Activité intégrative 2024

Itération 3

Durée : 10 heures

L'objectif de cette itération est de terminer une partie et de consulter des statistiques.

Planification de l'itération 3

1 : Découvrir l'énoncé (30 min)

En grand groupe, le responsable présente l'énoncé.

2 : Concevoir l'itération 3 (60 min et plus)

Comme son nom l'indique, la section sur la conception liste plusieurs responsabilités et classes candidates. Nous te proposons de réfléchir à la conception de l'application en trois étapes :

1. **Individuellement**, attribue les responsabilités à des candidats. En parallèle, identifie des responsabilités et des candidats manquants. [15 min]
2. **En petits groupes (3 à 5 étudiants)**, compare tes attributions ainsi que tes nouvelles responsabilités et classes candidates. Décline chaque responsabilité et collaborateur en méthodes et attributs. [20 min]
3. **En grand groupe**, deux petits groupes présentent les fruits de leurs réflexions aux autres groupes. Le but est d'évaluer les avantages et inconvénients des solutions proposées. [25 min]

3 : Implémenter et tester l'application (8 h 30)

Nous te recommandons d'implémenter progressivement l'application, une US après l'autre. **Attention,** ton implémentation doit réussir au moins 10 des 12 tests d'acceptation pour que l'itération soit jugée recevable.

User Stories

Cette troisième itération vise à détecter la fin d'une partie et à afficher des statistiques.

Note

Les TA ci-dessous représentent un échantillon des cas possibles. Nous nous réservons le droit de dévier de ces exemples.

AI-7 Finir une partie

En tant que joueur, je souhaite terminer une partie, afin de connaître mes résultats.

Détails

- Une partie se termine dès qu'un joueur relie deux de ses bords en formant un chemin composé uniquement de tuiles de sa couleur.
- Détecter un chemin entre deux bords revient à explorer les coordonnées des tuiles occupées par le joueur qui a la main. **L'algorithme à implémenter** est présenté à cette fin en annexe.
- Quand une partie se termine, l'application navigue vers l'écran de fin de partie.

Tests d'acceptation

- Soit la situation de jeu ci-dessous, quand le joueur rouge occupe la tuile (3 ; -3), hachurée sur l'image, alors l'application détecte une fin de partie et navigue vers l'écran de fin de partie.

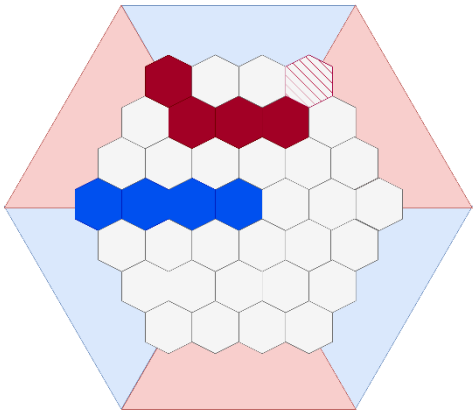

Figure 1 TA-7.1 avant occupation de la case

- Soit la situation de jeu ci-dessous, quand le joueur bleu occupe la tuile (1 ; 2), hachurée sur l'image, alors l'application détecte une fin de partie et navigue vers l'écran de fin de partie.

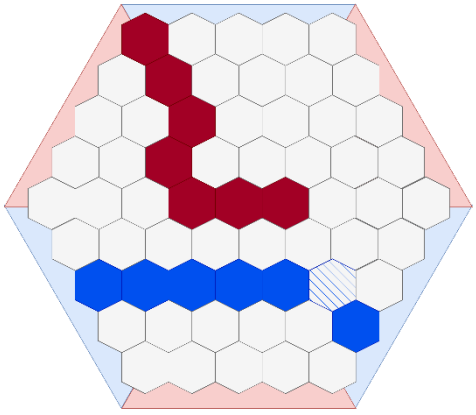

Figure 2 TA-7.2 avant occupation de la case

- Soit la situation de jeu ci-dessous, quand le joueur bleu occupe la tuile (2 ; -2), alors l'application détecte une fin de partie et navigue vers l'écran de fin de partie.

