << " ";

gotoxy(x, y + 1);

cout << "o";

enemy1Fired++;

}

else if (getCharAtxy(x, y + 1) == '|' || getCharAtxy(x, y + 1) == '<' || getCharAtxy(x, y + 1) == '>')

{

gotoxy(x, y);

cout << " ";

playerHealth--;

}

if (getCharAtxy(x, y + 1) == '\_' || getCharAtxy(x, y + 1) == '0' || getCharAtxy(x, y + 1) == '=' || getCharAtxy(x, y + 1) == '/' || getCharAtxy(x, y + 1) == '\\' || getCharAtxy(x, y + 1) == 'I' || getCharAtxy(x, y + 1) == ':')

{

gotoxy(x, y);

cout << " ";

}

}

}

}

}

void produceEnemy2Fire()

{

if (getCharAtxy(enemy2X + 1, enemy2Y + 3) == ' ')

{

setcolor(1);

gotoxy(enemy2X + 1, enemy2Y + 3);

cout << "+";

}

}

void moveEnemy2Fire()

{

int enemy2Fired = 0;

for (int x = 1; x < 80; x++)

{

for (int y = 10; y < 40; y++)

{

if (enemy2Fired > 0)

{

enemy2Fired = 0;

continue;

}

if (getCharAtxy(x, y) == '+')

{

if (getCharAtxy(x, y + 1) == ' ')

{

gotoxy(x, y);

cout << " ";

gotoxy(x, y + 1);

cout << "+";

enemy2Fired++;

}

else if (getCharAtxy(x, y + 1) == '|' || getCharAtxy(x, y + 1) == '<' || getCharAtxy(x, y + 1) == '>')

{

gotoxy(x, y);

cout << " ";

playerHealth--;

}

if (getCharAtxy(x, y + 1) == '\_' || getCharAtxy(x, y + 1) == '0' || getCharAtxy(x, y + 1) == '=' || getCharAtxy(x, y + 1) == '/' || getCharAtxy(x, y + 1) == '\\' || getCharAtxy(x, y + 1) == 'I' || getCharAtxy(x, y + 1) == ':')

{

gotoxy(x, y);

cout << " ";

}

}

}

}

}

void printEnemy2()

{

setcolor(1);

gotoxy(enemy2X, enemy2Y);

cout << "[ ]";

gotoxy(enemy2X, enemy2Y + 1);

cout << "|2|";

gotoxy(enemy2X, enemy2Y + 2);

cout << "\\~/";

}

void eraseEnemy2()

{

gotoxy(enemy2X, enemy2Y);

cout << " ";

gotoxy(enemy2X, enemy2Y + 1);

cout << " ";

gotoxy(enemy2X, enemy2Y + 2);

cout << " ";

}

void moveEnemy2(string enemy2direction)

{

eraseEnemy2();

if (enemy2direction == "right")

{

if (getCharAtxy(enemy2X + 3, enemy2Y) == ' ' && getCharAtxy(enemy2X + 3, enemy2Y + 1) == ' ' && getCharAtxy(enemy2X + 3, enemy2Y + 2) == ' ')

{

enemy2X += 1;

}

if ((getCharAtxy(enemy2X + 3, enemy2Y) == '<' || getCharAtxy(enemy2X + 3, enemy2Y + 1) == '<' || getCharAtxy(enemy2X + 3, enemy2Y + 2) == '<') || (getCharAtxy(enemy2X + 3, enemy2Y) == '|' || getCharAtxy(enemy2X + 3, enemy2Y + 1) == '|' || getCharAtxy(enemy2X + 3, enemy2Y + 2) == '|'))

{

playerHealth -= 1;

}

if ((getCharAtxy(enemy2X, enemy2Y + 3) == '<' || getCharAtxy(enemy2X + 1, enemy2Y + 3) == '<' || getCharAtxy(enemy2X + 2, enemy2Y + 3) == '<') || (getCharAtxy(enemy2X, enemy2Y + 3) == '|' || getCharAtxy(enemy2X + 1, enemy2Y + 3) == '|' || getCharAtxy(enemy2X + 2, enemy2Y + 3) == '|') || (getCharAtxy(enemy2X, enemy2Y + 3) == '>' || getCharAtxy(enemy2X + 1, enemy2Y + 3) == '>' || getCharAtxy(enemy2X + 2, enemy2Y + 3) == '>'))

{

playerHealth -= 1;

}

}

if (enemy2direction == "left")

{

if (getCharAtxy(enemy2X - 1, enemy2Y) == ' ' && getCharAtxy(enemy2X - 1, enemy2Y + 1) == ' ' && getCharAtxy(enemy2X - 1, enemy2Y + 2) == ' ')

{

enemy2X -= 1;

}

if ((getCharAtxy(enemy2X - 1, enemy2Y) == '>' || getCharAtxy(enemy2X - 1, enemy2Y + 1) == '>' || getCharAtxy(enemy2X - 1, enemy2Y + 2) == '>') || (getCharAtxy(enemy2X - 1, enemy2Y) == '|' || getCharAtxy(enemy2X - 1, enemy2Y + 1) == '|' || getCharAtxy(enemy2X - 1, enemy2Y + 2) == '|'))

