PROGRAMMATION OBJET AVANCEE

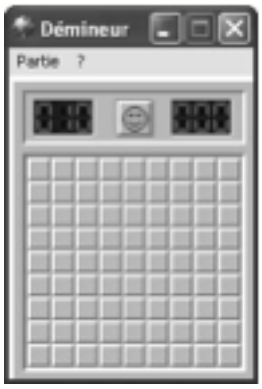
Examen

À rendre par mail au format PDF le dimanche 3 mai à 23h59 au plus tard

Seul(e), en binôme ou en trinôme au maximum

I. DESCRIPTION DU PROBLEME

On souhaite concevoir un jeu de démineur comme celui qui est livré avec le système d'exploitation Microsoft Windows. Le but du jeu est de trouver le plus rapidement possible toutes les cases du plateau contenant des mines sans les toucher.



Copie d'écran du jeu de démineur

Le jeu est composé d'un plateau rectangulaire, d'un chronomètre et d'un compteur de mines. Le plateau est un quadrillage de cases. Au début du jeu, toutes les cases du plateau sont couvertes, le compteur de mines indiquant le nombre de mines restant à localiser. Le chronomètre compte le nombre de secondes écoulées depuis le début de la partie. La partie commence lorsque la première case est découverte.

Quand une case est découverte, son contenu est affiché. Le contenu d'une case peut être : rien, une mine ou un nombre indiquant le nombre de mines présentes dans les cases voisines. Les scénarios suivants peuvent se produire lorsqu'une case est découverte, en fonction de son contenu :

1. Un chiffre – Il ne se passe rien.

2. Un blanc – Toutes les cases voisines sont dévoilées, à condition qu'elles ne soient pas signalées par un drapeau. Si l'une de ces cases voisines ne contient rien, le processus de découverte continue automatiquement à partir de cette case.

3. Une mine – Le jeu est terminé et le joueur a perdu.

Si elle est toujours couverte, une case peut être marquée en respectant les règles suivantes :

• Marquer une case qui n'est ni découverte ni marquée décrémente le compteur de mines restant à localiser et un drapeau apparaît sur la case. Il indique que cette case contient potentiellement une mine. Une case marquée d'un drapeau ne peut pas être découverte.

• Marquer une case déjà signalée d'un drapeau permet de la remettre dans son état initial, à savoir couverte et non marquée. Le compteur de mines est alors incrémenté de 1.

II. QUESTIONS

1 - Développez un diagramme de cas d'utilisation pour le jeu de démineur. On distinguera trois cas d'utilisation : « Jouer une partie de démineur », « Configurer le jeu » et « Consulter l'aide en ligne » .

2 - Développez un diagramme de séquence pour le cas d'utilisation « Jouer une partie de démineur » en suivant les étapes suivantes :

a) Le joueur va passer son temps à découvrir ou marquer des cases. Dans le cas idéal, il va finir par gagner et entrer dans les meilleurs scores (s'il jouait à un niveau prédéfini). Représenter tout cela par une grande boucle (fragment `loop`) dans le diagramme dans laquelle les opérations système consistant à marquer une case et à découvrir une case peuvent arriver dans un ordre quelconque (utiliser l'opérateur `alt`).

b) Ajouter la configuration du jeu comme une étape optionnelle avant de commencer à jouer et représenter le diagramme ainsi obtenu.

c) Représenter dans la boucle de jeu les trois possibilités : perte, gain ou cas normal. Les cas de perte ou gain arrêtent la partie, ce qui peut se représenter par l'opérateur prédéfini `break`. Représenter le diagramme ainsi obtenu.

3) Réalisez un diagramme de classes d'analyse ainsi qu'un diagramme d'états.

La principale difficulté consiste à bien représenter la double variabilité au niveau des cases :
• Le fait qu'une case soit minée ou pas est intrinsèque à la case et reste vrai du début à la fin de sa vie (nouvelle partie).
• Le fait qu'une case soit marquée ou pas varie durant sa vie : il s'agit donc d'une notion d'état.

Le diagramme de classes présenté doit être complété par un diagramme d'états de la classe *Case*.

Notez également qu'une case possède exactement 3 (coin), 5 (bord) ou 8 voisines. Certains attributs sont calculables donc dérivés, et les coordonnées d'une case jouent le rôle de qualificatif.

4) Réalisez les diagrammes de séquence de conception objet pour l'opération système consistant à marquer une case.

Une approche possible est la suivante : prendre comme objet contrôleur la « Partie ». L'objet « Partie » n'est pas l'expert des cases ni de leurs coordonnées. Il va donc déléguer le travail à l'expert des cases, soit le « Plateau ». Celui-ci va tout d'abord récupérer une référence sur la case. Pour cela, il fait appel à une collection triée de cases (une *Map*) pour trouver l'instance de case. Il va ensuite déléguer à celle-ci le traitement du marquage proprement dit.

Le traitement du marquage dépend de l'état de la case. Si l'on veut éviter une logique conditionnelle complexe et peu évolutive, il est intéressant de déporter le comportement variable dans de nouveaux objets et d'utiliser pour cela le design pattern *State* :

(https://en.wikipedia.org/wiki/State_pattern,

https://fr.wikibooks.org/wiki/Patrons_de_conception/État).

Remarquons que de nombreuses classes différentes vont avoir besoin d'un accès au contrôleur *Partie*, pour décrémenter ou incrémenter le compteur de mines comme indiqué sur le schéma précédent, mais aussi pour signifier la fin de la partie (dans le cas de la découverte d'une case minée). Or, la classe *Partie* ne doit avoir qu'une seule instance à la fois. Pour assurer qu'une classe ne sera instanciée qu'une seule fois et donner un accès global à cette instance unique, on pourra utiliser le design pattern *Singleton* :

(https://fr.wikibooks.org/wiki Patrons_de_conception/Singleton).

5) Réalisez les diagrammes de séquence de conception objet pour l'opération système consistant à découvrir une case.

Le début de l'interaction est très similaire à celle de l'opération système consistant à marquer une case. On peut adopter l'approche suivante : l'objet « Partie » va déléguer le travail à l'expert des cases, soit le « Plateau ». Celui-ci va tout d'abord récupérer une référence sur la case. Il va ensuite déléguer à celle-ci le traitement du message « découvrir », qui dépend de son état.

Nous pouvons utiliser également ici le design pattern *State*. Toutefois cette fois-ci, la situation se complique : si la case était dans l'état « Couverte », elle passe à l'état « Découverte », mais le traitement effectué dépend maintenant du type de case : minée ou pas. La méthode « dévoiler », appelée par l'état « Couverte » sur la case est elle-même polymorphe !

En effet, dévoiler interrompt brutalement la partie sur une case minée. Il faut alors arrêter le chronomètre, découvrir toutes les cases minées et signaler toutes les cases marquées à tort avec un X. S'il s'agit d'une case numérotée, il faut vérifier si la partie n'est pas gagnée (toutes les cases minées sont découvertes). Enfin, s'il s'agit d'une case vide, il faut propager le message « découvrir » à toutes les voisines .

6) Dessinez le diagramme de classes de conception objet en indiquant les design patterns utilisés.

* *

*