Name: Khushal Patil Class: SE-II (R-Batch)

Roll No.: 64

**Code:**

#include <iomanip>

#include <iostream>

using namespace std;

#define MAX\_NODES 10

class node {

   public:

    int data;

    node\* left;

    node\* right;

    node(int data = 0) {

        this->data = data;

        left = NULL;

        right = NULL;

    }

};

class OBST {

   public:

    int keys[MAX\_NODES] = {0};

    int p[MAX\_NODES] = {0};

    int q[MAX\_NODES] = {0};

    int n;

    int c[MAX\_NODES][MAX\_NODES] = {0};

    int w[MAX\_NODES][MAX\_NODES] = {0};

    int r[MAX\_NODES][MAX\_NODES] = {0};

    node\* root;

    OBST(int count, int keys\_arr[], int p\_arr[], int q\_arr[]) {

        root = NULL;

        n = count;

        for (int i = 0; i <= n; i++) {

            this->keys[i] = keys\_arr[i];

            this->p[i] = p\_arr[i];

            this->q[i] = q\_arr[i];

        }

    }

    void display\_matrix(int mat[MAX\_NODES][MAX\_NODES]) {

        for (int i = 0; i <= n; i++) {

            for (int j = 0; j <= n; j++) {

                if (i <= j) {

                    cout << setw(2) << mat[i][j] << " ";

                } else {

                    cout << "   ";

                }

            }

            cout << endl;

        }

    }

    int get\_w(int i, int j) {

        if (i >= j) {

            return q[i];

        }

        return w[i][j - 1] + p[j] + q[j];

    }

    void generate\_cost\_table() {

        for (int i = 0; i <= n; i++) {

            c[i][i] = 0;

            r[i][i] = 0;

            w[i][i] = q[i];

        }

        for (int diff = 1; diff <= n; diff++) {

            int i, j;

            i = 0;

            j = i + diff;

            while (j <= n) {

                int min\_cost = INT32\_MAX, min\_root;

                for (int k = i + 1; k <= j; k++) {

                    int cost = c[i][k - 1] + c[k][j];

                    if (cost < min\_cost) {

                        min\_cost = cost;

                        min\_root = k;

                    }

                }

                w[i][j] = get\_w(i, j);

                c[i][j] = min\_cost + w[i][j];

                r[i][j] = min\_root;

                i++;

                j++;

            }

        }

        cout << "Matrix C" << endl;

        display\_matrix(c);

        cout << "Matrix W" << endl;

        display\_matrix(w);

        cout << "Matrix R" << endl;

        display\_matrix(r);

    }

    node\* get\_node(int i, int j) {

        if (i == j) {

            return NULL;

        }

        int rij = r[i][j];

        node\* temp = new node(keys[rij]);

        temp->left = get\_node(i, rij - 1);

        temp->right = get\_node(rij, j);

        return temp;

    }

    void pre\_order(node\* temp) {

        if (temp != NULL) {

            cout << temp->data << " ";

            pre\_order(temp->left);

            pre\_order(temp->right);

        }

    }

    void generate\_obst() {

        root = get\_node(0, n);

        cout << "PreOrder Traversal Is" << endl;

        pre\_order(root);

        cout << endl;

    }

};

int main() {

    int k[] = {0,3,7,10,15,20,25};

    int p[] = {0,10,3,9,2,0,10};

    int q[] = {5,6,4,4,3,8,0};

    OBST o(6, k, p, q);

    o.generate\_cost\_table();

    o.generate\_obst();

    return 0;

}

**Output:**

Matrix C:

0 21 41 79 96 121 158

0 13 39 53 78 115

0 17 31 56 89

0 9 26 53

0 11 32

0 18

0

Matrix W:

5 21 28 41 46 54 64

6 13 26 31 39 49

4 17 22 30 40

4 9 17 27

3 11 21

8 18

0

Matrix R:

0 1 1 2 3 3 3

0 2 3 3 3 3

0 3 3 3 4

0 4 5 6

0 5 6

0 6

0

PreOrder Traversal Is:

10 3 7 25 20 15