***UNIVERSITATEA “STEFAN CEL MARE”, SUCEAVA***

***FACULTATEA DE INGINERIE ELECTRICA SI STIINTA CALCULATOARELOR***

***SPECIALIZAREA CALCULATOARE***

**PROIECT DISCIPLINA POO**

*[ Gestionarea unei biblioteci in C++ ]*

*Andries George Serban*

TEMA PROIECT

# **TEMA SI MOTIVATIA ALEGERII**

Tema proiectului este imprelentarea unui program in C++ care are ca scop gestionarea unei biblioteci. Acest program se va folosi de o baza de date (in cazul nostru este un fisier) si ne va permite sa adaugam / stergem carti cu scopul final de a organiza sau sorta pe anumite criterii.

CUPRINS

**Cuprins**

[TEMA SI MOTIVATIA ALEGERII](https://docs.google.com/document/d/1K19lHk5TpI3aszd6qhTGbMKaoqiE6SW6/edit#heading=h.gjdgxs) 1

[1. ELEMENTE TEORETICE](https://docs.google.com/document/d/1K19lHk5TpI3aszd6qhTGbMKaoqiE6SW6/edit#heading=h.30j0zll) 3

[1.1. DESCRIEREA PROBLEMEI](https://docs.google.com/document/d/1K19lHk5TpI3aszd6qhTGbMKaoqiE6SW6/edit#heading=h.1fob9te) 3

[1.2. ABORDAREA TEORETICA A PROBLEMEI](https://docs.google.com/document/d/1K19lHk5TpI3aszd6qhTGbMKaoqiE6SW6/edit#heading=h.3znysh7) 3

[1.3. ELEMENTE SPECIFICE POO](https://docs.google.com/document/d/1K19lHk5TpI3aszd6qhTGbMKaoqiE6SW6/edit#heading=h.2et92p0) 3

[2. IMPLEMENTARE](https://docs.google.com/document/d/1K19lHk5TpI3aszd6qhTGbMKaoqiE6SW6/edit#heading=h.3dy6vkm) 4

[2.1. TEHNOLOGII FOLOSITE](https://docs.google.com/document/d/1K19lHk5TpI3aszd6qhTGbMKaoqiE6SW6/edit#heading=h.1t3h5sf) 4

[2.2. DIAGRAMA DE CLASE](https://docs.google.com/document/d/1K19lHk5TpI3aszd6qhTGbMKaoqiE6SW6/edit#heading=h.4d34og8) 4

[2.3. ALTE CAPITOLE SPECIFICE 5](https://docs.google.com/document/d/1K19lHk5TpI3aszd6qhTGbMKaoqiE6SW6/edit#heading=h.17dp8vu)

[3. ANALIZA SOLUTIEI IMPLEMENTATE 6](https://docs.google.com/document/d/1K19lHk5TpI3aszd6qhTGbMKaoqiE6SW6/edit#heading=h.3rdcrjn)

[3.1. FORMATUL DATELOR DE I/O 6](https://docs.google.com/document/d/1K19lHk5TpI3aszd6qhTGbMKaoqiE6SW6/edit#heading=h.26in1rg)

[4. MANUAL DE UTILIZARE](https://docs.google.com/document/d/1K19lHk5TpI3aszd6qhTGbMKaoqiE6SW6/edit#heading=h.1ksv4uv) 8

[5. CONCLUZII](https://docs.google.com/document/d/1K19lHk5TpI3aszd6qhTGbMKaoqiE6SW6/edit#heading=h.44sinio) 10

[6. BIBLIOGRAFIE](https://docs.google.com/document/d/1K19lHk5TpI3aszd6qhTGbMKaoqiE6SW6/edit#heading=h.2jxsxqh) 11

[6.1. CARTI](https://docs.google.com/document/d/1K19lHk5TpI3aszd6qhTGbMKaoqiE6SW6/edit#heading=h.z337ya) 11

