Liceul Teoretic “Stephan Ludwig Roth”

Mediaș

Proiect

pentru obținerea

Atestatului Profesional la Informatică

Îndrumător: Autor:

prof. Enea Constantin Oinăcaru Adrian‑Cristian

Mediaș

2022

Liceul Teoretic “Stephan Ludwig Roth”

Mediaș

GESTIUNEA UNUI HOTEL

Mediaș

2022

Cuprins

[1.Introducere 1](#_Toc104148637)

[2.Managementul Hotelier 2](#_Toc104148638)

[2.1.Concepte de Bază pentru Gestiunea unui Hotel 2](#_Toc104148639)

[2.2.Prezentarea Programului 3](#_Toc104148640)

[2.2.1.Ecranul Principal 3](#_Toc104148641)

[2.2.2.Administrarea Programului 3](#_Toc104148642)

[2.2.3.Structurile ce stau la Baza Programului 4](#_Toc104148643)

[2.2.3.1.Managementul Resurselor Umane 4](#_Toc104148644)

[2.2.3.2.Evidența Structurilor de Cazare 7](#_Toc104148645)

[2.2.3.3.Analiza Feedbackului oferit de Clienți 9](#_Toc104148646)

[2.3.Aplicația care Realizează Gestiunea Hotelului 10](#_Toc104148647)

[2.3.1.Interfața Programului 10](#_Toc104148648)

[2.3.2.Managementul Angajaților 11](#_Toc104148649)

[2.3.3.Gestionarea Camerelor 15](#_Toc104148650)

[2.3.4.Programul Destinat Clienților 17](#_Toc104148651)

[3.Concluzii 19](#_Toc104148652)

[4.Bibliografie 20](#_Toc104148653)

[5.Anexă (Codul Sursă) 21](#_Toc104148654)

1.Introducere

În alegerea temei au contribuit mai mulți factori:

Actualitatea temei: hotelurile sunt un punct central în vederea cazării turiștilor, iar în prezent, se folosesc tot mai multe metode de a automatiza procesul de identificare a părților componente, ce fac posibilă buna funcționare a unității de cazare. Printre părțile menționate anterior, se află unele entități cheie, cum ar fi personalul hotelului, cazarea clienților în camere special amenajate, precum și părerea clienților care au folosit serviciile oferite. Este important de ținut evidența a tuturor acestor aspecte, pentru a putea îmbunătăți experiența clienților pe viitor, dar și pentru a realiza o imagine de ansamblu mai atractivă.

Aplicabilitatea programului conceput: s‑a creat un program care este util în gestiunea personalului, a camerelor și a clienților. Utilizatorii pot efectua numeroase acțiuni, printre care se numără adăugarea sau ștergerea. Aceste acțiuni au rolul de a păstra în permanență o listă actualizată a tuturor părților componente în parte.

Ușurința accesării funcțiilor programului: la început, s‑a avut în vedere nu doar crearea unui cod funcțional, ci și unul care să fie accesibil pentru o varietate mare de persoane, indiferent de gradul lor de cunoștințe în domeniu. Există întrebări interactive, care oferă utilizatorului opțiuni de alegere și implicit mai mult control asupra programului și care, combinate cu o interfață ușor de utilizat, contribuie la realizarea unei munci mai ușoare pentru persoana din spatele monitorului.

Sursa de inspirație pentru acest proiect o reprezintă pasiunea pentru înțelegerea acțiunilor, ce stau la baza unei afaceri de succes. S‑a optat pentru gestiunea unui hotel, pentru că această temă reprezintă o afacere pe un plan mai amplu și care, după ridicarea restricțiilor cauzate de pandemie și implicit gradul tot mai mare de ocupare în industria hotelieră, presupune un proiect important. Hotelul este structura de primire amenajată în clădiri sau în corpuri de clădiri, care pune la dispoziția turiștilor camere, garsoniere sau apartamente dotate corespunzător,asigură prestări de servicii specifice și dispune de recepție și de spații de alimentație în incintă.

2.Managementul Hotelier

2.1.Concepte de Bază pentru Gestiunea unui Hotel

La baza proiectului stă conceptul teoretic. Pentru o funcționare optimă a programului a fost nevoie în primul rând de o documentație în acest domeniu și de înțelegerea unor princpii teoretice fundamentale de gestionare.

Recepția (Front‑Office) reprezintă o secțiune importantă din componența hotelelui, fiind primul contact al clientului cu unitatea, dar și mijlocul de a comunica ulterior cu personalul. Fiecare funcție a acestui compartiment este bine definită și este împărțită pe mai multe categorii. Recepția propriu‑zisă are ca rol primirea clienților și disponibilitatea de a‑i îndruma în caz de nevoie. De asemenea, aici se efectuează și repartizarea camerelor, urmărind astfel camerele disponibile pentru ocupare. Datele clientului sunt preluate și transmise mai departe către casierie, unde acesta va fi înregistrat și unde va efectua plata, pentru a putea beneficia de serviciile oferite.

Disponibilitatea camerelor, precum și rolul și importanța rezervărilor sunt un alt punct teoretic important. Pentru clienți, rezervarea reprezintă siguranța închirierii spațiului dorit, fapt ce apare mai frecvent în apropierea unor evenimente importante și la care de regulă participă o mulțime considerabilă de persoane. Pe lângă aceasta, clienții vor putea planifica din timp camera dorită după preferință, iar tariful va fi cunoscut de la început, astfel că în eventualitatea unei răzgândiri, clientul poate opta pentru alte servicii ce i se potrivesc mai bine.

Pregătirea primirii clientului are loc după o rezervare (la o dată anume sau la momentul sosirii), prin care se confirmă astfel cazarea clientului în unitate. Acesta este momentul în care se verifică actele sau documentele necesare și constă în realizarea listei de sosiri, ce include numele clientului, tipul de cameră rezervat, numărul de persoane ce urmează să ocupe camera și, opțional, cerințe speciale ale clientului.

Primirea clienților este cel mai important proces și reprezintă baza funcționării hotelului. Orice persoană care intră în unitate trebuie tratată cu respect și îndrumată către punctele de interes. La primirea clienților se ține cont în mare parte de gradul de ocupare a hotelului. În cazul în care mai sunt camere disponibile, clientului i se va oferi inițial opțiunea camerei cele mai scumpe.

2.2.Prezentarea Programului

2.2.1.Ecranul Principal

Scopul programului este de a automatiza înregistrările necesare pentru a menține o listă actuală a părților componente ce fac posibilă funcționarea hotelului. La început, utilizatorul va fi întâmpinat de un mesaj de bun venit, însoțit de o instrucțiune care clarifică ce acțiune așteaptă programul din partea persoanei din spatele monitorului. Programul conține astfel trei puncte centrale de interes, și anume:

* Secțiunea angajați
* Secțiunea camere
* Secțiunea clienți

În plus, pe lângă cele trei puncte menționate anterior, programul dispune și de opțiunea de “Ieșire din aplicație”, care, dacă este accesată, va înștiința utilizatorul că a părăsit zona de gestiune a hotelului.

Ati parasit zona de gestiune a hotelului nostru. Pentru mai multe detalii nu ezitati sa ne contactati!

2.2.2.Administrarea Programului

Pentru folosirea aplicației este necesară utilizarea fișierelor externe, care conțin informații inițiale în legătură cu fiecare secțiune din proiect. Aici se găsesc date inițiale referitoare la personalul existent în hotel, alături de unele date personale, informații despre camerele disponibile, cât și detalii despre clienții care au acordat un feedback despre experiența lor pe timpul cazării. Modificările efectuate vor fi salvate pâna la ieșirea completă din aplicație. În cazul în care se dorește revenirea la punctul inițial, utilizatorul va închide programul și îl va redeschide. Prin acest proces vor rămâne valabile doar datele din fișierele externe.

2.2.3.Structurile ce stau la Baza Programului

Cele trei structuri existente în proiect conțin la rândul lor mai multe acțiuni executabile:

2.2.3.1.Managementul Resurselor Umane

În cazul în care utilizatorul dorește să acceseze “Sectiunea angajati”, va fi întâmpinat de un mesaj care confirmă că se află în secțiunea dorită. De aici, utilizatorul poate efectua una dintre acțiunile disponibile, apăsând pe numărul de ordine respectiv.

Dacă utilizatorul dorește să vizualizeze lista actuală a angajaților, va apăsa tasta “1”.

1. Prezentarea angajatilor

În această secțiune va apărea inițial un mesaj, care conține informații referitoare la numărul de angajați. Înainte de efectuarea vreunei modificări, aici vor fi afișați angajații trecuți în fișierul extern corespunzător angajaților, numit “angajati.in”. După adăugarea/ștergerea/modificarea datelor a cel puțin unui angajat, lista angajaților se va actualiza automat.

Accesarea celei de a doua secțiuni reprezintă căutarea angajaților după nume.

2. Cautarea angajatilor dupa nume

Inițial va fi afișată lista actuală a angajaților. Sub această listă, apare un mesaj care îi oferă posibilitatea utilizatorului să introducă numele angajatului pe care dorește să îl caute. Dacă numele introdus corespunde cu cel puțin unul al vreunui angajat din listă, pe ecran se va afișa doar angajatul căutat, împreună cu datele sale personale. În cazul în care numele introdus nu corespunde cu numele vreunui angajat, se va afișa un mesaj care informează utilizatorul că angajatul nu se află în listă.

În cazul în care se dorește să se adauge unul sau mai mulți angajați, se va apăsa tasta “3”.

3. Adaugarea unui nou angajat

În continuare, programul așteaptă să fie introduse inițial numele și prenumele noului angajat, pe care le compară cu cele ale angajaților deja existenți în listă, iar dacă numele și prenumele sunt identice cu ale unui alt angajat, se va afișa un mesaj, care îl atenționează pe utilizator că angajatul se află deja în listă.

Angajatul se afla deja in baza de date!

Asigurati-va ca angajatul nu se afla deja in baza de date si introduceti informatiile acestuia!

Dacă numele și prenumele introduse nu se află în listă, se poate trece la introducerea vârstei și a salariului. După ce toate datele necesare au fost introduse, programul oferă posibilitatea utilizatorului de a confirma că dorește să adauge angajatul. Prin această procedură, se poate renunța la adăugare, în cazul unei erori sau a unei răzgândiri, sau se poate confirma, caz în care angajatul, împreună cu informațiile sale, este adăugat în listă. Angajatul nou introdus va fi evidențiat prin culoare, iar utilizatorul decide dacă dorește să adauge alți angajați sau nu.

