**MULTIMEDIA ASSIGNMENTS**

**ASSIGNMENT: 1**

* Question: Write C programs to draw some basic shapes such as line, rectangle, ellipse, sector and polygon.
* Code:

#include<iostream.h>

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

#include<math.h>

#include<dos.h>

#define ROUND(p) ((int)(p+0.5))

void ddaline(int,int,int,int);//function for line drawing using dda

void bhsmline(int,int,int,int,int,int);//line using brasenhum algo

void draw\_circle(void);//function for circle drawing

void circle\_pt(int,int);//for plotting of the 8 symmetrical points

void ellipsdraw(float,float);//function for ellips drawing

void ellips\_pt(float,float);//for plotting of the 4 symmetrical points

void setpxl(int,int); //for thickness of the line

void axisxy(void);//function for drawing the coordinate

float m,a,b;

int width,scl;

int x\_max,y\_max,xc,yc,r;

int choice;

struct time t1,t2;//for execution time

void main()

{

int gd=DETECT,gm;

int x1,x2,y1,y2,cc;

do

{

printf("\npress 1 for dda");

printf("\npress 2 for brasenhum");

printf("\npress 3 for brasenhum with thickness");

printf("\npress 4 for circle");

printf("\npress 5 for circle fill");

printf("\npress 6 for ellipse");

printf("\npress 7 for ellipse fill");

printf("\npress 8 for rectangle");

printf("\npress 9 for sector");

printf("\npress 10 for polygon");

printf("\npress 11 for exit");

printf("\nenter your choice:");

scanf("%d",&choice);

initgraph(&gd,&gm,"C:\\TC\\BGI");//initiation of the graph

x\_max=getmaxx();//gives maximum value of x on the screen

y\_max=getmaxy();//gives maximum value of y on the screen

if(choice==1)

{

printf("\n you have choosed DDA LINE::");

printf("\nEnter the starting point coordinate...\n");

scanf("%d %d",&x1,&y1);

printf("\nEnter the ending point coordinate...\n");

scanf("%d%d",&x2,&y2);

printf("\nenter the width:");

scanf("%d",&width);//width=1 gives normal line

gettime(&t1);//get the current time

cleardevice();//clears the screen before drawing

axisxy();//coordinate drawing

ddaline(x1,y1,x2,y2);//call dda function

}

if(choice==2||choice==3)

{

printf("\nyou have choosed BRASENHUM LINE::");

printf("\nEnter the starting point coordinate...\n");

scanf("%d %d",&x1,&y1);

printf("\nEnter the ending point coordinate...\n");

scanf("%d%d",&x2,&y2);

printf("\nenter the scaling factor:");

scanf("%d",&scl);

if(choice==3)

{

printf("\nenter the width:");

scanf("%d",&width);

}

else

{

width=1;//if width not required the it is set to 1

}

gettime(&t1);//gets the current time in structure variable t1

//scaling factor is multiplied with the starting & ending point

//to get the scaling effect

//(0,0)is transformed to the centre of the screen

x1=(x\_max/2)+(scl\*x1);

y1=(y\_max/2)-(scl\*y1);

x2=(x\_max/2)+(scl\*x2);

y2=(y\_max/2)-(scl\*y2);

cleardevice(); //clears the screen before drawing

axisxy();//draws the coordinate

if(x1==x2)//it gives a line parallale to y axis

bhsmline(y1,x1,y2,x2,0,2);

else

{

//function called for 1,2,7,8 & other drawn as mirror image

m=(float)(y2-y1)/(x2-x1);//slope

if(abs(m)<1.0 && m >= 0.0) //1st and 5th octant

bhsmline(x1,y1,x2,y2,0,0);

else if(abs(m) >=1.0 && m>0.0) //2nd and 6th octant

bhsmline(y1,x1,y2,x2,0,2);

else if(abs(m)<1.0 && m <0.0) //8th and 4th octant

bhsmline(x1,y1,x2,y2,1,0);

else if(abs(m)>=1.0 && m<=0.0)//7th and 3rd octant

bhsmline(y1,x1,y2,x2,1,2);

