Assignment 2

Problems on Arrays

1. Write a program that, given an array A of n integers, finds out the largest and second largest elements of the array.
2. Write a program that, given an array A of n signed integers, finds out a subsequence i,i+1,...,j such that the sum A[i] + A[i+1] + ... + A[j] is maximum over all such subsequences. Note that the problem is trivial if all numbers are positive - your algorithm should work when the numbers may have different signs.
3. Given an integer array of length 2n-1 (where n is the number of distinct possible integers in the array), each integer appears twice except one. You have to find out the integer that appear once without using nested for loops. Example: If n=3, the array is 1, 22, 1, 5, 22, answer is 5
4. Given two integers n and k (where k < n), you have to rotate the array k positions right. Example: If n= 5 and k = 2,and initially array is 1, 22, 1, 5, 22, then after rotation, it will be 5, 22, 1, 22, 1.
5. You are given an array of n positive integers. If any element appears more than n/2 times, you return that element; otherwise return -1.
6. Given an integer array A of length, can you find the rank of the last element? The rank of an element of an array is its position in the sorted array. Example: int A[] = {5, 4, 3, 1, 2}. The rank of 2 is 1.

Problems on Strings

1. Find whether an input string is palindrome or not. Example madam, noon etc.
2. Given two strings a and b, check whether b is a substring of a or not. If b is a substring of a, then return the index of first occurrence of b in a; otherwise return -1.
3. Given two strings a and b, check whether b can be constructed from a by simply rotating towards right. If yes return 1; otherwise -1. Example: If a = abcd and b = dabc; then output is 1; If a = abcd and b = dacb, then output is -1.

4. Given a sentence, you have to reverse the sentence in the following manner. Example: Ankit is adept at Math. Output: Math at adept is Ankit.

Problems on Recursion

1. For k in N we have $a^{2k} = (a^k)^2$, and $a^{2k+1} = (a^k)^2 * a$. Use this observation to write a recursive function that, given a positive integer a and a non-negative integer b, computes the power a^b.
2. Write a recursive function that computes the binomial coefficient C(n, r) using the inductive definition: C(n, r) = C(n-1, r) + C(n-1, r-1) for suitable values of n and r. Supply appropriate boundary conditions.
3. **Tower of Hanoi**: There are three pegs A,B,C. Initially, Peg A contains n disks (with holes at their centers). The disks have radiuses 1,2,3, . . . ,n and are arranged in Peg A in increasing sizes from top to bottom, i.e., the disk of radius 1 is at the top, the disk of radius 2 is just below it, . . . , the disk of radius n is at the bottom. Your task is to move the disks from Peg A to Peg B in such a way that you are never allowed to move a larger disk on the top of a smaller disk. You may use Peg C as an auxiliary location for the transfer. Write a recursive function by which you can perform this transfer. Your function should print all disk movements.

   (Hint: First move the top n-1 disks from Peg A to Peg C using Peg B as an auxiliary location. Then move the largest disk from Peg A to Peg B. Finally, move the n-1 disks from Peg C to Peg B using Peg A as an auxiliary location.)

4. Show that Catalan numbers can be recursively defined as follows: C0 = 1, C1 = 1, and

   Cn = C0*Cn-1 + C1*Cn-2 + ... + Cn-2*C1 + Cn-1*C0 for n>=2.

a. Write an iterative function to compute Cn for a given n. (Remark: The computation of Cn requires all the previous values C0, C1, ..., Cn-1. So you are required to store Catalan numbers in an array.)

b. Write a recursive function to compute Cn such that each Ci is to be computed only once in the entire sequence of recursive calls.