**Clinic Appointment & Earnings Management System**

#include <iostream>

using namespace std;

const int MAX\_DOCTORS = 3;

const int MAX\_SLOTS = 5;

string doctors[MAX\_DOCTORS] = {"Dr. Alyan", "Dr. Shafaq", "Dr. Ayesha"};

float dailyEarnings[MAX\_DOCTORS] = {0};

bool slots[MAX\_DOCTORS][MAX\_SLOTS] = {false};

//Function prototypes

void displaymenu();

void addpatient();

void displaySlots();

void bookAppointment();

void cancelAppointment();

void inputDailyEarnings();

void calculateMonthlyEarnings();

int main() {

int choice;

do {

displaymenu();

cout << "Enter choice: ";

cin >> choice;

switch (choice) {

case 1:

addpatient();

bookAppointment();

break;

case 2:

cancelAppointment();

break;

case 3:

inputDailyEarnings();

break;

case 4:

calculateMonthlyEarnings();

break;

case 5:

cout << "Exiting program...\n";

break;

default:

cout << "Invalid choice\n";

}

} while (choice != 5);

return 0;}

void displaymenu() {

cout << "\nMenu:\n";

cout << "1. Book Appointment\n";

cout << "2. Cancel Appointment\n";

cout << "3. Input Daily Earnings\n";

cout << "4. Calculate Monthly Earnings\n";

cout << "5. Exit\n";

}

void addpatient() {

string patient;

cout << "\nPatient's name: ";

cin >> patient;

}

void displaySlots() {

cout << "\nAvailable Slots:\n";

for (int i = 0; i < MAX\_DOCTORS; i++) {

cout << "Doctor " << doctors[i] << ":\n";

for (int j = 0; j < MAX\_SLOTS; j++) {

cout << j + 1 << ". ";

if (slots[i][j]) {

cout << "Booked\n";

} else {

cout << "Available\n";

}

}

cout << "\n";

}}

void bookAppointment() {

int doctorIndex, slotIndex;

displaySlots();

cout << "Select doctor (1-" << MAX\_DOCTORS << "): ";

cin >> doctorIndex;

doctorIndex--; // Adjusting index for array

if (doctorIndex < 0 || doctorIndex >= MAX\_DOCTORS) {

cout << "Invalid doctor selection\n";

return;

}

cout << "Select slot (1-" << MAX\_SLOTS << "): ";

cin >> slotIndex;

slotIndex--; // Adjusting index for array

if (slotIndex < 0 || slotIndex >= MAX\_SLOTS) {

cout << "Invalid slot selection\n";

return;

}

if (slots[doctorIndex][slotIndex]) {

cout << "Slot already booked\n";

} else {

slots[doctorIndex][slotIndex] = true;

cout << "Appointment booked with " << doctors[doctorIndex] << "\n";

}

}

void cancelAppointment() {

int doctorIndex, slotIndex;

displaySlots();

cout << "Select doctor (1-" << MAX\_DOCTORS << "): ";

cin >> doctorIndex;

doctorIndex--; // Adjusting index for array

if (doctorIndex < 0 || doctorIndex >= MAX\_DOCTORS) {

cout << "Invalid doctor selection\n";

return;

}

cout << "Select slot (1-" << MAX\_SLOTS << "): ";

cin >> slotIndex;

slotIndex--; // Adjusting index for array

if (slotIndex < 0 || slotIndex >= MAX\_SLOTS) {

cout << "Invalid slot selection\n";

return;

}

if (!slots[doctorIndex][slotIndex]) {

cout << "Slot not booked\n";

} else {

slots[doctorIndex][slotIndex] = false;

cout << "Appointment canceled with " << doctors[doctorIndex] << "\n";

}

}

void inputDailyEarnings() {

int doctorIndex;

float earnings;

for (int i = 0; i < MAX\_DOCTORS; i++) {

cout << "Enter daily earnings for " << doctors[i] << ": ";

cin >> earnings;

dailyEarnings[i] += earnings;

}

}

void calculateMonthlyEarnings() {

for (int i = 0; i < MAX\_DOCTORS; i++) {

cout << "Monthly earnings for " << doctors[i] << ": $" << dailyEarnings[i] \* 30 << "\n";

}

}