Amadou-Yacine

COMPTE RENDU JAVA/CALCULATRICE

SOMMAIRE

I. Java : le mot à trouver

OBJECTIFS ATTENDUS :

L'objectif est de programmer un jeu qui fait intervenir deux utilisateurs. Le premier joueur doit donner un mot de minimum 5 lettres.
Le deuxième joueur doit deviner le mot avec un nombre d'essai limité à la taille du mot –2. Celui-ci n'a pour indication que le nombre de lettres du mot à trouver et le nombre de lettres communes et bien placées. Il peut choisir d'abandonner la partie en écrivant le mot « SOL », et peut choisir de rejouer ou non.

Les Consignes :

Les consignes sont les suivantes :

- Insister sur les contrôles tout au long du jeu
- Porter un soin particulier au dialogue et à la présentation des différents écrans du jeu
- Travailler avec des vecteurs ou des chaines de caractères

Ressources utilisées :

Eclipse, cours.

Algorithme :

Algorithme "Le mot à trouver"

Variable : numérique = TailleUtilisateur, TailleJoueur, nombreEssai, tour, i

  CC = mot, motJoueur, SOL

Variable indicée : MatriceUtilisateur[TailleUtilisateur], MatriceJoueur[TailleUtilisateur], MatriceAffichage[TailleUtilisateur] d'éléments de type chaine de caractère

Début

Rejouer <-- "oui"

Tantque (Rejouer = "oui") et (SOL != " non ")

afficher "Explication du jeu : Ce jeu se joue à deux et consiste à trouver un mot donné par le premier joueur. Le nombre d'essai dépend de la taille du mot -2."

afficher "[Joueur 1] : Veuillez entrer un mot à faire deviner au joueur 2 avec au minimum 5 lettres."

Saisir mot

TailleUtilisateur <-- longueur(mot)

Tantque(TailleUtilisateur < 5)

afficher "[Joueur 1] : Recommencez avec un mot comportant au minimum 5 lettres."

TailleUtilisateur <-- longueur(mot)

FTantque

nombreEssai = TailleUtilisateur - 2

Pour i allant de 1 à TailleUtilisateur

MatriceUtilisateur[i] <-- (mot,i,1)

FPour

```
Pour i allant de 1 à TailleUtilisateur

MatriceAffichage[i] <-- "_"

FPour


tour <-- 0

motJoueur <-- ""


Tantque ((tour != nombreEssai)) et (SOL != "oui et SOL != "non ")

afficher "[Joueur 2] : Essayez de déviner le mot, entrer un mot de " ,(TailleUtilisateur),  "
        lettres."

Saisir motJoueur

Si motJoueur <-- "Sol" afficher "[Joueur 2] : Vous avez perdu, voulez-vous rejouer oui/non ?"     Saisir
Rejouer

Sinon TailleJoueur <-- longueur(motJoueur)


Tantque TailleJoueur != TailleUtilisateur

afficher "[Joueur 2] : Recommencez entrer un mot à deviner de " +
        (TailleUtilisateur) +  "lettres."

TailleJoueur <-- longueur(motJoueur)

Ftantque


tour <-- tour + 1

afficher "Nombre de tour restant : " ,(nombreEssai-tour)"


Pour i allant de 1 à TailleUtilisateur

MatriceJoueur[i] <-- (motJoueur,i,1)

FPour


Pour i allant de 1 à TailleUtilisateur

Si MatriceJoueur[i] <-- MatriceUtilisateur[i]

MatriceAffichage[i] <-- MatriceUtilisateur[i]

FPour
```

afficher "Voici les lettres communes et bien placées : "

Pour i allant de 1 à TailleUtilisateur

afficher MatriceAffichage[i]

FPour

Si mot = motJoueur

afficher "[Joueur 2] : Vous avez gagné en ", tour, " tour", ", voulez-vous rejouer oui/non ?"

Saisir Rejouer

tour = nombreEssai;

Sinon Si tour = nombreEssai

afficher "[Joueur 2] : Vous avez perdu, voulez-vous rejouer oui/non ?"