Dacă utilizatorul apasă tasta “4”, are opțiunea de a șterge angajați din listă.

4. Stergerea unui angajat

Inițial va fi întâmpinat de lista curentă a angajaților. Sub această listă se află un mesaj care are rol informativ asupra răspunsului pe care programul se așteaptă să îl primească. După identificarea angajatului care urmează să fie eliminat din listă, se va introduce indexul (numărul de ordine) acestuia. Pe ecran se va afișa doar angajatul selectat, împreună cu un mesaj de confirmare, care oferă posibilitatea de a renunța la ștergere sau de a accepta eliminarea.

Angajatul: 3

Date personale:

Nume: Marinescu

Prenume: Florin

Varsta: 35 de ani

Salariul: 4871.12 lei

==============================

Confirmati ca doriti stergerea angajatului selectat din baza de date?

În continuare, utilizatorul va fi întrebat dacă dorește să elimine și alți angajați. În cazul unui răspuns afirmativ se va relua procesul menționat anterior, iar în caz contrar, se va afișa lista angajaților, după eliminările efectuate.

În “secțiunea angajați” se pot efectua și modificări ale datelor personale, apăsând tasta “5”.

5. Modificarea datelor a unui angajat

Din lista de angajați se va alege indexul (numărul de ordine) angajatului la care urmează să se efectueze modificări. Angajatul selectat va fi afișat pe ecran, însoțit de o întrebare la care utilizatorul va introduce ca răspuns ceea ce dorește să modifice, cum ar fi salariul. După aceasta, va fi introdusă noua valoare. Programul va afișa în continuare un mesaj de confirmare prin care se poate renunța la modificare sau se poate confirma aceasta. Dacă modificarea se confirmă, va apărea pe ecran un mesaj cu ceea ce s‑a modificat, dar și cu angajatul selectat la care s‑a efectuat modificarea. Utilizatorul are apoi opțiunea de a vedea lista actualizată a angajaților, iar cei la care s-au efectuat modificări sunt evidențiați prin culoare.

O ultimă opțiune în această secțiune o reprezintă ordonarea.

6. Ordonarea angajatilor

Apăsând tasta “6”, utilizatorul poate alege criteriul după care dorește să ordoneze angajații. În plus, după alegerea criteriului de ordonare, programul poate realiza atât o ordonare crescătoare, cât și o ordonare descrescătoare. După ce utilizatorul a specificat atât criteriul de ordonare, cât și tipul de ordonare, lista va fi redată conform cerinței utilizatorului. Se pot efectua alte ordonări ulterioare, dacă răspunsul la întrebarea ce este afișată după ultimul angajat din lista ordonată este “da”.

Pentru a ieși din “secțiunea angajați”, utlizatorul va apăsa tasta “0”, fiind astfel redirecționat către meniul principal.

0. Iesire din sectiunea angajati

2.2.3.2.Evidența Structurilor de Cazare

Dacă utilizatorul accesează “Sectiunea camere”, va fi întâmpinat de un mesaj, care confirmă că se află în secțiunea dorită.

La accesarea primei secțiuni vor fi afișate camerele disponibile.

1. Afisarea camerelor disponibile

Inițial, se vor afișa doar camerele care au fost înregistrate în fișierul extern numit “camere.in”, urmând ca după modificări, lista camerelor să se actualizeze constant, până la părăsirea aplicației.

Dacă utilizatorul dorește să adauge camere pe lista celor disponibile, va apăsa tasta “2”.

2. Adaugarea unei camere

Programul așteaptă informații referitoare la camera ce a devenit disponibilă pentru clienți, și anume numărul, tipul și costul per noapte al camerei. Dacă numărul camerei introdus se află deja pe lista camerelor disponibile, se afișează un mesaj care înștiințează utilizatorul asupra acestui fapt; astfel poate fi introdus un nou număr.

Camera introdusa este deja disponibila!

Dacă informațiile camerei sunt introduse corect și camera nu se afla deja pe listă, programul va afișa un mesaj de confirmare prin care utilizatorul poate renunța la adăugare sau o poate confirma. În cazul în care se renunță la adăugare, programul va afișa un mesaj corespunzător, iar în caz contrar, camera va fi adăugată în listă. Pe ecran va fi afișată lista actualizată, iar camera nou introdusă va fi evidențiată prin culoare. Utilizatorul are apoi opțiunea de a adăuga mai multe camere, dacă este cazul, repetând astfel procedurile menționate anterior.

Rezervarea camerelor este posibilă, apăsând tasta “3”.

3. Rezervarea unei camere

Pe ecran va fi afișată lista actuală a camerelor disponibile. În eventualitatea unei rezervări, se va proceda ca la ștergere, astfel se elimină camera selectată din lista celor disponibile. Programul așteaptă un identificator ca răspuns pentru a putea selecta camera dorită. Identificatorul este reprezentat de numărul camerei.

Dacă numărul camerei introduse nu se potrivește cu niciun număr din listă, înseamnă ca acea cameră fie nu există în hotel, fie este deja ocupată. Dacă există o potrivire între numărul camerelor din listă și numărul introdus, atunci camera selectată va fi afișată pe ecran, împreună cu un mesaj, care oferă utilizatorului posibilitatea să confirme sau să renunțe la rezervare.

Aceasta este camera selectata:

Camera nr. 56

Informatii generale:

Tip: simpla

Cost: 160 de lei/noapte

==============================

Confirmati ca doriti rezervarea acestei camere pentru un client?

Dacă rezervarea este confirmată, camera este eliminată din listă, iar utilizatorul are opțiunea de a rezerva și alte camere dacă este necesar, iar în caz contrar poate părăsi zona de rezervare a camerelor.

Pentru ordonare se va apăsa tasta “4”.

4. Ordonarea camerelor

Urmează completarea cu criteriul de ordonare (număr/cost) și cu tipul ordonării (crescător/descrescător). După efectuarea ordonării dorite, se poate realiza altă ordonare, astfel că dacă răspunde afirmativ, procesul menționat anterior se repetă, iar în caz contrar va fi redirecționat către “secțiunea camere”. Pentru a ieși din această secțiune se va apăsa tasta “0”.

0. Iesire din sectiuena camere

2.2.3.3.Analiza Feedbackului oferit de Clienți

Dacă persoana din spatele monitorului accesează “Sectiunea clienți”, programul va afișa mai întâi un mesaj care confirmă că se află în secțiunea dorită, după care afișează acțiunile ce se pot efectua în această secțiune.

Dacă utilizatorul apasă tasta “1”, programul va afișa clienții care au oferit un feedback despre experiența lor.

1. Prezentarea recenziilor clientilor

Datele memorate despre clienți sunt numele și prenumele lor, iar feedback‑ul lor constă în recenzia acordată, dar și în sugestiile și reclamațiile lor. Inițial, apar recenziile clienților înregistrați în fișierul extern “clienti.in”. În cazul formulării unei recenzii, pot fi adăugate și altele, însoțite , opțional, de sugestii și reclamații.

În cazul în care utilizatorul dorește să adauge o recenzie a unui client, va apăsa tasta“2”.

2. Adaugarea recenziei unui client

Programul va aștepta inițial numele și prenumele clientului. Dacă acestea corespund cu numele și prenumele unui client, care a adăugat deja o recenzie, programul va afișa un mesaj corespunzător. Dacă numele și prenumele nu corespund cu cele ale vreunui client deja înregistrat, se poate trece la recenzia sa, sugestiile, dar și reclamațiile acestuia. După introducerea feedback‑ului, programul afișează un mesaj de confirmare, iar utilizatorul poate decide dacă renunță la adăugarea recenziei sau dacă o acceptă. În cazul în care se renunță la adăugarea recenziei, utilizatorul poate reveni în această secțiune de adăugare și corecta eventuale greșeli care au condus la decizia de a renunța la adăugare. Dacă se confirmă adăugarea, lista recenziilor clienților va fi actualizată și afișată, iar recenzia nou adăugată va fi evidențiată prin culoare. Programul oferă apoi posibilitatea de a continua adăugarea altor recenzii dacă este necesar.

Utilizatorul poate vedea sugestiile și reclamațiile separat, apăsând tasta “3”, respectiv tasta “4”. Pentru a ieși din “secțiunea clienți”, se va apăsa tasta “0” și va fi redirecționat către meniul principal.

0. Iesire din sectiunea clienti

2.3.Aplicația care Realizează Gestiunea Hotelului

2.3.1.Interfața Programului

Codul din spatele programului se bazează pe meniuri și submeniuri. Această structură este necesară pentru o rulare continuă a programului, care poate fi întrerupt de către utilizator, apăsând tasta “0”. Submeniurile sunt apelate în meniul principal și funcționează ca secțiuni.

case '0':cout<<"Ati parasit zona de gestiune a hotelului nostru. Pentru mai multe detalii nu ezitati sa ne contactati! "<<endl;break;

case '1':meniu\_angajati();system ("pause");break;

case '2':meniu\_camere();system ("pause");break;

case '3':meniu\_recenzii();system ("pause");break;

Variabila “a” memorează o valoare introdusă de către utilizator și care trebuie să corespundă unui caz dintre cele listate în “case”. Pentru o structurare logică s‑au folosit subprograme diverse, care ulterior au fost apelate în submeniurile corespunzătoare.

a=getch();

} while (a<'0' || a>='4');

Pentru început, pe lângă declarerea bibliotecilor necesare funcționării programului, s‑au creat structuri utile pentru memorearea informațiilor de la angajați, camere și clienți. Variabilele care memorează numărul de angajați/camere/clienți din fișiere sunt declarate global (nu necesită declarare separată la crearea fiecărui subprogram necesar).

În continuare sunt realizate trei subprograme, care au rol estetic. Primul subprogram este folosit pentru aplicarea unei culori pentru o secțiune dorită. Subprogramul conține un parametru, prin intermediul căruia se specifică codul culorii care va fi introduse. Astfel, pentru aplicarea unei culori, s‑a apelat subprogramul în locul unde a fost nevoie și s‑a specificat codul culorii dorite.

culoare(3);

afisare\_clienti(index\_client);

culoare(7);

Culoarea cu codul 7 a fost introdusă, deoarece aceasta reprezintă culoarea “alb”. Prin această instrucțiune se revine la culoarea inițială.

Funcția “system (“cls”)” permite utilizatorului să vadă doar anumite părți din program la un moment dat și este necesară pentru o interfață simplă și ușor de utilizat de către orice persoană. Pentru a evita confuzii în legătură cu semnificația acestei funcții, s‑a denumit “curatare\_ecran()”.

