L'École Normale Supérieure de l'Enseignement Technique de Mohammedia

Master SDIA

Technologies du Web & Web Sémantique

# — TD n ∘ 1: Introduction à JavaScript —

## Types simples, variables et instructions de base

**Objectifs:** Le but de ce TD est de vous faire expérimenter les constructions de base du langage JavaScript (types simples, déclarations de variables, instructions de contrôle, itérations) qui sont très proches syntaxiquement de celles utilisées par le langage C.

## Exercice 1 : Conversion de températures

En utilisant la formule $\text{tempC}=(5/9)(\text{tempF}-32)$ écrire en langage JavaScript une fonction degreC qui affiche sa valeur en degrés centigrades ou degrés Celsius d'une température exprimée en degrés Fahrenheit

Exemples d'exécution du programme (en vert les valeurs introduites par l'utilisateur) :

```
Une température en Fahrenheit : 0.0

cette température équivaut a -17.8 degrés Celsius

Une température en Fahrenheit : 60.0

cette température équivaut a 15.6 degrés Celsius
```

## Exercice 2 : Conversion de durées

Écrire une fonction **hjms** en langage JavaScript qui pour un nombre de secondes donné calcule et renvoi son équivalent en nombre de jours, d'heures, de minutes et de secondes.

exemples d'exécution du programme :

```
Une durée en secondes : 235789

cette durée équivaut à 2 jours 17 heures 29 minutes 49 secondes

Une durée en secondes : 567231

cette durée équivaut à 6 jours 13 heures 33 minutes 51 secondes
```

## Exercice 2-bis : Améliorer le programme de conversion de durées

Si ce n'est déjà fait, améliorez votre programme **hjms** de sorte que lorsqu'une valeur (nombre de jours, d'heures , de minutes ou de secondes) est nulle elle n'apparaisse pas dans l'affichage, et que si elle vaut 1 l'unité soit affichée au singulier (sans s) comme dans l'exemple ci-dessous.

```
Une durée en secondes : 3621

Cette durée équivaut à 1 heure 21 secondes
```

## Exercice 3 : Classer 3 nombres

Écrire un programme **troisNombres** qui renvoi l'ordre croissant (du plus petit au plus grand) de 3 nombres.

```
1er nombre : 14
2ème nombre : 10
3ème nombre : 17
les nombres dans l'ordre croissant : 10 14 17
```

## Exercice 4 : Affichage de motifs - escaliers

Écrire un programme qui affiche un motif triangulaire dont la taille est fixée par une valeur.

Exemple de trace d'exécution:

```
Un motif  de taille= 7
*
**
***
****
*****
******
```

a) écrire une fonction **triangle1** affichant ce motif en utilisant uniquement des instructions tant que (while()).

b) écrire une fonction **triangle2** affichant ce motif en utilisant uniquement des instructions pour (for).

## Exercice 4-bis : Affichage de motifs - pyramides

Même exercice que le précédent mais le motif affiché n'est plus un triangle mais une pyramide (voir ci-dessous) et le choix des instructions pour le réaliser est laissé à votre jugement.

Exemple de trace d'exécution (en vert les valeurs introduites par l'utilisateur):

```
donnez taille du motif : 7
     *
    ***
   *****
  *******
 *********
***********
*************
```

## Exercice 5 : Tester si un nombre est premier

Un nombre est n premier si il a seulement deux diviseurs : 1 et n.

Écrire un programme **Premier** qui permet de tester si un nombre introduit par l'utilisateur est premier ou non.

Exemple de trace d'exécution (en vert les valeurs introduites par l'utilisateur):

```
Un entier positif  : 7

7 est un nombre premier



donnez un entier positif  : 25

25 n'est pas un nombre premier, il est divisible par 5
```

## Exercice 6 : Suite de Fibonacci

La suite de Fibonacci est définie par la formule de récurrence suivante:

$$u_0 = 0, u_1 = 1, \forall n \in \mathbb{N} \quad u_{n+2} = u_{n+1} + u_n$$

a) Ecrire un programme **Fibo1.** qui permet de calculer le n$^{ième}$ terme de la suite de Fibonacci, n étant fixé par l'utilisateur.

b) Ecrire un programme **Fibo2** qui permet d'obtenir la valeur et le rang du premier terme de cette suite supérieure à une valeur donnée par l'utilisateur.

## Exercice 7 : Valeur approchée de la $\sqrt{\ }$ d'un nombre réel positif

On considère un nombre réel positif A ; on sait que la suite $(u_{n\,\in\mathbb{N}})$ définie par un réel $u_0$ positif et par la relation de récurrence $u_{n+1} = \frac{1}{2}(u_n + \frac{A}{u_n})$ (pour n > 0) converge vers $\sqrt{A}$. On suppose le nombre A compris entre 1 et 100, et on prend $u_0 = \frac{A}{2}$.

Pour obtenir une valeur approchée de racine carrée de A, on cherche le premier terme $u_n$ tel que $|u_n^2 - A| < 10^{-5}$. Le nombre trouvé est une valeur approchée de racine carrée de A ( en effet $||u_n^2 - A| < 10^{-5}$.  implique que $|u_n - \sqrt{A}| < 10^{-5}$.

Écrire une fonction JavaScript **Raca1** qui permet de calculer et d'afficher la valeur approchée de la racine carrée de A définie ci-dessus.

Exemple de l'état de l'écran obtenu par exécution du programme **Raca1** :

```
Pour un nombre A entre 1 et 100: 19.23

Valeur approchée de la racine carrée  = 4.385202389856321
```