Constructorul este un tip special de metodă utilizat pentru a inițializa obiectul.

Constructorul Java este apelat la momentul creării obiectului. Acesta construiește valorile, adică furnizează date pentru obiect, de aceea este cunoscut sub numele de constructor.

Există practic două reguli definite pentru constructor:

1. Numele constructorului trebuie să fie același cu numele clasei sale.

2. Constructorul nu trebuie să aibă un tip de retur explicit.

Tipuri de constructori în Java

Există 3 tipuri de constructori:

1. Constructor fără parametri (default constructor)

2. Constructor parametrizat (parameterized constructor)

3. Constructor de copiere (copy constructor)

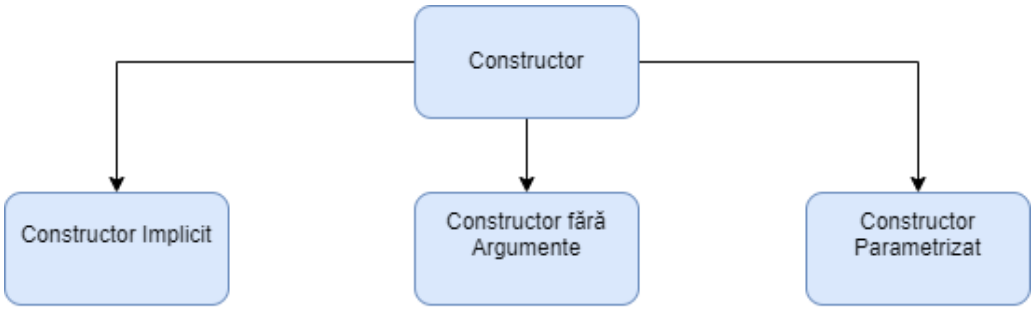

Figura 1. Tipuri de constructori

Constructor Implicit (Default Constructor):

Dacă nu definiți niciun constructor în clasa dvs., Java generează unul care nu primește niciun parametru în mod implicit. Acest constructor este cunoscut sub numele de constructor implicit. Nu îl veți găsi în codul sursă, dar va fi prezent acolo.

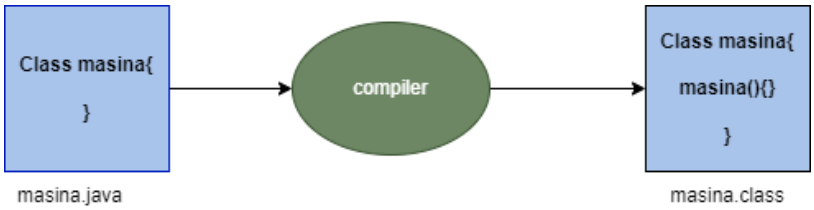

Figura 2. Creare clasa

Constructor fără Argumente în Java

Un constructor care nu are niciun parametru este cunoscut sub numele de Constructor fără Argumente.

Sintaxa constructorului implicit:

Copiază codul

<class_name>() {}

Exemplu de Constructor fără Argumente:

În acest exemplu, creăm un constructor fără argumente în clasa Bike. Acesta va fi apelat în momentul creării obiectului.

```
class Bike1
{
Bike1()
 {
 System.out.println("Bike is created");
 }
public static void main(String args[])
 {
 Bike1 b=new Bike1();
 }
}
```

Figura 3. Exemplu de constructor fara argumente

Regulă: *Dacă nu există niciun constructor într-o clasă, compilatorul creează automat un constructor implicit.*

Punct important:

Orice constructor pe care îl scrieți în clasa dvs. nu poate fi numit implicit deoarece îl scrieți dvs. Constructorul este numit implicit doar atunci când este generat de Java.

Constructor Parametrizat

Un constructor care are parametri este cunoscut sub numele de constructor parametrizat.

De ce să folosiți un constructor parametrizat?

Constructorul parametrizat este folosit pentru a furniza valori diferite obiectelor distincte.

Exemplu de constructor parametrizat

În acest exemplu, am creat constructorul clasei Student care are doi parametri. Putem avea orice număr de parametri în constructor.

```
class Student {
    int id;
    String name;

    // Constructor parametrizat
    Student(int i, String n) {
        id = i;
        name = n;
    }

    void display() {
        System.out.println(id + " " + name);
    }

    public static void main(String args[]) {
        // Crearea de obiecte Student folosind constructorul parametrizat
        Student s1 = new Student(111, "Karan");
        Student s2 = new Student(222, "Aryan");

        s1.display();
        s2.display();
    }
}
```

Figura 4. Exemplu de utilizare al Constructorului parametrizat

Diferența între constructor și metodă în Java

Există multe diferențe între constructori și metode. Acestea sunt prezentate mai jos.

Constructor	Metodă
Constructorul este folosit pentru a inițializa starea unui obiect.	Metoda este folosită pentru a expune comportamentul unui obiect.
Constructorul nu trebuie să aibă tip de retur.	Metoda trebuie să aibă tip de retur.
Constructorul este apelat implicit.	Metoda este apelată explicit.
Compilatorul Java furnizează un constructor implicit dacă nu aveți niciun constructor.	Metoda nu este furnizată de compilator în niciun caz.
Numele constructorului trebuie să fie același cu numele clasei.	Numele metodei poate fi sau nu același cu numele clasei.

