Observații

1. Implementările de Design Patterns trebuie să fie conform definiției GoF discutată în cadrul cursului și laboratoarelor. Implementările incomplete sau variațiile non-GoF ale patternurilor nu sunt acceptate și nu vor fi punctate.
2. Sunt luate în considerare doar implementările complete și corecte, implementările parțiale nu se punctează.
3. Soluțiile ce conțin erori de compilare nu vor fi evaluate.
4. Patternurile pot fi implementate separat sau utilizând același set de clase.
5. Implementările generale de design patterns, care nu sunt adaptate cerinței și nu rezolvă problema cerută nu vor fi luate în considerare.
6. Clasele/interfețele primite nu pot fi modificate.
7. Soluțiile vor fi verificate cu software antiplagiat. Partajarea de cod sursă între studenți nu este permisă. Soluțiile cu un nivel de similitudine peste 30% vor fi anulate.
8. Se acordă 1 punct din oficiu.

Cerințe Clean Code (nerespectarea lor va duce la depunctarea cu 2 puncte pentru fiecare cerință) - se pot pierde 8 puncte în total

1. Clasele, funcțiile, atributele și variabilele vor fi denumite conform convenției Java Mix CamelCase.
2. Fiecare pattern precum și clasa ce conține metoda `main()` vor fi definite în pachete diferite de forma `cts.nume.prenume.g<numarul_grupei>.<denumire_pattern>`, respectiv `cts.nume.prenume.g<numarul_grupei>.main` (studenții de la recuperare vor utiliza „recuperare” în loc de numărul grupei).
3. Clasele și metodele nu trebuie să încalce principiile KISS, DRY, YAGNI sau SOLID.
4. Numele claselor, metodelor, variabilelor și mesajele afișate la consolă trebuie să fie strict legate de subiectul curent (numele generice nu sunt acceptate). Mesajele afișate trebuie să simuleze scenariul cerut.
5. Nu sunt permise „magic numbers” sau valori hardcodate. Formatarea codului sursă trebuie să fie cea standard.

Realizați o aplicație software pentru o spălătorie auto automată.

3 p. Programul de spălare ales este dat de interfața `AbstractAutoWashOptions.` Indiferent de opțiunile selectate mașina va intra în tunelul de spălare, va trece prin zona de prespălare, apoi prin zona de perii, apoi prin zona de ceruire și polish și la final prin zona de uscare. Utilizați un design pattern ce va permite modelarea ordinii fazelor de spălare și va aplica respectiva fază de spălare dacă utilizatorul a selectat-o în programul său (prin intermediul clasei din familia `AbstractAutoWashOptions`).
1.5 p. Testați implementarea prin crearea a 3 programe de spălare diferite ce conțin diverse combinații de opțiuni (faze de spălare). Demonstrați faptul că de fiecare mașina respectă aceeași ordine a fazelor de spălare, dar doar opțiunile selectate sunt aplicate.

3 p. Aplicația aferentă spălătoriei auto rulează în permanență și nu poate fi oprită spre a fi recompilată. Având în vedere faptul că un program de spălare este derivat din interfața `AbstractAutoWashOptions,` utilizați un design pattern ce permite firmei să adauge posibilitatea de a avea opțiuni adiționale de spălare (spălare cu perii fine, spălare șasiu, spălare cu apă dedurizată, etc.)
1.5 p. Testați în main implementarea prin crearea a minim 3 instanțe de astfel de clase demonstrând faptul că opțiunile adiționale pot fi selectate în orice combinație.