L'Ère du Vent

version 1.0.0

DIVERSITY by EPITECH

I. Introduction

Après avoir difficilement traversé l'Ouragan dans une pluie torrentielle à dos de griffons, vous vous retrouvez dans l'œil du cyclone. Dans celui-ci, tout est redevenu très calme, mais pour combien de temps... ? Vous apercevez alors une maison en haut d'une montagne impossible à atteindre pour toutes créatures ne sachant pas voler.

Vous avancez alors dans la direction de la maison qui se trouve juste devant vous. Le cyclone se déplaçant très vite, vous bondissez dans l'énorme porte visible avant qu'un vent violent n'ait pu vous atteindre. Le reste de votre équipe a dû prendre la fuite à dos de griffons par manque de temps et est allé se réfugier au pied de la montagne à l'abri de l'Ouragan.

Bienvenue dans la Maison de l'Air !

La Maison de l'Air au sommet de la montagne

II. Consignes

* En cas de question, pensez à demander de l'aide à votre voisin de droite. Puis de gauche. Demandez enfin à un Cobra si vous êtes toujours bloqué(e).
* Vous avez tout à fait le droit d'utiliser internet pour trouver des réponses ou pour vous renseigner.
* Les "..." dans le code signifie que c'est un élément à compléter.
* Mettez le dossier « Air » qui vous a été donné dans votre dossier Github.

a. Le temps presse

À peine la porte franchie, vous vous retrouvez dans une grande salle lumineuse avec en son centre un vase posé sur un socle en pierre. La salle possède une autre porte verrouillée. Sur les murs vous pouvez lire des inscriptions vous expliquant pourquoi un Ouragan s'est formé. Vous apprenez que le Signe du Verseau trouve l'élément Eau beaucoup plus impressionnant que l'élément Air, de plus il a demandé pour pouvoir partir dans la Maison de l'Eau. L'esprit du Vent se mit alors en colère et est parti créer un Ouragan pour démontrer la pleine puissance de l'élément Air. Le Verseau s'excusa à multiple reprise mais l'Esprit du Vent était déjà devenu fou...

La salle avec le vase Verseau en son centre

Vous décidez alors de prendre le vase du Verseau pour voir ce qu'il y a à l'intérieur. Au premier contact avec le vase, la porte d'entrée se ferme brusquement et vous entendez des bruits sourds venant des quatre coins de la salle. L'oxygène de salle est absorbé de plus en plus vite. Si vous n'arrivez pas à arrêter le mécanisme, vous ne pourrez pas tenir plus de quarante-cinq minutes. Après dix secondes de panique, un mur se déplace lentement vous donnant accès à un pavé tactile. En vous approchant, vous remarquez qu'il s'agit du jeu du Simon.

Le jeu du Simon

Avant de commencer, quelques informations complémentaires sont nécessaires :

* Le seul fichier dans lequel vous devez programmer se nomme « simon.ts ».
* Pour pouvoir mettre à jour la page web avec vos modifications, vous devrez exécuter la commande suivante :

```
tsc index.ts simon.ts
```

Pour ce jeu, vous aurez besoin d'ajouter des séquences de couleur. Pour ce faire vous allez commencer par en ajouter une seule. Une fois cela fait, il vous faudra faire disparaître le bouton pour lancer une partie. Pour ce faire suivez ces étapes :

* Lorsque le bouton « lancer » est appuyé, le jeu doit commencer et donc une couleur doit être ajoutée.
* Pour ajouter une couleur dans la séquence, une variable « computerSequence » est à votre disposition. Vous devez écrire :

```
computerSequence.push(nombre);
```

Le nombre doit être contenu entre 0 et 3 inclus, signifiants la couleur. Pour que vous n'ayez pas la même séquence à chaque partie, rendez ce nombre aléatoire.

* Une fonction nommée « playAnimationOfColorSequence() » permet de lancer l'animation des couleurs. Elle réactive par la même occasion les actions du joueur à la fin de celle-ci.
* Pour éviter toute tricherie du joueur pendant l'animation, il vous faut bloquer ces actions le temps de l'animation. La variable « playerTurn » est faites pour ça, assignez-lui la valeur « false ».
* « hideTextAndButton() » est une fonction permettant de cacher le bouton « lancer » au début d'une partie.

