A Mini Project Report

On

**“Oil Spill Game”**

Submitted in partial fulfilment of the requirements for the award of degree of

**Bachelor of Engineering**

in

**Computer Science and Engineering**

Submitted by

Rohith Innamuri 1608-16-733-096  
 Sai Krishna Prasad Kammadanam 1608-16-733-108

Neelima Donkada 1608-16-733-099

Under the Guidance of

Dr. P. Vijayapal Reddy

Professor & Head, Department of CSE



**Department of Computer Science and Engineering**

**Matrusri Engineering College**

(Affiliated to Osmania University, Approved by AICTE)

Saidabad, Hyderabad - 500059

(2017-2018)

**Department of Computer Science and Engineering**

**Matrusri Engineering College**

(Affiliated to Osmania University, Approved by AICTE)

Saidabad, Hyderabad - 500059



CERTIFICATE

This is to Certify that A Mini Project report entitled **“Oil Spill Game**” is being submitted by Rohith Innamuri**(**1608-16-733-096**),** K. Sai Krishna Prasad**(**1608-16-733-108**),** Neelima Donkada(1608-16-733-099) in partial fulfilment of the requirement of the award for the degree of Bachelor of Engineering in “Computer Science and Engineering” O.U., Hyderabad during the year 2017-2018 is a record of bonafide work carried out by them under my guidance. The results presented in this mini project report have been verified and are found to be satisfactory.

**Mrs. K. Shalini Dr.P.Vijayapal Reddy Dr. P. Vijayapal Reddy**

**Assistant Professor Professor & Head Professor & Head**

**Dept. of CSE Dept. of CSE Dept. of CSE**

Project Coordinator Project Guide HOD

**External Examiner(s)**

**ACKNOWLEDGEMENT**

It is our privilege and pleasure to express our profound sense of respect, gratitude and indebtedness to our guide **Dr.P.Vijayapal Reddy**, Professor and Head, Department of Computer Science and Engineering, Matrusri Engineering College, for his indefatigable inspiration, guidance, cogent discussion, constructive criticisms and encouragement throughout this dissertation work.

We express our sincere thanks to mini project coordinator **Mrs. K. Shalini**, Assistant professor, Department of Computer Science and Engineering, Matrusri Engineering College, for her valuable suggestions and constant help in completing the work.

We extend our sincere thanks to **Dr. D. Hanumantha Rao**, Principal, Matrusri Engineering College, Saidabad, Hyderabad, for his encouragement and constant help.

We extend our sincere thanks to all the teaching and non-teaching staff of CSE Department for their support and encouragement.

Last but not least, we wish to acknowledge our friends and family members for giving moral strength and helping us to complete this dissertation.

**CONTENTS**

Abstract v

List of Figures vi

**S.No Chapter Page No**

1. **Chapter 1 - Introduction 1**
   1. Advantages 1
   2. Disadvantages 2
   3. Objectives 2
   4. Scope 3
   5. System Requirements 3
   6. Technology Used 3

1.6.1 Concepts Used 5

1. **Chapter 2 - Architecture & Implementation 8**
   1. Modules 8
   2. Code 10
2. **Chapter 3 - Output Screenshots 18**
3. **Chapter 4 - Conclusion 22**
4. **Chapter 5 - Future Enhancement 23**
5. **Chapter 6 - References 24**

**ABSTRACT**

The Oil Spill Game is our first step into game designing and developing. A console application is a computer program designed to be used via a text-only computer interface, such as a text terminal, the command line interface of some operating systems (Unix, DOS, etc.) or the text-based interface included with most Graphical User Interface (GUI) operating systems, such as the Win32 console. The required software and hardware are easily available and easy available and easy to work with.

Player will be given a container with only two ways to move, either left or right. Oil drops will be dropping from the top of the screen and player needs to catch the drops. Player will be given 5 lives. If the player misses a drop his lives will be decrease by 1. Game will end once he runs out of lives.

