IUT Metz — SAé 3.01 Développement d'une application — Passerelle
Tic Tac Toe

Remarque : les fonctions dans le sujet ne sont pas typées. Vous tacherez de faire un choix pertinent pour les types des paramètres et du retour.

1 Tic Tac Toe

Le jeu de Tic Tac Toe est un jeu qui se joue à deux joueurs sur une grille carrée de taille $n \times n$[1]. Les joueurs jouent à tour de rôle. Le premier joueur remplit une des cases avec un *X* puis le second joueur remplit une case libre avec un *O* et ainsi de suite. Le gagnant est celui qui arrive à aligner n symboles sur une ligne, une colonne ou une diagonale. Il se peut que la partie se termine par un match nul si la grille est remplie sans qu'aucun des joueurs n'arrive à aligner n symboles.

2 Représentation en mémoire d'une grille de Tic Tac Toe

Il faut d'abord commencer par définir la représentation mémoire d'une grille de taille $n \times n$. La structure de données que vous allez définir et utiliser doit remplir les conditions suivantes :

1. permettre représenter des grilles à deux dimensions ;
2. d'accéder directement à une case par son numéro de ligne et son numéro de colonnes (tous les deux un entier entre 0 et $n-1$) ;
3. permettre pour chaque case : i) de savoir si un symbole lui est affecté ; ii) si oui, quel est ce symbole ;
4. permettre de modifier le contenu des cases ;
5. la structure doit être homogène (elle ne peut stocker qu'un seul type de données).

Question 1. Écrivez une fonction `créeGrille(n)` dont la spécification est la suivante :

Entrée n : la taille de la grille d'un côté de la grille.

Sortie une grille vide : c'est-à-dire qu'aucun symbole n'est stockée dans la grille.

Question 2. Écrivez une fonction `tailleCôté( grille )` qui retourne la taille d'un côté de la grille passée en paramètre (c'est-à-dire n).

Question 3. Écrivez une fonction `estVide( grille , i, j)` qui prend en paramètre une grille et deux entiers i et j et qui retourne `True` si et seulement la case à la ligne i et la colonne j ne contient pas de symbole. Si i ou j n'est pas dans l'intervalle $[0, n-1]$, la fonction doit retourner `False`.

Question 4. Écrivez une fonction `écrire ( grille , i, j, symbole)` qui prend en paramètre une grille, deux entiers i et j et une chaîne de caractères `symbole`. La fonction doit affecter dans la grille à la case à ligne i et la colonne j le symbole passé en paramètre. Si la case contient déjà un symbole ou si i ou j n'est pas dans l'intervalle $[0, n-1]$ ou si le symbole n'est pas `'X'` ou `'O'`, la fonction ne doit pas modifier la grille.

1. Le format le plus courant est $n = 3$ mais votre programme doit savoir gérer des grilles carrées de taille quelconque

Question 5. Écrivez une fonction effacer (grille , i , j) qui retire le symbole stocké dans la grille dans la case de la ligne i et la colonne j. Il faut vérifier que i et j sont des valeurs correctes.

Question 6. Écrivez une fonction est (grille , i , j , symbole) qui retourne True si et seulement si la case à la ligne i et la colonne j dans la grille contient le symbole représentée par une chaîne de caractères. Si i ou j n'est pas dans l'intervalle $[0, n-1]$ ou si symbole n'est ni 'X' ou 'O', la fonction doit retourner False.

Question 7. Écrivez une fonction affiche (grille) qui affiche en mode texte dans le terminal la grille passée en paramètre comme illustré ci-dessous.

```
 X |   |   |   |
-------------------
   |   |   | O |
-------------------
   |   |   |   |
-------------------
   |   |   |   |
-------------------
   |   |   |   |
```

Directive : dans le reste de votre programme vous devrez **obligatoirement** utiliser les fonctions définies ci-dessus pour créer, modifier ou accéder à la représentation en mémoire de la grille.

