**DOCUMENTATIE PROIECT CTS**

Descrierea ideei proiectului:

Intr-un ParcAuto au loc zilnic transferuri de vehicule, insemnand achizitionarea de catre parc de noi vehicule,

respectiv vanzarea acestora.

Design Patter-uri implementate:

Acesta fiind singurul parc auto din oras, am putut implementa design patternului Singleton.

Este folosita clasa ParcAuto.

Prin design patternul FactoryMethod se defineste un anumit vehicul.

Am creat o enumeratie care definte tipul vehicului, si anume: autoturism, autobuz, motocicleta.

Am folosit clasele Vehicul, Autobuz, Autoturism, Motocicleta, VehiculFactory si interfata IComerciabil.

Am creat design patternul Builder pentru detalii referitoare la permisul auto al soferilor. Astfel, aflam informatii

despre categoria pe care o au, de cand timp detin carnetul si daca in momentul respectiv au facut vreun accident.

Am folosit clasele PermisAuto si PermisAutoBuilder care implementeaza interfata IPermisAutoBuilder.

Inainte de achizitionarea unei masini, cumparatorul doreste mai intai sa o testeze, iar apoi sa negocieze pretul pentru aceasta.

La randul lui, cumparatorul este interesat de pretul masinii, de marca acesteia, precum si de kilometrajul pe care il are la bord.

Astfel este implementat design patternul Facade care apeleaza metodele necesare pentru a cumpara masina.

Sunt folosite clasele Facade, CumparareVehicul, Cumparator.

Avand in vedere faptul ca parcul auto este unul construit acum mult timp si cunoscut de catre multi locatari, avem si posibilitatea

de a avea anumiti abonati(clienti fideli). Astfel, acesti clienti ai nostri vor fi informati in legatura cu masinile noi pe care le avem la

dispozitie, in functie de marca acesteia. Pentru aceasta functionalitate este implementat design patternul Observer.

Sunt folosite clasele Client, ParcAuto si interfetele Subject si Observer.

Parcul auto de asemenea permite vanzarea de pachete promotionale, ce pot fi constituite din mai multe vehicule

sau din alte pachete promotionale.

Teste:

AllTests imi ruleaza toate testele.

TestCasePachetPromotional verifica daca in momentul in care adaug 0 si nu un pachet promotional,

metoda returneaza false(verific sa nu mearga.

TestCaseProprietatiVehicul face testul astfel incat in momentul in care este inserat un an al fabricatiei negativ pentru un vehicului,

sa arunce o excepties. Asemanator in cazul in care introduc anul fabricatiei mai mare decat anul curent.

TestCaseSingleton testeaza daca se creeaza corect un obiect ParcaAuto, de tipul Singleton.

TestCaseVehicul verifica daca sunt introdusi corect parametrii pentru constructorul clasei Vehicul.

Mockito verifica daca imi adauga un pachet promotional.

TestRunner testeaza JUnit, insa in consola.

TestSuite aduna mai multe TestCase-uri si le ruleaza.