Université de Bourgogne
Année universitaire 2020-2021
Info 2A — programmation objet
Descriptif du projet

1. Description générale

L'objet de ce projet est de modéliser un jeu de plateau comportant un nombre pair de cases, où chaque case a une valeur masquée (un entier positif) et peut donner un indice (voir plus loin). Les joueurs doivent deviner les valeurs des cases.

Pour ce faire, à tour de rôle, chaque joueur désigne deux cases et le jeu lui restitue les deux indices, un indice par case. L'indice donné par une case ne porte pas uniquement sur sa propre valeur, mais sur les valeurs des deux cases désignées. L'intérêt du jeu est que les cases désignées peuvent donner des indices de natures différentes et que ces natures ne sont pas connues des joueurs.
Par exemple, supposons que la première case choisie ait pour valeur 3 et donne comme indice la somme des valeurs des deux cases (nature : *somme*), et la seconde case ait pour valeur 6 et pour indice associé le max des valeurs des deux cases (nature : *max*). Le jeu renvoie comme indice pour la première case la valeur 9, et pour la seconde la valeur 6. Par contre, si les indices des deux cases étaient de la même nature somme (respectivement max), les indices vaudraient tous deux 9 (respectivement 6).

Après avoir reçu les indices, le joueur peut tenter d'indiquer les valeurs des deux cases. Si sa réponse est juste pour les deux cases, celles-ci sont dévoilées (elle ne pourront plus être sélectionnées) et le score du joueur est crédité des valeurs de ces cases. Sinon, son score ne change pas et c'est au tour du joueur suivant.

Le jeu se termine quand toutes les cases ont été dévoilées. Le joueur gagnant est celui dont le score est le plus grand à la fin du jeu.

La richesse du jeu provient de la variété des natures d'indices disponibles (somme, produit, différence, max, min, valeur de l'autre case, valeur de la case,…). Au début du jeu, deux natures d'indices sont sélectionnées parmi les natures disponibles (la sélection peut être effectuée par les joueurs ou aléatoirement parmi les natures d'indices disponibles). Le plateau est alors généré avec, pour chaque case, une valeur et une des deux natures d'indices tirées aléatoirement.

2. Précisions

Joueurs

- ✓ Chaque joueur est doté d'un crédit initial de 0,
- ✓ Il peut y avoir un ou plusieurs joueurs. Dans une version simple du jeu, il peut n'y avoir qu'un seul joueur.

Cases

Au début du jeu, le nombre n (pair) de cases est choisi et un plateau rectangulaire comprenant ces cases est constitué. Pour l'affichage, il est préférable que ce rectangle soit le plus proche possible d'un carré et, donc que chaque dimension s'approche le plus possible de $\sqrt{n}$.
Chaque case constituant ce plateau comporte deux paramètres déterminés aléatoirement lors de la création de la case :

- ✓ Sa valeur, tirée entre 0 et une valeur maximale qui est un paramètre du jeu (par exemple 9 ; c'est plus simple en terme d'affichage).
- ✓ L'indice apporté par la case, qui est un type d'opération qui s'effectue sur la case qui le porte et une autre case. Cet indice est pris parmi les deux sélectionnés au début du jeu.

Tour du joueur

À son tour, le joueur effectue les actions suivantes :

- ✓ Sélection de deux cases en précisant leurs numéros de ligne et de colonne. Les deux valeurs des indices sont affichés.
- ✓ Proposition d'hypothèses sur les valeurs des deux cases.

Si le joueur a bien deviné les cases, celles-ci sont marquées visibles dans le plateau — ce qui permettra d'afficher le plateau en montrant désormais leur valeur et la nature de leur indice (Un caractère pourra symboliser chaque nature d'indice : *'+', '-', '>'*…).

3. Propositions et conseils de modélisation

Vous avez le choix de modéliser ce jeu de la manière que vous souhaitez dans le respect des principes énoncés ci-dessus. Voici cependant quelques éléments de réflexion que vous pouvez utiliser si vous avez des difficultés à aborder le problème.

Modélisation de l'« environnement »

En programmation objet, il est aisé de modéliser un objet isolé (cas d'une fraction par exemple), mais il est plus difficile de gérer les interactions entre plusieurs objets. Il y a donc souvent un « objet central » qui joue le rôle d'environnement pour les autres objets et permet ces interactions. Ici, cela peut être un jeu (ou partie), qui regroupe entre autres les joueurs avec le plateau de jeu et les différents paramètres (valeur max des cases, natures d'indices choisies, joueur courant…). On peut lui donner le rôle de piloter le déroulement du jeu vu précédemment, par exemple par une méthode `joue`.

Il faut éviter d'avoir des classes ou des méthodes volumineuses (par exemple la classe `Jeu` et sa méthode `joue`). Il est alors possible d'avoir aussi des classes pour des phases du jeu : tour d'un joueur, choix des cases, hypothèses sur les cases… avec leur propre méthode qui décrit l'action effectuée. La classe `Jeu` délèguerait alors une partie des actions à effectuer au tour de jeu, qui peut lui-même déléguer à d'autres classes une partie des actions qui le constituent.
Il faut aussi des classes périphériques pour gérer les joueurs — s'il y en a plusieurs — et le plateau (voir ci-dessous). Déchargez au maximum la classe `Jeu` de ce qui pourrait être pris en charge par une classe périphérique.

Représentation du plateau des indices, des cases

De même, il est bon d'avoir une classe `Plateau` qui doit contenir ses dimensions, la collection de ses cases, être capable de se remplir, de s'afficher, de déterminer si un indice de ligne ou de colonne est correct,…
Le plateau peut être vu comme un tableau à deux dimensions de cases.

