INFO0101

INTRODUCTION À L'ALGORITHMIQUE ET À LA PROGRAMMATION

COURS 3

FONCTIONS ET PROCÉDURES

Pierre Delisle, Cyril Rabat, Christophe Jaillet, Jessica Jonquet et François Alin
Département de Mathématiques et Informatique
Septembre 2017

Exemple 1

- Calculer le maximum de a et b, puis de c et d

Exemple 1 - Remarques

- Même suite d'actions pour le calcul des 2 maxima
- Seules les variables utilisées sont différentes
- Dans les deux cas, le résultat est une valeur de type entier
- Principe « Diviser pour régner »
  - Chaque fonction résout un sous-problème
  - L'algorithme appelle ces fonctions
  - Mise en commun d'actions : facilité de conception

Fonction

- Bloc (suite) d'actions (instructions), nommé et éventuellement paramétré, retournant un résultat
- Lorsqu'une fonction est déclarée, on peut l'appeler par son nom
  - L'appeler (utiliser) autant de fois que nécessaire
- Renvoie toujours un résultat (une valeur)
  - Convention : retourner valeur
  - On associe un type à une fonction : type du résultat
- Exemple 2
  - Calcul du maximum en utilisant une fonction

Les données pouvant être utilisées dans une fonction

Variables locales

- Les variables déclarées dans une fonction sont dites « locales » (à cette fonction)
- Elles sont utilisables uniquement dans le corps de la fonction (la suite d'actions composant la fonction)
- Elles n'existent que pour cette fonction

Paramètres

- Les paramètres permettent de transmettre des valeurs à une fonction
  - L'action réalisée dépend des valeurs transmises
- x et y sont appelés les « paramètres formels » de la fonction *max*
  - Ils sont nommés (x et y)
  - Ils sont également typés (x et y sont de type entier)
- a et b sont appelés les « paramètres effectifs » du 1er appel
  - À chaque paramètre formel, on associe un paramètre effectif (du même type)
  - De même, c et d sont les paramètres effectifs du second appel

Appel de fonction - 1

- Déclaration → paramètres formels
- La signature d'une fonction est définie par
  - Son nom
  - Le type du résultat
  - La liste des paramètres formels avec leurs types associés

```
Fonction f (i : entier, j : entier, c : caractère) : réel
```

- Utilisation (appel) → Paramètres effectifs

- Lors d'un appel de fonction, le déroulement est le suivant :
  - Création d'autant de variables qu'il y a de paramètres formels, en respectant le type de chaque paramètre
  - Initialisation des variables créées en recopiant la valeur des paramètres effectifs
  - Les variables créées seront accessibles par le nom du paramètre formel associé
- Paramètres et variables locales
  - Seules les actions du corps de la fonction peuvent les utiliser
  - Existent uniquement pendant la durée d'exécution de la fonction
  - Leur valeur sera perdue après le retour de l'appel

Exemple 3
Trace d'exécution de l'algorithme du maximum de l'exemple 2

```
                Fonction max (x : entier, y : entier) : entier
                    Déclarations
                        Variables locales
                            m : entier
                    Début
{m1}                    Si x >= y Alors
{m1a.1}                     m ← x
                        Sinon
{m1b.1}                     m ← y
                        FinSi
{m2}                    retourner m
                    Fin
```

```
                Algorithme Maximum
                    Déclarations
                        Variables
                            a, b, c, d,
                            MaxAB, maxCD        : entier
                    Début
{1}                     a ← lire () //idem pour b, c, d)
{2}                     maxAB ← max(a, b)
{3}                     maxCD ← max(c, d)
{4}                     écrire(maxAB, maxCD)
                    Fin
```

Procédure

- Bloc (suite) d'actions (instructions), nommé et éventuellement paramétré, ne renvoyant pas de résultat
- Paramètres formels et paramètres effectifs
  - Similaires aux fonctions
- Les fonctions et les procédures structurent les algorithmes !

```
Procédure afficherMultiples (x, y, m : entier)
    Déclarations
        Variables locales
            z : entier
Début
    //Pseudo-code de afficherMultiples…
Fin
```

```
Algorithme Multiples
    Déclarations
        Variables
            n1, n2, multiple : entier
Début
    n1 ← lire ()
    n2 ← lire ()
    multiple ← lire ()
    afficherMultiples(n1, n2, multiple)
Fin
```

FONCTIONS ET PROCÉDURES EN JAVA

Fonctionnement

- Avec Java, en Info0101
  - Un programme est défini par une classe
- Fonctions et procédures
  - Définies au niveau de cette classe
- Liste des paramètres
  - Chaque paramètre est précédé par son type
  - Les paramètres sont séparés par une virgule
- Mot clé *return*
  - Permet de définir la valeur du résultat de la fonction
  - Si des instructions sont placées après le mot clé *return*, elles ne seront jamais exécutées
- La suite d'instructions
  - Est définie dans un bloc → entre { et }
  - Commence par la déclaration des variables locales

- Déclaration d'une procédure

```
public static void nomProcedure(liste_param) {
    suite_instructions ;
    //y compris les déclarations de variables locales
}
```

- Déclaration d'une fonction

```
public static type_de_retour nomFonction(liste_param) {
    suite_instructions ;
    //y compris les déclarations de variables locales
    return valeur ; //finit en renvoyant une valeur
}
```

Exemple de code avec fonction et procédure

```
class Appels {
    public static void main (String [] args) {
        int i = 1;
        while (i < 2)
            i = f(i+1, i-3);
        System.out.println("Dans le main : i = " + i);
    }  //fin du main (programme principal

    public static int f (int i, int j) {
        p (i + j);
        System.out.println("Dans f : i = " + i + " ; j = " + j);
        return (2 * i – j + 3);
    }  //fin de la fonction f

    public static void p (int j) {
        int i = j + 3;
        System.out.println("Dans p : i = " + i + " ; j = " + j);
    }  //fin de p
}
```

Feuille de TD # 3

- Exercices 1 à 8

PROCHAIN COURS :

TABLEAUX