TEMA 6

*Să se realizeze un program care să asigure gestionarea taxelor studenţilor dintr-o universitate. Universitatea este structurată pe facultăţi şi departamente (ex. AC,CT, DCLS, CH, MEC). La fiecare student se cunoaşte: nume, prenume, marcă, an\_studiu, profil, suma\_plată, suma\_achitată.La pornirea aplicaţiei va fi afişat următorul meniu:*

1. *Încărcare gestiune din fişier.*
2. *Adăugare student nou la o anumită facultate.*
3. *Ştergere student de la o anumită facultate, dat după marcă.*
4. *Căutare student de la o anumită facultate.*
5. *Încasare student (se introduce suma plătită, se scade din suma de plată şi se adaugă la cea achitată).*
6. *Afişare studenţi care mai au de plătit de la o anumită facultate.*
7. *Calculare sumă încasată pe o anumită facultate.*
8. *Salvare gestiune în fişiere pe facultăţi (câte un fişier pentru fiecare facultate).*
9. *Ieşire*

Proiectul va fi predat pe CD împreună cu o documentaţie care să conţină:

* *numele, anul, grupa celor care au realizat proiectul*
* *descrierea proiectului (funcţii şi metode folosite);*
* *structura fişierelor folosite;*
* *explicarea detaliată a programului.*

Proiectul acesta, pregatit pentru examenul de la Programare Orientata pe Obiect/Object Oriented Programming {POO/OOP}, a fost pregatit si dezvoltat de studentii:

* Murariu Andrei-Stefanel, Grupa 203, Anul 2 – Invatamant cu Frecventa | 2022/2023
  + Cont Github: [iamxorum (Murariu Andrei-Stefanel) (github.com)](https://github.com/iamxorum)

Descrierea Proiectului | TEMA 6

Proiectul dezvoltat e varianta mai complexa si mai vasta fata de acela din cerinta; Programul se prezinta cu un menu la pornire; acesta Va permite incarcarea unui Database compus din 4 fisiere {SI ESTE OBLIGATORIU CA INCARCAREA SA SE FACA LA INCEPUT, deoarece nu exista inca implementata functia de a crea o noua universitate}:

1. Fisierul pentru incarcarea datelor ale universităților din fisierul universities.txt
2. Fisierul pentru incarcarea datelor ale facultăților din fisierul faculties.txt
3. Fisierul pentru incarcarea datelor ale departamentelor din fisierul departments.txt
4. Fisierul pentru incarcarea datelor ale studenților din fisierul students.txt

Dupa incarcarea datelor, in database-ul programului, din fiserele anticipate anterior, puteti lucra pe datele incarcate:

1. Puteti adauga o noua facultate
2. Puteti adauga un nou student
3. Puteti adauga un nou departament
4. Puteti elimina un student din database, introducand codul lui de student
5. Puteti cauta un student dintr-o anumita facultate, introducand codul lui de student
6. Puteti incarca achitarea partial/totala a taxei universitare din partea unui student in databse
7. Puteti cere o lista, pentru o anumita facultate, care contine toti studentii care inca mai trebuie sa plateasca taxa {si cat a ramas sa plateasca}
8. Puteti calcula suma care, o anumita facultate, a incasat-o de la toti studentii inscrisi
9. Puteti salva tot ce ati lucrat in program, actualizat in acelasi fisiere de incarcare a database-ului + va crea si o structura de directories pe universitati si facultati unde se vor afisa datele studentilor si departamentelor.

Trecem la descrierea programului:

IMPORTANT: In programul acesta s-au folosit ABSOLUTE PATHS

1. Adaugarea unei noi facultati



Acest cod implementează o funcționalitate de adăugare a unei facultăți la o universitate existentă.

1. Variabila "found" este inițializată la false și va fi utilizată mai târziu pentru a indica dacă universitatea a fost găsită sau nu.
2. Variabila "i" va fi utilizată pentru a stoca poziția universității în vectorul de universități.
3. Se apelă funcția "searchUniversity" din clasa "University" pentru a căuta universitatea dorită și pentru a determina poziția acesteia în vector.
4. Dacă universitatea a fost găsită (found = true), se adaugă o nouă facultate prin apelarea funcției "addFaculty" din clasa "University" și se actualizează lista de facultăți prin apelarea funcției "updateFacultyList".
5. Dacă universitatea nu a fost găsită, se afișează mesajul "University not found."
6. Se afișează un rând gol și se termină acțiunea.
7. Adaugarea unui nou student



Acest cod implementează o funcționalitate de adăugare a unui student la o facultate existentă.

