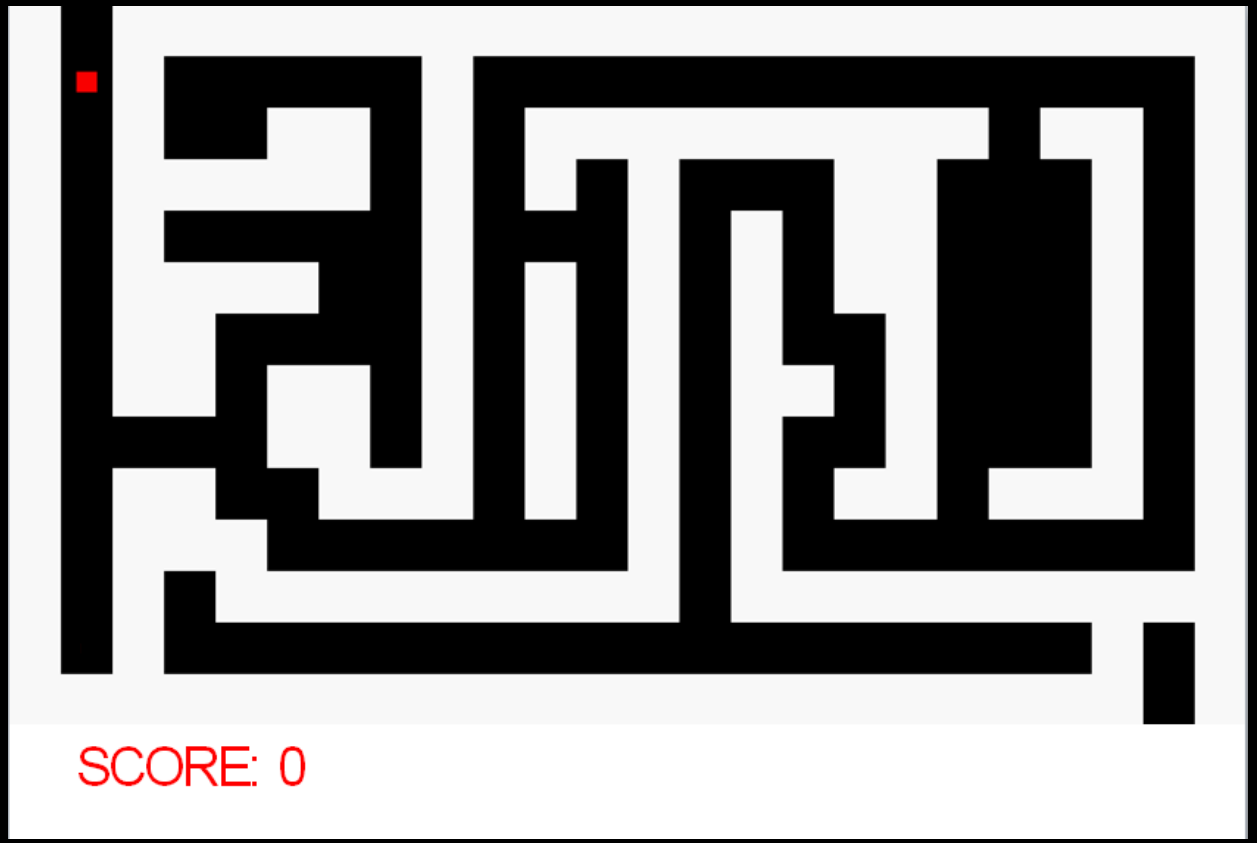
MY MAZE GAME

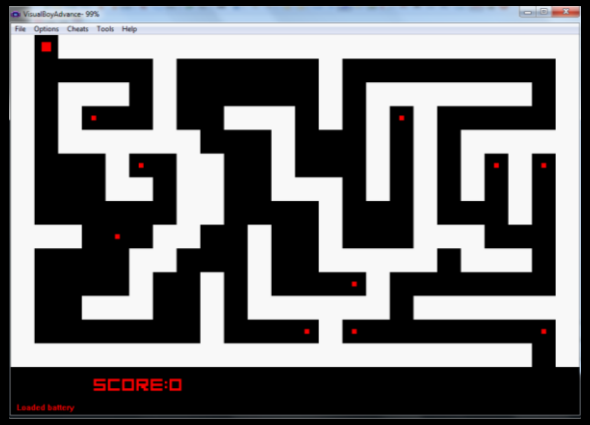
By Keiran Millar

STORYBOARD



BASIS OF MY GAME

In my game the objective is to get to the end of the maze. To do so you need to finish the 2 levels shown below



