

```

1  /* DEBUT DU CODE DE LA FONCTION 'analyserPoeme' (APPELEE PAR UN BOUTON)
   =====
2
3  Ce code a été testé sur les poèmes suivants :
4  - 'La ballade des pendus' de François Villon
5  - 'Heureux qui comme Ulysse...' de Joachim du Bellay
6  - 'Elsa' d'Aragon
7  et quelques fables de La Fontaine
8
9  */
10
11 function analyserPoeme() {
12
13   let analyse = '';
14
15   const toutlePoeme = document.getElementById(`poeme`).value;
16
17   // séparation du titre et du poème
18   const titre = toutlePoeme.slice(0,toutlePoeme.indexOf('\n\n')).trim();
19   const poeme = toutlePoeme.slice(2+toutlePoeme.indexOf('\n\n')).trim();
20
21   //richesse lexicale (nombre de mots uniques/ nombre total de mots)
22   let nmot_t = separerMots(poeme).length
23   let listeMots = compterMots(poeme);
24
25   analyse += `La richesse lexicale du poème est :
   ${Math.round(100*listeMots.length/nmot_t)}%`;
26
27   //les 10 mots les plus fréquents (titre exclus)
28   analyse += `\n\nLes 10 mots les plus fréquents sont :`
29   for (let i=0; i<10; i++){
30     if (listeMots[i] === undefined){break;};
31     analyse += `\n${i+1}: ${listeMots[i][0]}, ${listeMots[i][1]} occurrence(s)`;
32   };
33
34   let phrases = poeme.split(/[\.\?!]/);
35   if (phrases[phrases.length-1] === '') {z=phrases.pop();};
36
37   //le nombre de phrases (titre exclus)
38   analyse += `\n\nLe poème compte ${phrases.length} phrase(s).`
39
40   //la longueur moyenne des mots par phrase (titre exclus)
41   analyse += `\n\nLongueur moyenne des mots par phrase :`;
42   let lmot_t = 0;
43   phrases.forEach((ph, i) => {
44     if (ph !== '') {
45       mots = separerMots(ph);
46       lmot=0;
47       mots.forEach (m=> {lmot+=m.length;});
48       analyse += `\n${i+1} : ${Math.round(lmot/mots.length*10.0)/10.0}`;
49       lmot_t += lmot;
50     }
51   });
52   if (phrases.length>1) {analyse += `\nsoit une longueur moyenne de
   ${Math.round(lmot_t/nmot_t*10.0)/10.0} caractères.`;} else {analyse += '.'};
53
54   let strophes = poeme.split('\n\n');
55   analyse += `\n\nLe poème compte ${strophes.length} strophes :`;
56   nvers = 0;
57   tableau = [];
58   n = 0;
59   strophes.forEach(x => {
60     if (x !== '') {
61       n++
62       vers = x.split('\n')
63       i = vers.length
64       nvers+=i
65       if (tableau[i]===undefined) {tableau[i]=[n]} else {tableau[i].push(n)};
66     };
67   });
68   for(i=1; i<tableau.length; i++) {
69     if (tableau[i]!==undefined) {analyse+=tableau[i].length + ' de ' + i + ' vers (n°'
   + tableau[i] + '), '};

```

```

70 };
71
72 vers = poeme.split('\n');
73 analyse += ` \npour un total de ${nvers} vers : `;
74 tableau = [];
75 n = 0;
76 vers.forEach(x => {
77     if (x != '') {
78         n++;
79         i = compterSyllabes(x, false, false); // version h muet, sans diérèse
80         if (tableau[i]===undefined) {tableau[i]=[n]} else {tableau[i].push(n)};
81     };
82 });
83 for(i=1; i<tableau.length; i++) {
84     if (tableau[i]!==undefined) {analyse+=tableau[i].length + ' de ' + i + ' syllabes
85     (n°' + tableau[i] + '), '};
86 };
87 analyse = analyse.substring(0, analyse.length-2) + '.';
88 analyse += ` \n\nCet outil a été utilisé notamment pour les poèmes : 'La ballade des
89 pendus' de François Villon, 'Heureux qui comme Ulysse...' de Joachim du Bellay et
90 'Elsa' d'Aragon.`
91
92 let texte = document.getElementById(`texte`);
93 texte.value = analyse
94
95 }
96
97 function separerMots(texte) {
98     // sépare la phrase en mots en minuscules à l'aide de match (qui crée un tableau)
99     // et d'une regexp marchant en français :
100     // les mots avec '-' sont décomposés
101     return texte.toLowerCase().match(/[a-zéeàùâêîôûäëïöüçœ]+/g)
102 }
103
104 function compterMots(texte) {
105     // on crée le tableau des mots à l'aide de match et d'une rexexp marchant en
106     // français :
107     mots = separerMots(texte);
108
109     // on met les mots dans un objet, pour compter les occurrences de chaque mot, et
110     // ensuite le convertir en tableau
111     freqMots = [];
112     mots.forEach (m => {
113         if (freqMots[m]) {
114             freqMots[m]++;
115         } else {
116             freqMots[m]=1;
117         }
118     });
119     let compteMots = Object.entries(freqMots);
120
121     // tri du tableau en fonction de la fréquence décroissante
122     compteMots.sort((a, b) => b[1] - a[1]);
123
124     return compteMots;
125 }
126
127 function compterSyllabes(vers, haspire = false, dierease = false) {
128     /*haspire : true = h aspiré, false h muet
129     -----
130     dierease : true = application de la dierease, false
131     non-----
132
133     Principe du découpage en syllabe : le vers en entrée est transformé à coup
134     d'expressions régulières
135     en vers0 pour lequel les suites de voyelles de regex_syl permettent de définir
136     avec un minimum
137     d'erreurs des syllabes.
