CS 101 FINAL PROJECT REPORT

Lab Batch – 251 (Tuesday)

Mayuresh Patole 11D170012

Raanju Ragavendar.S 115280029 (Team Leader)

Siddardha P.S.M 110010042

N.Manoj Reddy 110010049

Ratan Vaibhav 110040102

Nikhil Sinha 115320012

Mumtaz 115090009

SPOT AND HIT

CONTENTS :

🡪 INTRODUCTION

🡪DESIGN OF THE GAME

🡪DESCRIPTION OF THE GAME

🡪BRIEF ALGORITHM OF THE PROGRAM

🡪SOFTWARE INTERFACE REQUIREMENTS

🡪STATUS OF COMPLETION

🡪INPUT / OUTPUT RESULTS

🡪FUTURE DEVELOPMENTS

🡪CONTRIBUTIONS OF INDIVIDUALS

INTRODUCTION:

In 1984 ,a Japanese gaming company Nintendo Entertainment System (NES) successfully launched a game called Duck Hunt in which players use a gun to shoot ducks on the screen for points. The game was not initially reviewed often, but given mediocre critical praise and positive gamer reaction. However it later became an icon to the gaming world.

Spot and Hit can rightly be called as a much simpler version of Duck Hunt. The very idea of spotting an object on the screen and hitting it has been an adaption from Duck Hunt.

Spot and Hit is a game in which balls of different colour keep appearing on the screen at random locations in depleting intervals of time. Every particular ball carries a specific number of points. Clicks on few of the balls render negative points while a few of them raise the score. The player should also be careful and accurate with his clicks as 'aimless' clicks made on the screen would result in deductions of points. The objective of the player should be to score the maximum number of points within the given time frame. The game gets much more intriguing as the speed of appearance of the balls increases as the user keeps scoring. This game has been designed to test the speed, accuracy, hand-eye coordination and memory(to some extent) of the player.

Design Of the Game :

The game begins with the launch of the main window that comprises of 3 buttons(Bitmap images) namely “Start Game” , “Instructions” , “Exit”.

When the player clicks “Start” a message is displayed ,alerting the user to get ready for the game and a new window is launched where the game is to be played. The game now begins and the user plays this game with the mouse(through mouse clicks). As the score of the player increases, the speed at which the images appear also increases. The player should also be careful that for every click on any place other than balls, the player will be penalised with negative points.

When the “Instructions” option is clicked, a new window launches which displays the instructions of the game.

When the “Exit” option is selected, the main window is closed with a “Goodbye” message.

**Description Of Program :**

The entire program comprises of a series of functions which perform various operations .All of these functions are globally declared (i.e) outside the "ApiMain()" function. This program works in a 'chain' like manner where initially a function is invoked and this function in turn calls upon another function and this process goes on until the whole program comes to a proper halt.

List of Functions used in the program -  
  
**void initialize()-**   
  
This function gives an alert message(welcome message) and launches the main window. There are basically 4 bitmap images loaded in the main window(including the background image) namely Start Game, Instructions and Exit .The inputs from the user through mouseclicks are recorded and every time the mouse is clicked, the function Mouseclick() is called upon.

**int Mouseclick(const Position &p)**   
  
This function checks for the position of the mouseclick through a pointer passed as parameter. Depending upon the position of the click,various other functions are further invoked.   
  
**void instructions()-**   
  
This is one such a function invoked through a mouseclick.This functions launches a new window and the instructions of the game are displayed.

**void gameinit()-**   
  
This function launches the game window after displaying an alert message. The game begins and the timer is set for 60 seconds. Also another timer is used to call upon the function 'W2TimerEvent()' every 900 milli-seconds. This time frame however changes as the game progresses. The mouseclicks made by the player on the screen are recorded and scores are displayed on the terminal window after each mouseclick.After the 60 seconds is completed, a complete performance statistics of the player is displayed on the terminal window. As the player keeps scoring in the game,the speed at which the function

'W2TimerEvent() 'is invoked increases. This speed can reach a maximum of 450 milli-seconds.

