**REPORT OF**

**COMPUTER GRAPHICS PROJECT**

**ON THE TOPIC**

**MOVING BALL SCREENSAVER**



Prepared by:

Subheksha Adhikari

Sec:B Roll: 71

BSc.CSIT 3rd sem / 2nd year

Submitted to

Mr. Ramesh Kharbuja

Bhaktapur Multiple Campus

Date-2020 september 5

**ACKNOWLEDGEMENT**

I would like to sincerely express my thanks of gratitude to computer graphics sir Mr. Ramesh Kharbuja for providing me this opportunity to do project on “Moving Ball Screensaver”. I am also thankful for his guidance, support and valuable suggestions and coordination on making this report. It was indeed a great learning experience.

I am also grateful to my seniors, elders, and my friends for their direct or indirect support, and valuable suggestions.

Finally, I would also like to take this opportunity to acknowledge the contribution of all those who have published their reports and articles in related field in various media for the benefit of student like me.

Contents

[**1. INTRODUCTION** 3](#_Toc50226570)

[1.1. BACKGROUND 3](#_Toc50226571)

[1.2. STATEMENT OF PROBLEM 4](#_Toc50226572)

[1.3. OBJECTIVE OF STUDY 4](#_Toc50226573)

[**2. PROJECT METHODOLOGY** 5](#_Toc50226574)

[2.1. LITERATURE REVIEW 5](#_Toc50226575)

[2.2. ARCHITECTURE 5](#_Toc50226576)

[2.3. TOOLS & LANGUAGES 7](#_Toc50226577)

[**3. RESULTS & DISCUSSION** 9](#_Toc50226578)

[3.1. OVERVIEW 9](#_Toc50226579)

[3.2. IMPLEMENTATION 10](#_Toc50226580)

[**4. PROJECT OF SCHEDULE** 17](#_Toc50226581)

[**5. CONCLUSION & RECOMMENDATION** 18](#_Toc50226582)

[5.1. CONCLUSION 18](#_Toc50226583)

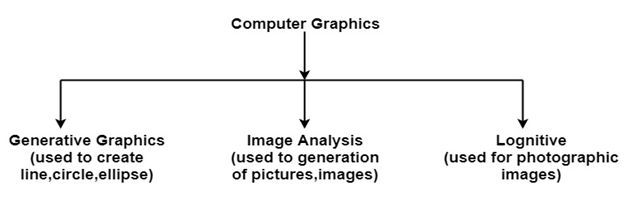
[5.2. LIMITATION OF STUDY 18](#_Toc50226584)

[5.3 FUTURE ENHANCEMENTS 18](#_Toc50226585)

[**References** 19](#_Toc50226586)

# **1. INTRODUCTION**

Computer Graphics refers to anything involved in creation, storage and manipulation of images of objects using computers. It is a very broad field, and one in which changes and advances seem to come at dizzying pace. These objects come from diverse fields such as physical, mathematical, engineering, architectural, abstract structures and natural phenomenon. Computer graphics is made up of number of pixels, where pixels is the smallest graphical picture or unit represented on computer screen.



## 1.1. BACKGROUND

Screensaver is an animate image that is activated on a personal computer display when no user activity has been sensed for a certain time. The original purpose of a screen saver was to prevent burn-in (the burning of an image into the phosphor inside the cathode ray tube after hours of the same image being displayed). In fact, today's CRT/LCD display technology makes burn-in unlikely except under extreme conditions. The prefect screensaver is the black screen. A good screensaver must have a dark background or scene or moving objects, slow motion, sensitive.

Moving ball screensaver is Sprite based screensaver, where sprite refers to animated or fixed objects which moves or is static on screen. The Sprite based screensaver is the most commonly used.

## 1.2. STATEMENT OF PROBLEM

During conduction of project, before time or after time some of the obstacles were generated. To state the fact of matter, I have clearly listed out as below:

* How to coordinate with pixels to use it effectively on graphics
* Improper arrangement of pixels affecting result of project

## 1.3. OBJECTIVE OF STUDY

Every study and research are done on the basis of its needs and objectives. With accomplishment of objectives we can get satisfactory results and conclusion after completion of the study or research.

