

Laboratoire semaine 11

1. Créez le dossier Lab7.

Partie 1 : Labo7A

2. Modifiez la fonction **isNumber()** dans la librairie **mv112.js** Pour qu'elle accepte maintenant les objets **Number**.

```
/**
 * Retourne true si data est de type number (de type number ou de type object).
 * Sinon elle retourne false
 * @param {String} data
 * @return {Boolean}
 */
function isNumber(data) {
    return ((!isNull(data)) && (typeof data == 'number') ||
        (typeof data == 'object' && data instanceof Number));
}
```

3. Modifiez la fonction **isString()** dans la librairie **mv112.js**. Pour qu'elle accepte les objets **String**.

```
✓ /**
 * Retourne true si data est une chaîne de caractères (de type string ou de type object).
 * Sinon elle retourne false
 * @param {String} data
 * @return {Boolean}
 */
✓ function isString(data) {
    return ((!isNull(data)) && (typeof data == 'string')
        || (typeof data == 'object' && data instanceof String));
}
```

4. Ajoutez la fonction **isArray()** dans **mv112.js**

```
/** Retourne true si une date est valide ou non.
 * Le jour, le mois et l'année sont situés dans une chaîne de caractères.
 * @param {Array} tabArray
 * @return {Boolean}
 */

function isTable(tabArray){
    return (!isNull(tabArray) && (typeof tabArray == 'object' && tabArray instanceof Array));
}
```

5. Ouvrez le fichier **Labo7A.htm** à l'aide de *Visual Studio Code* et remplacez toutes les occurrences de **VotreNom** par votre vrai nom.
6. La fonction **readAndValideData()** a déjà été programmée. Cette fonction appelle la fonction **readData()** et la fonction **valideData(objStudent)**.
 - a. Vous devez programmer la fonction **readData ()**. Cette fonction doit lire toutes les données de l'étudiant qu'il y a sur la page Web : son matricule, son code permanent, son nom, son prénom, son sexe, son jour de naissance, son mois de naissance et son année de naissance. Ces données doivent être placées dans un objet et cet objet doit être retourné.
 - b. Vous devez également programmer la **valideData(objStudent)**. Cette fonction doit faire exactement la même chose que la fonction **valideData()** du laboratoire 6B sauf que cette fonction doit aller chercher les données dans l'objet **objStudent** et non pas sur la page Web.

Partie 2 : Labo7B

7. Ouvrez les fichiers **Labo7B.htm** et **styleLabo7B.css** à l'aide de *Visual Studio Code* puis remplacez toutes les occurrences de **VotreNom** par votre vrai nom.
8. Ouvrez la page Web. Sur cette page Web, deux sous-divisions ont été ajoutées à l'intérieur de la division **divCorps**.

The screenshot shows a web browser window with a title bar. The page content is as follows:

Array
by YOURNAME

Computation

Sum:
Average:
Smallest:
Biggest:

COMPUTE

Enter numbers

Nombre 0:

Nombre 1:

Nombre 2:

Nombre 3:

Nombre 4:

Nombre 5:

Nombre 6:

- ✓ La sous-division **divPanneauResultats**. Visuellement, cette sous-division est collée sur la marge de gauche.
- ✓ Et la sous-division **divPanneauSaisie**. Visuellement, cette sous-division est collée sur la marge de droite.

- ✓ Dans le fichier **styleLabo7B.css**, observez le sélecteur **sFlottantAGauche** et le sélecteur **sFlottantADroite**.

```
.sFlottantAGauche { float:left; margin-bottom:5px;}  
.sFlottantADroite { float:right; margin-bottom:5px;}
```

