Le but est de développer une interface permettant de *jouer* au *démineur*.

Le démineur est un jeu classique initialement disponible sous Microsoft Windows. Je ne rappelle pas les principes du jeu que vous trouverez sur `https://fr.wikipedia.org/wiki/Démineur_(genre_de_jeu_vidéo)`. Nous vous demandons de programmer ce jeu (la version classique, en 2D, avec lequel le plateau est une grille carrée de cases carrées).

1 Le noyau fonctionnel (ou modèle)

Le noyau fonctionnel doit permettre de créer et gérer des plateaux carrés de taille $N \times N$. Chaque case peut avoir plusieurs états : présence d'une mine ou non ; présence d'un drapeau ou non ; découverte ou pas. Remarques : on ne peut pas découvrir une case qui possède un drapeau, et découvrir une mine termine la partie. Il faudra prévoir les méthodes permettant de

- manipuler et récupérer l'état d'une cellule précise ;
- découvrir une case (dont l'effet sera différent si la case contient une mine, ne contient aucune mine dans les cases voisines, ou bien si la case est voisine d'une case) ;
- récupérer dans un tableau une image du plateau ;
- initialiser le tableau (toutes les cases sont couvertes, et sans drapeau) en choisissant sa taille et le nombre de mines que l'on souhaite placer aléatoirement ;
- lire et sauvegarder les scores ;
- toute autre méthode que vous jugerez utile dans ce projet.

Le démineur étant très simple, seuls 2 entiers définissent les variantes de votre jeu : la taille N du plateau carré, le nombre m de mines cachées dans le plateau de jeu. Pour ces deux entiers nous devrons avoir 3 choix différents. On pourra démarrer le jeu avec les valeurs centrales.

2 Interface

L'interface devra permettre de jouer au démineur, mais nous demandons quelques spécificités sur cette interface. Votre application devra présenter deux fenêtres :
- la fenêtre *plateau*, elle ne contiendra que le plateau de jeu, c'est sur ce plateau que l'on clique pour poser les drapeaux et découvrir les cases ;
- la fenêtre *commandes*, c'est sur cette fenêtre que l'on pourra modifier les paramêtres du jeu, le stopper, le relancer, voir les scores, etc...

Il n'y a pas grand chose à dire sur *plateau*, un click découvre une case, un click droit pose/retire un drapeau sur une mine, vous devrez dessiner vous même la case (case non découverte avec ou sans drapeau, case découverte case découverte avec le nombre de voisins, mine explosée) ; mais *commandes* devra être organisée ainsi :

- une zone de texte permettant de rentrer le nom de joueur ;
- deux composants (des spinners par exemple) permettant de choisir la dimension de la grille ainsi que le nombre de mines (si l'on est en train de jouer, un changement sur une de ces composants doit ouvrir une fenêtre demandant si l'on veut interrompre le jeu et valider le changement, ou bien continuer à jouer et annuler le changement) ;
- un chronomètre, une simple zone texte affichant les secondes suffit, ce chronomêtre s'arrête lorsque la partie est finie, que la partie soit perdue ou gagnée ;
- un bouton de reset qui arrête le jeu, reinitialise le plateau de jeu et le chronomètre
- un indicateur analogique indiquant le nombre de cases découvertes, cet indicateur déplace une aiguille sur un demi cercle, le nombre 0 est à gauche, le nombre $N \times N - m$ est à droite, l'aiguille avance au fur et à mesure que le joueur découvre les case ;
- un bouton permettant d'afficher les 10 meilleurs scores.
- un bouton *Quit* en bas à droite de la fenêtre.

3 Travail demandé

Ce projet doit être réalisé en binôme. Pour l'interface, vous ne devez utiliser que les classes standards de swing (pas d'autres composants tous faits). Le code doit être propre, modulaire, bien indenté, documenté. Les

consignes données en début de semestre s'appliquent.

Pour ce projet vous devrez :

— réaliser le noyau fonctionnel de l'application.

— implémenter l'interface, vous devrez respecter l'agencement proposé pour les fenêtres, mais couleurs, décorations de boutons, et autres aspects esthétiques sont laissés à votre libre choix,

— gérer toutes les actions demandées par l'interface, boutons, souris, box, et zones de saisie diverses,

— gérer les fenêtres de confirmation pour les 2 manipulations changeant la configuration initiale du jeu.

Attention, vos fenêtres doivent pouvoir être redimensionnées jusqu'à un certain point sans que cela ne provoque de «catastrophe». Le code doit être aussi «propre» que possible : pas de redondance de code, un code lisible, modulaire (par exemple, changer la taille par défaut du plateau de jeu ne doit pas prendre plus de 10 secondes), et commenté.

L'ensemble des sources appartiendra au package `demineurjeu`, plus précisément, le noyau fonctionnel dans le package `demineur.model`, et l'IHM dans le package `demineur.ihm`. Libre à vous d'ajouter autant de sous-packages que nécessaires.

Vous rendrez un rapport d'au plus 20 pages, au format pdf (**à l'exclusion de tout autre**), expliquant comment est structurée votre application, comment vous avez résolu les problèmes de conception rencontrés, et comment votre application peut être étendue pour produire l'application rêvée.

Votre travail sera rendu sur UPdago dans une archive `nom1_nom2.tar.tgz`, où `nom1` et `nom2` sont évidemment vos noms. À l'intérieur de cette archive, on trouvera le répertoire `nom1_nom2` qui contiendra le rapport et le package `demineur` contenant votre code source.

Nous nous réservons le droit de convoquer certains binômes à un oral pour d'éventuelles précisions sur le travail rendu. Il est bien entendu qu'on attend un travail original (c'est à dire personnel, tout plagiat serait très fortement pénalisé), et équitablement réparti au sein de chaque binôme.

À noter enfin que les différentes consignes données dans les documents déjà distribués s'appliquent à ce projet.

4 Évaluation

Les critères pris en compte pour la notation :

— les différentes consignes sont respectées (structuration et format de l'archive rendue, encodage des caractères, etc...) ;

— le code est propre, modulaire, lisible, documenté... ;

— le code rendu compile et le programme peut être lancé et manipulé ;

— l'application est ergonomique ;

— si vous avez repéré des erreurs dans votre application, sans avoir le temps de les corriger, elles sont documentées dans le rapport ;

— le rapport donne les informations nécessaires pour comprendre votre démarche et utiliser votre application.