[6.3. SURSE BIBLIOGRAFICE DIVERSE](https://docs.google.com/document/d/1K19lHk5TpI3aszd6qhTGbMKaoqiE6SW6/edit#heading=h.2xcytpi) 11

CAPITOLUL I

# **ELEMENTE TEORETICE**

# **DESCRIEREA PROBLEMEI**

Biblioteca sufera de o problema uriasa in ceea ce priveste ordinea. Mi-am propus sa vin cu o solutie acestei probleme implementand un program in C++ care are ca scop ordonarea cartilor aflate in biblioteca dupa criterii variate, si simultan facand mult mai usoara gasirea unui exemplar.

# **ABORDAREA TEORETICA A PROBLEMEI**

Problema se va folosi de un fisier text din care va citi si in care va scrie datele programului.

Utilizatorul va avea posibilitatea de a introduce date atat direct in fisierul text, cat si in consola, apeland metodele special create pentru a simplifica citirea datelor. Acesta va permite adaugarea cartilor si stergerea celor deja existente. Se va folosi de o metoda de sortare pentru a ne afisa in consola elementele cerute.

# **ELEMENTE SPECIFICE POO**

Printre elementele specifice POO in proiect se vor regasi:

* Clasele: definirea acestora este esentiala pentru programarea orientata pe obiecte, acestea sunt sabloane pentru obiecte, care contin atribute(variabile) si metode(functii) care descriu comportamentul oriectelor.
* Obiectele: un obiect este o instanta a unei clase si poate fi creat dinamic in timpul rularii programului, acestea contin atribute definite in clasa corespunzatoare si pot apela metodele definite de clasa.
* Incapsuarea: procedeul de incapsulare se refera la impachetarea datelor si functionalitatilor intr-un singur obiect, aceasta protejeaza datele si functionalitatile de accesul neautorizat facand posibila abstractizarea.
* Abstractizarea: se refera la definirea unei interfete simplificate pentru un obiect, prin ascunderea detaliilor sale interne, face ca un obiect sa fie mai usor de utilizat si gestionat.

CAPITOLUL II

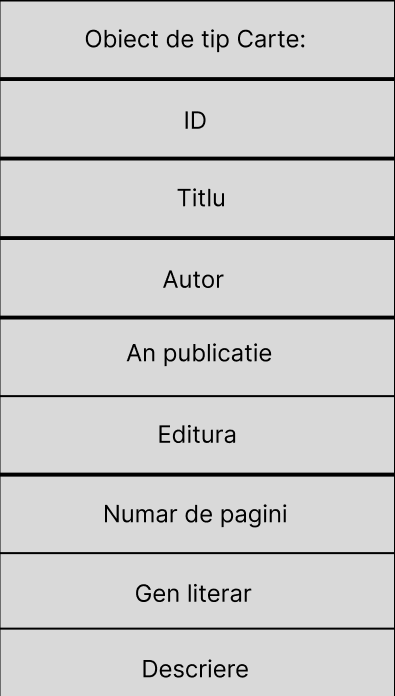
# **IMPLEMENTARE**

Problema a fost abordata in modul in care biblioteca ne cere sa gestionam cartile care se afla deja in aceasta pentru a usura cautarea utilizatorului. Totodata noi implementam functiile prin care putem adauga sau sterge carti in biblioteca, dupa bunul plac si dorintele cititorilor.

# **TEHNOLOGII FOLOSITE**

Programul este realizat integral in limbajul de programare C++. Iar ca mediul de dezvoltare(IDE) folosit am ales CodeBlocks, acesta fiind unul foarte usor de utilizat si cu care am interactionat anterior in cadrul facultatii.

# **DIAGRAMA DE CLASE, SCHEMA BLOC, WORKFLOW**



Proiectul utilizeaza un vector de obiecte in care sunt inmagazinate datele in timpul executiei programului, si un fisier pentru a le salva.

