# Exercice 6

# Manipulation des chaines de caractères, des nombres et tableaux

**Remise : avant la fin de la journée du vendredi 25 mars 2022**

1. **Partie 1**

Un clic sur le bouton ‘Traiter la chaine’ doit effectuer les opérations suivantes avec la phrase inscrite dans la zone de texte.

Remarquez qu’une zone de texte est créée à l’aide de la balise <input>. Cette zone de texte doit être utilisée par l’usager pour inscrire la phrase à traiter.

Dans le code javascript, la phrase inscrite par l’usager est récupérée grâce à **.value** qui permet de connaître le contenu de la zone de texte.

Utilisez maintenant la variable phrase pour effectuer le traitement qui suit :

* 1. Étant donné qu’il est possible que l’usager inscrive des espaces au début et à la fin de la phrase, utilisez ***.trim*** pour enlever ces espaces.
  2. Utilisez maintenant ***.length*** pour calculer le nombre de caractères qu’il y a dans la phrase. Dans le paragraphe *longueurChaine*, faites afficher ‘Votre phrase contient *N* caractères’ où N indique le nombre de caractères.
  3. Testez le fonctionnement.
  4. Dans le paragraphe *positionAllo*, faites afficher ‘Le mot Allo se trouve à la position N’ où N représente la position où débute le mot Allo. Pour trouver cette position, utilisez ***.indexOf()****.*
  5. Testez le fonctionnement.
  6. Il faut maintenant calculer le nombre d’espaces qu’il y a dans la phrase et le faire afficher dans le paragraphe *nombreEspaces*.

Trucs: Faites une boucle (utilisez ***.length***) et utilisez les ***[ ]*** pour accéder à un seul caractère à la fois et vérifier s’il s’agit d’un espace (″ ″ représente un espace).

* 1. Testez le fonctionnement.

1. **Partie 2**

Un clic sur le bouton ‘Démo math’ doit additionner 2 nombres entrés par l’usager.

* 1. Utilisez l’instruction ***prompt*** pour demander 2 nombres à l’usager et les mettre dans les variables *N1* et *N2*.
  2. Étant donné qu’une valeur entrée par l’usager est toujours considérée comme une string alors il faut convertir N1 et N2 pour qu’ils deviennent des nombres. Faites-le avec ***parseInt****()*.
  3. Utilisez aussi ***isNaN*** pour vérifier si les valeurs entrées par l’usager sont vraiment des nombres (utilisez les valeurs que vous venez de convertir). Si tel n’est pas le cas, faites afficher un message d’erreur dans le paragraphe *resultatMath*. S’il s’agit de nombres, additionnez-les et faites afficher la réponse dans le paragraphe.
  4. Testez le fonctionnement en entrant des nombres et en entrant du texte.

1. **Partie 3**

Un clic sur le bouton ‘Générer des nombres’ doit générer 5 nombres de façon aléatoire, les faire afficher et afficher le résultat de leur somme. Pour ce faire :

* 1. Le tableau *nombres* contenant 5 nombres est déclaré au début du fichier javascript. Il s’agit d’une variable globale car le tableau doit être utilisé par 2 fonctions différentes.

La fonction *GenererNombres*() doit générer 5 nombres entre 1 et 10 et les garder dans le tableau *nombres.* Pour ce faire :

* + - Dans la boucle de GenererNombres(), inscrivez l’instruction permettant de générer un nombre entre 1 et 10.
    - Le nombre généré doit être placé à la bonne position dans le tableau (1er nombre à la position 0, 2ième nombre à la position 1, …).
  1. La fonction *Calculer*() appelle *GenererNombre*(). Donc, au retour de la fonction, le tableau contient 5 nombres valides.

Il faut maintenant faire afficher ces nombres dans les <span> prévus à cet effet dans le code html. Pour ce faire :

* + - Utilisez la variable *elements* qui contient déjà les *span* en question. Dans la boucle, ajoutez l’instruction qui permet de faire afficher les nombres dans les <span> (1er nombre dans le 1er span, …).
  1. Testez le fonctionnement.
  2. Ajoutez des instructions qui calculent la somme des 5 nombres et la fait afficher dans le paragraphe ‘somme’.
  3. Testez le fonctionnement.

1. **Partie 4**

Faire afficher un chronomètre.

* 1. Dans InitialiserEvenements() on retrouve l’appel de la fonction DebuterChrono(). Enlever les commentaires devant cet appel afin de faire exécuter la fonction.
  2. La fonction DebuterChrono() contient l’instruction setInterval(″AfficherHeure()″, 1000).

Cela signifie que la fonction AfficherHeure() sera exécutée à toutes les 1000 millisecondes, donc à chaque seconde.

* 1. Puisque AfficherHeure() est exécuté à chaque seconde alors on peut utiliser AfficherHeure() pour augmenter le nombre de secondes et ainsi mettre en place notre chronomètre.

Une variable globale ‘seconde’ est déjà déclarée au début du fichier. C’est cette variable que nous allons utiliser pour augmenter les secondes.

On va procéder par étapes :

* Augmentez le nombre de secondes
* Dans l’élément html portant le id ‘chrono’, faites afficher les secondes
* Testez le fonctionnement
* Faites-en sorte de faire afficher les secondes sous le format *:ss.*

Ex : si les secondes sont de 5, cela affichera **:05**

Et si les secondes sont de 12, cela affichera **:12**

* Testez le fonctionnement
* Il reste à tenir compte des minutes. Lorsque les secondes atteignent 60, il faut donc ajouter une minute et remettre les secondes à 0.

Il est possible de créer ce fonctionnement avec des if mais je vous propose d’utiliser les mathématiques 😊. En effet :

* + si on divise les secondes par 60, cela nous donne le nombre de minutes
  + si on garde le **reste** de la division par 60, cela nous donne les secondes
  + ex : 25 secondes **/** 60 = 0 minute

25 secondes **%** 60 = 25 secondes (% est l’opérateur modulo qui donne le reste d’une division)

60 secondes **/** 60 = 1 minute

60 secondes **%** 60 = 0 secondes

* + donc :

minute = minute + Math.floor(seconde / 60) mettra les minutes à jour

seconde = seconde % 60 mettra les secondes à jour

* Modifiez maintenant l’affichage pour tenir compte des minutes *mm:ss.*

Ex : **01:05** ou **01:15** ou **10:08** ou …

* Testez le fonctionnement