Mechi Multiple Campus

(Tribhuvan University)

Bhadrapur, Jhapa

Lab Report of Computer Graphics & Animation (CACS-305)

Faculty of Humanities & Social Sciences

Tribhuvan University

Kritipur, Nepal

Submitted By

Name: Santosh Bhandari

Roll No: 58

Submitted To

Mechi Multiple Campus

Department of Bachelor in Computer

Bhadrapur, Jhapa, Nepal

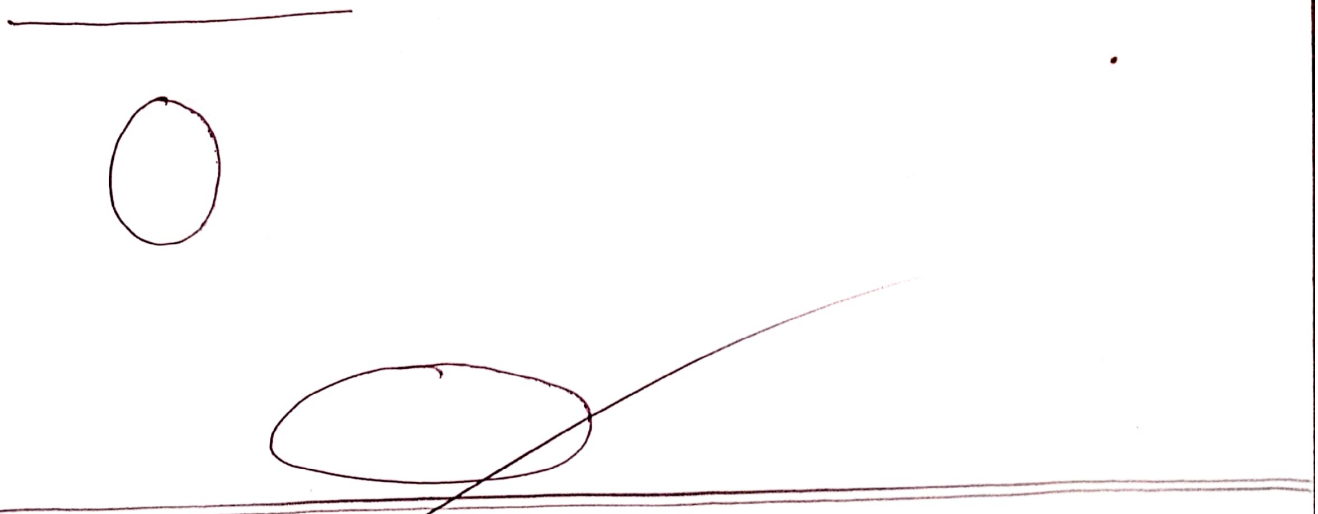
Table of Content

ND Kamble 10/16

1. WAP to draw different shapes like: line, circle, ellipse etc. using graphics function.

```
#include <stdio.h>
#include <conio.h>
#include <graphics.h>
void main(){
    int gd=DETECT,gm;
    initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");
    line(10,10,200,10);
    circle(100,200,50)
    ellipse(250,200,0,360,100,50);
    getch();
}
```

Output



2. WAP to implement ~~DDA~~ line Drawing Algorithm.

Algorithm

① Given a line with two end points (x_0, y_0) and (x_1, y_1)

② Calculate the difference between two end-points.

$$dx = x_1 - x_0$$
$$dy = y_1 - y_0$$

③ Calculate the Number of steps required
if $dx > dy$

```
        steps = |dx)
    else
        steps = |dy|
```

(4) Calculate the increment in x and y coordinate

```
xincrement = dx /(float) steps
yincrement = dy /(float) steps
```

(5) Put the pixel successfully incrementing x and y coordinate accordingly to draw a line

```
for(int v=0; v<steps; v++){
        x = x + xincrement;
        y = y + yincrement;
        putpixel (x,y);
}
```

Program

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<graphics.h>
void main(){
    int x,y,x0,y0,x1,y1,dx,dy,steps,i,gd=DETECT,gm;
    float xinc,yinc;
    initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");
    printf("Enter the line Endpoints:\n");
    scanf("%d%d%d%d",&x0,&y0,&x1,&y1);
    dx=x1-x0;
    dy=y1-y0;
    if(dx>dy)
        steps=abs(dx);
    else
        steps=abs(dy);
```

```
xinc = dx/(float) steps;
yinc = dy/(float) steps;
putpixel (x0, y0, WHITE);
x = x0;
y = y0;
for(i=0; i<steps; i++) {
        x = x+xinc;
        y = y+yinc;
        putpixel(x, y, WHITE);
        delay (100);
}
getch();
}
```

Output

Enter the line Endpoints:

100 100 200 100

---

3. WAP to implement Bresenham's line drawing algorithm.

Algorithm

① Given a line with two end-points (x_0, y_0) and (x_1, y_1)

② Calculate the value of initial decision parameter P_0 as

$$P_0 = 2 * dy - dx$$

Where $dy = y_1 - y_0$ and $dx = x_1 - x_0$

③ perform the following test, starting at $k = 0$

if $P_k < 0$

- the next point to plot is (x_{k+1}, y_k)
- the next decision parameter P_{k+1} is $P_{k+1} = P_k + 2x$

Otherwise
- the next point to plot is (x_{k+1}, y_{k+1})
- the value of next decision parameter p_{k+1} is

$$p_{k+1} = p_k + 2*dy - 2*dx$$

(4) Repeat Step (3) dx times

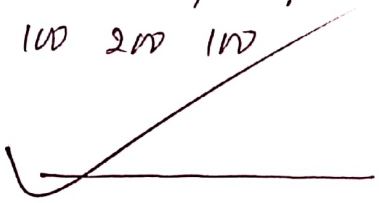
Program

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
void main(){
    int x0,y0,x1,y1,p,dx,dy,i,x,y,gd=DETECT,gm;
    initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");
    printf("Enter Line End points:\n");
    scanf("%d%d%d%d",&x0,&y0,&x1,&y1);
    dy=y1-y0;
    dx=x1-x0;
    x=x0;
    y=y0;
    putpixel(x,y,WHITE);
    p=2*dy-dx;
    for(i=0;i<dx;i++){
        if(p<0){
            x=x+1;
            putpixel(x,y,WHITE);
            p=p+2*dy;
        }else{
            x=x+1;
            y=y+1;
            putpixel(x,y,WHITE);
            p=p+2*dy-2*dx;
        }
    }
```

```
        delay(100);
    }
    getch(100);
}
```

Output

Enter Line End points:
100 100 200 100



4. WAP to implement mid-point Circle drawing algorithm.

Algorithm

(1) Input radius 'r' and centre (x_0, y_0) and obtain the first point on the circle centered at origin as

$$(x_0, y_0) = (0, r)$$

(2) Calculate the value of initial decision parameter P_0 as

$$P_0 = \frac{5}{4} - r$$

(3) At each point the x_k, starting at $k=0$ perform the following test:

if $P_k < 0$

- the next point to plot centered at origin as
  (x_{k+1}, y_k)
- $P_{k+1} = P_k + 2x_{k+1} + 1$

else

- the next point to plot centered at origin is
  (x_{k+1}, y_{k-1})
- $P_{k+1} = P_k + 2x_{k+1} + 1 - 2y_{k+1}$

(4) Determine the symmetry point on the other seven octants.

(5) Move each calculated point (x,y) into the circle centered at (x_c, y_c) as

$x = x + x_c$, $y = y + y_c$

(6) Repeat step 3 to 5 until $x \geq y$.

Program

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
void symmetry(int x, int y, int xc, int yc){
        putpixel(x+xc, y+yc, GREEN);
        putpixel(-x+xc, y+yc, GREEN);
        putpixel(x+xc, -y+yc), GREEN);
        putpixel(-x+xc, -y+yc, GREEN);
        putpixel(y+yc, x+xc, GREEN);
        putpixel(-y+yc, x+xc, GREEN);
        putpixel(y+yc, -x+xc, GREEN);
        putpixel(-y+yc, -x+xc, GREEN);
        delay(100);
}
void main(){
        int gd=DETECT, gm, x, y, p, r, xc, yc;
        initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");
        printf("Enter a center and Radius: \n");
        scanf("%d%d%d", &xc, &yc, &r);
        x=0;
        y=r;
        symmetry(x, y, xc, yc);
        delay(100);
        p=1-r;
```

```
do{
    if(p<0){
        x=x+1;
        y=y;
        Symmetry(x,y,xc,yc);
        p= p+2*x+1;
    }else{
        x=x+1;
        y=y-1;
        symmetry(x,y,xc,yc);
        p=p+2*x+1-2*y;
    }
}while(x<y);
getch();
}
```

Output

Enter a center and Radius:
200 200 50

5. WAP to implement mid-point ellipse drawing algorithm.

Algorithm

① Input center (x_c, y_c) and radii along x-axis r_x, radii along y axis, r_y and obtain the first point of the ellipse at origin as

$$(x_0, y_0) = (0, r_y)$$

(2) Calculate the value of initial decision parameter in region 1 as

$$P_{10} = ry^2 - rx^2 ry + \frac{1}{4} rx^2$$

(3) At the each position x_k, starting at $k=0$, perform the following test in region 1:-

if $P_{1k} < 0$

- The next point to plot ellipse centred at origin is (x_{k+1}, y_k)
- The $P_{1k+1} = P_{1k} + 2ry^2 x_{k+1} + ry^2$

else

- The next point to plot ellipse centred at origin is $(x_{k+1}, y_k - 1)$
- $P_{1k+1} = P_{1k} + 2ry^2 x_{k+1} + ry^2 - 2rx^2 y_{k+1}$

(4) Calculate the value of initial decision parameter at region 2 using last calculated point on the region 2, say (x_0, y_0)

(5) Calculate initial decision parameter value in region 2 as

$$P_{20} = ry^2 (x_0 + 1/2)^2 + rx^2 (y_0 - 1)^2 - rx^2 ry^2$$

(6) At each position y_k, starting at $k=0$, perform the following test in region 2

if $P_{2k} > 0$

- The next point to plot ellipse centred at origin is $(x_k, y_k - 1)$
- $P_{2k+1} = P_{2k} - 2rx^2 (y_k - 1) + rx^2$

else

- The next point to plot ellipse centred at origin is $(x_{k+1}, y_k - 1)$
- $P_{2k+1} = P_{2k} + 2ry^2 x_{k+1} - 2rx^2 y_{k+1} + rx^2$

(7) Determine the symmetric points on other 3 quadrants.

(8) Move each calculated point (x, y) on the ellipse centre (x_c, y_c) as

$$x = x + x_c$$
$$y = y + y_c$$

(9) Repeat the process for region 2 until $2ry^2 x_k \geq 2rx^2 y_k$ and region 2 until $(x, y) = (rx, 0)$

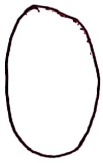
Program

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
void symmetry(int x, int y, int xc, int yc){
    putpixel(x+xc, y+yc, 15);
    putpixel(-x+xc, y+yc, 15);
    putpixel(x+xc, -y+yc, 15);
    putpixel(-x+xc, -y+yc, 15);
}
void main(){
    int gd=DETECT, gm, x, y, p1, p2, rx, ry, xc, yc;
    initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");
    printf("%d%d%d%d", &xc, &yc, &rx, &ry);
    x=0;
    y=ry;
    p1=ry*ry - rx*rx*ry + (1.0/4)*rx*rx;
    while(2*ry*ry*x < 2*rx*rx*y){
        if(p1<0){
            x=x+1;
            p1=p1+2*ry*ry*x+ry*ry;
        }else{
            x=x+1;
            y=y-1;
            p1=p1+2*ry*ry*x+ry*ry-2*rx*rx*y+rx*rx;
        }
        symmetry(x,y,xc,yc);
    }
    p2=ry*ry*(x+1.0/2)*(x+1.0/2)+rx*rx*(y-1)*(y-1)-rx*rx*ry;
    while(y>0){
        if(p2>0){
            x=x;
            y=y-1;
            p2=p2+rx*rx-2*rx*rx*y;
        }else{
```

```
        x=x+1;
        y=y-1;
        p2 = p2+2*x*x-2*x*x*y+2*y*y+x*x;
    }
    symmetry(x,y,xc,yc);
}
getch();
}
```