The objective of mine conducted project are:

* To learn primitive and graphics library tools
* To manipulate and create images using pixels and make movements on the screen

# **2. PROJECT METHODOLOGY**

## 2.1. LITERATURE REVIEW

Reference to the source code, the program was written on C language and have used Turbo C++ as a compiler in order to compile and run it. It include very in-built functions, user-defined functions and statements according to requirement of logic. Editor has compile it, execute it, run it using Turbo C++ compiler.

Various graphical tools are used from header file graphics.h in order to give effectiveness in program. In-built functions like putpixel(), getimage() are used for manipulating pixels of the screen. The various graphic function are used for visualization and various instruction for motion animation of project.

## 2.2. ARCHITECTURE

The architecture is a partial abstraction of system in which particular model is created. It is obtain by using scale version-physical representation. It is built to communicate design ideas and to study aspects of system. Architecture modeling involves identifying the characteristics of the system and expressing it as models so that the system can be understood. Architecture models allow visualization of information about the system represented by the model.

The architecture model of the project is stated by giving clear visualization through diagram(flow-chart):

Draw ball at (x,y)

Checkout within range

YES NO

Add dx, dy to (x, y)

Gone to boundary range

NO

Negate dx & dy

## 2.3. TOOLS & LANGUAGES

A compiler is a special program that processes statements written in a particular programming language and turns them into machine language or "code" that a computer's [processor](https://whatis.techtarget.com/definition/processor) uses. Typically, a programmer writes language statements in a language such as [Pascal](https://whatis.techtarget.com/definition/Pascal) or [C](https://searchwindowsserver.techtarget.com/definition/C) one line at a time using an editor. Turbo C++ is a discontinued C++ compiler and integrated environment originally from Borland. It was designed as a home and hobbyist counterpart for Borland C++. In this project, Turbo C++ is the compiler I have used to run source code of mine project.

C is a procedural programming language. It was initially developed by Dennis Ritchie as a system programming language to write operating system. The main features of C language include low-level access to memory, simple set of keywords, and clean style, these features make C language suitable for system programming like operating system or compiler development.

Graphical tools can be perform in the compiler with the help of graphics.h header file. In this header file there possess all the graphical functions which helps in manipulation and creation. In mine program I have used following graphical tools which are described as below:

* Getmaxx():

The function which returns the maximum x-coordinate for current graphics mode and driver.

Syntax: int getmaxx();

* Getmaxy():

The function which returns the maximum y-coordinate for current graphics mode and driver.

Syntax: int getmaxy();

* Putpixel():

The function which plots a pixel at location (x, y) of specified color.

Syntax: void putpixel(int x, int y, int color);

where (x, y) is location at which pixel is to be put and color specifies color of pixel.

* Getimage():

The function which saves a bit image of specified region into memory, region can be any rectangle.

Syntax: void getimage(int left, int top, int right, int bottom, void \*bitmap);

Where getimage copies an image from screen to memory. Left, top, right, and bottom define the area of the screen from which the rectangle is to be copied, bitmap points to the area in memory where the bit image is stored.

* Putimage():

The function which outputs a bit image onto the screen.

Syntax: void putimage(int left, int top, void \*ptr, int op);

where putimage puts the bit image previously saved with getimage back onto the screen, with the upper left corner of the image placed at (left, top). ptr points to the area in memory where the source image is stored. The op argument specifies a operator that controls how the color for each destination pixel on screen is computed, based on pixel already on screen and the corresponding source pixel in memory.

# **3. RESULTS & DISCUSSION**

## 3.1. OVERVIEW

As the tittle of the project moving ball screensaver is designed to get movements of ball on the screen. Basically using single pixels on the screen I have generated a 3D ball image on the screen which was motion on created general path. There are multiple balls on the screen which was originated from random coordinates of the screen. And using algorithm and logic, certain paths were created so that we can see motion on screen.

