|  |
| --- |
| Serial No: |
| **2nd Mid Term Exam** |
| **Total Time: 1 Hour** |
| **Total Marks: 45** |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Signature of Invigilator |

|  |
| --- |
| **CS2001 Data Structures** |
| Monday, September 26, 2022 |
| **Course Instructor** |
| Dr. Hashim Yasin  Dr. Anwer Shah  Mr. Muhammad Usman Joyia  Mr. Muhammad Yousuf |

|  |
| --- |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ . .  Roll No Section Signature |

## DO NOT OPEN THE QUESTION BOOK OR START UNTIL INSTRUCTED.

**Instructions:**

1. Verify at the start of the exam that you have a total of five (05) questions printed on seven (07) pages including this title page.
2. Attempt all questions on the question-book and in the given order.
3. The exam is closed books, closed notes. Please see that the area in your threshold is free of any material classified as ‘useful in the paper’ or else there may a charge of cheating.
4. Read the questions carefully for clarity of context and understanding of meaning and make assumptions wherever required, for neither the invigilator will address your queries, nor the teacher/examiner will come to the examination hall for any assistance.
5. Fit in all your answers in the provided space. You may use extra space on the last page if required. If you do so, clearly mark question/part number on that page to avoid confusion.
6. Use only your own stationery and calculator. If you do not have your own calculator, use manual calculations.
7. Use only permanent ink-pens. Only the questions attempted with permanent ink-pens will be considered. Any part of paper done in lead pencil cannot be claimed for checking/rechecking.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Q-1 | Q-2 | Q-3 | Q-4 | Q-5 | Total |
| **Total**  **Marks** | **10** | **10** | **08** | **07** | **10** | **45** |
| **Marks Obtained** |  |  |  |  |  |  |

**Vetted By: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Vetter Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **University Answer Sheet Required:** | **No** |  |  |  |  | **Yes** |  |

**Question No. 1 Multiple Choice Questions. [10 Marks]**

Consider the following statements and select all statements that apply to correctness of the statements. You need to fill the key given at the end of this question to **claim** your marks.

1. With implementation of Queue ADT with array what would be the initial values of front and rear
   1. -1, and 0
   2. 0, and 0
   3. 0 and -1
   4. 1, and 1
2. Which of the following linear DS allows the insertion at both ends but deletion at one end
   1. Deque
   2. Queue
   3. Restricted deque
   4. Linkedlist
3. The usage of pointer where nodes are logically adjacent to each other is referred to as
   1. Stack
   2. Queue
   3. Linked allocation
   4. Binary Search Tree
4. Which of the following DS is most feasible for dynamic implementation of deque
   1. Circular Array
   2. Singly Linkedlist
   3. Doubly Linkedlist
   4. All of above
5. Considering the array based implementation of queue, which of the following is true for isFull() operation Implementation where size of array is represented as SIZE
   1. Rear = (Front + 1) % SIZE
   2. Front = (Rear +1) % SIZE
   3. Rear = SIZE -1
   4. Front = Rear
6. Consider an N-Sized array-based Circular Queue (assuming that the indexes start from zero), if Front = N-3 and Rear= 6 (assume Rear is incremented before new insertion and the Front points to a value that is to be Dequeued next) , then how many elements Queue can have at max?
   1. 7
   2. 11
   3. 10
   4. 5
7. For a AVL trees, the magnitude of difference between height of left-subtree and right-subtree is:
   1. at least 1
   2. exactly 1
   3. exactly zero
   4. at most 1
8. If a node p at depth h−1 has a left child in the context of complete binary tree
   1. All the nodes right to node *p* must has both child
   2. All nodes right to node *p* must has right child
   3. All the nodes left to node *p* must has left child
   4. All the nodes left to node *p* must have both child
9. Number n of nodes in a binary tree of height h is
   1. log2(n + 1) – 1
   2. exactly 2h + 2h - 1
   3. At most 2h - 1
   4. At least h+1
10. For a Perfect binary tree
    1. There are exactly 2h - 1 external/leaf nodes in the perfect tree of height h
    2. There are exactly 2h internal nodes in the perfect tree of height h
    3. It is impossible to count exact number of internal nodes
    4. There are exactly (2h + 2h -1) nodes in the tree of height h