Output

Enter the centre and X Y radius:
100 100 20 30



6. WAP to perform 2D translation.

Algorithm

(1) Given a line with two end points (x_0, y_0) and (x_1, y_1).
(2) Plot the line
(3) Input the translation factor (tx, ty).
(4) Perform the operation

$x_0 = x_0 + tx;$
$y_0 = y_0 + ty;$
$x_1 = x_1 + tx;$
$y_1 = y_1 + ty;$

(5) Plot the line after applying translation factor
(6) END

Program

```
#include <stdio.h>
#include <conio.h>
#include<graphics.h>
void main(){
        int gd=DETECT,gm;
        int x0,y0,x1,y1,tx,ty;
        initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");
        printf("Enter the End point of a Line :\n");
        scanf("%d%d%d%d",&x0,&y0,&x1,&y1);
        line(x0,y0,x1,y1);
        printf("Enter a Translation Factor :\n");
        scanf("%d%d",&tx,&ty);
        x0+=tx;
        y0+=ty;
        x1+=tx;
        y1+=ty;
        line(x0,y0,x1,y1);
        getch();
}
```

Output

```
Enter the End point of a Line :
50 50 150 50

Enter a Translation Factor :
30 30
```

7. WAP to perform 2D rotation.

Algorithm

(1) Input the coordinates of the line (x_0, y_0) and (x_1, y_1)
(2) plot the line
(3) Ask for the degree of rotation (deg)
(4) Compute

$$x_0 = x_0 \cos(deg) - y_0 \sin(deg)$$
$$y_0 = y_0 \cos(deg) + x_0 \sin(deg)$$
$$x_1 = x_1 \cos(deg) - y_1 \sin(deg)$$
$$y_1 = y_1 \cos(deg) + x_1 \sin(deg)$$

(5) Plot the new rotated line

Program

```
#include<stdio.h>
#include<graphics.h>
#include<conio.h>
#include<math.h>
void main()
{
    int gd=DETECT,gm;
    int x0,y0,x1,y1,deg;
    initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");
    printf("Enter the End points of Line :\n");
    scanf("%d%d%d%d",&x0,&y0,&x1,&y1);
    line(x0,y0,x1,y1);
    printf("Enter a Degree for the Rotation :\n");
    scanf("%d",&deg);
    deg=deg*(3.1415/180);
    x0=(int) abs(x0*cos(deg)-y0*sin(deg));
```

```
y0 = (int) abs(y0 * cos(deg) + x0 * sin(deg));
x1 = (int) abs(x1 * cos(deg) - y1 * sin(deg));
y1 = (int) abs(y1 * cos(deg) + x1 * sin(deg));
line(x0, y0, x1, y1);
getch();
}
```

Output

```
Enter the End Point of a line:
100 100 200 100
Enter a Degree for the Rotation:
120
```

8. WAP to perform 2D Scaling.

Algorithm

1. Input the coordinates of line (x_0, y_0) and (x_1, y_1)
2. plot the given end points
3. Input the scaling factor value of X-axis(sx) and Y-axis(sy).
4. perform

$$x_0 = x_0 * sx$$
$$y_0 = y_0 * sy$$
$$x_1 = x_1 * sx$$
$$y_1 = y_1 * sy$$

5. plot the new scaled value

Program

```
#include<stdio.h>
#include<graphics.h>
void main(){
        int gd=DETECT,gm, x0,y0,x1,y1,sx,sy;
        initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");
        printf("Enter a End point of Line:\n");
        scanf("%d%d%d%d",&x0,&y0,&x1,&y1);
        line(x0,y0,x1,y1);
        printf("Enter the XY Scaling Factor:\n");
        scanf("%d%d",&sx,&sy);
        x0*=sx;
        y0*=sy;
        x1*=sx;
        y1*=sy;
        line(x0,y0,x1,y1);
        getch();
}
```

Output

```
Enter a End point of Line:
100 100 200 100
Enter the XY Scaling Factor:
2
2
```

3. WAP to perform 3D translation.

Algorithm

(1) Input the coordinates value of a line.
(2) Plot the line
(3) Input the translation factor t_x, t_y and t_z
(4) Perform

$$x' = x + t_x$$
$$y' = y + t_y$$
$$z' = z + t_z$$

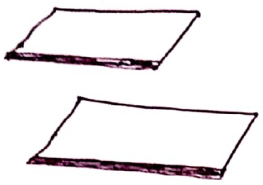
(5) Plot the new coordinates.

Program

```
#include <stdio.h>
#include <conio.h>
#include <graphics.h>
void main() {
        int x0,y0,x1,y1,tx,ty,gd=DETECT,gm;
        initgraph(&gd,&gm,"c:\\TURBOC3\\BGI");
        printf("Enter end-points of a line:\n");
        scanf("%d%d%d%d",&x0,&y0,&x1,&y1);
        bar3d(x0,y0,x1,y1,25,1);
        printf("Enter translation factor tx and ty:\n");
        scanf("%d%d",&tx,&ty);
        x0=x0+tx;
        y0=y0+ty;
        x1=x1+tx;
        y1=y1+ty;
        bar3d(x0,y0,x1,y1,25,1);
        getch();
}
```

Output

```
Enter end-points of a line:
100 100 200 100
Enter translation factor tx and ty:
20 30
```



10. WAP to perform 3D rotation.

program

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<graphics.h>
void main(){
    float x0, y0, x1, y1, r;
    int gd=DETECT, gm, xmid, ymid;
    initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");
    printf("Enter end-points of a Line:\n");
    scanf("%f%f%f%f", &x0, &y0, &x1, &y1);
    bar3d(x0, y0, x1, y1, 25, 1);
    printf("Enter rotation Angle:\n");
    scanf("%f", &r);
    r=r*(3.1415/180);
    xmid=(int)(x0+x1)/2;
    ymid=(int)(y0+y1)/2;
    x0 = abs(x0*cos(r)-y0*sin(r));
    y0 = abs(y0*cos(r)+x0*sin(r));
    x1 = abs(x1*cos(r)-y1*sin(r));
    y1 = abs(y1*cos(r)+y1*sin(r));
```

```
bar3d(xmid+x0, ymid+y0, xmid+x1, ymid+y1, 25, 1);
getch();
}
```

Output

```
Enter end-points of a line:
100 100 200 10
Enter rotation Angle:
60
```

11. WAP to perform 3D Scaling

Program

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
void main(){
    int x0,y0,x1,y1,sx,sy;
    int gd=DETECT,gm;
    initgraph(&gd,&gm,"C:\TURBOC3\BGI");
    printf("Enter end-points of a line:\n");
    scanf("%d %d %d %d",&x0,&y0,&x1,&y1);
    bar3d(x0,y0,x1,y1,25,1)
    printf("Enter Scaling factors sx and sy:\n");
```

```
scanf("%d%d", &sx, &sy);
x0 = x0*sx;
y0 = y0*sy;
x1 = x1*sx;
y1 = y1*sy;
bar3d(x0, y0, x1, y1, 25, 1);
getch();
}
```

Output

```
Enter end-points of a line:
100 100 150 100
Enter scaling factors sx and sy
2 2
```

12. Given a window lower left hand corner, Upper right hand corner and view port having lower left hand corner and Upper right hand corner. Map to find the point in viewport when window point is given using windows to viewport transformation.

```
#include<stdio.h>
void Conversion(float wx1, float wy1, float wx2, float wy2, float vx1, float vy1,
        float vx2, float vy2, float wx, float wy){
    float sx, sy, tx, ty, vx, vy;
    sx = (vx2 - vx1) / (wx2 - wx1);
    sy = (vy2 - vy1) / (wy2 - wy1);
    tx = vx1 - (sx * wx1);
    ty = vy1 - (sy * wy1);
    vx = sx * wx + tx;
```

```
    vy = sy * wy + ty;
    printf("The Converted point is (%.2f, %.2f)", vx, vy);
}
void main(){
    float wx1, wy1, wx2, wy2, vx1, vy1, vx2, vy2, wx, wy;
    printf("Enter the window lower left hand coordinates: \n");
    scanf("%f %f", &wx1, &wy1);
    printf("Enter the window Upper right hand coordinates: \n");
    scanf("%f %f", &wx2, &wy2);
    printf("Enter the viewport lower left hand coordinates: \n");
    scanf("%f %f", &vx1, &vy1);
    printf("Enter the viewport Upper right hand coordinates: \n");
    scanf("%f %f", &vx2, &vy2);
    printf("Enter the window point to be converted: \n");
    scanf("%f %f", &wx, &wy);
    Conversion(wx1, wy1, wx2, wy2, vx1, vy1, vx2, vy2, wx, wy);
}
```

