Coursework 1, deadline 20/11/18

October 19, 2018

1 Missing number problem for arrays with a fixed length

Recall the missing number problem we considered in the class (also available on the slides of lecture 1). In particular, we considered two algorithms, one with nested loops and one without them. This question is about the first algorithm (with nested loops). Assume that $n = 10$, the array is $[1, 2, 3, 4, 5, 6, 7, 9, 10]$ (number 8 is missing).

1. **10 points.** How many iterations of the outer loop will be needed for the program to learn that 8 is missing?
2. **10 points.** How many iterations of the inner loop will be needed to find out that 6 is in the array?

2 Missing number problem for arrays with an arbitrary length

For the same algorithm as in Question 1, assume that n is arbitrary and the array is $\{1, 2, \ldots, n-1\}$, i.e. number n is missing.

1. **7 points.** How many iterations of the outer loop will be needed to discover that number n is missing?
2. **8 points.** For each number i from 1 to n, how many iterations of the inner loop will be needed to process that number?
3. **5 points.** Calculate the total number of loop iterations of the inner loop.

3 Binary increment operation (20 points)

Consider a one-dimensional array A of 100 elements. That is, the elements are $A[0], \ldots, A[99]$. Suppose that the content of each element is either 0 or 1. That is, you can treat the array as a binary number with $A[0]$ being the leftmost digit.

Write a program $PlusOne(A)$ that adds 1 to the number stored in the array. For example, (assuming that the array contains only 3 elements) if $A[0] = 0, A[1] = 1, A[2] = 1$ then $PlusOne$ modifies the array to $A[0] = 1, A[1] = 0, A[2] = 0$.

Remark. If the array A contains all ones then the result of increment should be all zeroes.

4 Processing a binary matrix (20 points)

Let A be a $n \times n$ two-dimensional array (matrix), the content of each element is either 0 or 1. Write an $O(n^2)$ program that tests whether there are two 1 lying on the same row or the same column in A.

Hint: Make sure that you use a single loop per row and column.

5 A constant time algorithm

1. **15 points.** Assume that all the elements in the given array are pairwise distinct. Design an $O(1)$ algorithm that returns an element that is **NOT** the smallest one in the array.

2. **5 points.** Why would not your approach work for the case of repeated elements in the array?