

Lab/Devoir n°5 – Mardi/Vendredi 17/20 Octobre 2023

CSI2772A – Automne 2023

SITE - Université d'Ottawa

Dû EN LIGNE le vendredi 3 Novembre à 22:00

En groupes d'au plus deux étudiant(e)s.

/10

Nous explorons à nouveau les classes dans ce travail de laboratoire. Nous examinerons de plus près les classes travaillant ensemble et utilisant des boucles et l'indexation. Nous explorons un objet de classe en tant que variable dans une autre classe (agrégation).

Jeu de cartes 21 :

L'exemple consiste à écrire un programme permettant à l'utilisateur de jouer au jeu de cartes 21. L'utilisateur reçoit une carte à la fois et après chaque carte, décide de continuer ou non. Le but est, en additionnant les points des cartes que l'utilisateur a en main, de se rapprocher le plus possible de 21 sans dépasser cette valeur. Si l'utilisateur a plus de 21 points, il perd, sinon c'est au tour de l'ordinateur. Si au moment où il s'arrête, il a plus de 21 ou moins que l'utilisateur, l'utilisateur gagne, sinon l'ordinateur gagne. Un as vaut 1 ou 14.

3 classes sont fournies ci-joint ((fichiers *Card.h*, *CardsSet.h* et *Player.h*).

- **Classe Card** , doit permettre de :

- créer une carte : constructeur (fourni)
- connaître sa valeur : méthode `value()` (méthode en ligne fournie)
- écrire (afficher) les informations (couleur et valeur) : méthode `write()`

- **Classe CardsSet** :

La classe `CardsSet` devrait être capable de représenter à la fois le jeu entier (52 cartes) et le jeu de cartes qu'un joueur a en main. Les opérations à effectuer sont :

- créer une plateforme vide : constructeur (inline, fourni)
- créer un nouveau jeu qui correspond aux 52 cartes : `newSet()`
- mélanger ces cartes : `shuffle()`
- savoir combien de cartes sont dans le jeu : `numCards()` (méthode en ligne fournie)
- prendre une carte du jeu et la renvoyer : `take()`
- mettre une carte sur le plateau : `put()`
- trouver la carte en position *no* dans l'ensemble : `lookIn` (*no* = 1 pour le top du set)
- retirer toutes les cartes du jeu : `empty()` (méthode en ligne fournie)

- **Classe Player** qui représente le joueur:

La classe `Player` doit permettre de :

- prendre une carte au sommet du paquet et la mettre dans la main : `play()`
- compter les points des cartes en main et renvoyer ce nombre, aussi bien pour l'utilisateur que pour l'ordinateur : `countPoints()`

Question 1: (2 POINTS)

Implémentez la méthode suivante pour la classe `Card` :

- `void write()` : affiche la couleur suivie de la valeur de chaque carte.
(les valeurs vont de 1 à 13 (11 pour *jack* , 12 pour *queen* , et 13 pour *king*))

Question 2: (5 POINTS)

Implémentez les méthodes suivantes pour la classe `CardsSet`:

- `void novSet()`
- `void shuffle ()`
- `Card take()`
- `void put(Card k)`
- `Card lookIn(int no)`
 - `no = 1` for the top of the set

Pour mélanger les cartes, vous pouvez utiliser la fonction standard `rand` qui vous permet d'obtenir des nombres aléatoires. Le processus sera initialisé avec la fonction `srand` à laquelle le résultat de la fonction `time` sera passé en paramètre. Vous devez donc inclure les éléments suivants si vous le faites :

```
#include <cstdlib>
#include <ctime>
```

Question 3: (3 POINTS)

Implémentez les méthodes suivantes pour la classe `Player`:

- `int play ()`
- `int countPoints()`

en considérant que:

- Si chaque joueur a son propre paquet de cartes en main, le paquet dans lequel on pioche est commun aux deux joueurs, donc le paquet est une référence : `&`
 - `computer` est déclaré comme constant car cet attribut de joueur ne change pas
 - la méthode `play` peut utiliser une variable booléenne pour savoir si on veut une Carte supplémentaire ou non. La méthode `play` renvoie le nombre de points de la main.
 - la méthode `counPoints` est privée car utilisée en interne par la fonction `play`.
- Si nous dépassons 21 points, nous comptons l'As pour 1, sinon il compte pour 14.

Créez les fichiers sources `myFile.cpp` et le fichier d'en-tête `myFile.h`

Note: Vous n'êtes pas autorisé à modifier aucune déclaration de classe, ni le programme principale (*main*).

*/*Exemple de sortie*/*

Hello!

A new game?

y

You get Card: 5 of Clubs

Your score is 5 points

Any additional Card ? y

You get Card: Ace of Spades

Your score is 19 points

Any additional Card ? y

You get Card: 9 of Hearts

Your score is 15 points

Any additional Card ? y

You get Card: 8 of Clubs

Your score is 23 points

You lost!

A new game?

y

You get Card: 8 of Spades

Your score is 8 points

Any additional Card ? y

You get Card: King of Spades

Your score is 21 points

You won!

A new game?

n

Créer et soumettre un seul fichier zip

Directives

- Créez un répertoire que vous nommerez *Devoir5_ID* comme d'habitude, où vous remplacerez ID par votre numéro d'étudiant (celui qui soumet le devoir).
 - ✓ Mettez tous les fichiers (*myFile.cpp*, *myFile.h*, ...) dans votre répertoire compressé *Devoir5_ID.zip* pour soumission dans le campus virtuel Brightspace.
- Dans le répertoire *Devoir5_ID*, créez un fichier texte nommé *README.txt*, qui devra contenir **les noms des deux étudiant(e)s**, ainsi qu'une brève description du contenu :

Nom étudiant :

Numéro d'étudiant :

Code du cours : CSI2772A

Fraude scolaire :

Cette partie du devoir a pour but de sensibiliser les étudiants face au problème de fraude scolaire (plagiat). Consulter les liens suivants et bien lire les deux documents:

<https://www.uottawa.ca/administration-et-gouvernance/reglement-scolaire-14-autres-informations-importantes>

https://www.uottawa.ca/administration-et-gouvernance/sites/www.uottawa.ca/administration-et-gouvernance/files/processus_de_traitement_des_cas_de_fraude_academique_-_nov_2019.pdf

Les règlements de l'université seront appliqués pour tout cas de plagiat.

En soumettant ce devoir :

1. vous témoignez avoir lu les documents ci-haut ;
2. vous comprenez les conséquences de la fraude scolaire.