**LIST OF FIGURES**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Fig No.** | **Name of the Figure** | **Page No** |
| **1** | **3.1** | **Basic Output** | **18** |
| **2** | **3.2** | **Scoring** | **19** |
| **3** | **3.3** | **Loosing Lives** | **20** |
| **4** | **3.4** | **Game Over** | **21** |

**Chapter 1**

**Introduction**

The “Oil Spill Game” has been developed by using C++. Video game development is the process of creating a video game. Development is undertaken by a game developer. Research to date suggests that playing games can change the brain regions responsible for attention and visuospatial skills and make them more efficient. The researchers also looked at studies exploring brain regions associated with the reward system, and how these are related to game addiction.

**1.1 Advantages**

* + - Increases A Child's Memory Capacity.
    - Computer & Simulation Fluency.
    - Helps with Fast Strategic Thinking & Problem-Solving.
    - Develops Hand-Eye Coordination.
    - Few console games can be played in PC’s also.
    - Beneficial Specifically for Children with Attention Disorders.
    - Skill-Building (e.g. map reading)
    - Controller Standardization
    - A console lasts for 5 years without any damage so games are safe.

* 1. **Disadvantages**

* In console games we cannot update games by using graphics.
* Outdated quickly and higher priced games
* Some genres are completely absent for consoles.
* If a person plays for long time it effects his posture and body pains occur
* Playing for long hours cause eye site and it also creates tendency to skip meals
* It can also causes headache.
  1. **Objective**

The main objective of the Project on Oil Spill Game is to put our coding skills to practical use. Our game is about catching oil drops which are falling from top. Player need to catch them by moving the container. Person can catch until he runs out of lives.

In other words we can say that our project has following objectives:-

* Improves hand-eye coordination
* Improves problem-solving skills.
* Enhances memory
* Improves attention and concentration.
* It is a great source of learning
* Gaming is not only beneficial to adults and teenagers, but to children as well.
* Improves the brain's speed
* Enhances multitasking skills
  1. **Scope**

Game testing plays a crucial role in the development of new game. Game testers put games through the paces while still in development and when finished, to ensure gamers have a good experience. Game testers conduct game QA, or quality assurance, finding mistakes, bugs and other problems that could annoy or turn off the gaming community if they are not fixed.

* 1. **System requirements**
* Hardware requirements

Processor: Intel® Core™ i5-4210 CPU@1.70GHz 2.40Ghz

System type: 64-bit, x64-based processor

RAM: 2GB/4GB/8GB

* Software requirements

Operating System: Windows 7,8,10

IDE: Code Blocks 17.12

Atom Text Editor

* 1. **Technology Used**

This project “**OIL SPILL GAME”** is developed using the concepts of C++ programming language. C++ is a general purpose language that supports object oriented, procedural and generic programming. It is a middle-level language, as it encapsulates both high and low level language features.

C++ supports the object-oriented programming, the four major pillar of object oriented programming used in C++ are:

* Inheritance.
* Polymorphism.
* Encapsulation.
* Abstraction.

These are some features of C++ which makes it a user friendly:

* Simple.
* Machine Independent or Portable.
* Mid-level programming language.
* Structured programming language.
* Rich library.
* Memory Management.
* Fast Speed.
* Pointers.
* Recursion.
* Extensible.
* Object oriented.

The usage of C++:

* Window application.
* Client-Server application.
* Device drives.
* Embedded firmware etc.

