

**Department of Computer Science and Engineering**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **USN** |  |  |  |  |  |  |  |  |  |  |

**THIRDSEMESTER TEST -III**

|  |  |
| --- | --- |
| Subject: Data Structures Using C | **Session**: July - Dec 2019 |
| **Subject Code:**18CSI301 | **Duration**: 90 Minutes |
| **Date of Examination:27**/ 11/2019 | **Max Marks**: 03 X 15 = 45 |

**Note:**

* Answer **3 full questions** and each full question carries **15 Marks.**
* **Provide neat diagrams wherever applicable.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Q.No** | **Question** | **Marks** | **CO’s** | **Bloom’s**  **Level** |
| 1a | Define a) binary tree b) compete binary tree c) extended binary tree with suitable examples. | 05  01  02  02 | CO3 | L1 |
| 1b | Write C functions to perform following operations on a binary search tree: a) Creating BST b) To count the leaf nodes  #include <stdio.h>    struct node {  int data;  struct node \*left;  struct node \*right;  };    struct node\* getNewNode(int data) {  /\* dynamically allocate memory for a new node \*/  struct node\* newNode = (struct node\*)malloc(sizeof(struct node));    /\* populate data in new Node \*/  newNode->data = data;  newNode->left = NULL;  newNode->right = NULL;    return newNode;  }  struct node\* generateBTree(){  // Root Node  struct node\* root = getNewNode(1);  // Level 2 nodes  root->left = getNewNode(2);  root->right = getNewNode(3);  // Level 3 nodes  root->left->left = getNewNode(4);  root->left->right = getNewNode(5);  root->right->left = getNewNode(6);  root->right->right = getNewNode(7);  // Level 4 nodes  root->left->left->left = getNewNode(8);    return root;    }  Note: any relevent program also be considered.  /\*  Returns the count of leaf nodes in a binary tree  \*/  int count Leaf Node (struct node \*root)  {  /\* Empty(NULL) Tree \*/  if(root == NULL)  return 0;  /\* Check for leaf node \*/  if(root->left == NULL && root->right == NULL)  return 1;  /\* For internal nodes, return the sum of  leaf nodes in left and right sub-tree \*/  return countLeafNode(root->left) + countLeafNode(root->right);  }    int main() {  struct node \*root = generateBTree();    /\* Print number of lead nodes \*/  printf("Number of leaf Node : %d", countLeafNode(root));    getchar();  return 0;  } | 10 | CO3 | L2 |
| **OR** | | | | |
| 2a | What is a graph? Explain the properties of graphs.  A graph is a pictorial representation of a set of objects where some pairs of objects are connected by links. The interconnected objects are represented by points termed as vertices, and the links that connect the vertices are called edges.  Properties of graphs are :  1.Distance between two vertices  2.Eccentricity of a vertex  3.Radius of connected Graph  4.Diameter of a Graph  5.Central point | 05  03  02 | CO3 | L2 |
| 2b | Write breadth first traversal algorithm. Perform the BFS traversal for the given graph.              Note: alternative methods are also considered. | 10  02  02  02  02  02 | CO3 | L2 |
|  |  |  |  |  |
| 3a | Discuss Kruskal’s algorithm. Find the minimum spanning tree for the given graph using Kruskal’s algorithm.  https://www.geeksforgeeks.org/wp-content/uploads/Fig-11.jpg  The graph contains 9 vertices and 14 edges. So, the minimum spanning tree formed will be having (9 – 1) = 8 edges.    fig8new.jpeg (712×328)  Since the number of edges included equals (V – 1), the algorithm stops here. | 12  06  06 | CO4 | L3 |
| 3b | List any two differences between graphs and trees. | 3 | CO3 | L3 |
| **OR** | | | | |
| 4a | Describe the different modes of opening a file with an example.  A file can be opened in different modes. Below are some of the most commonly used modes for opening or creating a file.  r : opens a text file in reading mode.  w : opens or creates a text file in writing mode.  a : opens a text file in append mode.  r+ : opens a text file in both reading and writing mode. The file must exist.  w+ : opens a text file in both reading and writing mode. If the file exists, it's truncated first before overwriting. Any old data will be lost. If the file doesn't exist, a new file will be created.  a+ : opens a text file in both reading and appending mode. New data is appended at the end of the file and does not overwrite the existing content.  rb : opens a binary file in reading mode.  wb : opens or creates a binary file in writing mode.  ab : opens a binary file in append mode.  rb+ : opens a binary file in both reading and writing mode, and the original content is overwritten if the file exists.  wb+: opens a binary file in both reading and writing mode and works similar to the w+ mode for binary files. The file content is deleted first and then new content is added.  ab+: opens a binary file in both reading and appending mode and appends data at the end of the file without overwriting the existing content. | 7 | CO3 | L3 |
| 4b | Explain the memory representation of graphs with suitable example. | 8  04  04 | CO3 | L3 |
|  |  |  |  |  |
| 5a | Discuss DFS algorithm. Perform the DFS traversal for the given graph.  Image result for bfs traversal graph example | 09  04  05 | CO4 | L3 |
| 5b | Suppose friendly airways has 9 daily flights, as follows:  103 Atlanta to Houston,203 Boston to Denver,305 Chicago to Miami  106 Houston to Atlanta,204 Denver to Boston,308 Miami to Boston  201 Boston to Chicag,301 Denver to Reno,402 Reno to Chicago,502 hostan to bostan  Based on the above data solve the following   1. Is there any direct flight from Reno to Miami 2. What is the min no of stops path from Reno to Boston   Draw Graph Using DFS or BFS. | 06  03  03 | CO4 | L4 |
| **OR** | | | | |
| 6a | Name the data structure used to traverse the graph with BFS and found the shortest path from starting vertex to final vertex of the following graph, And list out the applications of BFS.    **C:\Users\Lenovo\Desktop\aa.jpg** | 09  03  06 | CO4 | L3 |
| 6b | Discuss the applications of trees and graphs.    Applications of Trees  1.Binary Search Trees(BSTs) are used to quickly check whether an element is present in a set or not.  2.Heap is a kind of tree that is used for heap sort.  3.A modified version of tree called Tries is used in modern routers to store routing information. | 06  02  02  02 | CO3 | L3 |

**Course Outcomes:**

|  |  |
| --- | --- |
|  | Understand the various types of data structures, operations and algorithms |
|  | Analyze the various algorithms used in linear and non-linear data structures. |
|  | Design the algorithm for stack, queues, list, trees and graphs. |
|  | Apply appropriate data structures for solving computing problems. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| L1 | L2 | L3 | L4 | L5 | L6 |
| Remembering | Understanding | Applying | Analyzing | Evaluating | Creating |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **CO/PO: Mapping** | | | | | | | | | | | | |
| (H/M/L indicates strength of correlation) H-High, M-Medium, L-Low | | | | | | | | | | | | |
| **Course Outcome**  **(COs)** | **Program Outcome (POs)** | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  | H | H | H | H | M | L | M | M | M | L | H | H |
|  | H | H | L | H | M | L | M | M | M | L | H | H |
|  | H | H | M | H | H | L | M | M | M | L | H | H |
|  | M | M | H | M | M | L | M | M | M | L | M | H |

**\*\*\*\*\***