{

playerHealth -= 1;

}

if ((getCharAtxy(enemy2X, enemy2Y + 3) == '<' || getCharAtxy(enemy2X + 1, enemy2Y + 3) == '<' || getCharAtxy(enemy2X + 2, enemy2Y + 3) == '<') || (getCharAtxy(enemy2X, enemy2Y + 3) == '|' || getCharAtxy(enemy2X + 1, enemy2Y + 3) == '|' || getCharAtxy(enemy2X + 2, enemy2Y + 3) == '|') || (getCharAtxy(enemy2X, enemy2Y + 3) == '>' || getCharAtxy(enemy2X + 1, enemy2Y + 3) == '>' || getCharAtxy(enemy2X + 2, enemy2Y + 3) == '>'))

{

playerHealth -= 1;

}

}

printEnemy2();

}

void produceEnemy3Fire()

{

if (getCharAtxy(enemy3X + 1, enemy3Y + 3) == ' ')

{

setcolor(4);

gotoxy(enemy3X + 1, enemy3Y + 3);

cout << "\*";

}

}

void moveEnemy3Fire()

{

int enemy3Fired = 0;

for (int x = 1; x < 100; x++)

{

for (int y = 10; y < 40; y++)

{

if (enemy3Fired > 0)

{

enemy3Fired = 0;

continue;

}

if (getCharAtxy(x, y) == '\*')

{

if (getCharAtxy(x, y + 1) == ' ')

{

gotoxy(x, y);

cout << " ";

gotoxy(x, y + 1);

cout << "\*";

enemy3Fired++;

}

else if (getCharAtxy(x, y + 1) == '|' || getCharAtxy(x, y + 1) == '<' || getCharAtxy(x, y + 1) == '>')

{

gotoxy(x, y);

cout << " ";

playerHealth--;

}

if (getCharAtxy(x, y + 1) == '\_' || getCharAtxy(x, y + 1) == '0' || getCharAtxy(x, y + 1) == '=' || getCharAtxy(x, y + 1) == '/' || getCharAtxy(x, y + 1) == '\\' || getCharAtxy(x, y + 1) == 'I' || getCharAtxy(x, y + 1) == ':')

{

gotoxy(x, y);

cout << " ";

}

}

}

}

}

void printEnemy3()

{

setcolor(4);

gotoxy(enemy3X, enemy3Y);

cout << "^-^";

gotoxy(enemy3X, enemy3Y + 1);

cout << "{3}";

gotoxy(enemy3X, enemy3Y + 2);

cout << "`T`";

}

void eraseEnemy3()

{

gotoxy(enemy3X, enemy3Y);

cout << " ";

gotoxy(enemy3X, enemy3Y + 1);

cout << " ";

gotoxy(enemy3X, enemy3Y + 2);

cout << " ";

}

void moveEnemy3(string enemy3direction)

{

eraseEnemy3();

if (enemy3direction == "diagonalUp")

{

if (getCharAtxy(enemy3X + 3, enemy3Y) == ' ' && getCharAtxy(enemy3X + 3, enemy3Y - 1) == ' ' && getCharAtxy(enemy3X + 3, enemy3Y + 1) == ' ' && getCharAtxy(enemy3X + 2, enemy3Y - 1) == ' ' && getCharAtxy(enemy3X + 1, enemy3Y - 1) == ' ' && getCharAtxy(enemy3X, enemy3Y - 1) == ' ')

{

enemy3X += 1;

enemy3Y -= 1;

}

if ((getCharAtxy(enemy3X + 3, enemy3Y) == '<' || getCharAtxy(enemy3X + 3, enemy3Y + 1) == '<' || getCharAtxy(enemy3X + 3, enemy3Y + 2) == '<') || (getCharAtxy(enemy3X + 3, enemy3Y) == '|' || getCharAtxy(enemy3X + 3, enemy3Y + 1) == '|' || getCharAtxy(enemy3X + 3, enemy3Y + 2) == '|'))

{

playerHealth -= 1;

}

if ((getCharAtxy(enemy3X, enemy3Y - 1) == '|' || getCharAtxy(enemy3X + 1, enemy3Y - 1) == '|' || getCharAtxy(enemy3X + 2, enemy3Y - 1) == '|') || (getCharAtxy(enemy3X, enemy3Y - 1) == ':' || getCharAtxy(enemy3X + 1, enemy3Y - 1) == ':' || getCharAtxy(enemy3X + 2, enemy3Y - 1) == ':'))

{

playerHealth -= 1;

}

if (getCharAtxy(enemy3X - 1, enemy3Y) == '|' || getCharAtxy(enemy3X - 1, enemy3Y + 1) == '|' || getCharAtxy(enemy3X - 1, enemy3Y + 2) == '|' || getCharAtxy(enemy3X - 1, enemy3Y) == '>' || getCharAtxy(enemy3X - 1, enemy3Y + 1) == '>' || getCharAtxy(enemy3X - 1, enemy3Y + 2) == '>')