}

}

if(choice==4||choice==5)

{

printf("\nyou have choosed CIRCLE DRAWING::");

printf("\nEnter the centre of the circle:\n");

scanf("%d %d",&xc,&yc);

printf("\nEnter the radius:");

scanf("%d",&r);

gettime(&t1);

cleardevice();

axisxy();

xc=xc+ROUND(x\_max/2);//origin is transformed to the

//center of the screen

yc=ROUND(y\_max/2)-yc;

// draw centre of the circle

putpixel(xc,yc,10);

draw\_circle();

}

if(choice==6||choice==7)

{

printf("\nyou have entered ELLIPSE::");

printf("\n\n Enter value of 'a' and 'b': ");

scanf("%f%f",&a,&b);

printf("\nenter the centre of ellips:");

scanf("%d%d",&xc,&yc);

gettime(&t1);

cleardevice();

axisxy();

//transforming origin to the center of the screen

xc=xc+(x\_max/2);

yc=(y\_max/2)-yc;

ellipsdraw(a,b);//call for ellips drawing

}

if(choice==8)

{

rectangle(200,50,350,150);

}

if(choice==9)

{

outtextxy(120, 460, "Sector");

sector(150, 400, 30, 300, 100,50);

}

if(choice==10)

{

outtextxy(340, 460, "Polygon");

line(350,450,350,410);

line(350,410,430,400);

line(430,400,350,350);

line(350,350,300,430);

line(300,430,350,450);

}

if(choice==11)

{

break;

}

getch();

closegraph();

printf("\ndo you want to continue then press 1:");

scanf("%d",&cc);

}

while(cc);

getch();

}

void axisxy(void)

{

char msg[80];

line(4,y\_max/2,x\_max-4,y\_max/2);//draw the X-axis

line(x\_max/2,textheight(msg),x\_max/2,y\_max-3\*textheight(msg));//Y-axis

gotoxy(25,30);//set the point where the time to be printed

rectangle(4,textheight(msg),x\_max-4,y\_max-3\*textheight(msg)+1);//border

}

void ddaline(int x1,int y1,int x2,int y2)

{

float x,y,xincr,yincr;

int k,step,dx,dy;

x=x1+float(x\_max/2);

y=float(y\_max/2)-y1;

dx=x2-x1;

dy=y2-y1;

if(abs(dx)>abs(dy))

step=abs(dx);

else

step=abs(dy);

xincr=dx/(float)step; //if m<1 then xincr=1 & yincr=m

yincr=dy/(float)step; //if m>1 the xincr=1/m & yincr=1

setpxl(ROUND(x),ROUND(y));

for(k=0;k<step;k++)

{

x+=xincr;

y-=yincr;

setpxl(ROUND(x),ROUND(y));

}

gettime(&t2);

gotoxy(10,30);

//time required=current time(t2)-previous time(t1)

printf("the time=%2d:%02d:%02d.%09ld",t2.ti\_hour-t1.ti\_hour, t2.ti\_min-t1.ti\_min, t2.ti\_sec-t1.ti\_sec, t2.ti\_hund-t1.ti\_hund);

}

void bhsmline(int xS,int yS,int xE,int yE,int p,int q)

{

//parameter p=1 if m<0 else p=0,as if m<0 then y=y-1 to be done

//parameter q=2 if abs(m)>1 else q=0,as if abs(m)>1 then role of

//x & y to be switched

int dx,dy,steps,i,dS,dE,dNE,x,y,xF;

struct time t1,t2;

gettime(&t1);

dx=abs(xE-xS);

dy=abs(yE-yS);

dS=(2\*dy)-dx;

dE=2\*dy;

dNE=2\*(dy-dx);

if(xS>xE) //drawing from p1 to p0

{

x=xE;

y=yE;

xF=xS;

}

else //drawing from p0 to p1

{

x=xS;

y=yS;

xF=xE;

}

while(x<xF)