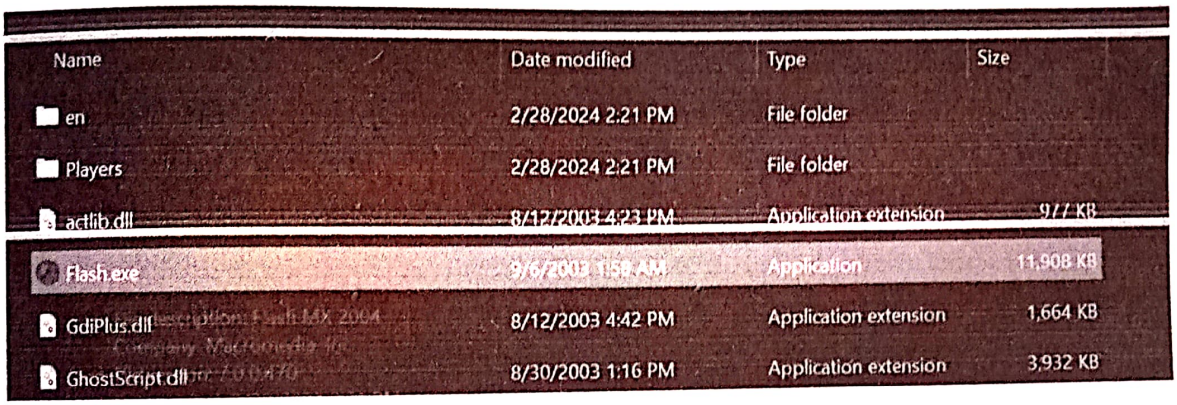
Output

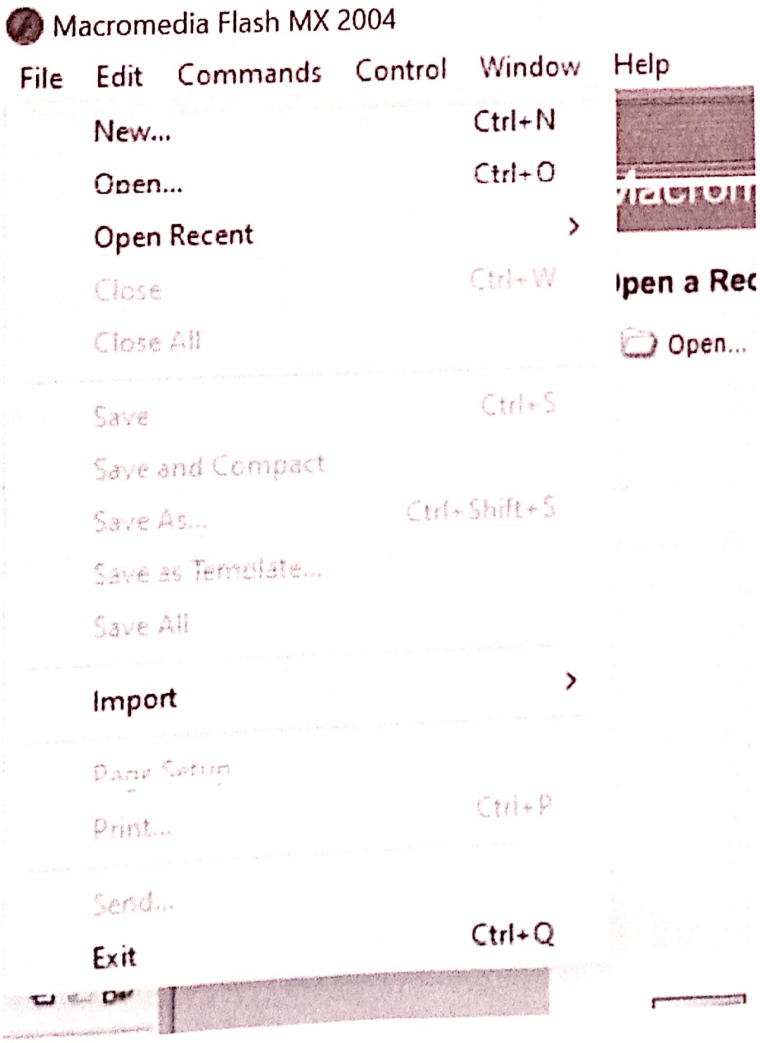
```
Enter the window lower left hand coordinates:
0 0
Enter the window Upper right hand coordinates:
4 4
Enter the viewport lower left hand coordinates:
0 0
Enter the viewport Upper right hand coordinates:
2 2
Enter the window point to be converted:
2 4
The converted point is (1.00, 2.00)
```

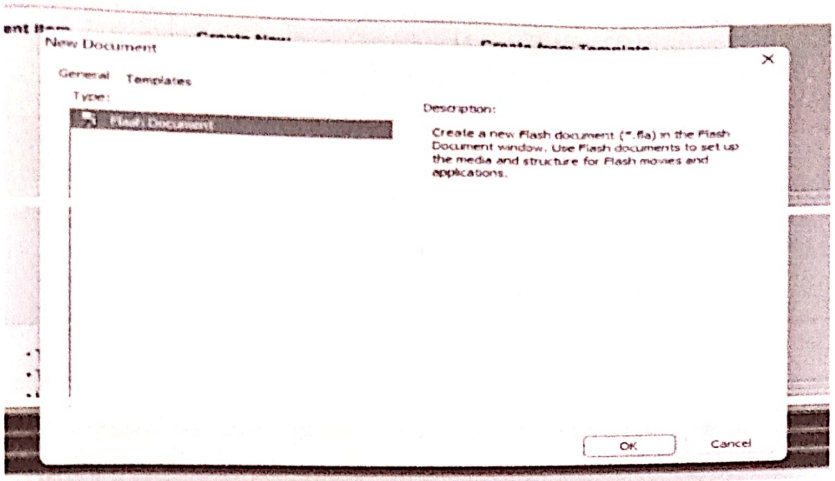
14. Write the step by step procedure with snapshot to set stage dimension in flash interface.

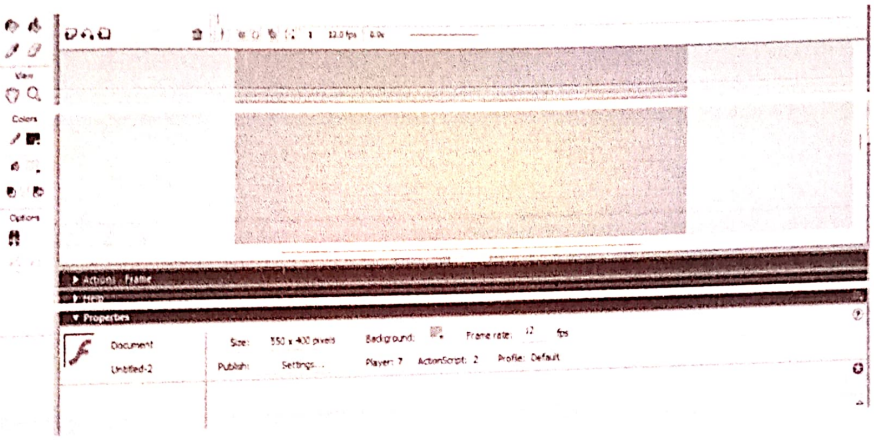
Steps for Setting Stage Dimension:

- Goto location where you install flash and double click on flash application.
- Create new flash document by clicking on File > New and on the
- General tab, select Flash Document.
- Goto properties by pressing Ctrl+F3 (or click windows menu and choose properties)
- Now, Change the stage dimension like: size, background color and frame rate





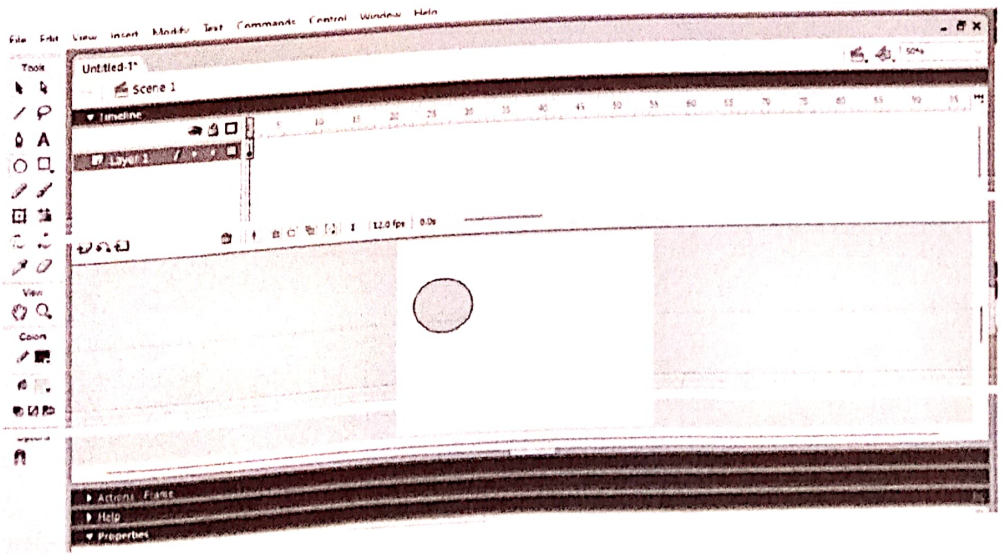


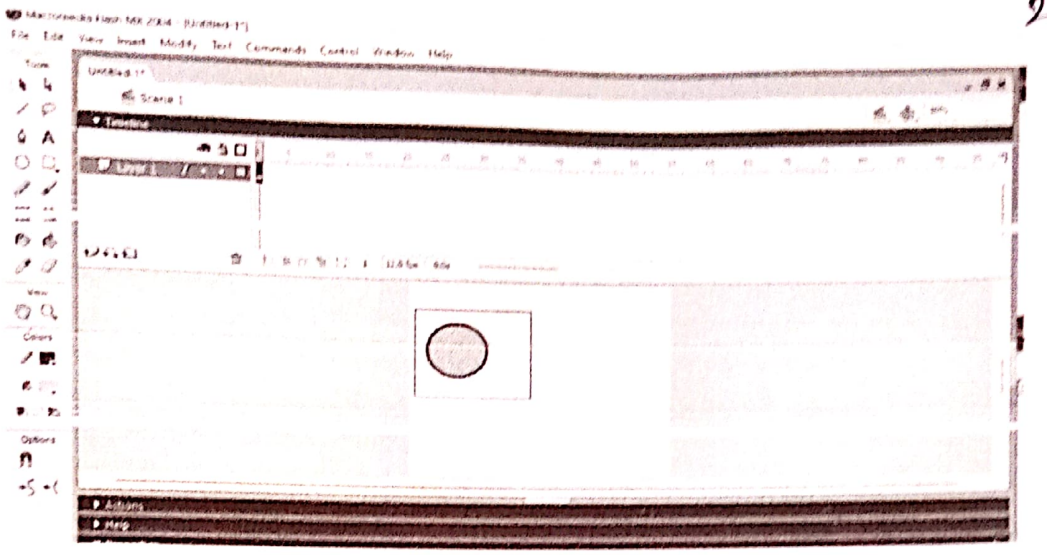


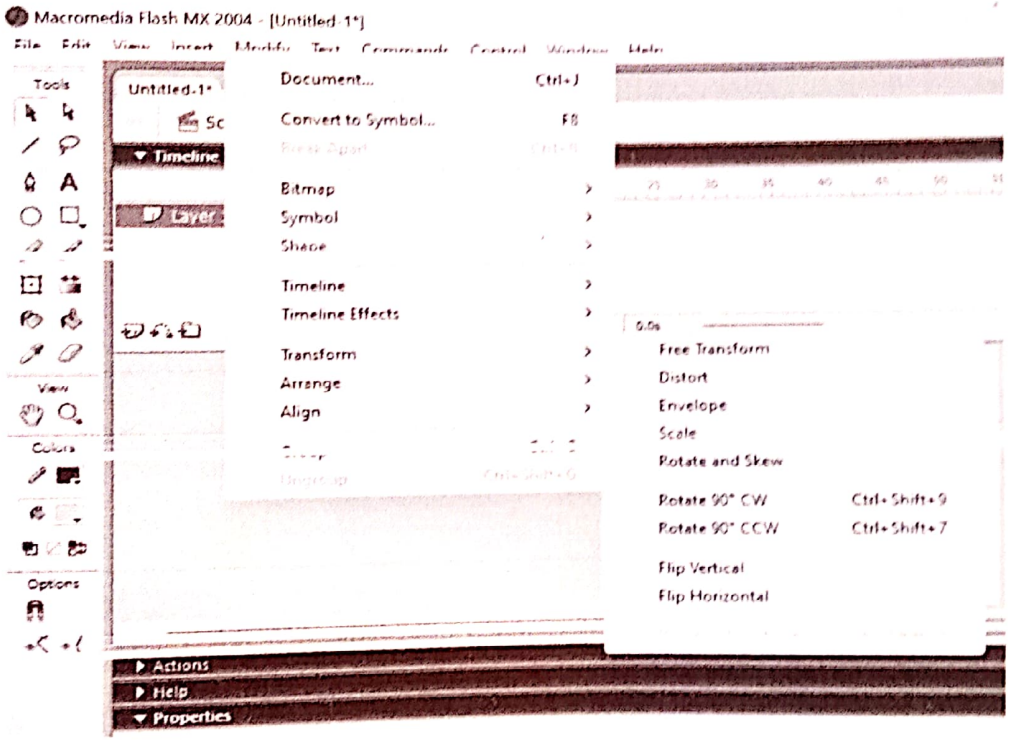
15. Write the step by step procedure with snapshot to modify different shapes in flash interface.

