## **PROIECT LA DISCIPLINA PROGRAMARE ORIENTATĂ PE OBIECTE**

**TEMA: Ghid Turistic**

**Abordare orientata obiect**

|  |  |
| --- | --- |
| Titular disciplină:  Șef lucr.dr.ing. Ioniță Irina |  |
|  | **Studenți:**  **Sfâșie Rafael**  **Petcu Stefan**  **Grupa 40317 semigrupa B** |

PLOIEŞTI 2022

**Cuprins**

[**PROIECT LA DISCIPLINA PROGRAMARE ORIENTATĂ PE OBIECTE** 1](#_Toc104404680)

[I.Prezentarea cerinței 3](#_Toc104404681)

[II. Etapele rezolvarii 6](#_Toc104404682)

[III.Concluzie 10](#_Toc104404683)

# I. Prezentarea cerinței

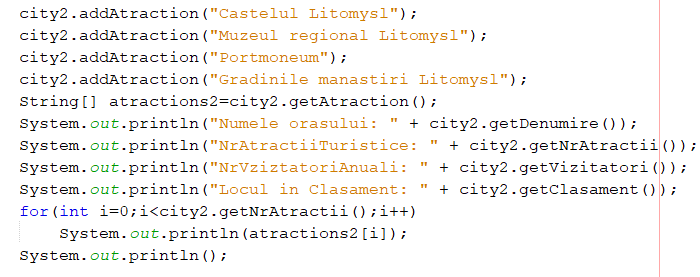
Proiectul prezentat este un Ghid turistic care a fost facut cu ajutorul limbajului de programare Java si al platformei de lucru numita NetBeans IDE 8.2, care este unul din mulțimea de medii de dezvoltare pentru limbajul de Java.

Java este un limbaj de programare apărut in anul 1992 si care a fost proiectat de James Gosling la Sun Microsystems, fiind mai apoi dezvoltat de Oracle. Java este utilizat in multe domenii cum ar fi dezvoltarea aplicațiilor de Android, crearea aplicațiilor bancare, precum si a aplicațiilor de desktop, cum ar fi Gmail, Yahoo sau pachetul Office.

Așa cum am spus si mai sus, proiectul ales este un ghid turistic care are o abordare orientata obiect. Acest lucru înseamnă ca fiecare entitate din proiect este privita ca un obiect din viața reala, cu caracteristici si comportament, aceste caracteristici si comportamente fiind date de tipuri de date si de metode (functii).Am putut sa folosim aceasta caracteristica deoarece Java este un limbaj orientat pe obiecte, acest lucru făcând posibila realizarea proiectului prezentat. De asemenea, in proiectul prezent am mai putut utiliza si doua proprietăți importante in Java, acestea fiind tablourile de obiecte si Mostenirea.

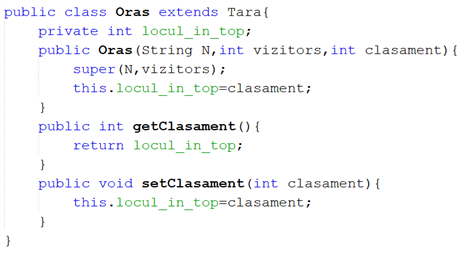
Am ales sa fac acest tip de proiect deoarece eu si colegul cu care am facut acest proiect suntem pasionați de calatorii si de istorie. Tarile care sunt prezentate in proiect sunt Cehia, Irlanda, Suedia, Germania si Austria, aceste tari avand o bogata istorie si consideram ca sunt potrivite pentru proiectul nostru.

Tablourile de obiecte reprezintă o lista de obiecte ce pot fi utilizate pentru a reprezenta mai multe obiecte. Declararea si crearea acestuia este asemanatoare cu crearea unui tablou de tipuri de date primitive.