The system console, computer console, root console, [operator](https://en.wikipedia.org/wiki/Computer_operator)'s console, or simply console is the text entry and display device for system administration messages, particularly those from the [BIOS](https://en.wikipedia.org/wiki/BIOS) or [boot loader](https://en.wikipedia.org/wiki/Boot_loader), the [kernel](https://en.wikipedia.org/wiki/Kernel_(computer_science)), from the [init](https://en.wikipedia.org/wiki/Init) system and from the [system logger](https://en.wikipedia.org/wiki/Syslog). It is a physical device consisting of a keyboard and a screen, and traditionally is a [text terminal](https://en.wikipedia.org/wiki/Text_terminal), but may also be a [graphical terminal](https://en.wikipedia.org/wiki/Graphical_terminal). System consoles are generalized to [computer terminals](https://en.wikipedia.org/wiki/Computer_terminal), which are abstracted respectively by [virtual consoles](https://en.wikipedia.org/wiki/Virtual_console) and [terminal emulators](https://en.wikipedia.org/wiki/Terminal_emulator). Today communication with system consoles is generally done abstractly, via the [standard streams](https://en.wikipedia.org/wiki/Standard_streams) ([stdin](https://en.wikipedia.org/wiki/Stdin), [stdout](https://en.wikipedia.org/wiki/Stdout), and [stderr](https://en.wikipedia.org/wiki/Stderr)), but there may be system-specific interfaces. Here we have used system console in our project.

.

A console application is a computer program designed to be used via a text-only computer interface, such as a text terminal, the command line interface of some operating systems (Unix, DOS, etc.) or the text-based interface included with most Graphical User Interface (GUI) operating systems, such as the Win32 console. The required software and hardware are easily available and easy available and easy to work with.

* + 1. **Concepts Used**

The following concepts were used for the execution of this project.

* **Inheritance**

Inheritance in Object Oriented Programming can be described as a process of creating new classes from existing classes.

New classes inherit some of the properties and behavior of the existing classes. An existing class that is "parent" of a new class is called a **base class**. New class that inherits properties of the base class is called a **derived class**.

Inheritance is a technique of code reuse. It also provides possibility to extend existing classes by creating derived classes.

* **Multiple Inheritance**

Multiple inheritance represents a kind of inheritance when a derived class inherits properties of **multiple** classes.

**Syntax:**  
Class DerivedClass: accessSpecifier BaseClass1, BaseClass2, …, BaseClassN

* **Pure Virtual Function**

A **pure virtual function** or **pure virtual method** is a virtual function that is required to be implemented by a derived class if the derived class is not abstract. Classes containing pure virtual methods are termed "abstract" and they cannot be instantiated directly. A subclass of an abstract class can only be instantiated directly if all inherited pure virtual methods have been implemented by that class or a parent class. Pure virtual methods typically have a declaration and no definition.

* **Control Structures**

A program is usually not limited to a linear sequence of instructions. During its process it may bifurcate, repeat code or take decisions. For that purpose, C++ provides control structures that serve to specify what has to be done by our program, when and under which circumstances.

* **Conditional Structures**
* **if and else**

The if else executes the codes inside the body of if statement if the test expression is true and skips the codes inside the body of else.

If the test expression is false, it executes the codes inside the body of else statement and skips the codes inside the body of if.

* **Nested if…else**

The if…else statement executes two different codes depending upon whether the test expression is true or false. Sometimes, a choice has to be made from more than 2 possibilities.

The nested if…else statement allows you to check for multiple test expressions and execute different codes for more than two conditions.

* **Iteration Structures**

Loops have a purpose to repeat a statement a certain number of times or while a condition is fulfilled.

* **While loop**

Its functionality is simply to repeat statement while the condition set in expression is true.

**Syntax:**

while(expression)

blocks of statements;

* **For loop**

The for loop is most convenient with counting loops -- i.e. loops that are based on a counting variable, usually a known number of iterations.

**Syntax:**

for (initialCondition; testExpression; iterativeStatement) {  
 statement1;

statement2;

// ...

statementN;   
}

The initialCondition runs once, at the start of the loop  
 The testExpression is checked. (This is just like the expression in a while loop). If it's false, quit. If it's true, then:  
 1. Run the loop body.  
 2. Run the iterativeStatement.  
 3. Go back to the testExpression step and repeat.

* **Console API Functions**

Console API is used to write information to the console.

* **SetConsoleCursorPosition function**

This function sets the cursor position in the specified console screen buffer.

**Chapter 2**

**Architecture and Implementation**

**2.1 Modules**

Following are the modules used in this project.

