

## Code:

### Bresenham's Circle :

*//bresenham's algorithm to draw a circle*

void MainWindow::on\_bresenhamCircle\_clicked()

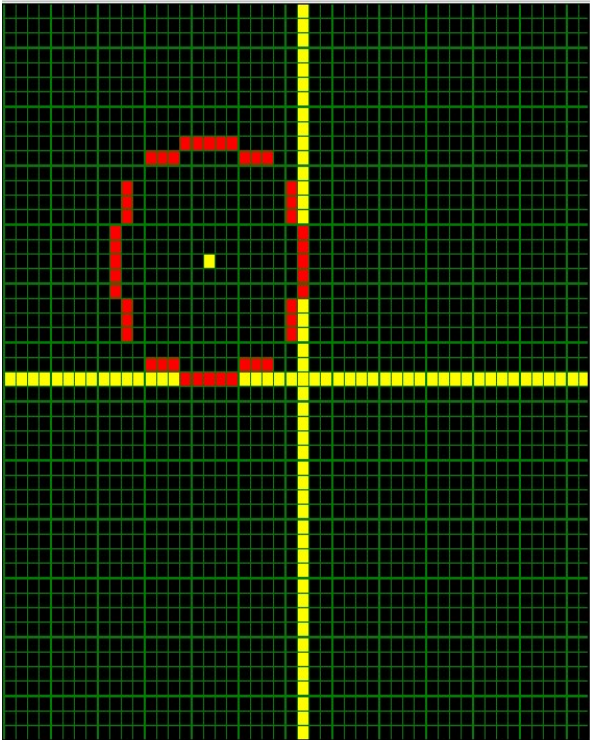
```
{
    int radius = ui->radiusSpinBox->value();
    p1.setX(ui->frame->x);
    p1.setY(ui->frame->y);
    int x0 = p1.x()/gridsize;
    int y0 = p1.y()/gridsize;
    x0 = x0*gridsize + gridsize/2;
    y0 = y0*gridsize + gridsize/2;
    int x = 0;
    int y = radius *gridsize;
    int p = (3 - 2*radius)*gridsize;
    auto start = high_resolution_clock::now();
    while(y > x) {
        point(x0 - x, y0 - y, 255, 0, 0);
        point(x0 + x, y0 - y, 255, 0, 0);
        point(x0 - x, y0 + y, 255, 0, 0);
        point(x0 + x, y0 + y, 255, 0, 0);
        point(x0 - y, y0 - x, 255, 0, 0);
        point(x0 + y, y0 - x, 255, 0, 0);
        point(x0 - y, y0 + x, 255, 0, 0);
        point(x0 + y, y0 + x, 255, 0, 0);

        x += gridsize;
        if(p <= 0) {
            p += 4*x + 6;
        } else {
            p += 4*(x-y) + 10;
            y -= gridsize;
        }
        delay(1);
    }
    auto end = high_resolution_clock::now();
    int executionTime = duration_cast<microseconds>(end - start).count();
    cout << "Execution Time for bresenham's circle drawing algorithm :- " << executionTime
    << "\n";
}
```

# Bresenham's Circle :

MainWindow

Santanu's Graphics Assignment



(451 X 451)

Reset

☒ Show Axes

Mouse Movement  
X : 22, Y : 5

Mouse Pressed  
X : -7, Y : 7

9

Set Grid

point1

point2

DDA Line

Bresenham Line

8

midpoint circle

bresenham circle

polar circle

0

0

midpoint ellipse

polar ellipse

X axis radius

Y axis radius

8 point fill

flood fill

boundary fill

set vertex

clear vertex

scan line

0

x translate

0

y translate

0.00

x scale

0.00

y scale

0.00

x shear

0.00

y shear