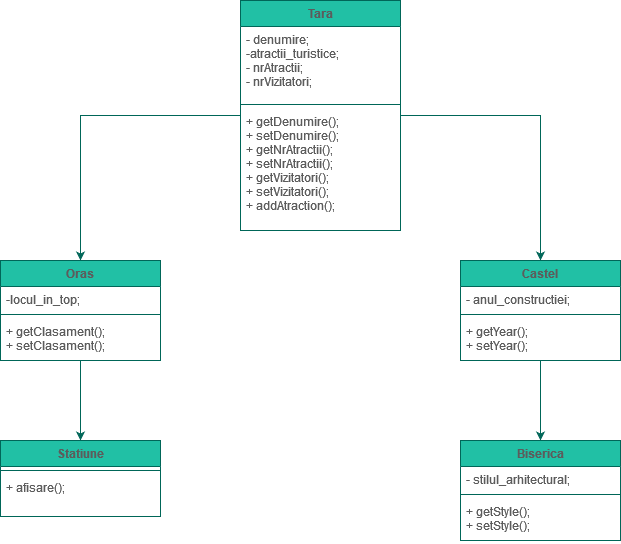


**Crearea unui tablou de obiecte in conformitate cu tema aleasa-Ghid turistic**

Mostenirea reprezintă alta caracteristica a limbajului Java ce a fost utilizata in proiectul prezentat. Aceasta consta din faptul ca dintr-o clasa de baza se pot deriva mi multe subclase ce pot fi utilizate pe parcursul subprogramului, clasele derivate moștenind datele si metodele clasei de baza. In Java doar o singura clasa poate fi clasa de baza,acest lucru însemnând ca o subclasa nu poate avea mai multe clase de baza. In proiectul nostru am utilizat mostenirea cand am creat clasele Oras, Castel, care au fost derivate din clasa de baza “Tara”, cand am creat clasa “Biserica” care este derivate din clasa “Castel” si clasa “Statiune” care este derivata din clasa “Oras”.



**Clasa Oras, derivata din Clasa Tara**

Diagrama UML a claselor

# II. Etapele rezolvării

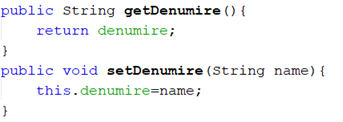
Proiectul nostru este alcătuit din șase clase, acestea sunt Tara, Oras, Castel, Biserica, Statiune. Proiectul a fost realizat pe un laptop care utilizează un procesor AMD Ryzen 7, având o memorie RAM de 16 GB, si utilizând mediul de dezvoltare pentru Java Netbeans IDE 8.2.

Prima data am creat clasa de baza, in cazul nostru aceasta fiind clasa Tara, cu datele membru „denumire”, care este de tipul String (sir de caractere), „atractii\_turistice”, care reprezintă un tablou de obiecte si este de forma String[], acesta atribuire specifica faptul ca este de tipul vector de sir de caractere, după aceea urmeaza „nrAtractiiTuristice” si „nrVizitatori”, care sunt de tipul integer

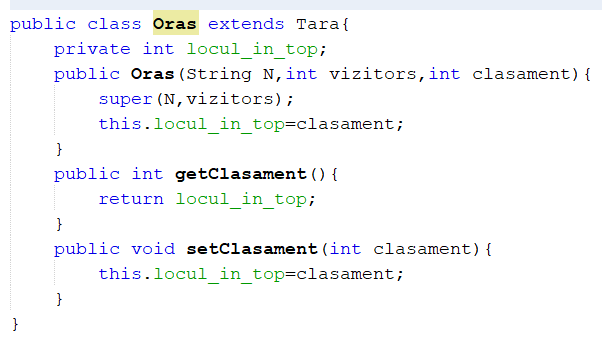
(intregi).

Dupa ce am specificat datele utilizate, vom crea un constructor cu parametri, care are rolul de a crea un obiect de tip Tara. In constructorul, pe care il vom numi Tara, la fel ca clasa in care lucram, deoarece daca punem o alta denumire, programul o sa ne întoarcă o eroare.

După aceea o sa cream metodele care returneaza datele introduse si care setează datele, dându-le o noua denumire (a se vedea imaginea de mai jos).



Metode ce returnează datele introdusesi le seteaza, dându-le astfel o noua denumire

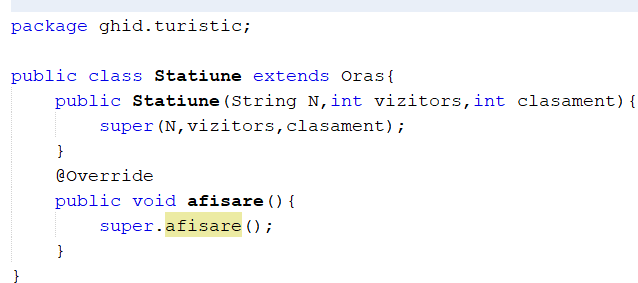
Următoarea etapa in rezolvarea problemei este crearea unor clase derivate care au la baza clasa Tara, acestea fiind clasa Oras soi clasa Castle, care mostenesc datele, constructorii si metodele clasei Tara, dar mai adauga o noua caracteristica, care este proprie (in cazul clasei Oras, care are ca data membru proprie „locul\_in\_top”, care este de tip String).

