Name: Prajakta Kumbhare

SY-IT(B)

Roll no.: 40

Topic: ADS Course project

Code:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include<time.h>

#include <windows.h>

char \*tarikhbaru[25];

void rectangle(int x,int y,int l,int b);// To make a rectangle starting from (x,y) with length l and breadth b

void gotoxy(int x, int y) {

    COORD coord;

    coord.X = x;

    coord.Y = y;

    SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), coord);

}//To move the cursor to x column and y row of output screen

COORD coord = {0, 0};

struct Node {

    char name[200]; // Increased size to accommodate first name and surname

    int prn;

    int days\_attended;

    float attendance;

    int attendanceEdited;

    struct Node\* left;

    struct Node\* right;

};

typedef struct Node Node;

Node\* createNode(char name[], int prn, int days\_attended, int lectures) {

    Node\* newNode = (Node\*)malloc(sizeof(Node));

    newNode->attendanceEdited = 0;

    if (newNode == NULL) {

        printf("Memory allocation failed.\n");

        exit(EXIT\_FAILURE);

    }

    strcpy(newNode->name, name);

    newNode->prn = prn;

    newNode->days\_attended = days\_attended;

    newNode->attendance = (days\_attended \* 100.0) / lectures;

    newNode->left = NULL;

    newNode->right = NULL;

    return newNode;

}

Node\* insertNode(Node\* root, char name[], int prn, int days\_attended, int lectures) {

    if (root == NULL) {

        return createNode(name, prn, days\_attended, lectures);

    } else {

        if (prn < root->prn) {

            root->left = insertNode(root->left, name, prn, days\_attended, lectures);

        } else {

            root->right = insertNode(root->right, name, prn, days\_attended, lectures);

        }

        return root;

    }

}

void inorderTraversal(Node\* root) {

    if (root != NULL) {

        inorderTraversal(root->left);

        printf("%-12d %-30s %.2f%%\n", root->prn, root->name, root->attendance);

        inorderTraversal(root->right);

    }

}

void writeToFile(Node\* root, FILE\* file) {

    if (root != NULL) {

        writeToFile(root->left, file);

        fprintf(file, "%-12d %-30s %.2f%%\n", root->prn, root->name, root->attendance);

        writeToFile(root->right, file);

    }

}

void generateWarningLetters(Node\* root) {

    if (root != NULL) {

        generateWarningLetters(root->left);

        generateWarningLetters(root->right);

        if (root->attendance < 75.0) {

            char filename[200]; // Increased size for filename

            sprintf(filename, "%s\_letter.txt", root->name);

            FILE\* file = fopen(filename, "w");

            fprintf(file, "Student Name: %s\nPRN: %d\n\nDear Mr./Ms.,\n\nSub: Warning Letter.\nThis is to inform you that your child %s has been continually failing to attend his/her class.\nThe total attendance is %.2f%% which is less than the 75%% criteria of the college.\nSeveral warnings were issued to your child in the past, but you have failed to address this matter.\nWe expect you to take immediate and serious action to help your child improve his/her attendance and expect to see some positive results by the end of this month. \nOur teachers are available for assistance in this regard should you require any help in bringing about a significant improvement in your child's attendance.\n\n\n\nThank you.\t\t\t\t\t\t\t\tSignature", root->name, root->prn, root->name, root->attendance);

            fclose(file);

        }

    }

}

Node\* searchByPRN(Node\* root, int prn) {

    if (root == NULL || prn == root->prn) {

        return root;

    }

    if (prn < root->prn) {

        return searchByPRN(root->left, prn);

    }

    return searchByPRN(root->right, prn);

}

void editAttendance(Node\* root, int prn, int newDaysAttended, int lectures) {

    if (root == NULL) {

        printf("Student with PRN %d not found.\n", prn);

        return;

    }

    if (prn == root->prn) {

        if (root->attendanceEdited) {

            printf("Attendance for PRN %d already edited. Cannot edit again.\n", prn);

        } else if (newDaysAttended <= lectures) {

            root->days\_attended = newDaysAttended;

            root->attendance = (newDaysAttended \* 100.0) / lectures;

            root->attendanceEdited = 1; // Set attendance edit flag

            printf("Attendance updated for PRN %d.\n", prn);

        } else {

            printf("Invalid input. Days attended cannot exceed total lectures.\n");

        }

    } else if (prn < root->prn) {

        editAttendance(root->left, prn, newDaysAttended, lectures);

