Lab 4

1.

1.1. i) The grid is filled; ii) A square in the bottom right corner of the last 20 squares is filled in; iii) All but the rightmost column is filled in, as it starts at -1 – outside the grid

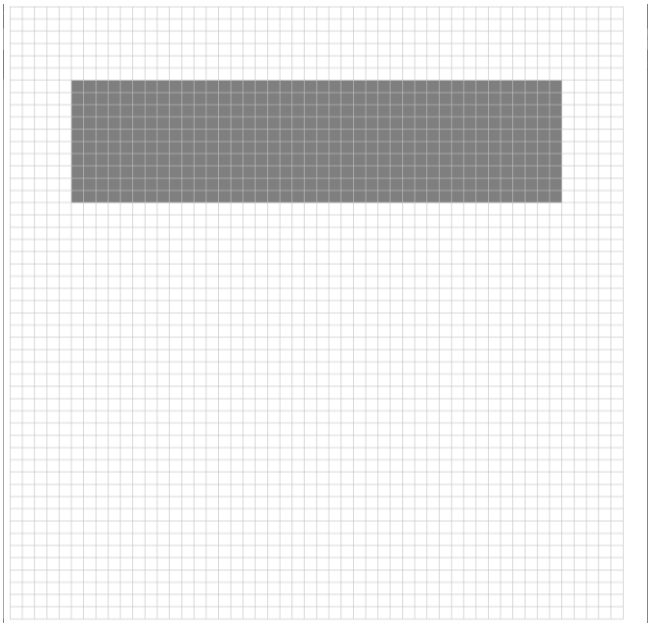
tsk_1 | Processing 3.5.3

File Edit Sketch Debug Tools Help

Java ▼

tsk_1

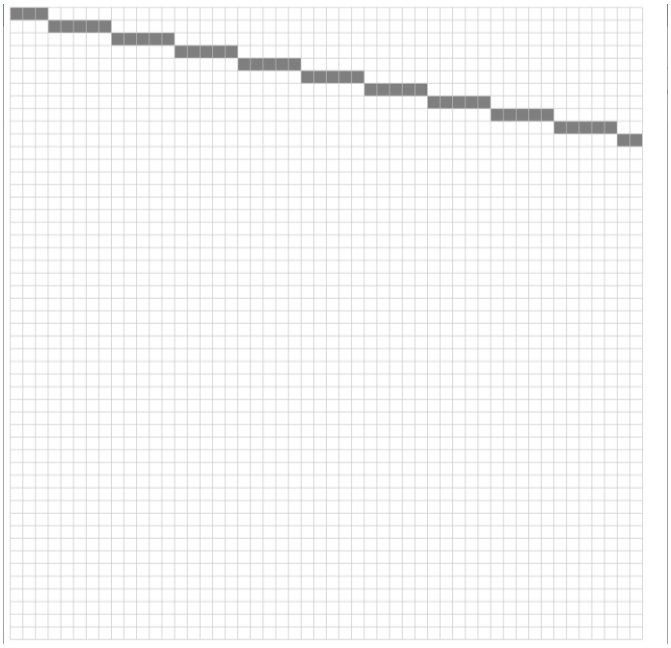
```
void setup() {
 size(1050, 1050);
 background(255);
 stroke(195);
 int[][] myNewData = new int[50][50];
 myClearData(myNewData);
 myRect(myNewData, 5, 6, 40 , 10);
 myDraw(myNewData);
}

void myDraw(int[][] array){
  for (int i = 0; i < array.length; i++) {
    for (int j = 0; j < array.length; j++) {
      if (array[j][i] == 1) {
        fill(127);
      }
      else {
        fill(255);
      }
      rect(10+20*i, 10+20*j, 20, 20);
    }
  }
}

int[][] myClearData(int[][] array) {
  for (int i = 0; i < array.length; i++) {
    for (int j = 0; j < array.length; j++){
      array[i][j] = 0;
    }
  }
  return array;
}

int[][] myRect(int[][] array, int xPos, int yPos, int wid, int hei) {
  for (int i = xPos; i < xPos + wid && i < array.length*array.length; i++) {
    for (int j = yPos; j < yPos + hei && j < array.length*array.length; j++) {
      if (j >= 0 && i >= 0) {
        array[j][i] = 1;
      }
    }
  }
  return array;
}
```



