Module 1 –Informatique – Algorithmique et Programmation Objet

Travaux Pratiques (16), Licence 1ère Année

Jeu de Damier

Nous souhaitons programmer le jeu de damier suivant :

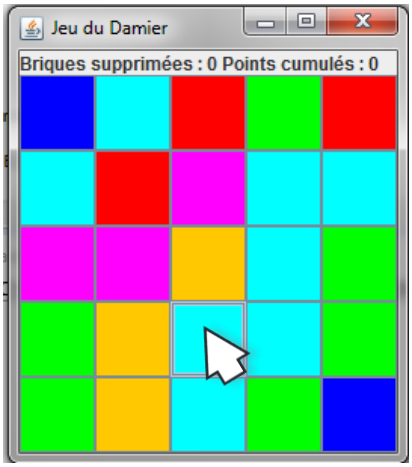
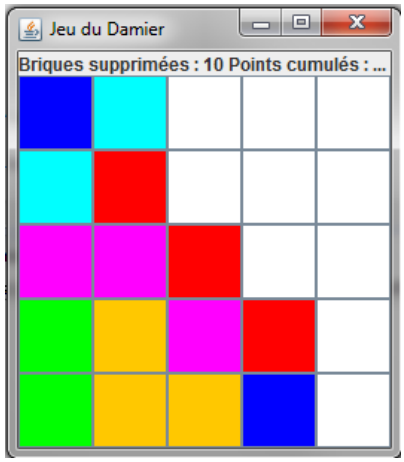
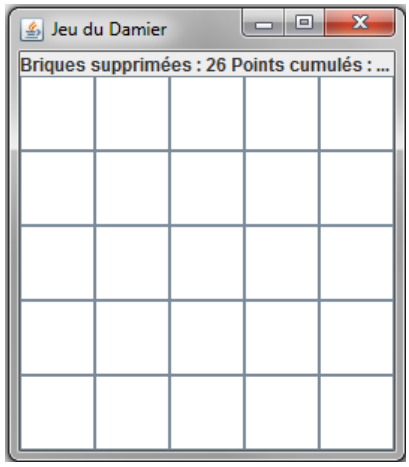


Fig 1 : Configuration initiale



Configuration intermédiaire



Configuration finale

Un damier rempli aléatoirement de briques de différentes couleurs est proposé à l'utilisateur (**Fig 1** ci-dessus).

L'utilisateur choisit une brique du damier (voir curseur **Fig 1** ci-dessus). La brique choisie ainsi que toutes celles adjacentes (verticalement et horizontalement) de même couleur sont supprimées. La suppression se répercute également aux briques adjacentes à ces dernières (voir **Fig 2** ci-contre).

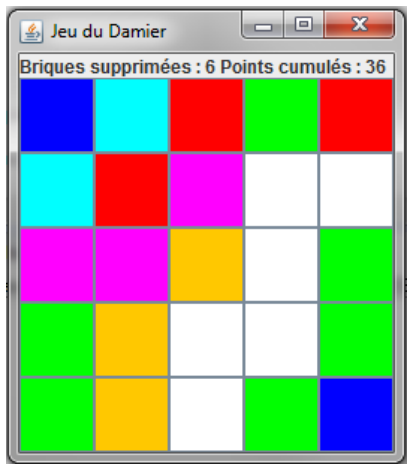


Fig 2 : Suppression briques

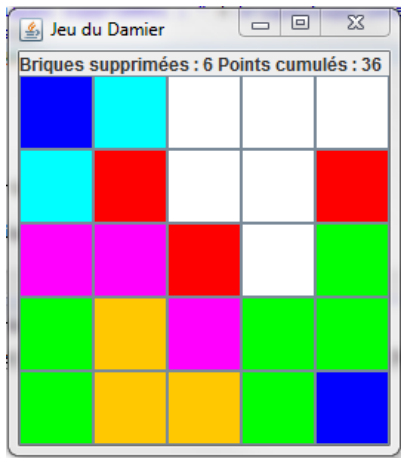


Fig 3 : Descente des briques

Les briques restantes "tombent" pour occuper les cases libérées par les briques supprimées (**Fig 3** ci-contre).

Si une colonne est vide (**Fig 4** ci-contre), celles qui ne le sont pas sont décalées vers la gauche de façon à les regrouper (**Fig 5** ci-contre).