{

if(q==2)

setpxl(y,x);//for abs(m)>1

else

setpxl(x,y);

x++;

if(dS<0) //selection of E

dS+=dE;

else //selection of NE

{

if(p==0)

y++;

else

y--;

dS+=dNE;

}

}

gettime(&t2);

gotoxy(10,30);

//time required=current time(t2)-previous time(t1)

printf("the time=%2d:%02d:%02d.%09ld",t2.ti\_hour-t1.ti\_hour, t2.ti\_min-t1.ti\_min, t2.ti\_sec-t1.ti\_sec, t2.ti\_hund-t1.ti\_hund);

}

void setpxl(int x,int y)

{

//putpixel(x,y,12);

int i,w;

w=width/2;

putpixel(x,y,25);

if(abs(m)<1) //cloumn wise pixel glows

{

for(i=1;i<=w;i++)

{

putpixel(x,y+i,25); //half if the width is above the line

putpixel(x,y-i,25); //and half below the line

}

}

else //row wise pixel glows

{

for(i=1;i<=w;i++)

{

putpixel(x+i,y,25);

putpixel(x-i,y,25);

}

}

}

void draw\_circle(void)

{

int h,x,y;

x=0;

y=r;

h=1-r;

circle\_pt(x,y);

while(y>x)

{

if(h<=0) //select E

{

h=h+(2\*x)+3;

}

else //select NE

{

h=h+2\*(x-y)+5;

y=y-1;

}

x=x+1;

circle\_pt(x,y);

//delay(40);

}

// draw centre of the circle

putpixel(xc,yc,10);

gettime(&t2);

//time required=current time(t2)-previous time(t1)

printf("the time=%2d:%02d:%02d.%09ld",t2.ti\_hour-t1.ti\_hour, t2.ti\_min-t1.ti\_min, t2.ti\_sec-t1.ti\_sec, t2.ti\_hund-t1.ti\_hund);

return;

}

void circle\_pt(int x,int y)

{

//for choice=5 filling is required

//filling done by drawing stlines between symmetrical points

if(choice==5)

{

setcolor(4);//set the current color

line(xc+x,yc+y,xc-x,yc+y);

}

putpixel(xc+x,yc+y,25);//octant-2

putpixel(xc-x,yc+y,25);//octant-3

if(choice==5)

{

setcolor(6);

line(xc-x,yc-y,xc+x,yc-y);

}

putpixel(xc-x,yc-y,25); //octant-6

putpixel(xc+x,yc-y,25); //octant-7

if(choice==5)

{

setcolor(8);

line(xc+y,yc+x,xc-y,yc+x);

}

putpixel(xc+y,yc+x,25);//octant-1

putpixel(xc-y,yc+x,25);//octant-4

if(choice==5)

{

setcolor(2);

line(xc-y,yc-x,xc+y,yc-x);

}

putpixel(xc-y,yc-x,25); //octant-5

putpixel(xc+y,yc-x,25); //octant-8

return;

}

void ellipsdraw(float a,float b)

{

float d,x,y;

d=(b\*b)-(b\*a\*a+0.25\*a\*a);//descision variable for region-1

x=0;

y=b;

ellips\_pt(x,y);

while (2\*x\*b\*b <= 2\*y\*a\*a)//drawing region-1

{

if (d<0)//selection of E

{

x=x+1;

y=y;

d=d+2\*x\*b\*b+3\*b\*b;

}

else//selection of SE

{

x=x+1;

y=y-1;

d=d+2\*x\*b\*b+3\*b\*b-2\*y\*a\*a+2\*a\*a;

}

ellips\_pt(x,y);

//delay(40);

}

d=(x+0.5)\*(x+0.5)\*b\*b+(y-1)\*(y-1)\*a\*a-a\*a\*b\*b;//initial value for region-2

ellips\_pt(x,y);

while (y>0)

{

if (d>0)//selection of SE

{

x=x;

y=y-1;

d=d-2\*y\*a\*a+3\*a\*a;

}

else //selection of S

{

x=x+1;

y=y-1;

d=d-2\*y\*a\*a+3\*a\*a+2\*x\*b\*b+2\*b\*b;

}

ellips\_pt(x,y);

