COMPUTER GRAPHICS

NAME : SHRIRANG. R. MHALGI

CLASS : S.E.

DIV : B

ROLL NO : 222006

PROBLEM STATEMENT :

Write a java program to generate Hilbert curve using fractals.

CODE :

**package** cgg\_final;

**import** java.awt.\*;

**import** java.applet.\*;

**public** **class** Hilbert **extends** Applet {

**private** SimpleGraphics sg=**null**;

**private** **int** dist0=512, dist=dist0;

**public** **void** init() {

dist0 = 512;

resize ( dist0, dist0 );

sg = **new** SimpleGraphics(getGraphics());

}

**public** **void** paint(Graphics g) {

**int** level=4;

dist=dist0;

**for** (**int** i=level;i>0;i--) dist /= 2;

sg.goToXY ( dist/2, dist/2 );

HilbertA(level); // start recursion

}

**private** **void** HilbertA (**int** level) {

**if** (level > 0) {

HilbertB(level-1); sg.lineRel(0,dist);

HilbertA(level-1); sg.lineRel(dist,0);

HilbertA(level-1); sg.lineRel(0,-dist);

HilbertC(level-1);

}

}

**private** **void** HilbertB (**int** level) {

**if** (level > 0) {

HilbertA(level-1); sg.lineRel(dist,0);

HilbertB(level-1); sg.lineRel(0,dist);

HilbertB(level-1); sg.lineRel(-dist,0);

HilbertD(level-1);

}

}

**private** **void** HilbertC (**int** level) {

**if** (level > 0) {

HilbertD(level-1); sg.lineRel(-dist,0);

HilbertC(level-1); sg.lineRel(0,-dist);

HilbertC(level-1); sg.lineRel(dist,0);

HilbertA(level-1);

}

}

**private** **void** HilbertD (**int** level) {

**if** (level > 0) {

HilbertC(level-1); sg.lineRel(0,-dist);

HilbertD(level-1); sg.lineRel(-dist,0);

HilbertD(level-1); sg.lineRel(0,dist);

HilbertB(level-1);

}

}

}

**class** SimpleGraphics

{

**private** Graphics g = **null**;

**private** **int** x = 0, y = 0;

**public** SimpleGraphics(Graphics g) { **this**.g = g; }

**public** **void** goToXY(**int** x, **int** y) { **this**.x = x; **this**.y = y; }

**public** **void** lineRel(**int** deltaX, **int** deltaY) {

g.drawLine ( x, y, x+deltaX, y+deltaY );

x += deltaX; y += deltaY;

}

}

OUTPUT :