During motion some of the balls gets strike to the boundary of screen and gets strike back as per code written in the program. Here the boundary screen was created using certain range of x and y coordinates of screen. When the ball strikes on boundary sound is produced and reflects backwards. Program created for this was on the concept of logics and algorithms. For this program I have used C language and some built-in and user-defined functions.

Fig: Image of ball created using pixel

Here using single pixels of screen with the help of putpixel() have created a ball. It was stored in certain memory space for further use. Then loop was created to increase the (x,y) coordinates until any key is pressed. During the continuation of previous loop, another loop helps to check if ball is within boundary. Until ball is within boundary it goes to previous loop otherwise program gets terminated. Here while there is strike on boundary there is sound effect. The termination of program is determined when (x,y) coordinates of ball is out of boundary range.

## 3.2. IMPLEMENTATION

Using C language below is the program for moving ball screen saver which was compile in turbo c++.

#include<iostream.h>

#include<graphics.h>

#include<dos.h>

#include<conio.h>

#include<stdlib.h>

#define DELAY 1

#define SOUND 3500

/\* 3d Ball \*/

char ball[20][20]=

{

{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0},

{0,0,0,0,0,0,12,12,12,12,12,12,12,12,0,0,0,0,0,0},

{0,0,0,0,12,12,12,12,12,12,12,12,12,12,12,12,0,0,0,0},

{0,0,0,12,12,12,12,12,12,12,12,12,12,12,12,12,12,0,0,0},

{0,0,12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,0,0},

{0,0,12,12,12,12,12,15,15,12,12,12,12,12,12,12,12,12,0,0},

{0,12,12,12,12,12,15,12,12,12,12,12,12,12,12,12,12,12,12,0},

{0,12,12,12,12,15,12,12,12,12,12,12,12,12,12,12,12,12,12,0},

{0,12,12,12,15,15,12,12,12,12,12,12,12,12,12,12,12,12,12,0},

{0,12,12,12,15,15,12,12,12,12,12,12,12,12,12,12,12,12,12,0},

{0,12,12,12,15,15,12,12,12,12,12,12,12,12,12,12,12,12,12,0},

{0,12,12,12,12,15,12,12,12,12,12,12,12,12,12,12,12,12,12,0},

{0,12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,0},

{0,12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,0},

{0,0,12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,0,0},

{0,0,12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,12,0,0},

{0,0,0,12,12,12,12,12,12,12,12,12,12,12,12,12,12,0,0,0},

{0,0,0,0,12,12,12,12,12,12,12,12,12,12,12,12,0,0,0,0},

{0,0,0,0,0,0,12,12,12,12,12,12,12,12,0,0,0,0,0,0},

{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}

};

void \*ptr1[4],\*ptr2[4];

/\* Condition to check out of range \*/

int COND(int x,int y)

{

if (x>10 && x<getmaxx()-10 && y>10 && y<getmaxy()-10)

return 1; //unsucessfull

else

return 0; //sucessfull

}

/\* This sub-routine will check which path to follow on stiking the wall

of

the container \*/

void state(int x,int y,int mode)

{

while(COND(x,y)&&!kbhit())

{

//lefttop

putimage(x+50,y+30,ptr1[4],COPY\_PUT);

putimage(x+70,y+50,ptr1[4],COPY\_PUT);

putimage(x+180,y+20,ptr1[4],COPY\_PUT);

putimage(x+200,y+170,ptr1[4],COPY\_PUT);

putimage(x+130,y+200,ptr1[4],COPY\_PUT);

//leftbottom

putimage(x,getmaxy()-y,ptr1[4],COPY\_PUT);

putimage(x,getmaxy()-y-250,ptr1[4],COPY\_PUT);

putimage(x+130,getmaxy()-y,ptr1[4],COPY\_PUT);

putimage(x+180,getmaxy()-y-170,ptr1[4],COPY\_PUT) ;

putimage(x+180,getmaxy()-y-140,ptr1[4],COPY\_PUT);

//righttop

putimage(getmaxx()-x,y+20,ptr1[4],COPY\_PUT);

putimage(getmaxx()-x,y+250,ptr1[4],COPY\_PUT);

putimage(getmaxx()-x-250,y+250,ptr1[4],COPY\_PUT);

putimage(getmaxx()-x-250,y+200,ptr1[4],COPY\_PUT);

putimage(getmaxx()-x-100,y+40,ptr1[4],COPY\_PUT);