Pour gérer les différents calculs d'indices, plusieurs solutions sont possibles, dont certaines qui utilisent des hiérarchies de classes (classe `Indice` et sous-classes pour les différents types d'indices, portant différents calculs : `IndiceSomme`, `IndiceProduit`…).
Quelle que soit la méthode de représentation employée, il doit être facile d'ajouter de nouvelles natures d'indices pour enrichir le jeu. Il faut que la modification du code soit minimale.

4. Modalités du travail à effectuer

Dans les conditions actuelles, le travail est prévu pour pouvoir être effectué seul, mais, si vous en avez l'opportunité, vous pouvez aussi le réaliser en binôme. Il doit être original, c'est-à-dire que vous pouvez échanger des idées entre monômes et/ou binômes, mais la réalisation finale présentée doit être spécifique et maîtrisée (vous devrez pouvoir répondre à des questions sur le fonctionnement et le code). De même, les rapports doivent être personnels.

4.1. Réalisation

Vous devez réaliser un programme en java qui modélise les éléments décrits par l'énoncé. Créez le programme progressivement en le testant systématiquement à chaque étape. Vous pouvez par exemple commencer par un programme comportant un seul joueur et deux indices, puis le complexifier si vous le pouvez.

Si TOUT ce qui est demandé fonctionne, vous pouvez ajouter d'autres fonctionnalités et des options ; par exemple, ajouter des joueurs, choisir au hasard celui qui débute le jeu, proposer des améliorations du jeu en terme d'affichage, de règles. Par exemple, on peut faire rejouer un joueur qui devine deux cases, tant qu'il n'échoue pas ou s'arrête de lui même, induire des pénalités quand un joueur échoue… Un joueur peut aussi reprendre les cases sélectionnées par le joueur précédent si celui-ci ne les avait pas découvertes. Il est aussi possible, en option, de proposer une version du jeu où les symboles correspondant aux indices des cases soient visibles sur le plateau (cela permet de conserver un jeu jouable en ayant plus de deux natures d'indices).
Il est aussi possible de proposer de rejouer le jeu en reprenant ou modifiant à la marge les caractéristiques du jeu précédent : liste de joueurs, les types d'indices, le nombre de cases…

Les classes devront comporter les constructeurs, accesseurs, modificateurs d'accès nécessaires ainsi, au besoin, que les méthodes `equals` et `toString` (le `toString` peut être intéressant notamment pour les affichages du plateau, des cases, des joueurs…). Sauf cas particulier, les attributs ne devront être accessibles que par leurs accesseurs.

4.2. Pré-rapport

Au cours de la semaine du 5 au 9 avril, vous devrez remettre à l'enseignant responsable un pré-rapport au format pdf (format aisément lisible sur tous les systèmes). Il doit rendre compte de votre analyse du problème. Son objet est de vous obliger à expliciter votre réflexion sur le projet et de nous permettre de vous donner si besoin en retour des conseils, que ce soit sur la façon de décrire l'analyse d'une situation, ou sur les solutions que vous proposez.
Il ne sera pas noté mais annoté. Il vous servira de base pour le rapport final demandé par ailleurs (voir plus loin).
Le pré-rapport contiendra les informations suivantes :

- ✓ **Diagramme de classes**. Description de la structure des classes à l'aide d'un diagramme, qui montre les relations d'utilisation et d'héritage ainsi que les attributs principaux.
- ✓ **Descriptif des classes**. Information sur le contenu des classes principales : rôles des attributs et des méthodes. Expliquez comment vous pensez réaliser les méthodes les plus complexes.

4.3. Démonstration

La démonstration se situera au cours de la semaine du 3 au 7 mai dans un créneau de 20 minutes, comportant votre présentation et les questions du/des enseignant/e/s. Ses modalités exactes seront définies en fonction des contraintes sanitaires du moment.
Elle est destinée à présenter les principales fonctionnalités de votre réalisation.

Il est important de faire attention aux points suivants :

- ✓ Respectez le temps imparti.
- ✓ Testez au préalable ce que vous voulez présenter (n'improvisez pas) ; les problèmes sous-jacents apparaissent souvent pendant les démos.
- ✓ Structurez votre démonstration autour de jeux d'essais, c'est-à-dire d'exemples déroulés qui montrent bien ce que fait votre programme. Pour ce faire, vous pouvez avoir un mode particulier de votre application qui montre les cases.
- ✓ Ne vous perdez pas dans les détails techniques en parlant de votre programme. L'enseignant vous posera des questions techniques s'il le juge utile.
- ✓ Tous les participants au projet doivent prendre la parole de manière à peu près égale. Vous devez être le plus naturel possible lors des passages de parole et parler distinctement.

En résumé, une démonstration doit être soigneusement préparée.

5. Échéances et conseils

Semaine du 5 au 9 avril

Remise du pré-rapport.

Semaine du 3 au 7 mai (la deuxième semaine après les vacances de Pâques)

Démonstration du projet dans un des créneaux qui seront proposés (vous devrez vous inscrire). Vous devez disposer d'un fichier compressé (extension zip : format assez standard) à votre/vos nom/s contenant :

- ✓ le rapport complété au format pdf. C'est le même rapport que le pré-rapport sauf qu'il porte sur le projet effectivement réalisé.
- ✓ Le code du projet,
- ✓ les autres documents que vous souhaitez communiquer (jeux d'essais).

Il est conseillé de ne dépasser les fonctionnalités de base que si ces dernières sont bien réalisées et que vous êtes à jour dans les autres matières.
Rappel : il est (vivement) conseillé d'obtenir avant tout une version qui fonctionne, même avec des fonctionnalités plus restreintes que celles qui sont spécifiées par l'énoncé.