```
/* DEBUT DU CODE DE LA FONCTION 'analyserPoeme' (APPELEE PAR UN BOUTON)
================================================

Ce code a été testé sur les poèmes suivants :
- 'La ballade des pendus' de François Villon
- 'Heureux qui comme Ulysse...' de Joachim du Bellay
- 'Elsa' d'Aragon
et quelques fables de La Fontaine

*/

function analyserPoeme() {

let analyse = '';

const toutlePoeme = document.getElementById(`poeme`).value;

// séparation du titre et du poème
const titre = toutlePoeme.slice(0,toutlePoeme.indexOf('\n\n')).trim();
const poeme = toutlePoeme.slice(2+toutlePoeme.indexOf('\n\n')).trim();

//richesse lexicale (nombre de mots uniques/ nombre total de mots)
let nmot_t = separerMots(poeme).length
let listeMots = compterMots(poeme);

analyse += `La richesse lexicale du poème est :
${Math.round(100*listeMots.length/nmot_t)}%`;

//les 10 mots les plus fréquents (titre exclus)
analyse += `\n\nLes 10 mots les plus fréquents sont :`
for (let i=0; i<10; i++){
    if (listeMots[i] === undefined){break;};
    analyse += `\n${i+1}: ${listeMots[i][0]}, ${listeMots[i][1]} occurrence(s)`;
};

let phrases = poeme.split(/[\.\?!]/);
if (phrases[phrases.length-1] === '') {z=phrases.pop();};

//le nombre de phrases (titre exclus)
analyse += `\n\nLe poème compte ${phrases.length} phrase(s).`

//la longueur moyenne des mots par phrase (titre exclus)
analyse += `\n\nLongueur moyenne des mots par phrase :`;
let lmot_t = 0;
phrases.forEach((ph, i) => {
    if (ph != '') {
        mots = separerMots(ph);
        lmot=0;
        mots.forEach (m=> {lmot+=m.length;});
        analyse += `\n${i+1} : ${Math.round(lmot/mots.length*10.0)/10.0}`;
        lmot_t += lmot;
    }
});
if (phrases.length>1) {analyse += `\nsoit une longueur moyenne de
${Math.round(lmot_t/nmot_t*10.0)/10.0} caractères.`;} else {analyse += '.';};

let strophes = poeme.split('\n\n');
analyse += `\n\nLe poème compte ${strophes.length} strophes : `;
nvers = 0;
tableau = [];
n = 0;
strophes.forEach(x => {
    if (x != '') {
        n++
        vers = x.split('\n')
        i = vers.length
        nvers+=i
        if (tableau[i]===undefined) {tableau[i]=[n]} else {tableau[i].push(n)};
    };
});
for(i=1; i<tableau.length; i++) {
    if (tableau[i]!=undefined) {analyse+=tableau[i].length + ' de ' + i + ' vers (n°'
    + tableau[i] + '), ';}
```

```
};

vers = poeme.split('\n');
analyse += `\npour un total de ${nvers} vers : `;
tableau = [];
n = 0;
vers.forEach(x => {
    if (x != '') {
        n++
        i = compterSyllabes(x, false, false );   // version h muet, sans diérèse
        if (tableau[i]===undefined) {tableau[i]=[n]} else {tableau[i].push(n)};
    };
});
for(i=1; i<tableau.length; i++) {
    if (tableau[i]!=undefined) {analyse+=tableau[i].length + ' de ' + i + ' syllabes (n°' + tableau[i] + '), ';}
};
analyse = analyse.substring(0, analyse.length-2) + '.';

analyse += `\n\n\Cet outil a été utilisé notamment pour les poèmes : 'La ballade des pendus' de François Villon, 'Heureux qui comme Ulysse...' de Joachim du Bellay et 'Elsa' d'Aragon.`

let texte = document.getElementById(`texte`);
texte.value = analyse

}

function separerMots(texte) {
    // sépare la phrase en mots en minuscules à l'aide de match (qui crée un tableau) et d'une regexp marchant en français :
    // les mots avec '-' sont décomposés
    return texte.toLowerCase().match(/[a-zéèàùâêîôûäëïöüçœæ]+/g)
}
function compterMots(texte) {
    // on crée le tableau des mots à l'aide de match et d'une rexexp marchant en français :
    mots = separerMots(texte);

    // on met les mots dans un objet, pour compter les occurrences de chaque mot, et ensuite le convertir en tableau
    freqMots = [];
    mots.forEach (m => {
        if (freqMots[m]) {
            freqMots[m]++;
        } else {
            freqMots[m]=1;
        }
    });
    let compteMots = Object.entries(freqMots);

    // tri du tableau en fonction de la fréquence décroissante
    compteMots.sort((a, b) => b[1] - a[1]);

    return compteMots;
}
function compterSyllabes(vers, haspire = false, dierese = false) {
    /*haspire : true = h aspiré, false h muet
    ------------------------------------------------------
      dierese : true = application de la dierese, false non--------------------------------------

    Principe du découpage en syllabe : le vers en entrée est transformé à coup d'expressions régulières
    en vers0 pour lequel les suites de voyelles de regex_syl permettent de définir avec un minimum
    d'erreurs des syllabes.
    Les principales transformations portent sur la suppression de la poncutation, les qu/gu devant voyelle, les e muets, les y voyelles
    et les syllabes 'i'+voyelle pour lesquelles se pose la question de la diérère
    */

        const regex_syl =
```

```
        /(eau|eui|oue|oui|uiè|ïeu|ieu|iou|iai|iau|aon|ai|âi|aî|au|ea|eô|ei|eu|ée|ia|iâ
        |ie|ié|iè|iê|io|iô|iu|oi|oî|ou|oû|où|ui|uî|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 = /(?<=[bcdfghjklmnpqrstvwxz])e(?= ?[\.\?,;:!])/g;
        const regex_emuet1 = /(es|ent)$/;
        if (haspire) {regex_emuet2 = /e(?=($| +[aeiouéèàâêîôûœ]))/g} else {
        regex_emuet2 = /e(?=($| +[aeiouéèàâêîôûœh]))/g};
        const regex_ponct = /[\.\?,;:!’']/g;
        const regex_y = /(?<=[bcdfghjklmnpqrstvwxz])y/g;
        const regex_iue = /(?<=[iu])e(?=s?($| ))/g;
        const regex_dierese = /(i|u|ou)(?=[aeouéèàâêôû])/g;
        const regex_plier = /(?<=[bcdfgptv][lr])(i|u|ou)(?=[aeouéèàâêôû])/g;

        let vers0 = vers.toLowerCase();

        vers0 = vers0.replace(regex_qgu, ''); //on supprime les 'u' dans 'qu' et 'gu'
        devant voyelle
        //vers0 = vers0.replace(regex_emuet0, 'a'); // premier traitement des e muets
        : on les garde entre une consonne et un signe de ponctuation --> à voir s'il
        faut garder cela --> a priori non
        vers0 = vers0.replace(regex_ponct, ' ').trim(); //on remplace les signes de
        ponctuation 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 (tans 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_y, 'i'); // on remplace les y derrière consonne
        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;
}
```