Feuille de TP N°01

Exercices simples pour commencer

Exercice 1 *Premier pas avec Eclipse*

a) Lancer Eclipse.

b) **Premier projet** : Créer un nouveau projet (`File` → `New` → `Project`).
- 1ère fenêtre : Select a wizard, sélectionner Java project
- 2ème fenêtre : donner un nom au projet puis Finish

Le projet sera placé à l'emplacement proposé par défaut par Eclipse (un sous-répertoire portant le nom du projet sera créé dans votre répertoire `Workspace`).

c) **Premier package**
- Dans la partie (Package Explorer), sélectionner le projet puis créer un nouveau package (clic droit, `New` → `Package`) Donner un nom à ce package : tp1

d) **Première classe**
- Dans la partie (Package Explorer), sélectionner le package tp1 puis créer une nouvelle classe (clic droit, `New` → `Class`)
  Donner un nom à la classe : `Prog` et vérifier que la case
  `public static void main(String[] args)` est bien cochée.
  Votre première classe est créée !

```
package tp1;
//-----------------------------------------
// First Program
//-----------------------------------------
public class Prog {
        public static void main(String[] args) {
                System.out.println("Bonjour")
        }
}
```

Profiter de cette occasion pour vous familiariser avec les fonctions principales de l'éditeur. Consulter l'aide disponible dans Eclipse ainsi que les différentes possibilités de configurer cet éditeur.

a) Sauver cette classe, et regarder dans l'onglet `Problems` en bas. Eclipse permet de compiler le code "à la volée" (c.-à-d. au fur et à mesure que le code est saisi) et affiche les résultats de la compilation dans cet onglet. Il est possible, en faisant un clic sur un message d'erreur, d'obtenir plus d'informations sur celle-ci et, notamment, des suggestions pour la corriger.

b) Corriger ce programme pour obtenir l'affichage correct.

c) Pour faire tourner le programme : clic droit sur `Prog` dans le Package Explorer puis : `Run as→ Java application`

Remarque 1 *Une classe est exécutable si et seulement si elle comporte une méthode ayant exactement la signature* `public static void main(String[] args)`

À l'aide de l'explorateur (Windows Explorer), aller voir le contenu de votre Workspace.
– Que contient-il ?
– Où se trouve le fichier source de votre premier programme, comment se nomme-t-il ?
– Où se trouve le résultat de la compilation ?

Exercice 2 *Arguments de la ligne de commande*

Les arguments de la ligne de commande sont représentés par un tableau de chaînes de caractères, conventionnellement nommé `args`, qui est l'unique paramètre de la méthode `main(String[]args)` : `args[0]` est le premier argument, `args[1]` le second, etc. On obtient le nombre d'éléments du tableau par `args.length`.
Lors de l'invocation de `main`, les éléments de ce tableau, `args[0]` ... `args[args.length-1]`, sont initialisés par les chaînes de caractères (mots séparés par des espaces) figurant sur la ligne de commande.

a) Modifier la classe `Prog` pour que le programme produise l'affichage : `Bonjour Toto !` (`Toto !` est passé en ligne de commande)
   – Que se passe-t-il si on ne donne pas d'argument sur la ligne de commande ?
   – À l'aide d'une conditionnelle, résoudre ce problème.

b) Modifier maintenant le programme pour que pour chaque nom donné sur la ligne de commande, il affiche une ligne de salutation.

c) La méthode `Integer.parseInt (chaîne)` convertit le paramètre chaîne en un entier.
   – Étant donné deux nombres entiers fournis en ligne de commande, modifier le programme pour qu'il les affiche puis affiche leur somme.
   – Modifier la classe `Prog` pour que le programme produise l'affichage de la somme de `n` entiers lus en ligne de commande.

Remarque 2 *Pour lancer ce programme depuis Eclipse en lui communiquant une suite d'arguments sur la ligne de commandes, il vous faut procéder comme suit :*
– *Cliquez sur la flèche-bas à la droite du bouton* `Run`.
– *Sélectionner l'item* `Run Configurations...` *dans le menu contextuel*
– *La fenêtre de configuration des exécutions est affichée*
– *Dans cette fenêtre, sélectionnez l'onglet* `Arguments`
– *Dans le champ* `Program Arguments` *écrivez les arguments (séparés par des espaces) que vous souhaitez transmettre à votre programme lors de son exécution*
– *Lancez l'exécution en cliquant sur le bouton* `Run`

Exercice 3 *Expressions booléennes*

Traduire sous forme d'expression booléenne les propositions suivantes :

a) la variable x contient une valeur numérique comprise entre 3.5 et 7 ;

b) la variable x contient une valeur strictement négative ou bien une valeur supérieure à 5 ;

c) l'une des deux propositions suivantes est vraie :
   - la variable y contient une valeur divisible par 2 ;
   - la variable y ne contient pas une valeur divisible par 3.

d) une et une seule des deux propositions suivantes est vraie :
   - la variable x n'est pas nulle ;
   - la variable z contient la même valeur que la variable x ;

e) soit le contenu de la variable a est divisible par celui de b, soit le contenu de la variable b est divisible par celui de a.