Clasa Oras derivata din clasa Tara

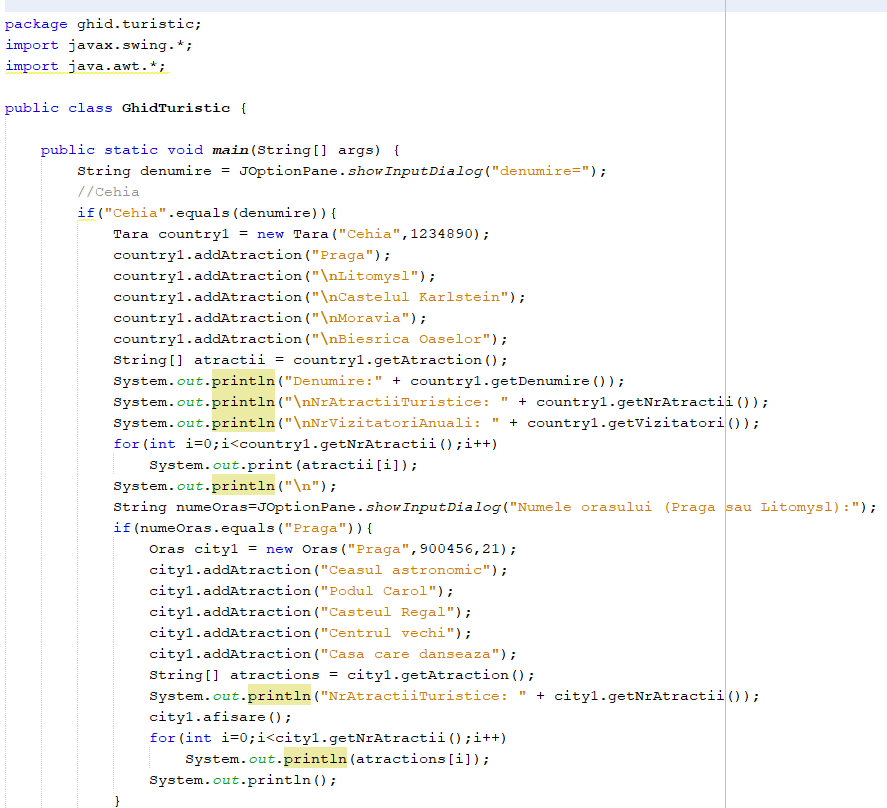
In continuare vom crea clase care sunt derivate din clasele derivate ale superclasei. Aceste clase sunt Clasa Biserica (derivata din clasa Castel) si clasa Statiune, care este derivata din clasa Oras. Aceleași principii care se aplica si la subclasele din superclasa, se aplica si claselor care sunt derivate din subclase.

Clasa Biserica derivata din Clasa Castel

Observam ca in subclasa Statiune nu am mai pus date membru deoarece nu mai era nevoie, deoarece aceasta moștenește datele membru, constructorul cu parametrii si metodele. Clasa Biserica are adăugată unt tip de date membru care este de tipul String si este reprezentata de stilul arhitectural al biserici (gotic, baroc, renascentist etc). De asemenea aceasta mai are si doua metode care setează stilul (setStyle(Stringstyle),”style” reprezentând stilul arhitectural ce va fi dat) si care returneaza stilul (getStyle()).



In final, vom crea funcția principala in care se adauga obiecte create cu clasele prezentate mai sus. Primul Obiect va fi o tara, care are denumirea Cehia si prima data o sa apelam constructorul cu parametrii, avand parametrii prezentati mai sus (denumire, nrVizitatoriAnuali). In acest obiect vom introduce alte obiecte, acestea fiind Oras, Castel, Biserica si Statiune, toate apelând constructorii cu parametrii, singura diferenta la aceste obiecte este ca au o funcție de afișare, care afișează datele din Constructor. La Cehia, am introdus doua orașe, acestea fiind Praga, care este capitala si Litomysl. Aici vom folosi o instructiune ce se gaseste in pachetul „javax.swing.\*”, acesta fiind JOptionPanel.showInputDialog(), pentru introducerea numele tarii pe care dorim sa o vizitam, ca mai apoi sa folosim aceeasi instructiune pentru a introduce de la tastatura numele orasului pe care dorim sa il vizitam in tara respectiva, asa cum am facut si cu obiectul Tara.



