Enseignement secondaire technique
Division technique générale

Notations à utiliser pour la désignation des identificateurs et la description des algorithmes à partir de l'année scolaire 2016/2017

Partie 1 - Conventions de noms

La notation des identificateurs se fait selon les conventions suivantes : [1]

- les identificateurs sont notés en anglais, (les commentaires en français)
- les identificateurs sont composés uniquement de lettres majuscules, minuscules, de chiffres et du symbole underscore '_'
- les identificateurs ne contiennent pas de caractères spéciaux (lettres accentuées etc) [2]
- l'identificateur d'une constante est écrite entièrement en lettres majuscules
- l'identificateur d'une classe commence toujours par une lettre majuscule
- l'identificateur d'une méthode, d'une variable, d'un paramètre ou d'un attribut commence toujours par une lettre minuscule. Si le nom se compose de plusieurs mots, la première lettre de chaque mot est écrite en majuscules, sauf pour le premier mot
- les identificateurs d'une classe ou d'un objet sont formés par des noms ou noms composés
- il est vivement recommandé que l'identificateur d'un paramètre commence par un 'p' [3]
- en principe les identificateurs avec une longue durée de vie ont des noms plus explicites que les identificateurs avec une courte durée de vie. Ainsi les variables locales ont souvent des noms assez courts tandis que les attributs et les paramètres ont des noms plus longs
- conventions de noms pour les méthodes :
  - les identificateurs des méthodes contiennent toujours un verbe
  - les identificateurs des accesseurs sont précédés du préfixe **get**
  - les identificateurs des manipulateurs sont précédés du préfixe **set**
  - les identificateurs des méthodes booléennes sont précédés du préfixe **is**
  - une méthode retournant une description textuelle de l'état actuel (des attributs) d'une instance d'une classe porte le nom **toString()** et elle est de la forme **public String toString() {…}**
- Les identificateurs des composants Swing ou AWT en NetBeans (-> cours de 12e et de 13e) se terminent par leur type (en omettant le préfixe 'J' pour les composants *Swing*).

Exemples :

- classes : `Clock, CreditCard`
- attributs : `minutes, numberOfBooks`
- variables: `i, n, count, sum`
- méthodes : `getRadius(), addHours(...)`
- paramètres : `pDiameter, pNumberOfBooks`
- composants : `startButton, resultLabel, divisorTextField`

1 Les conventions correspondent en principe aux conventions internationales usuelles en Java (→ voir p.ex.: http://geosoft.no/development/javastyle.html), les ajoutes ou exceptions à ces conventions sont explicitées ici.

2 Java accepte les identificateurs contenant des caractères spéciaux et notamment les lettres accentuées, mais des problèmes avec les pages de code subsistent lorsqu'on transfère le code d'un système vers un autre (Windows ↔ Linux ↔ MacOSX) ou aussi d'une application vers une autre (p.ex. : BlueJ ↔ NetBeans).

3 Cette convention permet aux élèves, débutants en programmation de mieux distinguer les paramètres des attributs et des variables. En outre (le mot clé 'this' n'étant pas encore prévu au programme de 3GIG) elle évite aussi les fausses affectations de la forme `hours=hours;` dans les constructeurs.

Partie 2 - Structogrammes

La description des algorithmes plus complexes se fait par des structogrammes. Cette représentation est à appliquer selon les conventions suivantes.

Notation des instructions de base

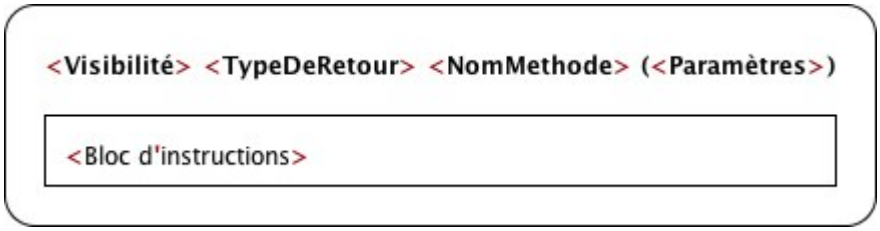
Élément	Structogramme	Code JAVA

écriture	écrire <Expression>	`System.out.println(<Expression>);`
		ou bien
		`System.out.print(<Expression>);`

lecture	lire <NomVariable>	*- pas prévu dans le cours -*
affectation	<NomVariable> ← <Expression>	`<NomVariable> = <Expression>;`

Notation des structures de base

Structure d'une méthode



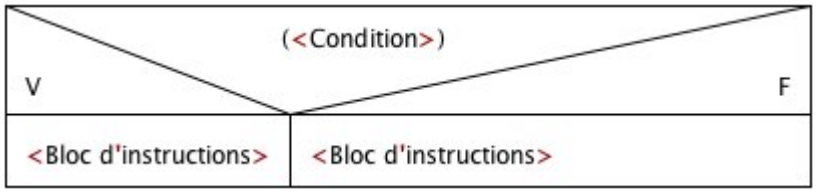
Exemple:

```
public double getSquareRoot(double pNumber)
{
      <Bloc d'instructions>
}
```

Structure séquentielle

Structure alternative

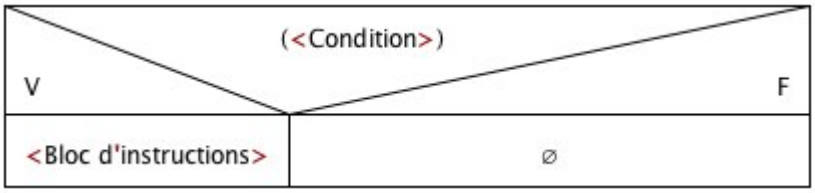
a) Structure alternative complète:



Code Java:

```
if (<Condition>)
{
      <Bloc d'instructions>
}
else
{
      <Bloc d'instructions>
}
```

b) Structure alternative réduite:

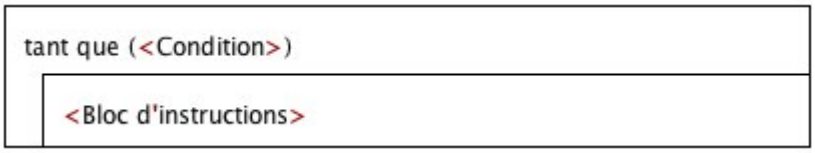


Code Java:

```
if (<Condition>)
{
      <Bloc d'instructions>
}
```

Structure répétitive

a) Variante *tant que*



Code Java:

```
while (<Condition>)
{
      <Bloc d'instructions>
}
```

b) Variante *pour*

Code Java:

si pas=1 ou si le pas n'est pas indiqué

```
for(int <NomVariable> = <ExprDébut> ; <NomVariable> <= <ExprFin> ; <NomVariable>++)
{
      <Bloc d'instructions>
}
```

ou bien

si pas= -1 :

```
for(int <NomVariable> = <ExprDébut> ; <NomVariable> >= <ExprFin> ; <NomVariable>--)
{
      <Bloc d'instructions>
}
```

Remarques générales:

- La déclaration des variables et des paramètres se fait comme dans le code source.
- En principe toutes les notations d'opérateurs, de méthodes et de constantes prédéfinis sont celles du langage de programmation.