**CG MINI PROJECT C-13**

**Title:Maze Runner**

**Name:Harsh Harwani-51,Jenil Jain-55,Vedant Jasani-59**

**Project Descrpition :**

The aim of the mini project is to implement the path finding game. The Path finding game containing a rectangular maze of any shape and size in which the horizontal and vertical lines represent the walls of the maze. Path finding game can be made difficult or easy, that is depend upon the maze. Maze can be implemented as in 2D, 3D or more higher dimensions. This game is very popular as puzzle solving. Hence it can be used as an effecting tool for logical reasoning and mental aptitude.

**Concepts Used:**

* Animation
* Flood Fill
* Looping
* Conditional Statememts

**Functions Used:**

1)**line()-line**() is a library **function** of **graphics**.h in c programming language which is used to draw a **line** from two coordinates. For example if you want to draw a **line** from point(x1,y1) to point(x2,y2) you have to use **line**() **function** like **line**(x1,y1,x2,y2);

2)**circle()-**The header file **graphics**.h contains **circle**() **function** which draws a **circle** with center at (x, y) and given radius. Syntax : **circle**(x, y, radius); where, (x, y) is center of the **circle**. 'radius' is the Radius of the **circle**.

3)**floodfill()-floodfill**() **function** fills an area with the desired color in all directions until it could see the border color.

4)**setfillstyle()-**The header file **graphics**.h contains **setfillstyle**() **function** which sets the current fill pattern and fill color. floodfill() **function** is used to fill an enclosed area.

5)**setcolor()**-The header file graphics.h contains setcolor() function which is used to set the current drawing color to the new color. **Syntax** : void setcolor(int color);

**Working:**

The main working of path finding game is to find out the path from given place to another place by using the movement of point. We use the special key button for the movement of point. The left key button is used to movement of point along  the X-axis as the value decreases, The right key button is used to movement of point along  the X-axis as the value increases, The up key button is used to movement of point along  the y-axis as the value increases, the down key button is used to movement of point along  the y-axis as the value decreases. There is also be given the time limitation, so it is necessary to find out the path within a given time interval.

**Implementation:**

#include <stdio.h>

#include <graphics.h>

#include <conio.h>

#include <stdlib.h>

int j, k, l;

void body() //figure

{

setcolor(3);

line(0, 100, 450, 100); //horizontal line

line(50, 130, 400, 130);

line(100, 160, 450, 160);

line(150, 190, 300, 190);

line(100, 220, 250, 220);

line(50, 250, 200, 250);

line(0, 400, 500, 400);

line(450, 370, 500, 370);

line(0, 100, 0, 400); //vertical line

line(50, 130, 50, 370);

line(100, 160, 100, 220);

line(100, 280, 100, 400);

line(150, 250, 150, 370);

line(200, 280, 200, 400);

line(250, 220, 250, 370);

line(300, 190, 300, 400);

line(350, 160, 350, 370);

line(400, 190, 400, 400);

line(450, 100, 450, 370);

setcolor(14);

circle(500, 385, 10);

setcolor(WHITE);

}

void ball(int j, int k, int l) //ball

{

circle(j, k, l);

setcolor(WHITE);

setfillstyle(SOLID\_FILL, 14);

floodfill(j, k, WHITE);

}

void condition()

{

if(k == 110)

{

if(j<450 && j>0)

k += 5;

}

if(j == 10)

{

if(k<400 && k>100)

j += 5;

}

if(k == 390)

{

if(j<500 && j>0)

k -= 5;

}

if(j == 440)

{

if(k>100 && k<380)

j -= 5;

}

if(k == 380)

{

if(j<500 && j>440)

k += 5;

}

if(k>=120 && k<=140)

{

if(j<410 && j>40 && k<=130)

k -= 5;

if(j<410 && j>40 && k>=130)

k += 5;

}

if(k>=150 && k<=170 && j>=90 && j<=450)

{

if(k <= 160)

k -= 10;

k += 5;

}

if(k>=180 && k<=200 && j>=140 && j<=310)

{

if(k >= 190)

k += 10;

k -= 5;

}

if(k>=210 && k<=230 && j>=90 && j<=260)

{ if(k >= 220)

k += 10;

k -= 5;

}

if(k<=260 && k>=240 && j<=210&& j>40)

{

if(k >= 250)

k += 10;

k -= 5;

}

if(k<=370 && k>=130 && j<=60 && j>=40)

{

if(j <= 50)

j -= 10;

j += 5;

}

if(k<=220 && k>=160 && j<=110 && j>=90)

{

if(j <= 100)

j -= 10;

j += 5;

}

if(k<=400 && k>=280 && j<=110 && j>=90)

{

if(j <= 100)

j -= 10;

j += 5;

}

if(k<=370 && k>=250 && j<=160 && j>=140)

{

if(j <= 150)

j -= 10;

j += 5;

}

if(k<=400 && k>=280 && j<=210 && j>=190)

{

if(j <= 200)

j -= 10;

j += 5;

}

if(k<=370 && k>=220 && j<=260 && j>=240)

{

if(j <= 250)

j -= 10;

j += 5;

}

if(k<=400 && k>=190 && j<=310 && j>=290)

{

if(j <= 300)

j -= 10;

j += 5;

}

if(k<=370 && k>=160 && j<=360 && j>=340)

{

if(j <= 350)

j -= 10;

j += 5;

}

if(k<=400 && k>=190 && j<=410 && j>=390)

{

if(j <= 400)

j -= 10;

j += 5;

}

}

void level\_1()

{

int i,m = 10;

j = 10;

k = 250;

l = 10;

char c;

cleardevice();

body(); //draw the figure

ball(j, k, l); //draw the ball

for(i = 0; i < m++; i++)

{

condition(); //cheacking condition

c = getch();

cleardevice();

if(c == 75 || c == 'a')

j = j - 5;

if(c == 80 || c == 's')

k = k + 5;

if(c == 77 || c == 'd')

j= j + 5;

if(c == 72 || c == 'w')

k = k - 5;

if(c == ' ')

return;

body(); //draw the figure

ball(j, k, l); //draw the ball

if(j >= 500)

return;

}

}

void main()

{

int gd = DETECT, gm;

int flag;

initgraph(&gd, &gm, "...\\BGI");

level\_1();

getch();

closegraph();}

**Output:**



