Projet 3

Date de remise : le jour de l'examen final à 23h55

Objectifs d'apprentissage

- Analyser un problème
- Créer une interface graphique conviviale
- Utiliser des structures de données avancées
- Implanter une application robuste
- Utiliser Javadoc

Type d'évaluation Sommative (10% de la note du cours)

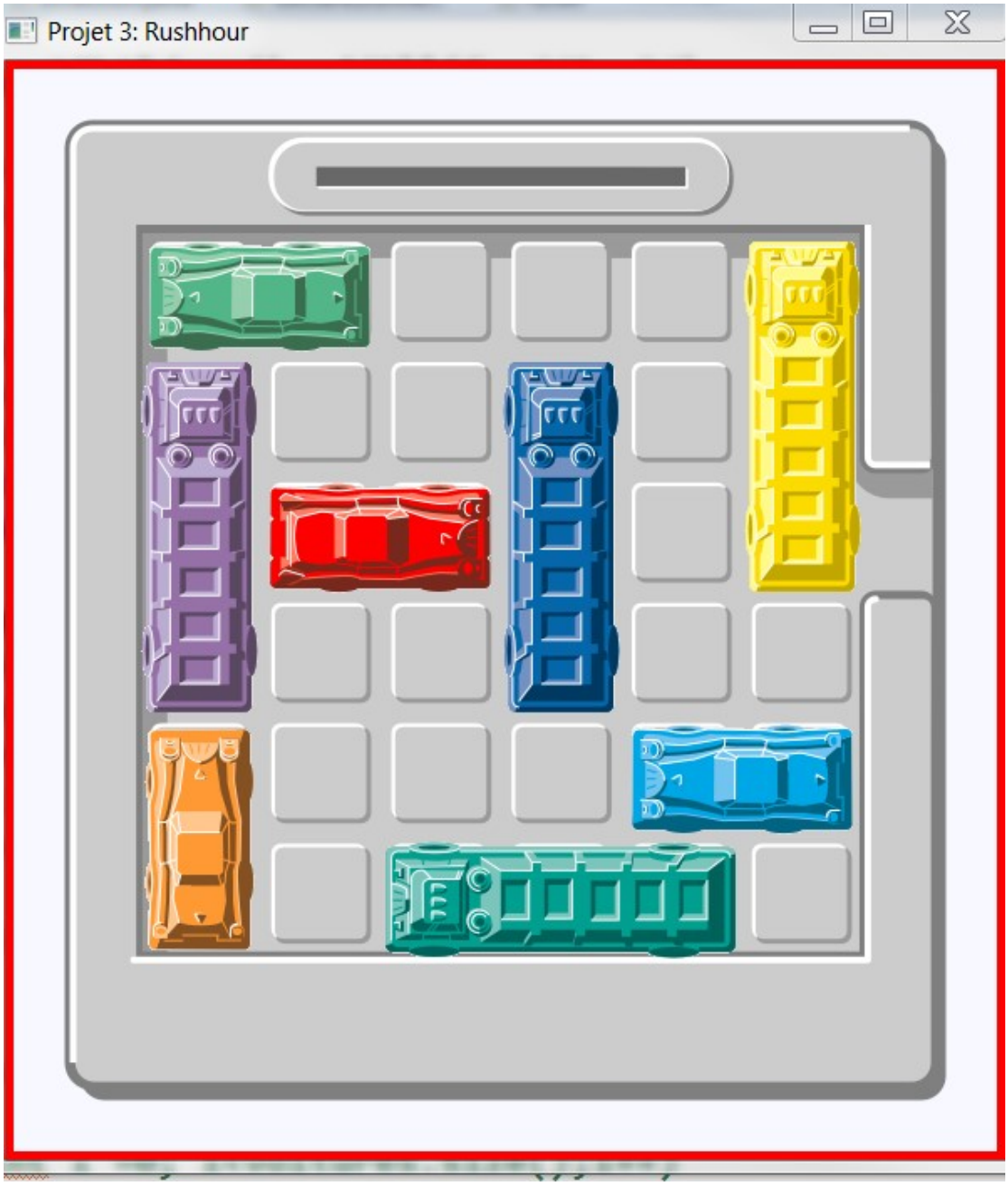
Consignes particulières

- Un projet clairement INCOMPLET ou un projet qui ne compile pas entraîne automatiquement la note 0. Il est donc préférable de remettre un projet en retard qu'un projet incomplet.
- Une pénalité de 10% s'applique pour chaque jour de retard (max. 40%).
- La validité est évaluée surtout en utilisant le programme. Vous n'obtenez donc aucun point pour le code non fonctionnel. De plus, si un bogue ou un code non fonctionnel empêche le professeur d'utiliser d'autres parties du programme, vous ne pourrez pas obtenir de points pour ces parties.

Énoncé du problème

Il s'agit de programmer le jeu *Rush hour*. Ce jeu simule une congestion automobile dans un stationnement à une heure de pointe. Le but est de diriger une voiture particulière (la rouge dans notre cas), bloquée par plusieurs autres véhicules, vers la sortie.

Voici un exemple de configuration initiale :


Projet 3: Rushhour

Règles du jeu

- Les véhicules sont disposés sur une grille de 6x6
- La direction des véhicules est soit horizontale, soit verticale
- Les véhicules disposés horizontalement peuvent être déplacés vers la gauche ou vers la droite.
- Les véhicules disposés verticalement peuvent être déplacés vers le haut ou vers le bas.
- Les véhicules ne peuvent être soulevés.

Données disponibles

Trois fichiers textes f1, f2 et f3 contenant chacun la description d'une configuration initiale**.** Une ligne du fichier contient les caractéristiques d'un véhicule: couleur, longueur, colonne, ligne, direction.

Voici le contenu du fichier *facile.txt* :

```
verte,2,0,0,H
mauve,3,0,1,V
rouge,2,1,2,H
bleu,3,3,1,V
jaune,3,5,0,V
orange,2,0,4,V
bleue,2,4,4,H
vert,3,2,5,H
```

où H désigne la direction horizontale et V la verticale.
La voiture verte occupe deux cases de la grille, elle se trouve à la colonne 0 et à la ligne 0 de la grille et est horizontale.
Notez que les coordonnées d'une voiture sont celles de sa case la plus à gauche pour une voiture horizontale et celle de sa case la plus haute pour une voiture verticale.

Travail à effectuer

1- Donnez le diagramme de classes de votre application
2- Lisez une configuration initiale à partir d'un fichier texte
3- Créez une interface graphique représentant cette configuration initiale
4- Programmez le jeu qui consiste à effectuer une série de déplacements des véhicules pour que la voiture rouge arrive à la sortie de la grille. Vous devez, à tout moment, afficher le nombre de coups (déplacements) effectués ainsi que le temps écoulé depuis de le début de la partie. Il doit toujours être possible de recommencer, c'est-à-dire remettre la grille à son état initial. Affichez un message de félicitations quand la partie est gagnée.

Important :

- Votre application doit contenir au moins **deux ensembles de classes** : un pour les classes de données et un pour les classes de l'interface graphique.
- Utilisez le type ***Enum*** pour la direction.
- Votre application doit permettre le jeu des trois cas : facile, moyen et difficile pour les données qui vous sont proposées (fichiers facile.txt, moyen.txt et difficile.txt). Le retour au menu doit être toujours possible.
- Affichez le temps dans l'interface graphique à l'aide d'un chronomètre. Vous devez le programmer en utilisant un objet Thread.
- Remettez une documentation javadoc de votre projet.
- Remettez le diagramme de classes de votre application.