**EX NO: 4**

**DATE: MOVIE -1**

**AIM**

To write a program for short movie with the help of 2D Primitive Drawing, 2D Transformation and 2D Line Clipping Algorithm.

**ALGORITHM**

* Stick man and the board is drawn using 2D Primitive Drawing.
* The stick strikes the board and passes through the centre of board.
* While passing through the centre of board, the stick is clipped with the help of 2D Line Clipping Algorithm.
* After striking the first board, the second board is enlarged with the help of 2D transformation (scaling).

**CREATION OF OBJECTS**

* **Stick Man**
* The face of the stick man is drawn using Bresenham’s circle algorithm.
* The body, right leg and right hand are drawn using Bresenham’s line drawing algorithm.
* The left leg and left hand is drawn using DDA line drawing algorithm.
* **Board**
* The board is drawn as concentric circle with help of Bresenham’s circle algorithm with same centre point and different radii.

**Bresenham’s Circle Drawing Algorithm-To draw the face of the Stick Man and concentric circles of Board**

**Function name :** circlemidpoint(xcen,ycen,radius)

**Input parameters:** xcen,ycen –centre point of face/board.

radius- radius of face/board.

1. Initialize the following variables  
    x=0

y=radius

p= (3-2)\*radius

1. Repeat the steps 7 to 9 till the value of x is less than radius and increment the value of x by1 each time.
2. If the value of p is less than 0
3. Calculate the value of p=p+(4\*x)+6
4. Else
5. Calculate the value of p=p+10+4\*(x-y)
6. Plot the pixel to display the face using put pixel function.

**Bresenham’s Line Drawing Algorithm-To draw the hand and leg of the Stick Man**

**Function name :** linebres(xa, ya, xb, yb)

**Input parameters**: xa, ya, xb, yb- end points of the line.

1. Initially calculate doublex=xend-x and doubley=yend-y.
2. To draw the body of stick man, p (slope of the centre line) is set to 0 since both the

x-end points of the line is same.

1. For right hand and right leg calculate p=2\*doubley-doublex.
2. Check Whether if xa is greater than xb,then do the following assignments
3. x=xb
4. y=yb
5. xend=xa
6. Else vice versa.
7. Repeat the step 8 to 9 till x is less than xend and increment the value of x by 1 each time.
8. If value of p is less than zero, calculate p=p+2\*x\*doubley.
9. Otherwise increment y by 1,then calculate p=p+ 2\*doubley‐2\*doublex.
10. Plot the pixel to draw the line using put pixel function.

**MOVEMENT OF OBJECTS**

* The stick while passing through the board is clipped with 2D Cohen Sutherland Line Clipping Algorithm

**2D Cohen Sutherland Line Clipping Algorithm to make the stick invisible when it passes through the centre of board**

**Function name:** clip(x0, y0, x1, y1, xwmin, ywmin, xwmax, ywmax)

**Input parameters**: x0, y0, x1, y1- end points of the line.

xwmin - minimum x-point of the clipping window .

ywmin - minimum y-point of the clipping window.

xwmax – maximum x-point of the clipping window.

ywmax – maximum y-point of the clipping window.

* Initially, set up the region code for top=0x1, bottom=0x2, right=0x4, left=0x8.
* Calculate the code0, code1 for the end points of the line with respect to the clipping window.
* If bitwise OR of code0 and code1 is set to 0, then line is accepted.
* Else if bitwise AND of code0 and code1 is not equal to 1, then the line is rejected.
* Else endpoints are in different regions, and then find one of the two points that is outside the viewport region.
* If code value specifies top region, then new (x,y) will be
  + x=x0+(x1-x0)\*(ywmax-y0)/(y1-y0);
  + y=ywmax;
* If code value specifies bottom region, then new (x,y) will be
  + x=x0+(x1-x0)\*(ywmin-y0)/(y1-y0);
  + y=ywmin;
* If code value specifies right region, then new (x,y) will be
* y=y0+(y1-y0)\*(xwmax-x0)/(x1-x0);
* x=xwmax;
* If code value specifies left region, then new (x,y) will be
* y=y0+(y1-y0)\*(xwmin-x0)/(x1-x0);
* x=xwmin;
* Set the new x, new y values to x0, y0 if codeout =code0 else set the values to x1,y1.
* Draw the line with new(x0, y0, x1, y1) coordinates.

