Project 3: Drawing Shapes

Learning Objectives

- Write a program in Java that utilizes variables, assignments, expressions, inputs, outputs, decision statements, loops, nested loops, and various nested statements.
- Design and implement an algorithm to solve a problem.
- Apply knowledge of compiling, running, testing, and debugging a Java program.
- Use good programming style and standards.

Introduction

In this project, we will demonstrate our knowledge of loops, decision statements, and nested statements to draw shapes. This project will consist of drawing rectangles, triangles, hexagons, octagons, and pentagons exactly as shown in the examples provided at the end of this document. Your program should be in class called **`DrawingShapes`** and stored in a file called **`DrawingShapes.java`**.

Program Requirements

1. You will need to comment each section of your source code that draws a shape. For example, you need to include comments for the section of source code that draws a rectangle, comments for the section of code that draws a triangle, and so forth for each shape required in this project.
2. All instructions in this document must be followed in order for full credit to be awarded. Following the instructions is a vital part to this and all programming assignments.
3. After receiving input from a user, your program should draw each shape using the correct number of space character(s), * character(s), and newline character(s) as shown in the Examples section. The space character in Java has the Unicode/ASCII value of 32. The space character in Java is ***NOT*** the same character as the null character, and null characters are ***NOT*** allowed to be used in your program. The use of a null character in your program may result in a failing grade on this project.
4. Your program must follow the examples provided in the Examples section, and your program's I/O (input and output) must match the examples exactly (otherwise, points will be deducted). If your program does not produce the same output as our examples, then your program has a bug, and you'll need to fix any bugs using any debugging techniques discussed this semester. There should be **NO** spaces to right of any *'s in your program's output, and the number of spaces in front of the *'s do matter in order to output the shape correctly based on the examples. After drawing the last * in the shape, your program should print a single newline character.
5. Your program must handle all error cases as shown in the examples. The valid shapes (`r t h o p`) are shown in the prompts in the examples, and any other input for shapes are considered as invalid. The user input for drawing an octagon is a lowercase `o`, the letter before p in the English alphabet. A length is valid if and only if it is greater than 1, and a height is valid if and only if it is greater than 1. Also, some shapes do not require height

as an input, and these shapes should not prompt the user for a height. Invalid inputs must be handled as shown in the examples.

6. For all sets of valid user input, your program must correctly draw the shape and dimension(s).

Additional Requirements

These are things that make the graders lives easier, and ultimately, you will see in the real world as well. Remember the teaching staff does not want to touch your code after they gave you requirements; they want to see the perfect results they asked for! Here is a checklist of things you can **lose points** for:

- (-3 points) If the source file(s)/class(es) are named incorrectly (case matters!)
- (-3 points) If your source file(s) have a package declaration at the top
- (-3 points) If any source file you submit is missing your Statement of Academic Honesty at the top of the source file. All submitted source code files must contain your Statement of Academic Honesty at the top of each file.
- (-3 points) If you have more than one instance of Scanner in your program. Your program should only have one instance of Scanner.
- (-3 points) Inconsistent I/O (input/output) that does not match our instructions or examples exactly (unless otherwise stated in an assignment's instructions). Your program's I/O (order, wording, formatting, etc.) must match our examples and instructions.
- (-6 points) Late penalties will be deducted.
- (-30 points) If the source file(s) are not submitted before the specified deadline's late period ends (48 hours after the deadline) or if they do not compile.
- If your (-3 points) comments or (-3 points) variables are "lacking"
    - Here, "lacking" means that you or a TA can find **any** lines of code or variables that take more than 10 seconds to understand, and there is no comment, or the variable name does not make sense (variable names like **b**, **bb**, **bbb**, etc. **will almost never be acceptable)**
- (-3 points) Indentation is not consistent throughout your source code
    - Refresh your memory of indentation patterns in chapter 2 in the course textbook
    - Be careful of a combination of tabs and spaces in your files (use one or the other)!
- (-30 points) For this assignment, you are **NOT** permitted to use a **break** or **continue** statement inside of the body of a loop, and using one of these statements in the body of a loop will result in a grade of zero on this assignment.