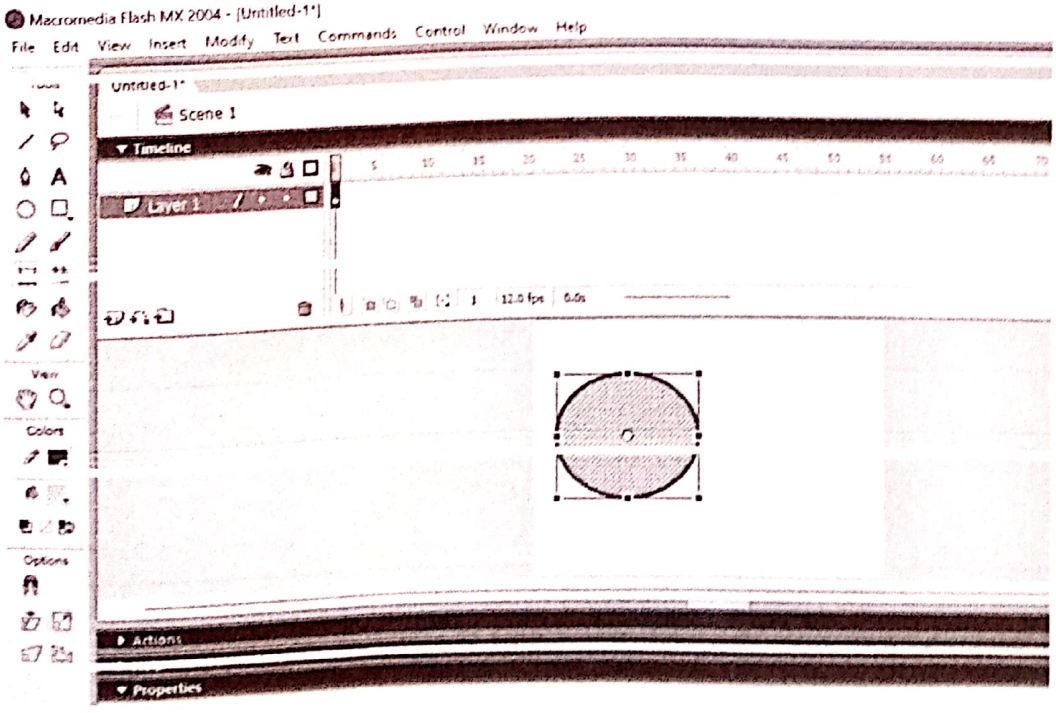
Drawing and Modifying Shapes

- Using Tools panel, draw any shape you would like to make.
- Select the object you draw by using selection tool.
- Click on modify submenu from the menu bar.
- Goto transform option and choose any operation: Scale, rotate, flip etc..) you would like to modify.
- Finally the shape is modified.





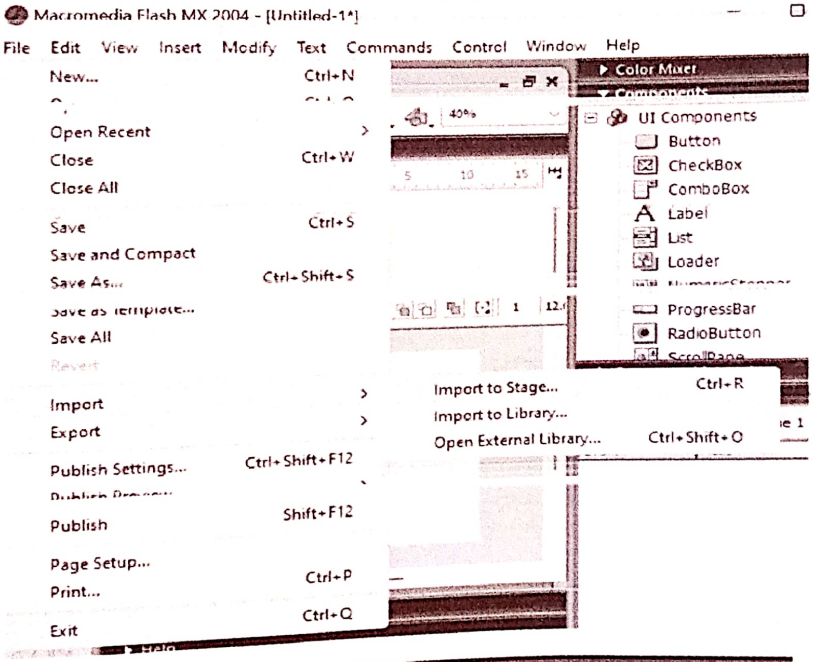

Macromedia Flash MX 2004 - [Untitled-1*]
File Edit View Insert Modify Text Commands Control Window Help
Tools
Untitled-1*
Timeline
Layer 1
Document... Ctrl+J
Convert to Symbol... F8
Bitmap
Symbol
Shape
Timeline
Timeline Effects
Transform
Arrange
Align
Free Transform
Distort
Envelope
Scale
Rotate and Skew
Rotate 90° CW Ctrl+Shift+9
Rotate 90° CCW Ctrl+Shift+7
Flip Vertical
Flip Horizontal
Actions
Help
Properties

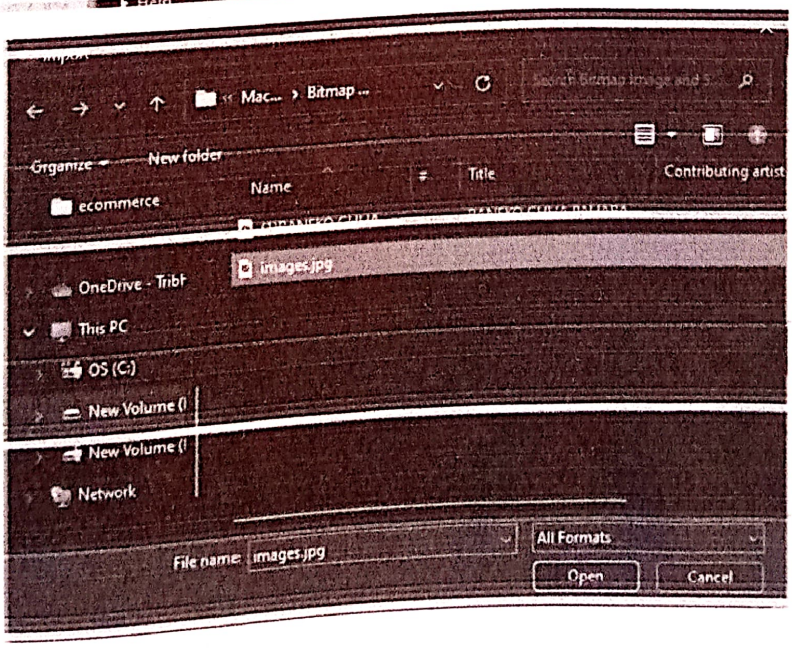

Macromedia Flash MX 2004 - [Untitled-1*]
File Edit View Insert Modify Text Commands Control Window Help
Untitled-1*
Scene 1
Timeline
Layer 1
Actions
Properties

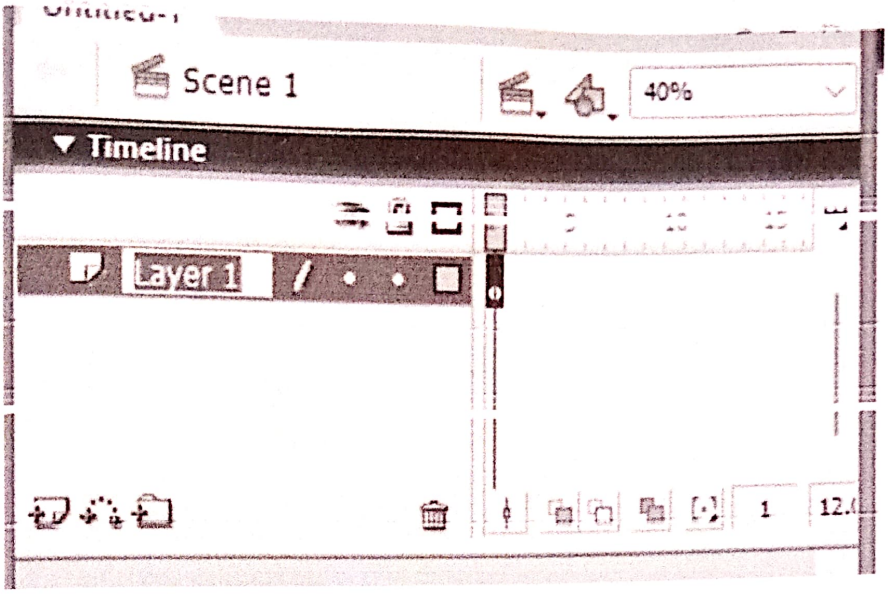
10. Write the step by step procedure with snapshot to work with bitmap images and sounds on background.