FTantque

Fin

Java :

```java
public class Trouve {

public static void main(String[] args) {

// < >

// Variables :
String MatriceUtilisateur[];
String MatriceJoueur[];
String MatriceAffichage[];
String mot;
String motJoueur;
String Rejouer;
String SOL = "";
int TailleUtilisateur;
int TailleJoueur;
int nombreEssai;
int tour;
```

```
// Début du programme, boucle de jeu :

Rejouer = "oui";
while(Rejouer.equalsIgnoreCase("oui") &&
(!SOL.equalsIgnoreCase("non"))) {
SOL = "";
//Explication du jeu :

System.out.println("Explication du jeu : ");
System.out.println();
System.out.println("Ce jeu se joue à deux et consiste à
trouver un mot donné par le premier joueur. Le nombre d'essai
dépend de la taille du mot -2.");
System.out.println();

// Saisie du mot par l'utilisateur

mot = Saisie.lire_String("[Joueur 1] : Veuillez entrer un mot
à faire deviner au joueur 2 avec au minimum 5 lettres.");
// Contrôle de saisie :
TailleUtilisateur = mot.length();
while(TailleUtilisateur < 5){
mot = Saisie.lire_String("[Joueur 1] : Recommencez avec un mot
comportant au minimum 5 lettres.");
TailleUtilisateur = mot.length();
}

// Calcul du nombre d'essai

nombreEssai = TailleUtilisateur - 2;


// Instanciation de la variables indicée utilisateur

//MatriceUtilisateur = new String [TailleUtilisateur];

MatriceUtilisateur = extraireLettres(mot);


// instanciation matrice affichage et remplir
MatriceAffichage = new String [TailleUtilisateur];

for(int i = 0; i < TailleUtilisateur; i++) {
MatriceAffichage[i] = "_";
}
```

```
// boucle essai du joueur :

tour = 0;
// initialisation mot joueur
motJoueur = "";

while((tour != nombreEssai &&
((!SOL.equalsIgnoreCase("oui")))) &&
(!SOL.equalsIgnoreCase("non"))) {
motJoueur = Saisie.lire_String("[Joueur 2] : Essayez de
déviner le mot, entrer un mot de " + (TailleUtilisateur) + "
lettres.");
if (motJoueur.equalsIgnoreCase("SOL")) {
 SOL = Saisie.lire_String("[Joueur 2] : Vous avez perdu,
voulez-vous rejouer oui/non ?");
} else {
 TailleJoueur = motJoueur.length();

// Contrôle de saisie du joueur
while((TailleJoueur != TailleUtilisateur)) {
motJoueur = Saisie.lire_String("[Joueur 2] : Recommencez
entrer un mot à deviner de " + (TailleUtilisateur) + "
lettres.");
 TailleJoueur = motJoueur.length();
 }


tour++;
System.out.println("Nombre de tour restant : " + (nombreEssai-
tour));
// Instanciation de la variables indicée joueur
MatriceJoueur = extraireLettres(motJoueur);


// Comparer les mots

for (int i = 0; i < TailleUtilisateur; i++) {
 if (MatriceJoueur[i].equalsIgnoreCase(MatriceUtilisateur[i]))
{
 MatriceAffichage[i] = MatriceUtilisateur[i];
 }
}
```

```
System.out.println("Voici les lettres communes et bien placées
: ");
for(int i = 0; i < TailleUtilisateur; i++) {
System.out.print(MatriceAffichage[i]+ " ");

}
System.out.println();


if(mot.equalsIgnoreCase(motJoueur)) {
Rejouer = Saisie.lire_String("[Joueur 2] : Vous avez gagné en
"+ tour + " tour" + ", voulez-vous rejouer oui/non ?");
tour = nombreEssai;
} else if (tour == nombreEssai) {
Rejouer = Saisie.lire_String("[Joueur 2] : Vous avez perdu,
voulez-vous rejouer oui/non ?");
}


}
}
}
}

//Fonction pour extraire les lettres d'un mot et les placer
dans un vecteur
public static String[] extraireLettres(String mot) {
String[] matrice = new String[mot.length()];

for (int i = 0; i < mot.length(); i++) {
matrice[i] = mot.substring(i, i + 1);
}

return matrice;
}



}
```

Explications :

- **MatriceUtilisateur**, **MatriceJoueur**, **MatriceAffichage**: Tableaux de chaînes de caractères pour stocker respectivement les lettres du mot à deviner, les lettres devinées par le joueur et les lettres affichées pendant le jeu.
- **mot**, **motJoueur**, **Rejouer**, **SOL**: Chaînes de caractères pour stocker respectivement le mot à deviner, le mot proposé par le joueur, la réponse à la question de rejouer, et la décision du joueur concernant la fin de la partie.
- **TailleUtilisateur**, **TailleJoueur**, **nombreEssai**, **tour**: Entiers pour stocker respectivement la taille du mot à deviner, la taille du mot proposé par le joueur, le nombre d'essais autorisés, et le numéro du tour en cours.