Pentru delimitarea angajaților/camerelor/clienților s‑au folosit multiple caractere “=”, care au fost implementate într‑un subprogram. Subprogramul va fi apelat ulterior după afișările necesare, iar ca parametru se va introduce numărul de astfel de caractere, reprezentând lungimea linii separatoare.

În “int main()” se citesc datele existente în fișierele externe, pentru fiecare secțiune în parte. De asemenea, aici se apelează meniul principal.

2.3.2.Managementul Angajaților

Pentru fiecare dintre acțiunile ce se pot efectua la “Sectiunea angajați”, s‑au creat subprograme diferite. La subprogramul de afișare s‑au luat în calcul trei cazuri, și anume când există doi sau mai mulți angajați în listă, când există un singur angajat sau când nu există niciunul. Variabila declarată global “n” memorează numărul de angajați existenți în listă. Prin intermediul variabilei “i” se parcurge vectorul care memorează informațiile angajaților din listă și afișează un mesaj corespunzător numărului de angajați găsiți.

for (int i=1;i<=n;i++)

afisare\_angajat(i);

Pentru a evita unele repetări, s‑a creat un subprogram “afisare\_angajat()”, care, după cum sugerează denumirea, are rolul de a printa angajații citiți de fiecare dată când este apelat. Ca parametru se va introduce variabila “i”, iar astfel se vor parcurge și afișa toți angajații.

Subprogramul de căutare conține o structură repetitivă de tipul “do while”, prin care se efectuează căutări, până când utilizatorul alege să părăsească această secțiune. Căutarea se realizează după nume, astfel că la început s‑a declarat o variabilă care memorează numele introdus de utilizator. Folosind funcția “strcmp”, se compară numele angajaților cu cel introdus de utilizator, iar un contor va reține numărul de potriviri. Dacă acest contor este mai mare decât 1, înseamnă că în listă se află mai mulți angajați cu același nume, caz în care vor fi afișați, dacă contorul este 1, se va afișa un singur angajat, iar dacă contorul este 0 se va afișa un mesaj ce informează că nu există vreun angajat cu numele introdus. În cazul în care numele a fost introdus cu o literă mică la început, subprogramul folosește codul ASCII pentru a transforma această literă în majusculă.

În continuare, subprogramul corespunzător adăugării de angajați este format din două părți importante. În prima parte,se declară răspunsul ce va fi folositor ulterior, indexul angajatului pentru a ține evidența angajatului nou introdus, dar și un semafor. Programul solicită date noi referitoare la un angajat pentru a‑l putea adăuga. Confirmarea datelor introduse se realizează printr‑un răspuns. Semaforul depinde de răspunsul introdus și reține fie valoarea 0, fie valoarea 1.

A doua parte este reprezentată de către adăugarea propriu‑zisă a angajatului, în cazul în care utilizatorul a confirmat acțiunea. Variabila “index\_angajat” reține informațiile ultimului angajat, care tocmai a fost adăugat, după care urmează secvența prin care angajatul nou introdus va fi evidențiat prin culoare. Programul așteaptă apoi un răspuns la întrebarea afișată, iar dacă utilizatorul dorește să adauge alți angajați, tastează “da” și reia procesul de la capăt. Pentru ca utilizatorul să poată contiuna să adauge alți angajați, s‑a folosit o structură repetitivă de tip “do while”, care se oprește în momentul în care utilizatorul nu mai dorește să adauge alte persoane în listă.

while(strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0);

În secvența de cod de mai sus se poate observa condiția care permite programului să ruleze atât timp cât utilizatorul dorește să mai adauge alți angajați, fapt realizat cu ajutorul funcției “strcmp”, comparând două șiruri de caractere ( șirul introdus de utilizator și cuvântul “da”).

Subprogramul destinat ștergerii angajaților are o structură similară cu cea a adăugării. Variabilele folosite corespund, la fel ca și condițiile repetitive de forma “do while”. Diferența dintre cele două subprograme este reprezentată de acțiunea ce urmează să fie efectuată. În variabila “index\_angajat” se memorează informațiile angajatului selectat și care urmează să fie eliminat. După confirmarea că utilizatorul dorește să șteargă angajatul selectat din listă, se produce eliminarea acestuia.

for (int i=1;i<=n;i++)

{

if (index\_angajat==i)

v[index\_angajat]=v[index\_angajat+1];

}

n--;

}

Din această secvență de cod se poate observa că angajatul memorat pe poziția “index\_angajat” este eliminat, iar variabila “n” primește o valoare inferioară, pentru a memora numărul nou de angajați, care acum este de “n-1”, deci fără angajatul care tocmai a fost șters din listă.

Următorul subprogram este destinat efectuării modificărilor. Acest subprogram cuprinde patru părți importante și asemănătoare între ele ca și structură. Fiecare parte este legată de modificarea a uneia dintre criteriile ce definesc angajatul (nume/prenume/vârstă/salariul). La început, variabila “index\_angajat” memorează angajatul la care se vor efectua modificări, iar șirul de caractere “răspuns” va memora ce dorește utilizatorul să modifice. În funcție de răspunsul său, subprogramul se va concentra pe o singură parte. După specificarea criteriului ce va fi modificat, programul va aștepta o valoare nouă pentru a putea realiza modificarea. Spre exemplu, dacă numele este criteriul ce necesită modificare, programul va aștepta introducerea unui nou șir de caractere, reprezentând numele, pentru a‑l înlocui. Numele nou introdus este memorat în variabila “nume\_introdus” și va fi comparat cu numele actual al angajatului, cu ajutorul funcției “strcmp”. Dacă cele două nume corespund, atunci se va afișa un mesaj de atenționare și procesul se repetă, până când se va introduce un nume nou, fapt realizat cu ajutorul unei structuri “do while”. După aceasta, se poate trece la confirmare. Pe baza răspunsului înregistrat, se va trece la următoarea etapă. Dacă se confirmă modificarea conturată, atunci numele va fi modificat, iar angajatul, conținând noul nume, va fi afișat. Atât în caz că se confirmă modificarea sau nu, utilizatorul are opțiunea de a efectua alte modificări ulterior. Acest proces se repetă până când nu se mai dorește modificarea vreunui criteriu. Pe același principiu funcționează și partea de modificare a prenumelui, a vârstei sau a salariului.

Pentru subprogramul de ordonare se va specifica criteriul după care se vor ordona angajații. Pentru o structurare mai logică s‑au realizat subprograme diferite, care vor fi apelate în subprogramul principal de ordonare. Fiecare dintre aceste subprograme conține ordonarea atât crescătoare, cât și descrescătoare a criteriului selectat. Pentru determinarea dacă se va realiza o ordonare crescătoare sau descrescătoare, s‑a folosit o variabilă “crescator”, care este de fapt un semafor ce reține valoarea 1 ptr o ordonare crescătoare și valoarea 0 pentru o ordonare descrescătoare.

if(strcmp(raspuns2,"crescator")==0 || (raspuns2,"Crescator")==0 || (raspuns2,"CRESCATOR")==0)

crescator=1;

else

crescator=0;

La început au fost declarate două variabile de tip char, “raspuns” și “raspuns2”. Prima variabila reține criteriul după care se efectuează ordonarea, iar a doua variabila reprezintă tipul de ordonare ce va fi executat. Subprogramul specific ordonării după un anumit criteriu va fi apelat apoi în subprogramul de ordonare, astfel efectuându‑se ordonarea după preferință. Utilizatorul poate opta pentru o nouă ordonare ulterior, acest proces fiind repetat până când nu se va mai răspunde afirmativ la întrebarea care verifică dacă utilizatorul dorește să efectueze alte ordonări. (până când iasă din structura repetitivă “do while”).

2.3.3.Gestionarea Camerelor

Pentru fiecare acțiune ce se poate executa în “Sectiunea camere”, s‑au folosit subprograme diferite. La subprogramul de afișare a camerelor disponibile, s‑a creat un alt subprogram numit “afisare\_camere”, care va fi apelat oriunde este nevoie de printarea camerelor disponibile, apelând acest subprogram și folosind parametrul “i” ca număr de ordine a camerei actuale. Subprogramul de afișare ia în calcul trei posibilități, și anume cazul în care există două sau mai multe camere disponibile, o cameră disponibilă sau nicio cameră disponibilă. După aceea, subprogramul afișează camerele ce se află în listă. Variabila declarată global “m” reprezintă numărul camerelor disponibile care se află în listă.

Subprogramul de adăugare urmează un fir similar cu cel al adăugării angajaților. La început, programul așteaptă introducerea unui număr, care va corespunde cu numărul camerei ce va fi trecută pe lista celor disponibile. Variabila “c[m+1].numar” memorează numărul nou introdus. Dacă numărul introdus corespunde cu numărul vreunei camere deja existente în listă, utilizatorul va trebui să introducă un nou număr, până când acesta nu mai corespunde cu numărul vreunei camere din listă. Variabila “index\_camera” memorează informațiile camerei introduse. Semaforul denumit “ok” verifică atât dacă informațiile introduse sunt noi și nu se află deja în listă, cât și dacă utilizatorul confirmă adăugarea camerei selectate. În aceste cazuri, variabila “ok” va avea valoarea 0. La adăugarea camerei, valoarea variabilei “m” se mărește, pentru a face posibilă încadrarea noii camere în listă. Ulterior se verifică prin funcția “strcmp”, dacă se confirmă adăugarea camerei. În caz afirmativ, se va parcurge vectorul care conține informații despre camere, cu ajutorul structurii “for” și se vor afișa, pe rând, camerele din listă, iar camera nou introdusă va fi afișată la final.

m++;

curatare\_ecran();

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

cout<<"Aceasta este lista noua cu cele "<<m<<" camere, iar camera nou introdusa este evidentiata prin culoare: "<<endl<<endl;

for (int i=1;i<m;i++)

afisare\_camere(i);

Procesul se poate repeta atât timp cât utilizatorul dorește să adauge alte camere (până când nu va mai introduce răspunsul “da”).

