HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

INF2 – Andrei Popescu-Belis et Gabriel Luthier

Labo 3 : Jeu des 7 familles

But

Ce projet vise à construire un système pouvant jouer au jeu des 7 familles. Le système sera implémenté sous la forme d'une classe `Joueur`, accompagnée d'une classe `Partie` qui contrôle le jeu. On souhaite faire jouer plusieurs joueurs entre eux, observer le déroulement de la partie, et faire des statistiques. *On ne souhaite pas qu'un humain puisse jouer contre le système.*

Source : https://dessinemoiunehistoire.net/

Règles du jeu

Le jeu des 7 familles utilise un jeu de cartes constitué de groupes de cartes qui sont appelés « familles ». Le but de chaque joueur est d'avoir un maximum de familles à la fin d'une partie. Les objets représentés sur les cartes varient d'un jeu à l'autre. À l'origine, une famille était constituée de six cartes (aïeule, aïeul, mère, père, fille, fils), et chaque famille était distinguée par une couleur.

Dans ce projet, on représentera les familles par des nombres (de 1 à NOMBRE_FAMILLES) et chaque membre d'une famille sera noté par une lettre : A, B, etc. (de A à un certain nombre de CARTES_PAR_FAMILLE). Deux autres paramètres sont le NOMBRE_JOUEURS et le nombre initial de CARTES_PAR_JOUEUR.

Par exemple, on peut définir un jeu avec 9 familles de 4 cartes chacune, auquel jouent 4 joueurs, et initialement on distribue 5 cartes à chaque joueur. Les cartes non distribuées sont posées sur la table et forment la pioche. Dans cet exemple, la pioche initiale contiendra 16 cartes ($9 \times 4 - 4 \times 5 = 16$).

Déroulement d'une partie

- Tour-à-tour, chaque joueur choisit librement un autre joueur, et lui demande un membre précis d'une famille, p.ex. : « Dans la famille 3, je te demande le B ». Le demandeur doit avoir en main au moins un membre de cette famille pour avoir le droit d'en demander un autre.
- Si le joueur à qui on demande possède la carte demandée, il la donne au premier, qui peut redemander une carte, toujours au même joueur. S'il ne possède pas cette carte, le demandeur prend une carte dans la pioche (s'il en reste), et le tour passe au joueur suivant.
- Lorsqu'un joueur a tous les membres d'une famille, il pose la famille sur la table et la conserve.
- Le jeu finit lorsque plus personne n'a de cartes en main et que la pioche est vide.
- À la fin, chaque joueur gagne autant de points que de familles posées devant lui.
- Après plusieurs parties, le gagnant est celui qui a le plus grand total de points.

Travail demandé

Le projet devra simuler plusieurs joueurs (instances de la classe `Joueur`) qui s'affrontent dans une partie (instance de `Partie`). Le système distribue les cartes (instances de la classe `Carte`) au hasard, puis tour-à-tour chaque joueur demande des cartes à un autre, selon les règles ci-dessus, mais sans stratégie

« intelligente ». À chaque fois qu'un joueur reçoit une carte, il vérifie s'il a complété une famille, auquel cas il la pose devant lui. On affichera la partie selon le modèle fourni en annexe.

Le système (dans le `main()`) fera jouer 100 parties avec les paramètres de l'exemple ci-dessus, et indiquera le total obtenu par chaque joueur, divisé par le NOMBRE_FAMILLES pour normaliser le score en %. Les joueurs jouent toujours dans le même ordre, et on vous demande de comparer deux cas :

(1) C'est toujours le même joueur qui commence (p.ex. le premier)
(2) Les joueurs commencent tour à tour, dans le même ordre que celui du jeu

Question. En répétant plusieurs fois la série de 100 parties avec le cas (1) puis le cas (2), peut-on dire que le joueur qui commence est favorisé ou défavorisé ?

Bonus. Copier la classe `Joueur` dans une nouvelle classe appelée `MeilleurJoueur` (ou faire une classe dérivée si on sait) puis améliorer la stratégie de `MeilleurJoueur` seulement (voir une suggestion ci-dessous). Faire plusieurs séries de 100 parties avec un joueur de cette classe et trois de l'autre, et observer si son score final est meilleur que celui des autres.

Projet à rendre

Le projet est à effectuer par groupes de trois. Il doit être **rendu avant la date indiquée sur Cyberlearn**, sous la forme d'un fichier ZIP appelé Labo3_Nom1_Nom2_Nom3.zip contenant :

- Les fichiers source commentés selon les indications données en INF1 et INF2. Si possible le dossier du projet sous NetBeans, sans les dossiers `build` et `dist`.
- Un rapport (PDF, 2-3 pages) présentant : les choix d'implémentation et les difficultés résolues ; les scores finaux des joueurs après plusieurs séries de 100 parties et la réponse à la question ci-dessus ; éventuellement la réponse à la question bonus.
- Un fichier texte contenant un exemple d'exécution d'une partie.

