|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **P.V.P Siddhartha Institute of Technology** | | | | | | | | | | | | | | | | | | | **Signature of Invigilator with date:** | | **Marks Obtained:** | |
| **Department of Computer Science and Engineering** | | | | | | | | | | | | | | | | | | |
| **Course: B.Tech** | | | | **Year: II** | | | | **Semester: II** | | | | **Objective: I** | | | | | | |
| **Regulation:PVP20** | | | | **Maximum Marks:10Marks** | | | | | | | | | **Session: F.N** | | | | | |
| **A.Y:2023-24** | | | | **Date:-29/02/24** | | | | | | | **Duration: 20 min** | | | | | | |  | | |  | |
| **Subject Code:20CS3402** | | | | | | | **Subject Name: Advanced Data Structures** | | | | | | | | | | | | | | | |
| **Registered Number:** | | | | | | | | | | | | **Name:** | | | | | | | | | | |
| **Answer all the Questions. Each Question carries ½ Mark 20×½ M=10M** | | | | | | | | | | | | | | | | | | | | | | |
| **S.No** | **Question** | | | | | | | | | | | | | | | | | | | **CO** | **Level** | **Answer** |
| **1.** | In hash table, each key is mapped into some number in the range 0 to \_\_\_\_\_ | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
| a)TableSize | | b)Tablesize-1 | | | | | | | c)Tablesize/2 | | | | | | | d)None | | | **CO1** | **L2** |  |
| **2.** | If, when an element is inserted, it hashes to the same value as an already inserted element, then we have a \_\_\_\_\_. | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
| a) Conflict | | | | b) Collision | | | | | c) Agreement | | | | | | d)None | | | |
| **3.** | \_\_\_\_\_\_\_ probing is a collision resolution method that eliminates the primary clustering problem of \_\_\_\_\_\_\_ probing | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
| a)Quadratic, Linear | | b)Quadratic, Double | | | | | | | c)Linear, Quadratic | | | | | | | d) None | | |
| **4.** | A family H of hash functions is universal, if for any x = y, the number of hash functions h in H for which h(x) = h(y) is at most \_\_\_\_\_\_\_\_ | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
| a)[H]/M | b) [H]\*M | | | | | | | c)M\[H] | | | | | | | | d)M\*[H] | | |
| **5.** | Extendible hashing, allows a search to be performed in **\_\_\_\_\_\_** disk accesses | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
| 1. One | 1. Two | | | | | | | 1. Three | | | | | 1. None | | | | | |
| **6.** | \_\_\_\_\_\_\_\_\_\_can be used to implement the insert and contains operations in constant average time | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
| a)Hash table | b)array | | | | | | | c)heap | | | | | d)None | | | | | |
| **7.** | A \_\_\_\_\_\_\_\_\_ is a data structure that allows at least the following two operations: insert, which does the obvious thing; and deleteMin, which finds, returns, and removes the minimum element | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
| a) Hash table | b)Priority queue | | | | | | | c) Tree | | | | | d)None | | | | | |
| **8.** | A binary search tree. This gives an \_\_\_\_average running time for both operations | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
| 1. O(logN) | **b)**O(N logN) | | | | | | | **c)** O(N) | | | | | **d)** O(N2) | | | | | |
| **9.** | A binomial tree, Bk, of height k is formed by attaching a binomial tree, \_\_\_\_, to the root of another binomial tree,\_\_\_\_\_\_\_\_\_\_\_. | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
| 1. Bk−1, Bk−1 | 1. Bk−2, Bk−2 | | | | | | | 1. Bk, Bk | | | | | 1. None | | | | | |
| **10.** | Binomial trees of height k have exactly \_\_\_ nodes. | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
| 1. 2k | 1. 2K | | | | | | | c)2/K | | | | | d)K2 | | | | | |
| **11.** | *AVL tree,* the balance condition must be easy to maintain, and it ensures that the depth of the tree is \_\_\_\_\_\_ | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
| O(logN) | **b)** O(N logN) | | | | | | | **c)** O(N) | | | | | **d)** O(N2) | | | | | |
| **12.** | Using division method, in a given hash table of size 157, the key of value 172 be placed at position \_\_\_\_\_\_\_\_\_\_. | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
| 1. 19 | | | | | 1. 72 | | | | | 1. 15 | | | | 1. 17 | | | | |
| **13.** | With what data structure can a priority queue be implemented? | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
|  | 1. Array | | | | | 1. List | | | | | 1. Heap | | | | 1. Tree | | | | |
| **14.** | What is the time complexity to insert a node based on key in a priority queue? | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
| 1. O(nlogn) | | | | | 1. O(log n) | | | | | 1. O(n) | | | | 1. O(n2) | | | | |
| **15.** | When do we prefer priority queues? | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
|  | | | | | | | | | | | | | | | | | | |
| **16.** | The number of trees in a binomial heap with n nodes is | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
| 1. Log n | | | | | 1. n | | | | | 1. n log n | | | | 1. n/2 | | | | |
| **17.** | Insertion in AVL takes **\_\_\_\_\_\_\_** timing | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
| 1. **Θ**(nlogn) | 1. **Θ**(log n) | | | | | | | 1. **Θ**(n) | | | | | 1. **Θ**(n2) | | | | | |
| **18.** | height of an AVL tree with n nodes is**\_\_\_\_\_\_\_\_**. | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
| 1. Nlog n | 1. log n | | | | | | | 1. n | | | | | 1. n2 | | | | | |
| **19.** | Insert 20 into AVL, Mention the resultant tree.  F2 Savita Engineering 28-3-23  D7 | | | | | | | | | | | | | | | | | | | **CO1** | **L2** |  |
|  | | | | | | | | | | | | | | | | | | |
| **20.** | Why do we prefer Red- Black trees over AVL trees? | | | | | | | | | | | | | | | | | | | CO1 | **L2** |  |
|  | | | | | | | | | | | | | | | | | | |