Cerințe obligatorii

1. Pattern-urile implementate trebuie sa respecte definiția din GoF discutată în cadrul cursurilor și laboratoarelor. Nu sunt acceptate variații sau implementări incomplete.
2. Pattern-ul trebuie implementat în totalitate corect pentru a fi luat în calcul.
3. Soluția nu conține erori de compilare.
4. Pattern-urile pot fi tratate distinct sau pot fi implementate pe același set de clase.
5. Implementările care nu au legătura funcțională cu cerințele din subiect NU vor fi luate în calcul (preluare unui exemplu din alte surse nu va fi punctată).
6. NU este permisă modificare claselor primite.
7. Soluțiile vor fi verificate încrucișat folosind MOSS. Nu este permisă partajarea de cod între studenți. Soluțiile care au un grad de similitudine mai mare de 30% vor fi anulate.

Cerințe Clean Code obligatorii (soluția este depunctata cu câte 2 puncte pentru fiecare cerința ce nu este respectată) - maxim se pot pierde 8 puncte

1. Pentru denumirea claselor, funcțiilor, testelor unitare, atributelor și a variabilelor se respecta convenția de nume de tip Java Mix CamelCase;
2. Pattern-urile și clasa ce conține metoda main() sunt definite in pachete distincte ce au forma *cts.nume.prenume.gNrGrupa.denumire_pattern*, *cts.nume.prenume.gNrGrupa.main* (studenții din anul suplimentar trec “as” în loc de gNrGrupa);
3. Clasele și metodele sunt implementate respectând principiile KISS, DRY și SOLID (atenție la DIP);
4. Denumirile de clase, metode, variabile, precum și mesajele afișate la consola trebuie sa aibă legătura cu subiectul primit (nu sunt acceptate denumiri generice). Funcțional, metodele vor afișa mesaje la consola care sa simuleze acțiunea cerută sau vor implementa prelucrări simple.

Se dezvoltă o aplicație software destinată unui magazin online.

5p. În cadrul unui magazin online, calculul totalului de plata pentru cosul de cumparaturi al unui client inclusiv transportul se realizeaza la sfarsit, in functie de totalul cosului de cumparaturi si de adresa clientului. Implementarea este disponibila in clasa Magazin. Ca o masura de promovare a magazinului, departamentul de market doreste adaugarea unei noi optiuni de discount de 10% aplicat doar o singura data la intreaga prima comanda pentru fiecare client in parte. Aceasta optiune nu trebuie sa implice modificari in codul existent. Sa se implementeze un modul intermediar care sa gestioneze noul context

3p. Pattern-ul este testat în main() prin utilizarea a cel putin 2 clienti care sa lanseze minim 3 comenzi prin intermediul noii implementari. Sa se evidentieze comenzile care au primit discountul respectiv.

1p. Sa se motiveze in scris alegerea design pattern-ului care modeleaza cel mai bine situatia enuntata. Motivarea trebuie sa contina elementele cheie care denota utilizarea respectivului design pattern.

5p. Se doreste implementarea unui modul care sa ofere posibilitatea clientilor de a vizualiza produse din cadrul magazinului online intr-o maniera arborescenta. Se definesc minim doua niveluri de agregare a produselor (categorii, subcategorii, etc). Implementarea va tine cont de interfata IProdus. La vizualizarea produselor, la nivel de categorii se afiseaza denumirea precum si numarul total de produse din respectiva categorie iar la nivel de produs, denumirea si numarul de bucati existente in stoc.

3p. Pattern-ul este testat în main() prin definirea și utilizarea unei structuri arborescente care să conțină minim 3 niveluri cu minim 6 elemente. Se dorește afișarea tuturor produselor separate pe categorii din cadrul magazinului online.

1p. Sa se motiveze in scris alegerea design pattern-ului care modeleaza cel mai bine situatia enuntata. Motivarea trebuie sa contina elementele cheie care denota utilizarea respectivului design pattern.