Boucles utilisées :

- **Boucle principale** (**while**):
  Gère le flux du jeu et contrôle si les joueurs veulent continuer à jouer.
- **Boucle de saisie du mot par le joueur 1**:
  Vérifie que le mot a au moins 5 lettres.
- **Boucle de saisie du mot par le joueur 2**:
  Contrôle la saisie du mot par le joueur 2 et vérifie que sa longueur correspond à celle du mot à deviner.
- **Boucle de traitement du jeu**:
  Gère les essais du joueur 2, compare les lettres saisies avec celles du mot à deviner, et affiche les lettres correctes.

Difficultés rencontrées :

- La gestion des boucles, en particulier pour sortir des boucles lorsque le joueur saisit "SOL".
- Nous avons dû faire face à des problèmes de logique et de contrôle des boucles, notamment pour garantir une sortie propre et cohérente du jeu. Malgré ces difficultés, nous avons persévéré pour résoudre les problèmes rencontrés et assurer le bon fonctionnement du jeu.

Partie 2 : Calculatrice

OBJECTIFS ATTENDUS :

L'objectif est de programmer une calculatrice en cascade avec du HTML, CSS et JavaScript. Elle permet d'effectuer une opération sur des entiers, les calculs pouvant être faits en cascade avec les touches de fonction suivantes :

- + pour l'addition
- - pour la soustraction
- * pour la multiplication
- C pour l'effacement
- F pour l'arrêt du traitement donc de la calculatrice

Les Consignes :

Les consignes sont les suivantes :

- Insister sur les contrôles tout au long du jeu
- Porter un soin particulier au dialogue et à la présentation des différents écrans du jeu

Ressources utilisées :

Visual studio code, https://www.w3schools.com/, cours.

HTML :

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="style.css">
    <title>Calculatrice</title>

</head>

<body>
    <header>
        <h1>Calculatrice</h1>
    </header>
    <main>
    <div class="cadre">
        <section class ="ecran">
            0
        </section>
        <section class="boutons">
                <div class="bouton-ligne">
                    <button class="bouton bis">
                        C
                    </button>
                    <button class="bouton">
```

```
            F
        </button>
        <button class="bouton">
            &divide;
        </button>
    </div>

    <div class="bouton-ligne">
        <button class="bouton">
            7
        </button>
        <button class="bouton">
            8
        </button>
        <button class="bouton">
            9
        </button>
        <button class="bouton">
            &times;
        </button>
    </div>

    <div class="bouton-ligne">
        <button class="bouton">
            4
        </button>
        <button class="bouton">
            5
        </button>
        <button class="bouton">
            6
```

```
            </button>
            <button class ="bouton">
                -
            </button>
        </div>

        <div class="bouton-ligne">
            <button class="bouton">
                1
            </button>
            <button class="bouton">
                2
            </button>
            <button class="bouton">
                3
            </button>
            <button class ="bouton">
                &plus;
            </button>
        </div>

        <div class="bouton-ligne">
            <button class="bouton zero">
                0
            </button>
            <button class="bouton">
                &equals;
            </button>
        </div>
</section>
```

```
    </div>
    </main>
    <script src="script.js"></script>
</body>
</html>
```

CSS :

```
body {
    background: url(/asset/img/uwu\ -\ Copie.png)center/cover;
    /* VH = 100 % de la taille de l'écran (viewport height) */
    min-height: 100vh;
}

h1{
    font-family: 'Franklin Gothic Medium', 'Arial Narrow', Arial, sans-serif;
    margin: 0;
    padding: 0;
    text-align: center;
    letter-spacing: 3px;
    font-size: 2.5;
}

.cadre{
    min-height: 500px;
    min-width: 340px;
    background: rgba(255, 151, 212, 0.644);
    /* Centrer une boite  */
    margin: 0;
    border: 2px solid rgba(240, 42, 164, 0.151);
```

```
    border-radius: 20px;
    box-shadow: 0px -1px 20px 15px #f56faf46;
    padding: 15px;
    position: absolute;
    top: 39%;
    left: 50%;
    transform: translate(-50%, -50%);
}

