Curs 3 POO
© Conf. univ dr. Crenguta M. Puchianu

- Metode. Declaratie si apel
- Metode cu numar variabil de parametri
- Metode supraincarcate
- Metode recursive

Metode

- O **metoda** este implementarea unui algoritm care se executa in mediul unui obiect sau al unei clase de obiecte.
- Sintaxa declaratiei de metoda:

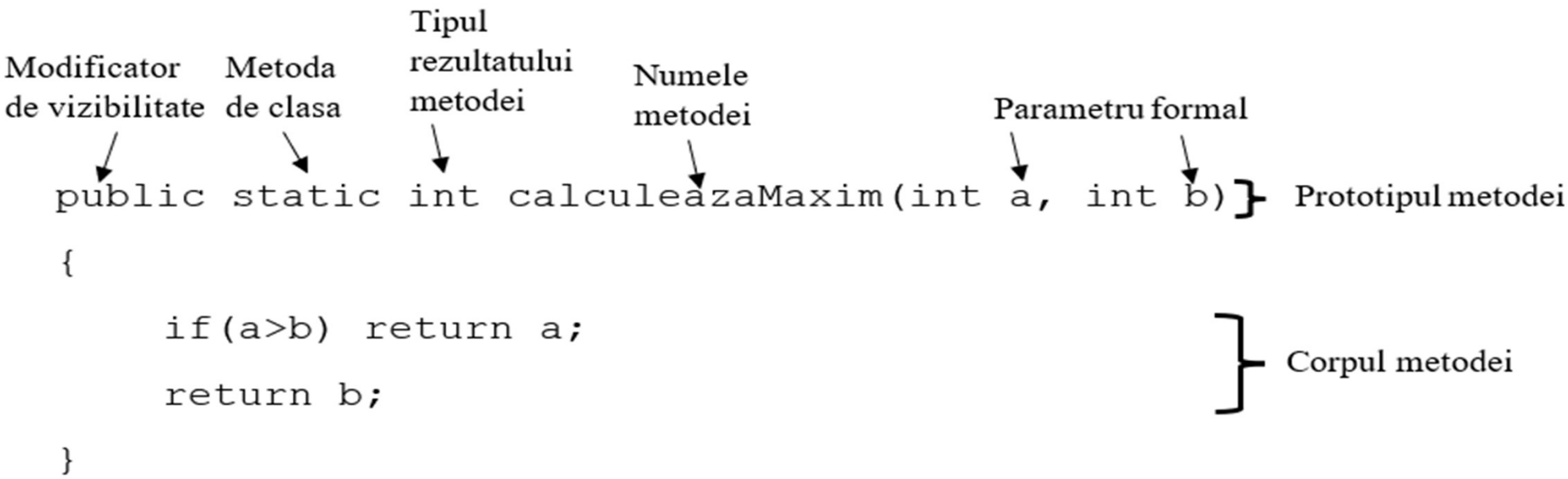
DeclaraţieMetodă ::= [**public** | **protected** | **private**] [**static** | **final**] *Tip Identificator* **(***ListaParametriFormali***)** *Bloc*

ListaParametriFormali ::= λ | *ParametruFormal* | {[**,** *ParametruFormal*]}+

ParametruFormal ::= *Tip Identificator* | *Tip… Identificator*

Bloc ::= **{** *ListaInstructiuni* **}**

Exemplu:



Metode (cont.)

- Sintaxa apelului unei metode:

ApelMetodă ::= [IdentificatorClasa**.**|IdentificatorVariabilaInstanta**.**|**super.**|
this.] Identificator **(**ListaParametriActuali**)**
ListaParametriActuali ::= λ | ParametruActual | {[, ParametruActual]}+
ParametruActual ::= Valoare | Expresie | IdentificatorVariabila | ApelMetoda
care returneaza o valoare

Transferul parametrilor la momentul apelarii unei metode se face in Java prin valoare.

Exemplu:

```
public static void schimba(int x, int y){
  int t;
  t=x;
  x=y;
  y=t;
 }
```

Metode cu numar variabil de parametri (varargs)

- In versiunea 5 Java a aparut posibilitatea de a defini o metoda cu numar variabil de parametri formali de un acelasi tip de date.