Bravo, vous avez réussi ! Votre jeu est presque fonctionnel à 100%, il ne vous manque plus qu'une dernière étape.

Pour avoir un jeu fonctionnel et savoir si le joueur a gagné ou fait une erreur il vous faut récupérer les boutons de couleur appuyés. La fonction « colorButtonPressed() » est à votre disposition. Les points suivants vous aideront dans cette tâche :

- Il faut vérifier si les animations de la séquence sont terminées avant d'effectuer des actions.
- Une fois la vérification faites, vous devez ajouter la couleur que le joueur a appuyée à la variable « playerSequence » qui est un tableau. Procédez de la même manière que pour « computerSequence ».
- Si les tableaux « playerSequence » et « computerSequence » font la même taille alors c'est que le joueur a complété une séquence de couleur. Il faut donc réinitialiser la séquence du joueur et ajouter une nouvelle couleur aléatoire dans la séquence de l'ordinateur.
- Pour savoir si le joueur a perdu, il vous faudra utiliser la fonction « wrongColor() » qui retourne la valeur « true » si le joueur s'est trompé, dans le cas contraire, il retournera « false ». Vous devez lui envoyer en paramètre la couleur que le joueur a appuyée.
- Pour gagner, le joueur doit réussir une séquence de 6 couleurs. Pour le détecter dans le code, vous devez vérifier que la longueur du tableau « playerSequence » est égale à 6.
- Pour fermer le jeu si l'on gagne ou si l'on perd, il vous faudra utiliser la fonction « closeGame() » qui prend en paramètre un booléen. Ce paramètre doit valoir « true » si le joueur gagne et « false » s'il perd.

⚠ ***Attention :*** *Réfléchissez bien à l'ordre de vos conditions pour ne pas casser le jeu !*

Super, vous avez réussi ! Alors que la salle se reremplis d'oxygène, la porte verrouillée menant vers la salle suivante s'ouvre. Avant de partir, vous décidez de récupérer le vase Verseau, qui sait, il vous sera peut-être utile par la suite.

b. Un monde de connaissance

Après avoir repris votre souffle, vous rentrez dans la salle suivante qui s'avère être une bibliothèque inhumainement grande. En entrant, vous sentez que toute l'histoire du royaume de Stellium est écrite dans ces livres. En avançant dans les rangées de livres, vous apercevez une page rouge qui attire votre regard. Curieux, vous regardez la page en question et découvrez alors de vieux symboles que vous êtes incapable de traduire. Dans le doute, vous mettez donc cette carte dans votre poche et avant de continuer d'avancer pour arriver enfin au centre de la pièce.

La bibliothèque nationale de Stellium

Mais il y a un problème, la chute d'une des étagères a créé un trou béant dans le sol, ce qui vous sépare de l'autre partie de la salle où se situe la sortie. Vous sentez des courants d'air venant de ce trou. Alors que vous réfléchissez à comment passer de l'autre côté, vous entendez des bruits étranges se rapprochant de plus en plus tout en faisant trembler le sol.

Vous vous retrouvez face à deux géants jumeaux faisant du cerf-volant, vous commencez à vous enfuir mais un des géants vous attrape et vous demande de l'aide. Vous remarquez que les deux géants ont un tatouage du Signe du Gémeaux. Ils se présentent, l'un se nomme Aello tandis que l'autre se nomme Ocypète.

Photo de famille de Ocypète et Aello

Ils vous proposent un marché contre un cerf-volant pour pouvoir voler jusqu'à l'autre côté sans accroc. Ils vous expliquent donc que leur frère Podarge vantard, les prend de hauts. Chaque année, une compétition de résolution d'anagramme familiale est organisée pour déterminer qui a le plus de connaissances. Mais c'est Podarge qui brille chaque année. Votre quête est donc de réaliser un solveur d'anagramme pour qu'ils puissent s'entrainer.