.ecran{
    background: rgba(255, 255, 255, 0.75);
    border: 1px solid rgba(255, 255, 255, 0.01);
    border-radius: 16px;
    box-shadow: 0 4px 30px rgba(35, 35, 35, 0.1);
    color: #232323;
    font-size: 35px;
    padding: 20px;
    text-align: right;
    width: 326px;
}

.bouton-ligne{
    display: flex;
    justify-content: space-between;
    margin: 5% 0;
}

.bouton{
    background: rgba(255, 255, 255, 0.75);
    border: 1px solid rgba(255, 255, 255, 0.01);
    border-radius: 16px;
```

```
    box-shadow: 0 4px 30px rgba(35, 35, 35, 0.1);
    color: #232323;
    flex-basis: 20%;
    font-family: inherit;
    font-size: 24px;
    height: 65px;

}

.bouton:last-child{
    background: rgba(255, 255, 255, 0.75);
    border: 1px solid rgba(255, 255, 255, 0.01);
    border-radius: 16px;
    box-shadow: 0 4px 30px rgba(35, 35, 35, 0.1);
    color: #fff;
    background: #d72880;
}


.bouton:hover{
    background-color: inherit;
}

.bouton:active{
    background-color: #fb78ffc0;
}

.bis{
    flex-basis: 47%;
}
```

```
.zero{
    flex-basis: 73%
}
```

Algorithme :

Algorithme “Calculatrice en cascade”

Variables : Numérique = Nb, Res

C = Op

Début

Op <-- “”

TantQue Op < > “F”

Afficher “ Quel est le premier nombre ?”

Saisir Res

Afficher “ Quel est votre opérateur ?”

Saisir Op

Tantque (Op < > “F”) ET (Op < > “C”)

Afficher “ Quel est votre deuxième nombre ?”

Saisir Nb

Si Op = “+” alors Res <-- Res + Nb

Sinon Si Op = “-” Alors Res <-- Res – Nb

Sinon Si Op = “*” Alors Res <-- Res * Nb

Fsi

Fsi

Fsi

Afficher “Votre résultat est “, Res

Afficher “ Quel est l’opérateur suivant ?”

Saisir Op

FTantQue

FtantQue

Afficher “Fin des calculs”

Fin

JavaScript :

```
// Variables
var totalEnCours = 0; // Variable pour stocker le total en cours
var zero = "0"; // Tampon pour stocker la valeur affichée sur l'écran
var operateurPrecedent; // Variable pour stocker l'opérateur précédent
var etat = "allumé"; // État actuel de la calculatrice



const ecran = document.querySelector('.ecran'); // Sélectionne l'élément
d'affichage sur l'écran


// Fonction pour traiter les clics sur les boutons
function clicBouton(valeur) {
  if (valeur == "F") {
    if (etat == "allumé") {
      etat = "éteint";
      ecran.innerText = "Éteint";
      // Désactive les boutons
      document.querySelector(".boutons").classList.add("désactivé");
      operateurPrecedent = null; // Réinitialise l'opérateur précédent
      totalEnCours = 0; // Réinitialise le total en cours
      zero = "0";
    } else {
      etat = "allumé";
      ecran.innerText = "0";
      // Active les boutons
      document.querySelector(".boutons").classList.remove("désactivé");
    }
    return;
  }
```

```
  if (etat == "éteint") {
    // Ignore les autres boutons si la calculatrice est éteinte
    return;
  }

  if (isNaN(valeur)) {
    // Traitement des symboles
    traiterSymbole(valeur);
  } else {
    // Traitement des nombres
    traiterNombre(valeur);
  }

  ecran.innerText = zero; // Met à jour l'affichage sur l'écran
}

// Fonction pour traiter les symboles (C, =, F, +, -, ×, ÷)
function traiterSymbole(symbole) {
  switch (symbole) {
    case "C":
      // Efface tout et réinitialise les variables
      zero = "0";
      totalEnCours = 0;
      operateurPrecedent = null;
      break;
    case "=":
      // Égal - effectue le calcul et met à jour le total en cours
      if (operateurPrecedent == null) {
        return;
      }
```

```
      effectuerOperation(parseInt(zero));
      operateurPrecedent = null;
      zero = totalEnCours;
      totalEnCours = 0;
      break;
    case "F":
      // Fonctionnalité "Éteindre" gérée dans la fonction clicBouton
      zero = "0";
      totalEnCours = 0;
      operateurPrecedent = null;
      break;
    case "+":
    case "-":
    case "×":
    case "÷":
      // Opérateurs mathématiques - effectue le calcul si nécessaire
      traiterMath(symbole);
      break;
  }
}

