SHRI VILEPARLE KELAVANI MANDAL'S
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC ACCREDITED with "A" GRADE (CGPA : 3.18)

Department of Information Technology

COURSE CODE: DJS22ITL502 **DATE:** 16-10-24

COURSE NAME: Advanced Data Structures Laboratory **CLASS:** TY B. TECH

NAME: Anish Sharma **ROLL:** I011

EXPERIMENT NO. 6

CO/LO: Choose appropriate data structure and use it to design algorithm for solving a specific problem

AIM / OBJECTIVE: To implement various operations on a Splay Tree.

PROPERTIES OF SPLAY TREE:
Self-Adjusting: Automatically rearranges the tree after each operation by splaying the accessed node to the root.
Amortized O(log n) Time: Insert, delete, and search operations have an amortized time complexity of O(log n).
No Strict Balancing: Unlike AVL or Red-Black trees, Splay Trees don't enforce strict balancing but remain efficient via splaying.
Fast Access to Recently Used Elements: Frequently accessed nodes move closer to the root, improving access speed for these elements.
Simple Structure: Requires no extra data for balancing, just binary tree nodes and rotations.

TECHNOLOGY STACK USED: C SOURCE CODE:

SHRI VILEPARLE KELAVANI MANDAL'S
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC ACCREDITED with "A" GRADE (CGPA : 3.18)

Department of Information Technology

```c
#include <stdio.h>
#include <stdlib.h>
 struct Node { int key; struct
Node *left, *right;
};

struct Node* newNode(int key) { struct Node* node = (struct
    Node*)malloc(sizeof(struct Node)); node->key = key; node-
    >left = node->right = NULL; return node;
}

struct Node* rightRotate(struct Node* x)
    { struct Node* y = x->left; x->left =
    y->right; y->right = x; return y;
}
```

SHRI VILEPARLE KELAVANI MANDAL'S
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC ACCREDITED with "A" GRADE (CGPA : 3.18)

Department of Information Technology

```
struct Node* leftRotate(struct Node* x)
    { struct Node* y = x->right; x-
    >right = y->left; y->left = x;
    return y;
}

struct Node* splay(struct Node* root, int key) {
    if (root == NULL || root->key == key) return
    root;

    if (key < root->key) { if (root->left
        == NULL) return root;

        if (key < root->left->key) { root->left->left =
            splay(root->left->left, key);

            root = rightRotate(root);
        }

        else if (key > root->left->key) { root->left->right =
            splay(root->left->right, key);

            if (root->left->right != NULL) root->left
                = leftRotate(root->left);
        }


        return (root->left == NULL) ? root : rightRotate(root);
    }

    else { if (root->right == NULL) return root; if (key <

        root->right->key) { root->right->left = splay(root-

        >right->left, key);

            if (root->right->left != NULL) root->right
                = rightRotate(root->right);
        } else if (key > root->right->key) { root->right->right
        = splay(root->right->right, key); root =
        leftRotate(root);
        } return (root->right == NULL) ? root :
        leftRotate(root);
    }
}
```

SHRI VILEPARLE KELAVANI MANDAL'S
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC ACCREDITED with "A" GRADE (CGPA : 3.18)

Department of Information Technology

SHRI VILEPARLE KELAVANI MANDAL'S
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC ACCREDITED with "A" GRADE (CGPA : 3.18)

Department of Information Technology

```
struct Node* insert(struct Node* root, int key) {
    if (root == NULL) return newNode(key); root =
    splay(root, key); if (root->key == key)
    return root; struct Node* newnode =
    newNode(key);

    if (key < root->key) { newnode->right
        = root; newnode->left = root-
        >left; root->left = NULL;
    } else { newnode->left = root;
    newnode->right = root->right;
    root->right = NULL;
    }

    return newnode;
}

struct Node* deleteKey(struct Node* root, int key) {

    struct Node* temp; if (root == NULL) return

    NULL; root = splay(root, key); if (root->key !=

    key) return root;

    if (root->left == NULL) {
        temp = root; root =
        root->right;
    } else { temp =
        root;
        root = splay(root->left, key); root->right
        = temp->right;
    } free(temp);

    return root;
} void inOrder(struct Node* root)
{ if (root != NULL) {
inOrder(root->left); printf("%d
", root->key); inOrder(root-
>right);
    }
}
```

SHRI VILEPARLE KELAVANI MANDAL'S
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC ACCREDITED with "A" GRADE (CGPA : 3.18)

Department of Information Technology

```
int main() { struct Node*
    root = NULL;

    root = insert(root, 100);
    root = insert(root, 50); root
    = insert(root, 200); root =
    insert(root, 40); root =
    insert(root, 30); root =
    insert(root, 20);
 printf("Inorder traversal of the splay tree is:\n");
    inOrder(root);
    printf("\n");

    root = deleteKey(root, 50); printf("Inorder
    traversal after deleting 50:\n"); inOrder(root);
    printf("\n");

    return 0;
}
```

OUTPUT:

```
Inorder traversal of the splay tree is:
20 30 40 50 100 200
Inorder traversal after deleting 50:
20 30 40 100 200
20 30 40 50 100 200
Inorder traversal after deleting 50:
Inorder traversal after deleting 50:
20 30 40 100 200
```

CONCLUSION: In this experiment we implemented various operations on a Splay Tree.

REFERENCES:

1. Peter Brass, "Advanced Data Structures", Cambridge University Press, 2008 2. Robert Sedgewick & Kevin Wayne, "Algorithms", 4th Edition, Addison-Wesley Professional, 2011.