Nume: Anca-Maria Colăcel

Grupa: 324CC

README

*** tema la POO ***

Consider gradul de dificultate al temei mediu spre dificil, dificultatea aparand la introducerea unor design pattern-uri. Timpul alocat pentru rezolvarea acestei teme a fost de aproximativ o saptamana jumatate. Clasele principale pentru acest proiect sunt: Catalog, User, Student, Parent, Assistant, Teacher, Grade, Group, Course, PartialCourse si FullCourse. Design Pattern-urile folosite sunt: Singleton, Factory, Builder, Observer, Strategy, Visitor si Memento, fiecare cerand introducerea unor clase suplimentare pentru a putea fi implementat. De asemenea, in cea de-a doua parte a temei, interfata grafica, am mai implementat inca 3 clase, cate una pentru fiecare pagina realizata, acestea fiind: PaginaStudent, PaginaProfesor, PaginaParinte. La toate acestea se mai adauga si clasa Test cu main-ul de verificare a functionalitatilor implementate mai sus.

Acestea fiind zise, voi detalia modul de implementare al claselor considerate mai importante dupa cum urmeaza:

---

1. Clasa Catalog
   - aceasta contine o variabila de tip ArrayList cu cursurile din acel catalog, impreuna cu constructor, metode de adaugare si stergere curs din catalog
   - de asemenea, constructorul este privat si este implementata metoda getInstance pentru implementarea design pattern-ului Singleton
   - aceasta clasa este si subiect in cadrul implementarii design pattern-ului Observer, astfel ca contine metodele din cadrul interfetei Subject, acestea fiind: addObserver, removeObserver si notifyObservers

2. Clasa User
   - acessta este o clasa abstracta
   - am implementat si clasa UserFactory cu metoda getUser care creeaza un user in functia de tipul sau care poate fi unul din cele 4 clase care implementeaza clasa abstracta

3. Clasele Student, Parent, Assistant, Teacher
   - acestea implementeaza clasa User
   - clasa Parent este de asemenea observator in cadrul implementarii design-pattern-ului Observer, astfel ca continue metodele din interfata Observer, aceasta fiind update
   - clasele Teacher si Assistant implementeaza interfata Element cu metoda accesot pentru design pattern-ul Visitor

4. Clasa Grade
   - variabilele din aceasta clasa sunt partialScore, examScore si course
   - am implementat metode de get si set pentru aceste variabile

- am implementat metoda getTotal ca fiind media dintre partialScore si examScore
- am implementat metoda compareTo care compara notele finale
- am suprascris metoda toString
- am scris de asemenea si metoda Clone care cloneaza partialScore, examScore, totalScore studentul si numele cursului; aceasta este utila pentru design pattern-ul Memento in realizarea back-up-ului pentru note

5. Clasa Group
   - aceasta continue variabilele asistent si ID pentru fiecare grupa
   - am implementat get si set pentru fiecare si am suprascris metoda toString

6. Clasa Course
   - este o clasa abtracta
   - aceasta continue urmatoarele variabile: nume curs, professor titular, lista de asistenti, notele, un dictionar cu ID-ul grupei si grupa respective si punctele credit pentru curs
   - am implementat get si set pentru fiecare variabila
   - am implementat de asemenea urmatoarele metode:
     - addAsistent - adauga un asistent intr-o grupa pe baza de ID
     - addStudent – adauga un student intr-o grupa pe baza de ID
     - addGroup – adauga o grupa in dictionarul cu grupe
     - getGrade – returneaza nota unui student dat ca parametru prin parcurgerea vectorului de note pana la gasirea notei corspunzatoare studentului cerut
     - addGrade – adauga o nota in multimea de note
     - getAllStudents – creeaza o lista cu toti studentii prin parcurgerea dictionarului cu grupe si luarea fiecarui student din fiecare grupa
     - getAllStudentsGrades – creeaza un dictionar cu studentul si nota fiecaruia prin parcurgerea listei cu toti studentii si obtinerea notei pentru fiecare student
     - getGraduatedStudent – este o metoda abstracta care va fi implementata de cele clase concrete care implementeaza clasa Course; aceasta va intoarce studentii care au promovat cursul tinand cont de anumite conditii

   - am creat o clasa interna CourseBuilder pentru design pattern-ul Builder care contine toate campurile din clasa Course, constructor si metode de get, mai apoi constructorul din clasa Course initializeaza fiecare camp utilizand o variabila de timp Course numita build
   - am creat o clasa interna Snapshot , in cadrul constructorului am clonat notele din fiecare grade din multimea de note; aceasta clasa interna continue metoda de back-up care se foloseste de o variabila backup de tip Snapshot pe care o initializeaza folosind constructorul, apoi metoda undo, toate acestea implementand design pattern-ul Memento
   - am luat de asemenea o variabila de tipul Strategy impreuna cu metodele setStrategie getBestStudent pentru implementarea design pattern-ului Strategy

7. Clasa PartialScore
   - aceasta clasa implementeaza clasa Course
   - continue la fel ca si aceasta o clasa interna FullCourseBuilder care implementeaza clasa abstracta CourseBuilder din Course

- implementeaza de asemenea metoda getGraduatedGrade, iar conditia pentru a identifica studentii care au promovat este ca acestia sa aiba nota totala >=5; am parcurs dictionarul cu situatie studentilor si am comparat notele finale acestora cu 5

8. Clasa FullCourse
   - este similara cu clasa PartialCourse, implementand de asemenea clasa Course si avand si clasa interna FullCourseBuilder
   - implementeaza de asemenea si metoda getAllGraduadedGrade, iar conditia este ca studentii care au promovat sa aiba nota de la partial >=3 si nota de la examen sa fie >=2

9. Clasa ScoreVisitor
   - este o clasa concreta care implementeaza interfata Visitor
   - implementeaza cele 2 metode de visit pentru Teacher si Assistant
   - in cadrul acestor metode am creat cele doua dictionare cu Teacher/Assistant si Tuple format din Student, numele cursului si nota sa la acest curs
   - clasa Tuple este o clasa interna din cadrul acestei clase

10. PaginaStudent
    - aceasta este o clasa care continue codul pentru interfata grafica pentru student
    - am folosit o lista in care am introdus toti studentii
    - la selectarea fiecarui student in cadrul altei liste se vor afisa studentii la care studentul respective este inrolat, iar la alegerea fiecarui curs intr un TextArea se vor afisa informatiile despre cursul respectiv de pilda profesorul titular, asistentii de la acel curs, asistentul acelui student impreuna cu notele sale
    - am folosit de asemenea si cateva label-uri pentru a specifica ce reprezinta fiecare element

11. PaginaProfesor
    - aceasta este o clasa care contine codul pentru interfata grafica pentru profesor
    - am folosit o lista care continue toti profesorii din catalog
    - la selectarea fiecarui profesor se afiseaza o lista cu notele pe care fiecare profesor le are de validat, dupa care exista butonul de validare, iar pentru ca validarea sa fie realizata cu succes trebuie ca in terminal sa se afiseze notificarile pentru fiecare parinte cu notele pentru copiii lor la cursul respectiv
    - exista de asemenea label-uri sugestive

12. PaginaParinte
    - aceasta este o clasa care contine codul pentru interfata grafica pentru parinte
    - am folosit o lista care continue toti parintii
    - la selectarea fiecarui parinte se afiseaza intr-o noua lista notificarile cu notele copilului respective la fiecare materie
    - exista de asemenea label-uri sugestive