8. Admitere facultate

Chira Tudor, 922

Enuntul problemei:

Se doreste dezvoltarea unui program pentru gestiunea candidatilor care se inscriu la facultate de matematica si informatica. Programul permite adaugarea unui nou candidat(cnp, nume, adresa, medie bac, mediile la matematica sau informatica din liceu; media de concurs este: 50% media de la bac si 50% media de la matematica/informatica din timpul liceului), modificarea datelor unui candidat deja inscris(nume, adresa, cnp, optiuni, media de la bac, mediile generale din liceu la matematica/informatica) generarea rezultatelor, afisarea tuturor candidatilor admisi pentru fiecare sectie, afisarea candidatilor respinsi.
Obs. Un candidat poate opta la oricare dintre sectiile oferite de facultate. Numarul, numele sectiilor si numarul de locuri disponibile sunt configurabile (nu sunt constante in program).

Functionalitati:

Data file: candidates.txt — contine lista tuturor candidatilor inscrisi
departments.txt — contine lista tuturor sectiilor disponibile

- Modificarea unei sectii existente
  - Input: Sectia selectata de user prin intermediul interfetei grafice, cu posibilitatea de a modifica Numar sectie : int, Numele sectiei : String, numarul de locuri disponibile : int
  - Output: Lista updatata continand sectiile, updatata cu modificarile aduse sectiei selectate, in "departments.txt"

- Adaugarea unui candidat nou
  - Input: CNP: String de 13 caractere, continand doar caractere numerice, nume: String, adresa : String, medie bac: float, medie mate/info : float, sectii alese : int (maxim 5 sectii) , media de admitere este calculata automat la adaugarea candidatului
  - Output: Lista updatata continand toti introdusi precendent + candidatul nou introdus, salvata in fisierul "candidates.txt"

- Modificarea datelor unui candidat existent
  - Input: Candidatul selectat de user prin intermediul interfetei grafice, cu posibilitatea de a modifica CNP: String de 13 caractere, continand doar caractere numerice, nume: String, adresa : String, medie bac: float, medie mate/info : float, sectii alese : int (maxim 5 sectii)
  - Output: Lista updatata continand canditatii, updatat cu modificarile aduse canditatului selectat, in "candidates.txt"

- Adaugarea unei sectii noi
  - Input: Numar sectie : int, Numele sectiei : String, numarul de locuri disponibile: int
  - Output: Lista updatata continand toate sectiile introduse precendent + sectia nou introdusa,

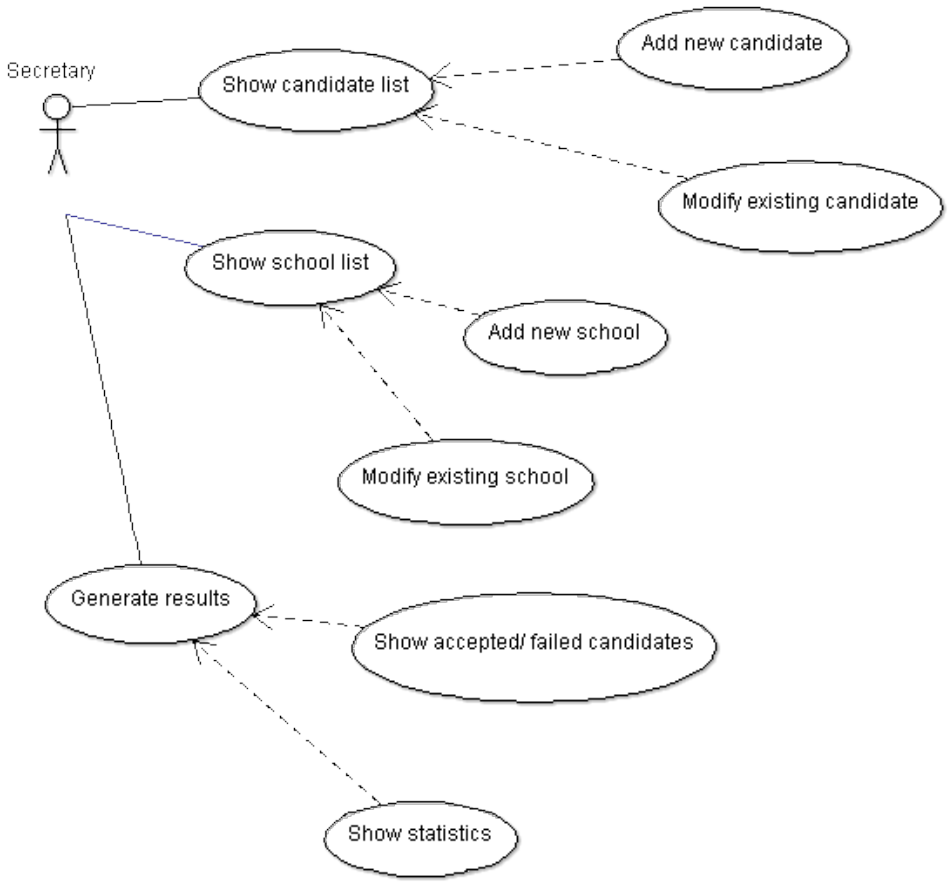
salvata in fisierul "departments.txt"

- Afisarea in interfata grafica a tuturor candidatilor admisi/ respinsi

- Generarea rezultatelor admiterii
  - Input: Lista candidatilor si a sectiilor
  - Output: Lista de candidati avand fiecare asociata o sectie