Subprogramul de rezervare funcționează pe același principiu cu cel de ștergere al angajaților. Aici se elimină camera selectată prin variabila “numar\_camera” din lista camerelor disponibile. Dacă numărul introdus nu se găsește în lista camerelor, utilizatorul va fi nevoit să introducă un alt număr, până când numărul va corespunde cu numărul unei camere din listă. Variabila “index\_camera” este la început 0, nefiind cunoscut indexul camerei care urmează a fi rezervată. Când numărul introdus va corespunde cu numărul unei camere din listă, atunci variabila “index\_camera” va prelua valoarea indexului actual al camerei selectate și va reține informațiile acesteia. După selectarea camerei care urmează să fie eliminată din listă, va fi nevoie de o confirmare pentru a putea realiza rezervarea camerei. Dacă se confirmă rezervarea, camera va fi eliminată din listă și se va afișa lista actualizată, fără camera tocmai rezervată.

for (int i=1;i<=m;i++)

{

if (numar\_camera==c[i].numar)

c[index\_camera]=c[index\_camera+1];

for (i=index\_camera;i<=m;i++)

c[i]=c[i+1];

}

m--;

Astfel, valoarea variabilei “m” se reduce la “m-1”, iar valoarea elementului aflat pe poziția “c[index\_angajat]” va prelua valoarea elementului din față, eliminând astfel camera dorită din lista celor disponibile, iar toate celelalte camere vor fi mutate cu o poziție mai în față, pentru a nu lăsa loc liber. Procesul se poate repeta ori de câte ori este nevoie, până când la întrebarea adresată pentru utilizator nu se va mai răspunde cu “da”. Astfel nu se mai execută structura repetitivă “do while”.

Subprogramul de ordonare este format din mai multe subprograme ce vor fi apelate în subprogramul principal de ordonare. Subprogramele care vor fi apelate în subprogramul principal de ordonare sunt similare, ambele efectuând atât ordonare crescătoare, cât și ordonare descrescătoare. Pentru realizarea acestor subprograme se folosește o variabilă numită “crescator”, care verifica tipul de ordonare implementat. După aceea, cu ajutorul a două variabile notate cu “i” și “j”, se parcurg camerele din listă și se verifică criteriul pentru care a fost realizat subprogramul (număr/cost), cu ajutorul funcției “strcmp”. În subprogramul principal de ordonare va fi nevoie de introducerea criteriului după care se ordonează camerele și tipul de ordonare dorit. După ce s‑a efectuat ordonarea, utilizatorul are opțiunea de a realiza alte ordonări. Procesul se repetă până când nu se va mai introduce răspunsul “da” la întrebarea adresată în legătură cu efectarea unei noi modificări. Astfel nu se mai execută structura repetitivă “do while”.

2.3.4.Programul Destinat Clienților

Fiecărei operații executabile în “Sectiunea clienți” îi corespunde un subprogram. Pentru afișarea clienților s‑a creat inițial un subprogram, numit “afisare\_recenzii\_clienti ( ) ”, care poate fi apelat ulterior în alte subprograme și care este util la afișarea recenziilor oferite de către clienți. Se diferențiază aici trei cazuri. Se va afișa un mesaj corespunzător, împreună cu lista recenziilor oferite de către clienți, dacă există cel puțin doi clienți în listă. Dacă există doar un client în listă, acesta va fi afișat, iar în cazul în care nu există niciunul, va fi afișat un mesaj corespunzător acestui caz. Cu ajutorul structurii de tip “for” se parcurge vectorul ce memorează informații despre clienți, prin intermediul variabilei “i”. Variabila declarată global “p” memorează numărul de recenzii oferite de către clienți.

Subprogramul de adăugare conține la început partea de introducere a informațiilor necesare pentru a putea înregistra recenzia clientului în listă. Dacă numele și prenumele clientului introdus, exprimate prin variabilele “d[p+1].nume2” și “d[p+1].prenume2” sunt identice cu cele ale unui client deja existent în listă, se va afișa un mesaj de atenționare asupra acestui fapt. După ce datele și informațiile necesare înregistrării recenziei au fost introduse, va fi nevoie de o confirmare pentru adăugare. Dacă acțiunea se confirmă, datele clientului, împreună cu recenzia și, eventual, sugestia și reclamația sa vor fi adăugate în listă. Pe ecran se va afișa lista actualizată.

p++;

curatare\_ecran();

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

cout<<"Aceasta este lista noua cu cei "<<p<<" clienti, iar clientul nou introdus este evidentiat prin culoare: "<<endl<<endl;

for (int i=1;i<p;i++)

afisare\_clienti(i);

Procesul se poate repeta, până când nu se va mai introduce răspunsul “da” la întrebarea în legătură cu adăugarea unei alte recenzii. Astfel nu se mai execută structura repetitivă “do while”.

Subprogramul de afișare a sugestiilor/reclamațiior citește doar sugestiile/reclamațiile clienților și le afișează pe ecran, împreună cu numele și prenumele clientului respectiv. Numărul de caractere “=” ce separă sugestiile una de alta (lungimea separatorului) este dată de lungimea maximă a unei sugestii/reclamații

3.Concluzii

În primul rând, acest proiect a contribuit la dezvoltarea abilității de a structura și gestiona o aplicație. Această abilitate reprezintă o calitate importantă pentru o persoană în societatea actuală. De asemenea, scrierea propriu‑zisă a codului a necesitat multă atenție și seriozitate, fapt ce poate avea ca efect îmbunătățirea atât a spiritului de observație, cât și a capacității de a rămâne calm și de a căuta o soluție la eventualele probleme care se pot ivi pe parcursul codării.

În al doilea rând, nevoia de a obține informații în timp real, necesită o monitorizare atentă prin utilizarea tot mai frecventă a sistemelor automate de memorare a datelor. Acesta este și scopul proiectului, de a realiza un program care stochează și actualizează o listă cu elemente bine definite. Realizarea proiectului a condus la observarea faptului că un asemenea program are multe puncte comune cu majoritatea device‑urilor pe care le folosim zilnic, cum ar fi telefonul mobil. Având în vedere că pe viitor, aceste programe se vor folosi pe o scară tot mai largă în numeroase domenii de activitate, utilizarea acestora devine nu doar o opțiune, ci chiar o necesitate.

În concluzie, acest proiect prezintă beneficii multiple atât pentru entități din industria hotelieră, cât și pentru utilizatori independenți.

4.Bibliografie

*cplusplus.com.* (2022, Mai 5). Retrieved from ASCII Codes: https://www.cplusplus.com/doc/ascii/

*cplusplus.com.* (2022, Aprilie 20). Retrieved from <cstring> string.h: https://www.cplusplus.com/reference/cstring/

*Microsoft Word SUPORT CURS receptioner.doc.* (2022, Mai 12). Retrieved from MANAGEMENTUL HOTELIER: https://geoturism.files.wordpress.com/2019/05/suport-curs-receptioner-atpro.pdf

5.Anexă (Codul Sursă)

#include<iostream>

#include<fstream>

#include<iomanip>

#include<conio.h>

#include<string.h>

#include<windows.h>

using namespace std;

struct angajat{

char nume[50];

char prenume[50];

int varsta;

float salariu;

};

struct camera{

int numar;

char tip[10];

float cost;

};

struct client{

char nume2[50];

char prenume2[50];

float recenzie;

char sugestie[200];

char reclamatie[200];

};

int n;

int m;

int p;

angajat v[100];

camera c[100];

client d[100];

void culoare(int culoare)

{

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE), culoare);

}

void curatare\_ecran()

{

system("cls");

}

void separare(int m)

{

for (int j=1;j<=m;j++)

cout<<"=";

cout<<endl;

cout<<endl;

}

void afisare\_angajat(int i)

{

cout<<"Angajatul: "<<i<<endl;

cout<<"Date personale: "<<endl;

cout<<"Nume: "<<v[i].nume<<endl;

cout<<"Prenume: "<<v[i].prenume<<endl;

if (v[i].varsta>20)

cout<<"Varsta: "<<v[i].varsta<<" de ani"<<endl;

else

cout<<"Varsta: "<<v[i].varsta<<" ani"<<endl;

cout<<"Salariul: "<<v[i].salariu<<" lei"<<endl;

separare(30);

}

void afisare ()

{

if (n>1)

{

cout<<"Aceasta este lista cu cei "<<n<<" angajati"<<endl<<endl;

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

else if (n==1)

{

cout<<"Acesta este angajatul din baza de date"<<endl<<endl;

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

if (n==0)

cout<<"Nu exista angajati in baza de date! Adaugati angajati pentru a-i putea afisa!"<<endl<<endl;

}

void cautare()

{

int sem=0;

char s[50];

int ct=0;

int index\_angajat;

cout<<"Aceasta este lista cu cei "<<n<<" angajati"<<endl<<endl;

for (int i=1;i<=n;i++)

{

afisare\_angajat(i);

}

cout<<"Introduceti numele angajatului: "<<endl;

cin>>s;

for (int i=0;i<strlen(s);i++)

if (s[i]==s[0])

if(s[i]>='a' && s[i]<='z')

s[0]=s[0]-32;

curatare\_ecran();

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

if (strcmp(v[i].nume,s)==0)

ct++;

}

if (ct==1)

{

cout<<"Angajatul cautat se afla in baza de date! Mai jos se gasesc informatiile sale: "<<endl<<endl;

}

if (ct >= 2)

{

cout<<"Angajatii cautati se afla in baza de date! Mai jos se gasesc informatiile lor: "<<endl<<endl;

}

for(int i=1;i<=n;i++)

{

if (strcmp(v[i].nume,s)==0)

{

afisare\_angajat(i);

}

}

for (int i=0;i<strlen(s);i++)

if (s[i]==s[0])

if(s[i]>='a' && s[i]<='z')

s[0]=s[0]-32;

if (ct==0)

cout<<"Angajatul cu numele "<<s<<" nu se afla in baza de date"<<endl;

}

// -------------------------------------------------------------------------------------------------------------------------------------------------

void adaugare()

{

char raspuns[10];

int ok=0;

int index\_angajat;

do

{

cout<<"Introduceti informatiile noului angajat: "<<endl<<endl;

cout<<"Nume: ";

cin>>v[n+1].nume;

cout<<endl;

cout<<"Prenume: ";

cin>>v[n+1].prenume;

cout<<endl;

for (int i=1;i<=n;i++)

for (int i=n;i>=1;i--)

if (strcmp(v[i].nume,v[n+1].nume)==0)

if (strcmp(v[i].prenume,v[n+1].prenume)==0)

{

do

{

ok=1;

cout<<"Angajatul se afla deja in baza de date!"<<endl;

cout<<"Asigurati-va ca angajatul nu se afla deja in baza de date si introduceti informatiile acestuia!"<<endl<<endl;;

cout<<"Nume: ";

cin>>v[n+1].nume;

cout<<endl;

cout<<"Prenume: ";

cin>>v[n+1].prenume;

cout<<endl;

}while (strcmp(v[i].nume,v[n+1].nume)==0 && strcmp(v[i].prenume,v[n+1].prenume)==0);

ok=0;