Structura unui obiect de tip Carte este prezentata in imaginea alaturata. Acesta contine campuri pentru ID, titlu, autor, an publicatie, editura, numar de pagini, gen literar si de asemenea contine o mica descriere pentru fiecare carte in parte.

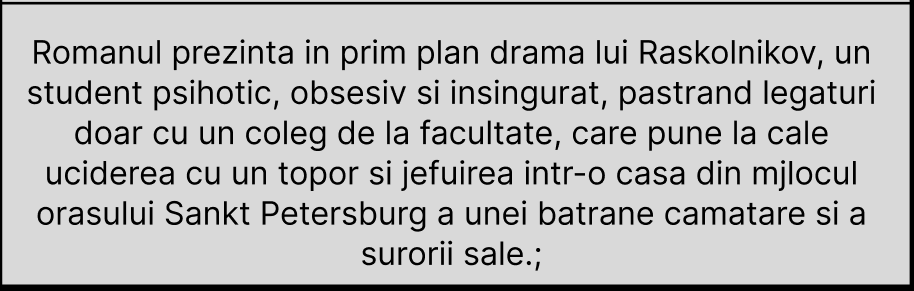
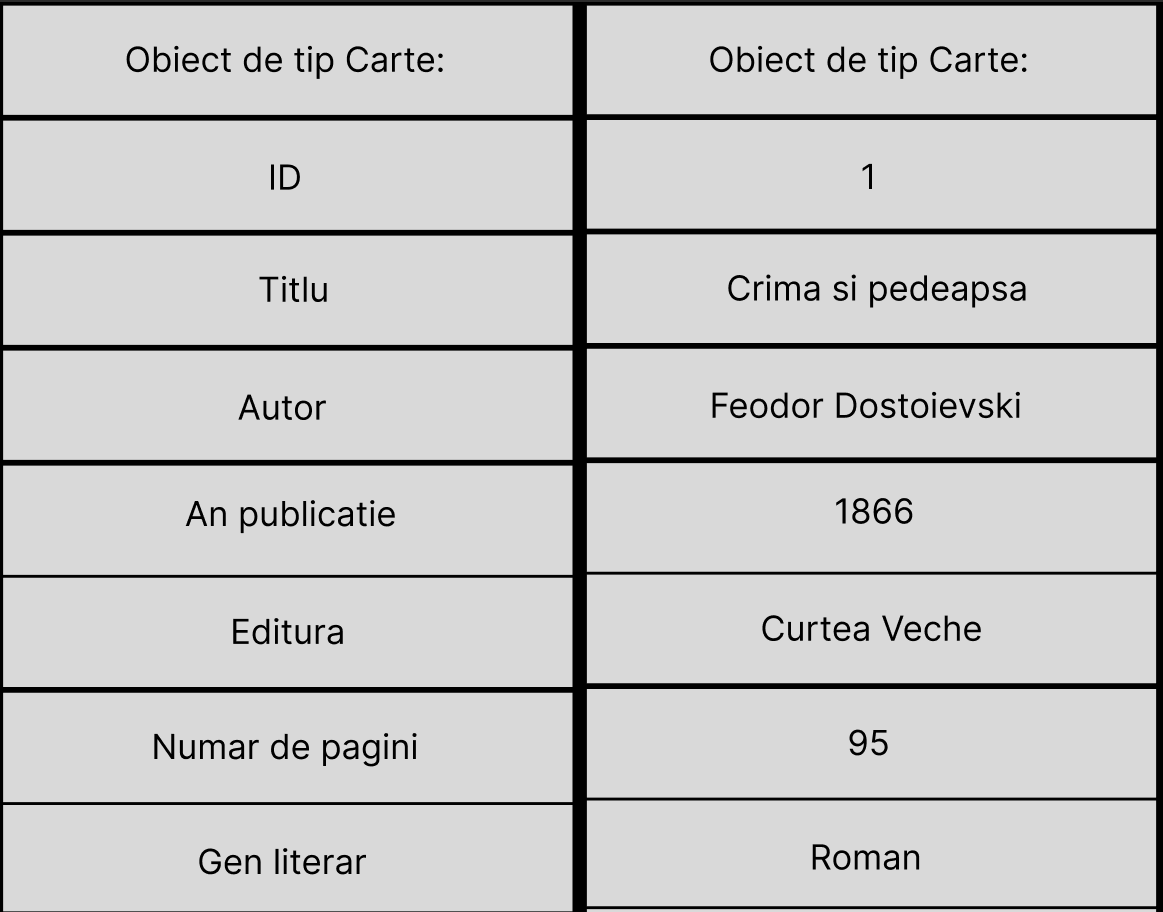
Pentru a ne folosi de obiect utilizam metode care citesc o carte din fisier si o introduc in vector, citesc o carte folosindu-se de datele introduse de user, afiseaza o carte