Codul in Java pentru vizitarea Cehiei



Rezultatul al alegerii Tarii Cehia si a orasului Litomysl

La fel am procedat cu Irlanda, Suedia, Germania si Austria, singura diferență fiind la Irlanda este ca am adăugat doar un oraș, si anume capitala Dublin.

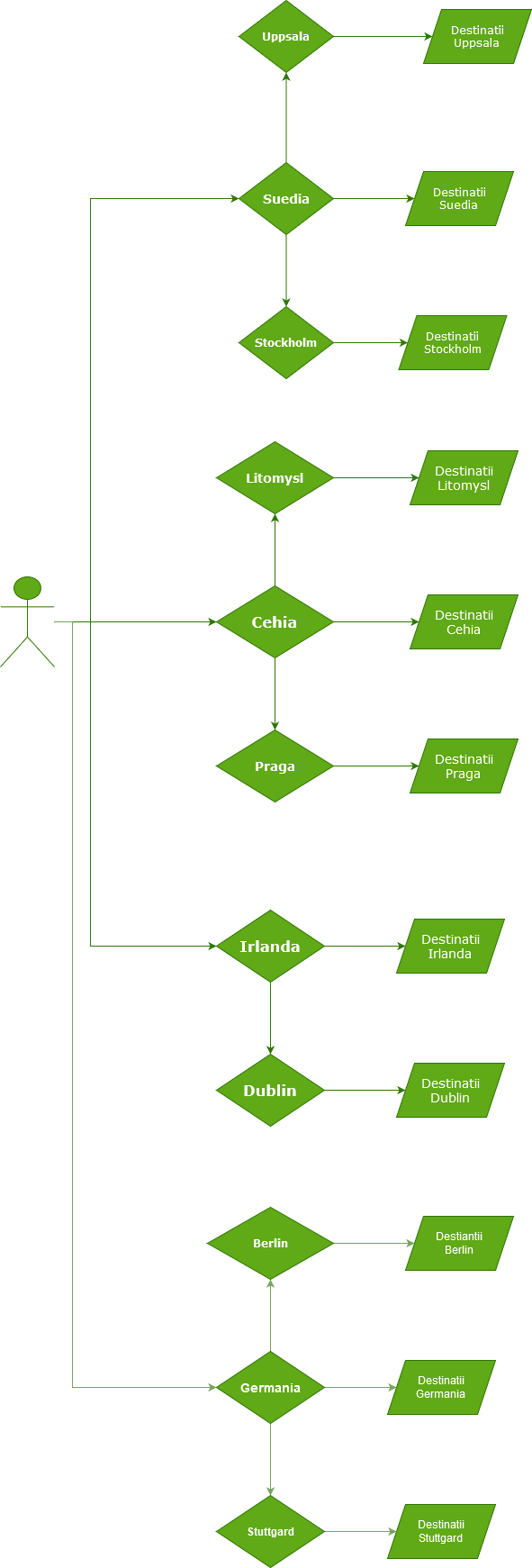


Diagrama UML a cazurilor de utilizare

# III. Concluzie

In concluzie, putem sa spunem ca proiectul nostru, care este un Ghid Turistic, se adresează persoanelor care vor sa aibă o experiența de CityBreak plăcută, utila si care vor sa viziteze țării cu arhitecturi si monumente spectaculoase, dar si țări încărcate de istorie. Clienții noștri pot vizita Orase medievale, castele, biserici si mănăstiri si se pot relaxa in diverse stațiuni de munte. Țările care au fost prezentate in proiect sunt Cehia, Irlanda, Suedia si Germania ,tari care au bogata istorie culturala si arhitecturala.

De asemenea, am făcut acest proiect deoarece amândoi suntem pasionați de istorie si de calatorii, si am decis să imbricam hobby-urile noastre in acest proiect care se adresează persoanelor care au aceleași pasiuni ca si noi.

Consideram ca acest proiect este util categoriilor de persoane enumerate mai sus deoarece este bine documentat si oferă informații precise despre locurile respective.