3 Jouer au Tic Tac Toe

Indication : dans un premier temps, on supposera que l'utilisateur entre toujours un entier (éventuellement négatif) quand on lui en demande un.

Le but est de mettre en place une partie de Tic Tac Toe. Au début, on demande la taille de la grille. Il faut que l'utilisateur entre un entier plus grand ou égal à 3. On lui demande la taille tant que cette condition n'est pas respectée.

Ensuite le jeu se déroule jusqu'à ce que la partie soit interrompue, qu'un des joueurs gagne ou que la grille soit pleine sans qu'il n'y ait de vainqueur. Un tour se déroule de la manière suivante :

1. On demande par un message auquel il faut répondre par le caractère 'O' ou le caractère 'N' si la partie doit continuer. On demande tant que l'utilisateur n'a pas répondu par 'O' ou non 'N' en lui indiquant un message d'erreur.
2. On indique à quel joueur c'est le tour.
3. On lui demande le numéro de la ligne où il veut jouer. Si l'utilisateur entre un entier négatif ou un entier strictement plus grand que $n-1$, il faut indiquer quel est le problème et recommencer la demande jusqu'à ce qu'une valeur correcte soit saisie.
4. On lui demande le numéro de la colonne où il veut jouer. Si l'utilisateur entre un entier négatif ou un entier strictement plus grand que $n-1$, il faut indiquer quel est le problème et recommencer la demande jusqu'à ce qu'une valeur correcte soit saisie. On ne redemande que l'indice de la colonne pas celui de la ligne.
5. Si la case n'est pas vide, on l'indique par un message et on recommence la saisie des coordonnées.
6. Le coup doit s'afficher sur la grille graphique.

7. Si quand on demande le numéro de la ligne ou de la colonne, le joueur n'entre rien et appuie juste sur entrée, on revient au début du tour (sans changer de joueur).

A la fin du jeu, on indique par un message si la partie a été interrompu ou si c'est un match nul ou si un joueur a gagné en indiquant lequel.

Voici un exemple d'affichage sur le terminal durant une partie (**Remarque :** il faut aussi afficher l'état de la grille après chaque coup mais cela n'est pas affiché pour ne pas prendre trop de place) :

```
Entrez la taille de la grille : 3
On continue ? [O]ui ou [N]on : O
C'est au tour du joueur X
Entrez le numéro de la ligne (appuyez sur entrée pour annuler la saisie) : 0
Entrez le numéro de la colonne (appuyez sur entrée pour annuler la saisie) : 0
On continue ? [O]ui ou [N]on : O
C'est au tour du joueur O
Entrez le numéro de la ligne (appuyez sur entrée pour annuler la saisie) : 1
Entrez le numéro de la colonne (appuyez sur entrée pour annuler la saisie) : 0
On continue ? [O]ui ou [N]on : O
C'est au tour du joueur X
Entrez le numéro de la ligne (appuyez sur entrée pour annuler la saisie) : 1
Entrez le numéro de la colonne (appuyez sur entrée pour annuler la saisie) :
On continue ? [O]ui ou [N]on : O
C'est au tour du joueur X
Entrez le numéro de la ligne (appuyez sur entrée pour annuler la saisie) :
On continue ? [O]ui ou [N]on : O
C'est au tour du joueur X
Entrez le numéro de la ligne (appuyez sur entrée pour annuler la saisie) : 1
Entrez le numéro de la colonne (appuyez sur entrée pour annuler la saisie) : 0
La case n'est pas vide
Entrez le numéro de la ligne (appuyez sur entrée pour annuler la saisie) : 0
Entrez le numéro de la colonne (appuyez sur entrée pour annuler la saisie) : 1
On continue ? [O]ui ou [N]on : O
C'est au tour du joueur O
Entrez le numéro de la ligne (appuyez sur entrée pour annuler la saisie) : 2
Entrez le numéro de la colonne (appuyez sur entrée pour annuler la saisie) : 1
On continue ? [O]ui ou [N]on : O
C'est au tour du joueur X
Entrez le numéro de la ligne (appuyez sur entrée pour annuler la saisie) : 0
Entrez le numéro de la colonne (appuyez sur entrée pour annuler la saisie) : 2
Le joueur X a gagné
```