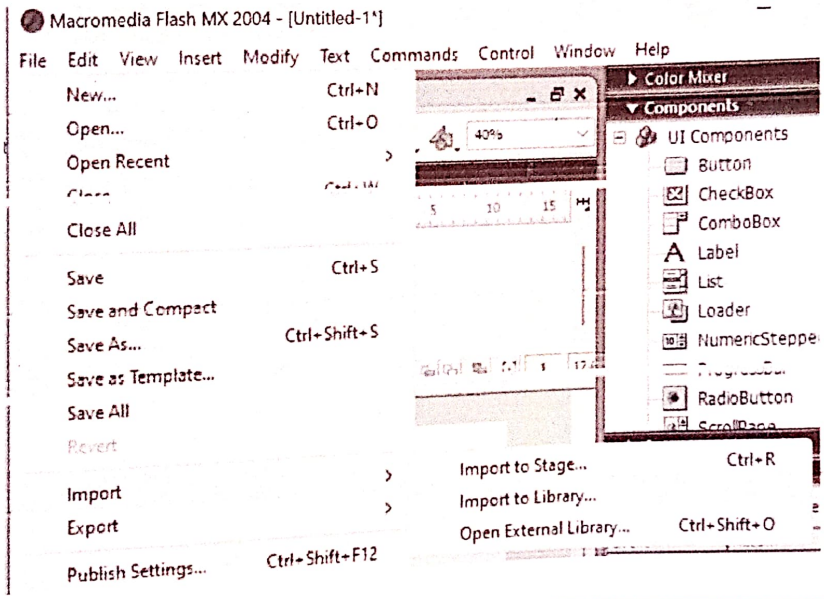
Bitmap Images and Sounds

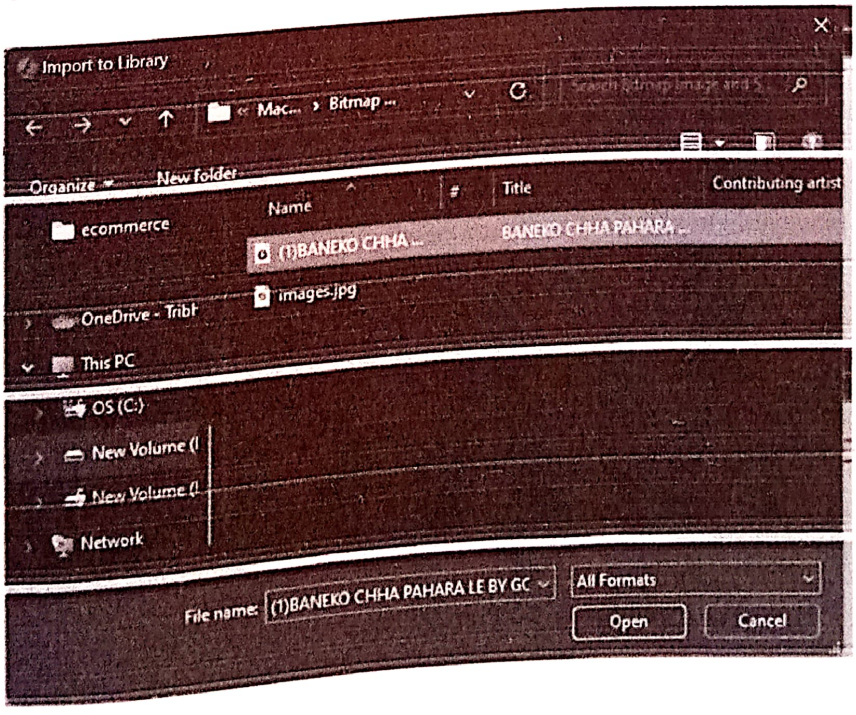
- Firstly, import image to stage (by clicking File->Import->import to stage and choosing image from drive location) and rename layer1 to image.
- Secondly, import sounds to library (by clicking File->Import->import to library and choosing sound from drive location) .
- Insert a new layer, named sound.
- Drag the frame1 to any frame number(say frame30) and right click on frame 30 and click insert keyframe.
- Goto properties and add sound that you have selected from the desired drive location.
- Click on control submenu and then play option (or Test movie option).

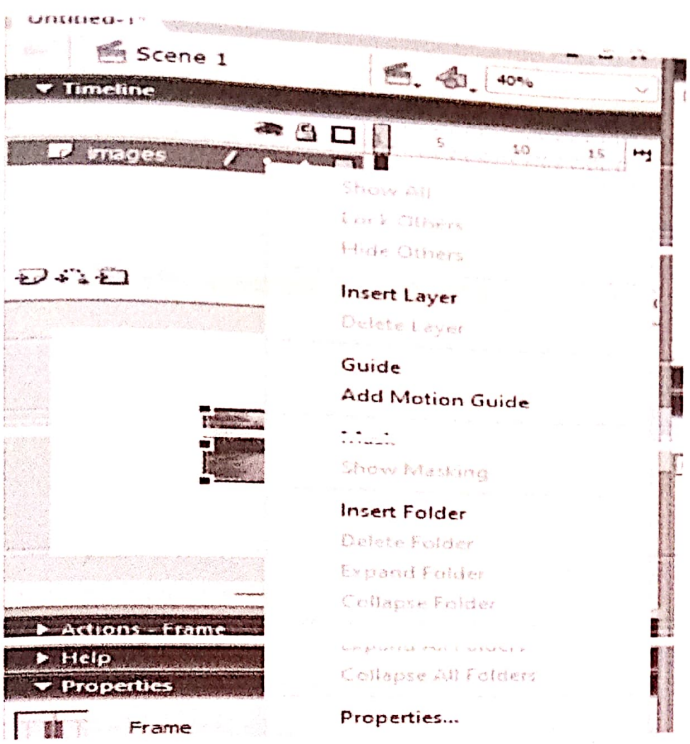


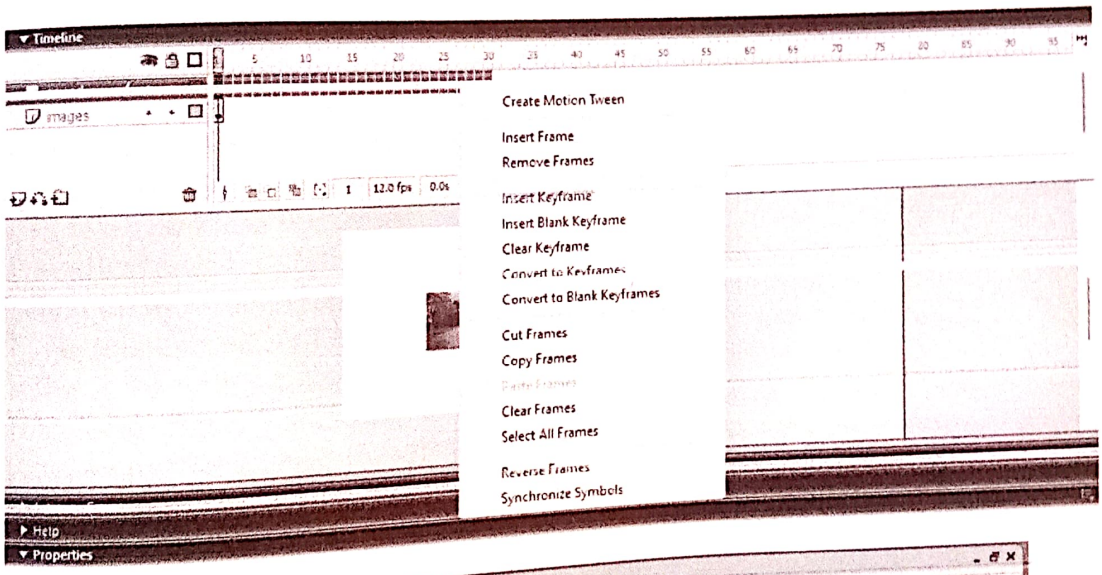


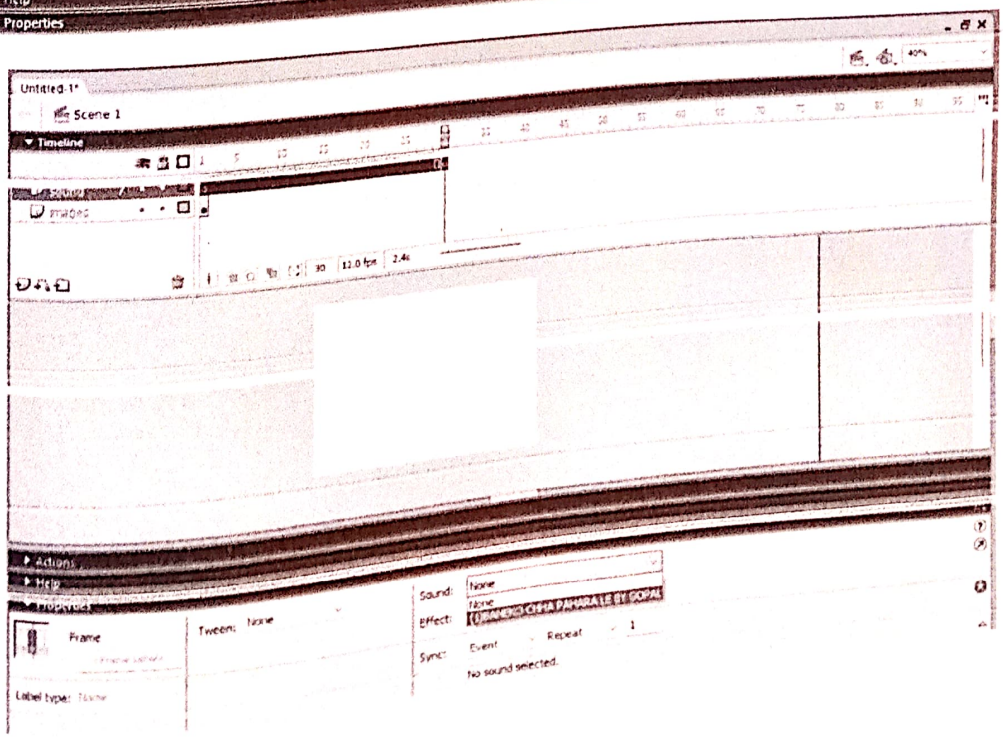

Scene 1
40%
Timeline
Layer 1
1

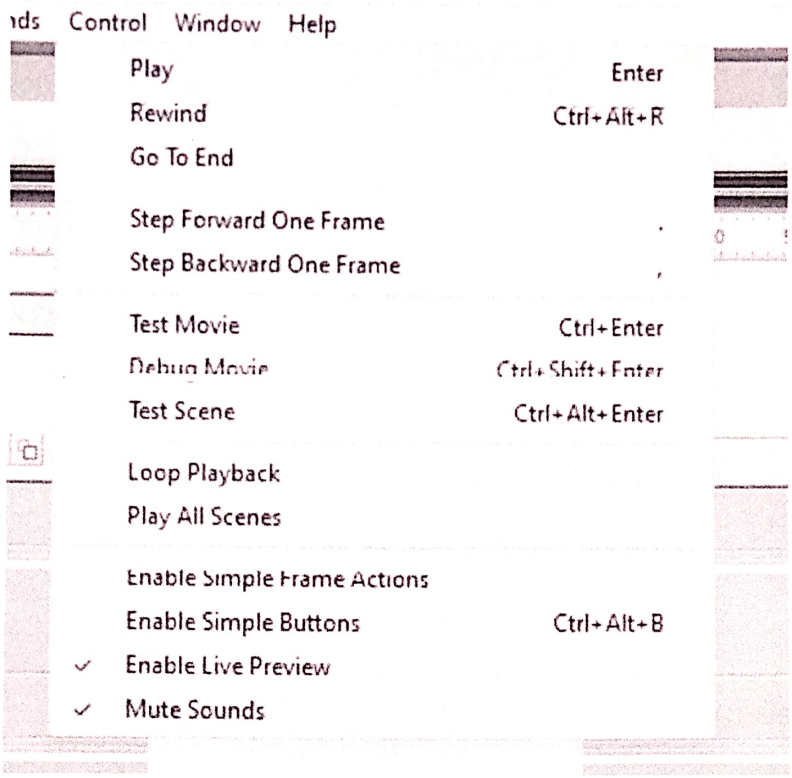

Macromedia Flash MX 2004 - [Untitled-1*]
File Edit View Insert Modify Text Commands Control Window Help
New... Ctrl+N
Open... Ctrl+O
Open Recent
Close All
Save Ctrl+S
Save and Compact
Save As... Ctrl+Shift+S
Save as Template...
Save All
Revert
Import
Export
Publish Settings... Ctrl+Shift+F12
Color Mixer
Components
UI Components
Button
CheckBox
ComboBox
Label
List
Loader
RadioButton
Import to Stage... Ctrl+R
Import to Library...
Open External Library... Ctrl+Shift+O


Import to Library
Organize
New folder
Name
Title
Contributing artist
ecommerce
(1)BANEKO CHHA ...
BANEKO CHHA PAHARA ...
images.jpg
OneDrive - Tribl
This PC
OS (C:)
New Volume (
New Volume (
Network
File name: (1)BANEKO CHHA PAHARA LE BY GC
All Formats
Open
Cancel


Scene 1
Timeline
40%
Images
Insert Layer
Delete Layer
Guide
Add Motion Guide
Show Masking
Insert Folder
Delete Folder
Expand Folder
Collapse Folder
Collapse All Folders
Properties...
Actions - Frame
Help
Properties
Frame