Contraintes et suggestions

- Utiliser trois classes : `Partie`, `Joueur`, `Carte` et séparer les fichiers .h et .cpp
- Une `Carte` est définie par la famille et le membre, qui sont à afficher comme un nombre suivi d'une lettre, mais peuvent être codés tous deux par des entiers (`unsigned short`)
- Un `Joueur` a un nom et deux tableaux de cartes : les `cartesEnMain` et les `famillesSurTable`
- Un `Joueur` doit pouvoir décider quelle carte demander (fonction membre `demanderCarte`) en fonction seulement des cartes qu'il a en main. Stratégie basique : compléter une famille au hasard ; ou, pour la tâche *bonus*, compléter la famille dont on possède déjà le plus de membres. *On ne demande pas de mémoriser les échanges précédents.*
- Un objet `Joueur` possède une fonction membre `detecterFamille` qui cherche dans les `cartesEnMain` si une famille est complète, et la pose dans les `famillesSurTable`
- La `Partie` gère la pioche de cartes et organise le jeu. Elle donne le tour à chaque joueur et tire au hasard le joueur à qui on demande des cartes (à condition qu'il en ait en main). Elle détecte aussi la fin de la partie, quand aucun joueur n'a plus de cartes en main et la pioche est vide.
- On contrôle depuis le `main()` le lancement de plusieurs parties successives et la gestion des scores totaux des joueurs, retournés par l'instance de `Partie` après chaque partie.

Annexe : affichage d'une partie (extrait)

```
Début de la partie de 7 familles
*** Tour 1 ***
Alice : 2C 9A 1D 4D 7B  []
Bobby : 2D 3D 4C 7A 8C  []
Carol : 4B 1B 9D 8A 5C  []
Danny : 1A 2B 3B 2A 5A  []
Pioche: 5D 6A 7C 6B 9B 6D 1C 3A 8D 4A 9C 3C
6C 5B 8B 7D
Alice demande à Bobby la carte 2A
  mais Bobby ne l'a pas
Alice prend une carte dans la pioche (7D)
Bobby demande à Alice la carte 2A
  mais Alice ne l'a pas
Bobby prend une carte dans la pioche (8B)
Carol demande à Alice la carte 4A
  mais Alice ne l'a pas
Carol prend une carte dans la pioche (5B)
Danny demande à Bobby la carte 1B
  mais Bobby ne l'a pas
Danny prend une carte dans la pioche (6C)
*** Tour 2 ***
Alice : 2C 9A 1D 4D 7B 7D  []
Bobby : 2D 3D 4C 7A 8C 8B  []
Carol : 4B 1B 9D 8A 5C 5B  []
Danny : 1A 2B 3B 2A 5A 6C  []
Pioche: 5D 6A 7C 6B 9B 6D 1C 3A 8D 4A 9C 3C
Alice demande à Bobby la carte 2A
  mais Bobby ne l'a pas
Alice prend une carte dans la pioche (3C)
Bobby demande à Alice la carte 2A
  mais Alice ne l'a pas
Bobby prend une carte dans la pioche (9C)
Carol demande à Bobby la carte 4A
  mais Bobby ne l'a pas
Carol prend une carte dans la pioche (4A)
Danny demande à Bobby la carte 1B
  mais Bobby ne l'a pas
Danny prend une carte dans la pioche (8D)
*** Tour 3 ***
Alice : 2C 9A 1D 4D 7B 7D 3C  []
Bobby : 2D 3D 4C 7A 8C 8B 9C  []
Carol : 4B 1B 9D 8A 5C 5B 4A  []
Danny : 1A 2B 3B 2A 5A 6C 8D  []
Pioche: 5D 6A 7C 6B 9B 6D 1C 3A
Alice demande à Bobby la carte 2A
  mais Bobby ne l'a pas
Alice prend une carte dans la pioche (3A)
Bobby demande à Danny la carte 2A
  et Danny donne la carte à Bobby
Bobby demande à Danny la carte 2B
  et Danny donne la carte à Bobby
Bobby demande à Danny la carte 2C
  mais Danny ne l'a pas
Bobby prend une carte dans la pioche (1C)
Carol demande à Danny la carte 4C
  mais Danny ne l'a pas
Carol prend une carte dans la pioche (6D)
Danny demande à Carol la carte 1B
  et Carol donne la carte à Danny
Danny demande à Carol la carte 1C
  mais Carol ne l'a pas
Danny prend une carte dans la pioche (9B)
*** Tour 4 ***
Alice : 2C 9A 1D 4D 7B 7D 3C 3A  []
Bobby : 2D 3D 4C 7A 8C 8B 9C 2A 2B 1C  []
Carol : 4B 9D 8A 5C 5B 4A 6D  []
Danny : 1A 3B 5A 6C 8D 1B 9B  []
Pioche: 5D 6A 7C 6B

...

*** Tour 15 ***
Alice :  [2C.2A.2B.2D.]
Bobby : 9C
[3D.3A.3B.3C.7A.7C.7B.7D.8C.8B.8A.8D.]
Carol : 9D 9A 9B  [4B.4A.4C.4D.]
Danny :
[1A.1B.1C.1D.5A.5D.5B.5C.6C.6A.6B.6D.]
Pioche:
Bobby demande à Carol la carte 9A
  et Carol donne la carte à Bobby
Bobby demande à Carol la carte 9B
  et Carol donne la carte à Bobby
Bobby demande à Carol la carte 9D
  et Carol donne la carte à Bobby
Bobby demande à Carol la carte 9A
  mais Carol ne l'a pas
Alice :  [2C.2A.2B.2D.]
Bobby :  [3D.3A.3B.3C.7A.7C.7B.7D.
8C.8B.8A.8D.9C.9A.9B.9D.]
Carol :  [4B.4A.4C.4D.]
Danny :  [1A.1B.1C.1D.5A.5D.5B.5C.
6C.6A.6B.6D.]
Pioche:
La partie est finie !
Nombre de tours : 15
RUN RÉUSSI  (temps total:  156ms)
```