UE Programmation

TP sur le chapitre 5 (étude de cas)

On souhaite programmer un jeu de type puissance 4, sans interface graphique, jouable soit à deux joueurs, soit à un joueur contre l'ordinateur.

Exercice 1 : Lisez sur Internet (par exemple, sur Wikipédia) les règles du jeu puissance 4.

Exercice 2 : Proposez une structure de données adaptée pour stocker la grille du jeu. Comment représentez-vous les cases vides, les cases ayant un pion du joueur 1, et les cases ayant un pion du joueur 2 ?

On va tout d'abord s'intéresser au jeu à deux joueurs, plus facile à programmer. Puis, on programmera l'intelligence artificielle permettant de jouer contre l'ordinateur. Vous n'hésiterez pas à modifier après chaque exercice la fonction `main`, afin de tester les fonctions que vous venez de programmer.

Par soucis de généralité, le nombre de colonnes (7 pour le puissance 4) sera défini dans la constante `COLONNES` de votre programme : il suffira de modifier cette constante pour que le jeu se joue sur un nombre de colonnes différent. De même, le nombre de lignes (6 pour le puissance 4) sera défini dans la constante `LIGNES`, et le nombre de pions à aligner (4 dans le puissance 4) sera défini dans la constante `PUISSANCE`.

Partie 1 : jeu à deux joueurs

Exercice 3 : Écrivez la fonction `initGrille`.

Exercice 4 : Écrivez la fonction `afficheGrille`.

Exercice 5 : Écrivez la fonction `coupJoueur` qui prend en paramètre la grille et le numéro du joueur, et qui :

- demande au joueur (en affichant son numéro) de saisir un numéro de colonne,
- s'assure que ce numéro est correct (c'est-à-dire qu'il est bien compris entre 1 et COLONNES, et que la colonne correspondante n'est pas remplie),
- retourne ce numéro de colonne.

Exercice 6 : Écrivez la fonction `mettrePion` qui prend en paramètre la grille, le numéro du joueur et la colonne choisie, et qui met un pion à ce joueur dans cette colonne.

Exercice 7 : Écrivez la fonction `aGagne` qui prend en paramètre la grille, le numéro d'un joueur, et la dernière colonne jouée, et renvoie toujours *false*.
Remarque : Cette fonction sera modifiée plus tard pour déterminer si le joueur a gagné ou non. En attendant, avoir une fonction « vide » permet de commencer à tester le jeu (sans la fin).

Exercice 8 : Écrivez une fonction `jeu` qui initialise la grille de jeu et renvoie un entier (égal à 1 pour le moment). Cette fonction aussi sera modifiée plus tard.

Exercice 9 : Écrivez la fonction `main` qui appelle la fonction `jeu`, puis affiche quel joueur a gagné : soit le joueur 1, soit le joueur 2, soit aucun des deux joueurs. Testez cette fonction `main` en

faisant retourner différents entiers à la fonction `jeu`.

Exercice 10 : Modifiez la fonction `jeu` pour qu'elle corresponde à présent au jeu complet. On rappelle que le jeu s'arrête soit quand un joueur a gagné (ce qui est indiqué par la fonction `aGagne`, encore incomplète), soit quand toutes les cases sont occupées par des pions.
Remarque : Dans cette fonction, la valeur 42 ne doit pas apparaître, bien qu'il s'agisse du nombre total de cases du puissance 4 traditionnel. Vous pourrez soit déterminer s'il y a `LIGNES*COLONNES` pions joués, soit déterminer si toutes les colonnes sont pleines.

Exercice 11 : Écrivez la fonction `alignementHorizontal` qui prend en paramètre la grille et la dernière colonne jouée, et indique si le dernier pion joué permet d'aligner `PUISSANCE` pions horizontalement. La fonction renvoie *true* si c'est le cas, et *false* sinon.

Exercice 12 : Écrivez la fonction `alignementVertical` qui indique si le dernier pion joué permet d'aligner `PUISSANCE` pions verticaux.

Exercice 13 : Écrivez la fonction `alignementSlash` qui indique si le dernier pion joué permet d'aligner `PUISSANCE` pions en diagonale dans le sens haut-droite / bas-gauche.

Exercice 14 : Écrivez la fonction `alignementAntiSlash` qui indique si le dernier pion joué permet d'aligner `PUISSANCE` pions en diagonale dans le sens haut-gauche / bas-droit.

Exercice 15 : Modifiez la fonction `aGagne`.

Partie 2 : jeu contre l'ordinateur

Exercice 16 : Modifiez la fonction `main` pour que le nombre de joueurs humains (1 ou 2) soit saisi en début de partie.

Exercice 17 : Créez une fonction `coupOrdinateur` qui renvoie un nombre aléatoire entre 1 et `COLONNES`.

Exercice 18 : Modifiez le programme de la manière suivante :
- passez le nombre de joueurs humains en paramètre de la fonction `jeu`,
- lorsque c'est au deuxième joueur de jouer et qu'il n'y a qu'un seul joueur humain, appelez la fonction `coupOrdinateur` au lieu de `coupJoueur`.

Exercice 19 : Modifiez la fonction `coupOrdinateur` pour qu'elle procède de la manière suivante :
- si l'une des colonnes permet de gagner, l'ordinateur joue cette colonne,
- sinon, si l'une des colonnes permet au joueur humain de gagner, l'ordinateur joue cette colonne,
- sinon, l'ordinateur joue aléatoirement une colonne non remplie.

Pour tester si l'une des colonnes permet à un joueur de gagner, vous pourrez faire une copie de la grille et appeler les fonctions `mettrePion` et `aGagne`.