LABORATOIRE 2 | *Informatique 1*
TYPES DE DONNEES

Hes·so VALAIS WALLIS
Haute Ecole d'Ingénierie
Hochschule für Ingenieurwissenschaften

Objectifs du laboratoire (2 périodes)

1. Le but premier de ce laboratoire est de se familiariser avec la structure d'un programme et l'environnement de développement *Eclipse*. Ce laboratoire a aussi pour but d'utiliser les différents types de données, d'expérimenter avec les variables ainsi que d'observer leurs limitations et leur conversion.
2. La durée estimée pour réaliser ce laboratoire est de deux périodes.
3. Vous pouvez trouver cette donnée sous forme électronique se trouve sur le site web du cours (http://inf1.begincoding.net/). Selon les laboratoires, vous y trouverez également un corrigé après celui-ci.

Partie 1 - Portée des variables

Tâche 1

Dans cette première tâche, vous allez voir les effets de la portée des variables et effectuer quelques petites expériences sur les portées.

1. Créez un projet *Eclipse* comme vu dans le premier laboratoire, nommez-le *Lab2*. N'oubliez pas de choisir l'option *"Don't create"* pour le fichier `module.info` lors de la création du projet.
2. Ajoutez une classe *Lab2_Task1.java* à votre projet et écrivez le programme ci-dessous :

```
public class Lab2_Task1 {
	public static void main(String args[]) {

	}
}
```

3. Exécutez votre programme.
4. Vérifiez qu'il n'y a pas d'erreur. Ce programme ne devrait rien faire du tout.
5. Ajoutez ce bloc de code dans votre programme actuel, dans le bloc le plus interne.

```
{
   int g=0;
   System.out.println(g);
}
```

6. Le code complet devrait ressembler à ce qui suit. Exécutez le programme et vérifiez que l'exécution affiche bien 0 dans la console :

```
public class Lab2_Task1 {
   public static void main(String args[]) {
      {
            int g = 0;
            System.out.println(g);
      }
   }
}
```

7. Faites la modification suivante dans le code que vous venez de coller. Que se passe-t-il et pourquoi ?

```
public class Lab2_Task1 {
    public static void main(String args[]) {
        {
              int g = 0;
        }
        System.out.println(g);
    }
}
```

8. Faites la modification suivante et expliquez ce qui se passe et pourquoi

```
public class Lab2_Task1 {
    public static void main(String args[]) {
        int g = 0;
        {
              System.out.println(g);
        }
    }
}
```

9. Écrivez maintenant un programme permettant de lire trois valeurs entières depuis la console (en vous inspirant du laboratoire 1). Vous devrez ensuite afficher sur la console la plus petite des trois valeurs entrées. Utilisez pour cela l'opérateur ? : vu en cours.

Partie 2 - Types de données et précision de calcul

Comme vu durant le cours sur les types, plusieurs types de variables sont utilisables autant pour les nombres entiers que pour les nombres à virgule flottante. Le choix du type d'une variable est toujours un compromis entre la taille qu'elle va prendre en mémoire, le temps de calcul et la perte de précision de calcul. Les exercices suivants ont pour but de montrer les limites des différents types de variable.

Tâche 2 (Précision de calcul)

1. Ajoutez une nouvelle classe, nommée `Lab2_Task2`, et ajoutez-y le code suivant :

```
public class Lab2_Task2 {
    public static void main(String args[]) {
        float a, b, c;
        a = 1f;
        b = 1e+8f;
        c = 1e17f;
        System.out.println(((a + b) * c - b * c) / c);
    }
}
```

2. Le résultat du calcul donne normalement :

$$\frac{(a+b)\cdot c - b\cdot c}{c} = a$$

3. Compilez et exécutez votre programme.
4. Vérifiez qu'il y a bien une différence entre le résultat affiché et le résultat attendu.

5. Expliquez cette différence.
6. Modifiez maintenant l'assignation des trois variables en utilisant des littéraux *double* plutôt que *float.* Le programme résultat est le suivant :

```
public class Lab2_Task2{
    public static void main(String args[]) {
        double a, b, c;
        a = 1;
        b = 1e+8;
        c = 1e+17;
        System.out.println(((a + b) * c - b * c) / c);
    }
}
```

7. Exécutez et expliquez le résultat.

Tâche 3 (Signe des nombres entiers)

1. Ajoutez une nouvelle classe, nommée `Lab2_Task3`, et ajoutez-y le code suivant :

```
public class Lab2_Task3 {
    public static void main(String args[]) {
        byte toto = (byte) 127;
        System.out.println(toto);
        toto = (byte) (toto + 1);
        System.out.println(toto);
    }
}
```