138     Les principales transformations portent sur la suppression de la ponctuation, les
139     qu/gu devant voyelle, les e muets, les y voyelles
140     et les syllabes 'i'+voyelle pour lesquelles se pose la question de la diérèse
141     */
142
143     const regex_syl =

```

```
/(eau|eui|oue|oui|uiè|ièu|ièu|iou|iai|iau|aon|ai|âi|âi|au|ea|eô|ei|eu|ée|ia|iâ|ie|ié|iè|iê|io|iô|iù|oi|oi|ou|ou|ou|ui|ui|œu|a|e|i|o|u|é|è|à|ù|â|ê|î|ô|û|ë|ï|ö|ü|œ)/g;
```

```
const regex_qgu=/ (?<=[qg])u( ?=[ 'aeioéèàâêîô])/g;
const regex_emuet0 = /(?<=[bcdfghjklmnpqrstvwxyz])e( ?= ?[\.\? ,;:!] )/g;
const regex_emuet1 = /(es|ent)$/;
if (haspire) { regex_emuet2 = /e( ?= ( $| +[aeiouéèàâêîôûœ] ) )/g } else {
  regex_emuet2 = /e( ?= ( $| +[aeiouéèàâêîôûœh] ) )/g;
}
const regex_punct = /[ \. \? , ; : ! ' ' ]/g;
const regex_y1 = /(?<=[bcdfghjklmnpqrstvwxyz])y/g;
const regex_y2 = /y( ?= ?( |$) )/g;
const regex_iue = /(?<=[iu])e( ?= s? ( $| ) )/g;
const regex_dierese = /(i|u|ou) ( ?= [aeouéèàâêîôû] )/g;
const regex_plier = /(?<=[bcdfgptv] [lr]) (i|u|ou) ( ?= [aeouéèàâêîôû] )/g;
```

```
let vers0 = vers.toLowerCase();
```

```
// les 2 instructions suivantes sont à faire avant la suppression de la
punctuation
vers0 = vers0.replace(regex_qgu, ''); //on supprime les 'u' dans 'qu' et 'gu'
devant voyelle ou apostrophe
//vers0 = vers0.replace(regex_emuet0, 'a'); // premier traitement des e muets
: on les garde entre une consonne et un signe de punctuation --> à voir s'il
faut garder cela --> a priori non
//
```

```
vers0 = vers0.replace(regex_punct, ' ').trim(); //on remplace les signes de
punctuation par un espace que l'on supprime en fin de vers
vers0 = vers0.replace(regex_emuet1, ''); // deuxième traitement des e muets :
cas des 'es' ou 'ent' en fin de vers que l'on supprime (sans pis pour les
adjectifs 'négligent' ou autres)
vers0 = vers0.replace(regex_emuet2, ''); // troisième traitement des e muets
(e suivi d'un espace, puis d'une voyelle ou e en fin de vers) : on les
supprime
vers0 = vers0.replace(regex_y1, 'i'); // on remplace les y derrière consonne
par des i
vers0 = vers0.replace(regex_y2, 'i'); // on remplace les y en fin de mot par
des i
vers0 = vers0.replace(regex_iue, ''); // on remplace les ie(s)/ue(s) en fin
de mot ou de vers par i(s)/u(s)
if (dierese){ vers0 = vers0.replace(regex_dierese, 'ih'); } //on remplace les
i/u/ou devant voyelle par 'ih' pour bien obtenir 2 syllabes
else { vers0 = vers0.replace(regex_plier, 'ih'); } // on remplace les i/u/ou
devant voyelle et derrière certaines paires de consonnes par 'ih' pour bien
obtenir 2 syllabes (cas de 'plier' -> 'pliher')
```

```
return vers0.match(regex_syl).length;
```

```
}
```