

INFO-F101 – Programmation

Le démineur

Année académique 2023-2024



Introduction

Le démineur est un jeu vidéo popularisé par Microsoft qui l'a fourni de manière standard avec son système d'exploitation Windows à partir de la version 3.1 jusqu'à la version 7. Il s'agit d'un jeu de réflexion pour un joueur unique. Le joueur est amené à essayer de trouver l'emplacement de bombes disséminées sur une grille à partir de l'information du nombre de bombes adjacentes aux cases dévoilées.

Au début du jeu, toutes les cases sont masquées. A chaque tour de jeu, le joueur choisit une case à démasquer. Si la case contient une bombe, il a perdu. Si la case ne contient pas de bombe mais qu'au moins une de ses cases limitrophes en contient une, un chiffre indiquant le nombre de bombes voisines est dévoilé. Enfin, si la case n'est attenante à aucune bombe, les cases adjacentes sont dévoilées de proche en proche jusqu'à arriver aux frontières de la grille ou à une case au voisinage explosif. Dans ce cas, l'information du nombre de bombes limitrophes à cette case est révélée. Lorsque toutes les cases ont été dévoilées — à l'exception des cases masquant une bombe — le joueur a gagné.

Différentes variantes du jeu existent. Dans certaines, il n'y a pas de hasard et le joueur a la certitude de pouvoir retrouver l'emplacement de toutes les bombes à partir de la déduction. Dans ce cas-là, le premier coup est un cas particulier car le joueur n'a encore aucune information lui permettant de mener cette déduction. Le jeu peut adapter la grille afin que le premier coup ne soit jamais perdant ou une convention où les quatre coins ne contiennent jamais de bombe peut-être adoptée.

Nous vous demandons dans le cadre de ce projet d'implémenter en python3 une version simplifiée du jeu décrite ci-après.

La grille

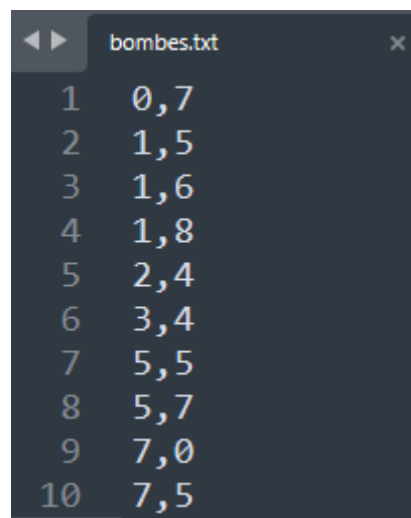
La variante du démineur proposée sera constituée de 3 niveaux.

- **Niveau 0 (Facile)** : La grille de jeu est de taille $N=9$ et la position des bombes est lue dans le fichier bombes.txt
- **Niveau 1 (Moyen)** : La grille de jeu de taille N donnée par le joueur avec $\frac{N*N}{5}$ bombes placées aléatoirement dans la grille
- **Niveau 2 (Difficile/WTF!)** : Comme pour le niveau 2, sauf que les bombes se déplacent à la fin de chaque tour du joueur si celui-ci n'a pas sélectionné une case contenant une bombe. Ainsi, une nouvelle grille de jeu est proposée à chaque tour. Celle-ci aura la même avancée de jeu mais des bombes placés différemment. Pensez-donc à recalculer les bombes adjacentes à vos cases déjà ouvertes.

Vous devez considérer une division entière dans le cas où le nombre de bombe est un nombre flottant. Les grilles considérées sont carrées de taille N donnée par le joueur en début de partie pour les niveaux 2 et 3. Pour la difficulté facile, votre jeu utilisera une grille existante qui lui sera transmise sous la forme d'un dictionnaire. Les clés du dictionnaire sont des tuples (ordonnée,abscisse) identifiant une case de la grille. Une bombe est indiquée en créant une entrée dans le dictionnaire associant au tuple identifiant la case la chaîne de caractères "B" comme valeur. Seules les cases comportant une bombe sont reprises dans le dictionnaire.

