Legare statică/dinamică – static/dynamic binding în Java

Programare Orientată pe Obiecte

Exercițiu propus

```
class Ana {
        public void print(Ana p) {
            System.out.println("Ana 1\n");
        }
}
class Mihai extends Ana {
        public void print(Ana p) {
            System.out.println("Mihai 1\n");
        }
        public void print(Mihai l) {
            System.out.println("Mihai 2\n");
        }
}
class Dana extends Mihai {
        public void print(Ana p) {
            System.out.println("Dana 1\n");
        }
        public void print(Mihai l) {
            System.out.println("Dana 2\n");
        }
        public void print(Dana b) {
            System.out.println("Dana 3\n");
        }
}
```

Exercițiu propus

```
public class Test{
    public static void main (String [] args) {
        Mihai stud1 = new Dana();
        Ana stud2 = new Mihai();
        Ana stud3 = new Dana();
        Dana stud4 = new Dana();
        Mihai stud5 = new Mihai();
 1      stud1.print(new Ana());
 2      ((Dana)stud1).print(new Mihai());
 3      ((Mihai)stud2).print(new Ana());
 4      stud2.print(new Dana());
 5      stud2.print(new Mihai());
 6      stud3.print(new Dana());
 7      stud3.print(new Ana());
 8      stud3.print(new Mihai());
 9      ((Dana)stud3).print(new Mihai());
10      ((Dana)stud3).print(new Dana());
11      stud4.print(new Dana());
12      stud4.print(new Ana());
13      stud4.print(new Mihai());
14      stud5.print(new Dana());
15      stud5.print(new Mihai());
16      stud5.print(new Ana());   } }
```

Ierarhie

Ana – print (Ana)

|

Mihai – print (Ana), print (Mihai)

|

Dana – print (Ana), print (Mihai), print (Dana)

Tip – nume -> obiect

- **Mihai - stud1 -> Dana**
- **Ana - stud2 -> Mihai**
- **Ana - stud3 -> Dana**
- **Dana - stud4 -> Dana**
- **Mihai - stud5 -> Mihai**

Output

1 Dana 1
2 Dana *2*
3 Mihai 1
4 **Mihai 1**
5 **Mihai 1**
6 **Dana 1**
7 **Dana 1**
8 **Dana 1**
9 *Dana 2*
10 *Dana 3*
11 *Dana 3*
12 *Dana 1*
13 *Dana 2*
14 **Mihai 2**
15 Mihai 2
16 Mihai 1

Explicații

1. stud1.print(new Ana())

   stud1 -> *Dana* apelează *Dana.print(Ana*)
2. ((Dana)stud1).print(new Mihai());

   stud1 -> *Dana* apelează *Dana.print(new* Mihai())
3. ((Mihai)stud2).print(new Ana());

   stud2 -> *Mihai* apelează *Mihai.print(new* Ana());
4. stud2.print(new Dana());

   stud2 este declarat *Ana*. Atunci când compilatorul se uită să vadă ce poate apela găseşte metoda *print(Ana*) din clasa *Ana*. La execuție, stud2 este un *Mihai* aşa că va apela metoda p*rint(Ana*) din clasa *Mihai*.
5. Samd