    } else {

        editAttendance(root->right, prn, newDaysAttended, lectures);

    }

}

void displayStatistics(Node\* root) {

    if (root == NULL) {

        printf("No data available.\n");

        return;

    }

    int totalStudents = 0;

    int totalAttendance = 0;

    void calculateStatistics(Node\* node)

    {

        if (node != NULL) {

            totalStudents++;

            totalAttendance += node->days\_attended;

            calculateStatistics(node->left);

            calculateStatistics(node->right);

        }

    }

    calculateStatistics(root);

    float averageAttendance = (float)totalAttendance / totalStudents;

    printf("Total Students: %d\n", totalStudents);

    printf("Total Attendance: %d\n", totalAttendance);

    printf("Average Attendance: %.2f%%\n", averageAttendance);

}

void wait(int sec)

{

    clock\_t end\_wait;

    end\_wait = clock() + sec \* CLK\_TCK ;

    while (clock() < end\_wait) {}

}

void loginsuccess()

{

    system("cls");

    gotoxy(3,2); printf("20%% Loading.");

    wait(1);    /\* Wait for 1 seconds \*/

    system("cls");

    gotoxy(3,2); printf ("40%% Loading. .");

    wait(1);

    system("cls");

    gotoxy(3,2); printf ("60%% Loading. . .");

    wait(1);

    system("cls");

    gotoxy(3,2); printf ("80%% Loading. . . .");

    wait(1);

    system("cls");

    gotoxy(3,2); printf ("100%% Loading. . . . .");

    wait(1);

    system("cls");

}

Node\* root = NULL;

void main\_menu()

{

    int choice;

    int lectures = 0;

    do{

        printf("\n\n\t\tSTUDENT ATTENDANCE SYSTEM");

        printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

    printf("Welcome\n");

    printf("I   Press 1 to Add a student.\n");

    printf("II  Press 2 to Display.\n");

    printf("III Press 3 to Create the data file.\n");

    printf("IV  Press 4 to Generate warning letters.\n");

    printf("V   Press 5 to Search by PRN.\n");

    printf("VI  Press 6 to Edit Attendance.\n");

    printf("VII Press 7 to Display Attendance Statistics.\n");

    printf("VIII Press 8 to Exit.\n");

    printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

        printf("\n\nEnter choice: ");

        scanf("%d", &choice);

        switch (choice) {

            case 1: {

                char firstName[100];

                char lastName[100];

                int prn;

                int days\_attended;

                printf("Enter the name of the student: ");

                scanf("%s", firstName);

                printf("Enter the last name of the student: ");

                scanf("%s", lastName);

                strcat(firstName, " "); // Add space between first name and last name

                strcat(firstName, lastName); // Concatenate first and last name

                printf("Enter PRN (8 digits): ");

                if (scanf("%d", &prn) != 1 || prn < 10000000 || prn > 99999999) {

                    printf("Invalid PRN. Please enter an 8-digit PRN.\n");

                    while(getchar() != '\n'); // Clear input buffer

                    break;

                }

                printf("Give the total number of lectures: ");

                scanf("%d", &lectures);

                printf("Enter the number of days attended out of %d: ", lectures);

                scanf("%d", &days\_attended);

                root = insertNode(root, firstName, prn, days\_attended, lectures);

                break;

            }

            case 2: {

                printf("List is:\n");

                printf("%-12s %-30s %s\n", "PRN", "Name", "Attendance");

                inorderTraversal(root);

                break;

            }

            case 3: {

                FILE\* file = fopen("Data.txt", "w");

                if (file == NULL) {

                    printf("Error creating the file.\n");

                    break;

                }

                fprintf(file, "%-12s %-30s %s\n", "PRN", "Name", "Attendance");

                writeToFile(root, file);

                fclose(file);

                printf("Data file created successfully.\n");

                break;

            }

            case 4: {

                generateWarningLetters(root);

                printf("Warning letters generated successfully.\n");

                break;

            }

            case 5: {

                int prn;

                printf("Enter PRN to search: ");

                scanf("%d", &prn);

                Node\* result = searchByPRN(root, prn);

                if (result != NULL) {

                    printf("%-12d %-30s %.2f%%\n", result->prn, result->name, result->attendance);

                } else {

                    printf("Student with PRN %d not found.\n", prn);

                }

                break;

            }

            case 6: {

            int prn, newDaysAttended;

            printf("Enter PRN to edit attendance: ");

            scanf("%d", &prn);

