PROGRAMARE AVANSATĂ PE OBIECTE

Laborator 2

Maria Cristina Chițu
mariachitu11@gmail.com

Problema 1

Creati o clasa **Coada** ce va implementa o coada pentru obiecte de tip int.

Date :

- Vector de int-uri
- Numarul de elemente din coada

Constructori:

- constructor fara parametri (dimensiunea implicita a cozii fiind 100 de elemente);
- constructor cu un parametru int, reprezentand dimensiunea cozii.

Metode:

- void push(int) – adauga un element in coada
- int pop() – scoate elementul corespunzator din coada si il returneaza
- boolean isEmpty() – verifica daca mai exista elemente in coada
- String toString() – afiseaza numarul de elemente din coada, urmat de fiecare element.
- void main(String[]) – metoda main instantiaza o coada si verifica operatiile de mai sus.

Problema 2

Sa se creeze o clasa cu **2 metode**:

- metoda public static void **main**
- metoda care sa verifice daca o **matrice** patratica este simetrica sau nu

O matrice simetrică este o matrice pătratică care este egală cu transpusa sa.

Exemplu matrice simterica:

$$A = \begin{bmatrix} 1 & 7 & 3 \\ 7 & 4 & -5 \\ 3 & -5 & 6 \end{bmatrix}$$

Problema 3

Definiti o clasa **Matrice** care va contine urmatoarele:

Date :

- Matrice de numere intregi
- Numarul de linii
- Numarul de coloane

Metode:

- Matrice **add**(Matrice m2) – returneaza suma dintre m2 si matricea curenta
- Matrice **multiply**(Matrice m2) – returneaza produsul dintre m2 si matricea curenta **ATENTIE**! Verificati daca dimensiunile celor 2 matrici corespund!
- Matrice **pow**(int power) – returneaza rezultatul ridicarii matricii curente la puterea power (folositi metoda multiply)
- String **toString**() – afiseaza matricea, nr de linii si de coloane

Instantiati doua obiecte de tip Matrice, in metoda statica main, si apelati metodele implementate.

Problema 4

Definiti o clasa executabila **Maximum** care are ca membru un numar intreg si contine metodele descrise mai jos. Implementati metodele astfel incat doar metoda cu un singur parametru sa foloseasca if.

Instantiati un obiect de tip Maximum, in metoda statica main, si apelati metodele implementate.

Ce principiu POO este evidentiat in acest exercitiu?

```
//returneaza maximul dintre nr (membrul clasei) si a
public int maxim(int a) {}
//returneaza maximul dintre nr, a si b
public int maxim(int a, int b){}
//returneaza maximul dintre nr, a, b si c
public int maxim(int a, int b, int c){}
//returneaza maximul dintre nr, a, b, c si d
public int maxim(int a, int b, int c, int d){}
```

Problema 5

Definiti o clasa **Complex** care modeleaza lucrul cu numere complexe.

Membrii acestei clase sunt:

- doua atribuite de tip **double** pentru partile reala, respectiv imaginara ale unui numar complex;
- un **constructorul** cu doi parametri de tip **double**, pentru setarea celor doua parti ale numarului (reala si imaginara);
- un **constructor** fara parametri care apeleaza constructorul anterior;
- o metoda, cu un singur parametru, de calcul a **sumei** a doua numere complexe;
- o metoda **toString** uzitata pentru afisarea pe ecran a valorii numarului complex;
- o metoda **equals**, cu un parametru de tip Object, care va returna true daca numerele comparate sunt egale, respectiv false in sens contrar;
- o metoda **main** pentru testarea functionalitatii clasei.

Problema 6

Creati un pachet “geometry” in care se vor construi clasele prezentate in continuare.

Implementati clasa **Punct** care defineste un punct din spatiul 2D.

Datele clasei (private):

- doua nr. intregi reprezentand cele doua coordonate ale punctului. (x, y)

Conctructorul clasei:

- un constructor fara parametri care instantiaza punctul O(0, 0).
- un constructor cu parametri care instantiaza punctul (x,y)

Metodele clasei

- int getX() = intoarce abscisa punctului;
- void setX(int x) = seteaza abscisa punctului;
- int getY() = intoarce ordonata punctului;
- void setY(int y) = seteaza ordonata punctului;
- String toString() = returneaza un String de forma (x, y);

- double distance(int, int) = calculeaza distanta dintre 2 puncte;
- double distance(Punct p1) = calculeaza distanta dintre 2 puncte.