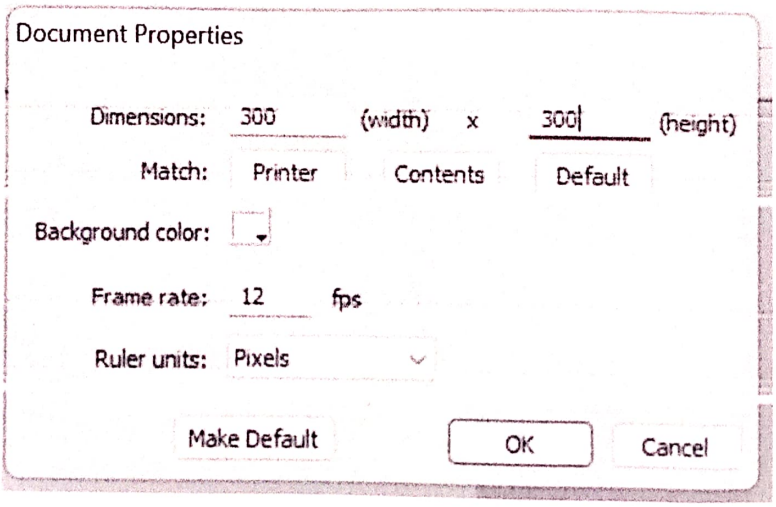

Timeline
Images
Create Motion Tween
Insert Frame
Remove Frames
Insert Keyframe
Insert Blank Keyframe
Clear Keyframe
Convert to Keyframes
Convert to Blank Keyframes
Cut Frames
Copy Frames
Clear Frames
Select All Frames
Reverse Frames
Synchronize Symbols
Help
Properties

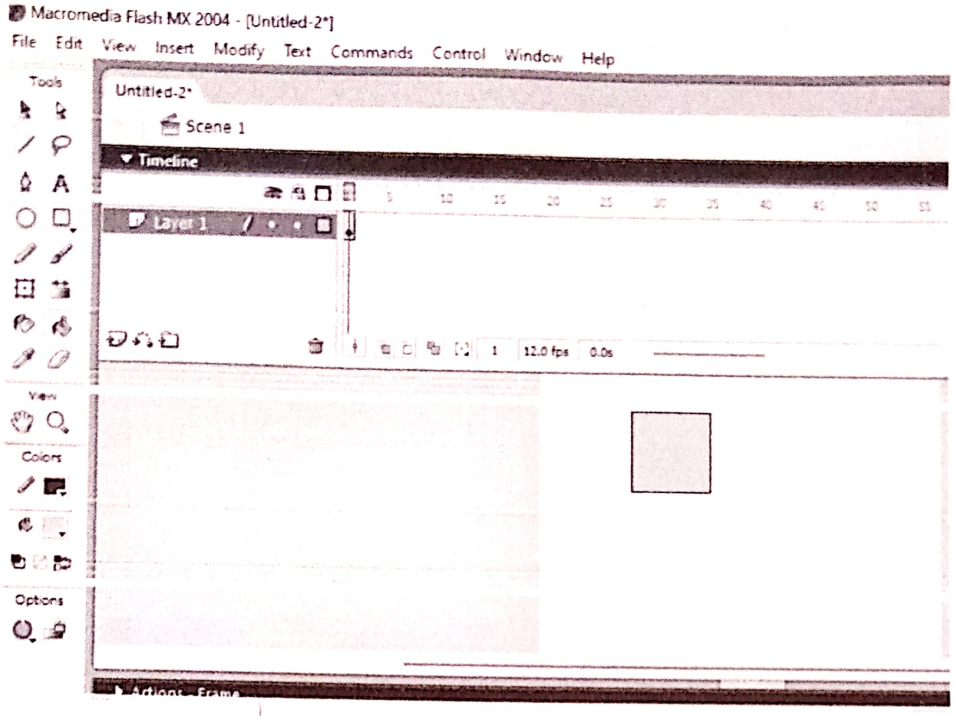

Untitled-1*
Scene 1
Timeline
Images
Actions
Help
Properties
Frame
Tween: None
Sound: None
Effect:
Sync: Event
Repeat
No sound selected.
Label type: None

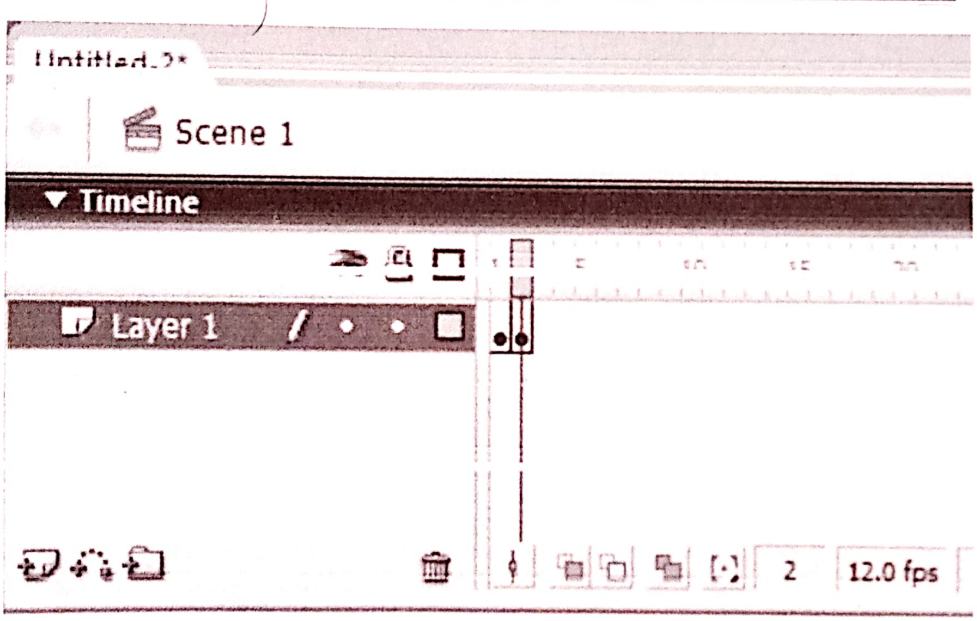


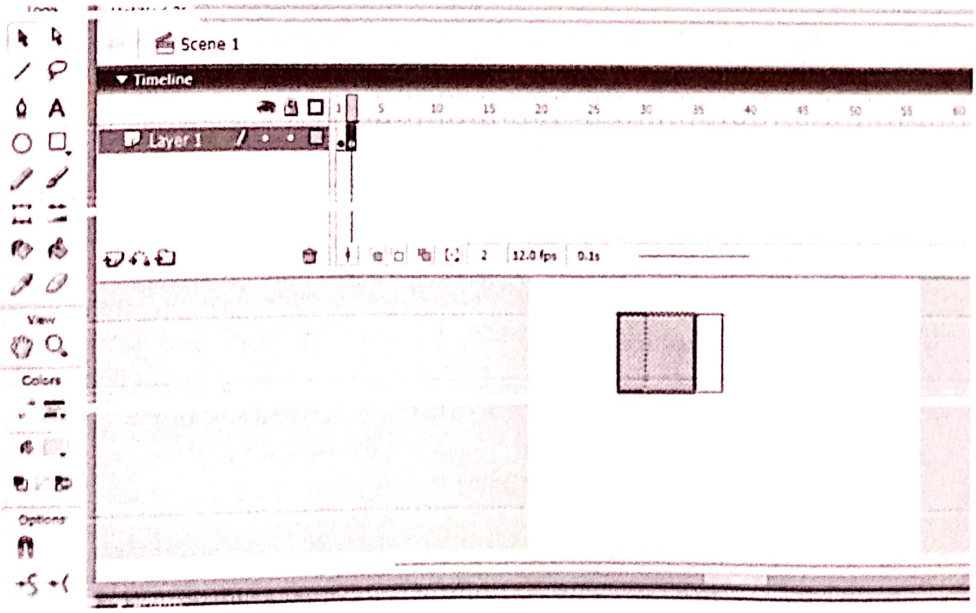
17. Write the step by step procedure with snapshot to perform frame by frame animation in flash interface.

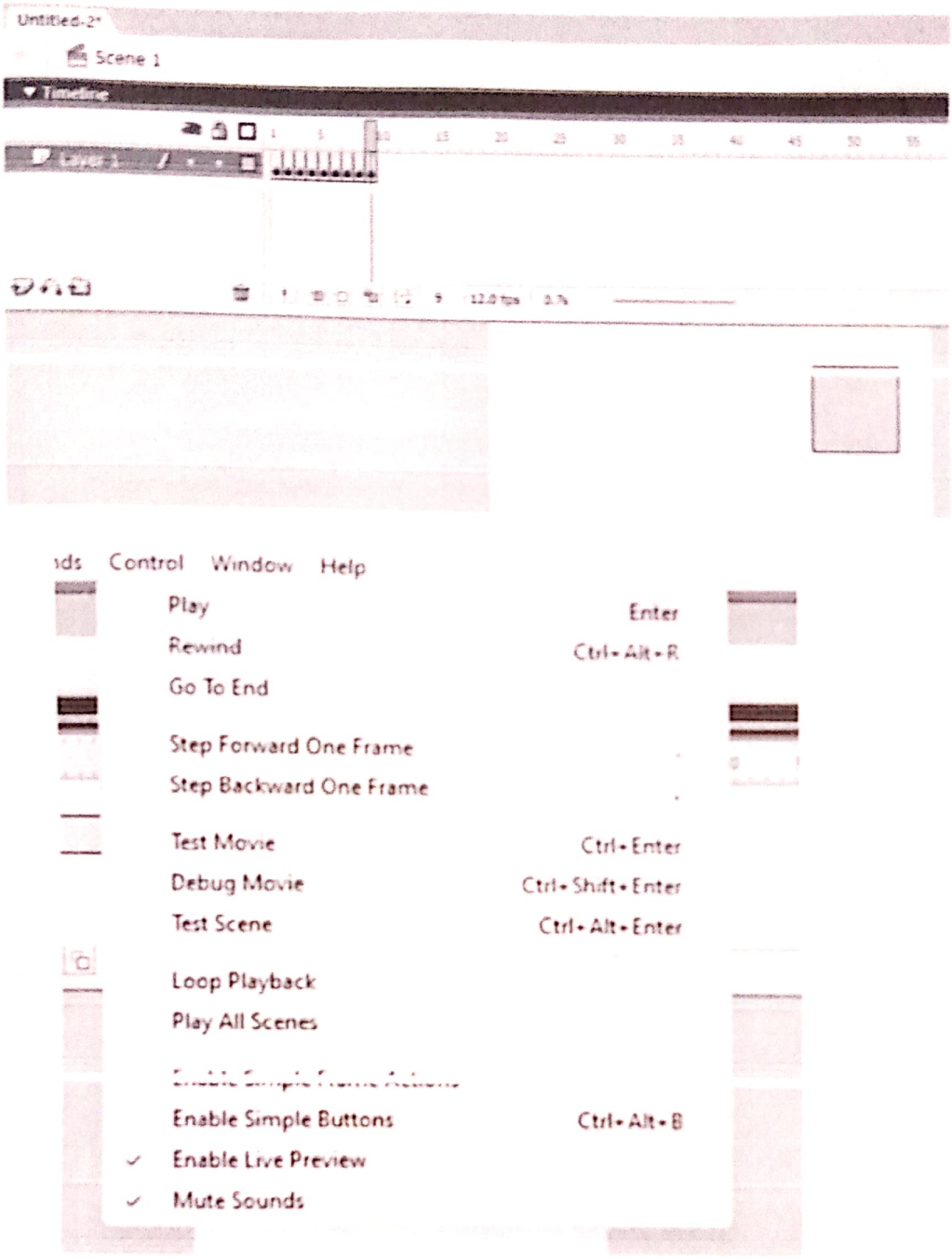
- Create a new Flash document
- Open the Modify -> Movie dialog box. For this exercise, select a width and height of 300*300 pixels. Leave every thing else the way it is.
- In the timeline, select Frame 1 on layer 1 and then draw square box on the stage. Fill it with a color you like using the point Bucket Tool
- Next select frame 2 and insert a new keyframe via the shortcut menu, or by pressing f6.(Notice how the playhead (the red bar on the top row of the timeline) moves to frame 2, indicating that frame 2 is now the active frame)
- Select the box you drew and move it a few pixels to the right.
- Repeat this process for frames 3 through to 5 times
- Now, at the frame 6 insert a new keyframe, and move the square a few pixels down. Repeat this process for frames 7 through 10.(At this point, you've created your first animation sequence)
- Select Control-> Play, or press the Enter key.(flash will play the animation for you by displaying each frame in succession.Note that you can loop the clip via the Control->Loop playback command)




