

## Introdurre

V20180924

# 1

## Ce înseamnă **object-oriented** ?

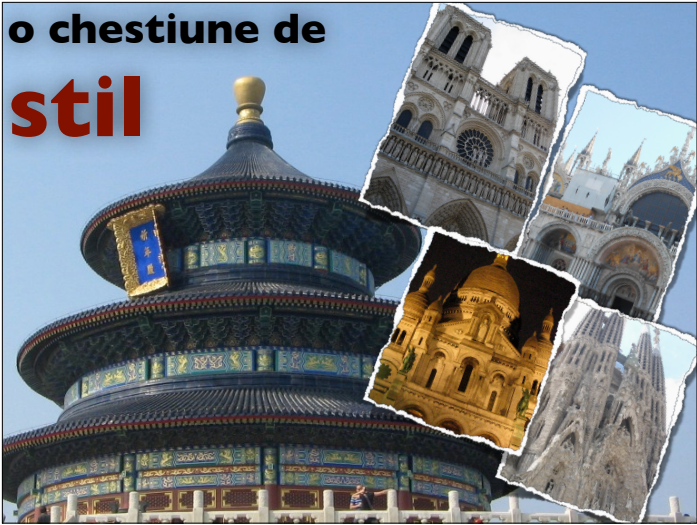
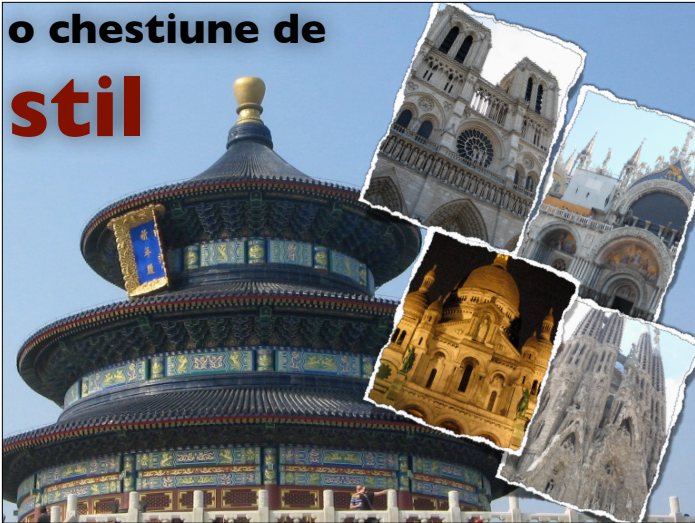
# 1

## Ce înseamnă **object-oriented** ?

# 1

## Ce înseamnă **object-oriented** ?

o chestiune de  
stil



## Decompoziție

**Sistem software**

## Decompoziție

Sub-sistem 2

Sub-sistem 3

Sub-sistem 1

Sub-sistem 4

**Sistem software**

# Decompoziție

A diagram consisting of a single rectangular box with a black border. Inside the box, the text "Sub-sistem I" is written in a black, sans-serif font, centered horizontally and vertically.

Sub-sistem I

Dr. Petru Florin Mihailescu

Sub-sistem I

# Decompoziție

The diagram illustrates the decomposition of a sentence into four rectangular boxes. The boxes are arranged in two rows of two. An arrow points from the text "Ce sunt aceste 'entități' ?" to the fourth box.

Ce sunt aceste "entități" ?

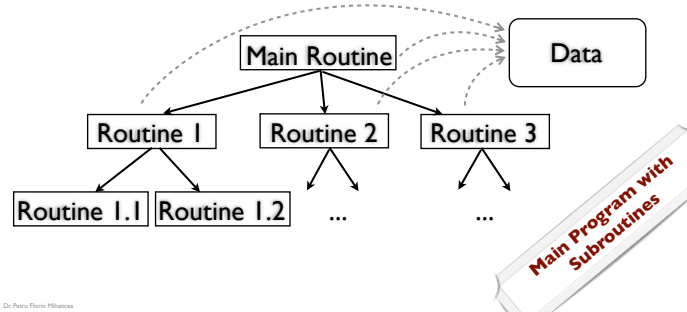
# Decompoziție

The diagram illustrates the decomposition of a sentence into four rectangular boxes. The boxes are arranged in two rows of two. An arrow points from the text "Ce sunt aceste 'entități' ?" to the fourth box.

Ce sunt aceste "entități" ?