        Node\* student = searchByPRN(root, prn);

        if (student != NULL) {

            printf("Current Attendance for PRN %d: %.2f%%\n", student->prn, student->attendance);

            printf("Enter new days attended: ");

            scanf("%d", &newDaysAttended);

            if (newDaysAttended <= lectures) {

                editAttendance(root, prn, newDaysAttended, lectures);

                printf("Attendance updated for PRN %d.\n", prn);

            } else {

                printf("Invalid input. Days attended cannot exceed total lectures.\n");

            }

            } else {

                printf("Student with PRN %d not found.\n", prn);

            }

                break;

            }

            case 7: {

                displayStatistics(root);

                break;

            }

            case 8:

                printf("\nExit system..\n");

                break;

            default:

                printf("Please enter a valid choice.\n");

        }

    } while (choice != 8); // This was previously misplaced

}

void adminLogin() //user will give attempt only 3 times

{

    char user\_name[10] = "admin"; //set username and password

    char password[10] = "admin";

    char name[10],passcode[10];

    int a,b,i=3;

    printf("\n\n-----\nLogin\n-----\n");

    do{

        printf("\nUsername : ");

        scanf(" %s", &name);

        printf("Password : ");

        scanf(" %s", &passcode);

        a = strcmp(user\_name,name);    //string compare function.. to compare username and password

        b = strcmp(password,passcode);

        i--;

    }while(a!=0 && b!=0 && i>0);

    if(a==0 && b==0)

    {

        loginsuccess();

        main\_menu();         //go to main menu function

    }

    if(i<=0) //if more than 3 times,the user does not have permission to access

    {

        printf("\nAccess Denied!!\n");

        system("pause");    //system will pause

        system("cls");  //clear screen

    }

}

void studentLogin()

{

    system("cls");

    int prn;

    rectangle(0,0,80,23);

    gotoxy(20,2); printf("Student Attendance Lookup");

    gotoxy(10,4); printf("----------------------------------------------");

    gotoxy(10,8); printf("Enter student PRN : ");

    scanf(" %d", &prn);

    Node\* result = searchByPRN(root, prn);

    if (result != NULL) {

        gotoxy(10,10); printf("PRN: %-12d", result->prn);

        gotoxy(10,12); printf("Name: %-30s", result->name);

        gotoxy(10,14); printf("Attendance: %.2f%%", result->attendance);

    } else {

        gotoxy(10,10); printf("Student with PRN %d not found.", prn);

    }

    // Add the following lines to pause and wait for user input before returning to main\_menu

    gotoxy(10,18);

    printf("\n\nPress any key to continue...");

    getchar(); // Consume the newline character left by scanf

    getchar(); // Wait for user to press any key

    // You can add additional logic here if needed

    // Return to main\_menu

    return;

}

void rectangle(int x,int y,int l,int b)

{

    int i,m;

    gotoxy(x,y); printf("%c",201);

    for(i=x+1;i<l-1;i++)

    {

        gotoxy(i,y);

        printf("%c",205);

    }

    gotoxy(i,y); printf("%c",187);

    for (m=y+1;m<b;m++)

    {

        gotoxy(x,m);

        for(i=x;i<l;i++)

        {

            if(i==x||i==l-1)

            {

                gotoxy(i,m); printf("%c",186);

            }

        }

    }

    gotoxy(x,m); printf("%c",200);

    for(i=x+1;i<l-1;i++)

    {

        gotoxy(i,m);

        printf("%c",205);

    }

    gotoxy(i,m); printf("%c",188);

}

int main() {

  int num;

    do{

        system("cls");

        rectangle(8,6,70,20);

        gotoxy(31,8);  printf("Main menu");

        gotoxy(31,9);  printf("---------");

        gotoxy(10,11); printf("1 - Admin Login");

        gotoxy(10,13); printf("2 - Student Login");

        gotoxy(10,15); printf("0 - Exit");

        gotoxy(10,17); printf("Enter choice: ");

        scanf(" %d", &num);

        switch(num)

        {

            case 1:

                system("cls");

                adminLogin();

                break;

            case 2:

                studentLogin();

                break;

            case 0: //27 is ASCII code of escape key, means program exits when user presses Esc key instead of 1 or 2

                printf("\n\n\n\n");

                exit(0);

                break;

        }

    }while(1);//infinite loop incase any key other than 0, 1, or 2 is pressed.

    return 0;

}