// Fonction pour traiter les opérateurs mathématiques (+, -, ×, ÷)
function traiterMath(symbole) {
  if (zero == "0") {
    return;
  }

  const intZero = parseInt(zero);

  if (totalEnCours == 0) {
    // Si le total en cours est 0, initialise-le avec la valeur du Zero
```

```
    totalEnCours = intZero;
  } else {
    // Sinon, effectue l'opération en cours avec la valeur du Zero
    effectuerOperation(intZero);
  }
  operateurPrecedent = symbole;
  zero = "0";
}

// Fonction pour effectuer l'opération en cours
function effectuerOperation(intZero) {
  if (operateurPrecedent == "+") {
    totalEnCours += intZero;
  } else if (operateurPrecedent == "-") {
    totalEnCours -= intZero;
  } else if (operateurPrecedent == "×") {
    totalEnCours *= intZero;
  } else if (operateurPrecedent == "÷") {
    totalEnCours /= intZero;
  }
}

// Fonction pour traiter les nombres
function traiterNombre(chaineNombre) {
  if (zero == "0") {
    zero = chaineNombre;
  } else {
    zero += chaineNombre;
  }
}
```

```
// Fonction d'initialisation, ajoutant un écouteur d'événements aux boutons de
la calculatrice

function initialiser() {

  document.querySelector(".boutons").addEventListener("click", function
(event) {

      clicBouton(event.target.innerText); });}// Appelle la fonction
d'initialisation au chargement de la page   initialiser();
```

Explications :

- **Variables :** On déclare plusieurs variables pour stocker le total en cours (**totalEnCours**), la valeur affichée sur l'écran (**zero**), l'opérateur précédent (**operateurPrecedent**) et l'état de la calculatrice (**etat**).
- **Fonction clicBouton**: Cette fonction est appelée lorsqu'un bouton de la calculatrice est cliqué. Elle prend la valeur du bouton cliqué en paramètre. Elle vérifie si la calculatrice est allumée ou éteinte, traite les symboles et les nombres, puis met à jour l'affichage sur l'écran.
- **Fonctions pour traiter les symboles et les opérateurs mathématiques :** Ces fonctions (**traiterSymbole** et **traiterMath**) sont utilisées pour effectuer les opérations appropriées en fonction du bouton cliqué. Par exemple, si l'utilisateur clique sur le bouton **+**, la fonction **traiterSymbole** est appelée avec le symbole **+**, qui traite alors l'opération d'addition.
- **Fonction effectuerOperation**: Cette fonction est appelée pour effectuer l'opération en cours. Elle prend le nombre actuellement affiché sur l'écran en paramètre et utilise l'opérateur précédent pour effectuer le calcul approprié.
- **Fonction traiterNombre**: Cette fonction est appelée lorsque l'utilisateur clique sur un bouton représentant un nombre. Elle met à jour la valeur affichée sur l'écran en concaténant le chiffre au nombre existant.
- **Fonction initialiser :** Cette fonction initialise la calculatrice en ajoutant un écouteur d'événements à tous les boutons de la calculatrice. Lorsqu'un bouton est cliqué, il appelle la fonction **clicBouton** avec la valeur du bouton cliqué en tant que paramètre.
- **Appel à la fonction d'initialisation :** Enfin, la fonction **initialiser** est appelée au chargement de la page pour activer les fonctionnalités de la calculatrice.

Difficultés rencontrées :

Nous avons rencontré des difficultés lors de l'adaptation de l'algorithme initial à JavaScript, notamment en ce qui concerne les interactions utilisateur, la gestion de l'état de calculatrice.

Conclusion :

Pour le jeu "Mot à trouver", l'objectif était de créer un jeu interactif où deux joueurs interagissent pour deviner un mot donné par le premier joueur. Le deuxième joueur doit trouver le mot en un nombre limité d'essais, avec pour seules indications la longueur du mot et le nombre de lettres bien placées. Cet exercice nous a permis de travailler sur l'utilisation de matrice et de chaine de caractère.

Concernant la calculatrice en cascade, l'objectif était de développer une calculatrice fonctionnelle en utilisant HTML, CSS et JavaScript. La calculatrice devait permettre d'effectuer des opérations mathématiques simples (+, -, ×, ÷) sur des entiers, avec la possibilité d'enchaîner les calculs. Cela nous a poussé à devoir chercher les connaissances requises pour réaliser les différents codes, et à organiser notre travail.