## Decompoziția **Algoritmică**

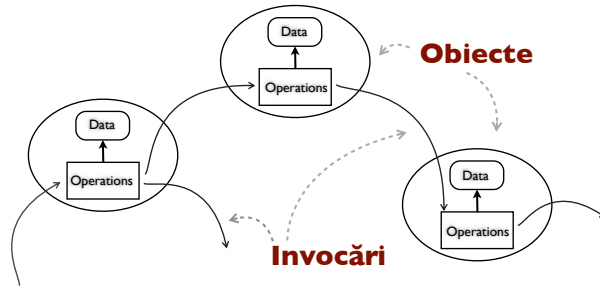
fiecare “entitate” din sistem reprezintă un **pas** de execuție dintr-un proces/sarcină de realizat



Dr. Petru Florin Mărcuș

## Decompoziția **Object-Oriented**

sistemul e descompus într-un set de **obiecte**  
(și **claselor** lor) ce colaborează între ele



## **Definiție** (parțială)

**Programarea orientată pe obiecte**

este o metodă de

**implementare a programelor**

în care acestea sunt organizate ca și

**colecții de obiecte**

**ce cooperează între ele**

[ pentru îndeplinirea funcționalităților  
acelor programe ]

# 2

## Primii pași în Java

Dr. Petru Florin Mihancea

## Limbaje Object-Oriented (I)

Limbaje dedicate programării obiectuale



C#



Eiffel



...

## Limbaje Object-Oriented (II)



James Gosling @ Sun Microsystem  
1995

Oracle Corporation  
2009 -

## Limbaje Object-Oriented (II)

TIOBE Index

Sep 2018	Sep 2017	Change	Programming Language	Ratings	Change
1	1		Java	17.436%	+4.75%
2	2		C	15.447%	+8.06%
3	5	▲	Python	7.653%	+4.67%
4	3	▼	C++		
5	8	▲	Visual Basic .NET		
6	4	▼	C#		
7	6	▼	PHP		
8	7	▼	JavaScript		
9	-	▲	SQL		
10	18	▲	Objective-C		

Worldwide, Sept 2018 compared to a year ago:

Rank	Change	Language	Share	Trend
1	▲	Python	24.58 %	+5.7 %
2	▼	Java	22.14 %	-0.6 %
3	▲	Javascript	8.41 %	+0.0 %
4	▼	PHP	7.77 %	-1.4 %
5		C#	7.74 %	-0.4 %
6		C/C++	6.22 %	-0.8 %
7		R	4.04 %	-0.2 %
8		Objective-C	3.33 %	-0.9 %
9		Swift	2.65 %	
10		Matlab		

PYPL Index

## Instalare

Java Installation Tutorial 



<http://labs.cs.upt.ro/~oose/pmwiki.php/OOP/Links>

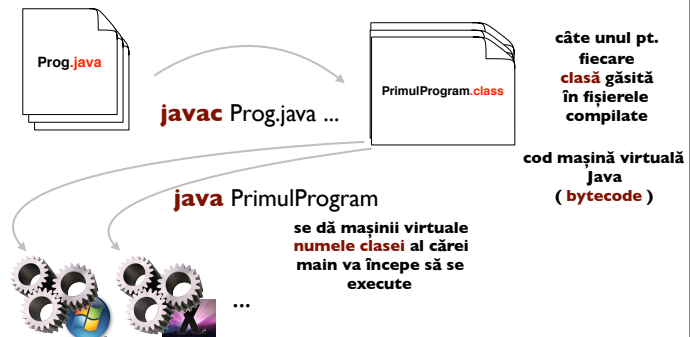
## Primul Program Java

```
class PrimulProgram {  
    public static void main(String argv[]) {  
        System.out.println("Hello world!");  
    }  
}
```

**metoda main** - aici începe programul și trebuie declarată exact ca mai sus

**orice clasă** poate avea un main al ei

## Compilare & Rulare



**Portabilitate** "write once, run anywhere"

Tipuri de date primitive

Tip	Biți	Domeniu / Exemple
byte	8	-128..127
short	16	-32768..32767
int	32	-2147483648..2147483647
long	64	-9223372036854775808..9223372036854775807
float	32	±1.4E-45..±3.4028235E+38
double	64	±4.9E-324..±1.7976931348623157E+308
char	16	ex. 'a', '\u03C0' (codificare Unicode)
boolean	-	true, false (sunt cuvinte cheie)

## Identificatori

### Nume date entităților de program

ex. variabile, parametrul unei funcții, etc.

Poate începe cu literă, \_, \$

Conține litere (mari/mici), \_, \$, cifre

Unele sunt rezervate deoarece reprezintă

**cuvinte cheie** (ex. class, public, if, ...)

**Atentie**  
Limbaajul e case-sensitive  
deci de exemplu  
identificatorul **a** respectiv  
**A** sunt diferiți

## Variabile (**locale**)

```
class Exemplu {  
    public static void main(String[] args) {  
  
        int a = 10;  
        String b = "Mama"; //String nu e tip primitiv  
        double c;  
  
        System.out.println(a);  
        System.out.println(b);  
        System.out.println(c); //eroare de compilare  
    }  
}
```

**Variabilele locale alocate pe stiva de execuție**

**Nu se inițializează implicit**

## Operatori

**Precedența** - ordinea aplicării

**Asociativitatea** - ordinea aplicării când avem aceeași precedență



## Post incrementare/decrementare

```
class Post {  
    public static void main(String[] args) {  
        double v = 0;  
        System.out.println(v++);  
        System.out.println(v);  
        char c = 'b';  
        System.out.println(c--);  
        System.out.println(c);  
    }  
}
```

