

TP JS 2 - Chaines et Tableaux

Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le site [MDN](#), et pour visualiser ce qui est attendu, ce [démonstrateur](#).

Exercice 1. Occurrences de sous-chaînes

Le fichier **nbOcc.html** importe **nbOcc.js** et affiche le résultat d'appels à la fonction **nbOcc()**. Cette dernière prend en paramètre une chaîne de caractères **s** et une sous-chaîne **ss** et détermine le nombre d'occurrences de **ss** dans **s**. Implémentez cette fonction en vous aidant de fonctions telles **indexOf**, **search** ou **substring**. L'affichage attendu est illustré en Figure 1.

nbOcc

doigts dans **Les doigts boudinés de l'ogresse, les doigts acérés de l'ogresse, ses doigts qu'il fallait à tout prix satisfaire**, = 3

notre dans **Le nombre faisait notre force et notre faiblesse; notre force, parce que nous étions une armée; notre faiblesse, car nous étions désunis**, = 4

FIGURE 1 – Recherche d'occurrences de sous-chaînes

Exercice 2. Parcours de tableaux

Le fichier **matrix.html** importe **matrix.js** et affiche le résultat d'appels à la fonction **sMatrix()**. Implémentez cette fonction qui prend en entrée une "matrice" d'entiers (un tableau de tableaux) et l'affiche sous forme matricielle dans une alerte. Utilisez impérativement la méthode d'itération **forEach**. L'affichage attendu est illustré en Figure 2.



FIGURE 2 – Matrice d'entiers dans une alerte

Exercice 3. Manipulation de tableaux

Le fichier `voyelles.html` importe `voyelles.js` et définit deux chaînes, l'une `s` contenant un texte “**Lo**rem ipsum...”, l'autre `voyelles` contenant la liste des voyelles. L’affichage attendu est illustré en Figure 3.

1. Transformez `s` en un tableau de caractères `tab` soit à l’aide d’une méthode de `String`, soit à l’aide d’une méthode de `Array`.
2. Eliminez de `tab` tout caractère qui n’est pas une voyelle en vous servant de `voyelles` et stockez le tableau résultat dans `tabv`.
3. Créez à partir de `tabv` un tableau associatif (objet `Map`) qui associera à chaque voyelle le nombre de ses occurrences dans `tabv`.
4. Déterminez la voyelle ayant le plus grand nombre d’occurrences et stockez la dans `maxv`.

Voyelles

```
1. tab = L,o,r,e,m,i,p,s,u,m,d,o,l,o,r,s,i,t,a,m,e,t,,c,o,n,s,e,c,t,e,t,u,r,a,d,i,p,i,s,c,i,n,g,e,l,i,t,,S,e,d,n,o,n,
    r,i,s,u,s,,S,u,s,p,e,n,d,i,s,s,e,l,e,c,t,u,s,t,o,r,t,o,r,,d,i,g,n,i,s,s,i,m,s,i,t,a,m,e,t,,,a,d,i,p,i,s,c,i,n,g,n,e,c,,
    u,l,t,r,i,c,i,e,s,s,e,d,,,d,o,l,o,r,,
2. tabv = o,e,i,u,o,o,i,a,e,o,e,e,u,a,i,i,i,e,i,e,o,i,u,u,e,i,e,e,u,o,o,i,i,i,i,a,e,a,i,i,i,e,u,i,i,e,e,o,o
3. map = a,4,e,13,i,17,o,9,u,6,y,0
4. maxv = i=17
```

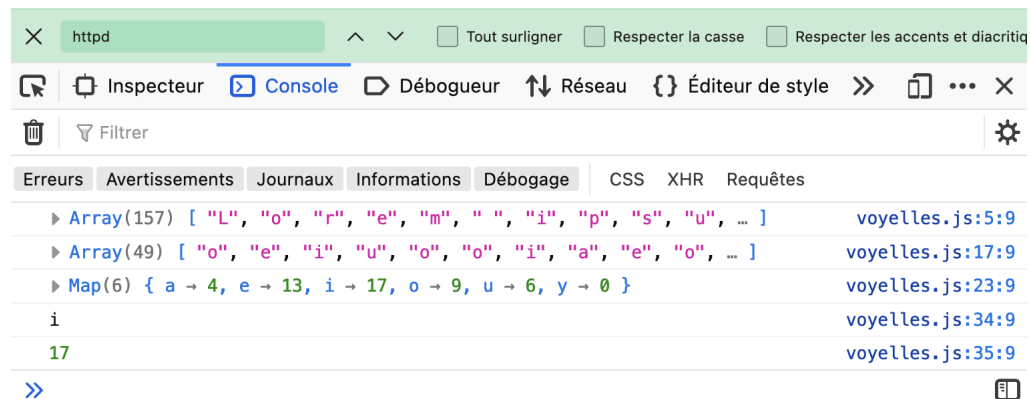


FIGURE 3 – Page web et sortie console attendues