}

if (ok==0)

{

cout<<"Varsta: ";

cin>>v[n+1].varsta;

curatare\_ecran();

cout<<"Introduceti informatiile noului angajat: "<<endl<<endl;

cout<<"Nume: "<<v[n+1].nume<<endl<<endl;

cout<<"Prenume: "<<v[n+1].prenume<<endl<<endl;

if (v[n+1].varsta>19)

cout<<"Varsta: "<<v[n+1].varsta<<" de ani"<<endl;

else

cout<<"Varsta: "<<v[n+1].varsta<<" ani"<<endl;

if (v[n+1].varsta>69 || v[n+1].varsta<18)

do

{

cout<<"Ati introdus o varsta care nu este eligibila pentru un angajat!"<<endl;

cout<<"Introduceti varsta noului angajat, care sa fie cuprinsa intre 18-69"<<endl;

cin>>v[n+1].varsta;

}while (v[n+1].varsta>69 || v[n+1].varsta<18);

cout<<endl;

cout<<"Salariul: ";

cin>>v[n+1].salariu;

curatare\_ecran();

cout<<"Introduceti informatiile noului angajat: "<<endl<<endl;

cout<<"Nume: "<<v[n+1].nume<<endl<<endl;

cout<<"Prenume: "<<v[n+1].prenume<<endl<<endl;

if (v[n+1].varsta>19)

cout<<"Varsta: "<<v[n+1].varsta<<" de ani"<<endl<<endl;

else

cout<<"Varsta: "<<v[n+1].varsta<<" ani"<<endl<<endl;

cout<<"Salariul: "<<v[n+1].salariu<<" lei"<<endl;

cout<<endl;

cout<<"Confirmati ca doriti adaugarea angajatului introdus? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

ok=0;

if (strcmp(raspuns,"nu")==0 || strcmp(raspuns,"Nu")==0 || strcmp(raspuns,"NU")==0)

ok=1;

if (ok==1)

{

cout<<"Ati renuntat la adaugarea noului angajat!"<<endl;

break;

}

if (ok==0)

{

n++;

curatare\_ecran();

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

cout<<"Aceasta este lista noua cu cei "<<n<<" angajati, iar angajatul nou introdus este evidentiat prin culoare: "<<endl<<endl;

for (int i=1;i<n;i++)

afisare\_angajat(i);

for (int i=1;i<=n;i++)

if (i==n)

{

index\_angajat=i;

culoare(3);

afisare\_angajat(index\_angajat);

culoare(7);

}

cout<<"Doriti sa adaugati alti angajati? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

curatare\_ecran();

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

ok=0;

}

if (strcmp(raspuns,"nu")==0 || strcmp(raspuns,"Nu")==0 || strcmp(raspuns,"NU")==0)

{

cout<<"Aceasta este lista noua cu cei "<<n<<" angajati, iar angajatul nou introdus este evidentiat prin culoare: "<<endl<<endl;

for (int i=1;i<n;i++)

afisare\_angajat(i);

for (int i=1;i<=n;i++)

if (i==n)

{

index\_angajat=i;

culoare(3);

afisare\_angajat(index\_angajat);

culoare(7);

}

}

}

}

}while(strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0);

}

//------------------------------------------------------------------------------------------------------------------------------------------------------

void stergere ()

{

bool ok=1;

char raspuns[10];

int index\_angajat=0;

do

{

curatare\_ecran();

if (n==1)

cout<<"Acesta este angajatul din baza de date:"<<endl<<endl;

else if (n>1)

cout<<"Acestia sunt cei "<<n<<" angajati din baza de date"<<endl<<endl;

else

{

cout<<"Nu exista angajati in baza de date!"<<endl<<endl;

ok=0;

}

if(ok==1)

{

for (int i=1;i<=n;i++)

afisare\_angajat(i);

cout<<"Alegeti indexul angajatului care urmeaza sa fie eliminat: "<<endl;

cin>>index\_angajat;

do

{

if (index\_angajat>n)

{

cout<<"Angajatul nu se afla in baza de date!"<<endl<<endl;

cout<<"Alegeti indexul angajatului care urmeaza sa fie eliminat: "<<endl;

cin>>index\_angajat;

}

}while (index\_angajat>n);

cout<<endl;

curatare\_ecran();

afisare\_angajat(index\_angajat);

cout<<"Confirmati ca doriti stergerea angajatului selectat din baza de date? (da/nu)"<<endl;

cin>>raspuns;

if (strcmp(raspuns,"da")==0)

{

for (int i=1;i<=n;i++)

{

if (index\_angajat==i)

v[index\_angajat]=v[index\_angajat+1];

}

n--;

}

curatare\_ecran();

cout<<"Doriti sa eliminati alti angajati din lista? (da/nu)"<<endl;

cin>>raspuns;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

if (strcmp(raspuns,"nu")==0 || strcmp(raspuns,"Nu")==0 || strcmp(raspuns,"NU")==0)

{

curatare\_ecran();

cout<<"Acestia sunt angajatii ramasi in lista: "<<endl<<endl;

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

}

}while(strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0);

}

void modificare()

{

int ok=1;

char raspuns[10];

int index\_angajat=0;

char nume\_introdus[50];

int var\_introdus;

float salariu\_introdus;

do

{

if (n==1)

cout<<"Acesta este angajatul din baza de date:"<<endl<<endl;

else if (n>1)

cout<<"Acestia sunt cei "<<n<<" angajati din baza de date"<<endl<<endl;

else

{

cout<<"Nu exista angajati in baza de date!"<<endl<<endl;

ok=0;

}

if (ok==1)

for (int i=1;i<=n;i++)

afisare\_angajat(i);

cout<<"Alegeti indexul angajatului care urmeaza sa fie modificat: "<<endl;

cin>>index\_angajat;

for (int i=1;i<=n;i++)

if (index\_angajat==i)

{

curatare\_ecran();

afisare\_angajat(index\_angajat);

cout<<"Ce doriti sa modificati la acest angajat? (nume/prenume/varsta/salariu)"<<endl;

}

cin>>raspuns;

if (strcmp(raspuns,"nume")==0 || strcmp(raspuns,"Nume")==0 || strcmp(raspuns,"NUME")==0)

{

int sem=0;

cout<<endl;

cout<<"Introduceti numele nou: "<<endl;

cin>>nume\_introdus;

cout<<endl;

for (int i=1;i<=n;i++)

if (strcmp(nume\_introdus,v[i].nume)==0)

{

do

{

sem=1;

cout<<"Numele introdus corespunde cu numele angajatului!"<<endl;

cout<<"Introduceti numele nou: "<<endl;

cin>>nume\_introdus;

cout<<endl;

}while (strcmp(nume\_introdus,v[i].nume)==0);

sem=0;

}

if (sem==0)

{

cout<<"Confirmati ca doriti modificarea numelui a angajatului selectat? (da/nu)"<<endl;

cin>>raspuns;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

cout<<"Numele s-a modificat!"<<endl<<endl;

strcpy(v[index\_angajat].nume,nume\_introdus);

afisare\_angajat(index\_angajat);

cout<<"Doriti sa vedeti lista actualizata a angajatilor? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

cout<<"Acestia sunt toti cei "<<n<<" angajati, iar angajatul la care s-au efectuat modificari este evidentiat prin culoare"<<endl<<endl;

for (int i=1;i<=n;i++)

{

if (i==index\_angajat)

{

culoare(3);

afisare\_angajat(i);

culoare(7);

}

else

{

afisare\_angajat(i);

cout<<endl;

}

}

}

cout<<"Doriti sa efectuati alte modificari? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

curatare\_ecran();

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

}

else //(strcmp(raspuns,"nu")==0 || strcmp(raspuns,"Nu")==0 || strcmp(raspuns,"NU")==0)

{

cout<<"Acestia sunt toti angajatii dupa efectuarea tutror modificarilor necesare: "<<endl<<endl;

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

}

else

{

curatare\_ecran();

cout<<"Ati renuntat la modificarea numelui!"<<endl<<endl;

cout<<"Doriti sa efectuati alte modificari? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

curatare\_ecran();

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

}

if (strcmp(raspuns,"nu")==0 || strcmp(raspuns,"Nu")==0 || strcmp(raspuns,"NU")==0)

{

cout<<"Acestia sunt toti angajatii dupa efectuarea tuturor modificarilor necesare: "<<endl<<endl;

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

}

}

}

if (strcmp(raspuns,"prenume")==0 || strcmp(raspuns,"Prenume")==0 || strcmp(raspuns,"PRENUME")==0)

{

int sem=0;

cout<<endl;

cout<<"Introduceti prenumele nou: "<<endl;

cin>>nume\_introdus;

cout<<endl;

for (int i=1;i<=n;i++)

if (strcmp(nume\_introdus,v[i].prenume)==0)

{

do

{

sem=1;

cout<<"Prenumele introdus corespunde cu prenumele angajatului!"<<endl;

cout<<"Introduceti prenumele nou: "<<endl;

cin>>nume\_introdus;

cout<<endl;

}while (strcmp(nume\_introdus,v[i].prenume)==0);

sem=0;

}

if (sem==0)

{

cout<<"Confirmati ca doriti modificarea prenumelui a angajatului selectat? (da/nu)"<<endl;

cin>>raspuns;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

cout<<"Prenumele s-a modificat!"<<endl<<endl;

strcpy(v[index\_angajat].prenume,nume\_introdus);

afisare\_angajat(index\_angajat);

cout<<"Doriti sa vedeti lista actualizata a angajatilor? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

cout<<"Acestia sunt toti cei "<<n<<" angajati, iar angajatul la care s-au efectuat modificari este evidentiat prin culoare"<<endl<<endl;

for (int i=1;i<=n;i++)

{

if (i==index\_angajat)

{

culoare(3);

afisare\_angajat(i);

culoare(7);

}

else

{

afisare\_angajat(i);

cout<<endl;

}

}

}

cout<<"Doriti sa efectuati alte modificari? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

curatare\_ecran();

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

}

if (strcmp(raspuns,"nu")==0 || strcmp(raspuns,"Nu")==0 || strcmp(raspuns,"NU")==0)

{

cout<<"Acestia sunt toti angajatii dupa efectuarea tutror modificarilor necesare: "<<endl<<endl;

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

}

else

{

curatare\_ecran();

cout<<"Ati renuntat la modificarea prenumelui!"<<endl<<endl;

cout<<"Doriti sa efectuati alte modificari? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

curatare\_ecran();

