S.No: 1	Exp. Name: ***Project Module***	**Date: 2024-06-11**

Aim:

Project Module

Source Code:

hello.c

ID: 2303811710421099

2023-2027-J

K.Ramakrishnan College of Technology

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct Student {
    int rollNumber;
    char name[50];
    int isPresent;
    struct Student* left;
    struct Student* right;
};

struct Student* createNode(int rollNumber, char name[]) {
    struct Student* newNode = (struct Student*)malloc(sizeof(struct Student));
    newNode->rollNumber = rollNumber;
    strcpy(newNode->name, name);
    newNode->isPresent = 0; // Initially marked as absent
    newNode->left = newNode->right = NULL;
    return newNode;
}

struct Student* insert(struct Student* root, int rollNumber, char name[]) {
    if (root == NULL) return createNode(rollNumber, name);

    if (rollNumber < root->rollNumber)
        root->left = insert(root->left, rollNumber, name);
    else if (rollNumber > root->rollNumber)
        root->right = insert(root->right, rollNumber, name);

    return root;
}

struct Student* search(struct Student* root, int rollNumber) {
    if (root == NULL || root->rollNumber == rollNumber)
        return root;

    if (rollNumber < root->rollNumber)
        return search(root->left, rollNumber);

    return search(root->right, rollNumber);
}

struct Student* deleteNode(struct Student* root, int rollNumber) {
    if (root == NULL) return root;

    if (rollNumber < root->rollNumber)
        root->left = deleteNode(root->left, rollNumber);
    else if (rollNumber > root->rollNumber)
        root->right = deleteNode(root->right, rollNumber);
    else {
        if (root->left == NULL) {
            struct Student* temp = root->right;
            free(root);
            return temp;
        } else if (root->right == NULL) {
```

```
            return temp;
        }

        struct Student* temp = root->right;
        while (temp && temp->left != NULL)
            temp = temp->left;

        root->rollNumber = temp->rollNumber;
        strcpy(root->name, temp->name);
        root->isPresent = temp->isPresent;
        root->right = deleteNode(root->right, temp->rollNumber);
    }

    return root;
}

void inorderTraversal(struct Student* root) {
    if (root != NULL) {
        inorderTraversal(root->left);
        printf("%d %s %d\n", root->rollNumber, root->name, root->isPresent);
        inorderTraversal(root->right);
    }
}

void markPresent(struct Student* root, int rollNumber) {
    struct Student* student = search(root, rollNumber);
    if (student != NULL) {
        student->isPresent = 1;
    } else {
        printf("Student with roll number %d not found.\n", rollNumber);
    }
}

void markAbsent(struct Student* root, int rollNumber) {
    struct Student* student = search(root, rollNumber);
    if (student != NULL) {
        student->isPresent = 0;
    } else {
        printf("Student with roll number %d not found.\n", rollNumber);
    }
}

void printPresentStudents(struct Student* root) {
    if (root != NULL) {
        printPresentStudents(root->left);
        if (root->isPresent == 1) {
            printf("%d %s\n", root->rollNumber, root->name);
        }
        printPresentStudents(root->right);
    }
}

void printAbsentStudents(struct Student* root) {
    if (root != NULL) {
        printAbsentStudents(root->left);
```

```
        }
        printAbsentStudents(root->right);
    }
}

int countStudents(struct Student* root) {
    if (root == NULL) return 0;
    return 1 + countStudents(root->left) + countStudents(root->right);
}

void printAllStudents(struct Student* root) {
    if (root != NULL) {
        printAllStudents(root->left);
        printf("Roll Number: %d, Name: %s\n", root->rollNumber, root->name);
        printAllStudents(root->right);
    }
}

int main() {
    struct Student* root = NULL;
    int choice, rollNumber;
    char name[50];

    do {
        printf("\nClass Attendance Management System\n");
        printf("1. Add Student\n");
        printf("2. Delete Student\n");
        printf("3. Update Student Name\n");
        printf("4. Mark Present\n");
        printf("5. Mark Absent\n");
        printf("6. Print Students\n");
        printf("7. Print Present Students\n");
        printf("8. Print Absent Students\n");
        printf("9. Print Total Student Count\n");
        printf("10. Print All Students with Roll Numbers\n");
        printf("0. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter student roll number: ");
                scanf("%d", &rollNumber);
                printf("Enter student name: ");
                scanf("%s", name);
                root = insert(root, rollNumber, name);
                break;
            case 2:
                printf("Enter student roll number to delete: ");
                scanf("%d", &rollNumber);
                root = deleteNode(root, rollNumber);
                break;
            case 3:
                printf("Enter student roll number to update: ");
                scanf("%d", &rollNumber);
```

```
                    printf("Enter new name: ");
                    scanf("%s", name);
                    strcpy(student->name, name);
                } else {
                    printf("Student with roll number %d not found.\n", rollNumber);
                }
                break;
            case 4:
                printf("Enter student roll number to mark present: ");
                scanf("%d", &rollNumber);
                markPresent(root, rollNumber);
                break;
            case 5:
                printf("Enter student roll number to mark absent: ");
                scanf("%d", &rollNumber);
                markAbsent(root, rollNumber);
                break;
            case 6:
                printf("Students in BST (inorder traversal):\n");
                inorderTraversal(root);
                break;
            case 7:
                printf("Present Students:\n");
                printPresentStudents(root);
                break;
            case 8:
                printf("Absent Students:\n");
                printAbsentStudents(root);
                break;
            case 9:
                printf("Total Student Count: %d\n", countStudents(root));
                break;
            case 10:
                printf("All Students with Roll Numbers:\n");
                printAllStudents(root);
                break;
            case 0:
                printf("Exiting...\n");
                break;
            default:
                printf("Invalid choice!\n");
        }
    } while (choice != 0);

    return 0;
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Hello World