Macromedia Flash MX 2004 - [Untitled-2*]
File Edit View Insert Modify Text Commands Control Window Help
Tools
Untitled-2*
Scene 1
Timeline
Layer 1
12.0 fps
0.0s
View
Colors
Options

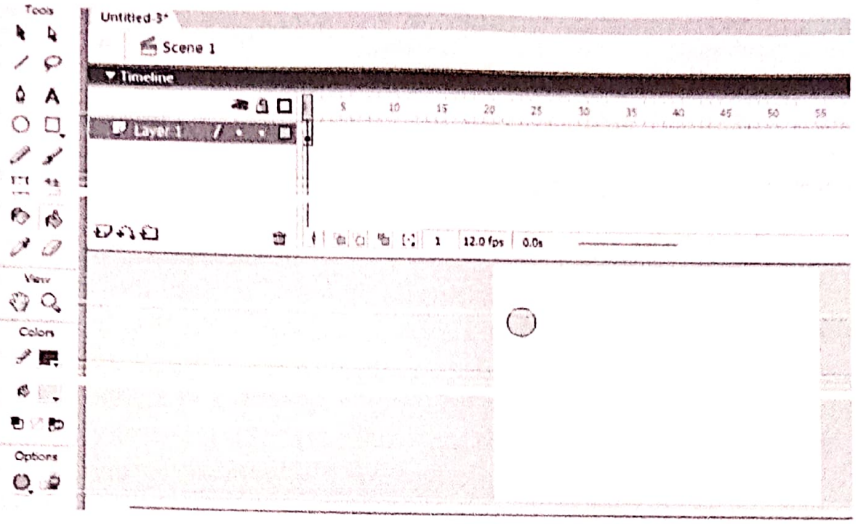

Scene 1
Timeline
Layer 1
2
12.0 fps

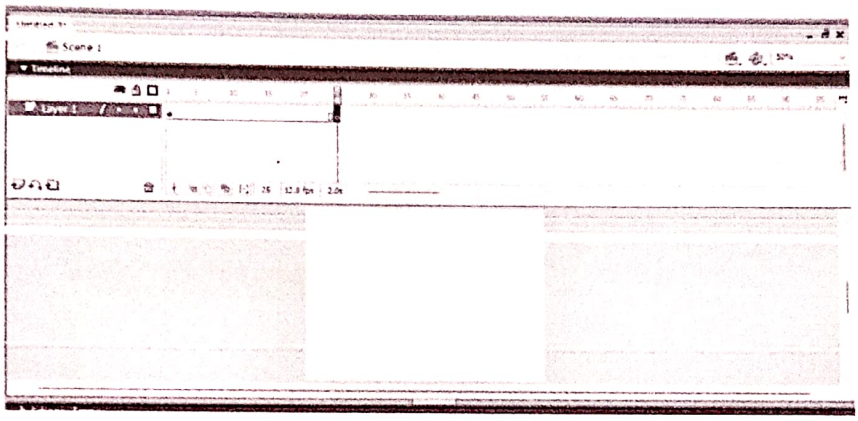

Scene 1
Timeline
Layer 1
2
12.0 fps
0.1s
View
Colors
Options

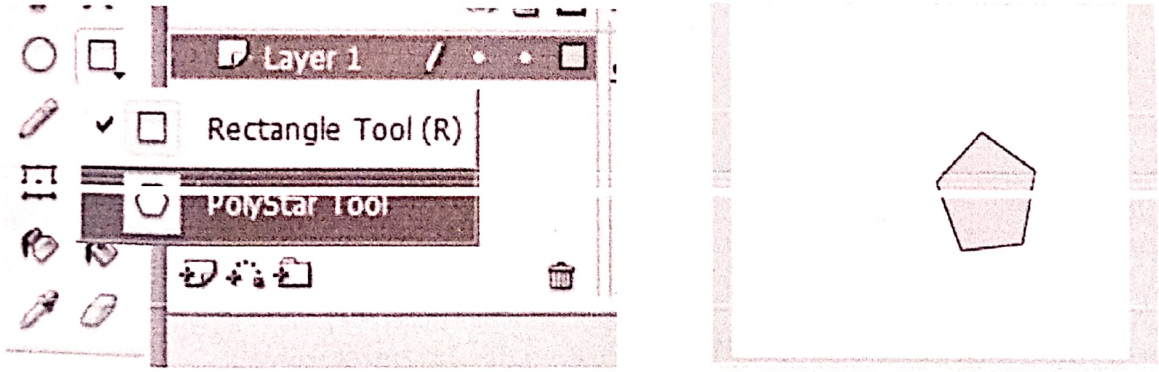


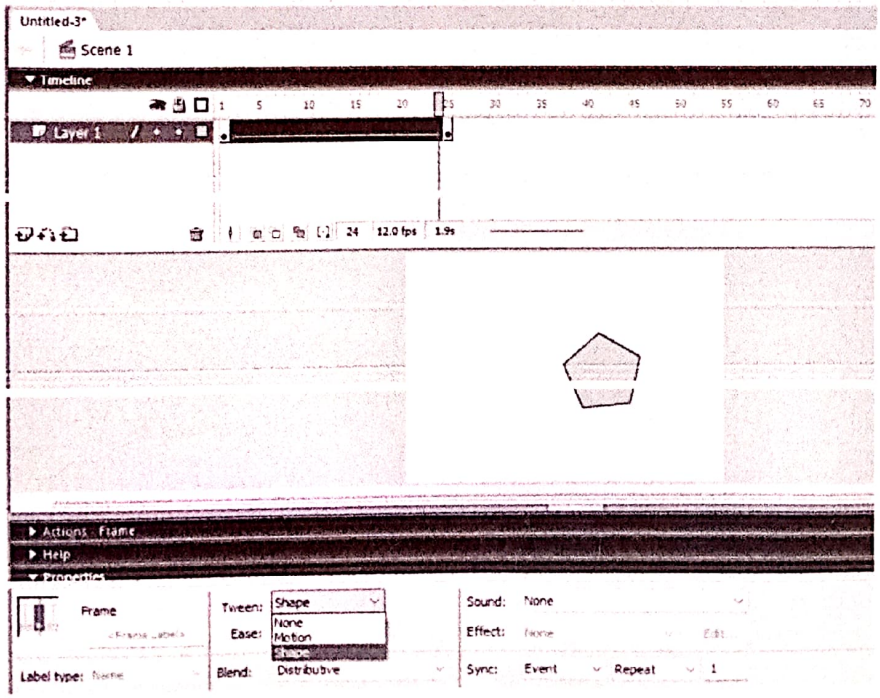
18. Write the step by step procedure with snapshot to perform shape tween in flash interface.

- Open a new flash file
- Select the first frame in Layer 1. Go to the stage and draw circle.
- Select frame 25 and insert a blank keyframe(f7)
- Still keeping play head on frame 25, draw a polygon on the stage using the polyester too.
- Select any frame between, 2 - 24 and select the shape from the tween drop-down menu in the Property inspector.
- Play you movie to view you motion tween.


