CSC 212 Tutorial #2
Recursion

Problem 1

Write the static recursive method *count* that takes as input an array of integers *data* and an integer x and counts the number of times x appears in *data*.
Method signature `public static int count(int[] arr, int x)`

Example 1.1.

arr = {*10, -8, 5, 4, 2, 5*} *after calling* `count(arr, 5)` $\to$ *2*

arr = {*10, -8, 5, 4, 2, 5*} *after calling* `count(arr, 3)` $\to$ *0*

Problem 2

Write the static recursive method *isPalindrome* that takes as input a generic array *data* of size n and determines if the n part of the array is the same as its reverse.
Method signature `public static <T> boolean isPalindrome(T[] arr, int n)`

Example 2.1.

{*'r', 'a', 'd', 'a', 'r'*} *after calling* `isPalindrome(arr, 5)` $\to$ *true*

{*1, 9, 9, 1, -6*} *after calling* `isPalindrome(arr, 4)` $\to$ *true*

{*"A", "B", "C"*} *after calling* `isPalindrome(arr, 3)` $\to$ *false*

Problem 3

In the *Towers of Hanoi* puzzle, we are given a platform with three pegs, a, b, and c, sticking out of it. On peg a is a stack of n disks, each larger than the next, so that the smallest is on the top and the largest is on the bottom. The puzzle is to move all the disks from peg a to peg c, moving one disk at a time, so that we never place a larger disk on top of a smaller one. See Figure 1 for an example of the case $n = 4$. Describe a recursive algorithm for solving the *Towers of Hanoi* puzzle for arbitrary n. (Exercise C-5.16 in the textbook)

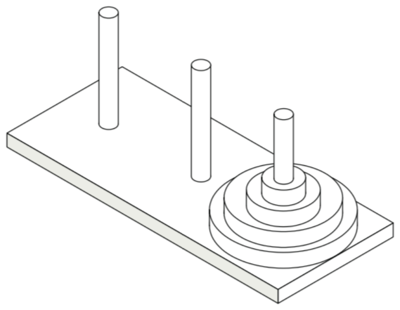

Figure 1: An illustration of the Towers of Hanoi puzzle