Avant de commencer, quelques informations complémentaires sont nécessaires :

* Le seul fichier dans lequel vous devez programmer se nomme « anagramme.ts ».
* Pour pouvoir mettre à jour la page web avec vos modifications, vous devrez exécuter la commande suivante :

```
tsc index.ts anagramme.ts data.ts
```

Pour réaliser un solveur d'anagramme, vous trouverez le fichier « data.ts » qui contient plus de 140 000 mots officiels du dictionnaire. Et il vous sera utile dès maintenant !

Pour simplifier l'utilisation de votre dictionnaire vous aurez besoin d'ajouter chaque mot dans une variable de type tableau de string.

```
let dico:string[] = …
```

A l'aide de la fonction « JSON.parse() » vous aurez la possibilité de remplir la variable « dico » en récupérant le contenu du fichier « data.ts ».

Bien, maintenant que vous avez stocké l'ensemble des mots dans un tableau, vous allez devoir récupérer l'index du mot tapé par l'utilisateur. La fonction « .indexOf() » vous retourne l'emplacement d'un mot dans un tableau. Si le mot a été trouvé, son index est retourné sinon la valeur de retour est -1. Mais quel mot faudrait-il chercher ? Vous avez peut-être remarqué que la fonction « anagram » avait en paramètre « input » qui représente le mot tapé par l'utilisateur.

```
dico.indexOf(…)
```

Pour récupérer la valeur de retour vous aurez besoin de la stocker dans une variable de type « number ».

```
let id: number = ...
```

A l'aide de la variable « id », « input » et « dico » vous disposez de tous les éléments nécessaires pour utiliser la fonction « displayResult() » qui vous est déjà fourni. Elle prend en paramètre l'index de l'élément trouvé dans le tableau "dico », le mot écrit par l'utilisateur et pour finir le mot trouver dans le tableau « dico ».

Bien joué ! Vous pouvez désormais trouver un mot dans le dictionnaire.

A l'aide de votre travail fourni précédemment vous avez la possibilité de vérifier si un mot existe dans votre dictionnaire. Une anagramme étant un mot obtenu en permutant les lettres d'un autre mot, il vous faudra donc trier le mot de l'utilisateur et les mots du dictionnaire par ordre alphabétique. Pour commencer vous aurez besoin d'enlever toutes les majuscules du mot contenu dans « input ». Pour réaliser cette tâche une fonction existe se nommant « .toLowerCase() ».

```
input.toLowerCase();
```

Maintenant que vous avez votre « input » sans majuscule, pour savoir si deux mots sont identiques, vous allez le tirer par ordre alphabétique. Procédez de la même manière pour les mots contenu dans le tableau « dico ».

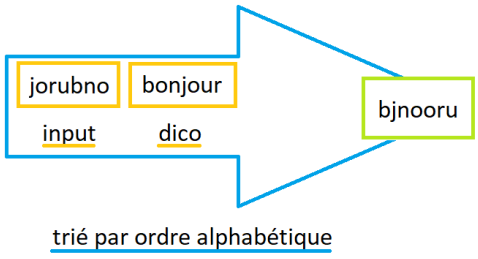


Schéma explicatif de l'algorithme

Pour ce faire, vous aurez besoin d'une nouvelle variable « tmpArray » elle devra être un tableau de string tout comme la variable "dico". Cette nouvelle variable devra contenir chaque lettre du mot contenu dans « input » et pour ça le fonction « .split() » vous sera utile.

```
let tmpArray: string[] = input.split('')
```

En suivant vous allez devoir trier votre tableau de lettre « tmpArray » à l'aide de la fonction « .sort() ».