//delay(40);

}

putpixel(xc,yc,5);

gettime(&t2);

//time required=current time(t2)-previous time(t1)

printf("the time=%2d:%02d:%02d.%09ld",t2.ti\_hour-t1.ti\_hour, t2.ti\_min-t1.ti\_min, t2.ti\_sec-t1.ti\_sec, t2.ti\_hund-t1.ti\_hund);

}

void ellips\_pt(float x,float y)

{

if(choice==7)

{

setcolor(4);

line(xc+x,yc+y,xc+x,yc-y);

}

putpixel(xc+x,yc+y,25);

putpixel(xc+x,yc-y,25);

if(choice==7)

{

setcolor(7);

line(xc-x,yc+y,xc-x,yc-y);

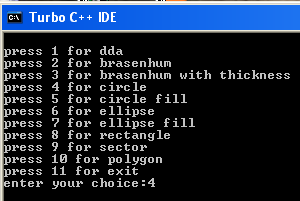
}

putpixel(xc-x,yc+y,25);

putpixel(xc-x,yc-y,25);

}

* Input: This is a menu driven function. Input will be taken as follows,

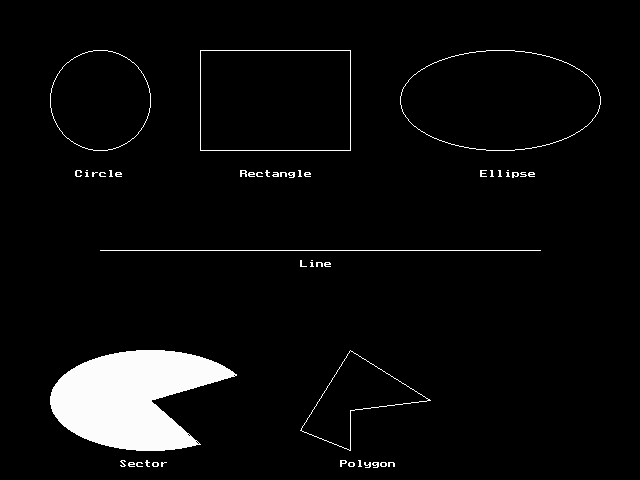


Here choice 4 is selected so the circle will be drawn. The input related to circle will be taken as follows,

Center of the circle and radius is taken from user.

Same thing occur for the other choices.

* Output:



This is not the exact out of the code above but the graphics figures will be of similar type. But in the code as it menu driven so all do not comes at once and it also shows filling by colour pixels.

**ASSIGNMENT: 2**

* Question: Write a program with colours, pixels, bar and clear device using random number generation. We have a function random(no), it returns a random number between 0 and no. The effect is by drawing random radius, random colour circles with same center and random pixels. Kbhit() function(define in conio.h) returns a nonzero value when a key is pressed in the keyboard. So, the loop will continue until a key is pressed.
* Code:

#include<graphics.h>

#include<stdlib.h>

int xMax,yMax;

void screenInitialize();

void draw();

void screenInitialize()

{

int gdriver = DETECT, gmode, errorcode;

initgraph(&gdriver, &gmode, "C:\\tc\\bgi");

outport(0x0378,0x00);

errorcode = graphresult();

if (errorcode != grOk)

{

printf("Graphics error: %s\n",grapherrormsg(errorcode));

printf("Press any key to halt:");

getch();

exit(0);

}

xMax=getmaxx();

yMax=getmaxy();

}

void draw()

{

setcolor(0);

setfillstyle(1,12);

bar(xMax/2-300,yMax/2-200,xMax/2+300,yMax/2+200);

setfillstyle(1,8);

bar(xMax/2-250,yMax/2-150,xMax/2+250,yMax/2+150);

while(!kbhit())

{

putpixel(random(439)+101,random(279)+101,random(16));

delay(300);

setcolor(random(16));

delay(300);

circle(320,240,random(170));

delay(300);

}

closegraph();

}

int main()

{

screenInitialize();

draw();

return 0;