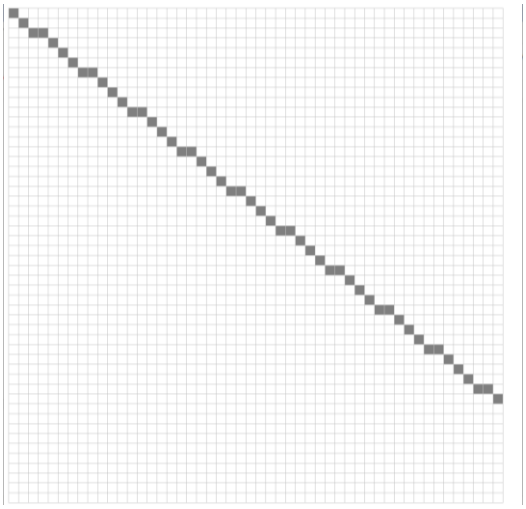
2.

```
void setup() {
 size(1050, 1050);
 background(255);
 stroke(195);
 int[][] myNewData = new int[50][50];
 myClearData(myNewData);
 myLine(myNewData, 0, 0, 50 , 10);
 myDraw(myNewData);
}

void myDraw(int[][] array){
  for (int i = 0; i < array.length; i++) {
    for (int j = 0; j < array.length; j++) {
      if (array[j][i] == 1) {
        fill(127);
      }
      else {
        fill(255);
      }
      rect(10+20*i, 10+20*j, 20, 20);
    }
  }
}

int[][] myClearData(int[][] array) {
  for (int i = 0; i < array.length; i++) {
    for (int j = 0; j < array.length; j++){
      array[i][j] = 0;
    }
  }
  return array;
}

int[][] myLine(int[][] array, int x1, int y1, int x2, int y2) {
  double xn = x2 - x1;
  double yn = y2 - y1;
  double m = yn/xn;
  double c = y1 - m * x1;
  for (int i = x1; i < x2; ++i) {
    double j = m * i + c;
    if (i < array.length && (int) Math.round(j) < array.length) {
      array[(int) Math.round(j)][i] = 1;
    }
  }

  return array;
}
```

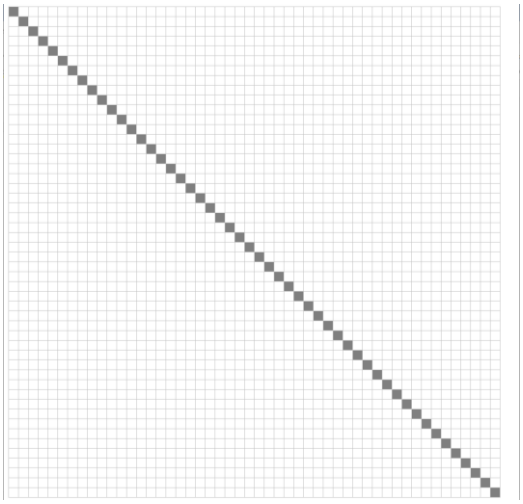
2.1. A diagonal line is drawn from (0, 0) to (50, 10)



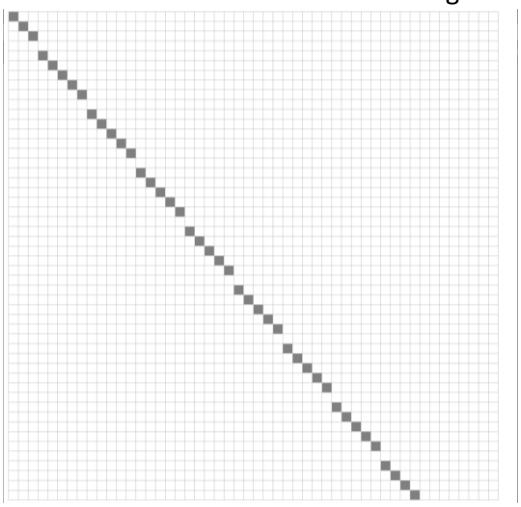
2.2. A diagonal line is drawn from (0, 0) to (50, 40)