If any of the above do not make sense to you, talk to a TA or ask your lab instructor.

Project Grading

All projects are graded out of a possible 30 points. Programs that do not compile will receive a grade of zero. You must make absolutely certain your program compiles before submitting, and you must thoroughly test your program with many different inputs to verify that it is working correctly. This project will be graded for correctness and adherence to all project instructions and

requirements on various test cases, and you are responsible for testing that your program works with all valid inputs and the invalid inputs shown in the examples. The test cases used for grading may differ than the examples provided in this project. All instructions must be followed in order to receive full credit.

Project Submission

After you have tested, tested, and retested that your code compiles and run correctly on many test cases (those provided in this document and many test cases that you create on your own), submit the file **`DrawingShapes.java`** (and only that file) via eLC.

Examples

Your program should work correctly on the examples below, and it should draw various shapes correctly when the inputs are valid. All input and output should be formatted as shown when run (note: the new line(s) separating each example is not part of your program's I/O, it is in this document to show the difference between examples). Each example is separate run of a correctly working program. Some examples include invalid inputs, and some do not.

```
Enter a shape: r t h o p
r
Enter a length
3
Enter a height
4
Below is a 3 by 4 rectangle of *
***
***
***
***

Enter a shape: r t h o p
t
Enter a length
6
Below is a triangle with two side lengths of 6 *
     *
    ***
   *****
  *******
 *********
***********
```

```
Enter a shape: r t h o p
h
Enter a length
10
Below is a hexagon with side lengths of 10 *
          **********
         ************
        **************
       ****************
      ******************
     ********************
    **********************
   ************************
  **************************
 ****************************
******************************
 ****************************
  **************************
   ************************
    **********************
     ********************
      ******************
       ****************
        **************
         ************
          **********
```

```
Enter a shape: r t h o p
o
Enter a length
9
Below is an octagon with side lengths of 9 *
         *********
        ***********
       ************
      **************
     ****************
    ******************
   ********************
  **********************
 ************************
**************************
**************************
**************************
**************************
**************************
**************************
**************************
**************************
 ************************
  **********************
   ********************
    ******************
     ****************
      **************
       ************
        ***********
         *********

Enter a shape: r t h o p
p
Enter a length
3
Below is a pentagon with 4 side lengths of 3 *
  *
 ***
*****
*****
*****

Enter a shape: r t h o p
R
Invalid shape
Goodbye!
```

```
Enter a shape: r t h o p
Z
Invalid shape
Goodbye!

Enter a shape: r t h o p
r
Enter a length
1
Length must be greater than 1
Goodbye!

Enter a shape: r t h o p
r
Enter a length
2
Enter a height
1
Height must be greater than 1
Goodbye!

Enter a shape: r t h o p
h
Enter a length
0
Length must be greater than 1
Goodbye!

Enter a shape: r t h o p
o
Enter a length
4
Below is an octagon with side lengths of 4 *
   ****
  ******
 ********
**********
**********
**********
**********
 ********
  ******
   ****
```

```
Enter a shape: r t h o p
h
Enter a length
3
Below is a hexagon with side lengths of 3 *
  ***
 *****
*******
 *****
  ***

Enter a shape: r t h o p
r
Enter a length
10
Enter a height
4
Below is a 10 by 4 rectangle of *
**********
**********
**********
**********

Enter a shape: r t h o p
p
Enter a length
7
Below is a pentagon with 4 side lengths of 7 *
      *
     ***
    *****
   *******
  *********
 ***********
*************
*************
*************
*************
*************
*************
*************
```

```
Enter a shape: r t h o p
o
Enter a length
13
Below is an octagon with side lengths of 13 *
             *************
           ***************
          *****************
         *******************
        *********************
       ***********************
      *************************
     ***************************
    *****************************
   *******************************
  *********************************
 ***********************************
*************************************
*************************************
*************************************
*************************************
*************************************
*************************************
*************************************
*************************************
*************************************
*************************************
*************************************
 ***********************************
  *********************************
   *******************************
    *****************************
     ***************************
      *************************
       ***********************
        *********************
         *******************
          *****************
           ***************
             *************
```