POO

- 50p - laborator → 2-6
- 40p - 4 teme
- 20p - interviu

- minim 25p în lab
- minim 50p în total

✓ - OO
✓ - Obiect - Clasă
✓ - Câmp - Metodă
- variabile - referințe
- constructor - obiecte noi

Obiectul → secvență de program care (entitate) realizează o funcție clară în cadrul aplicației

Clasa → categoria din care face parte un obiect.

Obiectul este instanța a clasei.

C

```
struct Stack {
    int *vector;
    unsigned capacity;
    unsigned stack_head;
};

struct Stack * create(unsigned capacity){
    struct Stack *newStack =
        (struct Stack *) malloc(sizeof(
                                struct Stack));
    ...
}
```

Java

```
class Stack {
    Câmpuri [ int [] vector;
              int stackHead;

    Metode [ void push(int element){
                //check if full
                vector[stackHead++] =
                    element;
             }
    }
    ...
    Stack myStack;
    myStack.push(3);
```

Variabile

- primitive (byte, short, int, long, float, double, char, boolean) — 8 biți
- referință ↔ obiecte

-2^{n-1} ; $2^{n-1} - 1$

byte	8 biți	2^8 valori	-128; 127
short	16 biți	2^{16} valori	-32768; 32767 — $(-2^{15}; 2^{15}-1)$
int	32 biți	2^{32} valori	-2^{31} ; $2^{31}-1$
long	64 biți	2^{64} valori	— —
float	32 biți		
double	64 biți		
char	16 biți	→ caractere	
boolean	true / false	→ 8 biți	

true != 1
false != 0

```
class Persoana{
    int varsta; //-> primitivă
    String nume; //-> referință
}
```

REFERINȚA ≠ OBIECT

Referința este un mod de acces la obiect.

În Java clasele încep cu literă mare.