4 Saisie d'un entier

On veut maintenant ajouter une protection contre le fait que l'utilisateur peut ne pas entrer un entier quand on lui en demande. Une saisie correcte d'un entier doit suivre le principe suivant :

— un nombre quelconque (cela peut être 0) d'espaces ;
— éventuellement le signe '+' ou le signe '-' ;
— un séquence de chiffres entre 0 et 9, s'il y a un signe, il faut qu'il n'y ait pas d'espace entre le signe et le premier chiffre ;
— un nombre quelconque (cela peut être 0) d'espaces.

Si ce que l'utilisateur saisi ne respecte pas ces règles, il faut le signaler et redemander tant qu'il n'a pas saisie un entier correct. Voici des exemples de saisie non correctes :

```
Entrez la taille de la grille : +
Il faut saisir un entier
Entrez la taille de la grille : +    2
Il faut saisir un entier
Entrez la taille de la grille : timoléon
Il faut saisir un entier
Entrez la taille de la grille :      -23 timoléon
Il faut saisir un entier
Entrez la taille de la grille : 42 23
Il faut saisir un entier
```

5 Mis en place d'un historique

Le but est maintenant de mettre en place un historique des coups joués et de pouvoir annuler des coups joués. Votre programme doit :

- à chaque coup joué, sauvegarder le coup dans l'historique ;
- si l'historique n'est pas vide, affichez à l'écran le dernier coup joué et demander aux joueurs s'ils veulent l'annuler par une question qui doit être répondu par 'O' ou par 'N' (on demande jusqu'à ce que un caractère correct soit saisie) ;
- si le coup est annulé, il faut mettre à jour la grille et l'affichage graphique, retirer le coup de l'historique, et remettre le bon joueur courant.
- le dernier coup enregistré dans l'historique s'il existe devient le prochain coup à annuler.