}

**ASSIGNMENT: 3**

* Question: Given a finite number of cities (choose cities from 1 to a finite number) along with the distance of travel ( distance between two cities are randomly selected) between each pair of them. The aim is to find the cheapest distance of visiting all the cities and returning to the starting point.
* Code:

#include <stdio.h>

#define MAX 100

#define INFINITY 999

int tsp\_dp (int c[][MAX], int tour[], int start, int n);

int main()

{

int n; /\* Number of cities. \*/

int i, j; /\* Loop counters. \*/

int c[MAX][MAX]; /\* Cost matrix. \*/

int tour[MAX]; /\* Tour matrix. \*/

int cost; /\* Least cost. \*/

clrscr();

printf ("This program demonstrates the TSP problem.");

printf ("\nHow many cities to traverse? ");

scanf ("%d", &n);

printf ("Enter the cost matrix: (999: no connection)\n");

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

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

scanf ("%d", &c[i][j]);

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

tour[i] = i;

cost = tsp\_dp (c, tour, 0, n);

printf ("Minimum cost: %d.\nTour: ", cost);

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

printf ("%d ", tour[i]+1);

printf ("1\n");

getch();

return 0;

}

int tsp\_dp (int c[][MAX], int tour[], int start, int n)

{

int i, j, k; /\* Loop counters. \*/

int temp[MAX]; /\* Temporary during calculations. \*/

int mintour[MAX]; /\* Minimal tour array. \*/

int mincost; /\* Minimal cost. \*/

int ccost; /\* Current cost. \*/

/\* End of recursion condition. \*/

if (start == n - 2)

return c[tour[n-2]][tour[n-1]] + c[tour[n-1]][0];

/\* Compute the tour starting from the current city. \*/

mincost = INFINITY;

for (i = start+1; i<n; i++)

{ for (j=0; j<n; j++)

temp[j] = tour[j];

/\* Adjust positions. \*/

temp[start+1] = tour[i];

temp[i] = tour[start+1];

/\* Found a better cycle? (Recurrence derivable.) \*/

if (c[tour[start]][tour[i]] +

(ccost = tsp\_dp (c, temp, start+1, n)) < mincost) {

mincost = c[tour[start]][tour[i]] + ccost;

for (k=0; k<n; k++)

mintour[k] = temp[k];

}

}

/\* Set the minimum-tour array. \*/

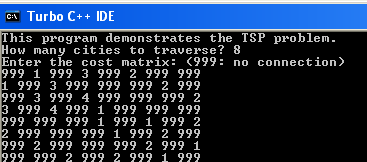
for (i=0; i<n; i++)

tour[i] = mintour[i];

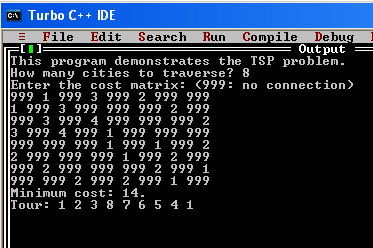
return mincost;

}

* Input: Input taken as follows,



* Output: output will be as follows,



**ASSIGNMENT: 4**

* Question: Read and display an image. A region of interest (ROI) is a portion of an image to filter or perform some other operation on. You can define ROI by creating a binary mask, which is a binary image that is the same size as the image you want to process with pixels that define the ROI set to 1 and all other pixel set to 0.

Creating a Binary mask

You can use the createMask method of the imroi base class to create a binary mask for any type of ROI object --- impoint, imlime, imrect, imellipse, impoly, or imfreehand. The createMask method returns a binary image the same size as the input image, containing 1s inside the ROI and 0s everywhere else.

For example, suppose you want to filter the grayscale image I, filtering only those pixels whose values are greater than 0.5. You can create the appropriate mask with this command:

BW= (I>0.5).