Une fois vos lettres triées, vous aurez besoin qu'elle redevienne un seul et même mot. Pour cela vous allez devoir utiliser la fonction « .join() » pour ajouter chaque élément du tableau « tmpArray » dans une nouvelle variable que vous nommerez « sortedInput ».

```
let sortedInput : string = …
```

Maintenant vous allez devoir trier votre tableau « dico ». Pour ça vous aurez besoin de :

- Un autre tableau de string qui tout comme « sortedInput » contiendra les mots de « dico » trié.
- Faire une boucle pour parcourir tous les éléments de votre tableau « dico ». A l'intérieur de cette boucle vous aurez besoin de trier chaque mot de « dico » comme vous avez fait pour « input ».
- Pour finir et pour ajouter les mots triés à l'intérieur de votre nouveau tableau de string, vous aurez besoin d'utiliser la fonction « .push() » qui sert à ajouter un élément à la fin d'un tableau.

Maintenant que vous avez votre tableau trié et votre input trié, vous pouvez changer votre recherche faite précédemment.

```
let id: number = sortedDico.indexOf(sortedInput)
```

Génial ! Vous avez réussi votre solveur, les géants sont satisfaits du programme. Aello vous pose tout en haut d'une étagère et il vous donne le cerf-volant comme promis. Vous vous en servez alors comme planeur pour atteindre l'autre côté de la pièce. Après votre atterrissage, vous apercevez une porte, mais alors que vous allez passer de l'autre côté, vous tombez sur un livre nommé "Le vase du Verseau". En le feuilletant rapidement vous apprenez que l'eau contenue dans le Verseau efface le passé pour tous ceux qui entre à son contact. Après ces révélations, vous décidez de passer la porte pour vous rendre dans la prochaine salle.

c. Garde ta ligne

Une fois à l'intérieur, vous découvrez une magnifique statue de jeune femme représentant l'ordre, la loi et la coutume. Elle tient dans sa main droite un glaive et dans l'autre la Balance. Cependant vous remarquez que la Balance penche anormalement d'un côté alors qu'elle est vide. Derrière elle se trouve une représentation céleste géante peinte. Le soleil illuminant la pièce par on ne sait quelle sorcellerie.

Statue de la Balance

Au centre de la pièce votre regard se pose sur un serpent ballonné et mal en point. En vous approchant de lui il vous chuchote quelques mots pour vous demander de l'aide. Vous arrivez aussi à distinguer les mots "Manger" et "Stèle". En vous retournant, vous apercevez la nourriture entassée, composée de bananes ou encore de burgers. Une pierre dressée et située juste devant la statue et les inscriptions ci-dessous y sont écrites :

"Ce signe bien connu,

Symbole d'équilibre et de justice,

Aide le pêcheur assidu,

Dans sa pêche aux écrevisses."

Vous comprenez immédiatement qu'il y a un lien entre la nourriture et l'équilibre de la Balance. Vous retournez voir le serpent afin de le guider dans ses repas pour le guérir de ses maux de ventre !

Avant de commencer, quelques informations complémentaires sont nécessaires :

* Le seul fichier dans lequel vous devez programmer se nomme « snake.ts ».
* Pour pouvoir mettre à jour la page web avec vos modifications, vous devrez exécuter la commande suivante :

```
tsc index.ts snake.ts
```

Votre serpent est incapable de changer de direction une fois la partie lancée. Il aura tendance à faire des choses très étrange une fois sortie de l'écrans mais on réglera se problème plus tard. Pour lui faire changer de direction, rendez-vous dans la fonction « buttonPressed() ». Suivez alors ces instructions :

* Le contenu des guillemets doit être remplis avec ces valeurs : « ArrowDown » / « ArrowUp » / « ArrowLeft » / « ArrowRight ». Elles correspondent aux touches directionnelles du clavier.
* Une fois cela fait, il vous faudra appeler la fonction « changeDirection() » avec comme paramètre ces valeurs en fonction de la condition précédente : « Direction.left » / « Direction.down » / « Direction.right » / « Direction.up ». Veillez à faire cette appelle avant le « break ».

Pour information, un « switch » est une suite de « if » ne portant que sur une variable. On peut le résumé comme : si la variable vaut 3 alors fais ci, si elle vaut 5 alors fais ça sinon fais ci. Le « break » signifie que l'on veut sortir du « switch ».

Bravos vous avez réussi à le faire changer de direction ! Il reste beaucoup de chose à faire alors ne tardez pas à passer à la prochaine partie.

