GRILE CTS

1. Termenul de “zombie code”se refera la:
   a. Codul sursa ce contine multe comentarii de tip TODO
   b. Legacy code(cod sursa vechi)
   c. Codul sursa ce contine bucati de cod comentate
   d. Codul sursa folosit
2. O modalitate de a evita update-uri (noi versiuni) dese pentru aplicatia dezvoltata este:
   a. Mutarea constantelor din cod in baza de date
   b. Scrierea de metode scurte
   c. Folosirea de enumeratii in detrimental sirurilor de caractere
   d. Stabilirea unor date fixe la care se lanseaza update-uri
3. Proiectarea pe baza de contract (Design by contract) este amintita de urmatorul pricipiu:
   a. Liskov Substitution
   b. Interface segregation
   c. Single responsibility
   d. Open closed
4. Litera “I” din principiul SOLID provine de la:
   a. Iskov Substitution
   b. Inversion Dependency
   c. Interface Dependency
   d. Interface Segregation
5. Sistemul Hungarion Notation face referire la:
   a. Scrierea denumirilor de clase cu litere capital
   b. Folosirea tipului de data in denumire a variabilei
   c. Folosirea underscore in denumire a variabilelor
   d. Folosirea vizibilitatii in denumire a atributelor
6. Copierea si alipirea unei bucati de cod(Copy/Paste) duce la incalcarea urmatorului principiu:
   a. YAGNI
   b. SOLID
   c. KISS
   d. DRY
7. Urmatorea secventa de cod incalca principiul:

```
Class Prelucrari{
        Public void afiseazaUtilizatori(){
        //….}
        Public void prelucrareDate(){
        //…}
        Pulib void salveazaConstanteleInDb(){
        //..}
};
```

a. Interface segregation
b. Liskov substitution
c. Open-closed
d. Single responsibility

8. Urmatoarea secventa de cod face referire la:

```
Class View extends…{
        Public View(DataService _dataService, LoginService _loginService){
                _dataService = super.dataService;
                _loginService = super.loginService;
        }
};
```

a. Liskov substitution
b. Interface segregation
c. KISS
d. Dependency inversion

9. Urmatorea ierarhie de clase aplica principiul:

```
Interface Redimensionabil{
//..}
Class EcranMare implements Redimensionabil{
//..}
Class EcranMic implements Redimensionabil{
//..}
```

a. KISS
b. Interface segregation
c. Single responsibility
d. Design by contract

10. Urmatoarea secventa de cod incalca principiul:

```
Boolean denumireUtilizatorOk = false;
Boolean parolaUtilizatorOk = false;
if(this.denumireUtilizator != null &&
          !this.denumireUtilizator.equals("")) {
     denumireUtilizatorOk = true;
}
if(this.parolaUtilizator != null &&
          !this.parolaUtilizator.equals("")) {
     parolaUtilizatorOk = true;
}
if(denumireUtilizatorOk && parolaUtilizatorOk) {
     return true;
}
else {
     return false;
}
```

a. KISS
b. YAGNI

c. DRY
d. SOLID

11. Care dintre urmatoarele comenzi Git este folosita pentru a copia modificarile aflate in Repository-ul aflat pe server pe masina de lucru
    a. $ git commit
    b. $ git pull
    c. $ git push
    d. $ git add
12. Care dintre urmatorele comenzi Git este folosita pentru a crea o copie locala a proiectului de pe Repo-ul principal
    a. $ git init
    b. $ git clone
    c. $ git pull
    d. $ git add
    e. $ git push
13. Care dintre urmatorele comenzi Git este folosita pentru a salva modificarile dintr-o sesiune de lucru
    a. $ git branch
    b. $ git commit
    c. $ git init
    d. $ git status
    e. $ git pull
14. Care dintreurmatoarele comenzi Git este folosita pentru a trece pe branch-ul numit Proiect_CTS
    a. $ git branch
    b. $ git merge proiect_CTS
    c. $ git checkout –b proiect_CTS
    d. $ git checkout proiect_CTS
15. Care dintre urmatoarele comenzi Git determina starea curenta a proiectului
    a. $ git init
    b. $ git log
    c. $ git add
    d. $ git status
16. Se considera urmatorul scenariu:

    “ACME Inc. dezvolta o solutie software pt un restaurant, a.i. chelnerul sa poata prelua comenzile direct de pe telefonul mobil. Comenzile sunt preluate de la client si ele sunt create pe loc, fiind automat alocat bucatrul specializat pe acel fel de mancare, ingredientele folosite si alte cerinte special ale clientului. Aceste detalii sunt puse de aplicatie, fara a fi necesar interventia chelnerului care doar selecteaza felul de mancare solicitat. Comenzile sunt trimise bucatarilor la finalizarea comenzii pentru masa respective, urmand sa fie executate in functie de gradul de incarcare al fiecarui bucatar.”