* Code:

%----------- READ THE IMAGE AND DISPLAY IT ------------%

img = imread('cameraman.tif');

h\_im = imshow(img);

%-------FIND REGION OF INTEREST(ROI): CREATE MASK & APPLY FILTER ------%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%---------- ROI OF POINT SHAPE -----------%

e1 = impoint(gca,100,100);

BW1 = createMask(e1,h\_im);

figure, imshow(BW1);

H = fspecial('unsharp');

J1 = roifilt2(H,img,BW1);

figure, imshow(J1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%---------- ROI OF LINE SHAPE -----------%

e2 = imline(gca,[100 150;170 150]);

BW2 = createMask(e2,h\_im);

figure, imshow(BW2);

H = fspecial('unsharp');

J2 = roifilt2(H,img,BW2);

figure, imshow(J2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%---------- ROI OF RECTANGLE SHAPE -----------%

e3 = imrect(gca,[10 10 100 100]);

BW3 = createMask(e3,h\_im);

figure, imshow(BW3);

H = fspecial('unsharp');

J3 = roifilt2(H,img,BW3);

figure, imshow(J3);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%---------- ROI OF POLYGON SHAPE -----------%

c = [222 272 300 270 221 194];

r = [21 21 75 121 121 75];

e4 = impoly(gca,[67 47;67 97;121 125;167 95;167 46;121 19]);

BW4 = createMask(e4,h\_im);

figure, imshow(BW4);

H = fspecial('unsharp');

J4 = roifilt2(H,img,BW4);

figure, imshow(J4);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%---------- ROI OF ELLIPSE SHAPE -----------%

e5 = imellipse(gca,[38 30 150 100]);

BW5 = createMask(e5,h\_im);

figure, imshow(BW5);

H = fspecial('unsharp');

J5 = roifilt2(H,img,BW5);

figure, imshow(J5);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%---------- ROI OF FREEHAND SHAPE -----------%

e6 = imfreehand(gca);

% ---- wait on interactive freehand selection window-----%

pos=wait(e6);

%------double click to resume opertion after selection---%

BW6 = createMask(e6,h\_im);

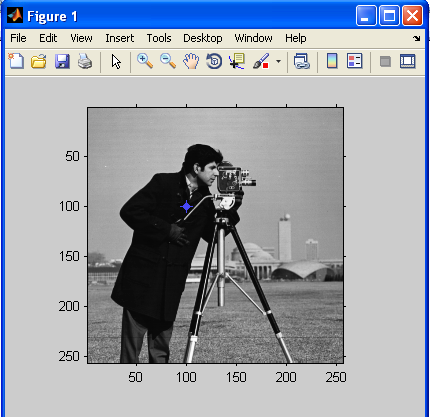
figure, imshow(BW6);

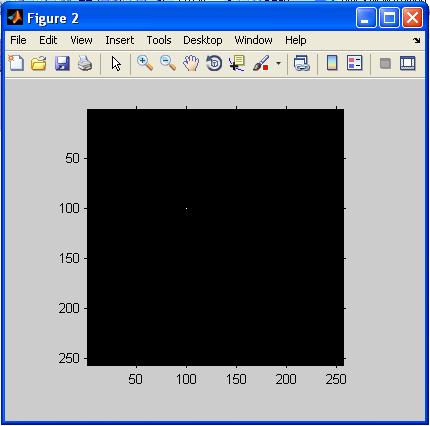
H = fspecial('unsharp');

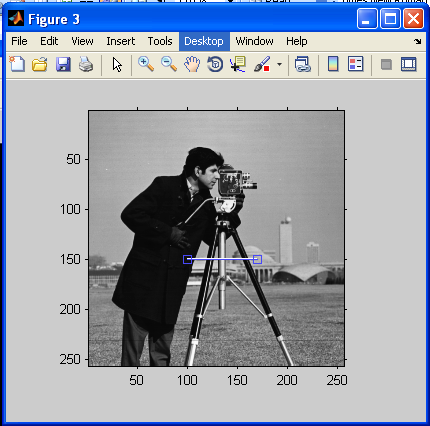
J6 = roifilt2(H,img,BW6);

