

# LAB ASSIGNMENT-3

CSN-261

Rushiprasad Sagare

CSE- 2<sup>nd</sup> Year

18114071

Sub-batch O3

# Problem 1-

Given the set of integers, write a C++ program to create a binary search tree (BST) and print all possible paths for it. You are not allowed to use subarray to print the paths. Convert the obtained BST into the corresponding AVL tree for the same input. AVL tree is a self-balancing binary search tree. In an AVL tree, the heights of the two child subtrees of any node differ by at most one; if at any time they differ by more than one, rebalancing is done to restore this property.

Convert the obtained BST into the corresponding red-black tree for the same input. Red-Black Tree is a self-balancing Binary Search Tree (BST) where every node follows following rules.

- 1) Every node has a colour either red or black.
- 2) Root of tree is always black.
- 3) There are no two adjacent red nodes (A red node cannot have a red parent or red child).
- 4) Every path from a node (including root) to any of its descendant NULL node has the same number of black nodes.

Write a menu driven  
program as follows:

1. To insert a node in the BST and in the red-black tree
2. To create AVL tree from the inorder traversal of the BST
3. To print the inorder traversal of the BST/AVL/red-black tree
4. To display all the paths in the BST/AVL tree/red-black tree
5. To print the BST/AVL tree/red-black Tree in the terminal using level-wise indentation (print colour for red-black tree)
6. Exit

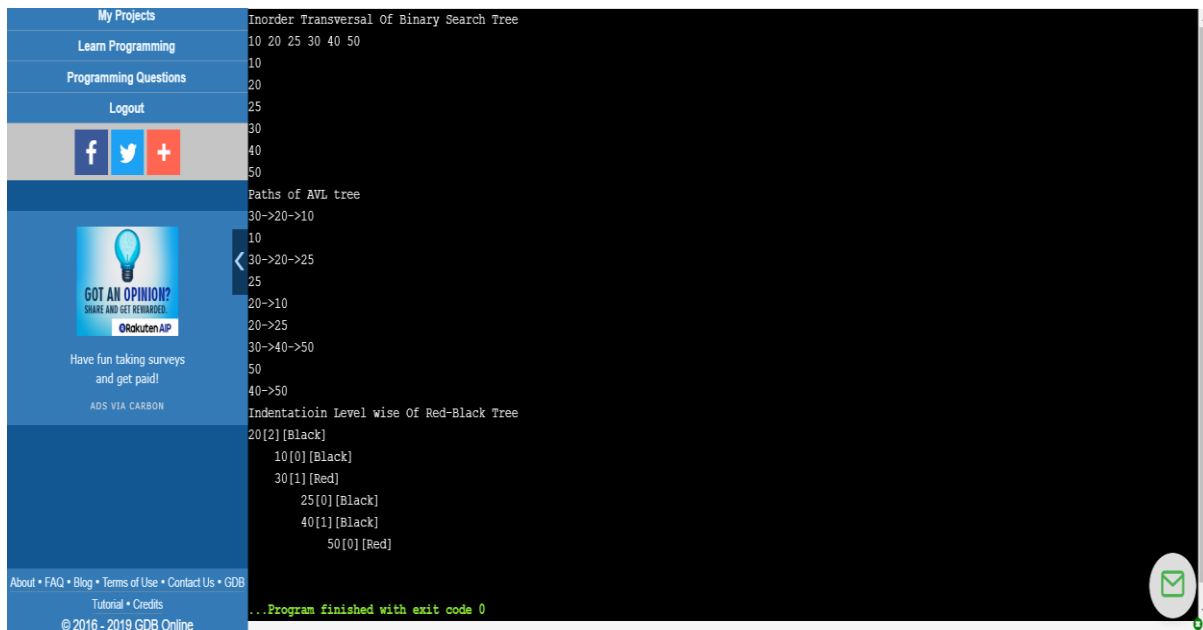
Data Structure Used- Binary Search Tree

AVL Tree

Red Black Tree

## Algorithm-

- 1)Used cpp to program.
- 2) Used 3 separate classes for bst,avl and rbt.
- 3)Used struct node to make a node of the tree
- 4)Data structure used tree .
- 5)Used concepts of rotation to make red black.  
tree from given binary search tree .
- 6)Made avl tree by inorder traversal of the  
elements of binary search tree.