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

}

if (strcmp(raspuns,"nu")==0 || strcmp(raspuns,"Nu")==0 || strcmp(raspuns,"NU")==0)

{

cout<<"Acestia sunt toti angajatii dupa efectuarea tuturor modificarilor necesare: "<<endl<<endl;

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

}

}

}

if (strcmp(raspuns,"varsta")==0 || strcmp(raspuns,"Varsta")==0 || strcmp(raspuns,"VARSTA")==0)

{

int sem=0;

cout<<endl;

cout<<"Introduceti varsta noua: "<<endl;

cin>>var\_introdus;

cout<<endl;

for (int i=1;i<=n;i++)

if (var\_introdus==v[i].varsta)

{

do

{

sem=1;

cout<<"Varsta introdusa corespunde cu varsta angajatului!"<<endl;

cout<<"Introduceti varsta nou: "<<endl;

cin>>var\_introdus;

cout<<endl;

}while (var\_introdus==v[i].varsta);

sem=0;

}

if (sem==0)

{

cout<<"Confirmati ca doriti modificarea varstei a angajatului selectat? (da/nu)"<<endl;

cin>>raspuns;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

cout<<"Varsta s-a modificat!"<<endl<<endl;

v[index\_angajat].varsta=var\_introdus;

afisare\_angajat(index\_angajat);

cout<<"Doriti sa vedeti lista actualizata a angajatilor? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

cout<<"Acestia sunt toti cei "<<n<<" angajati, iar angajatul la care s-au efectuat modificari este evidentiat prin culoare"<<endl<<endl;

for (int i=1;i<=n;i++)

{

if (i==index\_angajat)

{

culoare(3);

afisare\_angajat(i);

culoare(7);

}

else

{

afisare\_angajat(i);

cout<<endl;

}

}

}

cout<<"Doriti sa efectuati alte modificari? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

curatare\_ecran();

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

}

if (strcmp(raspuns,"nu")==0 || strcmp(raspuns,"Nu")==0 || strcmp(raspuns,"NU")==0)

{

cout<<"Acestia sunt toti angajatii dupa efectuarea tutror modificarilor necesare: "<<endl<<endl;

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

}

else

{

curatare\_ecran();

cout<<"Ati renuntat la modificarea varstei!"<<endl<<endl;

cout<<"Doriti sa efectuati alte modificari? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

curatare\_ecran();

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

}

if (strcmp(raspuns,"nu")==0 || strcmp(raspuns,"Nu")==0 || strcmp(raspuns,"NU")==0)

{

cout<<"Acestia sunt toti angajatii dupa efectuarea tuturor modificarilor necesare: "<<endl<<endl;

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

}

}

}

if (strcmp(raspuns,"salariu")==0 || strcmp(raspuns,"Salariu")==0 || strcmp(raspuns,"SALARIU")==0)

{

int sem=0;

cout<<endl;

cout<<"Introduceti salariul nou: "<<endl;

cin>>salariu\_introdus;

cout<<endl;

for (int i=1;i<=n;i++)

if (salariu\_introdus==v[i].salariu)

{

do

{

sem=1;

cout<<"Salariul introdus corespunde cu salariul angajatului!"<<endl;

cout<<"Introduceti salariul nou: "<<endl;

cin>>salariu\_introdus;

cout<<endl;

}while (salariu\_introdus==v[i].salariu);

sem=0;

}

if (sem==0)

{

cout<<"Confirmati ca doriti modificarea salariului a angajatului selectat? (da/nu)"<<endl;

cin>>raspuns;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

cout<<"Salariul s-a modificat!"<<endl<<endl;

v[index\_angajat].salariu=salariu\_introdus;

afisare\_angajat(index\_angajat);

cout<<"Doriti sa vedeti lista actualizata a angajatilor? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

cout<<"Acestia sunt toti cei "<<n<<" angajati, iar angajatul la care s-au efectuat modificari este evidentiat prin culoare"<<endl<<endl;

for (int i=1;i<=n;i++)

{

if (i==index\_angajat)

{

culoare(3);

afisare\_angajat(i);

culoare(7);

}

else

{

afisare\_angajat(i);

cout<<endl;

}

}

}

cout<<"Doriti sa efectuati alte modificari? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

curatare\_ecran();

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

}

if (strcmp(raspuns,"nu")==0 || strcmp(raspuns,"Nu")==0 || strcmp(raspuns,"NU")==0)

{

cout<<"Acestia sunt toti angajatii dupa efectuarea tutror modificarilor necesare: "<<endl<<endl;

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

}

else

{

curatare\_ecran();

cout<<"Ati renuntat la modificarea salariului!"<<endl<<endl;

cout<<"Doriti sa efectuati alte modificari? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

curatare\_ecran();

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

}

if (strcmp(raspuns,"nu")==0 || strcmp(raspuns,"Nu")==0 || strcmp(raspuns,"NU")==0)

{

cout<<"Acestia sunt toti angajatii dupa efectuarea tuturor modificarilor necesare: "<<endl<<endl;

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

}

}

}

}while (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0);

}

void ord\_alfabetica\_nume(int crescator)

{

if (crescator==1)

{

for (int i=1;i<=n;i++)

for (int j=1;j<=n;j++)

if (strcmp(v[i].nume,v[j].nume)<=0)

{

angajat aux=v[i];

v[i]=v[j];

v[j]=aux;

}

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

else

{

for (int i=1;i<=n;i++)

for (int j=1;j<=n;j++)

if (strcmp(v[i].nume,v[j].nume)>=0)

{

angajat aux=v[i];

v[i]=v[j];

v[j]=aux;

}

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

}

void ord\_alfabetica\_prenume(int crescator)

{

if (crescator==1)

{

for (int i=1;i<=n;i++)

for (int j=1;j<=n;j++)

if (strcmp(v[i].prenume,v[j].prenume)<=0)

{

angajat aux=v[i];

v[i]=v[j];

v[j]=aux;

}

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

else

{

for (int i=1;i<=n;i++)

for (int j=1;j<=n;j++)

if (strcmp(v[i].prenume,v[j].prenume)>=0)

{

angajat aux=v[i];

v[i]=v[j];

v[j]=aux;

}

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

}

void ord\_varsta(int crescator)

{

if (crescator==1)

{

for (int i=1;i<=n;i++)

for (int j=1;j<=n;j++)

if (v[i].varsta<=v[j].varsta)

{

angajat aux=v[i];

v[i]=v[j];

v[j]=aux;

}

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

else

{

for (int i=1;i<=n;i++)

for (int j=1;j<=n;j++)

if (v[i].varsta>=v[j].varsta)

{

angajat aux=v[i];

v[i]=v[j];

v[j]=aux;

}

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

}

void ord\_salariu(int crescator)

{

if (crescator==1)

{

for (int i=1;i<=n;i++)

for (int j=1;j<=n;j++)

if (v[i].salariu<=v[j].salariu)

{

angajat aux=v[i];

v[i]=v[j];

v[j]=aux;

}

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

else

{

for (int i=1;i<=n;i++)

for (int j=1;j<=n;j++)

if (v[i].salariu>=v[j].salariu)

{

angajat aux=v[i];

v[i]=v[j];

v[j]=aux;

}

for (int i=1;i<=n;i++)

afisare\_angajat(i);

}

}

void ordonare()

{

int crescator;

int ok=1;

char raspuns[15];

char raspuns2[15];

do

{

if (n==1)

cout<<"Acesta este angajatul din baza de date:"<<endl<<endl;

else if (n>1)

cout<<"Acestia sunt cei "<<n<<" angajati din baza de date"<<endl<<endl;

else

{

cout<<"Nu exista angajati in baza de date!"<<endl<<endl;

ok=0;

}

if (ok==1)

for (int i=1;i<=n;i++)

afisare\_angajat(i);

cout<<"Alegeti dupa ce criteriu doriti sa ordonati angajatii (nume/prenume/varsta/salariu) "<<endl;

cin>>raspuns;

cout<<endl;

cout<<"Doriti sa ordonati crescator sau descrescator? (crescator/descrescator)"<<endl;

cin>>raspuns2;

if(strcmp(raspuns2,"crescator")==0 || (raspuns2,"Crescator")==0 || (raspuns2,"CRESCATOR")==0)

crescator=1;

else

crescator=0;

curatare\_ecran();

if (strcmp(raspuns,"nume")==0 || strcmp(raspuns,"Nume")==0 || strcmp(raspuns,"NUME")==0)

{

cout<<"Aceasta este lista angajatilor ordonati dupa nume: "<<endl<<endl;

ord\_alfabetica\_nume(crescator);

cout<<endl;

cout<<"Doriti sa efectuati alta ordonare? (da/nu)"<<endl;

cin>>raspuns;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

for (int i=1;i<=n;i++)

afisare\_angajat(i);

curatare\_ecran();

}

else

{

curatare\_ecran();

ok=0;

}

}

if (strcmp(raspuns,"prenume")==0 || strcmp(raspuns,"Prenume")==0 || strcmp(raspuns,"PRENUME")==0)

{

cout<<"Aceasta este lista angajatilor ordonati dupa prenume: "<<endl<<endl;

ord\_alfabetica\_prenume(crescator);

cout<<endl;

cout<<"Doriti sa efectuati alta ordonare? (da/nu)"<<endl;

cin>>raspuns;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

for (int i=1;i<=n;i++)

afisare\_angajat(i);

curatare\_ecran();

}

else

{

curatare\_ecran();

ok=0;

}

}

if (strcmp(raspuns,"varsta")==0 || strcmp(raspuns,"Varsta")==0 || strcmp(raspuns,"VARSTA")==0)

{

cout<<"Aceasta este lista angajatilor ordonati dupa varsta: "<<endl<<endl;

ord\_varsta(crescator);

cout<<endl;

cout<<"Doriti sa efectuati alta ordonare? (da/nu)"<<endl;

cin>>raspuns;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

for (int i=1;i<=n;i++)

afisare\_angajat(i);

curatare\_ecran();

}

else

{

curatare\_ecran();

ok=0;

}

}

if (strcmp(raspuns,"salariu")==0 || strcmp(raspuns,"Salariu")==0 || strcmp(raspuns,"SALARIU")==0)

{

cout<<"Aceasta este lista angajatilor ordonati dupa salariu: "<<endl<<endl;

ord\_salariu(crescator);

cout<<endl;

cout<<"Doriti sa efectuati alta ordonare? (da/nu)"<<endl;

cin>>raspuns;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

for (int i=1;i<=n;i++)

afisare\_angajat(i);

curatare\_ecran();

}

else

{

curatare\_ecran();

ok=0;

