Cégep de Saint-Hyacinthe
Département d'informatique

Programmation orientée objet (420-2DP-HY)

TP1 – Jeu de Taquin

Préparé par
Martin Lalancette
bureau D 2330A

DESCRIPTION

But : Conception d'un programme en C# console avec l'utilisation d'une classe et comportant l'utilisation des notions vues la session précédente.

Objectifs :

- Créer une classe
- Créer une application de type console
- Générer un diagramme de classe
- Réviser les notions de tableau à deux dimensions
- Réviser les structures de décision simples et/ou multiples
- Réviser les structures de répétition
- Pratiquer les notions de classe de base (classe, champs, méthodes, propriétés).
- Utiliser les constantes
- **Isoler la logique du jeu de l'affichage (Encapsulation)**
- Respecter les normes de programmation
- Utilisation d'un gestionnaire de versions (DevOps – Azure Repos – GIT)

Durée : 6 h

Pondération : 10 pts (**en équipe de deux obligatoire**)

Remise : À la fin de la semaine 5.

Contenu général :

- Remettre vos documents d'analyse et votre solution contenant le projet dans un document .ZIP via LÉA.

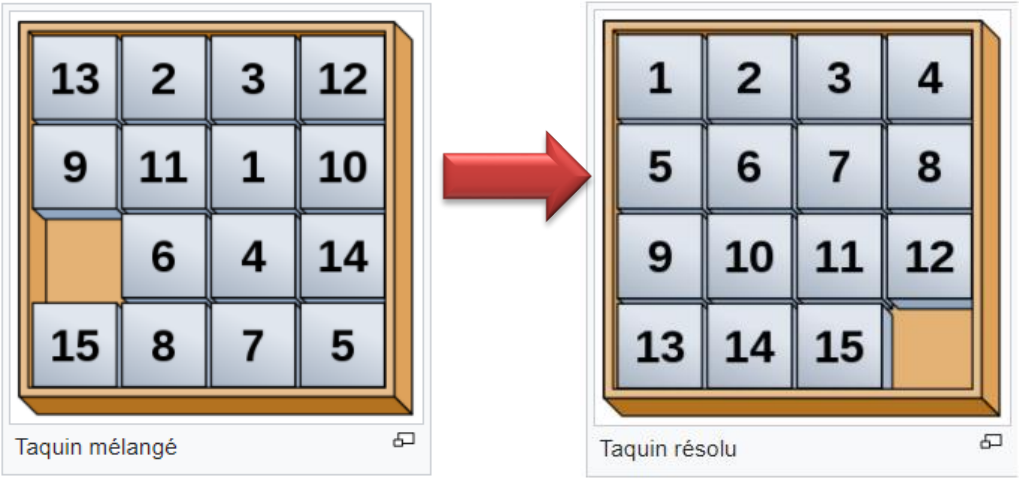
Notes

- Conserver une copie de sécurité. Il est de **votre responsabilité** de conserver une copie de sécurité dans l'éventualité où la lecture des données serait impossible. Cette copie doit **être disponible sur demande**.

Spécifications du travail

Ce travail pratique est à faire en **équipe de deux personnes obligatoires**.

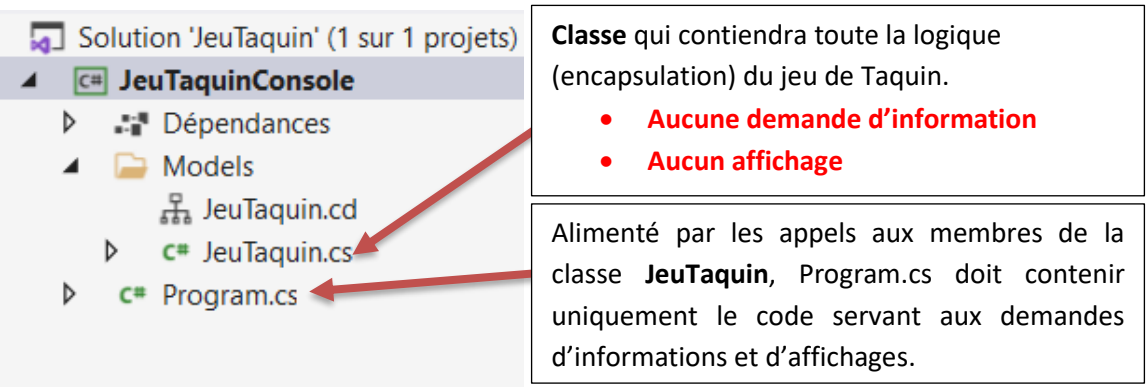
Jeu de Taquin : Le jeu consiste à replacer dans l'ordre des carreaux numérotés qui ont été préalablement mélangés. La dimension de la grille peut varier. Exemple de grille 4 x 4:



(Wikipédia - L'encyclopédie libre, 2022)

Le joueur doit déplacer une case numérotée vers la case vide pour tenter de résoudre la grille en plaçant les chiffres en ordre croissant. Les chiffres doivent s'afficher en ordre numérique: 1, 2, 3, … Le but est de résoudre la grille en effectuant le moins de déplacements.

Vous devez créer une solution qui contiendra un projet : **Application console (.NET Core).** Votre projet devra être **structuré selon MVVM**, c'est-à-dire comme suit :



Gestion de versions (DevOps – Azure Repos GIT)

Vous devez former une équipe de deux personnes et en informer votre enseignant via MIO. Dans LÉA, il y aura un fichier EXCEL contenant le lien DevOPS – GIT de votre équipe que vous devrez utiliser pour effectuer les transactions GIT.

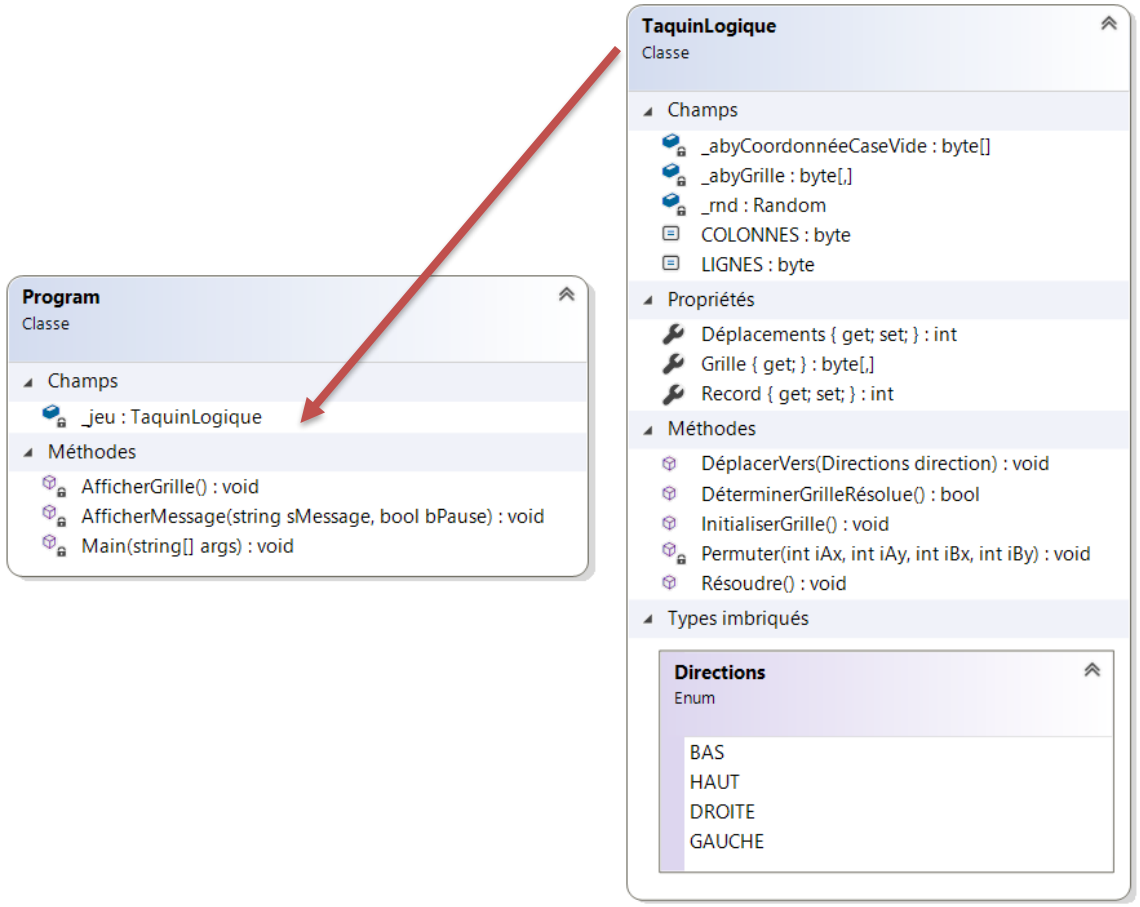
Consignes :