1. Variabila "found" este inițializată la false și va fi utilizată mai târziu pentru a indica dacă facultatea a fost găsită sau nu.
2. Variabilele "i" și "j" vor fi utilizate pentru a stoca pozițiile universității și facultății respective în vectorii de universități și facultăți.
3. Se apelă funcția "searchFaculty" din clasa "University" pentru a căuta facultatea dorită și pentru a determina pozițiile universității și facultății respective în vectorii de universități și facultăți.
4. Se obține lista de facultăți prin apelarea funcției "getListFaculties" din clasa "University".
5. Se verifică dacă poziția facultății este mai mică decât numărul de elemente din lista de facultăți.
6. Dacă facultatea a fost găsită (found = true), se creează un nou student prin apelarea funcției "createStudent" din clasa "Faculty" și se adaugă la lista de studenți prin apelarea funcției "addStudent".
7. Universitatea și facultatea sunt actualizate prin apelarea funcției "updateFaculty" din clasa "University".
8. Dacă facultatea nu a fost găsită, se afișează mesajul "Faculty not found."
9. Lista de facultăți este golita prin apelarea funcției "clear".
10. Se afișează un rând gol și se termină acțiunea.
11. Adaugarea unui nou departament



1. Variabila "found" este inițializată la false și va fi utilizată mai târziu pentru a indica dacă facultatea a fost găsită sau nu.
2. Variabilele "i" și "j" vor fi utilizate pentru a stoca pozițiile universității și facultății respective în vectorii de universități și facultăți.
3. Se apelă funcția "searchFaculty" din clasa "University" pentru a căuta facultatea dorită și pentru a determina pozițiile universității și facultății respective în vectorii de universități și facultăți.
4. Se obține lista de facultăți prin apelarea funcției "getListFaculties" din clasa "University".
5. Se verifică dacă poziția facultății este mai mică decât numărul de elemente din lista de facultăți.
6. Dacă facultatea a fost găsită (found = true), se creează un nou departament prin apelarea funcției "createDepartment" din clasa "Faculty" și se adaugă la lista de departamente prin apelarea funcției "addDepartment".
7. Universitatea și facultatea sunt actualizate prin apelarea funcției "updateFaculty" din clasa "University".
8. Dacă facultatea nu a fost găsită, se afișează mesajul "Faculty not found."
9. Lista de facultăți este golita prin apelarea funcției "clear".
10. Se afișează un rând gol și se termină acțiunea.
11. Stergerea unui student

Immagine che contiene testo

Descrizione generata automaticamente

Acest caz se ocupă cu ștergerea unui student din lista studenților unei facultăți. În primul rând, se caută facultatea în care se află studentul prin intermediul funcției "searchFaculty". Apoi, se solicită introducerea codului studentului și se parcurge lista de studenți din facultatea găsită. Dacă studentul cu codul introdus este găsit, acesta este eliminat din lista de studenți a facultății. În final, lista de facultăți din universitatea găsită este actualizată.

1. Cautarea unui student

Immagine che contiene testo

Descrizione generata automaticamente

Acest caz verifică dacă există o facultate în universitatea selectată. Dacă da, se cere utilizatorului să introducă codul de identificare al studentului pe care dorește să îl caute. Programul parcurge lista de studenți a facultății selectate și verifică dacă există un student cu codul introdus. Dacă există, studentul este afișat pe ecran. Dacă nu există, programul afișează un mesaj că studentul nu a fost găsit.

1. Incasare student



Această instrucțiune caz tratează opțiunea 7 din meniul principal al programului. Funcția caută un student în universitatea selectată de utilizator și întreabă utilizatorul să introducă codul studentului pe care dorește să îl caute. Dacă studentul este găsit, programul afișează informații despre taxa anuală a studentului și întreabă utilizatorul să introducă suma pe care dorește să o plătească. Apoi, programul calculează dacă studentul a plătit taxa integral sau dacă mai trebuie să plătească și afișează un mesaj adecvat în funcție de situație.

1. Lista, pentru o anumita facultate, care contine toti studentii care inca mai trebuie sa plateasca taxa



Acest caz se referă la afișarea studenților care încă nu au plătit întreaga taxă anuală de la o facultate specifică. Utilizatorul va fi întrebat să selecteze facultatea și apoi programul va afișa lista cu studenții care trebuie să plătească încă taxa. Dacă toți studenții au plătit taxa, se va afișa un mesaj corespunzător.

1. Calculare suma care, o anumita facultate, a incasat-o de la toti studentii inscrisi



Instrucțiunea de caz 9: Se caută facultatea specificată de utilizator și se calculează totalul sumei de bani primită de această facultate din taxele anuale plătite de studenți. Rezultatul este afișat pe ecran.

1. Immagine che contiene testo

   Descrizione generata automaticamenteSalvare database in fisier

Acest cod se ocupa cu scrierea unor informatii despre universitati, facultati, studenti si departamente intr-un fisier text. Este creat un director "Universities" la calea specificata "C:\Users\4ndr3\Desktop\Facultate\POO\Proiect\_0902\Universities". Apoi, se scriu informatiile despre universitati in fisierul "universities.txt". Informatiile despre facultati sunt scrise in fisierul "faculties.txt". Informatiile despre studenti sunt scrise in fisierul "students.txt". Pentru fiecare facultate din fiecare universitate, este creat un director cu numele facultatii in directorul "Universities". Apoi, in acest director, se scrie informatia despre departamentele din facultate si studentii din facultate in fisierul cu acelasi nume ca si numele facultatii si extensia ".txt". Toate informatiile sunt scrise folosind obiecte de tip ofstream. Se folosesc functia mkdir pentru a crea directoare si functia strcpy si strcat pentru a manipula sirurile de caractere (calea catre directoare si fisiere).