Rappel

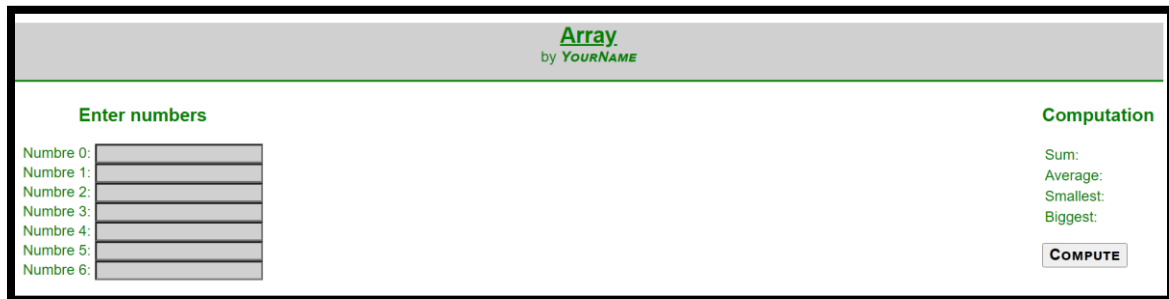
- ✓ **float:left** signifie que cet élément flotte vers la gauche. Ou, en d'autres mots, que cet élément se colle sur l'élément situé à sa gauche. S'il n'y a pas d'élément situé à sa gauche, cet élément se colle sur la marge de gauche (c'est le cas ici).
- ✓ **float:right** signifie que cet élément flotte vers la droite. Ou, en d'autres mots, que cet élément se colle sur l'élément situé à sa droite. S'il n'y a pas d'élément situé à sa droite, cet élément se colle sur la marge de droite (c'est le cas ici).

Pour voir l'effet que cela donne, agrandissez puis rapetissez horizontalement la page Web et observez le comportement des sous-divisions.

Maintenant, modifiez la position horizontale des deux sous-divisions de la manière suivante puis ouvrez la page Web.

```
<div id="divPanneauResultats" class="sFlottantADroite">  
<div id="divPanneauSaisie" class="sFlottantAGauche">
```

Observez que la position horizontale des sous-divisions a été inversée.



Le but de cette application est de calculer puis d'afficher la somme, la moyenne, le plus petit et le plus grand nombre parmi 7 nombres tapés par l'utilisateur.

Observez les points suivants :

- Les zones de texte reliés aux nombres tapés par l'utilisateur portent le nom de **tbNombre0** jusqu'à **tbNombre6**.

```
Nombre 0: <input id="tbNombre0" name="tbNombre0" class="sCouleurFondZoneDeTexteNormal" type="text" /><br />
Nombre 1: <input id="tbNombre1" name="tbNombre1" class="sCouleurFondZoneDeTexteNormal" type="text" /><br />
Nombre 2: <input id="tbNombre2" name="tbNombre2" class="sCouleurFondZoneDeTexteNormal" type="text" /><br />
Nombre 3: <input id="tbNombre3" name="tbNombre3" class="sCouleurFondZoneDeTexteNormal" type="text" /><br />
Nombre 4: <input id="tbNombre4" name="tbNombre4" class="sCouleurFondZoneDeTexteNormal" type="text" /><br />
Nombre 5: <input id="tbNombre5" name="tbNombre5" class="sCouleurFondZoneDeTexteNormal" type="text" /><br />
Nombre 6: <input id="tbNombre6" name="tbNombre6" class="sCouleurFondZoneDeTexteNormal" type="text" />
```

- Les étiquettes qui vont contenir les résultats portent respectivement le nom de **lblSomme**, **lblMoyenne**, **lblPlusPe-tit** et **lblPlusGrand**.
- Lorsque l'utilisateur clique sur le bouton **btnCalculer**, la fonction **computeNumbers()** est automatiquement appelée. Cette fonction a déjà été programmée.

Votre Travail est le suivant:

Écrivez les fonctions suivantes :

1. readNumberAndReturnArrayNumbers()

Cette fonction doit lire chacun des nombres tapés par l'utilisateur dans la zone de texte correspondante, valider le nombre, et, s'il est valide, convertir ce nombre lu en donnée de type number et l'ajouter à la fin d'un tableau (vide au point de départ). A la fin, la fonction doit retourner ce tableau.

Si l'utilisateur n'a tapé que des caractères blancs dans la zone de texte correspondante (utilisez la fonction **isWhiteSpace()**), cette fonction doit mettre la couleur de fond de la zone de texte à une couleur rougeâtre (définissez un nouveau sélecteur dans styleLabo7B.css) et ne doit pas ajouter ce nombre à l'intérieur du tableau.

Enter numbers

Nombre 0:	wewe
Nombre 1:	235
Nombre 2:	258
Nombre 3:	a25
Nombre 4:	
Nombre 5:	
Nombre 6:	

Si l'utilisateur a tapé autre chose qu'un nombre dans la zone de texte correspondante (utilisez la fonction **hasOnlyNumber()**), cette fonction doit mettre la couleur de fond de la zone de texte à une couleur jaunâtre (définissez un nouveau sélecteur dans styleLabo7B.css) et ne doit pas ajouter ce nombre à l'intérieur du tableau.