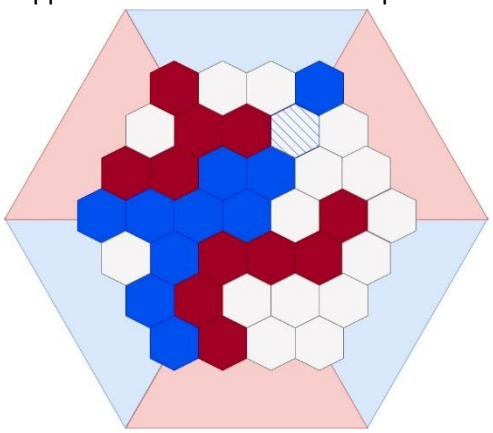

Figure 3 TA-7.3 avant occupation de la case

- Soit la situation de jeu ci-dessous, quand le joueur rouge occupe la case (0 ; -2), alors l'application détecte une fin de partie et navigue vers l'écran de fin de partie.

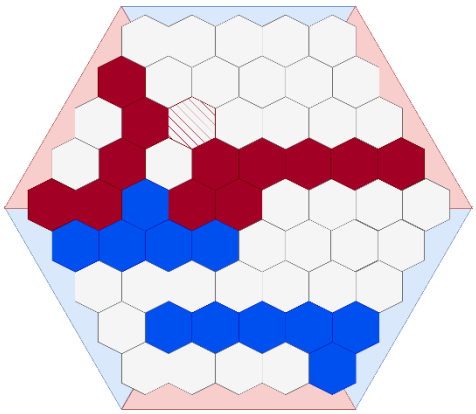

Figure 4 TA-7.4 avant occupation de la case

AI-8 Afficher les résultats d'une partie

En tant que joueur, je veux connaître les résultats de ma dernière partie, afin de savoir si j'ai bien joué.

Détails

- Les résultats d'une partie sont affichés par l'écran de fin de partie.
- Certains résultats seront utilisés pour actualiser les statistiques sur l'ensemble des parties (voir AI-9).
- Le premier résultat à afficher est le nom du vainqueur et sa couleur.
- Le second résultat est le score du vainqueur. Soient T_{min}, le nombre minimum de tuiles à occuper pour gagner et $T_{vainqueur}$, le nombre de tuiles qui composent le chemin du vainqueur. Le score du vainqueur vaut $\max(1 ; T_{min} - (T_{vainqueur} - T_{min}))$.
- Le troisième résultat est le score du perdant. Le score du perdant vaut $\max(0 ; (T_{min} - (T_{vainqueur} - T_{min})) / 2)$.
- Un score admet des demi-points.
- Le dernier résultat est le taux de remplissage du plateau qui est le rapport entre le nombre de tuiles du plateau et le nombre de tuiles occupées. Ce taux sera arrondi et affiché en pourcents (42% au lieu de 41,51%, par exemple).

Tests d'acceptation

Le tableau ci-dessous résume les résultats attendus pour les parties joués lors des tests d'acceptation de l'US 7.

Important

Le tableau part du principe qu'il n'y a pas de « swapping ». Toutefois, nous pouvons modifier cette hypothèse pendant la correction. Dans ce cas, le nom du joueur change, pas sa couleur.

Test d'acceptation	Vainqueur	Score du vainqueur	Score du perdant	Taux de remplissage
TA-7.1	P1 (rouge)	5 – (5 – 5) = 5.0	5/2 = 2.5	09/37 = 0.243…= 24%
TA-7.2	P2 (bleu)	6 – (7 – 6) = 5.0	5/2 = 2.5	14/61 = 0.229…= 23%
TA-7.3	P2 (bleu)	5 – (7 – 5) = 3.0	4/2 = 1.5	22/37 = 0.5945…= 59%
TA-7.4	P1 (rouge)	6 – (8 – 6) = 4.0	5/2 = 2	24/61 = 0.393…= 39%

AI-9 Afficher les statistiques sur l'ensemble des parties

En tant que joueur, je souhaite connaître les statistiques sur mes parties, afin d'en apprendre plus sur elles.