}

}

}while (ok==1);

}

void afisare\_camere(int i)

{

cout<<"Camera nr. "<<c[i].numar<<endl;

cout<<"Informatii generale: "<<endl<<"Tip: "<<c[i].tip<<endl<<"Cost: "<<c[i].cost<<" de lei/noapte"<<endl;

separare(30);

}

void camere()

{

if (m>1)

{

cout<<"Aceasta este lista cu cele "<<m<<" camere"<<endl<<endl;

for (int i=1;i<=m;i++)

afisare\_camere(i);

}

else if (m==1)

{

cout<<"Aceasta este camera din baza de date"<<endl<<endl;

for (int i=1;i<=m;i++)

afisare\_camere(i);

}

if (m==0)

cout<<"Nu exista camere disponibile in baza de date! Adaugati camere pentru a le putea afisa!"<<endl<<endl;

}

void adaugare\_camere()

{

int ok=0;

char raspuns [20];

int index\_camera=0;

do

{

cout<<"Introduceti informatiile camerei care a fost eliberata si este acum disponibila clientilor: "<<endl<<endl;

cout<<"Camera nr. ";

cin>>c[m+1].numar;

for (int i=1;i<=m;i++)

for (int i=m;i>=1;i--)

if (c[m+1].numar==c[i].numar)

{

do

{

ok=1;

cout<<"Camera introdusa este deja disponibila!"<<endl;

cout<<"Introduceti informatiile camerei care a fost eliberata si este acum disponibila clientilor!"<<endl<<endl;

cout<<"Camera nr.: ";

cin>>c[m+1].numar;

}while (c[m+1].numar==c[i].numar);

ok=0;

}

if (ok==0)

{

cout<<"Tip: ";

cin>>c[m+1].tip;

cout<<"Cost: ";

cin>>c[m+1].cost;

curatare\_ecran();

cout<<"Introduceti informatiile camerei care a fost eliberata si este acum disponibila clientilor: "<<endl<<endl;

cout<<"Camera nr. "<<c[m+1].numar<<endl;

cout<<"Tip: "<<c[m+1].tip<<endl;

cout<<"Cost: "<<c[m+1].cost<<" de lei/noapte"<<endl<<endl;

cout<<"Confirmati ca doriti adaugarea camerei introduse? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

ok=0;

if (strcmp(raspuns,"nu")==0 || strcmp(raspuns,"Nu")==0 || strcmp(raspuns,"NU")==0)

ok=1;

if (ok==1)

{

cout<<"Ati renuntat la adaugarea camerei!"<<endl;

break;

}

if (ok==0)

{

m++;

curatare\_ecran();

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

cout<<"Aceasta este lista noua cu cele "<<m<<" camere, iar camera nou introdusa este evidentiata prin culoare: "<<endl<<endl;

for (int i=1;i<m;i++)

afisare\_camere(i);

for (int i=1;i<=m;i++)

if (i==m)

{

index\_camera=i;

culoare(3);

afisare\_camere(index\_camera);

culoare(7);

}

cout<<"Doriti sa adaugati alte camere? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

curatare\_ecran();

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

ok=0;

}

if (strcmp(raspuns,"nu")==0 || strcmp(raspuns,"Nu")==0 || strcmp(raspuns,"NU")==0)

{

cout<<"Aceasta este lista noua cu cele "<<m<<" camere, iar camera nou introdusa este evidentiata prin culoare: "<<endl<<endl;

for (int i=1;i<m;i++)

afisare\_camere(i);

for (int i=1;i<=m;i++)

if (i==m)

{

index\_camera=i;

culoare(3);

afisare\_camere(index\_camera);

culoare(7);

}

}

}

}

}while(strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0);

}

void rezervare()

{

int numar\_camera;

int index\_camera=0;

int ok=1;

char raspuns[20];

do

{

if (m==1)

cout<<"Aceasta este camera disponibila:"<<endl<<endl;

else if (m>1)

cout<<"Acestea sunt cele "<<m<<" camere disponibile"<<endl<<endl;

else

{

cout<<"Nu exista camere disponibile!"<<endl<<endl;

ok=0;

}

if(ok==1)

{

for (int i=1;i<=m;i++)

{

afisare\_camere(i);

}

cout<<"Alegeti numarul camerei care a fost rezervata de catre un client: "<<endl;

cin>>numar\_camera;

for (int i=1;i<=m;i++)

if (numar\_camera==c[i].numar)

index\_camera=i;

if (index\_camera==0)

{

do

{

cout<<"Camera nu este disponibila!"<<endl<<endl;

cout<<"Alegeti numarul camerei care a fost rezervata de catre un client: "<<endl;

cin>>numar\_camera;

index\_camera=0;

for (int i=1;i<=m;i++)

if (numar\_camera==c[i].numar)

index\_camera=i;

}while (index\_camera==0);

}

cout<<endl;

curatare\_ecran();

cout<<"Aceasta este camera selectata: "<<endl<<endl;

afisare\_camere(index\_camera);

cout<<"Confirmati ca doriti rezervarea acestei camere pentru un client? (da/nu)"<<endl;

cin>>raspuns;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

for (int i=1;i<=m;i++)

{

if (numar\_camera==c[i].numar)

c[index\_camera]=c[index\_camera+1];

for (i=index\_camera;i<=m;i++)

c[i]=c[i+1];

}

m--;

}

curatare\_ecran();

cout<<"Doriti sa rezervati alte camere? (da/nu)"<<endl;

cin>>raspuns;

index\_camera=0;

curatare\_ecran();

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

for (int i=1;i<=m;i++)

afisare\_camere(i);

curatare\_ecran();

}

if (strcmp(raspuns,"nu")==0 || strcmp(raspuns,"Nu")==0 || strcmp(raspuns,"NU")==0)

{

curatare\_ecran();

cout<<"Acestea sunt cele "<<m<<" camere disponibile ramase: "<<endl<<endl;

for (int i=1;i<=m;i++)

afisare\_camere(i);

}

}

}while(strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0);

}

void ord\_nr\_camere(int crescator)

{

if (crescator==1)

{

for (int i=1;i<=m;i++)

for (int j=1;j<=m;j++)

if (c[i].numar<c[j].numar)

{

camera aux=c[i];

c[i]=c[j];

c[j]=aux;

}

for (int i=1;i<=m;i++)

afisare\_camere(i);

}

else

{

for (int i=1;i<=m;i++)

for (int j=1;j<=m;j++)

if (c[i].numar>c[j].numar)

{

camera aux=c[i];

c[i]=c[j];

c[j]=aux;

}

for (int i=1;i<=m;i++)

afisare\_camere(i);

}

}

void ord\_cost\_camere(int crescator)

{

if (crescator==1)

{

for (int i=1;i<=m;i++)

for (int j=1;j<=m;j++)

if (c[i].cost<c[j].cost)

{

camera aux=c[i];

c[i]=c[j];

c[j]=aux;

}

for (int i=1;i<=m;i++)

afisare\_camere(i);

}

else

{

for (int i=1;i<=m;i++)

for (int j=1;j<=m;j++)

if (c[i].cost>c[j].cost)

{

camera aux=c[i];

c[i]=c[j];

c[j]=aux;

}

for (int i=1;i<=m;i++)

afisare\_camere(i);

}

}

void ordonare\_camere()

{

int crescator;

int ok=1;

char raspuns[15];

char raspuns2[15];

do

{

if (m==1)

cout<<"Aceasta este camera disponibila:"<<endl<<endl;

else if (m>1)

cout<<"Acestea sunt cele "<<m<<" camere disponibile"<<endl<<endl;

else

{

cout<<"Nu exista camere disponibile!"<<endl<<endl;

ok=0;

}

if (ok==1)

for (int i=1;i<=m;i++)

afisare\_camere(i);

cout<<"Alegeti dupa ce criteriu doriti sa ordonati camerele (numar/cost) "<<endl;

cin>>raspuns;

cout<<endl;

cout<<"Doriti sa ordonati crescator sau descrescator? (crescator/descrescator)"<<endl;

cin>>raspuns2;

if(strcmp(raspuns2,"crescator")==0 || (raspuns2,"Crescator")==0 || (raspuns2,"CRESCATOR")==0)

crescator=1;

else

crescator=0;

curatare\_ecran();

if (strcmp(raspuns,"numar")==0 || strcmp(raspuns,"Numar")==0 || strcmp(raspuns,"NUMAR")==0)

{

cout<<"Aceasta este lista camerelor disponibile ordonate dupa numar: "<<endl<<endl;

ord\_nr\_camere(crescator);

cout<<endl;

cout<<"Doriti sa efectuati alta ordonare? (da/nu)"<<endl;

cin>>raspuns;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

for (int i=1;i<=m;i++)

afisare\_camere(i);

curatare\_ecran();

}

else

{

curatare\_ecran();

ok=0;

}

}

if (strcmp(raspuns,"cost")==0 || strcmp(raspuns,"Cost")==0 || strcmp(raspuns,"COST")==0)

{

cout<<"Aceasta este lista camerelor disponibile ordonate dupa cost: "<<endl<<endl;

ord\_cost\_camere(crescator);

cout<<endl;

cout<<"Doriti sa efectuati alta ordonare? (da/nu)"<<endl;

cin>>raspuns;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

curatare\_ecran();

for (int i=1;i<=m;i++)

afisare\_camere(i);

curatare\_ecran();

}

else

{

curatare\_ecran();

ok=0;

}

}

}while (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0);

}

void afisare\_clienti(int i)

{

cout<<"Clientul: "<<i<<endl;

cout<<"Date personale: "<<endl<<"Nume: "<<d[i].nume2<<endl<<"Prenume: "<<d[i].prenume2<<endl<<"Recenzia acordata: "<<d[i].recenzie<<" stele"<<endl<<"Sugestia: "<<d[i].sugestie<<endl<<"Reclamatia: "<<d[i].reclamatie<<endl;

separare(30);

}

void afisare\_recenzii\_clienti(int i)

{

cout<<"Clientul: "<<i<<endl;

cout<<"Date personale: "<<endl<<"Nume: "<<d[i].nume2<<endl<<"Prenume: "<<d[i].prenume2<<endl<<"Recenzia acordata: "<<d[i].recenzie<<" stele"<<endl;

separare(30);

}

void clienti()

{

if (p>1)

{

cout<<"Aceasta este lista cu cei "<<p<<" clienti"<<endl<<endl;

for (int i=1;i<=p;i++)

afisare\_clienti(i);

}

else if (p==1)