Tools
Untitled-3*
Scene 1
Timeline
Layer 1
12.0 fps
0.0s
View
Colors
Options

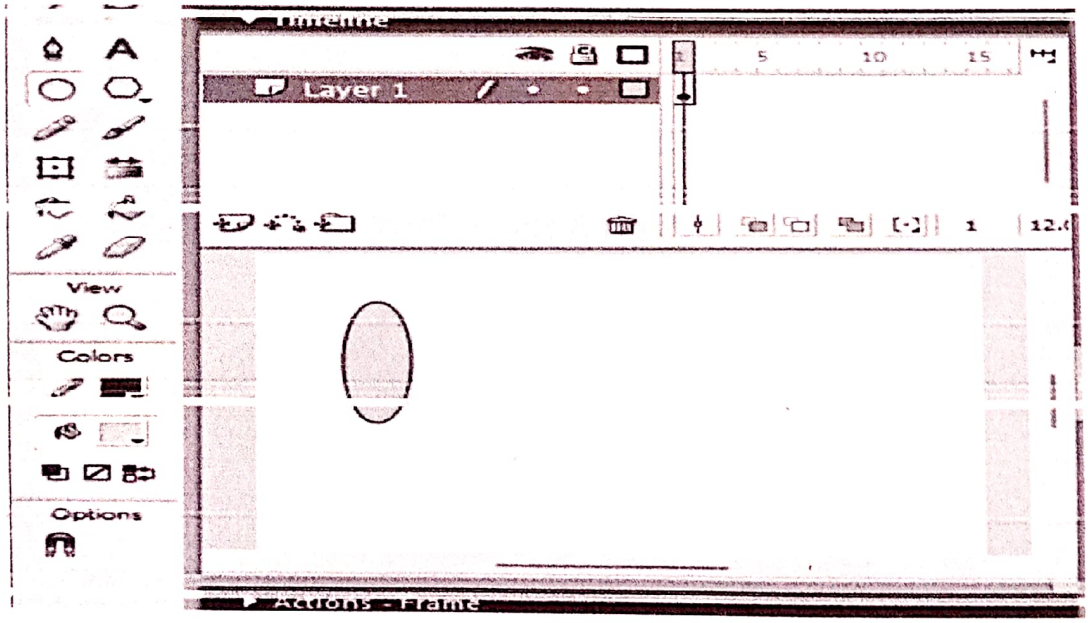


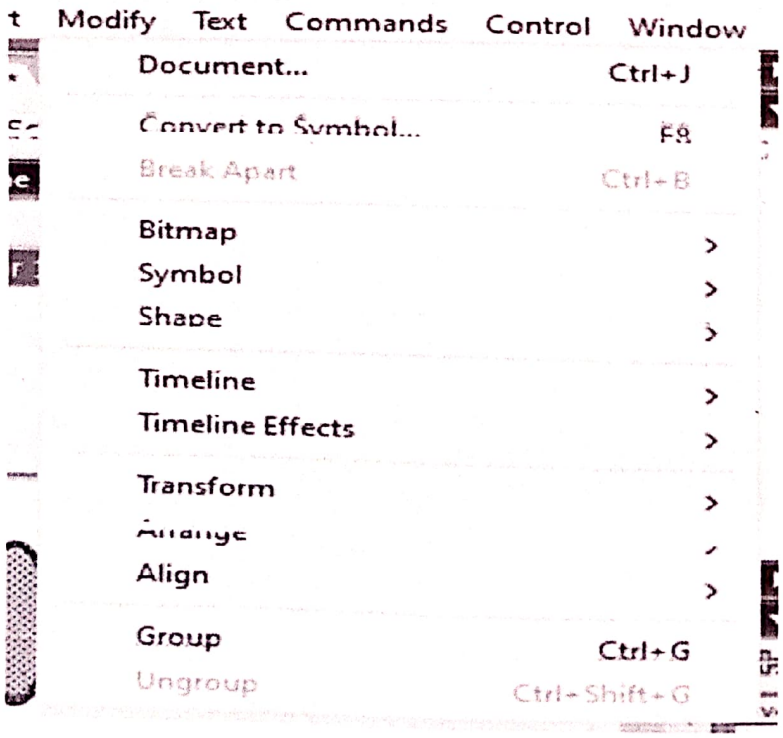

Layer 1
Rectangle Tool (R)
PolyStar Tool

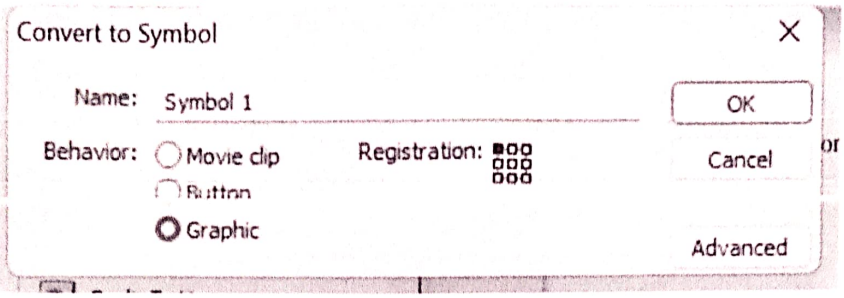

Untitled-3*
Scene 1
Timeline
Layer 1
24
12.0 fps
1.9s
Actions - Frame
Help
Properties
Frame
Tween:
Shape
None
Motion
Shape
Ease:
Blend:
Distributive
Sound:
None
Effect:
Sync:
Event
Repeat
1
Label type:

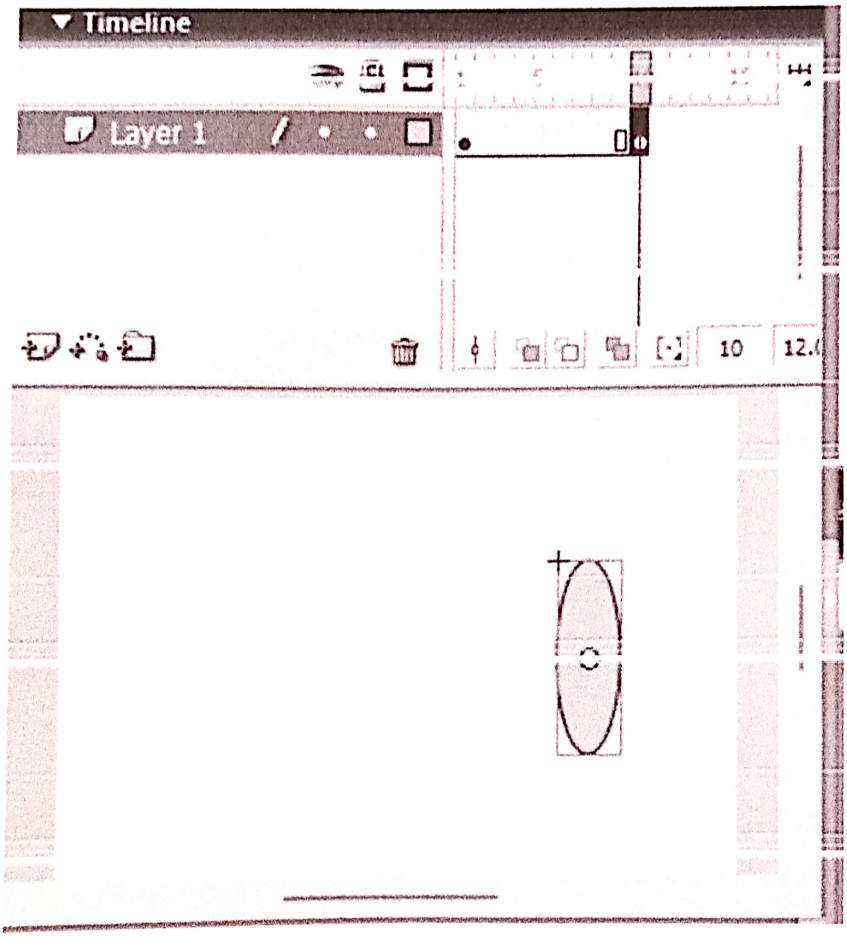
19. Write the step by step procedure with snapshot to perform motion tween in flash interface.

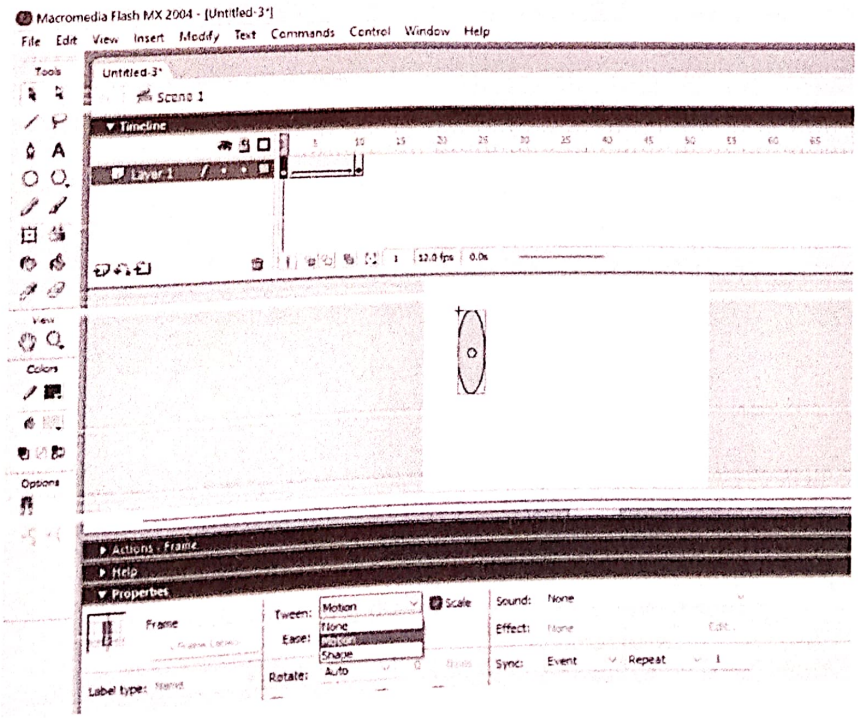
- Create a new Flash document
- Use the Oval tool to draw a simple ellipse and fill it with you favorite color.(notice that Flash automatically adds a keyframe at the frame 1 when you create the ellipse)
- Convert the ellipse to a graphic symbol by seelcting Modfy->Convert to Symbol, or by pressing f8. Make sure that the Graphich radio button is selected, and name the symbol Oval and click Ok.
- Click on frame 10 and isnert a new keyframe
- Move the ellipse to a new location on the stage
- Click on frame 1 in the timeline. In the properties Inspector, select "MOTION" in the tween drop-down menu
- Play the animation.

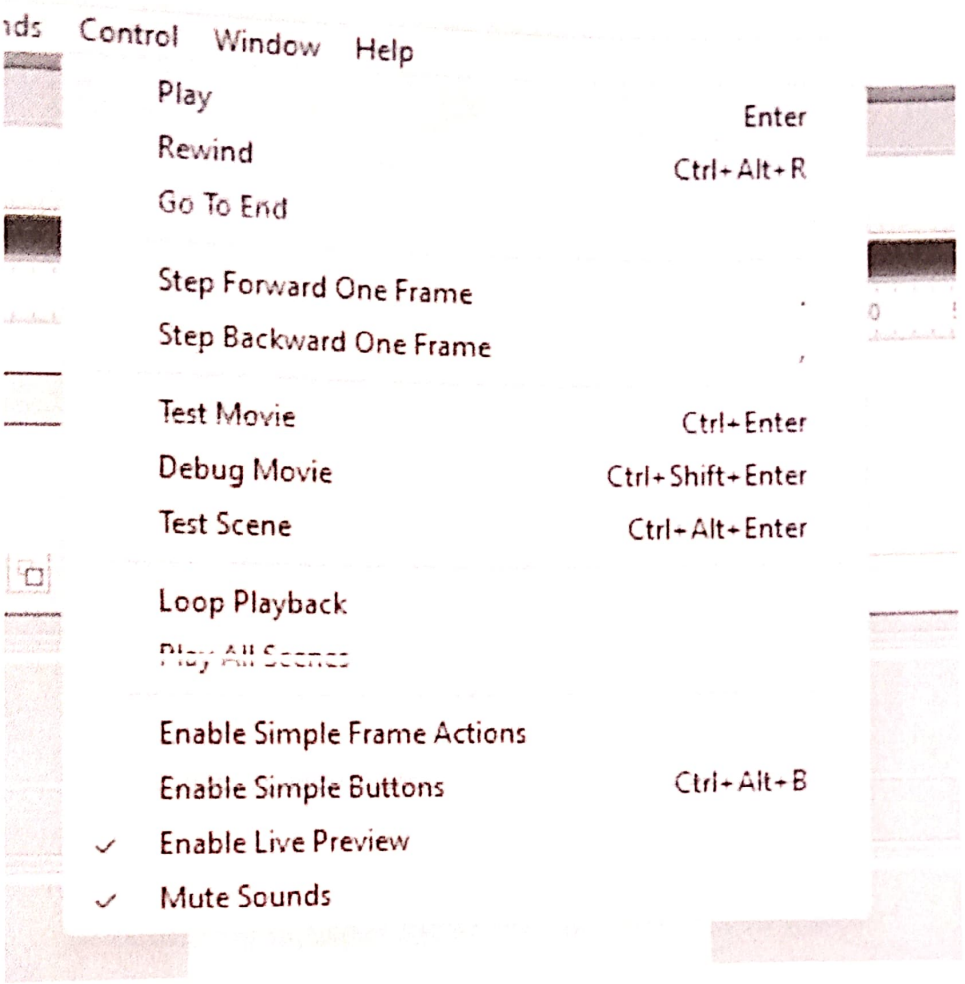





Convert to Symbol
Name: Symbol 1
OK
Behavior: Movie clip
Registration:
Cancel
Button
Graphic
Advanced


Timeline
Layer 1
10


Macromedia Flash MX 2004 - [Untitled-3*]
File Edit View Insert Modify Text Commands Control Window Help
Tools
Untitled-3*
Scene 1
Timeline
Layer 1
Actions - Frame
Help
Properties
Frame
Tween: Motion
Ease:
Rotate: Auto
Scale
Sound: None
Effect: None
Sync: Event
Repeat


ıds
Control
Window
Help
Play
Enter
Rewind
Ctrl+Alt+R
Go To End
Step Forward One Frame
.
Step Backward One Frame
,
Test Movie
Ctrl+Enter
Debug Movie
Ctrl+Shift+Enter
Test Scene
Ctrl+Alt+Enter
Loop Playback
Play All Scenes
Enable Simple Frame Actions
Enable Simple Buttons
Ctrl+Alt+B
Enable Live Preview
Mute Sounds