DÉPARTEMENT
Sciences Informatiques

DU Développeur niveau II

Algorithmique et Python

Mini projet : puissance 4

© JLB

Dans le cadre de l'évaluation du module « Algorithmique et Python », nous vous proposons de développer un jeu de **Puissance 4** avec, dans un premier temps, son étude algorithmique puis, dans un second temps, son codage en Python.

1) Modélisation d'un jeu de Puissance 4

Nous allons donc programmer un jeu de **Puissance 4** en ligne de commande (les plus rapides et motivés pourront éventuellement s'amuser à créer une interface graphique MAIS SEULEMENT SI le programme en ligne de commande est complet et fonctionnel !).

Rappel de la règle du jeu

Le jeu de **Puissance 4** se joue à deux joueurs avec une grille de 7 colonnes et 6 rangées. Un joueur possède des jetons rouges et l'autre des jetons jaunes. Les joueurs mettent chacun leur tour un jeton dans une colonne. Le but est d'être le premier à aligner 4 jetons verticalement, horizontalement ou en diagonale. Si personne n'arrive à aligner 4 jetons avant que la grille ne soit remplie il y a match nul.

Premières réflexions

La grille du jeu sera représenté par un tableau à deux dimensions. Par exemple, les `0` correspondent à des cases vides, les `1` aux jetons du joueur 1 et les `2` aux jetons du joueur 2 :

```
+---+---+---+---+---+---+---+
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
+---+---+---+---+---+---+---+
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
+---+---+---+---+---+---+---+
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
+---+---+---+---+---+---+---+
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
+---+---+---+---+---+---+---+
| 0 | 0 | 2 | 1 | 0 | 0 | 0 |
+---+---+---+---+---+---+---+
| 0 | 0 | 1 | 2 | 0 | 0 | 0 |
+---+---+---+---+---+---+---+
```

Pour votre modélisation, je vous propose d'utiliser les fonctions suivantes :

— `grille_init()` : fontion qui renvoie un tableau de 6 lignes et 7 colonnes remplies de zéros ;

— `affiche_grille(tab)` : fonction qui affiche la grille du jeu dans la console de la façon la plus esthétique possible (voir la suggestion ci-dessus à éventuellement améliorer) ;
— `colonne_libre(tab,colonne)` : fonction qui renvoie un booléen indiquant s'il est possible de mettre un jeton dans la colonne (indique si la colonne n'est pas pleine) ;
— `place_jeton(tab,colonne,joueur)` : fonction qui place un jeton du joueur (1 ou 2) dans la colonne. Elle renvoie la grille modifiée ;
— `horizontale(tab,joueur)` : fonction qui renvoie `True` si le joueur a au moins 4 jetons alignés dans une ligne ;
— `verticale(tab,joueur)` : fonction qui renvoie `True` si le joueur a au moins 4 jetons alignés dans une colonne ;
— `diagonale(tab,joueur)` : fonction qui renvoie `True` si le joueur a au moins 4 jetons alignés dans une diagonale ;
— `gagne(tab,joueur)` : fonction qui renvoie `True` si le joueur a gagné ;
— `tour_joueur(tab,joueur)` : fonction qui permet au joueur de placer un jeton dans la colonne choisie. Elle indique si la colonne est pleine et permet alors au joueur de choisir une autre colonne ;
— `egalite(tab)` : fonction qui renvoie `True` s'il y a égalité et `False` sinon ;
— `jouer(tab)` : fonction qui permet aux deux joueurs de jouer chacun leur tour. Elle vérifie que les joueurs n'ont pas gagné à la fin de leur tour. Si l'un des deux a gagné ou s'il y a égalité, elle donne le résultat.

Barême indicatif

Tâche	Algorithme	Code Python
`grille_init()`	1	1
`affiche_grille(tab)`	2	2
`colonne_libre(tab,colonne)`	1	1
`place_jeton(tab,colonne,joueur)`	2	2
`horizontale(tab,joueur)`	2	2
`verticale(tab,joueur)`	1	1
`diagonale(tab,joueur)`	3	3
`gagne(tab,joueur)`	1	1
`tour_joueur(tab,joueur)`	1	1
`egalite(tab)`	1	1
`jouer(tab)`	1	1
commentaires	4	1
documentation	-	3
Total	**20**	**20**

Enfin, des points supplémentaires seront acquis pour :

Tâche	
respect des délais	1
respect des formats	1
algorithme/code bien construit	1
interface améliorée	2
Total	**5**

Cette note globale sur 45 points sera finalement ramenée à un total de 20.

2) Modélisation : les rendus obligatoires

La modélisation algorithmique du programme principal et des différentes fonctions, y compris la spécification des variables pour chacune des parties, est à rédiger et à envoyer par mail (`jean-luc.bourdon@u-cergy.fr`) sous forme d'un document PDF placé dans une archive nommée `DU_II-NOM_Prenom-AlgoPython-Projet_Algo.zip` uniquement et au plus tard le **lundi 31 octobre**, **22h00**. Tout envoi hors délai se verra pénalisé (0,5 pt par demie journée de retard avec un maximum de 2 pts de pénalité).

3) Programmation (codage et tests) : les rendus obligatoires

Le codage en Python du programme principale et des différentes fonctions sera à envoyer par mail (jean-luc.bourdon@u-cergy.fr) sous forme d'un fichier texte d'extension **.py** placé dans une archive nommée `DU_II-NOM_Prenom-AlgoPython-Projet_Code.zip` uniquement et au plus tard le **lundi 7 novembre**, **16h00**. Tout envoi hors délai se verra pénalisé (0,5 pt par demie journée de retard avec un maximum de 2 pts de pénalité).