**TRANSFORMATION OF OBJECT-Board is enlarged with 2D Transformation**

1. The size of the circle is enlarged with the scaling factor (s).
2. Scaling factor (s) is set as twice the radius.
3. With new scaling factor as radius of the circle ,circlemidpoint() is called and stick is also clipped while passing through the second board.

**CODE**

#include<graphics.h>

#include<stdio.h>

#include<conio.h>

#include<math.h>

void clip(int,int,int,int,int,int,int,int);

int round(float a)

{

return (int)a+0.5;

}

enum { TOP=0x1, BOTTOM=0x2, RIGHT=0x4, LEFT=0x8 };

//Code to draw the left leg and left hand using DDA line drawing algorithm

void line1(int x0,int y0,int xend,int yend)

{

int dx=xend-x0,dy=yend-y0,steps,k;

float xinc,yinc,x=x0,y=y0;

if(fabs(dx)>fabs(dy))

steps=fabs(dx);

else

steps=fabs(dy);

xinc=(float)dx/(float)steps;

yinc=(float)dy/(float)steps;

putpixel(round(x),round(y),RED);

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

{

x+=xinc;

y+=yinc;

putpixel(round(x),round(y),25);

}

}

//Code to draw the face of the stick man using Bresenham’s circle drawing algorithm

void circlemidpoint(int h,int k,int radius)

{

int p,x,y;

x=0;

y=radius;

p=3-2\*radius;

do

{

if(p<0)

{

p+=4\*x+6;

putpixel(h+x,k+y,25);

putpixel(y+h,x+k,25);

putpixel(-y+h,x+k,25);

putpixel(-x+h,y+k,25);

putpixel(-x+h,-y+k,25);

putpixel(-y+h,-x+k,25);

putpixel(y+h,-x+k,25);

putpixel(x+h,-y+k,25);

x++;

}

else

{

p+=4\*(x-y)+10;

putpixel(h+x,k+y,25);

putpixel(y+h,x+k,25);

putpixel(-y+h,x+k,25);

putpixel(-x+h,y+k,25);

putpixel(-x+h,-y+k,25);

putpixel(-y+h,-x+k,25);

putpixel(y+h,-x+k,25);

putpixel(x+h,-y+k,25);

x++;

y--;

}

}while(x<=y);

}

//Code to draw the centre line, right hand and right leg using Bresenham’s line drawing algorithm

void linebres(int xa,int ya,int xb,int yb)

{

int dx,dy,x,y,xend,yend,p;

dx=abs(xa-xb);

dy=abs(ya-yb);

if(dx==0)

p=0;

else

p=2\*dy-dx;

if(xa>xb)

{

x=xb;

y=yb;

xend=xa;

yend=ya;

}

else

{

x=xa;

y=ya;

xend=xb;

yend=yb;

}

while(x<xend || y<yend)

{

x=x+1;

if(p<0)

{

p=p+(2\*dy);

}

else

{

y+=1;

if(dx!=0)

{

p=p+2\*dy-2\*dx;

putpixel(x,y,25);

}

else

{

p=0;

putpixel(xa,y,25);

}

}

}

}

//main function

void main()

{

int hcen,xcen,ycen,loop\_var,ccenx=50,cceny=300,st\_st=ccenx+20,st\_end;

int gdriver=DETECT,gmode;

initgraph(&gdriver,&gmode,"C:\\turboc3\\BGI");

//code to draw the stick man

while(st\_st<(getmaxx()-20))

{

cleardevice();

st\_end=st\_st+40;

circlemidpoint(ccenx,cceny,20);

linebres(ccenx,cceny+20,ccenx,cceny+80);

linebres(ccenx,cceny+30,ccenx+20,cceny+55);

linebres(ccenx,cceny+80,ccenx+20,cceny+100);

line1(ccenx,cceny+30,ccenx-20,cceny+55);

line1(ccenx,cceny+80,ccenx-20,cceny+100);

if(st\_end<=ccenx+260)

