**/\* Program No. :**

**Aim : WAP for 2D transformations i.e. translating, scaling and rotating of a given object of n vertices.**

**\*/**

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

#include<math.h>

#include<graphics.h>

#define no\_ver 5

int x[no\_ver],y[no\_ver],vertices,i;

void draw()

{

for(i=0;i<(vertices-1);i++)

line(x[i],y[i],x[i+1],y[i+1]);

line(x[0],y[0],x[vertices-1],y[vertices-1]);

}

void initial\_draw()

{

printf("\n\nEnter the number of vertices : ");

scanf("%d",&vertices);

if(vertices>no\_ver)

{

printf("\n\n\tMaximum number of vertices are %d",no\_ver);

printf("\n\tPress any key to exit.");

getch();

exit(1);

}

else

{

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

{

printf("\n\nEnter the coordinates of vertice %d : ",i+1);

printf("\n\t\tx : ");

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

printf("\t\ty : ");

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

}

clrscr();

setcolor(2);

draw();

getch();

}

}

void translate(int tx,int ty)

{

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

{

x[i]+=tx;

y[i]+=ty;

}

}

void translation()

{

int tx,ty;

printf("\nEnter the translation vector, x : ");

scanf("%d",&tx);

printf("\nEnter the translation vector, y : ");

scanf("%d",&ty);

clrscr();

setcolor(2);

draw();

setcolor(4);

translate(tx,ty);

draw();

}

void scale(int fx,int fy,float sx,float sy)

{

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

{

x[i]=(x[i]\*sx)+(fx\*(1-sx));

y[i]=(y[i]\*sy)+(fy\*(1-sy));

}

}

void scaling()

{

int fx,fy;

float sx,sy;

printf("\nEnter the scaling vector, x : ");

scanf("%f",&sx);

printf("\nEnter the scaling vector, y : ");

scanf("%f",&sy);

printf("\nEnter the coordinates of fixed point, x : ");

scanf("%d",&fx);

printf("\nEnter the coordinates of fixed point, y : ");

scanf("%d",&fy);

clrscr();

setcolor(2);

draw();

setcolor(4);

scale(fx,fy,sx,sy);

draw();

}

float a\_value(float angle)

{

return(angle\*3.14/180);

}

void rotate(int px,int py,float angle)

{

int tempx,tempy;

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

{

tempx=x[i];

tempy=y[i];

x[i]=px+((tempx-px)\*cos(a\_value(angle)))-((tempy-py)\*sin(a\_value(angle)));

y[i]=py+((tempx-px)\*sin(a\_value(angle)))+((tempy-py)\*cos(a\_value(angle)));

}

}

void rotation()

{

int px,py;

float angle;

printf("\nEnter the rotation angle : ");

scanf("%f",&angle);

printf("\nEnter the coordinates of fixed point, x : ");

scanf("%d",&px);

printf("\nEnter the coordinates of fixed point, y : ");

scanf("%d",&py);

clrscr();

setcolor(2);

draw();

setcolor(4);

rotate(px,py,angle);

draw();

}

void main()

{

int choice1;

char choice2;

int gdriver = DETECT, gmode, errorcode;

initgraph(&gdriver, &gmode, "C:\\TC3.0\\BGI");

errorcode = graphresult();

if (errorcode != grOk)

{

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

printf("Press any key to exit.");

getch();

exit(1);

}

do

{

clrscr();

printf("\n\n\t\t\t\tMenu");

printf("\n\n\t1. Translation");

printf("\n\t2. Scaling");

printf("\n\t3. Rotation");

printf("\n\t4. Exit");

printf("\n\n\t\tEnter your choice (1-4) : ");

scanf("%d",&choice1);

switch(choice1)

{

case 1:clrscr();

initial\_draw();

clrscr();

translation();

getch();

clrscr();

printf("\n\n\t\tWant to continue (y/n) : ");

scanf("%s",&choice2);

break;

case 2:clrscr();

initial\_draw();

clrscr();

scaling();

getch();

clrscr();

printf("\n\n\t\tWant to continue (y/n) : ");

scanf("%s",&choice2);

break;

case 3:clrscr();

initial\_draw();

clrscr();

rotation();

getch();

clrscr();

printf("\n\n\t\tWant to continue (y/n) : ");

scanf("%s",&choice2);

break;

case 4:exit(0);

default:printf("\n\n\t\tIncorrect choice.\n\t\tPlease try again.");

getch();

choice2='y';

}

}while(choice2=='y'||choice2=='Y');

getch();

closegraph();

}

**/\***

**Name : Rohit Aggarwal**

**Roll No. : 7CS-097**

**\*/**