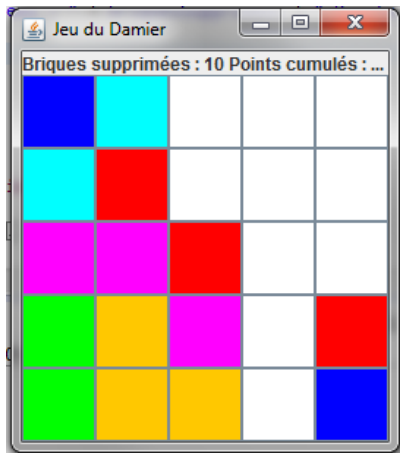


Fig 4 : Colonne 4 vide

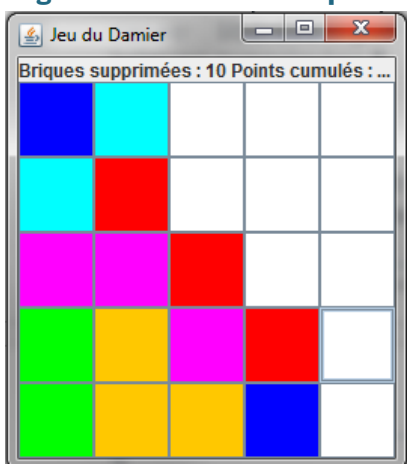


Fig 5 : Décalage des colonnes non vides vers la gauche

L'utilisateur choisit successivement une nouvelle brique jusqu'à leur suppression totale.

Le but du jeu est de supprimer la totalité des briques du damier en un nombre minimum de coups (choix de l'utilisateur). Pour le calcul des points, on peut définir que : lorsqu'en un coup **n** briques sont supprimées, le joueur accumule **nxn** points. La taille du damier doit être quelconque.

Vous pouvez programmer ce jeu par vous-même ou suivre les indications suivantes :

Classe **Brique** : extension de la classe **JButton** comprenant :
- Les attributs **couleur**, **ligne**, **colonne** (dans le damier) et **active** (booléen précisant si la brique a déjà été sélectionnée ou pas).
- Un constructeur acceptant comme paramètre les coordonnées (ligne et colonne) de la brique, choisit aléatoirement sa couleur, la rend active et indique sa taille,
- Des setters et getters pour tous ses attributs,
- Une méthode **Color choixCouleur()** qui retourne une couleur (Color) choisie aléatoirement,
- Une méthode **void cliquee()** pour désactiver (active=false) une brique et changer sa couleur en blanc.

Classe **Damier** : extension de la classe **JFrame** et implémentant l'interface **ActionListener** comprenant :
- Les attributs **briquesSupprimees** (pour détecter la fin du jeu), **points** (cumulés), le **damier** (**JPanel**) et une matrice carrée **briques** d'objets **Brique**.
- Un constructeur qui crée la matrice **briques** et initialise chaque élément par un **objet Brique**, ajoute chacun de ces derniers dans le **damier** (attribut **JPanel damier**). **Note** : quand on crée un **objet Brique**, on lui passe en paramètres ses coordonnées (**ligne** et **colonne** dans la matrice **briques**).
- Une méthode récursive **void supprimeBrique(Brique[][] b, int l, int c)** qui supprime (désactive) la brique à la ligne **l** et à la colonne **c** de la matrice **b** ainsi que toutes celles adjacentes (verticalement et horizontalement) de même couleur. La suppression se répercute également aux briques adjacentes à ces dernières,
- Une méthode **void tomberBrique(Brique[][] b)** qui fait déplacer, dans chacune des colonnes, les briques vers le bas de façon qu'il n'y ait pas de brique supprimée (désactive) entre deux briques actives,
- Une méthode **boolean colonneVide(Brique[][] b, int c)** qui revoie true si la colonne **c** de la matrice **b** est vide (toutes les briques sont désactivées) sinon false,
- Une méthode **void copierColonne(Brique[][] b, int i, int j)** qui recopie la colonne **j** dans la colonne **i** de la matrice **b**,
- Une méthode **void regrouperColonnes(Brique[][] b)** qui regroupe sur la gauche les colonnes non vides de la matrice **b**.
- Une méthode **void actionPerformed(ActionEvent e)** qui se déclenche chaque fois qu'une brique est cliquée, et qui réalise le jeu.

On pourra choisir une stratégie de placement **BorderLayout** pour l'objet **Damier** (mettre le score au nord et le **JPanel damier** au centre) et **GridLayout** pour le **JPanel damier**.