//rightbottom

putimage(getmaxx()-x,getmaxy()-y,ptr1[4],COPY\_PUT);

putimage(getmaxx()-x-100,getmaxy()-y-230,ptr1[4],COPY\_PUT);

putimage(getmaxx()-x-200,getmaxy()-y-170,ptr1[4],COPY\_PUT);

putimage(getmaxx()-x-250,getmaxy()-y,ptr1[4],COPY\_PUT);

putimage(getmaxx()-x-150,getmaxy()-y-20,ptr1[4],COPY\_PUT);

//extra

putimage(x,y,ptr1[4],COPY\_PUT);

putimage(x,getmaxy(),ptr1[4],COPY\_PUT);

putimage(getmaxx(),y,ptr1[4],COPY\_PUT);

putimage(getmaxx(),getmaxy(),ptr1[4],COPY\_PUT);

switch(mode)

{

case 0:

x++;

y++;

break;

case 1:

x++;

y--;

break;

case 2:

x--;

y++;

break;

case 3:

x--;

y--;

break;

}

nosound();

}

cleardevice();

if(x>=(getmaxx()-10)||x<=10)

{

sound(SOUND);

switch(mode)

{

case 0:

state(--x,--y,2);

break;

case 1:

state(--x,++y,3);

break;

case 2:

state(++x,--y,0);

break;

case 3:

state(++x,++y,1);

break;

}

}

else

if(y>=getmaxy()-10||y<=10)

{

sound(SOUND);

switch(mode)

{

case 0:

state(--x,--y,1);

break;

case 1:

state(--x,++y,0);

break;

case 2:

state(++x,--y,3);

break;

case 3:

state(++x,++y,2);

break;

}

}

else

exit(0);

}

void main()

{

int gm,gd=DETECT;

int i,j;

initgraph(&gd,&gm,"C:\\turboc3\\bgi");

setbkcolor(WHITE);

for(i=0;i<20;i++)

for(j=0;j<20;j++)

if(ball[i][j]!='0')

putpixel(10+i,10+j,ball[j][i]);

getimage(10,10,30,30,ptr1[4]);

cleardevice();

getimage(10,10,30,30,ptr2[4]);

cleardevice();

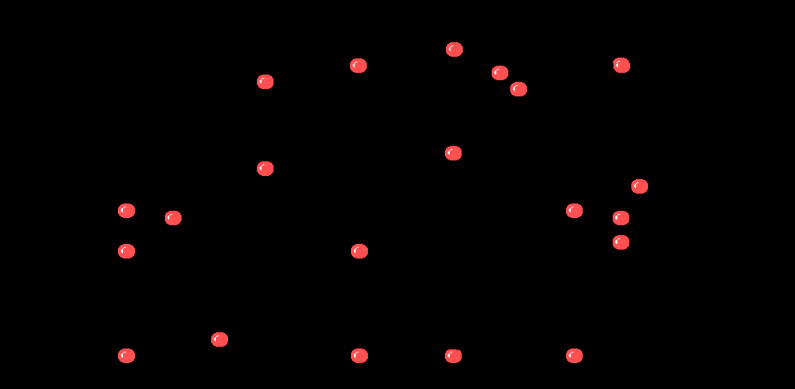
/\* start with (20,20) \*/

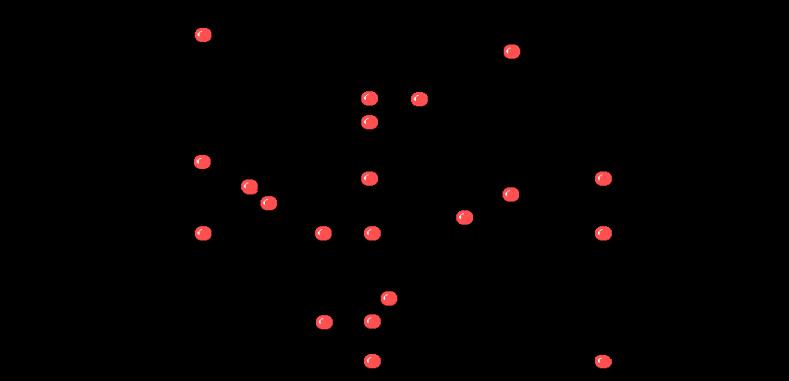
state(20,20,0);

getch();