To be able to exit the second level you need to collect all of the red pellets then the door opens to allow you to leave the level

COLLISION DETECTION

This is how I managed collision detection in my game. When the user moves it checks to see if there is a wall where they will be moving to and if there is they will be moved back 4. Moving them back 4 makes it look like the player is bouncing off the wall so the player knows the can't go in that direction.

```
if ((currentKeys & KEY_UP) == 0)
{
    playerY--;
    if(map2[playerY/10][playerX/10] == 1 || map2[playerY/10][((playerX + 3)/10)] == 1)
    {
        playerY += 4;
    }
}
```

If the user goes off the screen my code sets them so that they do not move.

MAIN GAME LOOP

In my int main I have this small loop, it allows the game to go between level one and two easily, the reason i set the starting x and y to different numbers at the end is so when you go back to level 1 it allows you to start at the bottom left since you have come back from level 2

```
bool won = false;
int startingX = 13;
int startingY = 3;
while (true)
{
do
{
Level1(startingX, startingY);

won = Level2();
startingX = 223;
startingY = 124;
}while(won != true);
```

MOST DIFFICULT PART

The most difficult part of my game is drawing all the graphics, since I used bitmap mode 4 I had to figure out how to use Flip buffers and I had to think of different loops to draw my graphics.

This is how I drew my first level:

Where there is a 1 in my Map array I draw a 10x10 block so the map fills the screen

```
int map1[16][24]=
{
{1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1},
{1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1},
{1,0,1,1,1,0,1,0,0,0,0,0,1,0,1,1,1,1,1,1,1,0,1},
{1,0,1,2,0,0,1,0,0,1,1,1,0,1,0,1,2,1,0,0,0,0,0,1},
{1,0,1,1,1,1,1,1,1,0,0,0,1,0,0,0,1,0,1,0,1,1,1,1,1,1},
{1,0,0,0,1,2,0,1,1,0,0,1,0,0,0,1,0,1,0,1,2,1,2,1},
{1,0,0,0,1,1,0,1,1,0,0,1,1,1,0,1,0,1,0,1,0,1,0,1},
{1,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,0,1,0,0,0,1,0,1},
{1,1,1,1,0,2,0,1,1,0,0,1,0,0,1,1,0,0,1,1,1,0,0,0,1},
{1,0,0,0,0,1,1,0,0,0,1,0,0,1,1,1,1,1,1,0,1,1,1,0,1},
{1,0,0,0,0,1,0,0,1,0,1,0,0,0,2,1,0,0,0,0,0,0,0,1},
{1,0,0,1,1,1,0,0,1,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1},
{1,0,0,0,0,0,0,0,1,0,0,0,2,1,2,0,0,0,0,0,0,0,2,1},
{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1}};
```

```
void Draw_Map()
{
    for(int i = 0; i < 24; i++)
    {
        for(int j = 0; j < 16; j++)
        {
            if (map1[j][i] == 1)
            {
                Draw_Block((i*10),(j*10));
            }
            if (map1[j][i] == 2)
            {
                Draw_Pellet((i*10),(j*10));
            }
        }
    }
}
```

MOST DIFFICULT PART

How I drew my score display:

I had many arrays like the one on the left for numbers 1 to 9 and then I used a case statement seen below to draw the score required.

```
const int Num0[5][5]=
{
{1,1,1,1,1},
{1,0,0,0,1},
{1,0,0,0,1},
{1,0,0,0,1},
{1,1,1,1,1}
};
```

```
case 200 : for(int i = 0; i < 5; i++)
{
    for(int j = 0; j < 5 ; j++)
    {
        if(Num2[i][j] == 1)
        {
            PlotPixel8(67 + j, 145 + i, 1);
        }
        if(Num0[i][j] == 1)
        {
            PlotPixel8(73 + j, 145 + i, 1);
        }
        if(Num0[i][j] == 1)
        {
            PlotPixel8(79 + j, 145 + i, 1);
        }
    }
}; break;
```

IMPROVEMENTS

If I was to make improvements to my game I could add some more levels and I could try add enemies that would come after the user.