**int Mouseclick1(const Position &p)-**  
This function checks for the position of the mouseclick through a pointer passed as parameter. Depending upon the position of the click, the score is appropriately chamged and displayed in the terminal window. This function also is responsible for the varied speeds at which the function 'W2TimerEvent()' is called upon..As the value of the variable 'count' increases,the speed at which the function 'W2TimerEvent()' is called upon also increases.

**int W2TimerEvent()-**   
  
This function is responsible for the random generation of images at any point of time during the game. It uses the rand() function and decides which of the images is to be loaded. ALl the 4 images thus have an equal chance of appearing on the screen at any time during the game.

**void display(SimpleWindow &W, BitMap &B)-**   
  
This function is responsible for the random generation of co-ordinates at which the images are to be displayed.

**int W2close()-**   
  
This function displays an alert message and closes the main window.

**Brief Algorithm Of the Program ::**

Step 0 --> Define the 3 Simple Windows W1,W2 and W3.

      -->Define the bitmaps WBmp,W1Bmp,W2Bmp,W3Bmp,W4Bmp,W5Bmp,W6Bmp,W7Bmp,W8Bmp,W9Bmp,W10Bmp.

      -->INITIALIZE GLOBAL VARIABLES count,good,bad,aimless and assign each of them to 0.   
  
Step 1 --> Inside ApiMain(), Call the function initialize()   
  
Step 2 --> Input from the user through MOUSE-CLICK. Every MOUSE CLICK calls the function Mouseclick() through

"SetMouseClickCallback(Mouseclick) ".   
        
      --> If Mouseclick is inside "START" bitmap --> Call the function   
 'gameinit()'.   
      --> Else if Mouseclick is inside "EXIT" bitmap -->Call the function   
 'close()'.   
      --> Else if Mouseclick is inside "INTSRUCTIONS" bitmap -->Call the   
 function 'instructions()'.

Step 3 --> Function gameinit()   
      --> Timer 'T1' is started. Function W2close() is called after 60 seconds.   
     --> Timer 'T2' on game window is started.

Function W2TimerEvent() is called every 900 milli-seconds initially.   
      --> (Game begins).Input from the user through MOUSE-CLICKS. Every MOUSE CLICK calls the function 'Mouseclick1()'.   
  
        
      --> Function Mouseclick1()   
            
           ~ if(W3Bmp.IsInside(p)) count=count+3;  good++;   
           ~ else if(W4Bmp.IsInside(p))   
             count=count-2;  bad++;   
  
           ~ else if(W5Bmp.IsInside(p))   
             count=count+1;     good++;   
  
           ~ else if(W6Bmp.IsInside(p))   
             count=count-1;    bad++;   
  
           ~ else count=count-1; aimless++;   
  
      --> ~ if(count>=2&&count<=4)   
            Stop the timer 'T2' and now call Function W2TimerEvent() is   
called every 850 milli-seconds.   
  
          ~ else if(count>=5&&count<=8   
            Stop the timer 'T2' and now call Function W2TimerEvent() is   
called every 750 milli-seconds.   
  
          ~  else if(count>8&&count<=11)   
             Stop the timer 'T2' and now call Function W2TimerEvent() is   
called every 650 milli-seconds.   
                
  
          ~  else if(count>11)   
             Stop the timer 'T2' and now call Function W2TimerEvent() is   
called every 450 milli-seconds.   
              
        
      --> Function W2TimerEvent()   
            
          Assign int x=rand()%4+1.   
            
          ~if(x==1)   
           Erase existing bitmaps.   
           display(W2,W3Bmp);   
  
          ~else if(x==2)   
           Erase existing bitmaps.   
           display(W2,W4Bmp);   
  
          ~else if(x==3)   
           Erase existing bitmaps.   
           display(W2,W5Bmp);   
            
          ~else if(x==4)   
           Erase existing bitmaps.   
           display(W2,W6Bmp);   
  
      --> Function display(SimpleWindow &W, BitMap &B)   
  
          Erase bitmap B.   
          Assign x=rand()%11+1.   
          Assign y=rand()%11+1.   
          Set the position to Position(x,y).   
          Draw the bitmap in this position.   
  
