Fiind dat codul din clasele *CTSMath* și *Pantof*, să se implementeze un sistem de testare, bazat pe JUnit, care să verifice că metodele rulează și generează rezultate pentru seturi de date de intrare diferite. În caz contrar, metodele vor fi corectate.

**10p.** Să se implementeze design pattern-ul Memento în scopul salvării și restaurării versiunilor de contracte (clasa *VersiuneContract*) astfel încât un client să aibă posibilitatea de a restaura starea comenzii actuale (clasa *Comanda*) cu starea anterior salvată. Design pattern-ul se testează în *main().*

**10p.** Să se scrie minim 4 unit teste (**1.25p** per unit test) pentru*constructor Pantof, respectiv nrCifre()*,gestionate în cazuri de test diferite, dintre care obligatoriu:

* Un unit test pentru verificarea condiției *Right*
* Un unit test pentru verificarea unei condiții pentru care funcția testată generează *excepție*
* Un unit test pentru verificarea *Existence*
* Un unit test pentru verificare altei condiții decât cele anterioare, la alegere

**2.5p.** Să se implementeze pentru metoda *setCuloare()* din clasa *Pantof* un unit test pentru verificare de tip *inverse*.

**2.5p.** Să se implementeze pentru metoda *nrCifre()* din clasa *CTSMath* un unit test pentru verificare de tip *cross-check*.

**7.5p.** Să se implementeze o suită de teste care să conțină toate metodele dintr-un caz de test (3p) și anumite metode din al 2-lea caz de test (la alegere, nu toate) (4.5p).

**7.5p.** Dacă în clasa *CTSMath* se adaugă metoda *getPretProdus(),* să se implementeze un *mock object* pentru interfața *IProdus* care să permită testarea metodei în lipsa unui obiect de tip produs real.

\*codul sursa scris trebuie sa respecte principiile Clean Code cu privire la denumirile variabilelor si metodelor, aranjarea codului si complexitatea instrucțiunilor; nerespectarea acestor reguli duce la depunctarea soluției cu 10p

\*\*sunt luate in considerare doar soluțiile care trec de compilare fără erori