# **ALTE CAPITOLE SPECIFICE**

Datele in fisierul text sunt structurate in modul urmator: fiecare linie in parte reprezinta informatiile despre o singura carte, iar pe randul respectiv acestea sunt separate intre ele prin caracterul “ ; ”. Datele sunt separate pe mai multe categorii iar fiecare dintre acestea are o semnificatie diferita.

Ordinea in care acestea sunt prezente in fisier este urmatoarea:

* ID;
* Titlu;
* Autor;
* Anul publicatiei;
* Editura;
* Numarul de pagini;
* Genul literar;
* Descriere;

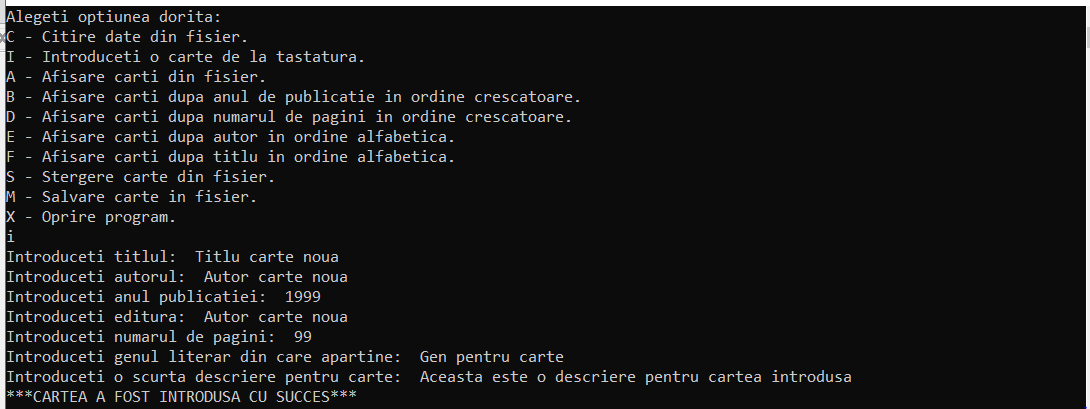
Un exemplu pentru aceasta poate fi vazut in urmatoarea imagine:

CAPITOLUL III

# **ANALIZA SOLUTIEI IMPLEMENTATE**

# **FORMATUL DATELOR DE I/O**

Pentru a introduce date in program avem doua optini. Prima optiune si cea mai putin practica este sa introducem datele dorite direct in fisierul text de care se foloseste proiectul, in formatul dorit, iar la cea mai mica greseala in format programul nu va mai rula si vom avea o eroare. A doua optiune si cea mult mai usoara este sa ne folosim de functia programului care ne permite sa introducem date in acesta. Ne va fi specificata ordinea in care trebuie sa introducem informatiile pentru fiecare camp continut de structura carte.



