Slide 1

Les fonctions

Slide 2

-I. Les délégués

Slide 3

0. Résumé des épisodes précédents

Slide 4

Le noyau impératif : décl., affectation, séquence, test, boucle

Variables mutables et finales

e : `Var` $\longrightarrow$ `Ref`

m : `Ref` $\longrightarrow$ `Val`

mais finalement

e : `Var` $\longrightarrow$ `Val` ($\supseteq$ `Ref`)

m : `Ref` $\longrightarrow$ `Val`

Θ(t,e,m) = v

Σ(p,e,m) = m'

Slide 5

- $\Sigma(\{T\ x = t;\ q\},e,m) =$
  $\Sigma(q,(e+(x=r)),(m+(r=v)))$
  $\Sigma(\{final\ T\ x = t;\ q\},e,m) = \Sigma(q,e+(x=v),m)$
  r n'apparaît pas dans e et m et $v = \Theta(t,e,m)$
- $\Sigma(x=t;,e,m) = m+(e(x)=v)\ [v=\Theta(t,e,m)]$
- $\Sigma(\{p_1\ p_2\},e,m) = \Sigma(p_2,e,\Sigma(p_1,e,m))$
- $\Sigma(\text{if (b) } p_1 \text{ else } p_2,e,m) = \Sigma(p_1,e,m)$
  si $\Theta(b,e,m) = true$
  $\Sigma(\text{if (b) } p_1 \text{ else } p_2,e,m) = \Sigma(p_2,e,m)$
  si $\Theta(b,e,m) = false$
- $\Sigma(\text{while (b) } q,e,m) = \lim_n \Sigma(p_n,e,m)\ p_n = \ldots$

Slide 6

Les mystères de la déclaration

```
int x = 3; x = x + 1;
```

```
int x = 3; q
```

Pourquoi pas `int x = 3;` et une séquence ?

Slide 7

Les mystères de la déclaration

```
int x = 3; x = x + 1;
```

```
int x = 3; q
```

Pourquoi pas `int x = 3;` et une séquence ?

Éléments d'une séquence : exécutés dans le même env.

$\Sigma(\{p_1\ p_2\},e,m) = \Sigma(p_2,e,\Sigma(p_1,e,m))$

Or la déclaration modifie l'environnement de q

$\Sigma(\{T\ x = t;\ q\},e,m) = \Sigma(q,e+(x=r),m+(r=v))$

Slide 8

Un exercice

```
p = {{int x = 3; x = x + 1;} y = 8;}
e = [y = r]
m = [r = 7]
```

$\Sigma(p,e,m)$?

Slide 9

Faites ce que je dis, pas ce que je fais

```
{
   {
      int x = 3;
      x = x + 1;
   }
   y = 8;
}
```

```
p = {{int x = 3; x = x + 1;} y = 8;}
```

Idem toujours des {} dans un `while` ou un `if`

Slide 10

I. Les fonctions

Slide 11

```
System.out.print("Le RER en direction de ");
System.out.print("Saint-Rémy-lès-Chevreuse");
System.out.print(" partira à ");
System.out.println("8h50");
System.out.println();
System.out.println();
System.out.println();
System.out.print("Le RER en direction de ");
System.out.print("Massy-Palaiseau");
System.out.print(" partira à ");
System.out.println("8h55");
System.out.println();
System.out.println();
System.out.println();
```

Slide 12

```
static void sauterTroisLignes () {
 System.out.println();
 System.out.println();
 System.out.println();}
```

Slide 13

```
System.out.print("Le RER en direction de ");
System.out.print("Saint-Rémy-lès-Chevreuse");;);
System.out.print(" partira à ");
System.out.println("8h50");
sauterTroisLignes();

System.out.print("Le RER en direction de ");
System.out.print("Massy-Palaiseau");;);
System.out.print(" partira à ");
System.out.println("8h55");
sauterTroisLignes();
```