1. Scrieți un program în Java care să permită citirea și afișarea unei matrice pătrate. Programul va solicita utilizatorului să introducă numărul de linii și coloane ale matricei (care vor fi egale, formând astfel o matrice pătrată) și apoi va solicita introducerea elementelor matricei. Dacă utilizatorul introduce un număr incorect de elemente pe o linie, programul va completa restul elementelor cu zero și va afișa un mesaj de avertizare.

```java
package lab4;
import java.util.Scanner;
public class MatricePatrata {
    private int n;                // dimensiunea matricei (n x n)
    private int[][] matrice;      // matricea propriu-zisă
    // Constructor parametrizat - initializează o matrice de dimensiune n
    public MatricePatrata(int n) {
        this.n = n;
        matrice = new int[n][n];
    }
    // Metodă pentru citirea matricei de la tastatură
    public void citireMatrice() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Introduceți elementele matricei (" + n + "x" + n + "):");
        for (int i = 0; i < n; i++) {
            System.out.print("Linia " + (i + 1) + ": ");
            String linie = scanner.nextLine();
            String[] valori = linie.trim().split("\\s+"); // separă elementele prin spatiu
            // Verifică dacă sunt mai putine elemente decât n
            if (valori.length < n) {
                System.out.println("Avertisment: Număr insuficient de elemente pe linia " + (i + 1) +
                                   ". Valorile lipsă vor fi completate cu 0.");
            }
            // Completează matricea cu valorile citite sau cu 0
            for (int j = 0; j < n; j++) {
                if (j < valori.length) {
                    matrice[i][j] = Integer.parseInt(valori[j]);
                } else {
                    matrice[i][j] = 0; // completează cu 0 dacă lipsesc elemente
                }
            }
        }
    }
```

Figura 1. Functia de creare a Matricei

```java
    // Metodă pentru afisarea matricei
    public void afisareMatrice() {
        System.out.println("\nMatricea introdusă este:");
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                System.out.print(matrice[i][j] + "\t");
            }
            System.out.println();
        }
    }
```

Figure 2. Metoda de afisare

```java
    // Metoda main
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Introduceți dimensiunea matricei pătrate: ");
        int dim = scanner.nextInt();
        scanner.nextLine(); // consumă linia rămasă după nextInt()

        MatricePatrata m = new MatricePatrata(dim);
        m.citireMatrice();
        m.afisareMatrice();
    }}
```

2. Se cere să se scrie un program Java care definește o clasă numită **Carte**. Clasa va conține trei atribute: titlu, autor și anPublicare. Se va crea un **constructor fără parametri** care inițializează aceste câmpuri cu valori implicite: titlul va fi „Necunoscut”, autorul „Anonim”, iar anul publicării 0. Programul va afișa apoi aceste informații folosind o metodă numită **afisareCarte()**.

```
package lab4;

public class Carte {
    private String titlu;
    private String autor;
    private int anPublicare;
    public Carte() {
        titlu = "Necunoscut";
        autor = "Anonim";
        anPublicare = 0;
    }

    public void afisareCarte() {
        System.out.println("Titlu: " + titlu);
        System.out.println("Autor: " + autor);
        System.out.println("An publicare: " + anPublicare);
    }

    public static void main(String[] args) {
        Carte c1 = new Carte();
        System.out.println("Detalii carte:");
        c1.afisareCarte();
    }
}
```

3.Se cere să se realizeze un program Java care definește o clasă numită Student.Clasa va conține trei atribute: nume, varsta și medie.Programul va citi aceste date de la tastatură și va crea un obiect de tip Student folosind un constructor parametrizat.Se va implementa o metodă numită afisareStudent(), care va afișa toate informațiile despre studentul introdus.

```java
package lab4;
import java.util.Scanner;
public class Student {
    private String nume;
    private int varsta;
    private double medie;
    // Constructor parametrizat
    public Student(String nume, int varsta, double medie) {
        this.nume = nume;
        this.varsta = varsta;
        this.medie = medie;
    }
    // Metodă pentru afisare
    public void afisareStudent() {
        System.out.println("Nume: " + nume);
        System.out.println("Vârstă: " + varsta);
        System.out.println("Medie: " + medie);
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Introduceți numele studentului: ");
        String nume = sc.nextLine();
        System.out.print("Introduceți vârsta: ");
        int varsta = sc.nextInt();
        System.out.print("Introduceți media: ");
        double medie = sc.nextDouble();

        Student s = new Student(nume, varsta, medie);
        System.out.println("\nDatele studentului:");
        s.afisareStudent();
    }
}
```

4. Se cere să se realizeze o clasă numită **Produs**, care să conțină trei atribute: denumire, pret și cantitate. Programul va citi aceste date de la tastatură și va apela un **constructor parametrizat** pentru a crea obiectul. Clasa va conține o metodă **valoareTotala()**, care calculează valoarea totală a produsului (pret * cantitate), precum și o metodă **afisareProdus()**, care afișează toate informațiile despre produs, inclusiv valoarea totală calculată.

```java
package lab4;
import java.util.Scanner;
public class Produs {
    private String denumire;
    private double pret;
    private int cantitate;
    // Constructor parametrizat
    public Produs(String denumire, double pret, int cantitate) {
        this.denumire = denumire;
        this.pret = pret;
        this.cantitate = cantitate;
    }
    // Metodă pentru calculul valorii totale
    public double valoareTotala() {
        return pret * cantitate;
    }
    // Metodă pentru afisare
    public void afisareProdus() {
        System.out.println("Denumire: " + denumire);
        System.out.println("Preț unitar: " + pret + " lei");
        System.out.println("Cantitate: " + cantitate);
        System.out.println("Valoare totală: " + valoareTotala() + " lei");
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Introduceți denumirea produsului: ");
        String denumire = sc.nextLine();
        System.out.print("Introduceți prețul unitar: ");
        double pret = sc.nextDouble();
        System.out.print("Introduceți cantitatea: ");
        int cantitate = sc.nextInt();
        Produs p = new Produs(denumire, pret, cantitate);
        System.out.println("\nDetaliile produsului:");
        p.afisareProdus(); }}
```