{

circlemidpoint(ccenx+200,cceny+20,10);

circlemidpoint(ccenx+200,cceny+20,30);

circlemidpoint(ccenx+500,cceny+20,10);

circlemidpoint(ccenx+500,cceny+20,30);

clip(st\_st,cceny+20,st\_end,cceny+20,ccenx,cceny+20,ccenx+210,cceny+25);

}

else if(st\_end>=ccenx+230)

{

circlemidpoint(ccenx+500,cceny+20,10);

circlemidpoint(ccenx+500,cceny+20,30);

circlemidpoint(ccenx+500,cceny+20,50);

clip(st\_st,cceny+20,st\_end+40,cceny+20,ccenx+230,cceny+20,ccenx+510,cceny+25);

}

st\_st+=5;

delay(80);

}

getch();

closegraph();

}

/\*

void translation(int x1,int y1,int tx,int ty,int radius)

{

int newx,newy;

newx=x1-tx;

newy=y1+ty;

circle(newx,newy,radius);

line(newx,newy,newx,newy-radius);

//delay(1000);

// cleardevice();

//Rotation of line

// rotation(newx,newy,radius);

}

//Rotation of line by rotation angle(theta) with respect to the reference point i.e.centre of the circle

void rotation(int xc,int yc,int radius)

{

int loop,theta=360,xr=0,yr=0;

for(loop=0;loop<3;loop++)

{

circle(xc,yc,radius);

xr=floor(xc+(radius\*(cos(theta\*3.14/180))));

if(theta!=270)

yr=floor(yc+(radius\*(sin(theta\*3.14/180))));

else

yr=floor(yc+(radius\*(-1\*sin(theta\*3.14/180))));

line(xc,yc,xr,yr);

delay(1000);

cleardevice();

theta=theta-90;

}

}\*/

//Cohen-Sutherland Line Clipping Algorithm

void clip(int x0, int y0,int x1,int y1,int xwmin,int ywmin,int xwmax,int ywmax)

{

unsigned int code0,code1,codeout;

int accept=0,done=0,x,y;

//Calculation of code for endpoints of line

code0=calcode(x0,y0,xwmin,ywmin,xwmax,ywmax);

code1=calcode(x1,y1,xwmin,ywmin,xwmax,ywmax);

do

{

if(!(code0 | code1))

{

accept=1;

done=1;

}

else if(code0 & code1)

done = 1;

else

{

codeout=code0?code0:code1;

if(codeout & TOP)

{

x=x0+(x1-x0)\*(ywmax-y0)/(y1-y0);

y=ywmax;

}

else if( codeout & BOTTOM)

{

x=x0+(x1-x0)\*(ywmin-y0)/(y1-y0);

y=ywmin;

}

else if ( codeout & RIGHT)

{

y=y0+(y1-y0)\*(xwmax-x0)/(x1-x0);

x=xwmax;

}

else

{

y=y0+(y1-y0)\*(xwmin-x0)/(x1-x0);

x=xwmin;

}

if(codeout==code0)

{

x0=x;

y0=y;

code0=calcode(x0,y0,xwmin,ywmin,xwmax,ywmax);

}

else

{

x1=x;

y1=y;

code1=calcode(x1,y1,xwmin,ywmin,xwmax,ywmax);

}

}

} while(done==0);

if(accept)

line(x0,y0,x1,y1);

}

//Calculation of code for the endpoints of line with respect to clipping window

int calcode (int x,int y,int xwmin,int ywmin,int xwmax,int ywmax)

{

int code =0;

if(y> ywmax)

code|=TOP;

else if( y<ywmin)

code|=BOTTOM;

else if(x > xwmax)

code|=RIGHT;