Creati o clasa executabila **TestPunct**, in acelasi pachet cu clasa Punct, care calculeaza distanta dintre punctele A(1, 3) si B(-1, 2).

Puteti accesa datele clasei Punct in metoda main din clasa TestPunct?

Problema 7

Un **produs** este caracterizat prin **nume** (String), **pret** (double) si **cantitate** (int). Un **magazin** are un **nume** (String) si contine 3 **produse**.

Creati clasele **Produs** si **Magazin** corespunzatoare specificatiilor de mai sus. In fiecare clasa, implementati **constructorul** potrivit, astfel incat caracteristicile instantelor sa fie setate la crearea acestora.

Clasa **Produs** contine o metoda toString, care va returna un String sub forma "Produs: nume_produs; pret: pret_produs; cantitate: cantitate" si o metoda getTotalProdus care va returna un double, produsul dintre cantitate si pret.

Clasa **Magazin** contine o metoda toString care afiseaza toate componentele magazinului (va apela metoda toString pentru fiecare produs) si o metoda getTotalMagazin care va calcula suma totalurilor produselor si o va returna.

Creati, intr-o metoda main, un obiect de tip Magazin, uzitand obiecte anonime in cadrul instantierii.

Problema 8

- Creati un pachet nou in care sa definiti clasa **Student** cu campurile private pentru **nume** si **prenume**.
- Declarati un membru public static “**contor**” care va fi incrementat cu 1 de fiecare data cand va fi create o noua instanta de Student.
- Implementeaza o metoda care sa afiseze informatii despre un student.

- Implementeaza o metoda main in care sa creezi 5 obiecte de tip Student si sa le adagi intr-un array.
- Ulterior, parcurge array si afiseaza informatii despre fiecare student.
- Afiseaza valoarea contorului folosind numele clasei, nu instanta! (Student.contor)

Problema 9

Creeaza o clasa cu 2 metode:

- Public static void main method
- Metoda care cauta un element intr-un int array, iar apoi copiaza intr-un alt array toate elementele din vectorul initial de la indexul la care s-a gasit elemental cautat si pana la final. Metoda are ca parametrii array si elemental cautat.
- Se vor afisa array-ul initial si final.

ATENTIE! Se va locura cu functiile din clasa **Arrays**.

Exemplu:

int arr[] = { 1, 5, 6, 9, 12 ,22 ,7 ,20 };

int item = 9;

int result[] = { 9, 12 ,22 ,7 ,20 };

Problema 10

Creati un pachet “Scoala” in care se vor define urmatoarele clase.

Sa se scrie clasele **Student**, respectiv **Catalog** care au urmatoarele proprietati.

Student este caracterizat prin:

- **nume**(String),
- **medieSem1**(double),
- **medieSem2**(double).

Clasa va contine de asemenea:

- metoda de tip **toString** care va returna datele fiecarui student in parte;
- o metoda **getMedieAn1**() care va returna un double – media anuala a studentului curen.

Catalog contine campurile:

- grupa(String)
- 5 studenti.

Scrieti de asemenea:

- metoda de tip **toString**() corespunzatoare clasei (metoda va apela metoda toString() a fiecarui student in parte, definita anterior)
- metoda getMedieClasa() (va intoarce media anuala a celor 5 studenti).

Creati, intr-o metoda **main**, un obiect de tip **Catalog**, uzitand obiecte anonime in cadrul instantierii.

Modificatori de acces

Visibility	Public	Protected	*Default*	Private
From the same class	Yes	Yes	Yes	Yes
From any class in the same package	Yes	Yes	Yes	No
From a subclass in the same package	Yes	Yes	Yes	No
From a subclass outside the same package	Yes	Yes, *through inheritance*	No	No
From any non-subclass class outside the package	Yes	No	No	No

- **public** - The method can be called from any class.
- **private** - The method can only be called from within the same class.
- **protected** - The method can only be called from classes in the same package or subclasses.
- **( ) - Default** (Package Private) - The method can only be called from classes in the same package.