Outre les bombes, qui doivent impérativement être dans un dictionnaire comme mentionné ci-dessus, vous êtes libres d'utiliser les structures de données qui vous semblent les plus judicieuses au sein de votre programme.

La grille du niveau 1 est de taille $N=9$ et les positions des bombes sont récupérés dans le fichier bombes.txt .



1	0,7
2	1,5
3	1,6
4	1,8
5	2,4
6	3,4
7	5,5
8	5,7
9	7,0
10	7,5

Affichage

Lors de chaque tour de jeu, vous afficherez préalablement une représentation de la grille dans la console à partir de caractères (il n'y a pas d'interface graphique). Vous demanderez ensuite au joueur la case qu'il souhaite démasquer via son ordonnée et son abscisse.

Vous réalisez l'affichage à votre meilleure convenance mais il doit être intelligible pour le joueur, voici un exemple :

```
      1   2   3   4   5   6   7   8   9
-----
A | * | * | * | * | * | * | * | * |
-----
B | * | * | * | * | * | * | * | * |
-----
C | * | * | * | * | * | * | * | * |
-----
D | * | * | * | * | * | * | * | * |
-----
E | * | * | * | * | * | * | * | * |
-----
F | * | * | * | * | * | * | * | * |
-----
G | * | * | * | * | * | * | * | * |
-----
H | * | * | * | * | * | * | * | * |
-----
I | * | * | * | * | * | * | * | * |
-----
```

Veillez entrer la lettre d'une ligne :

Dans cet exemple, le caractère * signifie que la case n'a pas encore été dévoilée au joueur.

Début de la partie

Au début de la partie, le jeu demandera au joueur la difficulté :

» » Veuillez entrer un chiffre pour choisir la difficulté Facile (0), Moyen (1), Difficile (2) :

Les valeurs 0, 1, 2 représentent respectivement Facile, Moyen, Difficile. Pour les difficultés Moyen et Difficile, il demandera ensuite la taille N de la grille à générer. Il vous est également demandé de ne pas mettre de bombe dans les coins de votre grille (0,0), (0,N-1), (N-1,0), (N-1,N-1) , ceci pour permettre au joueur d'avoir un premier coup assuré.

Tours de jeu

A chaque tour de jeu, le joueur devra choisir une case qui est un couple ligne et colonne. Pour ce faire, il sera demandé au jeu de choisir une ligne et ensuite la colonne. Lorsque le joueur a choisi sa case, différentes possibilités peuvent advenir :

- Le joueur a gagné : si toutes les cases vides ont été trouvées. Vous afficher un message de félicitations et terminez le programme.
- Le joueur a perdu : il a sélectionné une case sous laquelle est dissimulée une bombe, vous en informez le joueur et le programme se termine également.
- Le joueur a choisi une case ayant une ou plusieurs bombes directement adjacentes, seule cette case est alors dévoilée avec un chiffre indiquant le nombre de bombes adjacentes, le jeu continue, la grille est affichée et le joueur est à nouveau amené à choisir une case.
- Le joueur a choisi une case sans bombe sur les cases adjacentes. La version du jeu que nous vous demandons de réaliser est ici simplifiée. Une aire n'est pas dévoilée mais uniquement les 8 cases directement limitrophes à la case sélectionnée. Avec, à chaque fois, un chiffre indiquant le nombre de bombes adjacentes à cette case ou rien si il n'y a pas de bombe adjacente. Le jeu continue ensuite normalement.

