Department of Computer Science and Engineering.
Institute of Technical Education & Research, SOA University

Mid–Semester Examination, October–2016
Algorithm Design–II (CSE 4131)

Semester: 5th | Branch: CSE, CSIT | Full mark: 30 | Time: 120 Mins.

Subject Learning Outcome	*Taxonomy Level	Ques. No.	Marks
(i) understand the network flow problem and apply it to real–world problems.	L3	2(a), 2(b), 2(c)	2+2+2
(ii) use a greedy approach to solve an appropriate problem and prove if the greedy rule chosen leads to an optimal solution.	L3, L4, L5,	3(a), 3(b), 3(c)	2+2+2
(iii) use recursive backtracking to solve an appropriate problem and identify errors in incorrect implementations.	L3, L4, L5	4(a), 4(b), 4(c)	2+2+2
(iv) describe various heuristic problem solving methods.			
(v) use dynamic programming to solve an appropriate or provide a recursive solution using memoization.	L3, L4, L5	5(a), 5(b), 5(c)	2+2+2
(vi) – distinguish between computationally tractable and intractable problems. – define and relate class–P, class–NP and class NP–complete. – given a problem in NP, define an appropriate certificate and the verification algorithm.			
(vii) understand the concept of approximation ratio (with emphasis on constant–factor approximation)			
(viii) identify and apply an appropriate algorithmic approach to solve a problem.			
(ix) given a numerical problem, explain the challenges to solve it.			

*Bloom's taxonomy levels: Knowledge (L1), Comprehension (L2), Application (L3), Analysis (L4), Evaluation (L5), Creation (L6)

Answer all five questions.
All questions carry equal marks. All bits of each question carry equal marks.

Q1.(a) //Computation of the sum, S, of the first n natural numbers ($n \geq 1$). 2

```
sum1(n)
    i ← 1
    S ← 1
    while i < n do
        i ← i +1
        S ← S + i
    endwhile
return S
```

The precondition is P = {$n \geq 1$} and the postcondition is Q = {$S = 1 + 2 + ... + n$}. Since the sum will be computed by successively adding the current term, i, an adequate loop invariant could be $S = 1 + 2 + . . . + i$.

The following(*sum2(n)*) is another version of this above(*sum1(n)*) algorithm.

```
sum2(n)
    S ← 0
    i ← 1
    while i ≤ n do
        S ← S + i
        i ← i +1
    endwhile
return S
```

Check the correctness of *sum2(n)* by identifying a loop invariant, which may be a modified form of that is used in *sum1(n)*.

(b) Assume that a mergesort algorithm in the worst case takes 30 seconds for an input of size 64. Find the approximate maximum input size of a problem that can be solved in 6 minutes? 2

(c) Give a suitable matching between the algorithm and data structures given below. 2

1) Breadth First Search	1) Stack
2) Depth First Search	2) Queue
3) Prim's Minimum Spanning Tree	3) Union Find
4) Kruskal' Minimum Spanning Tree	4) Priority Queue

Q2. Consider the network flow problem with the following edge capacities, c(u,v) for edge (u,v):
c(s,2)=2, c(s,3)=13, c(2,5)=12, c(2,4)=10, c(3,4)=5, c(3,7)=6, c(4,5)=1, c(4,6)=1, c(6,5)=2, c(6,7)=3, c(5,t)=6, c(7,t)=2

Department of Computer Science and Engineering.
Institute of Technical Education & Research, SOA University

Mid–Semester Examination, October–2016
Algorithm Design–II (CSE 4131)

(a) Draw the flow network. 2

(b) Run the Ford–Fulkerson algorithm to find the maximum flow. Show each residual graph. 2

(c) Show the minimum cut. 2

Q3.(a) C1 = { a = 00, b = 01, c = 10, d = 11 }, C2 = { a = 0, b = 110, c = 10, d = 111 }, C3 = { a = 1, b = 110, c = 10, d = 111 }. 2
Given an encoded message, decoding is the process of turning it back into the original message. A message is uniquely decodable (vis–a–vis a particular code) if it can only be decoded in one way.
(i) using C1 decode 010011, using C2 decode 1100111 and using C3 decode 1101111.
(ii) show that every message encoded using C1 and C2 are uniquely decodable(decipherable), but not C3.

(b) Using Greedy–choice–property find an optimal solution to the knapsack instance n = 7, m = 15, (p1, p2, p3, p4, p5, p6, p7) = (10, 5, 15, 7, 6, 18, 3), and (w1, w2, w3, w4, w5, w6, w7) = (2, 3, 5, 7, 1, 4, 1). 2

(c) Find the maximum size set of mutually compatible activities for the following table using a suitable greedy choice property. Also find the number of element comparisons in this computation and state greedy–choice–property used. 2

Activity	1	2	3	4	5	6	7	8	9
Start time	9	3	6	11	2	0	12	7	4
Finish time	12	5	9	14	4	3	16	10	8

Q4. (a) "*Backtracking ensures correctness by enumerating all possibilities. It ensures efficiency by never visiting a state more than once.*" True or False. Justify your answer constructing state–space–diagram for 3–Queen problem. 2

(b) Identify the implicit and explicit constraints in backtracking formulation of Sudoku problem. 2

(c) A *derangement* is a permutation p of {1,2,...,n} such that no item is in its proper position, i.e., $p_i \neq i$ for all $1 \leq i < n$. Write an efficient backtracking formulation that constructs all the derangements of n items. 2

Q5. (a) Explain how caching reduces recomputing the same subproblem by constructing the computation tree for computing 5th Fibonacci number recursively. 2

(b) Consider the following partially–filled table for Longest Increasing Subsequence(LIS) problem. Fill the blank cells using dynamic programming. 2

i	1	2	3	4	5	6	7	8	9
A[i]	2	4	3	5	1	7	6	9	8
LIS[i]									
All possible Optimal–subsequence(s)									

(c) In what aspect(s) dynamic programming is better in comparison to exhaustive search techniques? 2

****End of Questions****