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 corect 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 1 punct pentru fiecare cerința ce nu este respectată) - maxim se pot pierde 4 puncte

1. Pentru denumirea claselor, funcțiilor, testelor unitare, atributelor și a variabilelor se respectă convenția de nume de tip Java Mix CamelCase;
2. Pattern-urile și clasa ce conține metoda main() sunt definite în 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 legătură 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 pentru promovarea unor spectacole. Spectacolele pot fi de teatru, concert sau stand-up comedy

4p. Aplicația implementată trebuie să permită gestionarea detaliilor pentru fiecare tip de spectacol în parte. Pentru orice tip de spectacol trebuie definite: numele (titlul), locația și data și ora desfășurării. Fiecare tip de spectacol are anumite detalii specifice (de asemenea, obligatorii), cum ar fi: teatru - regizor, actori; concert - genul de muzică, lista de artiști; stand-up comedy - limbaj licențios?, lista comedianților. Toate clasele aferente tipurilor de spectacol vor implementa interfața **IShow**.

```
public interface IShow {
    public void printDetails();
    public String getName();
    public String getLocation();
    public String getDateTime();
}
```

1p. În main() trebuie să se implementeze o secvență de cod relevantă pentru evidențierea design pattern-ului implementat.

Promovarea spectacolelor se face prin intermediul afișelor.

4p. Aplicația implementată va asigura generarea afișelor de promovare, după cum este descris în continuare. P**entru fiecare tip de spectacol** se va genera o imagine (logo) reprezentativă care va apărea pe toate afișele spectacolelor de tipul respectiv (ex.: toate spectacolele de tip teatru vor avea aceeași imagine). Imaginea este una complexă, generarea ei durează destul de mult, de aceea implementarea trebuie să asigure optimizarea procesului de creare a afișelor. Afișele vor conține, bineînțeles, și detaliile spectacolului promovat. Implementarea acestei componente de generare a afișelor va fi realizată prin implementarea interfeței **IPoster.**

```
public interface IPoster {
   public void print();
}
```

1p. În main() trebuie să se implementeze o secvență de cod relevantă pentru evidențierea design pattern-ului implementat.