==========Tema POO==========
Ionescu Valentin Stefan 322CC
Gradul de dificultate:mediu
Timp alocat:o saptamana

Catalog:
am implementat design patternul Singleton
de fiecare data cand adaug un curs,parcurg fiecare grade din curs si creez un observer
am creat o lista de notificari unde stochez toate notificarile create
functia notifyObservers:
-parcurg fiecare observer si daca studentul din observer corespunde cu studentul din notificarea primita ca parametru, o adaug in lista de notificari

Clasele User, Student, Parent, Assistant, Teacher le am implementat conform patternu lui UserFactory
-Clasele Teacher si Assistant implementeaza interfata Element din Visitor
-Clasele Student si Assistant implementeaza interfata Comparable
-compar primul nume de la fiecare student
-daca primul nume este egal,le compar cel de al doilea nume

Clasa Grade:
toti membrii clasei sunt privati
am implementat pentru fiecare membru getter si setter
pe langa getter si setter am implementat functia clone ce cloneaza o instanta a clasei grade

Clasa Group:
clasa mosteneste colectia ordonata TreeSet de tip Student pentru a i pune in ordine alfabetica
toti membrii clasei sunt privati
am implementat pentru fiecare membru getter si setter

Clasa Course:
toti membrii clasei sunt privati
am implementat pentru fiecare membru getter si setter
am folosit sablonul de proiectare Builder
am implementat toate functiile cerute
functia AddAssistant: verific daca in dictionar se afla o grupa cu ID ul indicat si fara asistent
in caz afirmativ,il setez ca asistent si verific daca se afla in Colectia de asistenti

functia AddStudent: verific daca in dictionar se afla o grupa cu ID ul indicat
in caz afirmativ,verific daca studentul se afla in grupa respectiva si il adaug in caz negativ

functiile addGroup: creez o instanta group daca e cazul si verific daca o grupa cu ID ul indicat exista deja in dictionar
in caz negativ,adaug grupa in dictionar

functia getGrade: parcurg TreeSet ul de tip grade din curs si verific daca numele stundentului se potriveste cu cel dat ca parametru
daca gasesc o potrivire intorc grade ul stundentului

functia addGrade: parcurg TreeSet ul de tip grade din curs si verific daca numele stundentului se potriveste cu cel dat ca parametru
daca nu gasesc o potrivire

functia getAllStudents: parcurg dicitonarul cu fiecare grupa,iar pentru fiecare grupa parcurg studentii si ii adaug in lista cu studenti

functia getAllStudents: : parcurg TreeSet ul de tip grade din curs si adaug fiecare nota cu studentul sau in dictionar

am implementat clasa privata Snapshot care contine un TreeSet de tip grade
am implementat functiile makebackup care copiaza notele din curs in TreeSet ul auxiliar din Snapshot si functia undo care reda notele din Snapshot
am implementat desing patternul Strategy : dau ca parametru colectia de note ale studentilor (TreeSet)
parcurg colectia si returnez grade ul in funtie de strategia aleasa

Clasele PartialCourse si FullCourse
contin clasa interna Builder
returneaza o lista de studenti care au trecut materia(pentru fiecare student verific notele)

Design patternul Observer : clasa Notification primeste ca parametrii numele cursului si nota
functia update: foloseste lista de notificari mentionata anterior
parcurg lista pana gasesc o potrivire intre studentul si cursul dat ca parametru si studentul si cursul din lista
in caz afirmativ,sterg notificarea veche din lista si adaug notificarea noua
clasa Catalog implementeaza interfata Subject

Desing patternul Strategy : dau ca parametru colectia de note ale studentilor (TreeSet)
parcurg colectia si returnez grade ul in funtie de strategia aleasa

Desgin patternu Visitor : in clasa ScoreVisitor am implementat urmatoarele functii:
functia add_grade_teacher adaug in dictionarul examScores numele profesorului si lista cu notele pe care acesta urmeaza sa le valideze
functia add_grade_assistant face acelasi lucru dar pentru asistent
functiile visit :adaug in Colectia oficiala de note a cursului fiecare nota din dictionare
apelez functia notifyObservers pentru fiecare nota adaugata
dupa ce am adaugat notele in catalog ,sterge notele adaugate din dictionare

Desgin patternul Memento:
am implementat clasa privata Snapshot care contine un TreeSet de tip grade
am implementat functiile makebackup care copiaza notele din curs in TreeSet ul auxiliar din Snapshot si functia undo care reda notele din Snapshot
am implementat desing patternul Strategy : dau ca parametru colectia de note ale studentilor (TreeSet)
parcurg colectia si returnez grade ul in funtie de strategia aleasa

Interfata Grafica:
pentru autentificare: username ul este primul nume ,iar parola este al doilea nume
am implementat tot ce se cerea in enunt,bonusuri si cateva chestii in plus
am creat o pagina principala unde un User se poate autentifica sau poate sa vada situatia Cursurilor

Pagina Student:
studentul poate vedea fiecare curs la care este inscris si se pot afisa informatii suplimentare despre un anumit curs daca acesta este selectat
am adaugat un buton de "logout" in care se revine la pagina principala

Pagina Teacher si Assistant:

Fiecare poate vedea la ce cursuri preda,poate valida notele
am adaugat un buton de "logout" in care se revine la pagina principala
daca notele au fost validate si userul revine pe cont,acesta nu o sa mai aiba nimic de validat

Pagina Parent:
am adaugat un buton de "check" in care afisez notificarile primite(fiecare parinte vede situatia scolara a copilului sau (notele la fiecare materie))
am adaugat un buton de "logout" in care se revine la pagina principala

Pagina Cursinfo:
Pagina afiseaza in detaliu toate datele depre un curs,lista de studenti,asistenti,note,precum si cel mai bun elev de la curs
poate aduga un student nou la o grupa existenta
poate adauga un asistent la o noua grupa
am adaugat un buton in care pot face backup la note si un buton care face undo la note

!!!!Testarea am facut o cu testul oferit,parsandu l cu Json simple !!!!!