Université de Versailles – Saint-Quentin-en-Yvelines
IATIC 3 ISTY – Projet 2021-2022
Encadrante : Laurence Pilard
laurence.pilard@uvsq.fr

Jeu 2048

Ce projet DOIT être réalisé en binôme.

1 Les règles du jeu[†]

Le but du jeu est de *fusionner* des nombres ensemble (puissances de 2) afin d'atteindre le nombre ultime *2048* et gagner la partie !

L'aire du jeu 2048 est une grille de 8 lignes par 8 colonnes avec donc 64 cellules carrées (habituellement, le jeu est de dimension 4×4, mais il vous est demandé dans ce projet d'implémenter un jeu de dimension 8×8). Chaque cellule peut être vide ou contenir un nombre. Au début du jeu, il y a deux carrés (également appelés *briques*) avec un chiffre 2 à l'intérieur, tout le reste est vide.

Lorsque vous parvenez à faire entrer en collision 2 briques avec le même numéro dedans, elles *fusionnent* en une seule nouvelle brique dont le numéro sera l'addition des deux nombres précédents : 2+2=4, 4+4=8,..., 1024+1024=2048 !

Pour déplacer les briques sur la grille, le joueur doit juste choisir une direction (haut, droite, bas ou gauche). Toutes les briques vont se déplacer dans la direction choisie, jusqu'à ce qu'elles fusionnent avec une brique de même valeur ou bien qu'elles soient bloquées par une brique avec un numéro différent. La direction choisie est indiquée par le joueur en appuyant sur des boutons de directions affichés dans l'interface. Le jeu se joue donc avec la souris.

Pour une version en ligne du jeu, se référer au site web suivant :

`http://gabrielecirulli.github.io/2048/`

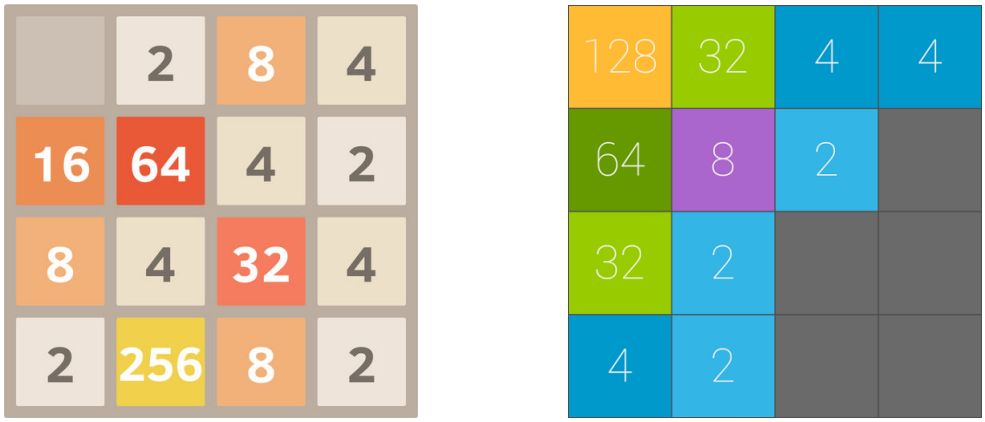
Si vous ne connaissez pas ce jeu, il vous est vivement conseillé d'y jouer plusieurs parties avant de commencer à coder.

2 L'implémentation

- Vous devez implémenter le jeu présenté ci-dessus. **Aucun changement de règle n'est accepté**.
- **Une seule variables globale est autorisée** : celle qui est de type tableau à 2 dimensions et qui contient le plateau de jeu.

[†] `http://games.tooliphone.net/fr/game/2048`

- Vous devez implémenter votre jeu dans le langage C et utiliser l'interface graphique basée sur *sdl* et utilisant la bibliothèque *graphics.c* fournie en cours.



Interfaces graphiques : Il vous est demandé d'implémenter deux interfaces graphiques. Ces interfaces sont présentées dans la figure ci-dessus. Vous devez conserver l'épaisseur des traits entre les cases (traits épais dans le jeu de gauche et fins dans celui de droite) et vous devez choisir des couleurs différentes pour chaque jeu. ATTENTION, les images montrent deux plateaux de jeu de dimension 4×4, mais vous devez implémenter un jeu de dimension 8×8.

Modèle de programmation MVC : Dans la mesure où votre jeu doit fonctionner sous deux interfaces graphiques différentes, vous devez utiliser la méthode MVC de programmation. Dans cette technique, votre code est divisé en trois parties :

- Le *Modèle*, qui contient toutes les fonctions relatives à la partie logique du jeu (`init_plateau`, `deplace_brique, applique_fusion, calcul_score`,...).
- La *Vue*, qui contient toutes les fonction relatives à l'affichage du jeu (`affiche_plateau`, `affiche_score, affiche_menu_accueil`,...).
- Le *Contrôleur*, qui contient toutes les fonctions faisant le lien entre le modèle et la vue. (`main`,...)