* **GameObject**

This class has a pure virtual function **draw()** which is used in classes “**container**” and “**drop**”.

* **container**This class contains four functions which takes care of the container. This class inherits the class **GameObject**.
* **ocontainer()**

This method is a part of Container class which takes arguments as x and y co-ordinates and symbol of container.

* **draw()**

This method is the pure virtual function initialized in **GameObject** class which prints the container symbol at the co-ordinates specified by **ocontainer()** method.

* **movement()**

This method takes the input from the keyboard given by the user and sends the input to **bounce()** method.

* **bounce()**

This method takes the input provided by **movement()** method and increases/decreases the co-ordinates accordingly which thereby providing movement of the container.If the container hits the boundary it bounces back and goes in opposite direction.

* **drop**

This class has three functions which takes care of the drop. This class inherits both GameObject and container.

* **odrop()**

This method takes y co-ordinate and symbol of drop as input.

* **movement()**

This method handles the movement of the drop and also monitors lives which in turn contributes to increment and decrement of score and also to restart the game by user’s choice once it ends.

* **draw()**

This method prints the leaderboards.

* **main()**

This method coordinates the order of execution of the other methods and this method has the main game loop. This method also takes care of positioning of the oil drops.

**2.2 Code**

#include<iostream>

#include<conio.h>

#include"windows.h"

#define RIGHT 0

#define LEFT 1

#define number 10

using namespace std;

bool running;

int score = 0;

int lives = 5;

int direction = 1;

int currX;//Updated X-Co-ordinates of container

/\*This function is used to point the cursor in console at the location of our choice\*/

void gotoxy(int x, int y)

{

COORD coord;

coord.X = x;

coord.Y = y;

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

}

class GameObject

{

public:

virtual void draw() = 0;

};

class container : public GameObject

{

public:

int containerX;

int containerY;

char container\_symbol;

void ocontainer(char csymbol, int cx, int cy)

{

container\_symbol = csymbol;

containerX = cx;

containerY = cy;

}

/\*This function is to draw the container\*/

void draw()

{

for (int z = 0; z < 8; z++)

{

gotoxy(containerX+z,containerY );

cout<<container\_symbol;

}

}

/\*This function takes the input from the keyboard\*/

void movement()

{

if (\_kbhit())

{

char ch = \_getch();

switch (ch)

{

case 'a':

direction = RIGHT;

break;

case 'd':

direction = LEFT;

break;

case 'e':

running=false;

exit(0);

break;

case 'A':

direction = RIGHT;

break;

case 'D':

direction = LEFT;

break;

case 'E':

running=false;

exit(0);

break;

}

}

}

/\*This method takes care of the movemet of the container and bounces back the container when it hits the boundary\*/

void bounce()

{

switch (direction)

{

case RIGHT: containerX--;

currX=containerX;

break;

case LEFT: containerX++;

currX=containerX;

break;

}

if (containerX >= 60)

{

direction = RIGHT;

}

if (containerX<= 10)

{

direction = LEFT;

}

}

};

class drop :public GameObject, public container

{

public:

int isalive=0;//Reset's drop's location, updates it's movement and displays scoreboard

int dropX;//X- Co-ordinate of drop

int dropY;//Y- Co-ordinate of drop

char drop\_symbol;

void odrop(char dsymbol,int dy)

{

drop\_symbol = dsymbol;

dropY = dy;

}

/\*This function takes care of the movement of the drop, increment and decrement of score and lives respectively, restarting of game when it ends by player's choice\*/

void movement()

{

dropY++;

if((dropY==23)&&(dropX ==currX||dropX == currX+1||dropX == currX+2||dropX == currX+3||dropX == currX+4||dropX == currX+5||dropX == currX+6||dropX == currX+7))

{

dropY=0;

score=score+50;

isalive=0;

}

if (dropY==25)

{

--lives;

dropY=0;

isalive=0;

if(lives==0)

