Imperial College London – Department of Computing

MSc in Computing Science

580: Algorithms

Assessed Coursework 2

1. Arrays $A[1, ..., M]$ and $B[1, \dots, N]$ contain *sets* of integers (there are no duplicates within each sequence) in ascending order. The *set difference* $C = A \setminus B$ is an ordered array that contains all elements of A that are not in B.

   (a) A *naive* algorithm to compute C would scan the whole sequence B to check for the presence of each element of A. What will be the upper and lower bounds of the time complexity of such an algorithm?

   (b) Write a $O(M + N)$-time algorithm to solve the set difference problem. What is the lower (Ω) bound for the time complexity of your algorithm?

2. Given an array $A[1, \dots, N]$ of N integers, the procedure LONGEST should return the length of the longest strictly increasing sequence within A. This sequence does not have to be contiguous, but the ordering of A should be preserved, and each element must be strictly less than the next. So, given $A = [56, -12, 4, 34, -3, 5, 35]$, the longest increasing sequence is either $[-12, 4, 34, 35]$ or $[-12, -3, 5, 35]$ or $[-12, 4, 5, 35]$ (there might be more than one longest sequence), and the length is 4.

   (a) Using a dynamic programming approach, write a procedure for LONGEST that runs in $O(N^2)$ time.

   To succeed in this task you will need to decompose the problem into subproblems. Start by considering the following. If you know the length of the longest increasing sequence within A that finishes with $A[i]$, for all $i < j$, what is the length of the longest sequence that finishes with $A[j]$?

   (b) If your solution was implemented recursively, without using dynamic programming, what would be the time complexity of the algorithm? Consider all input cases.

Submission

Submit By: 1900, Tuesday 5th March 2019

Submit your *typed* answers to CATE in a file named `cw2.pdf` by the deadline above. Scanned copies of hand-written answers are not acceptable. Procedures can be written in either pseudocode or Java. If you are using LaTeX, then two suggested ways of typesetting procedures are to use a `verbatim` environment:

```
\begin{verbatim}
  Anything typed here will
    be output exactly as it
    is written
  in your source file

\end{verbatim}
```

or an `algorithmic` environment which creates this sort of output:

```
procedure SWAP($A, i, j$)
    if $i \leqslant j$ then
        $temp = a_i$
        $a_i = a_j$
        $a_j = temp$
    end if
end procedure
```

See https://en.wikibooks.org/wiki/LaTeX/Algorithms for details.