- Afisarea in interfata grafica a statisticilor privind candidatii
  - Input: Rezultatele admiterii
  - Lista de statistici constand in numarul de candidati in functie de prioritatea optiunii

Use cases



Subalgorithms

-subalg addCandidate
Data: string CNP, string name, string address, float medieBac,float medieMP, int[] options
Result: a new Candidate object is created and added to the Candidate repository
Precond: Candidate with same CNP doesn't already exist (if that is the case, an exception will
be thrown)
Postcond: Candidate with inpuit data exists in the Candidate list

-subalg removeCandidate
Data: string CNP
Result: Candidate having input CNP is removed from the Candidate list
Precond:Candidate with input CNP exists in database, else an exception is thrown
Postcond: Candidate removed from Candidate repository

-subalg addDepartment
Data: int number, string name, int places
Result: a new Department object is added to the department database
Precond: Department with same number doesn't already exist in database
Postcont: Department with input data added to database

-subalg removeDepartment
Data: int number
Result: Department with input number is deleted from the database , if it exists, else an exception is thrown
Precond: Department with same input number already exists in the database
Postcond: Department deleted from department repository

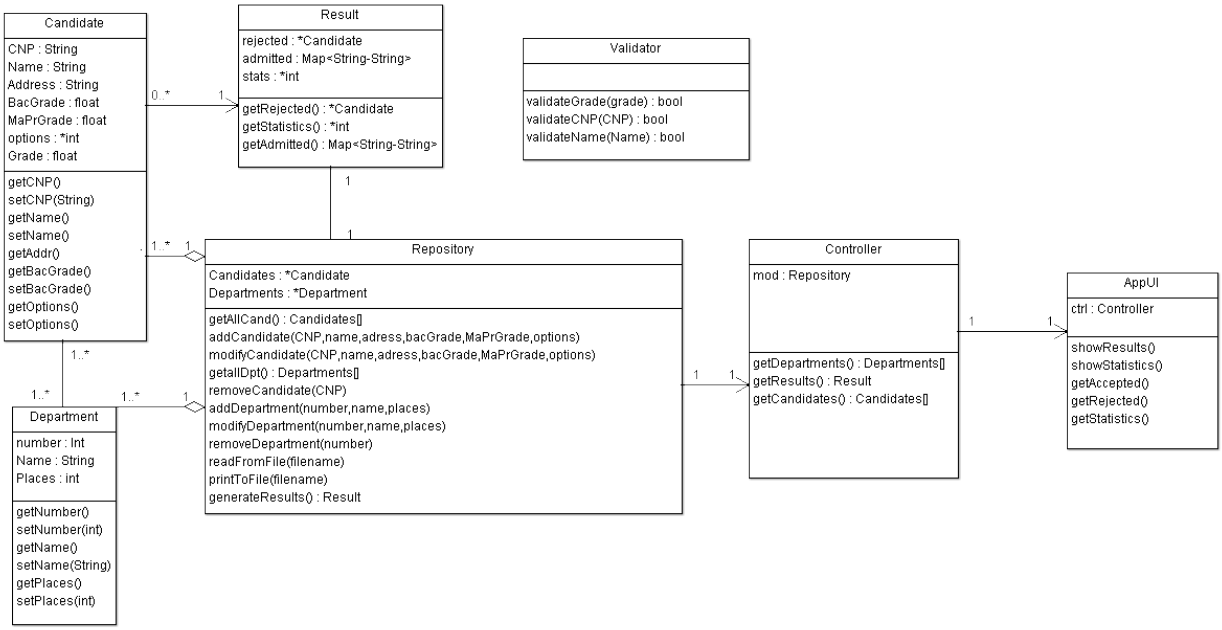
-subalg readFromFile
Data: string filename1, string filename2, name of the files containing candidate and department data
Result: candidate/department repository is populated
Precondition: files filename1 and filename 2 exist in the filesystem, else exception is thrown
Postcondition: repository not empty

-subalg generateResults
Data: existing candidate and department list
Result: Dictionary[Candidate: int] admitted ,a dictionary consisting of candidates and the departments they've been admitted,
List[Canditate} rejected, a list of all the rejected candidates
int[] statistics - an integer list quantifying the number of candidates admitted by the priority of their option
Precondition: candidate and department list contain data for result generation

```
        Postcondition: admitted, rejected and statistics  not empty

void generateResults() {
        //accepted = boolean verifying if the candidated has been accepted yet or not
        //candidates = list of all current candidates
        //admittedTo = list of all  current occupied places at departments, initialised with  0
        //rejected = list of all candidates not accepted to any of the departments, initialised as
                empty
        //accepted = dictionary associating candidates with their final option
        //statistics = int array counting students accepted by priority of option
                int[] admittedTo;
                Candidate[] rejected;
                Dictionary[Candidate,int] accepted;
                int[] statistics;
                bool accepted=false;
                candidates.Sort(); //sort candidates by grade
                foreach(Candidate c in candidates){
                        accepted=false;
                        for int (i=1;i<= candidates.options.length;i++){
        if(admittedTo[candidates.options[i]]<departments[ candidates.options[i]].places){
                statistics[i]++;
                accepted=true;
                admitted.add(c,options[i]);
        }
                if  (accepted==false)
                        rejected.add(c);}}
```

Class diagram



Model