1. Durant la conception de ce projet, chaque membre de l'équipe devra effectuer au minimum 5 modifications et les pousser (push) dans le référentiel central. À chaque modification poussée, vous devez écrire un commentaire significatif.

2. Créer la solution **JeuTaquin** et le projet **JeuTaquinConsole** de type Console (.NET Core) et le structurer selon **MVVM**. Faire un premier dépôt initial. S'assurer que chaque membre peut récupérer ce projet et opérer avec GIT.

Programmation de la classe JeuTaquin et de la console (.NET Core)

Cette partie consiste à créer une classe nommée **JeuTaquin** qui servira à **ENCAPSULER les logiques** associées au jeu de Taquin. Voici un diagramme contenant les membres avec leur signature que vous devez programmer:



Propriétés :

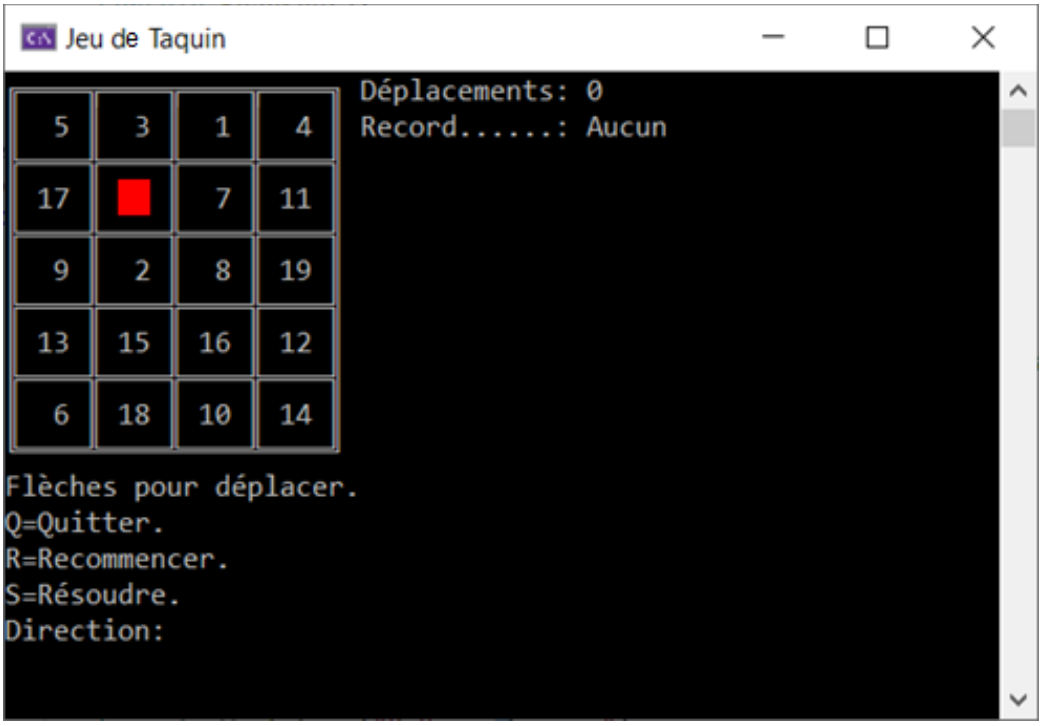
- **Déplacements** : Permet de connaitre le nombre de fois qu'un déplacement a été réalisé.
- **Grille** : Permet de donner accès au contenu de la grille interne (gérée à l'interne). Va servir à alimenter l'affichage.
- **Record** : Permet de garder en mémoire le plus petit nombre de déplacements effectués pour résoudre une grille.

