Due Date
Wednesday, October 30, 2013

Program objectives
The objectives of this assignment are as follows.
An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution (ABET b).

Value
This program is worth 15 points. The distribution of points will be as follows.

Criterion	**Value**
Global functions	1
Recursive functions	3
Program style (includes greeting)	3
Correct output with annotations	8

Delivery method
Archive and compress your files using the tar command. Attach the tar file, named hw6.tar, to an email that you send to the class at csc2421@orion.ucdenver.edu In the Subject field, type HW6. In the body of the email type your name, then send the mail.

Problem
This is a two-part problem.

1. We can sort an array of elements (assuming there is a meaningful way to compare the elements) by using the following algorithm. Recursively sort the first *n*-1 elements of an *n*-element array. Then, place the *n*th element in its proper position within the *n*-1 sorted elements. Using this algorithm, write a global template that sorts an array of T. To test your algorithm, generate and save *n* pseudorandom integers in the range [*low*, *high*]. Display the original array in a row-column format (15 integers per line with evenly spaced columns), then display the sorted array in the same format.
2. Write a recursive function (non-template) that prints a pattern of 2*p* lines of the character * such that the first line has 1 character, the second line 2 characters, … , and the *p*th line *p* characters. This is followed immediately by another *p* lines, such that the first line has *p* characters, the second line has *p*-1 characters, … , and the *p*th line has 1 character.

Input
Command line positional parameters *n*, *low*, *high*, *p* (in that order). For example,
a.out 1000 25 99 15.

Output

1. The original array of integers formatted such that there are 15 integers per line with a field width of 4 between columns. The sorted array (smallest to largest) with the same formatting.
2. The pattern of the character * as specified in part 2 of the problem specification.