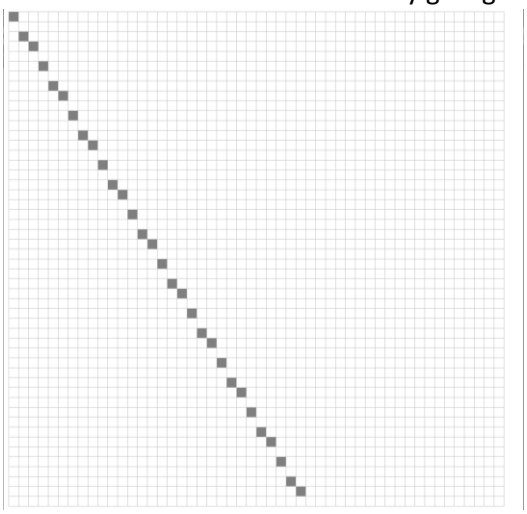
2.3. A diagonal line is drawn from (0, 0) to (50, 50) this is straighter than the others as it has a direct diagonal path between the two coordinates



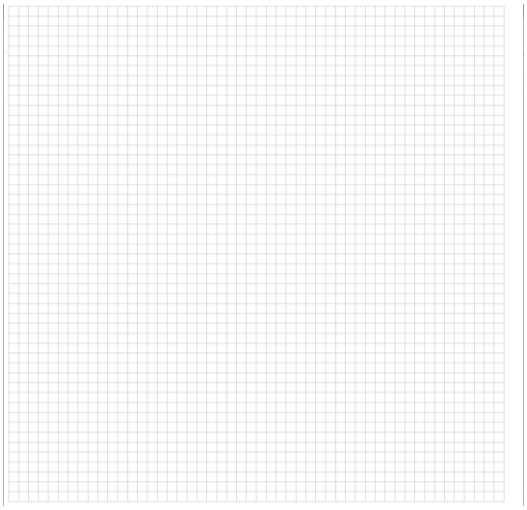
2.4. A diagonal line is drawn from (0, 0) to a point outside the grid. This causes the line to be stretched and so some of the points are missing. This could be fixed by giving the Y-axis calculations a more accurate rounding



2.5. A diagonal line is drawn from (0, 0) to (30, 50). Because the Y-axis calculations have a decimal value rounded to the nearest whole number, some of the points are missing from the line. This could also be fixed by giving the Y-axis calculations a more accurate rounding



2.6. The line starts drawing at the edge of the grid, but the algorithm used does not account for a line going in a negative direction, so no line is drawn. This could be fixed by determining which point is the smallest, and then drawing from that point



3.

3.1. It doesn't look like a good circle as there are many missing pixels and some erroneous ones. I could improve the circle by giving a more accurate rounding or using a finer/larger grid

3.2. The problem with the naïve circle algorithm is that circles are not very well represented with such large pixel sizes – they require finer detail for their curved sides

```
void setup() {
 size(1050, 1050);
 background(255);
 stroke(195);
 int[][] myNewData = new int[50][50];
 myClearData(myNewData);
 myCircle(myNewData, 20, 20, 10);
 myDraw(myNewData);
}

void myDraw(int[][] array){
  for (int i = 0; i < array.length; i++) {
    for (int j = 0; j < array.length; j++) {
      if (array[j][i] == 1) {
        fill(127);
      }
      else {
        fill(255);
      }
      rect(10+20*i, 10+20*j, 20, 20);
    }
  }
}

int[][] myClearData(int[][] array) {
  for (int i = 0; i < array.length; i++) {
    for (int j = 0; j < array.length; j++){
      array[i][j] = 0;
    }
  }
  return array;
}

int[][] myCircle(int[][] array, int xPos, int yPos, float rad) {
  for (int i = 0; i < array.length; i++) {
        array[(int)(yPos+sin(i)*rad)][(int)(xPos+cos(i)*rad)] = 1;
      }
  return array;
}
```