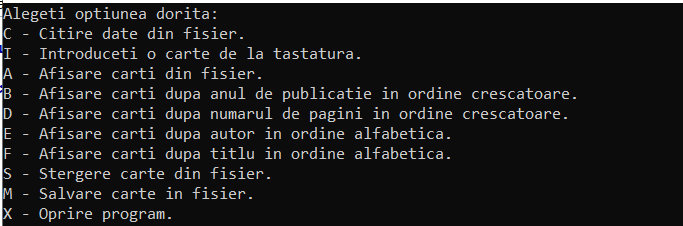
Datele de iesire ale programului pot fi furnizate in mai multe moduri. Informatiile despre carti pot fi afisate in ordinea in care au fost introduse cartile sau putem sa ordona cartile dupa cateva criterii. Acestea pot fi sortate crescator dupa anul publicatiei, crescator dupa numarul de pagini, alfabetic dupa autorul cartii sau titlul acesteia.



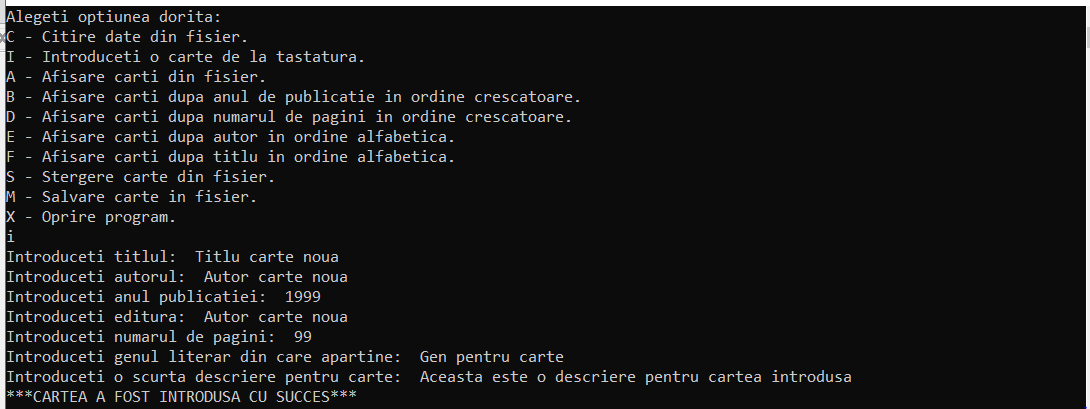
In figura anterioara putem observa cartile afisate in ordinea in care acestea au fost salvate iar in cea ce urmeaza sunt ordonate dupa numarul de pagini.

CAPITOLUL IV

# **MANUAL DE UTILIZARE**

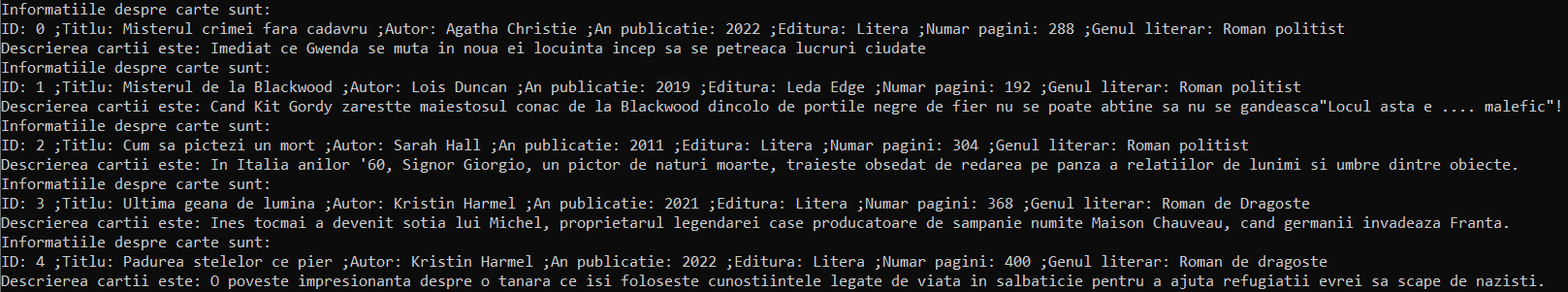
Prima data cand pornim programul vom fi intampinati in consola de aceste optiuni.