Exemplu: public static int calculeazaMaxim(int... a){
...}

Observatii:

- Parametrul variabil al metodei este tratat ca un tablou de elemente de tipul de date declarat;
- O metoda cu varargs poate avea mai multi parametri, dar numai un singur parametru sa fie variabil si acesta sa fie ultimul in lista parametrilor formali;
- O metoda NU poate avea mai multi parametri variabili;

- O metoda cu varargs poate fi apelata fara niciun parametru actual.

Instructiuni de intrerupere a executarii unei metode

Denumire instrucţiune	Sintaxa
Ieşirea dintr-o metodă cu un rezultat	return Expresie;
Ieşirea dintr-o metodă fără rezultat (void)	return;
Lansarea unei excepţii	throw Expresie;
Instrucţiunea try-catch	try{ ListaInstrucţiuni }catch(TipExceptie Identificator){ListaInstrucţiuni } catch(TipExceptie Identificator) {ListaInstrucţiuni } ...
Instrucţiunea try-catch-finally	try{ ListaInstrucţiuni }catch(TipExceptie Identificator){ ListaInstrucţiuni } catch(TipExceptie Identificator) {ListaInstrucţiuni } ... finally{ ListaInstrucţiuni }

Clasa Math

Metoda	Efect
double abs(x)	Valoare absoluta
double exp(x)	e^x
Double log(x)	ln x
double ceil(x)	Cel mai mic numar intreg convertit la double >= x
double floor(x)	Cel mai mare numar intreg convertit la double <=x
max(x,y)	Maximul dintre x si y

Metoda	Efect
min(x,y)	Minimul dintre x si y
double pow(x,y)	x^y
double sqrt(x)	$\sqrt{x}$
cos(x)	Cosinusul unghiului transmis ca parametru
double random()	Un numar real generat pseudoaleator subunitar ([0,1))

Metode supraincarcate

- **Semnătura unei metode** este o combinaţie dintre numele metodei şi tipul parametrilor în ordinea declarării lor în definirea metodei.

- Limbajul Java admite *supraîncărcarea* metodelor, adică metodele dintr-o clasă pot avea acelaşi nume, este suficient să difere prin semnătură (adică, prin tipul şi/sau numărul parametrilor).

- Când o metodă statică supraîncărcată este apelată se execută următorii paşi:
  1. se evaluează numărul şi tipul parametrilor actuali ai metodei;
  2. se caută metoda a cărei semnătură mapează strict numărul şi tipul parametrilor. Dacă nu este găsită o astfel de metodă, se converteşte tipul fiecărui parametru la cel mai apropiat şi se reia pasul 2.
  3. pasul 2 se termină cu executarea unei metode, în cazul în care este găsită sau, altfel, cu eroare de compilare.

Clasa JOptionPane

- public static String showInputDialog(Component componentaParinte, Object mesaj)
- public static String showInputDialog(Component componentaParinte, Object mesaj, Object valoareInitiala)
- public static String showInputDialog(Component componentaParinte, Object mesaj, String titlu, int tipMesaj)
- public static Object showInputDialog(Component componentaParinte, Object mesaj, String titlu, int tipMesaj, Icon icon, Object[] valoriSelectie, Object valoareInitiala)
- public static String showInputDialog(Object mesaj)
- public static String showInputDialog(Object mesaj, Object valoareInitiala)
- public static void showMessageDialog(Component componentaParinte, Object mesaj)

Metode recursive

- O **metoda recursiva** este o metoda care se autoapeleaza.

- Sunt două cazuri:
  - apelul este direct, adică metoda M are un apel la ea însăşi în corpul său;
  - apelul este indirect: o metodă M_1 apelează o metodă M_2, care apelează direct M_3, s.a.m.d. până când o metodă M_k din secvenţă apelează M_1.

Apelul recursiv al unei metode trebuie să fie condiţionat de o expresie boolean-a care să impiedice apelul în cascadă (la infinit); aceasta ar duce la o eroare de program – depăşirea stivei:

```
void p(   ){ //metoda recursiva
 p();  //apel infinit
}

static int calculeazaF(int n){
 if (n<=0)
  return 1;
  else return n* calculeazaF(n-1);
}
```