Test - Calitate si testare software

Calitate si testare Software (Academia de Studii Economice din Bucure□ti)

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>.pattern.<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.

3p. Un senzor de calitate a aerului inteligent va trimite în fiecare dimineață un raport privind calitatea aerului estimată pentru ziua respectivă. Clasa aferentă raportului implementează interfața `AbstractAirQualityReport`. Tipul de raport este influențat de valoarea nivelului de particule fine (PM 2.5) estimată de modulul de machine learning extern senzorului (va proveni din exterior) astfel: dacă valoarea este cuprinsă în intervalul 0-30 se va genera un raport de aer curat; dacă valoarea este cuprinsă în intervalul 31-70 se va genera un raport de avertizare; dacă valoarea depășește 71 se va genera un raport de risc asupra sănătății. Utilizați un design pattern ce permite crearea raportului corespunzător valorii PM 2.5 primită de senzor.

1.5p. Testați implementarea prin crearea a 3 rapoarte diferite pentru trei valori diferite ale PM 2.5 primite de senzor (câte o valoare pentru fiecare interval).

3p. Un senzor de calitate a aerului oferă o plajă largă de masurători în funcție de model precum: nivel PM 2.5, nivel PM10, nivel VOC, nivel CO2, nivel CO, temperatură, umiditate, etc. Clasa aferentă unui senzor este derivată din interfața `AbstractAirQualitySensor`. Aplicația permite crearea de senzori ce dispun de toate capabilitățile sau doar o parte din ele. Un senzor odată creat nu mai poate fi modificat. În plus, se dorește ca procedeul de creare a unui senzor să poată utiliza o singură linie de cod. Implementați un design pattern ce rezolvă situația descrisă.

1.5p. Testați în main implementarea prin crearea a minim 3 senzori ce conțin diverse combinații de capabilități. Demonstrați faptul că fiecare dintre cei 3 senzori poate fi creat într-o singură linie de cod și că instrucțiunile de creare sunt independente (faptul că un senzor are o anumită capabilitate nu va influența capabilitățile senzorilor ce vor fi creați în viitor).