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

End-Semester Lab. Examination, July-2022
Algorithm Design-2 (CSE 4131)

Semester: 4th
Full mark: 15

Branch: CSE, CS&IT
Time: 90 Mins.

All questions are compulsory.

Q1. Give the Java/C/C++/Python code implementation of the following problem. 5
Sequence alignment problem using dynamic programming.

Q2. Using backtracking, let us generate all possible permutations of a given set S = {7,5,9}, using the code given in section-7.1.2 of book (i.e. The Algorithm Design Manual by Steven S. Skiena). In how many number of steps the arrangement {5,7,9} will be generated and in that step what are the contents of k, in_perm[i] and in_perm[a[i]]? Demonstrate the steps neatly. (Refer to the code below) 5

```
generate_subsets(int n){
        backtrack(a[],0,n);
}
backtrack(int a[],int k,int n) {
      if(is_a_solution(a[],k,n))
            process_solution(a[],k,n);
      else {
            k = k+1;
            construct_candidates(a[],k,n,c,&nc);
            for(i=0; i<nc; i++) {
                a[k] = c[i];
                make_move(a[],k,n);
                backtrack(a[],k,n);
                unmake_move(a[],k,n);
                if(finished) return; // finished = FALSE
            }
      }
}
```

Q3. Design an efficient algorithm to find the peak element in a 2D array. An element is a peak element if it is greater than or equal to its four neighbors, left, right, top and bottom. For example neighbors for A[i][j] are A[i-1][j], A[i+1][j], A[i][j-1] and A[i][j+1]. For corner 5 elements, missing neighbors are considered of negative infinite value.

Below are some facts about this problem:

1. A Diagonal adjacent is not considered as neighbor.
2. A peak element is not necessarily the maximal element.
3. More than one such elements can exist.
4. There is always a peak element. We can see this property by creating some matrices using pen and paper.

****End of Questions****

Instructions:

The evaluation will be done in the following ways:

- Correct implementation with satisfactory response to on-spot questions: 5 / 5
- Correct implementation with unsatisfactory response to on-spot questions: 3 / 5
- Incorrect/partial (min. 80%) implementation with satisfactory response to on-spot questions: 3 / 5
- Incorrect/partial (min. 80%) implementation with unsatisfactory response to on-spot questions: 2 / 5
- No implementation with satisfactory response: 1.5 / 5
- No implementation with unsatisfactory response: 0.5 / 5
- Plagiarized code: -2.5 / 5

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

END SEMESTER EXAMINATION, JULY-2022

Computer Science Workshop-2 (CSE 3141)

Semester: 4th **Branch: CSE-M**
Full mark: 60 **Time: 3 Hours**

Subject/Course Learning Outcome	*Taxonomy Level	Ques.No.	Marks

Analysis algorithm, using time and space complexity	L3, L4	Q1(a), (b), (c)
		Q2(a), (b), (c)
			2+2+2
			2+2+2

Understanding and effectively use ADT, java collections, sorting, and searching.	L1, L3	Q3(a), (b), (c)
		Q4(a), (b), (c)
		Q6(a), (b), (c)
			2+2+2
			2+2+2
			2+2+2

Applying linked list, stack, queue on different; problem solving.	L1, L3, L4	Q5(a), (b), (c)
		Q7(a),(b), (c)
			2+2+2
			2+2+2

Applying tree data structure on problem solving.	L1, L3, L4	Q8(a), (b), (c)
		Q9(a), (b), (c)
		Q10(a), (b), (c)
			2+2+2
			2+2+2
			2+2+2

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

Answer all questions. All questions carry equal marks.

Q1. (a) Write a recursive method to implement Tower of Hanoi problem. [2]

(b) Write a recursive method to search an element using binary search algorithm. 1 [2]

(c) Analyse the time complexity of binary search algorithm. 2 [2]

Q2. (a) Find the time complexity of the recurrence relation using the Master Method. Determine which case (if any) of the master the theorem applies and write down the time complexity. If Master Method does not apply to the recurrence, then justify.

$T(n) = 3T(n/4) + n \lg n$ [2]

(b) $T(n) = 2T(n/2) + n \lg n$ [2]

(c) Find the time complexity of the following code.

```
for(int i = n ; i >= 1; i = i/2)
        statement1;
```

2

Q3. (a) Write a method input() to represent an undirected graph using adjacency list representation. (Use LinkedList class for graph representation) [2]

(b) Implement a BFS() method to perform Breadth-First Search (BFS). In the BFS algorithm, a graph is traversed in a layer-by-layer fashion. (Use ArrayDeque class for implementation of Queue) [2]

(c) Create the required class and main() method to execute the above methods. [2]

Note: Write a single program for Q3(a), (b), and (c).

Q4. (a) Write a static method partition() to compute the index q (pivot) as part of this partitioning procedure. Pivot means arranging the elements of an array so that all elements smaller than the pivot are left of the pivot, and greater elements present right of the pivot. 2 [2]

Scanned with CamScanner

(b) Implement the quicksort algorithm using a recursive static method named quickSort(). 2 [2]

(c) Create the required class and main() method to execute the above methods. 2 [2]

Note: Write a single program for Q4 (a), (b), and (c).

Q5. (a) Write a method to insert an element in sorted order in the linked list given the head pointer. [2] 2

(b) Implement a method to reverse a singly linked list. 1 or 2 [2]

(c) Write searchList() method to search an element in the linked list. 2 [2]

Note: All bits of Q5 are static methods, so no main() method and class are required.

Q6. (a) In the given list of −ve and +ve numbers, write a static method to separate −ve numbers from the +ve numbers. 1 or 2 [2]

(b) In the given list of 1's and 0's, write a static method to separate 0's from 1's. [2]

(c) In the given list of 0's, 1's, and 2's, write a static method to separate 0's, 1's, and 2's. [2]

Note: All bits of Q6 are static methods, so no main() and class are required.

Q7. (a) Given a stack whose elements are sorted, write a function sortedInsert() which will insert elements in sorted order, with the highest at the top and lowest at the bottom. 1 or 2 [2]

(b) Implement a function to convert infix expression to postfix expression. 1 or 2 [2]

(c) Create a class and all the required methods to convert infix expression to postfix expression. [2] 1 or 2

Note: Write a single program for Q7 (b) and (c).

Q8. (a) Write a method to create a Complete Binary Tree (CBT) from values given as an array. 1 or 2 [2]

(b) Write a method to perform a Depth-First Search (DFS) of a binary tree without using recursion. (Use ArrayDeque class for implementation of Stack) 2 [2]

(c) Create the required class and main() method to execute the above methods. 2 [2]

Note: Write a single program for Q8 (a), (b) and (c).

Q9. (a) Write an insert() method to insert an element into Binary Search Tree (BST). 2 [2]

(b) Implement isBST() to check whether a given Binary Tree is a BST or not. 2 [2]

(c) Create a class and required methods for the creation of BST. 2 [2]

Note: Write a single program for Q9 (a), (b) and (c).

Q10. (a) Construct a Binary Search Tree (BST) by inserting in order of the following integers, 50, 15, 62, 5, 20, 58, 91, 3, 8, 37, 60, 25 2 [2]

(b) Find the output of inorder, preorder, and postorder traversal of the BST created in Q10(a) [2] 2

(c) Write the search time of Complete Binary Tree (CBT) and BST for the best and worst cases. Justify your answer. 1 or 2 [2]

Note: No need to write code for Q10 (a) and (b). Draw the tree and find the output of its traversal algorithms.

.... *End of Questions* ....

40/60 | 45/60