    Ce pattern ofera solutie pentru aceasta problema?

    a. Command
    b. Chain of Responsibility
    c. Memento
    d. State
17. Care dintre urmatorele concepte nu este o componenta obligatorie utilizata in definirea unui design pattern:
    a. Avantajele si dezavantajele oferite de pattern
    b. Problema pentru care pattern-ul ofera solutie
    c. Numele pattern-ului
    d. Implementarea pattern-ului in java
    e. Solutia oferita de pattern descrisa prin diagrame sau pseudo-cod
18. Un design pattern reprezinta:
    a. O solutie la o problema comuna in POO
    b. Un algoritm utilizat in poo
    c. O structura de date utilizata in poo
    d. O schema pt un tip particular de clasa
19. Care dintre urmatoarele GoG design patterns este de tip Comportamental:
    a. Decorator
    b. Strategy
    c. Builder
    d. Singleton
20. Ce GoF pattern trebuie implementat daca se doreste implementarea unei solutii care sa permita alegerea la run-time a agloritmului/functiei necesare procesarii unui set de date. Solutia trebuie sa permita modificarea librariei de functii insa nu a clasei ce gestioneaza datele.
    a. Wrapper
    b. Façade
    c. Decorator
    d. Strategy
21. Care combinatie reprezinta tipuri corecte de design pattern-uri?
    a. Creationale, Comportamentale, Structurale
    b. Creationale, Mediatoare, Adaptoare
    c. Creationale, Consumatoare, Compozite
    d. Mediatoare, Structurale, Creationale
22. Ce pattern permite extinderea functionalitatii unui obiect, in mod dynamic, la run-time?
    a. Façade
    b. Adapter
    c. Composite
    d. Decorator
23. Memento este un design pattern de tip:
    a. Mediator
    b. Comportamental
    c. Nu este design pattern

d. Creational
e. Adaptor
f. Structural

24. Incerci sa adaugi in Solutia ta o clasa scrisa in alt proiect pentru a servi diferiti clienti. Toate celelalte clase au aceeasi interfata, insa clasa adaugata are o interfata cu totul diferita de ceea ce asteapta clientii. Cu toate acestea contine toate functionalitatile necesare. Ce fel de refactorizare este necesara pt a face aceasta clasa potrivita in sistemul tau, cu minim de effort?
    a. Se aplica pattern-ul proxy
    b. Se aplica pattern-ul builder
    c. Se aplica patter-ul adapter
    d. Se defineste o noua clasa care implementeaza interfata asteptata si se copiaza prin copy&paste din ace clasa in aceasta
    e. Se aplica pattern-ul façade
25. Daca se doreste implementarea unei solutii in care toti clientii ce folosesc clasa A sa aiba acces la aceeasi instanta de tip A, ce GoF pattern se va folosi:
    a. NU este nevoie de pattern deoarce Solutia se implementeaza marcand clasa A ca fiind final
    b. Se implementeaza singleton pentru clasa A
    c. NU este nevoie de pattern deoarece Solutia implementeaza definind clasa A ca fiind abstracta
    d. Se implementeaza memento pentru clasa A
26. Alege afirmatia incorecta cu privire la TDD:
    a. Pasii din TDD sunt: Scrie si ruleaza teste, Corecteaza metoda, Refactorizeaza
    b. In situatia in care testul genereaza fails, testul se corecteaza
    c. Se bazeaza pe repetitia unui ciclu de dezvoltare simplu
    d. TDD descrie conceptul de Test Driven Development
27. Nu este principiu de tip Correct boundary conditions specific analizei datelor folosite in Unit Testing:
    a. Valoare are tipul cerut de tipul variabilei
    b. Codul refera componente externe care nu sunt controlate direct
    c. Valorea are formatul corect
    d. Valoarea este intre limitele(maxim si minim) acceptate
    e. Setul de valori trebuie sa fie ordonat sau nu
    f. Valoarea exista
28. Se considera urmatorul UniTest. Ce se afiseaza in urma executiei acestuia?

```
public class Test {

@BeforeClass
public void setUpBeforeClass() { System.out.print("Before ");}

@Test
public void test1() { System.out.print("Test1 ");}

@Test
public void test2() { System.out.print("Test2 ");}

@After
public void tearDown() { System.out.print("After ");}
}
```