{

cout<<"Acesta este clientul din baza de date"<<endl<<endl;

for (int i=1;i<=p;i++)

afisare\_clienti(i);

}

if (p==0)

cout<<"Nu exista clienti in baza de date! Adaugati recenzii ale clientilor pentru a le putea afisa!"<<endl<<endl;

}

void adaugare\_clienti()

{

char raspuns[10];

int ok=0;

int index\_client;

do

{

cout<<"Introduceti informatiile noului client: "<<endl<<endl;

cout<<"Nume: ";

cin>>d[p+1].nume2;

cout<<endl;

cout<<"Prenume: ";

cin>>d[p+1].prenume2;

cout<<endl;

for (int i=1;i<=p;i++)

for (int i=p;i>=1;i--)

if (strcmp(d[i].nume2,d[p+1].nume2)==0)

if (strcmp(d[i].prenume2,d[p+1].prenume2)==0)

{

do

{

ok=1;

cout<<"Clientul introdus a distribuit deja recenzia sa!"<<endl;

cout<<"Introduceti informatiile noului client: "<<endl<<endl;

cout<<"Nume: ";

cin>>d[p+1].nume2;

cout<<endl;

cout<<"Prenume: ";

cin>>d[p+1].prenume2;

cout<<endl;

}while(strcmp(d[i].nume2,d[p+1].nume2)==0 && strcmp(d[i].prenume2,d[p+1].prenume2)==0);

ok=0;

}

if (ok==0)

{

cout<<"Recenzia: ";

cin>>d[p+1].recenzie;

if (d[p+1].recenzie > 5 || d[p+1].recenzie <= 0)

do

{

cout<<"Recenzia distribuita trebuie sa fie cuprinsa intre 1 si 5 stele!"<<endl;

cout<<"Recenzia: ";

cin>>d[p+1].recenzie;

}while (d[p+1].recenzie > 5 || d[p+1].recenzie < 0);

curatare\_ecran();

cout<<"Introduceti informatiile noului client: "<<endl<<endl;

cout<<"Nume: "<<d[p+1].nume2<<endl<<endl;

cout<<"Prenume: "<<d[p+1].prenume2<<endl<<endl;

cout<<"Recenzia: "<<d[p+1].recenzie<<" stele"<<endl;

cout<<endl;

cout<<"Sugestia: ";

cin.getline(d[p+1].sugestie,100);

cin.getline(d[p+1].sugestie,100);

cout<<endl;

cout<<"Reclamatia: ";

cin.getline(d[p+1].reclamatie,100);

cout<<endl;

cout<<"Confirmati ca doriti adaugarea recenziei introduse? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

ok=0;

if (strcmp(raspuns,"nu")==0 || strcmp(raspuns,"Nu")==0 || strcmp(raspuns,"NU")==0)

ok=1;

if (ok==1)

{

cout<<"Ati renuntat la adaugarea recenziei clientului!"<<endl;

break;

}

if (ok==0)

{

p++;

curatare\_ecran();

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

{

cout<<"Aceasta este lista noua cu cei "<<p<<" clienti, iar clientul nou introdus este evidentiat prin culoare: "<<endl<<endl;

for (int i=1;i<p;i++)

afisare\_clienti(i);

for (int i=1;i<=p;i++)

if (i==p)

{

index\_client=i;

culoare(3);

afisare\_clienti(index\_client);

culoare(7);

}

cout<<"Doriti sa adaugati alte recenzii ale clientilor? (da/nu)"<<endl;

cin>>raspuns;

cout<<endl;

curatare\_ecran();

if (strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0)

ok=0;

}

if (strcmp(raspuns,"nu")==0 || strcmp(raspuns,"Nu")==0 || strcmp(raspuns,"NU")==0)

{

cout<<"Aceasta este lista noua cu cei "<<p<<" clienti, iar clientul nou introdus este evidentiat prin culoare: "<<endl<<endl;

for (int i=1;i<p;i++)

afisare\_clienti(i);

for (int i=1;i<=p;i++)

if (i==p)

{

index\_client=i;

culoare(3);

afisare\_clienti(index\_client);

culoare(7);

}

}

}

}

}while(strcmp(raspuns,"da")==0 || strcmp(raspuns,"Da")==0 || strcmp(raspuns,"DA")==0);

}

void sugestie()

{

int m=0;

cout<<"Sugestiile clientilor: "<<endl<<endl;

for (int i=1;i<=p;i++)

{

if (strlen(d[i].sugestie) + strlen(d[i].nume2) + strlen(d[i].prenume2)>m)

m=(strlen(d[i].sugestie) + strlen(d[i].nume2) + strlen(d[i].prenume2));

if (m>120)

m=113;

}

for (int i=1;i<=p;i++)

{

cout<<d[i].nume2<<" "<<d[i].prenume2<<" : \""<<d[i].sugestie<<" \""<<endl;

separare(m+7);

}

cout<<endl;

cout<<"Multumim tuturor pentru sugestiile dumneavoastra, care ne ajuta sa ne imbunatatim serviciile!"<<endl<<endl;

}

void reclamatie()

{

cout<<"Reclamatiile clientilor: "<<endl<<endl;

int m=0;

for (int i=1;i<=p;i++)

{

if (strlen(d[i].reclamatie) + strlen(d[i].nume2) + strlen(d[i].prenume2)>m)

m=(strlen(d[i].reclamatie) + strlen(d[i].nume2) + strlen(d[i].prenume2));

if(m>120)

m=113;

}

for (int i=1;i<=p;i++)

{

cout<<d[i].nume2<<" "<<d[i].prenume2<<" : \""<<d[i].reclamatie<<" \""<<endl;

separare(m+7);

}

cout<<endl;

cout<<"Ne cerem scuze pentru aceste inconveniente si ne straduim sa efectuam schimbarile necesare!"<<endl<<endl;

}

void meniu\_angajati()

{

char a;

do

{

curatare\_ecran();

cout<<"Aceasta este sectiunea de gestiune a angajatilor!"<<endl;

cout<<"Apasati tasta corespunzatoare numarului de ordine a actiunii pe care doriti sa o efectuati! "<<endl<<endl;

cout<<"1. Prezentarea angajatilor"<<endl;

cout<<"2. Cautarea angajatilor dupa nume "<<endl;

cout<<"3. Adaugarea unui nou angajat"<<endl;

cout<<"4. Stergerea unui angajat"<<endl;

cout<<"5. Modificarea datelor a unui angajat"<<endl;

cout<<"6. Ordonarea angajatilor"<<endl;

cout<<"0. Iesire din sectiunea angajati"<<endl;

do

{

a=getch();

} while (a<'0' || a>='7');

curatare\_ecran();

switch (a)

{

case '0':break;

case '1':afisare();system ("pause");break;

case '2':cautare();system ("pause");break;

case '3':adaugare();system ("pause");break;

case '4':stergere();system ("pause");break;

case '5':modificare();system ("pause");break;

case '6':ordonare();system("pause");break;

}

}while (a!='0');

}

void meniu\_camere()

{

char a;

do

{

curatare\_ecran();

cout<<"Aceasta este sectiunea de gestiune a camerelor!"<<endl;

cout<<"Apasati tasta corespunzatoare numarului de ordine a actiunii pe care doriti sa o efectuati! "<<endl<<endl;

cout<<"1. Afisarea camerelor disponibile"<<endl;

cout<<"2. Adaugarea unei camere"<<endl;

cout<<"3. Rezervarea unei camere"<<endl;

cout<<"4. Ordonarea camerelor"<<endl;

cout<<"0. Iesire din sectiuena camere"<<endl;

do

{

a=getch();

} while (a<'0' || a>='5');

curatare\_ecran();

switch (a)

{

case '0':break;

case '1':camere();system ("pause");break;

case '2':adaugare\_camere();system ("pause");break;

case '3':rezervare();system ("pause");break;

case '4':ordonare\_camere();system ("pause");break;

}

}while (a!='0');

}

void meniu\_recenzii()

{

char a;

do

{

curatare\_ecran();

cout<<"Aceasta este sectiunea de gestiune a recenzilor oferite de clientii care au folosit serviciile noastre!"<<endl;

cout<<"Apasati tasta corespunzatoare numarului de ordine a actiunii pe care doriti sa o efectuati! "<<endl<<endl;

cout<<"1. Prezentarea recenziilor clientilor"<<endl;

cout<<"2. Adaugarea recenziei unui client"<<endl;

cout<<"3. Sugestiile clientilor"<<endl;

cout<<"4. Reclamatiile clientilor"<<endl;

cout<<"0. Iesire din sectiunea clienti"<<endl;

do

{

a=getch();

} while (a<'0' || a>='5');

curatare\_ecran();

switch (a)

{

case '0':break;

case '1':clienti();system ("pause");break;

case '2':adaugare\_clienti();system ("pause");break;

case '3':sugestie();system ("pause");break;

case '4':reclamatie();system ("pause");break;

}

}while (a!='0');

}

void meniu\_principal()

{

char a;

do

{

curatare\_ecran();

cout<<"Bine ati venit in zona de gestiune a hotelului!"<<endl;

cout<<"Apasati tasta corespunzatoare numarului de ordine a sectiunii pe care doriti sa o accesati! "<<endl<<endl;

cout<<"1. Sectiunea angajati"<<endl;

cout<<"2. Sectiunea camere"<<endl;

cout<<"3. Sectiunea clienti"<<endl;

cout<<"0. Iesire din aplicatie"<<endl;

do

{

a=getch();

} while (a<'0' || a>='4');

curatare\_ecran();

switch (a)

{

case '0':cout<<"Ati parasit zona de gestiune a hotelului nostru. Pentru mai multe detalii nu ezitati sa ne contactati! "<<endl;break;

case '1':meniu\_angajati();system ("pause");break;

case '2':meniu\_camere();system ("pause");break;

case '3':meniu\_recenzii();system ("pause");break;

}

}while (a!='0');

}

int main()

{

ifstream in ("angajati.in");

in>>n;

for (int i=1;i<=n;i++)

in>>v[i].nume>>v[i].prenume>>v[i].varsta>>v[i].salariu;

in.close();

ifstream f ("camere.in");

f>>m;

for (int i=1;i<=m;i++)

f>>c[i].numar>>c[i].tip>>c[i].cost;

in.close();

ifstream g ("clienti.in");

g>>p;

for (int i=1;i<=p;i++)

{

g>>d[i].nume2>>d[i].prenume2>>d[i].recenzie;

g.getline(d[i].sugestie,200);

g.getline(d[i].reclamatie,200);

}

in.close();

meniu\_principal();

return 0;

}