**Solution Key**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr** | **A** | **B** | **C** | **D** |
| **1** |  |  |  |  |
| **2** |  |  |  |  |
| **3** |  |  |  |  |
| **4** |  |  |  |  |
| **5** |  |  |  |  |
| **6** |  |  |  |  |
| **7** |  |  |  |  |
| **8** |  |  |  |  |
| **9** |  |  |  |  |
| **10** |  |  |  |  |

**Question No. 2 Circular Deque ADT [10 Marks]**

A queue is different from a stack in the way how the insertion and removal work as shown below (first image from left to right). However, to efficiently use the space in the queue, we consider the circular nature of baseline data structure while implementing it as shown below (right side). Now, consider the concept of **Deque** where insertion and deletions can be done from both ends. In this task, we want to implement deque concept for the stacks with top and bottom as sentinel markers with initial values (-1, -1). You are required to provide the implementation of deque using circular array for following operations considering the stack ADT have no information about the current number of items in the stack.

1. bool pushTop(int x)
2. bool popBottom()

**Text, letter

Description automatically generatedShape

Description automatically generated**

|  |  |
| --- | --- |
| bool pushTop(int x){ | bool popBottom() { |

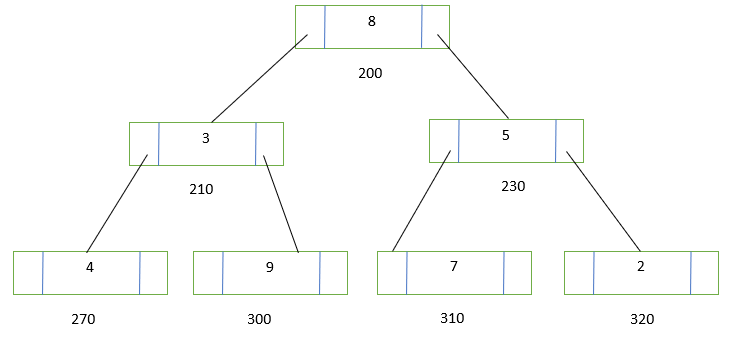
**Question No. 3 Tree Properties and Applications [2+6 = 08 Marks]**

1. Generate a tree from the following given traversals

*Pre order traversal 4, 7, 9, 6, 3, 2, 5, 8, 1  
Post order traversal 6, 3, 9, 7, 8, 3, 1, 2, 4.*

|  |  |
| --- | --- |
|  |  |

1. Consider following binary tree containing seven nodes with addresses 200, 210, 230, 270, 300, 310 and 320. Write a recursive function preorder(parameter) for preorder traversal, trace the preorder using activation records on stack. Further analyze it for number of function calls and number of activation records on stack.



|  |  |
| --- | --- |
|  |  |

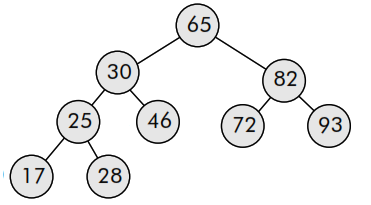
**Question No. 4 Binary Search Tree [07 Marks]**

Assume a Binary Search Tree with nodes is already made in memory. Write down a recursive function to find if every non leaf node has degree 1?

|  |
| --- |
|  |

**Question No. 5 AVL Tree [ 3+7 = 10 Marks]**

1. Construct the AVL tree using the following values, 1,3, and 2. If the tree becomes imbalanced, make it balanced by performing the correct type of rotations. Perform all the steps involved in the process. **(3)**
2. Consider the following AVL tree structure,



Insert a new element 12in the above AVL tree. If the tree becomes imbalanced, make it balanced by performing the rotations. **(7)**

**Note:** Construct the final AVL tree in the given box as under to get full marks. You can perform all steps involve in the process of rotations as **rough** work (if it is necessary) on the other side of the paper but it contains **NO** marks.

|  |
| --- |
| **Final AVL Tree:** |