### OUTPUT

```
0.0  
1.0  
b  
a
```

Trebuie aplicat  
unei variabile

## Pre incrementare/decrementare

```
class Pre {  
    public static void main(String[] args) {  
        double v = 0;  
        System.out.println(++v);  
        System.out.println(v);  
        char c = 'b';  
        System.out.println(--c);  
        System.out.println(c);  
    }  
}
```

### OUTPUT

```
1.0  
1.0  
a  
a
```

## Operatori aritmetici

```
class Aritmetici {  
    public static void main(String[] args) {  
        int a = 1000;  
        double b = 19.4;  
        char c = 'a';  
        System.out.println(a + -b);  
        System.out.println(c + 1);  
        System.out.println(a / 3);  
        System.out.println(a % 3);  
        System.out.println(((double)a) / 3);  
    }  
}
```

+ - (unari)  
\* / %  
+ -

### OUTPUT

```
980.6  
98  
333  
1  
333.3333333333333
```

## + e și concatenare de String-uri

```
class Concatenari {  
    public static void main(String[] args) {  
        int a = 10;  
        System.out.println("a" + "=" + a);  
        System.out.println(a + 1 + "=a");  
    }  
}
```

**Dacă unul din operanzii lui + e String  
celălalt e convertit la String**

### OUTPUT

a=10  
11=a

## Operatori logici

negație	!
conjuncție	& &&
disjuncție	
sau-exclusiv	^

Operanzii trebuie să fie de tip  
boolean iar valoarea dată e  
tot un boolean

**&& și || se execută cu scurtcircuitare,  
prima dată operandul stâng**

### Exemplu

a && b

dacă **a** este false sigur toată expresia e false (nu se mai evaluează restul)

a || b

dacă **a** este true sigur toată expresia e true (nu se mai evaluează restul)

## Operatori relaționali & egalitate

== !=

Operanzii de tipuri  
comparabile iar valoarea  
dată e de tip boolean

< <=  
> >=

Operanzii sunt tipuri  
numerice ori char iar  
valoarea dată e de tip  
boolean

## Operatori be biți

~			negare
<<	>>	>>>	deplasări st. / dr. / dr. cu 0
&			și
^			sau-exclusiv
			sau

Operanzii trebuie să fie  
numerici întregi sau char

## Operatori de atribuire

=  
+=  
\*=  
...

O atribuire e în sine o  
expresie ce ia valoarea  
atribuită

```
class Atribuire {  
    public static void main(String[] args) {  
        int a = 7, b = 10, c;  
        c = a = b;  
        System.out.println(a + " " + b + " " + c);  
        a = 7;  
        b = 10;  
        c = a += b;  
        System.out.println(a + " " + b + " " + c);  
    }  
}
```

**OUTPUT**  
10 10 10  
17 10 17

## Operatorul **condițional**

**exp\_booleană ? exp1 : exp2**

**dacă exp\_booleană e **true****  
**operatorul întoarce valoarea lui **exp1****

**dacă exp\_booleană e **false****  
**operatorul întoarce valoarea lui **exp2****

## Instrucțiuni de control (I)

```
if ( expresie_de_tip_boolean ) {  
    ...  
} else {  
    //ramura else evident poate lipsi  
}
```

```
if (1) {  
    //eroare de compilare deoarece nu se  
    //face conversie la boolean; unde se cere  
    //boolean, neaparat trebuie sa avem boolean  
}
```



## Instrucțiuni de control (II)

```
switch( expresie ) {  
    //in esenta expresie trebuie sa fie  
    //char,byte,short,int (de la 1.7 si String)  
    case ExpConstanta1 : ...  
  
    case ExpConstanta2 : ...  
                        break;  
    case ExpConstanta3 : ...  
                        break;  
    default: ...  
}
```

**fall-through  
execution**

## Instrucțiuni de control (III)

```
char c = ...  
switch(c) {  
    case '1': System.out.println("one");  
                break;  
    case '2': System.out.println("two");  
    case '3': System.out.println("three");  
}
```

**dacă c e '1' pe ecran avem**

**one**

**dacă c e '2' pe ecran avem**

**two**

**three**

**...**

## Instrucțiuni de control (IV)

```
while( expresie_de_tip_boolean ) {  
    ...  
}
```

```
do {  
    ...  
} while(expresie_de_tip_boolean);
```

```
for( initializare; expresie_de_tip_boolean; update ) {  
    ...  
}
```

## Instrucțiuni de control (V)

```
int i = 0;
while( i < 10 ) {
    if(i == 5) {
        break;
    }
    System.out.println(i);
    i = i + 1;
}
```

### OUTPUT

0  
1  
2  
3  
4

```
int j;
for(j = 0; j < 10; j++) {
    if(j == 5) {
        continue;
    }
    System.out.println(j);
}
```

### OUTPUT

0  
1  
2  
3  
4  
6  
7  
8  
9