```java
/**
 *Elliot Duncan
 *Horton 7th
 *5/9/24
 *
 *@(#)Seed.java
 *
 *Stores the starting state and reads from a file
 */

import java.io.File;
import java.io.FileNotFoundException;
import java.util.Arrays;
import java.util.Scanner;
import java.util.stream.Collectors;

public class Seed implements Comparable {
  private int[][] cells;
  private String name;
  /**
   *Constructs a new seed given a filename
   *@param filePath, the path to the file, usually "Patterns/_____"
   */
  public Seed(String filePath) {
    this.cells = decode(readFile(filePath));
    name = filePath;
  }
  /**
   * returns the size on the X axis of the seed (the horizontal distance between
   * the left-most and right-most points)
   * @return the X size
   */
  public int getSizeX() { return cells[0].length; }
  /**
   * returns the size on the Y axis of the seed (the vertical distance between
   * the top-most and bottom-most points)
   * @return the Y size
   */
  public int getSizeY() { return cells.length; }

  /**
   * returns the number of cells, or the count of the cells whose state > 0
   * @return the number of cells
   */
  public int getNumOfCells() {
    int count = 0;
    for (int[] i : cells)
      for (int j : i)
        count += j;
    return count;
```

```
}

/**
 *Returns the filename of the seed that was created.
 *@return the name of the current seed.
 */
public String getName() { return name; }
/**
 *Returns the Array that the seed is composed of
 *@return the seed
 */
public int[][] getCells() { return cells; }

/**
 *String representations of the
 *object, in the form of an array with the state
 *of each cell and '#' is an alive cell
 *@return the string representation
 */
@Override
public String toString() {
  return Arrays.asList(cells)
       .parallelStream()
       .map(s -> {
         return Arrays.stream(s)
              .mapToObj(c -> { return (c == 1) ? "#" : " "; })
              .collect(Collectors.joining(""));
       })
       .collect(Collectors.joining("\n"));
}

/**
 *Seeds are equal if they share a name
 *@return whether they are equal
 */
@Override
public boolean equals(Object o) {
  Seed other = (Seed)o;
  return this.name.equals(other.name);
}

/**
 * Compares the Area of the bounding box of the seed
 *@return comparison of the Areas
 */
@Override
public int compareTo(Object o) {
  Seed other = (Seed)o;
  int thisArea = (this.getSizeY() * this.getSizeX());
  int otherArea = (other.getSizeY() * other.getSizeX());
```

```
  return Integer.compare(thisArea, otherArea);
}

private String readFile(String filePath) {
  try {
    // read seed from file
    Scanner input = new Scanner(new File(filePath));
    String rle = "";

    // collect to string
    while (input.hasNext())
      rle += input.nextLine() + '\n';

    input.close();

    // remove lines starting with '#'
    rle = rle.replaceAll("#.*\n", "");

    // initialize size value from header lines
    String headerX = rle.split(",")[0].substring(4);
    String headerY = rle.split(",")[1].substring(5);

    // return the pixel data only
    return rle
        .replaceFirst(".*\n", "") // Remove the header line
        .replaceAll("\n", "");    // concatenates     into one string

  } catch (FileNotFoundException e) {
    System.err.println("Try a different filepath!\n" + e);
    return null;
  }
}
private int[][] decode(String encoding) {
  int index = 0;

  String test = "";

  // The actual decoding,     the     cells collect into 'test'
  while (index < encoding.length()) {
    int runLength = 1;

    String num = "";
    while (Character.isDigit(encoding.charAt(index))) {
      num += encoding.charAt(index++);
      runLength = Integer.parseInt(num);
    }

    char token = encoding.charAt(index++);
    token = token == '$' ? '\n' : token;
```

```
      for (int i = 0; i < runLength; i++)
        test += token;
    }
    test = test.substring(0, test.length() - 1);
    String[] s = test.split("\n");
    int maxLen = Arrays.stream(s).map(String::length).reduce(0, Math::max);
    int[][] cellsTemp = new int[s.length + 1][maxLen];

    // 2d array representation
    for (int i = 0; i < s.length; i++) {
      for (int j = 0; j < s[i].length(); j++) {
        char c = s[i].charAt(j);
        if (c == 'o')
          cellsTemp[i][j] = 1;
        else
          cellsTemp[i][j] = 0;
      }
    }
    return cellsTemp;
  }
}
```