Détails

- Ces statistiques seront également affichées par l'écran de fin de partie.
- Les statistiques sont calculées à partir des résultats des parties (voir récit précédent).
- Si une partie est abandonnée, les statistiques ne sont pas actualisées.
- Les deux premières statistiques sont les scores totaux des joueurs. Ces totaux correspondent à la somme des scores respectifs des joueurs.
- La troisième statistique est le taux de recouvrement moyen d'une partie. Il correspond à la moyenne arithmétique des taux de recouvrement des parties. Le taux de recouvrement moyen sera formaté comme le taux de recouvrement d'une partie.

Tests d'acceptation

On suppose qu'on joue les tests d'acceptation dans l'ordre. Le tableau suivant présente les valeurs attendues après chaque partie.

Test d'acceptation	Score de P1	Score de P2	Taux remplissage moyen
TA-7.1	5.0	2.5	24%
TA-7.2	5 + 2.5 = 7.5	2.5 + 5 = 7.5	(24% + 23%) / 2 = 24%
TA-7.3	5 + 2.5 + 1.5 = 9	2.5 + 5 + 3 = 10.5	(24% + 23% + 59%) / 3 = 35%
TA-7.4	5 + 2.5 + 1.5 + 4 = 13	2.5 + 5 + 3 + 2 = 12.5	(24% + 23% + 59% + 39%) / 4 = 36%

Concevoir la solution de l'itération 3

Nous vous proposons une liste de responsabilités à attribuer. Libre à vous d'ajouter d'imaginer de nouvelles classes candidates. Ces nouveaux éléments sont à intégrer aux cartes CRC précédentes.

Responsabilités

- Détecte un chemin entre deux bords
- Calcule le score du vainqueur
- Calcule le score du perdant
- Calcule le taux de recouvrement
- Connait son score total
- Calcule le taux de remplissage moyen

Consignes et contraintes

Questions d'algorithmique

It-3-q1. Précondition et postcondition de l'algorithme de détection de chemin

Quelle(s) précondition(s) les entrées de l'algorithme de détection de chemin doivent-elles respecter ? Quelle(s) postcondition(s) la sortie doit-elle respecter ? Attention, votre postcondition doit lier les entrées aux sorties.

Répondez à cette question dans l'en-tête de la méthode qui réalise cette opération. Indiquez clairement, dans l'en-tête de la classe GameOverSupervisor où cette méthode est implémentée.

It-3-q2. CTT de l'algorithme de détection de chemins

Quelle est la CTT de l'algorithme de détection de chemin ? Construisez votre raisonnement à l'aide des structures de contrôles et des méthodes appelées, en particulier celles des collections. Indiquez la signification des variables.

Répondez à cette question dans l'en-tête de la méthode qui réalise cette opération. Indiquez clairement, dans l'en-tête de la classe GameOverSupervisor où cette méthode est implémentée.

It-3-q3. Choix du type et de l'implémentation d'une collection pour mémoriser les cases à visiter

Le document en annexe qui présente l'algorithme de détection de chemins mentionne une collection de cases ou tuiles à visiterappelée « voisinsAVisiter ».

Quelle interface de collection (List, Queue, Deque, (Sorted)Set, (Sorted)Map, etc.) allez-vous choisir pour représenter les cases à visiter ? Justifiez votre choix en identifiant les principales opérations dont vous aurez besoin pour implémenter l'algorithme ? Avez-vous notamment besoin d'accéder à un élément précis ? Si oui, sur base de quelle sorte de clé ?

Quelle implémentation de collection (ArrayList, LinkedList, ArrayDeque, HashSet, TreeMap, etc.) allez-vous choisir pour ces cases ? Justifiez votre choix en déterminant les CTT des principales opérations que vous utiliserez dans votre algorithme.

Répondez à ces questions dans la Javadoc de la classe dans lequel se trouve le calcul de CTT demandé au point précédent.

Programmation Orientée Objet

Les consignes et les contraintes de l'itération 1 restent valables pour l'itération 3.