Major Exam (COL 351)

Read the instructions carefully:

- You need to justify correctness and running time of each algorithm.
- If using dynamic programming, you must explain the meaning of table entries, and explain the order of computing them.
- For NP-completeness reduction, you can use the fact that 3-SAT, Independent Set, Vertex Cover, Partition, Subset Sum, Clique and Hamiltonian Cycle (in directed and in undirected graphs) are NP-complete.
- No argument should use examples – they will be ignored.
- You can assume any result proved during the lectures (but cannot assume any other result which is in the book or in the tutorials).

1. You are organizing a sports event, where each contestant has to swim 20 rounds in a pool, and then run 3 kilometers. However, the pool can be used by only one person at a time. In other words, the first contestant swims 20 rounds, gets out and then starts running. As soon as this first person is out of the pool, a second contestant begins swimming the 20 rounds; as soon as he/she is out and starts running, a third contestant begins swimming... and so on.) **(6 marks)**

   You are given a list of n contestants, and for each contestant you are given the time it will take him/her to complete swimming 20 rounds of the pool, and the time it will take him/her to run 3 kilometers. Your job is to decide on a schedule for the event, i.e., an order in which to sequence the starts of the contestants. The completion time of a schedule is the earliest time at which all contestants will be finished with swimming and running. Give an efficient algorithm that produces a schedule whose completion time is as small as possible.

   Example: Suppose there are two contestants C_1, C_2 with swimming and running times being $(10, 5)$ for C_1 and $(2, 8)$ for C_2. If we schedule them as C_1, C_2, then C_1 will finish swimming and running by time $10 + 5 = 15$. C_2 can start swimming at time 10, and so, will finish by time $10 + 2 + 8 = 20$. Note that both contestants will finish by time 20, and so the completion time is 20. If we order them as C_2, C_1, then C_2 will finish by time 10 and C_1 by time 17. Therefore, the completion time is 17. Thus, the best ordering is C_2, C_1.

2. Let S be a set of $n + 1$ distinct integers. You can assume that S is given as an array. You are given an unsorted array A of size n containing exactly n out of the $n + 1$ integers in S. Give an $O(n)$ time algorithm to find the integer from S which is not in A. The only operation allowed on numbers in S (or A) is comparison (you are NOT allowed to perform addition, subtraction, multiplication, etc. on these numbers). **(5 marks)**

3. You are given an array A containing n numbers (which could be positive, zero or negative rational numbers). A sub-array $A[i, j]$ of A, where $i \leq j$, is defined by the sequence $A[i], A[i + 1], \ldots, A[j]$. For each such sub-array, define $P(i, j)$ as the product of the entries in $A[i, j]$. Give an $O(n)$ time algorithm to find the largest value of $P(i, j)$ overall sub-arrays $A[i, j]$ (note that the algorithm just outputs a number). You can assume that arithmetic operations like multiplication on numbers in A take constant time. **(6 marks)**

   Example: Suppose A is $\{-3, 10, -6, 7, 2, -1\}$. Assuming that the first element of A is denoted by $A[1]$, the sub-array $A[1, 4]$ has total product $-3 \times 10 \times -6 \times 7 = 1260$, whereas sub-array $A[3, 6]$ has total product $-6 \times 7 \times 2 \times -1 = 84$.

4. A town has r residents $R_1, \ldots, R_r$, q clubs $C_1, \ldots, C_q$ and p political parties $P_1, \ldots, P_p$. Each resident is a member of exactly one club and belongs to exactly one political party. You want to form a governing council for the town. The governing council must contain exactly l_i members from club C_i, for $i = 1, \ldots, q$; but it can have at most u_k members from the political party P_k, for $k = 1, \ldots, p$. Give an efficient algorithm which either finds such a council or declares that no such council is possible. Assume that the membership information is given in a suitable data-structure. **(4 marks)**

5. Consider the following optimization version of PARTITION problem. You are given n integers $x_1, \ldots, x_n$ and would like to partition them into sets A and B such that $\max(\texttt{Sum}(A), \texttt{Sum}(B))$ is minimized, where $\texttt{Sum}(X)$ denotes the sum of all the numbers in X. Given a solution A, B, define its value as the quantity $\max(\texttt{Sum}(A), \texttt{Sum}(B))$. Consider the following greedy algorithm – initialize sets A and B to emptyset. Consider the numbers $x_1, \ldots, x_n$ iteratively. When looking at x_i, if $\texttt{Sum}(A) < \texttt{Sum}(B)$, add x_i it to A, else add it to B.

hint.

   (a) Prove that there is an example for which this algorithm has value 3/2 times the minimum possible value for this example. **(1 mark)**

   (b) Prove that this algorithm has the property that for any input, its value is at most 3/2 times the minimum possible value for this input. **(6 marks)**

   (c) Suggest an algorithm which would have better ratio than 3/2 – you need not prove anything, just give a one line algorithm. **(1 mark)**

6. Consider the following variant of max-flow: you are given a directed graph G with positive integer edge capacities, two vertices s and t. You would like to find a flow from s to t such that flow on every edge e is either 0 or u_e, where u_e denotes the capacity of e – we call such a flow a "saturating flow". Prove that the following problem is NP-complete: given a directed graph G with vertices s, t and positive integer edge capacities, and a parameter k, is there a saturating flow from s to t of value at least k ? **(6 marks)**