Votre serpent doit reprendre des forces. Pour ce faire, il faut que de la nourriture apparaisse sur le terrain. Rendez-vous dans la fonction « addFood() », ces instructions vous aideront dans sa réalisation :

* Il vous faut initialiser deux variable « x » et « y » à 0. N'oublier pas d'indiquer qu'il s'agit de nombre via le mot clé « number ».
* Une variable « emptyPlaceFound » de type « boolean » doit être initialisé à « false ».
* Afin d'avoir des emplacements de nourriture aléatoire, faite une boucle allant de 0 à 10.
* Ajoutez une condition à cette boucle. Il faut que la variable « emptyPlaceFound » soit aussi égale à « false ». Si elle passe à « true », il faut sortir de la boucle.
* Dans cette boucle, vous devez assigner à « x » et « y » un nombre entier aléatoire entre 1 et 16 inclus.
* Appelez ensuite la fonction « checkEmptyCase() » en lui envoyant « x » et « y » en paramètre. Elle renverra une valeur égale à « true » si la case est vide, sinon c'est « false » qui est renvoyé. Faites en sorte de stocker cette valeur de retour dans la variable « emptyPlaceFound ».
* Faites une condition qui vérifie si vous avez réussis à trouver un emplacement vide. Si c'est le cas, appeler la fonction « placeFood() » en lui passant le type de nourriture et les coordonnées de la case en question. Sinon appelez la fonction « findEmptyPlace() » en lui envoyant le type. Elle s'occupera de placer l'aliment sur une case vide.

Maintenant que la nourriture apparaît sur la carte, vous allez devoir vérifier si la tête du serpent se trouve sur de la nourriture. Vous allez devoir ajouter une condition dans la fonction « checkSnakeEatFood() ». Voici des informations complémentaires pour bien comprendre :

* La position de la tête du serpent est représenté par « snake[0].position[0] » pour sa position en « x » et « snake[0].position[1] » pour sa position en « y ».
* La position des aliments est obtenable via « food.position[0] » pour la position en « x » et « food.position[1] » pour la position en « y ».
* Cette fonction sera appelée autant de fois qu'il y a d'aliment sur la carte, vous n'avez donc pas besoin de faire de boucle pour vérifier aliment par aliment.
* Si la tête du serpent est bien sur un aliment, alors vous devez appeler la fonction « updateFoodOnMap() » en lui envoyant comme paramètre « food » et « index ».

Félicitation, votre serpent peut manger les aliments se trouvant sur la carte et cela fait grandir son corps. Avant de passer à la suite, jetez un œil en bas à gauche à la « barre d'équilibre ». Celle-ci change de valeur à chaque fois que le serpent mange un aliment tout comme le score qui augmente. Vous obtiendrez plus d'information sur ces éléments dans la parti suivante.

Le jeu est presque terminé. En effet pour le moment vous ne pouvez ni gagner ni perdre. Le jeu doit s'arrêter lorsque le serpent entre en collision avec son corps ou avec les bords de la carte. Il y a une dernière possibilité, vous devez garder une alimentation équilibrée car si la barre est remplie ou si elle est vide, c'est aussi une fin de jeu. Afin de gagner, une fois le jeu arrêté, le joueur doit avoir atteint un score de 500 ou plus, dans le cas contraire, le joueur a perdu. Pour ce faire, voici quelques instructions :

