WhatsApp | 2-3 Trees - Data | Trie Data Structu | Trie Data Structu | FREE AI Code Ge | ChatGPT | Online C Compil

programiz.com/c-programming/online-compiler/

Programiz
C Online Compiler

main.c

Share | Run

```c
#include <stdio.h>
#include <stdlib.h>
#define RED   0
#define BLACK 1
typedef struct RBNode {
    int data;
    int color; // RED or BLACK
    struct RBNode *left, *right, *parent;
} RBNode;
RBNode* createNode(int data) {
    RBNode *newNode = (RBNode *)malloc(sizeof(RBNode));
    newNode->data = data;
    newNode->color = RED;
    newNode->left = newNode->right = newNode->parent = NULL;
    return newNode;
}
void leftRotate(RBNode **root, RBNode *x) {
    RBNode *y = x->right;
    x->right = y->left;
    if (y->left != NULL) {
        y->left->parent = x;
    }
    y->parent = x->parent;
    if (x->parent == NULL) {
        *root = y;
    } else if (x == x->parent->left) {
        x->parent->left = y;
    } else {
        x->parent->right = y;
    }
    y->left = x;
    x->parent = y;
}
void rightRotate(RBNode **root, RBNode *y) {
    RBNode *x = y->left;
    y->left = x->right;
```

Output | Clear

```
/tmp/xBghnDugUC.o
In-order Traversal: 10 (BLACK) 15 (RED) 20 (BLACK) 25 (RED) 30 (BLACK)


=== Code Execution Successful ===
```

Feels hotter Now | Search | ENG IN | 14:11 01-08-2024

Programiz
C Online Compiler

main.c

Share | Run

```c
void rightRotate(RBNode **root, RBNode *y) {
    RBNode *x = y->left;
    y->left = x->right;
    if (x->right != NULL) {
        x->right->parent = y;
    }
    x->parent = y->parent;
    if (y->parent == NULL) {
        *root = x;
    } else if (y == y->parent->right) {
        y->parent->right = x;
    } else {
        y->parent->left = x;
    }
    x->right = y;
    y->parent = x;
}
void fixInsert(RBNode **root, RBNode *z) {
    while (z->parent != NULL && z->parent->color == RED) {
        if (z->parent == z->parent->parent->left) {
            RBNode *y = z->parent->parent->right;
            if (y != NULL && y->color == RED) {
                z->parent->color = BLACK;
                y->color = BLACK;
                z->parent->parent->color = RED;
                z = z->parent->parent;
            } else {
                if (z == z->parent->right) {
                    z = z->parent;
                    leftRotate(root, z);
                }
                z->parent->color = BLACK;
                z->parent->parent->color = RED;
                rightRotate(root, z->parent->parent);
            }
        } else {
```

Output | Clear

```
/tmp/xBghnDugUC.o
In-order Traversal: 10 (BLACK) 15 (RED) 20 (BLACK) 25 (RED) 30 (BLACK)


=== Code Execution Successful ===
```

main.c

Share | Run

```c
            RBNode *y = z->parent->parent->left;
            if (y != NULL && y->color == RED) {
                z->parent->color = BLACK;
                y->color = BLACK;
                z->parent->parent->color = RED;
                z = z->parent->parent;
            } else {
                if (z == z->parent->left) {
                    z = z->parent;
                    rightRotate(root, z);
                }
                z->parent->color = BLACK;
                z->parent->parent->color = RED;
                leftRotate(root, z->parent->parent);
            }
        }
    }
    (*root)->color = BLACK;
}
void insert(RBNode **root, int data) {
    RBNode *z = createNode(data);
    RBNode *y = NULL;
    RBNode *x = *root;

    while (x != NULL) {
        y = x;
        if (z->data < x->data) {
            x = x->left;
        } else {
            x = x->right;
        }
    }
    z->parent = y;
    if (y == NULL) {
        *root = z;
    } else if (z->data < y->data) {
```

Output | Clear

```
/tmp/xBghnDugUC.o
In-order Traversal: 10 (BLACK) 15 (RED) 20 (BLACK) 25 (RED) 30 (BLACK)


=== Code Execution Successful ===
```

Programiz
C Online Compiler

main.c

Share Run

```c
        if (z->data < x->data) {
            x = x->left;
        } else {
            x = x->right;
        }
    }
    z->parent = y;
    if (y == NULL) {
        *root = z;
    } else if (z->data < y->data) {
        y->left = z;
    } else {
        y->right = z;
    }
    fixInsert(root, z);
}
void inorderTraversal(RBNode *root) {
    if (root != NULL) {
        inorderTraversal(root->left);
        printf("%d (%s) ", root->data, root->color == RED ? "RED" : "BLACK");
        inorderTraversal(root->right);
    }
}
int main() {
    RBNode *root = NULL;
    insert(&root, 10);
    insert(&root, 20);
    insert(&root, 30);
    insert(&root, 15);
    insert(&root, 25);
    printf("In-order Traversal: ");
    inorderTraversal(root);
    printf("\n");
    return 0;
}
```

Output

Clear

```
/tmp/xBghnDugUC.o
In-order Traversal: 10 (BLACK) 15 (RED) 20 (BLACK) 25 (RED) 30 (BLACK)


=== Code Execution Successful ===
```