a. Before Test1 Test2 After
b. Before Test1 After Test2 After
c. Before Test1 After Before Test2 After
d. Before Test1 Before Test2 After

29. Alegeti afirmatia corecta privind Unit Testing-ul
    a. Un unit test este o secventa de cod scris de un programator pentru a evalua o clasa sau o metoda
    b. Are loc in faza de dezvoltare si este un instrument destinat programatorilor
    c. Toate afirmatiile enumerate sunt corecte
    d. Un unit test evaluaza modul de functionare al unei metode intr-un context bine definit
    e. Metoda simpla si rapida de testare a codului sursa de catre programatori
30. Se considera urmatorul UnitTest. Ce se afiseaza in urma executiei acestuia?

```
public class Test {
@Before
public void setUp () { System.out.print("Before ");}

@Test
public void test1() { System.out.print("Test1 ");}

@Test
public void test2() { System.out.print("Test2 ");}

@AfterClass
public void tearDownAfterClass() { System.out.print("After ");}
}
```

    a. Before Test1 Before Test2 After
    b. Before Test1 Test2 After
    c. Before Test1 After Before Test2 After
    d. Before Test1 After Test2 After
31. Care dintre urmatorele afirmatii despre Git nu este adevarata:
    a. Nu exista o autoritate central de control-toti au aceleasi drepturi
    b. Este un sistem distribuit pentru controlul versiunilor
    c. Toti utilizatorii lucreaza pe masinile de dezvoltare in mod offline fara a conecta in permanenta Repository-ul
    d. Este un sistem centralizat pentru controlul versiunilor
    e. Toti utilizatorii au acces la istoricul modificarilor
32. Principiul Hollywood(“Don’t call us, we’ll call you”) a inspirit urmatorul principiu SOLID:

a. Single reponsability
b. Interface segregation
c. Liskov substitution
d. Dependency inversion

33. Alegeti afirmatia care nu reprezinta o caracteristica a Unit Testing-ului
    a. Are loc in faza de dezvoltare si este un instrument destinat programatorilor
    b. Un unit test este o secventa de cod scrisa de un programator pentru a evalua o clasa sau metoda
    c. Metoda simpla si rapida de testare a codului sursa de catre programatori
    d. Toate afirmatiile enumerate sunt corecte
    e. Un unit test evalueaza modul de functionare al unei metode intr-un context bine definit
34. O clasa de tipul “AlteOperatii” ce contine operatii ce nu au fost grupate in clasa incalca:
    a. Interface segregation
    b. Open closed
    c. Single responsibility
    d. Liskov substitution
35. Principiul YAGNI se refera la:
    a. Crearea de constructori cu parametri in toate clasele unde exista constructor fara parametri
    b. Scrierea apriorica a tuturor claselor, atributelor si metodelor ce vor fi eventual necesare
    c. Scrierea apriorica doar a interfetelor ce vor fi necesare
    d. Scrierea tuturor claselor, atributelor si metodelor ce sunt absolute necesare
36. Daca se doreste implementarea unei solutii in care obiectele complexe sunt construite printr-un mecanism independent de procesul de realizare efectiva a obiectelor a.i. clientul sa nu cunoasca detaliile interne ale obiectului, cel mai potrivit GoF pattern este:
    a. Singleton
    b. Adapter
    c. Builder
    d. Decorator
    e. Factory
37. Care dintre urmatoarele comenzi Git arata toate commit-urile care au fost facute:
    a. $ git log
    b. $ git init
    c. $ git status
    d. $ git add
38. Ce design pattern se foloseste pentru modelarea comportamentului la evenimentul onClick pentru un element de tip Button?

a. Observer
b. Chain of responsibility
c. State
d. Visitor