Exemple d'exécution

Exemple début de partie

```
###
Welcome in the minesweeper game.
Try to find all cases without bombs.
Good luck !
###
Choose your difficulty.
For project Grid press 0 , easy press 1 , hard press 2 : 0
We planted 10 bombs. Good luck !
      1  2  3  4  5  6  7  8  9
-----
A | # | # | # | # | # | # | # | # |
-----
B | # | # | # | # | # | # | # | # |
-----
C | # | # | # | # | # | # | # | # |
-----
D | # | # | # | # | # | # | # | # |
-----
E | # | # | # | # | # | # | # | # |
-----
F | # | # | # | # | # | # | # | # |
-----
G | # | # | # | # | # | # | # | # |
-----
H | # | # | # | # | # | # | # | # |
-----
I | # | # | # | # | # | # | # | # |
-----

### Round 1
--> 71 cases to find.
Choose your line (A to I) :
```

Exemple niveau 2

```
### Round 6
--> 61 cases to find.
Choose your line (A to I) : D
Choose your column (1 to 9) : 7
  1  2  3  4  5  6  7  8  9
-----
A |  |  |  | # | # | # | # | # |
B |  |  |  | # | # | # | # | # |
C | # | # | # | # | # | 4 | 2 | 2 | 1 |
D | # | # | # | # | # | 2 |  |  | # |
E | # | # | # | # | # | 2 | 2 | 1 | # |
F | # | # | # | # | # | # | 2 | # | # |
G | # | # | # | # | # | # | # | # | # |
H | # | # | # | # | # | # | # | # | # |
I | # | # | # | # | # | # | # | # | # |
-----

### Round 7
--> 54 cases to find.
Choose your line (A to I) : C
Choose your column (1 to 9) : 5
  1  2  3  4  5  6  7  8  9
-----
A |  |  |  | # | # | # | # | # | # |
B |  |  |  | # | # | # | # | # | # |
C | # | # | # | # | B | 4 | 2 | 2 | 1 |
D | # | # | # | # | # | 2 |  |  | # |
E | # | # | # | # | # | 2 | 2 | 1 | # |
F | # | # | # | # | # | # | 2 | # | # |
G | # | # | # | # | # | # | # | # | # |
H | # | # | # | # | # | # | # | # | # |
I | # | # | # | # | # | # | # | # | # |
-----

###
BOUM, You lost !
###
PS C:\Users\yarsl\OneDrive\Bureau>
```

Exemple niveau 3

Dans cet exemple, un print du dictionnaire a été ajouté uniquement pour vous aider à comprendre le changement entre le round 3 et le round 4.

```
### Round 3
--> 39 cases to find.
Choose your line (A to G) : A
Choose your column (1 to 7) : 5
  1  2  3  4  5  6  7
-----
A | # | # | # | # | 3 | # | # |
B | # | 3 | # | # | # | # | # |
C | # | # | # | # | # | # | # |
D | # | # | # | # | # | # | # |
E | # | # | # | # | # | # | # |
F | # | 1 | # | # | # | # | # |
G | # | # | # | # | # | # | # |
-----
--> You succesfully passed this step, a new grid is now generated !
{(2, 2): 'B', (5, 4): 'B', (6, 1): 'B', (4, 4): 'B', (6, 5): 'B', (2, 1): 'B', (4, 0): 'B', (1, 6): 'B'}
### Round 4
--> 37 cases to find.
Choose your line (A to G) : A
Choose your column (1 to 7) : 1
  1  2  3  4  5  6  7
-----
A |  |  | # | # | 3 | # | # |
B | 1 | 2 | # | # | # | # | # |
C | # | # | # | # | # | # | # |
D | # | # | # | # | # | # | # |
E | # | # | # | # | # | # | # |
F | # | 1 | # | # | # | # | # |
G | # | # | # | # | # | # | # |
-----
--> You succesfully passed this step, a new grid is now generated !
{(6, 2): 'B', (0, 5): 'B', (6, 4): 'B', (4, 1): 'B', (2, 0): 'B', (3, 5): 'B', (4, 5): 'B', (1, 3): 'B'}
```

Consignes pour la remise du projet

Les consignes pour la remise du projet sont disponibles en ligne sur la page du cours sur l'Université Virtuelle. Ces consignes sont à respecter *scrupuleusement*; relisez-les attentivement avant la remise !

Pour toute question concernant l'énoncé, nous vous invitons à vous adresser à Yasin Arslan (yasin.arslan@ulb.be).

Date limite de remise : 15 novembre 8h