1. Exécutez le programme et expliquez le résultat.
2. Modifiez la déclaration de la variable *toto* et les castings, en utilisant le type *short*, plutôt que le type *byte*.
3. Exécutez le programme et expliquez le résultat.
4. Modifiez maintenant l'assignation de la variable *toto* en y allouant la valeur 2^{15}-1, plutôt que la valeur 127. Vous pouvez utiliser la fonction `Math.pow(2, 15)`. L'assignation résultante est la suivante :

```
short toto = (short)(Math.pow(2, 15)-1);
```

5. Exécutez le programme et expliquez le résultat.
6. Utilisez maintenant les valeurs littérales suivantes d'assignation : 2^{16} et 2^{17}. Expliquez le résultat.
7. Assigner maintenant les valeurs hexadécimales suivantes : `0x7FFF`, `0xFFFF` et `0xFFFFF`.
8. Exécutez et expliquez les résultats.

Tâche 4 (Caractères Unicode et casting)

1. Ajoutez une nouvelle classe, nommée `Lab2_Task4`, et ajoutez-y le code suivant :

```
public class Lab2_Task4 {
    public static void main(String args[]) {
        char toto = 'c';
        System.out.println(toto);
        System.out.println((int)(toto));
    }
}
```

2. Exécutez le programme et expliquez les résultats.

3. Que se passe-t-il si vous mettez `char toto = (char) 24661` ?
4. Retrouvez ce symbole sur http://www.unicode-table.com/

Partie 3 - Séquence d'instructions

Le bout de texte suivant

```
public static void main(String args[])
```

se nomme le *point d'entrée* du programme. Cela signifie que votre programme *commence ici*.

La première instruction contenue dans le bloc suivant ce bout de texte sera donc première exécutée. A l'intérieur de ce bloc, la séquence d'instructions sera exécutée dans l'ordre où elles sont écrites.

"TextTools"

Comme nous ne sommes qu'au début du cours de *Java*, vous n'avez pas encore tous les outils pour écrire des programmes complets. De ce fait, nous vous fournissons sur le site web du cours le fichier *TextTools.java* qui vous fournira une partie du code nécessaire pour cet exercice. Vérifiez que le fichier est bien dans le même répertoire que votre projet et n'oubliez pas de rafraîchir Eclipse (F5) pour afficher ce fichier. Le fichier *TextTools* (on parle normalement de **classe**) vous permet d'entrer des valeurs (texte, nombres…) au clavier et d'effectuer des opérations sur du texte. Cette classe contient des fonctions que vous pouvez utiliser de la manière suivante :

```
TextTools.nom_de_la_fonction(arguments éventuels)
```

Ainsi, par exemple, pour utiliser la fonction `deleteVowels(String s)` de la classe *TextTools*, il faut écrire :

```
TextTools.deleteVowels("Hello");
```

Pour connaître les différentes possibilités que vous offre la classe *TextTools*, les différentes fonctions possibles sont affichées lorsque vous tapez *TextTools* suivi d'un point dans *Eclipse*.

Tâche 5

1. Créez une nouvelle classe `Lab2_Task5` et assurez-vous qu'elle inclue les instructions `readText()` et `System.out.println(String s)` comme indiqué ci-dessous :

```
public class Lab2_Task5 {
    public static void main(String args[]) {
        String s = TextTools.readText();
        System.out.println(s);
    }
}
```

2. Exécutez ce code.
3. Analysez le comportement de cette application et mettez les commentaires correspondants à chaque ligne (qui indiquent ce que font chaque fonction).

Tâche 6

1. Complétez le bloc d'instructions pour réaliser la séquence suivante :
    a. Mémoriser une chaîne de caractères.
    b. Afficher le texte mémorisé.
    c. Mettre la première lettre en majuscule.
    d. Afficher le texte modifié.
2. Compilez et exécutez.

Tâche 7

1. Modifiez le bloc d'instructions pour réaliser la séquence suivante :
    a. Mémoriser une chaîne de caractères
    b. Mettre la chaîne mémorisée en majuscules
    c. Afficher la chaîne mémorisée
    d. Afficher la chaîne en inversant tous les caractères (sans changer la chaîne mémorisée)
    e. Afficher "Hello World"
    f. Mettre la chaîne mémorisée en minuscules
    g. Afficher la chaîne mémorisée
    h. Enlever les consonnes de la chaîne mémorisée
    i. Afficher la chaîne mémorisée
    j. Mémoriser une nouvelle chaîne de caractères
    k. Crypter la nouvelle chaîne de caractères
    l. Afficher la chaîne cryptée
    m. Décrypter la chaîne de caractères et l'afficher (en une seule ligne de code)
2. Compilez et exécutez

Question optionnelle : Expliquez la méthode de cryptage.

Tâche 8

1. Écrivez la séquence qui met en majuscules la première et la dernière lettre d'une chaîne de caractères.
2. Vérifiez que votre programme fonctionne correctement en l'essayant sur différentes chaînes de caractère.