Slide 14

Le passage d'arguments

```
static void annoncerRer (final String d,
                         final String h) {
 System.out.print("Le RER en direction de ");
 System.out.print(d);
 System.out.print(" partira à ");
 System.out.println(h);
 System.out.println(); System.out.println();
 System.out.println();}

annoncerRer("Saint-Rémy-lès-Chevreuse","8h50");
annoncerRer("Massy-Palaiseau","8h55");
```

Le retour de valeur

Slide 15

```
a = 3;
b = 4;
c = 5;
d = 12;
u = Math.sqrt(a * a + b * b);
v = Math.sqrt(c * c + d * d);
```

Slide 16

```
static double hypothenuse  (final double x,
                            final double y) {
 return Math.sqrt(x * x + y * y);}

a = 3;
b = 4;
c = 5;
d = 12;
u = hypothenuse(a,b);
v = hypothenuse(c,d);
```

Slide 17

Un peu de syntaxe

Java :

```
static T f (final T1 x1, ..., Tn xn) p
```

Caml (arguments toujours finaux) :

```
let f x1 ... xn = t in p
```

```
let hypothenuse x y = sqrt(x *. x +. y *. y)
```

C :

```
T f (const T1 x1, ..., Tn xn) p
```

```
int f (const int x, int y) return x + 1;
```

Slide 18

L'instruction `return`

Caml :

```
let hypothenuse x y = sqrt(x *. x +. y *. y)
```

En Java et C la valeur à renvoyer doit être précédée de `return`

```
static double hypothenuse  (final double x,
                            final double y) {
 return Math.sqrt(x * x + y * y);}
```

`return` interrompt le déroulement de la fonction

Slide 19

```
static int signe (final int x) {
if (x < 0) return -1;
if (x == 0) return 0;
return 1;}
```

Les fonctions et les procédures

Slide 20

Une fonction peut

– effectuer une action (*e.g.* afficher quelque chose, modifier la mémoire)
– retourner une valeur

Une fonction qui ne retourne pas de valeur : une procédure

Certains langages (Pascal) : syntaxe différente

Java et C : type de retour remplacé par `void` (`void` pas un type)

Caml : une procédure retourne une valeur de type `unit`

Slide 21

Les fonctions et les procédures

Appel d'une fonction : expression

```
x = hypothenuse(3,4) + 8;
```

Appel d'une procédure : instruction

```
sauterTroisLignes();
```

Mais ... un appel de fonction peut aussi être une instruction

Slide 22

Les variables globales

```
int x;
x = 3;
x = 0;
```

```
static void reset () {x = 0;}
```

```
int x;
x = 3;
reset();
```

Slide 23

```
static int x;

static void reset () {x = 0;}

x = 3;
reset();
```

Slide 24

Le programme principal

```
static T1 x1 = t1 ;
...
static Tn xn = tn ;

static ... f1 (...) ...
...
static ... fn' (...) ...

p
```

Le programme principal

Slide 25

```
class Prog {
 static T1 x1 = t1 ;
 ...
 static Tn xn = tn ;
 static ... f1 (...) ...
 ...
 static ... fn' (...) ...
 public static void main (String [] args) {
 p}}
```

Slide 26

II. La fonction Σ

La dernière fois

Slide 27

La valeur d'une expression

Θ(t,e,m) = v

Ce qui se passe quand on exécute une instruction

Σ(p,e,m) = m'

Trois nouveautés

Slide 28

1. Outre une instruction, un environnement et une mémoire, on a besoin d'un environnement global pour les définitions de fonctions et les variables globales

Σ(p,e,m,G) = m'

Idem pour les expressions

Slide 29

2. Calculer la valeur d'une expression peut désormais modifier la mémoire : `f(4)`

```
static int f (int x) {
  n = n + 1;
  return 2 * x;}
```

$\Theta(t,e,m,G) = (v,m')$

Slide 30

3. Il faut traduire le fait que l'instruction `return` interrompt le déroulement de la fonction

Slide 31

3. Il faut traduire le fait que l'instruction `return` interrompt le déroulement de la fonction

Est-il toujours vrai que

$\Sigma(\{p_1\ p_2\},e,m,G) = \Sigma(p_2,e,\Sigma(p_1,e,m,G),G)$

?