Par exemple, si on s'intéresse à la fonctionnalité qui gère les points du joueur. Dans le *modèle*, on trouvera la fonction qui calcule le nombre de points actuels du joueur, en fonction du contenu du plateau de jeu (le tableau à deux dimensions) et qui renvoie donc un entier. Dans la *vue*, on trouvera la fonction qui affiche le nombre de points actuels du joueur. Enfin, c'est au niveau du *contrôleur*, qu'on appellera ces deux fonctions.
Il ne vous est pas demandé de faire un fichier par partie. En revanche, vous devez rendre un fichier `.c` dans lequel vous avez trié vos fonctions. Ainsi, vous devez avoir un code source qui contient une première partie intitulée *La vue* (ce titre est indiqué en commentaire bien sûre) et qui regroupe toutes les fonctions relatives à la vue. De même, vous devez avoir une partie pour le *modèle* et une partie pour le *contrôleur*.

Type de fusion : Il vous est demandé d'implémenter deux types de fusions : la fusion *classique* et la fusion *totale*. Supposons que le joueur indique la direction *droite* alors que plateau de jeu contient la ligne suivante :

2	8	vide	8	vide	16	16	16

Dans le cas de la fusion classique, chaque brique fusionne avec *au plus une* autre brique. On obtient alors:

vide	vide	vide	vide	2	16	16	32

Dans le cas de la fusion totale, les briques fusionnent entre elles *tant que c'est possible*. On obtient alors:

vide	vide	vide	vide	vide	vide	2	64

Le choix du type de fusion doit être fait par le joueur en début de partie et ne peut pas être modifié en cours de partie.

Ajout de numéros sur le plateau de jeu : Après chaque déplacement des briques dans une direction définie par le joueur :

- Si au moins une brique est déplacée ou si au moins une fusion est effectuée, alors une nouvelle brique contenant le numéro 2 apparaît.
- Sinon, aucune nouvelle brique n'apparaît.

La position d'apparition de la brique doit se faire selon le niveau de difficulté choisi par le joueur. En mode *facile*, cette position est tirée aléatoirement parmi toutes les cases libres du plateau de jeu. En mode *difficile*, la position la moins avantageuse pour le joueur est choisie.
Le niveau de difficulté doit être choisi par le joueur en début de partie et ne peut pas être modifié en cours de partie.

Aide au prochain coup : Le joueur a la possibilité d'activer l'option *d'aide au prochain coup*. Cette aide permet d'obtenir un conseil sur le meilleur prochain coup à jouer. Elle peut être activée ou désactivée *à tout moment* de la partie (en cliquant sur un bouton par exemple).
Lorsque l'aide est activée, avant chaque coup du joueur, l'interface affiche la direction conseillée. Puis, le joueur peut suivre le conseil ou non. Attention, à partir du moment où le joueur a activé cette aide, celle-ci reste active jusqu'à ce que le joueur la désactive.

Score : Le score du joueur est la somme de tous les numéros présents sur le plateau de jeu. Enfin, à chaque coup au cours duquel l'aide est activée, le score diminue de 1. Il vous est demandé d'afficher le score à tout moment.

Début et fin de parte : A chaque nouvelle partie, le programme propose au joueur de choisir son interface graphique. Il propose également de choisir son mode de jeu (facile ou difficile), ainsi que le type de fusion (classique ou totale). Une fois la partie terminée, le jeu doit afficher si la partie est gagnée ou perdue ainsi que le score, puis proposer une nouvelle partie aux joueurs. Pour cette nouvelle partie, le programme devra à nouveau laisser le choix au joueur de l'interface graphique, du mode de jeu et du type de fusion.
La partie est gagnée si la somme des numéros sur le plateau de jeu est 2048 ou plus, elle est perdue si plus aucune case n'est libre sur le plateau et que plus aucune fusion n'est possible.

3 Ce qu'il faut rendre

Date importante. Le programme devra être rendu pour le **mardi 5 octobre, 8H** au plus tard. Une pénalité de 5 points par jour de retard sera appliquée.

Comment rendre le projet ? Le projet devra être envoyé par mail à l'adresse suivante : `laurence.pilard@uvsq.fr`

Que faut-il rendre ? Il est demandé de rendre **le code source** de votre programme. (votre fichier .c)

Sous quelle forme doit se présenter votre programme ? Votre programme devra se compiler **sans erreur ni warning** dans les salles machines de l'ISTY. Il est *impératif* de mettre, en début de chaque fichier, un commentaire rappelant vos nom et prénom.

Triche : Si deux projets sont similaires dans leur réalisation, la note minimale entre ces deux projets sera prise, puis divisée par deux, puis donnée comme note aux deux projets. Généralisable à n projets similaires.

4 Notation

1. Le code : Une attention particulière sera portée sur votre code (environ 50% de la note) :
   - respect du modèle MVC
   - usage pertinent des fonctions;
   - noms de variables et de fonctions explicites;
   - code bien indenté ;
   - bonne lisibilité du code en général.
2. L'exécution : environ 50% de la note. Concernant principalement le respect des règles, la jouabilité, l'ergonomie du jeu et l'absence de bug.