Voici un exemple d'affichage des interactions :

```
Entrez la taille de la grille : 3
On continue ? [O]ui ou [N]on : O
C'est au tour du joueur X
Entrez le numéro de la ligne (appuyez sur entrée pour annuler la saisie) : 0
Entrez le numéro de la colonne (appuyez sur entrée pour annuler la saisie) : 0
On continue ? [O]ui ou [N]on : O
Dernier coup joué = (0, 0, 'X')
Voulez-vous annulez ce coup ? [O]ui ou [N]on : N
C'est au tour du joueur O
Entrez le numéro de la ligne (appuyez sur entrée pour annuler la saisie) : 1
Entrez le numéro de la colonne (appuyez sur entrée pour annuler la saisie) : 0
On continue ? [O]ui ou [N]on : O
Dernier coup joué = (1, 0, 'O')
Voulez-vous annulez ce coup ? [O]ui ou [N]on : O
On continue ? [O]ui ou [N]on : O
Dernier coup joué = (0, 0, 'X')
Voulez-vous annulez ce coup ? [O]ui ou [N]on : N
C'est au tour du joueur O
Entrez le numéro de la ligne (appuyez sur entrée pour annuler la saisie) : 1
Entrez le numéro de la colonne (appuyez sur entrée pour annuler la saisie) : 0
On continue ? [O]ui ou [N]on : O
Dernier coup joué = (1, 0, 'O')
Voulez-vous annulez ce coup ? [O]ui ou [N]on : N
C'est au tour du joueur X
Entrez le numéro de la ligne (appuyez sur entrée pour annuler la saisie) : 1
Entrez le numéro de la colonne (appuyez sur entrée pour annuler la saisie) :
On continue ? [O]ui ou [N]on : O
Dernier coup joué = (1, 0, 'O')
Voulez-vous annulez ce coup ? [O]ui ou [N]on : N
```

```
C’est au tour du joueur X
Entrez le numéro de la ligne (appuyez sur entrée pour annuler la saisie) :
On continue ? [O]ui ou [N]on : O
Dernier coup joué = (1, 0, ’O’)
Voulez-vous annulez ce coup ? [O]ui ou [N]on : N
C’est au tour du joueur X
Entrez le numéro de la ligne (appuyez sur entrée pour annuler la saisie) : 1
Entrez le numéro de la colonne (appuyez sur entrée pour annuler la saisie) : 0
La case n’est pas vide
Entrez le numéro de la ligne (appuyez sur entrée pour annuler la saisie) : 0
Entrez le numéro de la colonne (appuyez sur entrée pour annuler la saisie) : 1
On continue ? [O]ui ou [N]on : O
Dernier coup joué = (0, 1, ’X’)
Voulez-vous annulez ce coup ? [O]ui ou [N]on : N
C’est au tour du joueur O
Entrez le numéro de la ligne (appuyez sur entrée pour annuler la saisie) : 2
Entrez le numéro de la colonne (appuyez sur entrée pour annuler la saisie) : 1
On continue ? [O]ui ou [N]on : O
Dernier coup joué = (2, 1, ’O’)
Voulez-vous annulez ce coup ? [O]ui ou [N]on : N
C’est au tour du joueur X
Entrez le numéro de la ligne (appuyez sur entrée pour annuler la saisie) : 0
Entrez le numéro de la colonne (appuyez sur entrée pour annuler la saisie) : 2
Le joueur X a gagné
```

6 Indications de notation (non contractuel)

— Représentation en mémoire de la grille : 5 points
— Jeu qui marche : 5 points
— Lecture des entiers : 5 points
— Historique : 5 points
— Malus pour code non propre, mauvais noms de variables ou de fonctions, découpage pouvant être amélioré des fonctions, généralisation des fonctions, factorisation des fonctions, code inutilement compliqué ...

7 Consignes

— Le projet se fait en binôme.
— Le langage de programmation est TypeScript. Un projet rendu en tout autre langage aura la note 0.
— Si le programme comporte des erreurs de syntaxe, la note sera 0.
— Il est interdit d'utiliser des notions de TypeScript non vues en cours. Le non respect de cette consigne entraînera une note de 0.
— Le programme ne doit pas lever d'exception s'il est utilisé correctement. Il vaut mieux ne pas mettre une fonctionnalité qui causerait un bug. Le non respect de cette consigne entraînera une note de 0.
— Tout code recopié d'internet ou d'une source quelconque et non identifié comme tel sera considéré comme du plagiat. La note du projet sera 0.
— Un projet qui serait trop constitué de code recopié d'internet ou d'une autre source ne sera pas considéré comme un travail personnel et entraînera la note de 0.
— Si des codes rendus sont avérés être des copies les uns des autres entre des étudiants, la note sera 0 pour tous les étudiants concernés.
— Si les noms des variables et des fonctions ne sont pas explicites, la note finale sera 0.
— S'il n'y a pas un découpage à minima des fonctions, la note sera 0.
— L'utilisation de variables globales (autrement dit une fonction ne doit pas accéder à une variable qui n'est ni un de ses paramètres, ni déclarée localement) entraînera une note de 0.
— Le projet sera à déposer sur ARCHE dans un dépôt prévu à cet effet. L'ensemble du code sera à placer dans un fichier tictactoe.ts (seul ce fichier sera à rendre).
— La date limite de rendu du projet sera fixé ultérieurement.

Un site donnant quelques conseils sur ce qu'est un code "propre" [2] : `https://damien.pobel.fr/post/clean-code/`

2. Les remarques vont bien au-delà de ce que l'on fait dans ce cours.