

1 Exercises

1. For the first program you will ask the user for the number of rows they want to display. Create a two dimensional array of integers with that many rows and that many columns. So if the user input 5 you would create a 5×5 array. Populate the first column of the array with 1 in each cell. Starting with the second row do the following to each row of the array. For each row, start in the second column and replace the entry in each cell with the sum of the entries directly above it and the one above and to the left. For example, if we have a row 1, 5, 7, 2, then the next row would be 1, 6, 12, 9.

1	5	7	2
1	6	12	9

Once this is done print out the array to the screen, but only print the entries that are not 0. An example run with 10 rows is below.

Input the number of rows: 10

```

1
1      1
1      2      1
1      3      3      1
1      4      6      4      1
1      5     10     10     5      1
1      6     15     20     15     6      1
1      7     21     35     35     21     7      1
1      8     28     56     70     56     28     8      1
1      9     36     84    126    126     84     36     9      1

```

2. This exercise is very similar to the last one. For this program you will ask the user for the number of rows they want to display. Create a two dimensional array of integers with that many rows and that many columns. Populate the first column of the array with 1 in each cell. Starting with the second row do the following to each row of the array. For each row, start in the second column and replace the entry in each cell with the sum of the entries directly above it and the one above and to the left but this time take the sum mod 2, meaning that you will calculate the sum and then take the remainder when you divide by 2 (remember the % operator). For example,

1	1	0	1
1	0	1	1

Now print the array out as follows, if the array entry is a 1 print out an asterisks (*) and if the array entry is 0 print out a single space.

2 Extra Credit

Take the running program from number 2 above and do the following to it.

1. Add in the following import statements.

```
import java.awt.Color;
import java.awt.image.BufferedImage;
import java.io.File;
import javax.imageio.ImageIO;
```

2. Add in the following method.

```
public static void exportImage(int A[][]) {
    int rows = A.length;
    int cols = A[0].length;

    BufferedImage bi = new BufferedImage(cols, rows, BufferedImage.TYPE_INT_RGB);

    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            Color col;
            if (A[i][j] == 1)
                col = new Color(255, 255, 255);
            else
                col = new Color(0, 0, 0);

            bi.setRGB(j, i, col.getRGB());
        }
    }

    try {
        File outputfile = new File("sierpinski.png");
        ImageIO.write(bi, "png", outputfile);
    } catch (Exception e) {
    }
}
```

3. In the main, after the array has been populated with 1's and 0's do not print out the '*', instead call the exportImage method.
4. Run the program on the array size 1024. Look in the project folder for the file sierpinski.png and open it up in an image viewer. Now rerun the program with larger array sizes and after each run open the file sierpinski.png in an image viewer.

3 Submit

1. For Number 1 just submit just the Java code file containing the program and a Word, LibreOffice, or text file containing at least 3 runs of the program with different array sizes.
2. For Number 2 submit the Java code file containing the program and a Word, LibreOffice, or text file containing runs using 4, 8, 16, 32, 64, and 128 rows.
3. For the extra credit just submit the Java code file of the program.