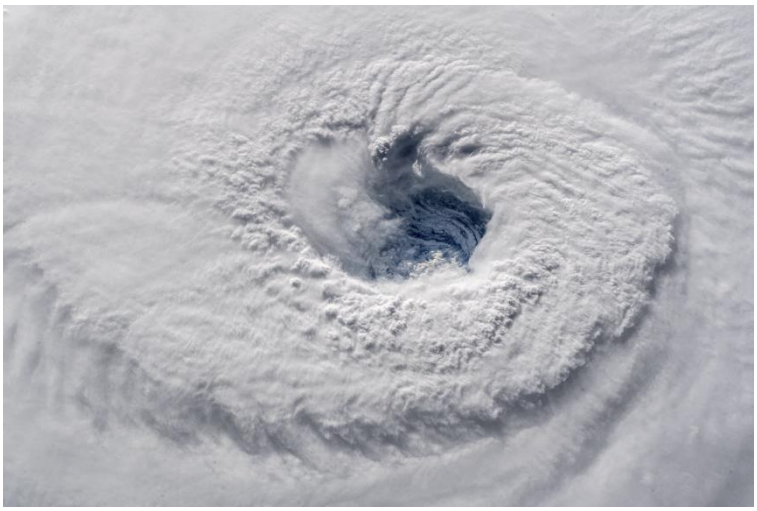
* Pour fermer le jeu et pouvoir relancer une nouvelle partie, il vous faudra appeler la fonction « closeGame() ». Pour indiquer que le joueur a gagné, vous devrez lui envoyer « true » en paramètre, dans le cas contraire vous devrez lui envoyer « false ».
* La fonction « checkBodyCollision() » retourne « true » si la tête du serpent est rentrée en collision avec son corps.
* La valeur en « x » et « y » de la position de la tête du serpent doit rester entre les valeurs de 1 à 16 inclus. Sinon c'est que le serpent est sorti de la carte.
* La valeur de la « barre d'équilibre » s'obtient via « balanceBar.value » et elle correspond à un nombre entier. Si sa valeur n'est plus comprise entre 0 et 100 exclus, alors le joueur doit perdre.
* Ajoutez une autre condition pour savoir si le joueur à un score supérieur ou égale à 500. Appelez la fonction « closeGame() » en conséquence.

Félicitations ! La Balance est désormais parfaitement équilibrée et le serpent se sent beaucoup mieux. Avec un grand sourire, il vous remercie et vous ouvre la porte en appuyant sur un bouton avec sa tête. De l'autre côté se trouve un toboggan sans fin. Le serpent vous indique qu'il s'agit de la sortie. Vous décidez alors de le prendre...

Au pied de la montagne

d. Aussi rapide que le vent

Après quelques minutes de descente, vous vous retrouvez au pied de la montagne. L'équipe au complet vous y attendez. Ils étaient contents de vous revoir, quand tout à coup, vous apercevez une masse bleutée vous courir dessus. Vous arrivez à l'esquiver in extremis. Vous arrivez à distinguer un géocoucou bleu criant des « Bip Bip » à tout va. Sa vitesse de sprint est tel qu'il créait des mini-tornades sur son passage. Ses yeux rouges croisent les vôtres et vous comprenez alors que l'Esprit maléfique du Vent manipule l'oiseau. Le scientifique vous explique que c'est cette créature qui a créé l'Ouragan. Vous devez le faire entrer en contact avec l'eau du Verseau pour chasser l'esprit qui le hante. Mais vous devez trouver un moyen de l'immobiliser !



L'Ouragan prend de plus en plus d'ampleur

Après une courte réflexion le scientifique vous propose de créer un programme pour augmenter votre vitesse. Vous viens alors l'idée de vous inspirer du FastFingers pour pouvoir attraper ce géocoucou.

Après un petit peu d'entrainement, vous attendez son passage devant vous tout en esquivant les mini-tornades créé sur son passage.

L'Esprit du Vent pas très content

Avant de commencer, quelques informations complémentaires sont nécessaires :

* Le seul fichier dans lequel vous devez programmer se nomme « fastFinger.ts ».
* Pour pouvoir mettre à jour la page web avec vos modifications, vous devrez exécuter la commande suivante :

```
tsc index.ts fastFinger.ts data.ts
```

Pour réaliser un test de dactylographie vous aurez besoin, tout comme pour le défi du gémeaux, de récupérer les 140 000 mots présents dans « data.ts ».

