**EXERCICE RECAPITULATIF : TABLEAUX-STRUCTURES-FONCTIONS**

On vous demande d’écrire un **programme complet en C** qui permet d’effectuer des parties de MasterMind et de retenir les scores des joueurs dans le tableau ***TabJeu***.

Chaque cellule de ce tableau correspond à un joueur et reprend :

* le ***pseudo*** du joueur (15 caractères maximum)
* le ***score*** du joueur (un entier égal aux nombres de parties gagnées)

**regle du jeu :**  
L’initiateur du jeu crée une combinaison secrète composée de 4 pions qu’il choisit parmi 8 pions "couleur" (B pour bleu – F pour fuchsia - J pour jaune - M pour mauve – N pour noir - O pour orange - R pour rouge et V pour vert) et la place dans TabCombi .   
Les 4 pions de la combinaison secrète ne doivent pas être spécialement différents !

Le joueur dispose ensuite de dix tentatives maximum pour découvrir la combinaison secrète. A chacune de ses propositions (TabPropo) , le joueur reçoit en retour les informations suivantes : le nombre de pions de la bonne couleur et bien placés et le nombre de pions de la bonne couleur mais mal placés.   
La partie se termine lorsque la combinaison secrète est trouvée (4 pions de la bonne couleur et bien placés) ou lorsque le nombre d’essais maximum est atteint.

**Exemple de déroulement d’une partie :**

|  |  |  |  |
| --- | --- | --- | --- |
| V | V | R | J |
|  |  |  |  |  |
| V | V | R | J |
|  |  |  |  |
|  |  |  |  |
| V | R | R | V |
| X | V | X | J |
| Y | R | Y | V |
|  |  |  |  |
| X | X | X | J |
| Y | R | Y | Y |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| V | V | R | J |
|  |  |  |  |
| V | V | R | J |
| X | X | X | X |
| Y | Y | Y | Y |
|  |  |  |  |
|  |  |  |  |
| X | X | X | X |
| Y | Y | Y | Y |
|  |  |  |  |  |
|  |  |  |  |  |

Combinaison secrète (TabCombi )

**1er essai :**

Copie combinaison secrète (TabCombiTest)

Proposition (TabPropo)

Suite à la fonction ***testBienPlace***  (TabCombiTest)

(TabPropo)

* bbp = 2

Suite à la fonction ***testMalPlace*** (TabCombiTest)

(TabPropo)

* bmp = 1

**2ième essai :**

Copie combinaison secrète (TabCombiTest)

Proposition (TabPropo) :

Suite à la fonction ***testBienPlace***  (TabCombiTest)

(TabPropo)

* bbp = 5

Suite à la fonction ***testMalPlace***  (TabCombiTest)

(TabPropo)

* bmp = 0

**Remarques :**

* suivre le DA en respectant scrupuleusement la démarche proposée ainsi que le nom des variables
* toute constante doit être définie comme une constante symbolique
* la structure correspondant à un joueur est à décrire avant la fonction principale
* tout module est une fonction à décrire avant la fonction principale

**description des fonctions :**

* la fonction ***menu*** a pour but d’afficher les couleurs des pions sous le format suivant :

"Combinaison composée de 4 pions de couleur - Tapez :

B pour bleu   
 F pour fuchsia

J pour jaune  
 M pour mauve

N pour noir

O pour orange

R pour rouge

V pour vert "

* la fonction ***initJeu*** a pour but de remplir la combinaison gagnante avec quatre pions "couleur" donnés par l’initiateur de la partie. *// A ne pas décrire – écrire uniquement le prototype*
* la fonction ***obtention*** a pour but d’obtenir les quatre pions "couleur" d’une combinaison, pion par pion dans un tableau de quatre caractères et de le renvoyer au programme appelant ;
* la fonction ***copie*** a pour but de copier un tableau de caractères représentant une combinaison dans un autre tableau, pion par pion ;
* la fonction ***testBienPlace*** a pour but de comparer la combinaison proposée à la combinaison gagnante et de déterminer et renvoyer au programme appelant le nombre de pions bien placés. Lorsqu’un pion est bien placé, il faut remplacer la valeur du pion dans la combinaison gagnante par le caractère ‘X’ et dans la combinaison proposée par le caractère ‘Y’ (afin de ne pas en tenir compte dans le comptage des pions mal placés).
* la fonction ***testMalPlace*** a pour but de comparer la combinaison proposée à la combinaison gagnante et de déterminer et renvoyer au programme appelant le nombre de pions de couleur présent dans la combinaison mais mal placés. Lorsqu’un pion est trouvé, on remplace également la valeur du pion dans la combinaison gagnante par le caractère ‘X’ et dans la combinaison proposée par le caractère ‘Y’ (afin de ne plus en tenir compte pour la suite). *// A ne pas décrire – écrire uniquement le prototype*
* la fonction ***ajoutTab*** permet de mettre à jour le tableau ***TabJeu*** garni de ***nj*** éléments à partir du pseudo du joueur ***pseudoJ*** et du nombre de pions bien placés ***bbp***.   
  Deux cas sont à prendre en considération :  
  - si le joueur est déjà dans le tableau, mettre à jour son score (en cas de partie gagnée- bbp=4) ;

- si le joueur n’est pas dans le tableau, l’ajouter (pseudo et score – 0 ou 1 selon le cas).

* la fonction ***afficheTab*** a pour but d’afficher le pseudo et le score des joueurs repris dans le tableau ***TabJeu*** ; *// A ne pas décrire – écrire uniquement le prototype*

\* MasterMind  
  
 // Déclaration de toutes les variables et initialisation de la variable ***nj*** à 0

sortir "Nouvelle partie ? OUI/NON "

obtenir ***Rep***

tant que ***Rep*** est égal à "OUI"

↓

*initJeu*

↓ TabCombi

sortir "Joueur - entrez votre pseudo :"

obtenir ***pseudoJ***

***bbp*** =0  
***essai*** = 1

tant que ***bbp*** est inférieur à 4 et ***essai*** est inférieur ou égal à NE (nombre d’essais max)

↓ TabCombi

*copie*

↓ TabCombiTest

sortir « essai n° », ***essai***

↓

*menu*

↓

*obtention*

↓ TabPropo

↓ TabPropo,TabCombiTest

*testBienPlace*

↓ bbp, TabPropo,TabCombiTest

↓ TabPropo,TabCombiTest

*testMalPlace*

↓ bmp

sortir "Pions corrects : ",***bbp***, " bien place(s) et ",***bmp*** " mal place(s)"

incrémenter ***essai*** de 1

si ***bbp*** est égal à 4

sortir pseudoJ,", vous avez gagne au bout de ", essai, "essai(s) !"

sinon

sortir "Vous avez perdu !"

↓ pseudoJ,bbp,TabJeu,nj

*ajoutTab*

↓ nj

sortir "Nouvelle partie ? OUI/NON "

obtenir ***Rep***

↓ TabJeu,nj

*afficheTab*

↓