{

playerHealth -= 1;

}

}

if (enemy3direction == "diagonalDown")

{

if (getCharAtxy(enemy3X - 1, enemy3Y + 1) == ' ' && getCharAtxy(enemy3X - 1, enemy3Y + 2) == ' ' && getCharAtxy(enemy3X - 1, enemy3Y + 3) == ' ' && getCharAtxy(enemy3X, enemy3Y + 3) == ' ' && getCharAtxy(enemy3X + 1, enemy3Y + 3) == ' ' && getCharAtxy(enemy3X + 2, enemy3Y + 3) == ' ')

{

enemy3X -= 1;

enemy3Y += 1;

}

if ((getCharAtxy(enemy3X - 1, enemy3Y) == '>' || getCharAtxy(enemy3X - 1, enemy3Y + 1) == '>' || getCharAtxy(enemy3X - 1, enemy3Y + 2) == '>' || getCharAtxy(enemy3X - 1, enemy3Y + 3) == '>') || (getCharAtxy(enemy3X - 1, enemy3Y) == '|' || getCharAtxy(enemy3X - 1, enemy3Y + 1) == '|' || getCharAtxy(enemy3X - 1, enemy3Y + 2) == '|' || getCharAtxy(enemy3X - 1, enemy3Y + 3) == '|'))

{

playerHealth -= 1;

}

if ((getCharAtxy(enemy3X, enemy3Y + 3) == '<' || getCharAtxy(enemy3X + 1, enemy3Y + 3) == '<' || getCharAtxy(enemy3X + 2, enemy3Y + 3) == '<') || (getCharAtxy(enemy3X, enemy3Y + 3) == '|' || getCharAtxy(enemy3X + 1, enemy3Y + 3) == '|' || getCharAtxy(enemy3X + 2, enemy3Y + 3) == '|') || (getCharAtxy(enemy3X, enemy3Y + 3) == '>' || getCharAtxy(enemy3X + 1, enemy3Y + 3) == '>' || getCharAtxy(enemy3X + 2, enemy3Y + 3) == '>'))

{

playerHealth -= 1;

}

}

printEnemy3();

}

void menu()

{

setcolor(14);

int x = 40, y = 1;

gotoxy(x, y);

cout << " ====== ++++++";

gotoxy(x, y += 1);

cout << " === ++";

gotoxy(x, y += 1);

cout << " == ++";

gotoxy(x, y += 1);

cout << "= ONTRA ++ HOOTER";

gotoxy(x, y += 1);

cout << " == ++";

gotoxy(x, y += 1);

cout << " === ++";

gotoxy(x, y += 1);

cout << " ====== +++++++";

}

void gameOver()

{

setcolor(4);

gotoxy(43, 9);

cout << "\*\*\*\*\* Game Over \*\*\*\*\*";

}

void wonTHEgame()

{

setcolor(9);

gotoxy(43, 9);

cout << "\*\*\*\*\* You Won The Game \*\*\*\*\*";

}

void pressTOgoBACK()

{

setcolor(7);

gotoxy(45, 30);

cout << "Press any key to go back...";

getch();

main();

}

void box()

{

setcolor(10);

int a = 20, b = 10;

gotoxy(a, b);

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

gotoxy(a, b += 1);

cout << "| |";

gotoxy(a, b += 1);

cout << "| |";

gotoxy(a, b += 1);

cout << "| |";

gotoxy(a, b += 1);

cout << "| |";

gotoxy(a, b += 1);

cout << "| |";

gotoxy(a, b += 1);

cout << "| |";

gotoxy(a, b += 1);

cout << "| |";

gotoxy(a, b += 1);

cout << "| |";

gotoxy(a, b += 1);

cout << "| |";

gotoxy(a, b += 1);

cout << "| |";

gotoxy(a, b += 1);

cout << "| I=====I |";

gotoxy(a, b += 1);

cout << "| I I |";

gotoxy(a, b += 1);

cout << "| I 0 I I |";

gotoxy(a, b += 1);

cout << "| I I I |";

gotoxy(a, b += 1);

cout << "| I |";

gotoxy(a, b += 1);

cout << "| 0 I |";

gotoxy(a, b += 1);

cout << "| :: =======I |";

gotoxy(a, b += 1);

cout << "| :: :: |";

gotoxy(a, b += 1);

cout << "| ::: ::: |";

gotoxy(a, b += 1);

cout << "| 0 :::: 0 :::: 0 |";

gotoxy(a, b += 1);

cout << "|========== ========= |";

gotoxy(a, b += 1);

cout << "| I I |";

gotoxy(a, b += 1);

cout << "| I 0 I |";

gotoxy(a, b += 1);

cout << "| 0 I-----I |";

gotoxy(a, b += 1);

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

}

string setcolor(unsigned short color)

{

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

SetConsoleTextAttribute(hcon, color);

return "";

}

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 : ' ';

}

void gotoxy(int x, int y)

{

COORD coordinates;

coordinates.X = x;

coordinates.Y = y;

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

}