Dans la fonction « setRandomWord() », vous allez devoir dans un premier temps choisir un mot aléatoire dans votre dictionnaire, puis l'ajouter dans « listOfWord ».

```
for(…) {
      let randomWord = dico[Math.floor(Math.random() * dico.length)]
      listOfWord = ...
}
```

Maintenant que vous avez votre liste de mots, vous allez devoir diviser vos mots par lettre, pour les afficher sur votre page web. Cette partie vous servira à informer l'utilisateur qu'il s'est trompé à un endroit précis par un marqueur rouge.

```
var dividedWords: string[] = new Array;
dividedWords = listOfWord.split('');

for (var i = 0 ; i < listOfWord.length ; i++) {
    dividedWords[i] = '<span id="car' + (i + 1) + '">' +
dividedWords[i] + '</span>';
}

nbCar = listOfWord.length;
```

Pour transmettre ces informations à la page html vous aurez besoin d'appeler la fonction « setDividedWords() avec en paramètre « dividedWords ».

Vous avez maintenant toute les cartes en main pour réaliser le fastFinger. Pour commencer dans la fonction « fastFinger() », il faudra récupérer en temps réel les mots écrits par l'utilisateur. Pour ceci vous avez à disposition la fonction « getValue() » qui vous retourne les lettres tapées par le joueur.

Une fois en possession des valeurs, vous pouvez vérifier si l'utilisateur c'est trompé de charactère. Il serait intéressant de chercher comment fonctionne « .substr() »

```
if (... != listOfWord.substr(...)) {
}
```

Si cette condition est validée c'est que l'utilisateur a fait une faute de frappe. Il faut donc lui faire savoir qu'il a fait une erreur. Pour cela nous aurons besoin de réaliser une condition pour savoir si la variable booléenne « curseurErrBol » est égal à « false ».

```
if (curseurErrBol == false) {
      ...
}
```

Pour surligner l'endroit où se trouve la faute, vous aurez besoin de remplir la variable « curseurErrPos » avec la position de l'erreur. Une fois fait, vous aurez besoin de mettre « curseurErrBol » à « true ».

Une fois sortie de cette condition, pour gérer la couleur de la zone d'écriture de l'utilisateur, la fonction « setColor() » est à votre disposition. Elle prend en paramètre une couleur sous forme hexadécimale pour colorer le fond de votre boîte de saisie.

Maintenant que la condition d'erreur est faite, il ne vous reste plus qu'à faire un « else » à la suite et le remplir. Il devra contenir :

* Une mise à jour de la variable « curseurErrBol ».
* D'un « setColor() ».
* D'une condition qui vérifie si la longueur tapée par l'utilisateur et le nombre de caractère qui devait être tapé est identique.

Dans le cas où cette condition est validée, le joueur à fini. Vous avez donc une fonction disponible « end() » qui permet de calculer et d'afficher les résultats du joueur.

Pour finir et pour afficher le curseur d'erreur, vous devrez à la fin de la fonction « fastFinger() », appeler une fonction nommé « curseur() ». Elle prend en paramètre, la variable contenant les mots écrits par l'utilisateur ainsi que « curseurErrPos ».

III. Conclusion

Mais trop tard, il est déjà à votre portée ! Vous reculez de peur mais vous trébuchez sur un rocher. Le vase que vous teniez fermement jusqu'à présent vous glisse des mains et son contenu si précieux se déverse sur lui. L'eau coulant sur la créature luit et un nuage d'une douceur indescriptible se forme autour de lui, le surélevant. Après quelques secondes, l'oiseau redescend au fur et à mesure que le nuage se dissipe.

Ayant repris ses esprits, il vous fait un grand sourire, il vous remercie ! L'esprit maléfique du Vent sort de l'enveloppe corporelle de l'oiseau bleu. Un peu perdu et sans réel but, il se met à disparaitre au gré du vent. Tandis que vous vous remettez de vos émotions, vous demandez au géocoucou s'il peut vous accompagner vous et votre équipe à la Maison de la Terre. Il vous répond qu'il ne peut pas car il est exténué d'avoir couru sans repos depuis le début de ces évènements. Cependant il connaît quelqu'un qui n'habite pas très loin et qui possède un dirigeable. Vous prenez alors la route pour rendre visite à cette personne...

Le dirigeable qui vous attend