Méthodes :

- **DéplacerVers** : Cette méthode doit prendre la direction en paramètre et déplacer le chiffre vers la case vide si autorisé. Cette méthode sera appelée dans Program.cs lorsque l'utilisateur appuiera sur une flèche.

- **DéterminerGrilleRésolue**: Cette méthode permet de valider si tous les chiffres de la grille sont à la bonne position. Elle permet d'indiquer si le joueur est gagnant ou non.
- **InitialiserGrille:** Cette méthode permet de préparer la grille en mélangeant les chiffres de **façon SOLUBLE** ➔ Conseil: Remplir le tableau dans l'ordre des chiffres et simuler entre 400 ou 500 déplacements au hasard (relié à la case vide). Cette méthode devra être appelée juste avant de commencer une partie ou bien lorsque le joueur souhaite recommencer.
- **Permuter:** Cette méthode privée permet d'échanger la valeur contenue dans une case vers une autre.
- **Résoudre** : Cette méthode consiste à placer dans le bon ordre les chiffres de la grille. Elle est utile pour valider le message de victoire et elle sera utilisée dans le TP2.

Vous devez programmer les algorithmes de ses membres. **Les logiques du jeu doivent se concentrer dans cette classe**. Par la suite, vous devez ajouter les appels dans **Programs.cs** afin de faire fonctionner le jeu. Voici l'interface à programmer :



Lien vidéo : https://web.microsoftstream.com/video/1b5e9873-7d7a-4fce-aa7f-456840abcc70

Consignes :

- Déclarer un objet **JeuTaquin** dans Program.cs et l'utiliser (Faire appel à ses membres).
- Utiliser les flèches pour déplacer un chiffre près de la case vide (rouge). Indices: Utiliser **ReadKey**(), la propriété **Key** et l'énumération ConsoleKey (à consulter).
- Programmer les opérations: Quitter, Recommencer et résoudre.

Barème d'évaluation

Gestion de versions (Préparation, 5 modifications minimums par développeur, etc.) ..... /1.0
Classe JeuTaquin : Déclaration des membres selon l'analyse (signatures)..... /2.0
Classe JeuTaquin : Programmation de la logique de la méthode **InitialiserGrille** ..... /1.0
Classe JeuTaquin : Programmation de la logique de la méthode **DéplacerVers** ..... /1.0
Classe JeuTaquin : Programmation de la logique de la méthode **Permuter**..... /0.5
Classe JeuTaquin : Programmation de la logique de la méthode **DéterminerGrilleRésolue** ..... /1.0
Classe JeuTaquin : Programmation de la logique de la méthode **Résoudre** ..... /1.0
Console : Affichage de la grille avec les chiffres et la case vide de couleur..... /1.0
Console : Programmation des opérations demandées (flèches, quitter, recommencer, etc.)... /1.0
Console : Calcul et affichage des statistiques demandées..... /0.5
Note totale* ..... /10.0

* Tout travail plagié en partie ou en totalité se verra attribuer une note totale de 0 %.

PDEA #1: Lors d'activités d'évaluation sommative en classe ou hors classe (documentation, rapport de laboratoire, rapport de stage, examen), une **pénalité maximale de 10 %** peut être retranchée de la note finale de ladite évaluation (le barème étant de **0,5%/erreur incluant les fautes répétitives**). Pour les commentaires dans les programmes informatiques, le barème est de **0,5% par faute**. Pour les interfaces utilisateur, le barème est de **1% par faute**.

PDEA #4 : Toute évaluation sommative remise après la date d'échéance fixée se voit attribuer la note zéro pour les étudiants **de 2e à 6e session**. Afin de faciliter l'accueil et l'intégration des nouveaux étudiants de **1re session**, cette règle s'appliquera de la façon suivante : 30 % de pénalité pour une 1re offense (à condition que la remise soit faite dans les 24 heures suivant la date de remise officielle), 100% de pénalité pour les offenses subséquentes.

Références

Wikipédia - L'encyclopédie libre. (2022, 01 17). *Taquin*. Récupéré sur Wikipédia - L'encyclopédie libre: https://fr.wikipedia.org/wiki/Taquin