Quid de `if (x < 0) return -1; return 1;` ?

Slide 32

3. Il faut traduire le fait que l'instruction `return` interrompt le déroulement de la fonction

Est-il toujours vrai que

$\Sigma(\{p_1\ p_2\},e,m,G) = \Sigma(p_2,e,\Sigma(p_1,e,m,G),G)$

?

Quid de `if (x < 0) return -1; return 1;` ?

$\Sigma(p,e,m,G)$ est désormais ou bien `(normal,m')` ou bien `(return,v,m')`

L'évaluation d'une expression

Slide 33

- Θ(x,e,m,G) = (m(e(x)),m), si x mutable,
- Θ(x,e,m,G) = (e(x),m), si x finale,
- Θ(c,e,m,G) = (c,m),
- Θ(t + u,e,m,G) = (v + w,m'')
  où (v,m') = Θ(t,e,m,G)
  et (w,m'') = Θ(u,e,m',G),
- idem pour les autres opérations

Slide 34

- Θ(f(t_1,...,t_n),e,m,G)?

– $\Theta(f(t_1,\ldots,t_n),e,m,G)$?

x_1, ..., x_n arguments formels et p le corps de f (dans G)

e′ environnement de variables globales de G

Slide 35

– $\Theta(f(t_1,\ldots,t_n),e,m,G)$?

x_1, ..., x_n arguments formels et p le corps de f (dans G)

e′ environnement de variables globales de G

$(v_1,m_1) = \Theta(t_1,e,m,G)$

$(v_2,m_2) = \Theta(t_2,e,m_1,G)$

$\ldots\ (v_n,m_n) = \Theta(t_n,e,m_{n-1},G)$

Slide 36

Slide 37

– $\Theta(f(t_1,...,t_n),e,m,G)$?

  x_1, ..., x_n arguments formels et p le corps de f (dans G)

  e' environnement de variables globales de G

  $(v_1,m_1) = \Theta(t_1,e,m,G)$

  $(v_2,m_2) = \Theta(t_2,e,m_1,G)$

  $... (v_n,m_n) = \Theta(t_n,e,m_{n-1},G)$

  $e''=e'+(x_1 = v_1)+ (x_2 = r_2)+ ...+ (x_n = r_n)$

  $m'' = m_n + (r_2 = v_2) + ... + (r_n = v_n)$

  Si $\Sigma(p,e'',m'',G)$ a la forme $(\texttt{return},v,m''')$

  alors (v,m''')

L'exécution des instructions

Slide 38

– déclaration, affectation, test, boucle : ras

– séquence : si $\Sigma(p_1,e,m,G) = (\texttt{normal},m')$

  alors $\Sigma(\{p_1\ p_2\},e,m,G) = \Sigma(p_2,e,m',G)$

  et si $\Sigma(p_1,e,m,G) = (\texttt{return},v,m')$

  alors $\Sigma(\{p_1\ p_2\},e,m,G) = (\texttt{return},v,m')$

Slide 39

– Appel de fonction idem expressions
  si Σ(p,e'',m'',G) = (normal,m''') alors
  (normal,m''')
  et si (return,v,m''') alors (normal,m''')
– si Θ(t,e,m,G) = (v,m'),
  alors Σ(return t;,e,m,G) = (return,v,m')

Slide 40

Un exemple

```
static double hypothenuse  (final double x,
                            final double y) {
 return Math.sqrt(x * x + y * y);}

public static void main  (String [] args) {
 a = 3;
 b = 4;
 u = hypothenuse(a,b);
 System.out.println(u);}
```

Et si a, b, u locales à main ?

Slide 41

Execices 2.3, 2.4, 2.5.

Slide 42

La prochaine fois :

Le passage par valeur et le passage par référence - la récursivité