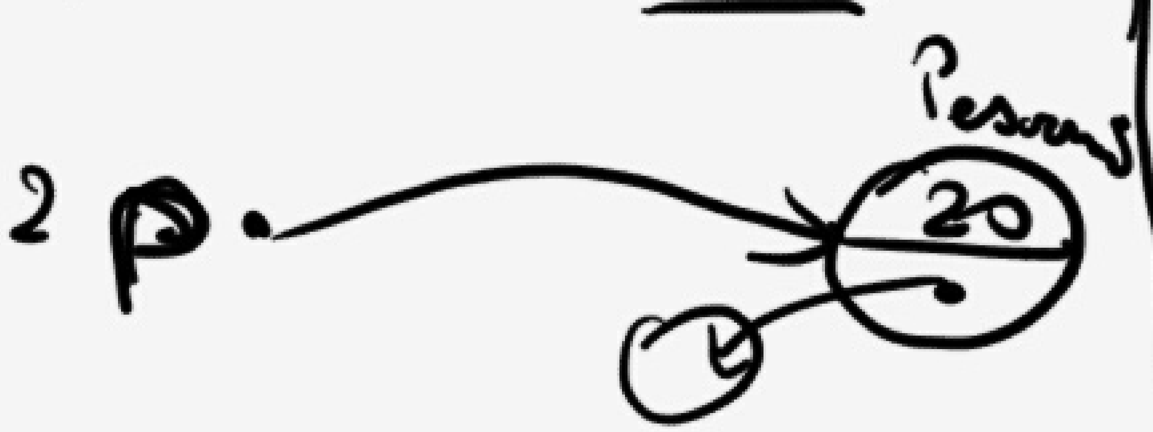
String -> clasă
-> șiruri de caractere

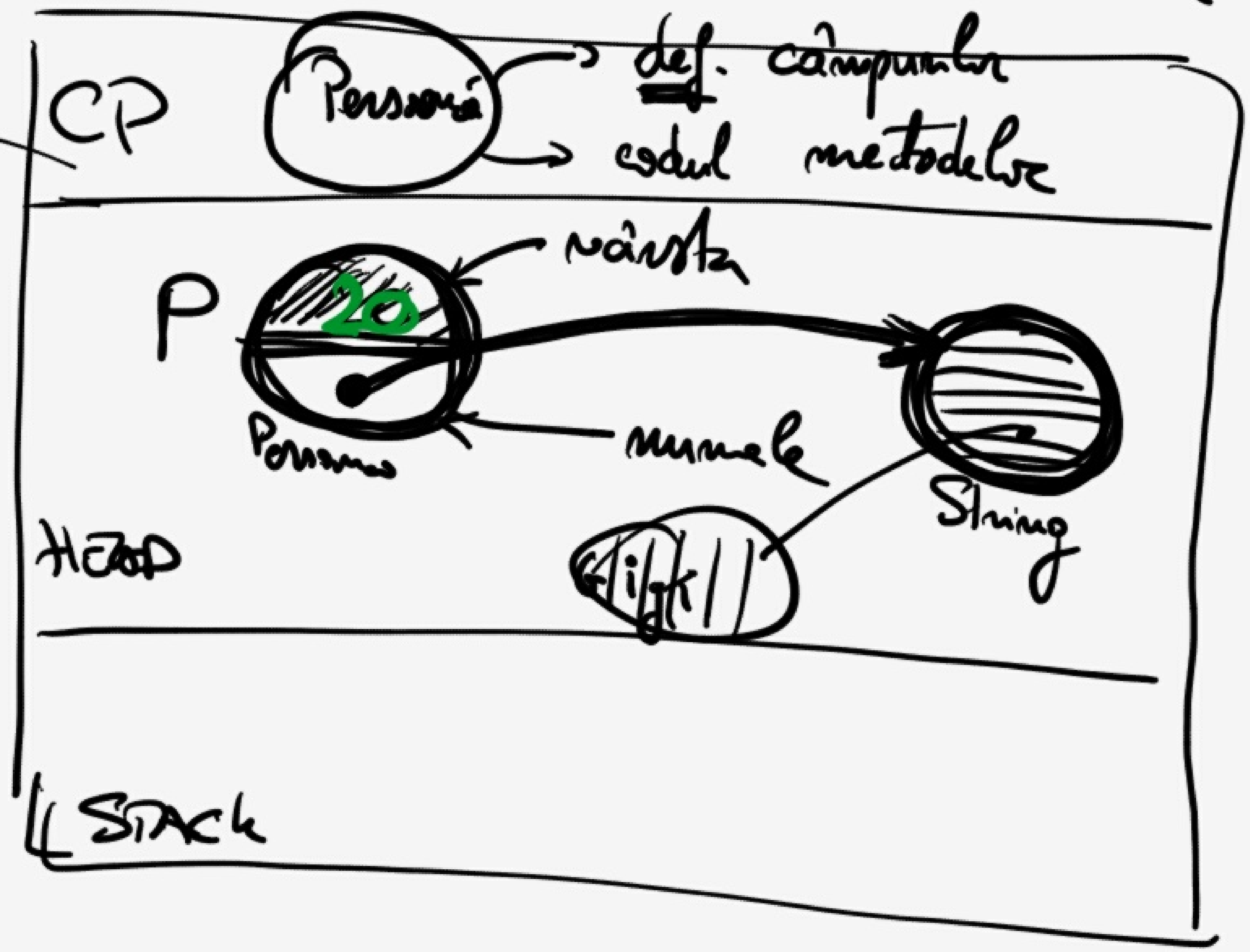
```
1: Persoana p;
2: p = new Persoana();
        a      b

p.vârsta = 20;
p.nume = "Gigel";
```

```
class String{
    char[] sir;
}
```

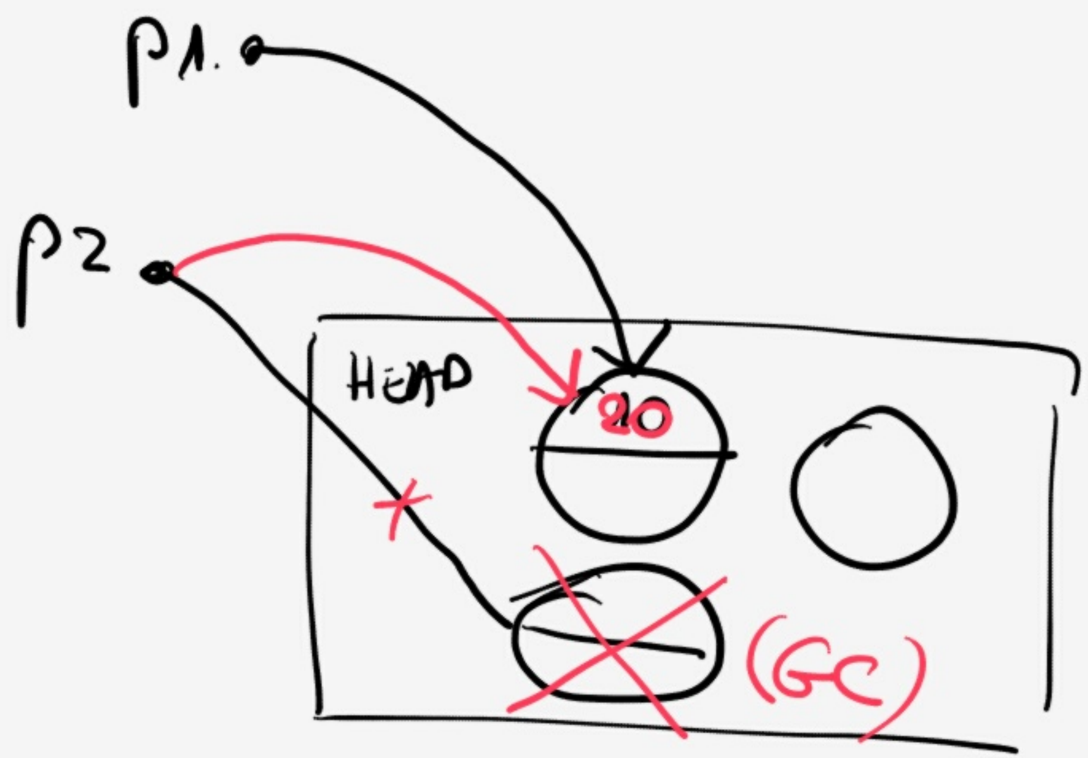
1: p -> null

2 p.





```
class Persoana {
    int varsta;
    String nume;
}

//---
Persoana p1 = new Persoana();
Persoana p2 = new Persoana();

p1.varsta = 10;
p2 = p1;
p1.varsta = 20;
System.out.printf("p2.varsta = %d", p2.varsta);
```



Imutabil → un obiect odată creat, el nu mai poate fi modificat

Constructorul este o metodă care se folosește atunci când se necesită un obiect nou.

- are același nume cu numele clasei
- nu are tip returnat.

```
class Persoana {
    int varsta;
    String nume;

    Persoana() {
        varsta = -1;
        nume = null;
    }
}
```

Dacă și nu mai dacă într-o clasă nu se definește explicit un constructor compilatorul va introduce implicit un constructor gol fără argumente.