{   
 char option;

gotoxy(0,4);

cout<<"Lives left:"<<"0/5";

running=false;

gotoxy(0,5);

cout<<"Score:"<<score;

gotoxy(0,6);

cout<<"!!!GAME OVER!!!";

gotoxy(0,7);

cout<<"DO YOU WANT TO PLAY AGAIN??(Y/N)";

cin>>option;

if(option=='Y'||option=='y')

{

running=true;

lives=5;

score=0;

int main();

}

else

exit(0);

}

}

}

/\*This displays the leaderboard\*/

void draw()

{

{

gotoxy(dropX, dropY);

cout << drop\_symbol;

gotoxy(0,1);

cout<<"OIL SPILL GAME";

gotoxy(0,2);

cout<<"Controls:";

gotoxy(0,3);

cout<<"A-Left,D-Right,E-Quit";

gotoxy(0,4);

cout<<"Lives left:"<<lives<<"/5";

gotoxy(0,5);

cout<<"Score:"<<score;

/\* These statements are used to check the co-ordinates of drop and container (For Deveoper's use only)

gotoxy(0,11);

cout<<"Current containerX is "<<currX;

gotoxy(0,12);

cout<<"Current container Y is"<<containerY;

gotoxy(0,13);

cout<<"Current Drop y is:"<<dropY;

gotoxy(0,14);

cout<<"Drop X co-ordinate is "<<dropX;\*/

}

}

};

/\*This function is used to clear the console screen\*/

void clrscr()

{

system("cls");

}

int main()

{

container c;

drop d;

drop \*drops[number];//pointer object to use it in for loop of main game

c.ocontainer('X', 38, 23);//sends arguements to ocontainer

for(int y = 0; y<number; y++)

{

drops[y] = &d;//pointer object holds the address of object

d.odrop('0',0);

}

/\*This is the main game loop\*/

while (running=true)

{

clrscr();

for (int z = 0; z < number;z++)

{

if (drops[z]->isalive ==0)

{

drops[z]->dropX = rand() %40+20;// location of the drop, updates after each loop

drops[z]->isalive = 1;

}

if(drops[z]->isalive !=0)

{

drops[z]->movement();

drops[z]->draw();

break;

}

}

c.draw();

if(kbhit())c.movement();

c.bounce();

Sleep(100);//to suspend the execution of a program(makes the game slower i.e playable)

}

return 0;

}

**Chapter 3**

**Output Screenshots**

****

Fig 3.1 Basic Output



Fig 3.2 Scoring

****

Fig 3.3 Loosing lives

****

Fig 3.4 Game Over

**Chapter 4**

**Conclusion**

Our project is only a humble venture to satisfy the needs to manage the project work. Several user friendly coding have also adapted. When games are played in moderation and with mindfulness, they are a viable source of stress relief as well as a catalyst for mental health improvement and development of social skills The objective is to improve the reflexes of the person playing games.

**At the end it is concluded that we have made effort on following points….**

* A description of the background and context of the project.
* Made statement of the aims and objectives of the project.
* We had made efforts to make a container.
* We had made efforts to create drop and to move it faster.
* We described the requirement specification of the system and the actions that can be done on these things.
* We understood the problem domain and implemented code to overcome the problem.
* We included features and operations in detail, including screen layouts.
* Finally code is implemented and game is ready for playing.

**Chapter 5**

**Future Enhancements**

* + - This project can be implemented by using graphics.
    - Projects can be updated and reprogrammed to be released as an Android application.
    - A better and faster algorithm can be used for drops.
    - Multiple levels can be implemented.
    - This game can be online game also.
    - A new set of controls can be used for the movement of “Container”.
    - e.g: Mouse motion control, joystick.

**Chapter 6**

**References**

**World Wide Web**

[1] GitHub:

<https://githhub.com/>

[2] Dzone:

<https://dzone.com/>

[3] Stackoverflow:

<https://stackoverflow.com/>

[4] Cppreference:

<https://en.cppreference.com/w/>