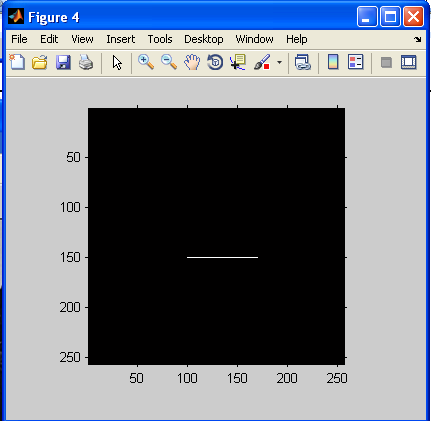
figure, imshow(J6);

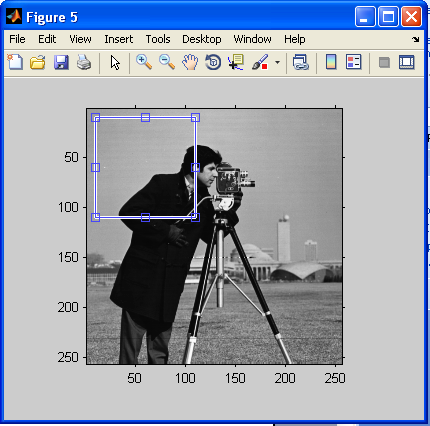
* Output:

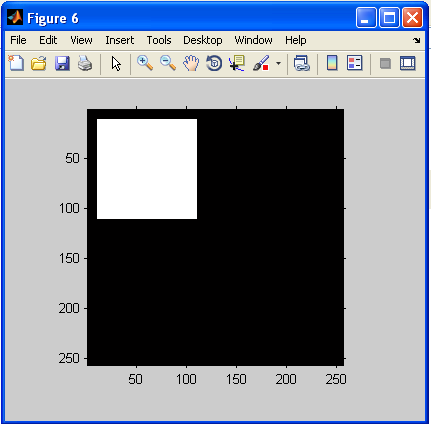


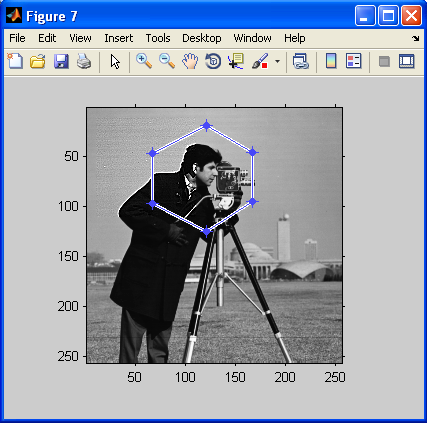


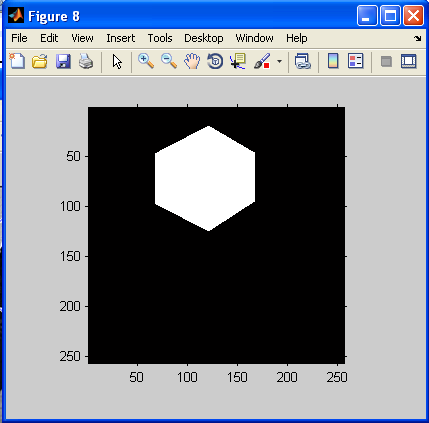


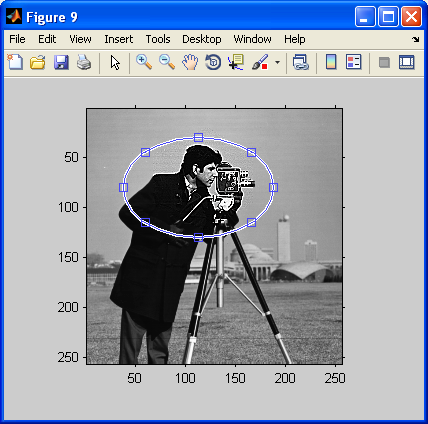


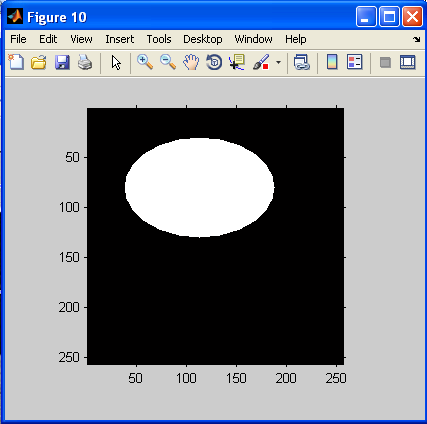


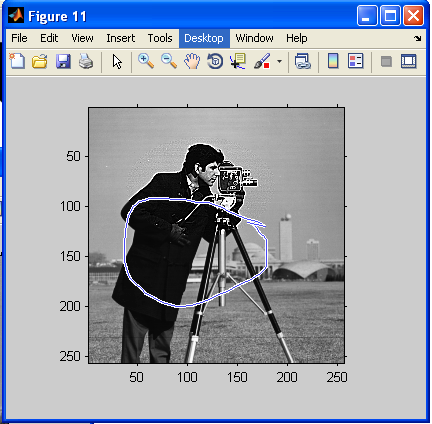


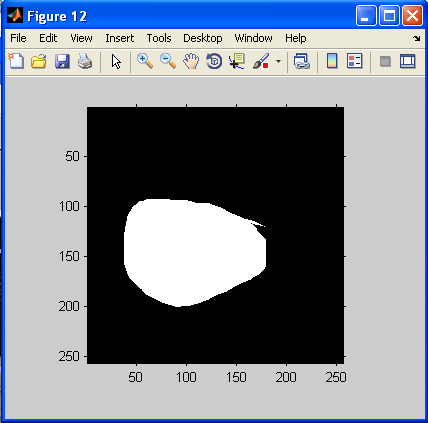


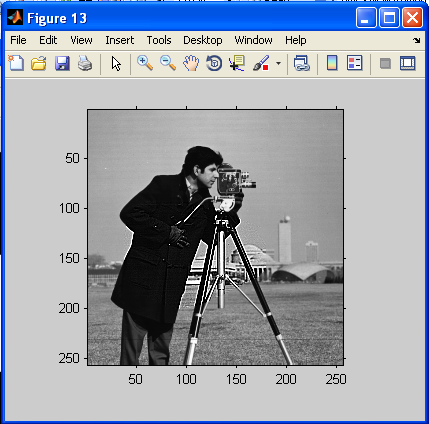












**ASSIGNMENT: 5**

* Question: The various mouse functions can be accessed by setting up the AX register with different values (service number) and issuing interrupt number 51. The functions are listed below,

1. Reset mouse and get status
2. Hide mouse pointer
3. Show mouse pointer

* Code:

#include<graphics.h>

#include<stdio.h>

#include<conio.h>

#include<dos.h>

union REGS in,out;

int callmouse()

{

in.x.ax=0;

int86(51,&in,&out);

if(out.x.ax==0)

printf("\nmouse not avelable");

else

{

printf("\nmouse avelable");

printf("\npress any key to continue:");

getch();

}

in.x.ax=1;

int86(51,&in,&out);

return 1;

}

void mouseposi(int &xpos,int &ypos,int &click)

{

in.x.ax=3;

int86(51,&in,&out);

click=out.x.bx;

xpos=out.x.cx;

ypos=out.x.dx;

}

int mousehide()

{

in.x.ax=2;

int86(51,&in,&out);

return 1;

}

void setposi(int &xpos,int &ypos)

{

in.x.ax=4;

in.x.cx=xpos;

in.x.dx=ypos;

int86(51,&in,&out);

}

int main()

{

int x,y,cl,a,b;

clrscr();

int g=DETECT,m;

initgraph(&g,&m,"c:\tc\bgi");

a=100;

b=400;

setposi(a,b);

printf("\npress any key to check mouse status...");

getch();

callmouse();

do

{

mouseposi(x,y,cl);

gotoxy(10,9);

printf("\n\tMouse Position is: %d,%d",x,y);

printf("\n\tClick: %d",cl);

printf("\n\tPress any key to hide the mouse");

}while(!kbhit());

getch();

mousehide();

printf("\n\n\tPress any key to Exit");

getch();

}