Exercice 4 *Instructions de contrôle - Sélection*

Un examen comporte trois épreuves notées `n1`, `n2`, `n3`, pondérées par trois coefficients `c1`, `c2`, `c3`, fixés à priori. Ecrire un programme qui saisit au clavier ces trois notes (voir la classe `Scanner` en annexe), calcule la moyenne pondérée de ces trois notes, puis affiche la mention correspondante : **Ajourné** ($n < 10$), **Passable** ($10 \leq n < 12$), **ABien** ($12 \leq n < 14$), **Bien** ($14 \leq n < 16$), **TBien**($n > 16$).

Exercice 5 *Instructions de contrôle - Boucles*

Ecrire un programme qui calcule la valeur de $n!$. On fera déterminer par l'ordinateur un entier `Fmax` tel que si `n > Fmax`, la valeur de $n!$ est supérieure à `Integer.MAX_VALUE`.

Exercice 6 *Méthodes de classe*

Indiquer le type du résultat de chacun des appels suivants (Voir l'annexe pour la description de la classe `Math`) :
`Math.sqrt(-2)`, `Math.sqrt(4)`, `Math.sqrt(4l)`, `Math.sqrt(2f)`,
`Math.max(2l,3)`, `Math.max(4f,3d)`, `Math.min(2.5,-3)`,
`Math.abs(-2.5)`, `Math.abs(-2)`, `Math.abs((byte)-4)`.

Ecrire une méthode qui calcule la fonction `f` de $\mathbb{R}$ dans $\mathbb{R}$ définie par $f(x) = \sqrt{sin^2(x) + \pi}$.

Exercice 7

Pour tester la qualité de la méthode `java.lang.Math.random()`, écrire un programme qui tire `n` nombres au hasard (par exemple, `n = 100 000`) et qui calcule la moyenne et l'écart-type de la suite ainsi obtenue.
Pour obtenir des informations sur la méthode `random()` consulter la documentation en ligne de l'API (s'initier à l'utilisation de la documentation de l'API est un des buts de cet exercice).

Rappel de formules :
Si `n` est le nombre de termes :
- la moyenne est $\bar{x} = \frac{1}{n} \sum_i x_i$
- l'écart-type est la moyenne quadratique des écarts à la moyenne $\bar{x}$. On le note habituellement s (de l'anglais *standard deviation*) : $\sqrt{\frac{1}{n} \sum_i (x_i - \bar{x})^2}$

Exercice 8

Écrire une méthode de signature
`public static int askBirthYear()` qui va demander à l'utilisateur d'entrer son année de naissance sur l'entrée standard et doit la renvoyer. On va pouvoir faire :

```
public static void main (String[] args) {
  System.out.println ("Vous␣êtes␣né␣en␣" + askBirthYear());}
```

Voici un exemple d'exécution de ce programme :

```
Quelle est votre année de naissance : euh...
Quelle est votre année de naissance : sais plus
Quelle est votre année de naissance : 1984
Vous êtes né en 1984
```

Tant que l'utilisateur n'entre pas un entier, la méthode doit donc lui demander quelle est son année de naissance. (Voir la méthode `static boolean hasNextInt()` - Annexe).

Exercice 9 *Les tableaux*

Écrire une méthode qui reçoit en paramètre deux tableaux de caractères de taille identique et qui retourne un tableau deux fois plus grand comprenant sous forme alternée les caractères des deux tableaux d'origine.

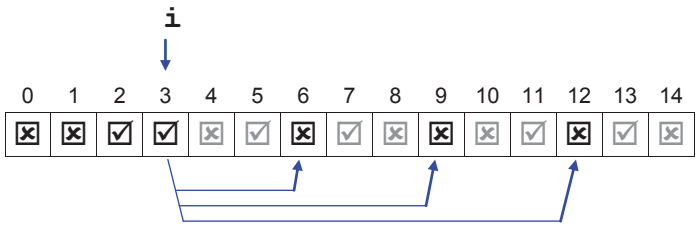
Exemple : {'a','b','c'} et {'x','y','z'} → {'a','x','b','y','c','z'}

Exercice 10

Un nombre premier est un entier naturel qui a exactement 2 diviseurs (1 et lui-même). Écrire une méthode qui retourne un tableau de `n` booléens, où la case d'indice `i` indique si `i` est un nombre premier.

Algorithme (en pseudo-code) pour remplir ce tableau :

Initialiser à faux les cases `0` et `1`, et à vrai toutes les autres cases
pour chaque case `i` dans l'ordre croissant
si cette case est à vrai
mettre à faux toutes les cases d'indice multiple de `i`



Bon travail !