

TP n°1

Les structures de contrôles

Exercice : Les valeurs Min et Max

Écrire un programme qui affiche dans la console “n nombres” choisis aléatoirement entre 0 et 100.

Commencez par demander à l'utilisateur combien de nombre est-ce qu'il veut.

Puis tirez autant de nombre aléatoire. Pour cela, utilisez la méthode *nombreAleatoire* fournie dans la classe *Utile.java*. Cette méthode retourne une valeur entière comprise entre les paramètres qui lui sont donnés.

```
int valeur = nombreAleatoire(12, 15);
```

Dans cet exemple, la variable *valeur* peut être initialisée par 12, 13, 14 ou 15.

Ajoutez à votre programme le code permettant de connaître le min ou la max des “n nombres”. Attention : il n'est pas demandé d'utiliser un tableau.

Exemple de résultat :

```
Combien de nombre voulez-vous ? 10
Le nombre n°1 est 43
Le nombre n°2 est 83
Le nombre n°3 est 30
Le nombre n°4 est 38
Le nombre n°5 est 6
Le nombre n°6 est 97
Le nombre n°7 est 66
Le nombre n°8 est 34
Le nombre n°9 est 4
Le nombre n°10 est 59

Le maximum est 97 et le minimum est 4.
```

1° - Réaliser cet algorithme.

2° En déduire sa complexité.

Exercice : Chiffres d'un nombre entier

En utilisant les instructions / et %, écrire un programme qui demande un entier et affiche le chiffre des unités, le chiffre des dizaines, le chiffre des centaines et ainsi de suite tant qu'il y a des chiffres.

Le programme affiche comme dernière ligne le nombre saisi mais à l'envers.

Exemple :

```
Saisir un nombre : 98532
--> 2
--> 3
--> 5
--> 8
--> 9
A l'envers le nombre est 23589
```

1° - Réaliser cet algorithme.

2° En déduire sa complexité.

Exercice : nombres premiers

Écrire un programme qui demande à l'utilisateur un entier naturel $n < 1000$ et qui affiche tous les nombres premiers inférieurs ou égaux à n .

Commencez par écrire une méthode qui détermine si un entier est premier (on rappelle qu'un nombre est premier s'il a deux diviseurs et deux seulement : 1 et lui-même).

Exemple :

```
Saisir un nombre : 18
--> 2
--> 3
--> 5
--> 7
--> 11
--> 13
--> 17
```

1° - Réaliser cet algorithme. Optimisez cet algorithme au maximum (pas en enlevant vos commentaires et en mettant tout sur une ligne).

2° En déduire sa complexité.