Step 4 --> Function instructions()   
      -->Display instructions through RenderText(...).   
  
Step 5 --> Function close()   
      --> Close the window W1.   
      --> Alert "Goodbye".   
      --> Output variables good,bad,aimless and count

Software Interface Requirements :

The program has been written basically in C++ language. In order to introduce graphics into the game, EzWindows API has been used in designing the game. In order to successfully run the program, the following are the minimum requirements ::

(a)Ubuntu (32-Bit)(version 11.1 is preferred).

(b)EzWindows installed with necessary(mentioned in the appendix) library files.(X11 library files have to updated and installed through the Synaptic Packet Manager on Ubuntu v11.1).

Status Of Completion :

The program is found to be satisfying all the design specifications and is functioning without any errors. The program has been tested with various inputs and the desired outputs have been recorded.

Input / Output Results :

***As this program is based entirely on MOUSE CLICKS made by the user, the program has to be tested for various inputs (MOUSECLICKS) given by the user on all the 3 “Simple Windows” in the game.***

***Firstly in the home page we have different images of “START” , “INSTRUCTIONS” AND “EXIT”.***

* ***If the mouseclick is done on the START IMAGE then it invokes a function and that opens a new window and gives you an instruction that game’s going to be started.*** 
  + ***If the game window is opened, then random balls of different colors appear on the screen , they are RED,BLUE,YELLOW AND BLACK.***
  + ***If the image of the blueball is hit then the score gets added by +2 points.***
  + ***If the red ball is hit then the score gets added by -2 points.***
  + ***If the yellow ball is hit then score gets added by 1 point.***
  + ***If the black ball is hit then score gets added by -1 point.***
  + ***And even the aimless hit , i.e. hitting in any other random position other than the balls is penalised by -1 point.***
* ***In the similar way , if the mouseclick is given on the INSTRUCTIONS , it opens a new window with instructions required for the gameplay.***
* ***If the EXIT is clicked , then the main window of the game closes and it displays a text GOODBYE.***
* ***And in the other case if it is clicked at any other place on the home window , it doesn’t invoke any of the function so it does nothing.***

Future Developments :

The project can be improved on the basis of customization and more user adjustable . The program can be easily so modified that the user can select the time for which one game runs . The difficulty level can also be user defined . The player can select the variety of colours of the marbles and similar customization options can be included . The main page can be so modified that it can include the ' options ' button . This will help the user customize the features specified above. Thus the project can be developed and morphed into a better game in future .

***CONTRIBUTIONS OF INDIVIDUALS***

***Raanju Ragavendar.S(Team Leader)***

* ***Contributed ideas towards the main framework of the game.***
* ***Contributed towards the formation of the algorithm of the game design.***
* ***Contributed towards the coding part of the game.***
* ***Compiled and debugged the final program after obtaining the group inputs.***
* ***Responsible for testing the final program.***
* ***Contributed towards preparation of the final project report and SRS Stage 1.***

***Mayuresh Patole:***

* ***Contributed towards the design and functioning of the home page.***
* ***Comntributed towards the design of INSTRUCTIONS page.***
* ***He contributed towards the documentation of SRS Stage1 and final Project report.***

***Siddardha.P.S.M.:***

* ***Contributed his ideas to the main logic of the game.***
* ***Worked for the main game coding along with Raanju Ragavendar.S.***
* ***Helped in the documentation of the project.***

***Mumtaz Hussain:***

* ***No contribution.***

***Manoj reddy:***

* ***Worked for the design and making of the Main page and Instructions page with Mayuresh , Ratan Vaibhav and Nikhil.***
* ***Worked for the making of SRS Stage 1 and Final Project Report.***

***Nikhil Sinha:***

* ***Contributed ideas for the main theme of the project.***
* ***He worked for the design and making of the Main page and Instructions page with Mayuresh and Ratan Vaibhav and Nikhil.***
* ***Contributed in preparation of Final Project report.***

***Ratan Vaibhav:***

* ***He made his contribution for the design and making of the homepage.***
* ***Contributed for the making of SRS Stage 1 and Final Project Report.***