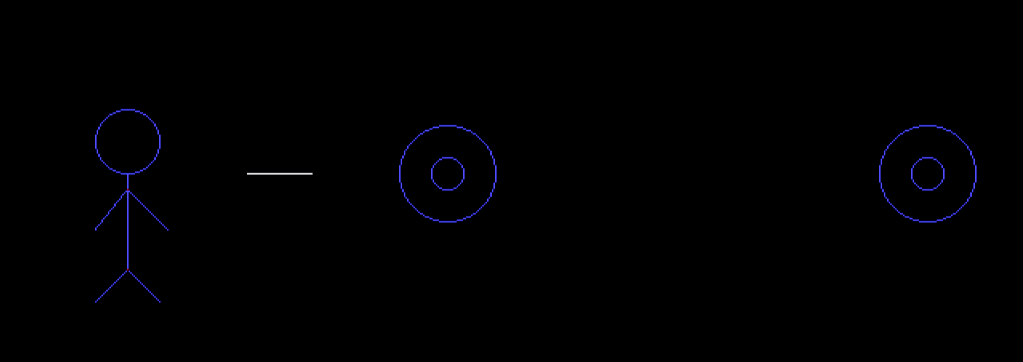
else if ( x< xwmin)

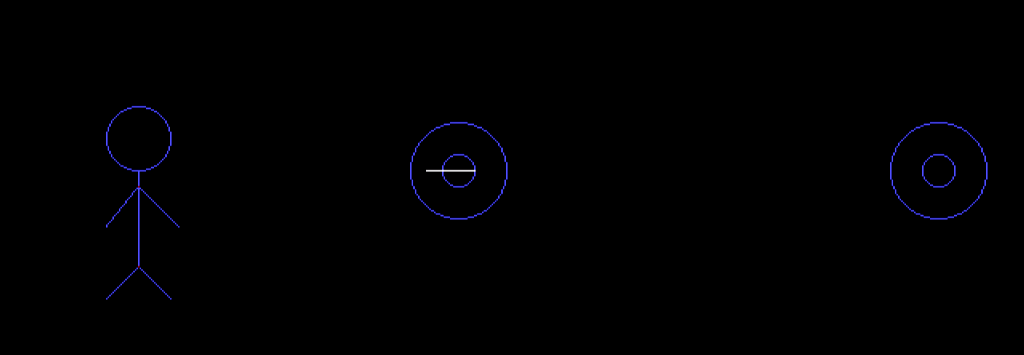
code|=LEFT;

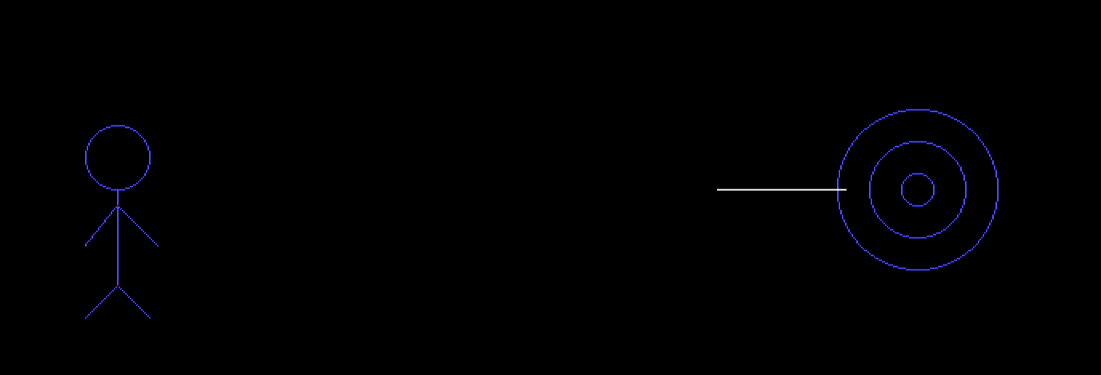
return(code);

}

**OUTPUT**

****

****

****

|  |  |  |
| --- | --- | --- |
| **ALGORITHM** | **CODE** | **OUTPUT** |
|  |  |  |

**RESULT**

Thus the program for movie is written and executed successfully with the help of 2D Primitive Drawind,2D Transformation and 2D Line Clipping Algorithm.