UNIVERSITY OF ALBERTA

Augustana Computing Science 112

Assignment #7
Recursion & Interfaces

Goals:

- To be able to write recursive code in Java, including using wrappers and being able to reduce complexity by using wrappers properly.
- To differentiate between a recursive method and a recursive data type.
- To understand quicksort partitioning.
- To write code in a manner forced by an interface, as might happen in industry when a manager writes the interface and then the programmer creates the functioning code.

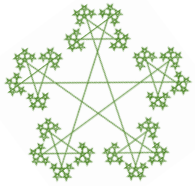
Instructions:

This assignment is to be completed ***individually***, i.e. without collaboration. Do not use loops anywhere in your solution: **no `while` nor `for`** reserved words! Any repetition in the code should be handled with recursion and auxiliary methods may be required to make that possible. Do not import any libraries. Place your code in a file called "**`Assign7.java`**".

You must have descriptive variable names, proper indentation, method headers and a file header. As well, your name, ID number, and date must be at the top of the file.

Reference:

Heise, Chapter 7



The Program to Complete:

Implement the `RecursionLabInterface` that is on eClass (ensure you have the proper class declaration line to prove this). You will be required to write two methods: `printTriangles` and `posBeforeNeg`.

`printTriangles(int size)` → `void`

This method prints multiple triangles, using stars, to standard output. The first triangle is of the given size, specified in the parameter. Each successive triangle is one size smaller, ending with a triangle of size one (a single star). If the size is 0 or negative, this method will print "Size too small" and move the output cursor to the next line.

For example, if the size is 5, the method will print, starting at the left-hand side:

```
    *
   * *
  * * *
 * * * *
* * * * *

   *
  * *
 * * *
* * * *
```

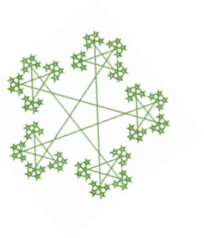
```
   *
  * *
* * *

  *
* *

*
```

Every printed line ends with "\r\n" and there is a single space between each star with no spaces at the end of any line. Java's `println` automatically inserts "\r\n" for line endings. Be careful if you make explicit line breaks – you must add both characters. A single blank line (with "\r\n") is between each set of triangles and there is no leading or ending blank line (though the output cursor ends at the beginning of the next line). The base of all triangles is at the left-hand side (with no spaces). You must match all spaces, letters, and line endings exactly.

`posBeforeNeg(int[] anArray)` → `void`



This method will change the contents of `anArray`, so that all positive integers precede all negative integers in the array. Zeros, being neither positive nor negative, do not move. Note that the order of the integers may not be maintained. All swapping of elements is done "in place" so that minimal extra space is used (in $\Theta(1)$) and with only one pass through the array (in $\Theta(n)$).

Handle all 1D arrays of integers, including the null and empty arrays.

Hand in:

Submit the following two files on eClass:

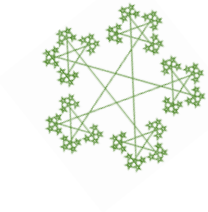
- `Assign7.java`
- `Assign7.pdf`, which is a `.pdf` file of your Java code

Practice Questions – Assignment 7 Concepts:

These are self-check questions; do not hand in solutions. These questions are provided so that you can check whether you have learned the concepts we are expecting you to learn. No solutions are provided, and we encourage you to discuss these questions with other students.

1. Where did you use recursion?
2. Did you have a recursive method or a recursive data structure?
3. How does the use of methods (functions) aid the creation of recursion?
4. Do you have any wrapper methods? Which ones and what is their purpose?
5. What is a base case? Identify all your base cases.
6. What is a recursive step? Identify all your recursive steps.
7. How do you move the recursion towards base case in each of the methods that are recursive?
8. Why are parameters important in recursion?
9. Did you need to change your way of thinking in order to program recursively? If so, how so?

10. Do you believe that recursion is as powerful as looping with `while`/`for`? Why or why not?
11. Can you describe the recursion in the stars that are scattered throughout this assignment?



Images
Stars: https://mathworld.wolfram.com/StarFractal.html
Comic: https://arnoldzwicky.org/2018/04/08/three-weekend-cartoons-pop-goes-the-caveman-couple-recursively/