The screenshot displays a web-based IDE interface with a dark theme. On the left, a sidebar contains navigation links: 'My Projects', 'Learn Programming', 'Programming Questions', and 'Logout'. Below these are social media icons for Facebook, Twitter, and a general share icon. Further down is a 'GOT AN OPINION?' survey banner for Rakuten App, and at the bottom, a footer with links for 'About', 'FAQ', 'Blog', 'Terms of Use', 'Contact Us', 'GDB Tutorial', and 'Credits', along with the copyright notice '© 2016 - 2019 GDB Online'. The main editor area shows C++ code for two tree operations. The first section, 'Inorder Transversal Of Binary Search Tree', includes an array of values {10, 20, 25, 30, 40, 50} and a loop that prints the elements in order. The second section, 'Paths of AVL tree', lists several paths: '30->20->10', '30->20->25', '20->10', '20->25', '30->40->50', and '40->50'. The third section, 'Indentatioin Level wise Of Red-Black Tree' (note the typo), shows a level-wise traversal: '20[2] [Black]', '10[0] [Black]', '30[1] [Red]', '25[0] [Black]', '40[1] [Black]', and '50[0] [Red]'. The code concludes with '...Program finished with exit code 0'. A small green checkmark icon is visible in the bottom right corner of the editor area.

```
My Projects
Learn Programming
Programming Questions
Logout
f
Twitter
+
GOT AN OPINION?
SHARE AND GET REWARDS
Rakuten App
Have fun taking surveys
and get paid!
ADS VIA CARBON
About • FAQ • Blog • Terms of Use • Contact Us • GDB
Tutorial • Credits
© 2016 - 2019 GDB Online

Inorder Transversal Of Binary Search Tree
10 20 25 30 40 50
10
20
25
30
40
50

Paths of AVL tree
30->20->10
10
30->20->25
25
20->10
20->25
30->40->50
50
40->50

Indentatioin Level wise Of Red-Black Tree
20[2] [Black]
10[0] [Black]
30[1] [Red]
25[0] [Black]
40[1] [Black]
50[0] [Red]

...Program finished with exit code 0
```

## Problem 2-

For a given sequence of positive integers  $A_1, A_2, \dots, A_N$  in decimal, find the triples  $(i, j, k)$ , such that  $1 \leq i < j \leq k \leq N$  and  $A_i \oplus A_{i+1} \oplus \dots \oplus A_{j-1} = A_j \oplus A_{j+1} \oplus \dots \oplus A_k$ , where  $\oplus$  denotes bitwise XOR. This problem should be solved using dynamic programming approach and linked list data structures.

Input:

(a) Number of positive integers  $N$ .

(b)  $N$  space-separated integers  $A_1, A_2, \dots, A_N$ .

Output: Print the number (count) of triples and list all the triplets in lexicographic order (each triplet in a new line).

Data Structures used- Linked List , Queue .

Algorithm-

- 1) The numbers are stored in a linked list.

- 2) A xorfunc is used to xor the elements of a linked list from start to end.
- 3) Triplets are counted using count function.
- 4) These triplets are printed using showTriplets function.
- 5) Used concept of dynamic programming
- 6) Used 2 pointers to traverse in linked list

```
168     printf("( %d , %d , %d )\n",hd->i,hd->j,hd->k);
169     hd=hd->link;
170 }
171 }
172
173 int main()
174 {
175
176     int n;
177     printf("Enter the number of positive integers.\n");
178     scanf("%d",&n);
179     int a[n];
180     printf("Enter the numbers.\n");
181     for(int i=0;i<n;i++){
182         scanf("%d",&a[i]);
183     }
184     for(int i=0;i<n;i++){
185         insert(a[i]);
186     }
187
188     int c=countTriplets(first,last);
189     printf("%d\n",c);
190     showTriplets();
191     return 0;
192 }
193
```

Enter the number of positive integers.  
3  
Enter the numbers.  
5 2 7  
2  
( 1 , 2 , 3 )  
( 1 , 3 , 3 )