}

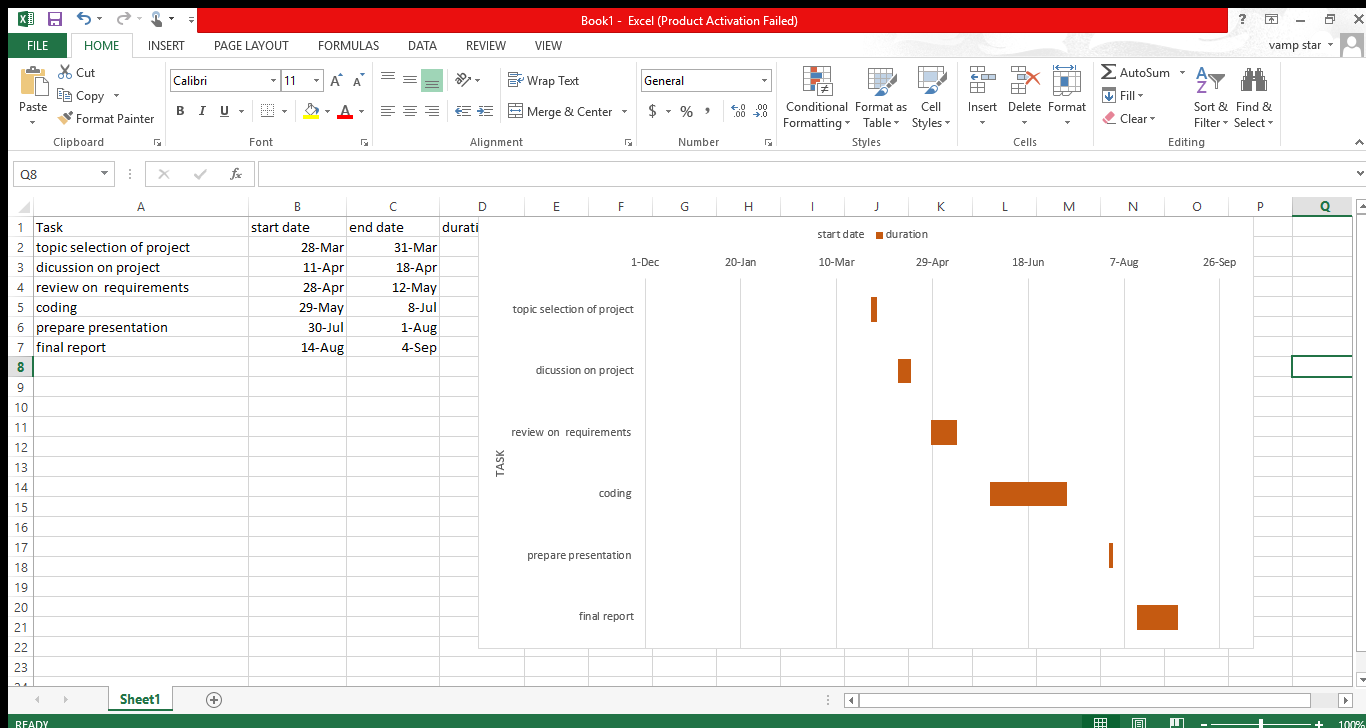
Output of above program:



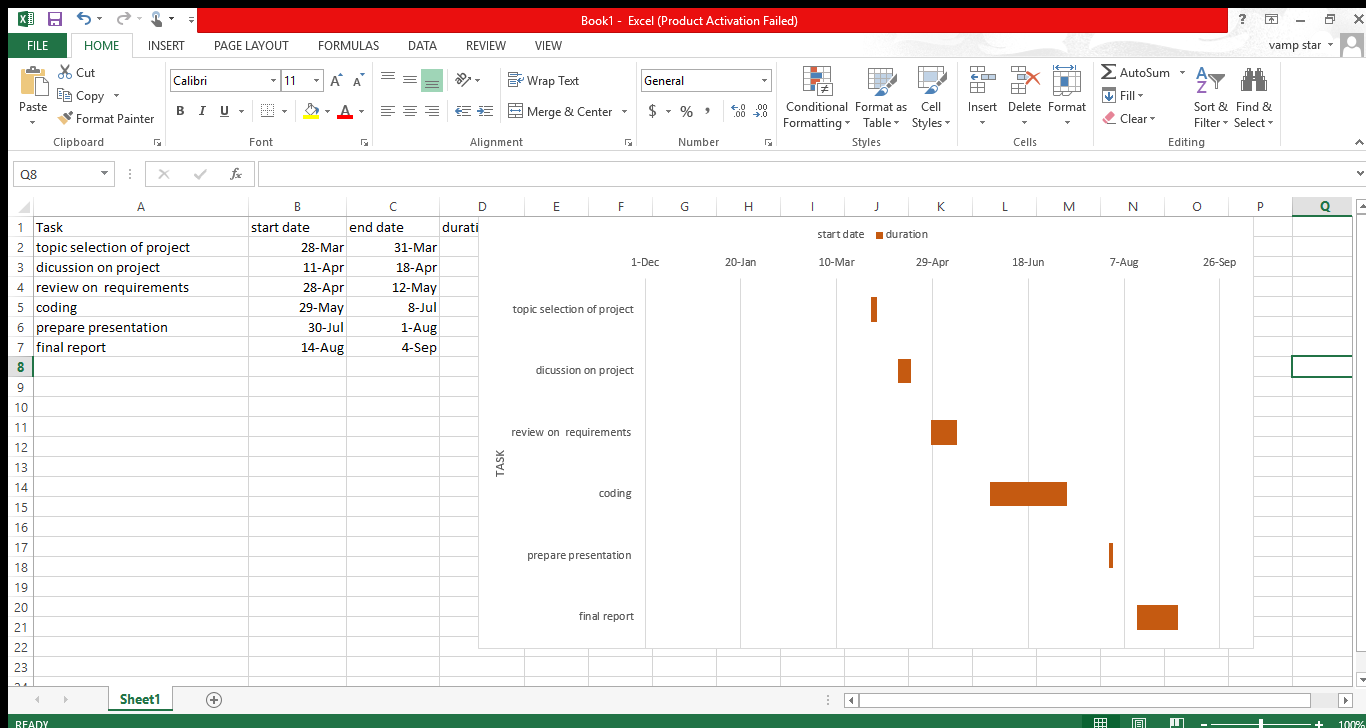


# **4. PROJECT OF SCHEDULE**

Time schedule:



Gantt chart:



# **5. CONCLUSION & RECOMMENDATION**

## 5.1. CONCLUSION

As per obtained result, using above implemented code it can be conclude as proper use of pixels manipulation and creation. Moving ball program has used primitive graphic tools to create efficiency. Using various header files and its in-built functions these project was completed. After completion of object, there is proper knowledge on computer graphics, flexibility on coding, and proper guidance to upcoming studies.

## 5.2. LIMITATION OF STUDY

During completion of this project, there were some limitation which were generated. These were some pointers which may have slightly affected in my project.

* Due to use of random coordinates some of objects get overlapped which visualized as missing on screen.
* During collisions of objects it was not highly sensitive which resulted in not striking back.

## 5.3 FUTURE ENHANCEMENTS

The limitations listed above can be overcome by enhancing it an advance or recruited way. So, with consideration to limitation we can solve and emphasize it for future enhancements.

* Instead of giving random coordinate we can make particular pathway for each objects on the screen.
* Proper sound effects and sensitive logic can get 100% efficiency on result.

# **References**

Books:

* Computer graphics C version by Hearn & Baker

Website:

* <http://www.sourcecodesworld.com/source/show.asp?ScriptID=1251>