Object - Oriented Programming

- Descompozitia în sub-sisteme (obiecte)

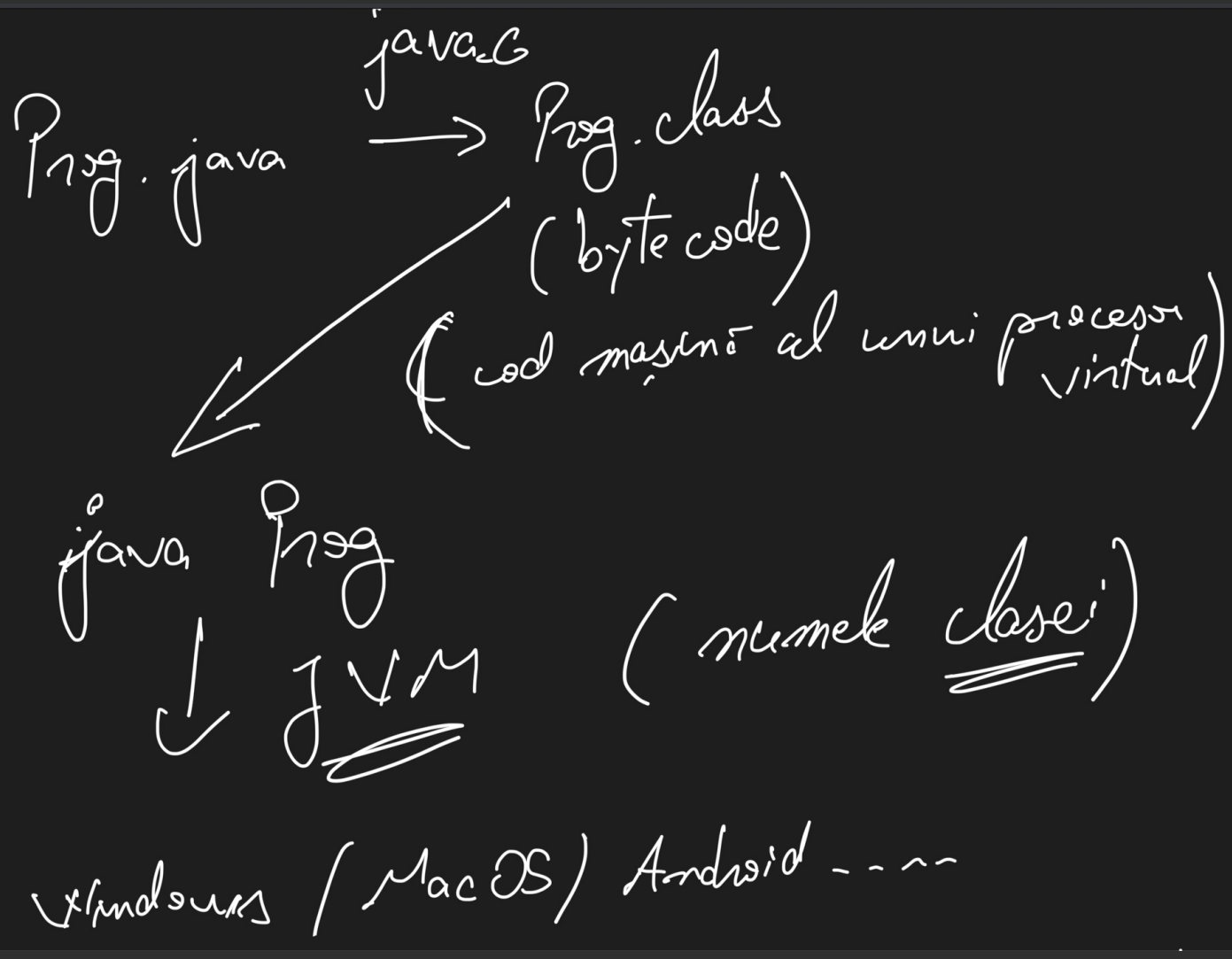
– sistemul este descompus într-un set de obiecte (si clasele lor)

```
class Main {
    public static void main(String[] args) {

    }
}
```

Metoda main - aici începe programul

! Orice clasă poate avea main-ul propriu



Portabilitate → "write once, run anywhere"

TIPURI DE DATE PRIMITIVE

	biți
byte	8
short	16
int	32
long	64
float	32
double	64
char	16
boolean	-

IDENTIFICATORI

case – sensitive

keywords

Operatorii logici → se aplică doar pe tip de data boolean

&& / ||

cu scurtcircuitare left -> right

Operatori relaționali == !=

↓ return value

True / False

< <=

> >=

Operatori pe biti

~ | << >>/>>> | & | ^ | |

Op. atribuire

=

+=

*=

--

Op. conditional

expr_bool ? exp1 : exp2

True False

CLASE și OBIECTE

Clasa: def. un set de obiecte care aceeași struct. și comp.

Obiect: instanță a unei clase

Clasa : implementarea unui object

```
class Clock {
    private int hour, minute, second;
    public void setTime(int h, int m, int s)
{
```

Variabile referință

```
Clock clock1, clock2;
```

null = valoare ce arată ca nu mai există referința

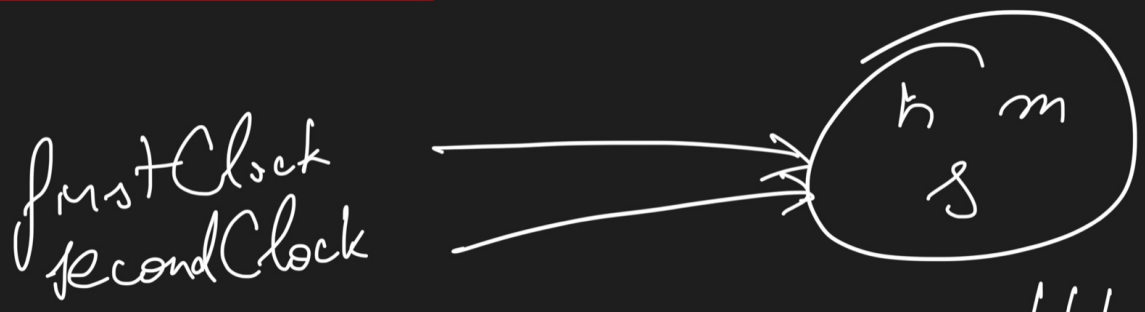
```
clock1 = null;
```

Invocarea metodelor

```
clock1.setTime(5, 10, 0);
```

```
(int hour ...)
this.hour = (hour >= 0) ...
```

ALIASING



același zonă de memorie !!!

Constructori

1) Numele exact cu numele clasei
2) Nu are tip returnat

→ se poate executa o dată la crearea obiectului

```
class Clock {
    public Clock(int h, m, s)
    {

    }
}
```

```
Clock clock = new Clock(12, 12, 12);
```

- constructorul este generat automat (fără argumente) (no-arg)

Specificatori de acces

private -> doar în interiorul clasei

public -> accesat de oriunde