Programul nu va avea nimic stocat in baza de date(in cazul nostru fisierul text) si va trebui sa introducem informatii despre cateva carti pentru a vedea cum functioneaza acesta. Va trebui sa apelam optiunea “I” care ne va ajuta sa introducem datele pentru prima carte.



Dupa ce am citit datele pentru o carte trebuie si dorim sa o salvam in fisierul text trebuie sa apelam optiunea “M”.

Vom repeta acest proces de cateva ori pentru a putea observa mai bine ce se intampla in continuare in aplicatie.

Dupa introducerea in fisier a catorva carti putem apela optiunea una din optiunile de afisare incluse in program. Introducem caracterul “A” si ne vor fi afisate informatiile despre fiecare carte introdusa. Pentru fiecare exemplar sunt specificate: titlul, autorul, anul publicatiei, editura, numarul de pagini, genul literar si o scurta descriere.

Pe langa aceasta metoda de a afisare mai sunt disponibile alte 4 metode care ne permit sa ordonam datele dupa anumite criterii pentru o mai buna citire si o intelegere mai usoara a acestora, atunci cand cautam un lucru specific. Aceste 4 criterii sunt: ordonare crescatoare dupa anul in care a fost publicata cartea, ordonare crescatoare dupa numarul de pagini pe care aceasta il contine, ordonare alfabetica dupa titlui sau ordonare alfabetica dupa autorul acesteia.

Pe langa optiunile de introducere, salvare si afisare a datelor avem si posibilitate de a sterge o anumita carte din fisier, procesul realizandu-se pe baza unui ID introdus de utilizator.



Programul poate fi oprit oricand din executie cu optiunea “X”.

Cand vom deschide din nou aplicatia se vor citi automat datele salvate in fisierul text in executiile anterioare si nu va mai fi necesara introducerea tuturor informatiilor.

CAPITOLUL V

# **CONCLUZII**

In opinia mea aplicatia de gestiune a unei biblioteci s-a dovedit a fi putin mai complicata decat ma asteptam, nu din cauza complexitatii arhitecturii acesteia, ci din vina tuturor problemelor care au aparut pe parcursul dezvoltarii structurii unei clase care contine mai multe atribute si metode, cat si construirea fiecarei metode cu scopul de a realiza o anumita operatiune si evitarea tuturor conflictelor ce pot aparea.

Cateva avantaje pentru realizarea aplicatiei in limbajul de programare C++ sunt: este un limbaj de nivel scazut si permite programatorilor sa optimizeze si sa gestioneze resursele sistemului ce duce la o performanta ridicata; ofera un set bogat de functionalitati si caracteristici care permite dezvoltarea unei game largi de aplicatii; permite contrulul direct asupra memoriei ceea ce ofera posibilitatea de a aloca sau dealoca resurse intr-un mod foarte eficient; numarul mare de biblioteci care face posibila dezvoltarea rapida a aplicatiilor si extindere a functionalitatilor.

Cu toate cele enumerate mai sus, limbajul de programare C++ nu este perfect, acestuia ii lipseste un sistem de gestionare automata a memoriei ce poate preveni scurgerile de memorie si erorile relative acestora.

O directie in care aplicatia poate continua dupa parerea mea este scalarea acesteia, mai multi utilizatori sa aiba acces la ea, inlocuirea fisierului text pentru stocarea datelor cu o baza de date si dezvoltarea unei interfete mult mai prietenoasa pentru cei ce o folosesc.

CAPITOLUL VI

# **BIBLIOGRAFIE**

# **CARTI**

# Totul despre C si C++ manualul fundamental de programare in C si C++ - Kris Jamsa, Lars Klander.

# **SURSE BIBLIOGRAFICE DIVERSE**

<https://www.geeksforgeeks.org/c-plus-plus/?ref=shm>

<https://github.com/>

<https://cplusplus.com/reference/cstring/strcmp/>

<https://stackoverflow.com/>