Si ce que l'utilisateur a tapé est un nombre, cette fonction doit mettre la couleur de fond de la zone de texte à une couleur verdâtre (définissez un nouveau sélecteur dans styleLabo7B.css) puis ajouter ce nombre à l'intérieur du tableau (après l'avoir converti en donnée de type number).

Indice : Pour lire chacun des nombres, utilisez une boucle for. Par exemple :

```
for(var i = 0; i < 7; i++) {
    const strNumber = document.getElementById("tbNombre"+i).value;
```

Contrainte : Vous devez utiliser la méthode **.push()** pour ajouter le nombre à la fin du tableau.

2. readAndDisplayBiggestNumber(tabNumbers)

Cette fonction doit rechercher le plus grand nombre qu'il y a dans le tableau de nombres **tabNumbers** passé en paramètre et afficher le résultat dans l'étiquette **lblPlusGrand**.

Si le tableau est vide, votre fonction doit afficher : **Aucun nombre valide.**

Contrainte : Le plus grand nombre doit toujours être affiché avec 2 chiffres après le point (utilisez la méthode **.toFixed(2)**).

3. readAndDisplaySmallestNumber(tabNumbers)

Cette fonction doit rechercher le plus petit nombre qu'il y a dans le tableau de nombres **tabNumbers** passé en paramètre et afficher le résultat dans l'étiquette **lblPlusPetit**.

Si le tableau est vide, votre fonction doit afficher : **Aucun nombre valide.**

Contrainte : Le plus petit nombre doit toujours être affiché avec 2 chiffres après le point (utilisez la méthode **.toFixed(2)**).

4. computeAndDisplaySumNumbers(tabNumbers)

Cette fonction doit calculer la somme de tous les nombres qu'il y a dans le tableau de nombres **tabNumbers** passé en paramètre et afficher le résultat dans l'étiquette **lblSomme**.

Si le tableau est vide, votre fonction doit afficher : **Aucun nombre valide.**

Contrainte : La somme doit toujours être affichée avec 2 chiffres après le point (utilisez la méthode **.toFixed(2)**).

5. computeAndDisplayAverageNumbers(tabNumbers)

Cette fonction doit calculer la moyenne de tous les nombres qu'il y a dans le tableau de nombres **tabNumbers** passé en paramètre et afficher le résultat dans l'étiquette **lblMoyenne**.

Si le tableau est vide, votre fonction doit afficher : **Aucun nombre valide.**

Contrainte : La moyenne doit toujours être affichée avec 2 chiffres après le point (utilisez la méthode **.toFixed(2)**).

Voici quelques captures écran :

Array
by YOURNAME

Enter numbers

Nombre 0:

Nombre 1:

Nombre 2:

Nombre 3:

Nombre 4:

Nombre 5:

Nombre 6:

Computation

Sum: Aucun nombre valide.

Average: Aucun nombre valide.

Smallest: Aucun nombre valide.

Biggest: Aucun nombre valide.

COMPUTE

Array
by YOURNAME

Enter numbers

Nombre 0:

Nombre 1:

Nombre 2:

Nombre 3:

Nombre 4:

Nombre 5:

Nombre 6:

Computation

Sum: 1136.00

Average: 568.00

Smallest: -98.00

Biggest: 1234.00

COMPUTE

Array
by YOURNAME

Enter numbers

Nombre 0:

Nombre 1:

Nombre 2:

Nombre 3:

Nombre 4:

Nombre 5:

Nombre 6:

Computation

Sum: -8497098.00

Average: -1213871.14

Smallest: -8742424.00

Biggest: 242424.00

COMPUTE

Array
by YOURNAME

Enter numbers

Nombre 0:

Nombre 1:

Nombre 2:

Nombre 3:

Nombre 4:

Nombre 5:

Nombre 6:

Computation

Sum: Aucun nombre valide.

Average: Aucun nombre valide.

Smallest: Aucun nombre valide.

Biggest: Aucun nombre valide.

COMPUTE

Remise

- ✓ Comprimez votre dossier et transmettez le fichier Labo7.zip sur LEA