TP 1

1. Hello World !

Dans un premier temps, créez un fichier que vous nommerez **HelloWorld.java**
Une fois cela fait, je vous invite à y copier le code ci-dessous :

```
class HelloWorld {
	public static void main(String[] args) {
		System.out.println("Hello World!");
	}
}
```

Utilisez la commande suivante dans un terminal afin de compiler le programme :

```
javac HelloWorld.java
```

Une fois le programme compilé, vous verrez qu'il y a un nouveau fichier (**HelloWorld.class**) dans votre répertoire. Ce fichier est généré par Java à la compilation de votre programme, il contient du bytecode Java (langage spécifique à Java) qui sera analysé et exécuté lorsque vous lancerez la commande suivante :

```
java HelloWorld
```

2. Algèbre de base

Récupérez le fichier "Tp1Exo2.java" et familiarisez vous avec les opérations de base.

3. Fibonacci

Rappels :

La déclaration d'une fonction se fait de cette manière :

```
public static type maFonction(type maVariable) {
        // Ici le code de la fonction
}
```

type est le type de retour de la fonction (si votre fonction ne retourne rien, mettre **void**).
type est le type du paramètre donné en entrée à la fonction.

Si vous voulez appeler cette fonction depuis le ***main*** :

```
class HelloWorld {
        public static void main(String[] args) {
                int maVariable = 0;
                double resultat = maFonction(maVariable);
        }
}
```

Dans cet exemple, **maFonction** prend en paramètre un int et retourne un double.

Le but de cet exercice est de réaliser une fonction qui calcule le n-ième terme de la suite de Fibonacci pour une valeur donnée en entrée.

Pour rappel, les deux premiers termes sont 0 et 1. Ensuite, chaque terme successif est la somme des deux termes précédents. Ainsi 0+1=1, 1+1=2, 1+2=3, 2+3=5, 3+5=8, etc.

Comme c'est la coutume, nous noterons par Fn le n-ième terme de cette suite, en commençant par n=0. Ainsi donc, la suite de Fibonacci F0, F1, F2, … peut être entièrement définie par les formules suivantes :

$$F0 = 0$$
$$F1 = 1$$
$$Fn = Fn-2 + Fn-1 \text{ pour } n \geq 2$$

Source : http://images.math.cnrs.fr/Mysteres-arithmetiques-de-la-suite-de-Fibonacci.html

4. Chaînes de caractères et boucles

Récupérez le fichier "Tp1Exo4.java" et codez la méthode "estPalindrome".

Pour comparer deux String utiliser la méthode **equals**.

5. Les tableaux

Avant de vous lancer dans l'exercice, ouvrer le fichier **Tableau.java** et lisez-le.
Écrire une fonction qui fait un tri rapide sur un tableau.

La méthode consiste à placer un élément du tableau (appelé pivot) à sa place définitive, en permutant tous les éléments de telle sorte que tous ceux qui sont inférieurs au pivot soient à sa gauche et que tous ceux qui sont supérieurs au pivot soient à sa droite.
Cette opération s'appelle le partitionnement. Pour chacun des sous-tableaux, on définit un nouveau pivot et on répète l'opération de partitionnement. Ce processus est répété récursivement, jusqu'à ce que l'ensemble des éléments soit trié.
Concrètement, pour partitionner un sous-tableau :

- le pivot est placé à la fin (arbitrairement), en échangeant avec le dernier élément du sous-tableau ;
- tous les éléments inférieurs au pivot sont placés en début du sous-tableau ;
- le pivot est déplacé à la fin des éléments déplacés

Source : https://fr.wikipedia.org/wiki/Tri_rapide

6. Mini projet : "Plus ou Moins"

Le but de cet exercice est de réaliser un petit jeu qui s'appelle "Plus ou moins". C'est un jeu où il faut trouver le nombre qui a été tiré au sort par le programme avant que le nombre d'essai arrive à 0. Il devra répondre aux spécifications suivantes :

- Tout d'abord, le programme devra tirer au sort un nombre entre 1 et 100 qui sera le nombre à trouver.

- Il vous demandera ensuite de deviner le nombre. Vous entrez donc un nombre entre 1 et 100.

- Le programme compare le nombre que vous avez entré avec le nombre qu'il a tiré au sort. Il vous dit si le nombre voulu est supérieur ou inférieur à celui que vous avez entré. Si votre nombre n'est pas le bon, il décrémente le nombre d'essai.

- Puis l'ordinateur vous redemande le nombre.

Et ainsi de suite, jusqu'à ce que vous trouviez le bon nombre ou que le nombre d'essai arrive à zéro. Afin de lire une entrée depuis le terminal, vous pouvez utiliser la classe Scanner :

```
import java.util.Scanner;
Scanner scanner = new Scanner(System.in);
int value = scanner.nextInt();
```

Pour réaliser ce projet, créer un nouveau dossier dans lequel vous allez faire cet exercice. Lors que votre programme répondra aux spécifications et sera fonctionnel, je vous propose de faire un **jar** (une archive exécutable et portable). Un jar n'est rien d'autre qu'une archive qui contient les fichiers nécessaires à l'exécution d'un programme. Avant toute chose, créer un fichier que vous nommerez **MANIFEST.MF** :

```
Main-Class: MaClass
```

Remplacer MaClass par le nom de votre classe, **ne pas oublier le retour à la ligne**. Le fichier manifest permet d'indiquer la classe qui sera lancée à l'exécution du programme, il permet aussi de d'indiquer le dossier qui contient les ressources utilisées par un programme. Après avoir compilé votre programme, lancez la commande suivante :

```
jar cvmf MANIFEST.MF PlusOuMoins.jar *.class
```

Pour lancer l'exécutable :

```
java -jar PlusOuMoins.jar
```

7. Objets et cycle de vie

Récupérez le fichier "Singleton.java" et implémentez le fameux design pattern aussi simplement que possible.