Travail 4

Pondération: 8%
Remise : vendredi 10 mai avant minuit (via Léa)
Prenez le soin d'enlever le ipch, le .db et les dossiers debug

Le but de ce travail est de mettre en pratique les notions entourant la surcharge d'opérateur ainsi que l'utilisation de la STL. Une attention particulière doit être portée sur l'application du principe Tell don't Ask.

Le travail peut se faire en équipe de 2.

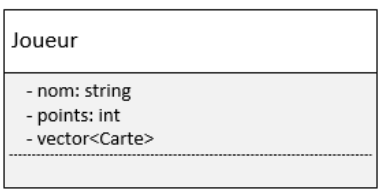
Exigences

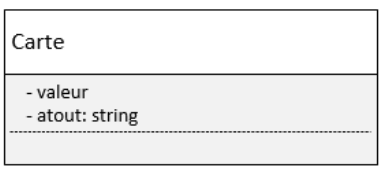
- Vous devez utiliser les vector de la STL.
- **Vous devez utiliser les algorithmes de la STL à chaque fois qu'il est possible de le faire.**
- **Vous devez évidemment surcharger tous les opérateurs nécessaires à l'emploi de ces vector et de ces algorithmes.**
- **Pour les autres traitements, demandez-vous s'il est possible d'utiliser un opérateur surchargé. Si oui, surchargez et utilisez cet opérateur.**
- Ne surchargez que les opérateurs nécessaires (pas besoin de surcharger par paire) et n'oubliez pas qu'un opérateur doit garder sa signification.
- L'utilisation d'un algorithme de la STL doit toujours être privilégiée à l'utilisation d'un opérateur surchargé (ex : utiliser count() au lieu de parcourir un vector et de vérifier si la valeur == celle désirée).
- Respectez les principes de la programmation objet.
- **N'oubliez pas que les objets doivent être traités à l'endroit où ils sont déclarés (ex : les cartes sont déclarées dans le joueur).**
- Utilisez vos objets le plus souvent possibles et non les valeurs affichées à l'écran.
- Respectez les standards de programmation et les bonnes techniques de programmation.

Description du travail

Le travail consiste à programmer un petit jeu de hasard utilisant des cartes. Les joueurs possèdent plusieurs cartes (relation **d'agrégation**). Les classes suivantes sont utilisées.

Après transformation, les classes à utiliser pour le travail sont les suivantes.





- Le projet contient déjà les classes à utiliser.
    - Vous **ne** devez **pas** ajouter d'attributs dans les classes.
    - Vous devez évidemment ajouter des méthodes.
    - N'ajoutez **pas** de set() dans les classes.
    - Étant donné qu'il s'agit d'une relation d'agrégation (et non de composition), vous pouvez faire des return du type d'une classe ☺.
    - Vous devez surcharger les opérateurs nécessaires au bon fonctionnement de la STL et des algorithmes.
- La classe Donnee est fournie.
    - Elle contient un vector de Joueur, lesquels sont initialisés à l'aide du constructeur.
- Dans MyForm :
    - Au début de la 'form' est déclaré un objet de type Donnees, 3 variables globales ainsi qu'une constante maxCartes.
        - Vous ne devez pas ajouter d'autres variables globales.
        - Vous ne devez pas ajouter d'autres constantes et ce, dans l'ensemble de l'application.
    - Une partie du code pour afficher est déjà fournie. Remarquez l'emploi de List servant à garder des ListBox et des TextBox. Cela permet d'utiliser des indices lors de l'affichage (voir AfficherValeurs).

Fonctionnalités à programmer

- Complétez la fonction AfficherLesCartes() afin de faire afficher toutes les cartes de chaque joueur dans les ListBox en respectant les critères qui suivent :
    - Pour chaque carte, faites afficher la valeur ainsi que l'atout de la carte. Pour ce faire, faites une fonction ToString() dans la classe Carte qui retournera les 2 valeurs déjà concaténées.
    - Les cartes doivent être triées afin de les voir en ordre croissant de la valeur.
    - Vous pourrez tester l'affichage au prochain point.
- Complétez le bouton 'Distribuer' qui doit :
    - Donner 10 cartes à chaque joueur. Les cartes doivent être générées au hasard en utilisant la fonction GenererCarte() qui existe déjà. Il est possible que les mêmes cartes se répètent. Les cartes de chaque joueur s'affichent grâce à AfficherLesCartes.
    - Générer une carte chanceuse (variable globale) et la faire afficher.
    - Générer une valeur chanceuse entre 1 e t13 (variable globale) et la faire afficher.
    - Générer un atout chanceux (variable globale) et le faire afficher.
- Complétez le bouton 'Total' qui doit, pour chaque joueur :
    - Faire la somme de ses cartes.
    - Ajouter cette somme aux points du joueur.
    - Faire afficher les points obtenus pour chacun des joueurs en utilisant AfficherValeurs.
    - Un message ainsi que les points totaux de chaque joueur s'affichent déjà.

- Complétez le bouton ‘Pareil’ afin de donner des points à chaque joueur selon le nombre de cartes pareilles qu’il possède:
    - 4 cartes pareilles ou plus : 50 points
    - 3 cartes pareilles : 10 points
    - 2 cartes pareilles : 5 points
    - Les points obtenus doivent s’afficher.
    - Un message ainsi que les points totaux de chaque joueur s’affichent déjà.
- Complétez le bouton ‘Carte chanceuse’ afin que chaque joueur qui possède la carte chanceuse obtienne 25 points. Encore une fois faites afficher les points obtenus par chaque joueur.
- Complétez aussi le bouton ‘Valeurs chanceuses’ afin que chaque joueur :
    - Qui possède la valeur chanceuse obtienne des points équivalents à la valeur de la carte.
    - Qui possède l’atout chanceux obtienne 5 points supplémentaires.
    - N’oubliez pas de faire afficher les points obtenus par chaque joueur.
- Complétez le bouton ‘Grandes Valeurs’ qui doit ajouter et faire afficher 50 points à chaque joueur qui possèdent au moins 5 cartes dont la valeur est plus grande que 10.
- En tout temps, le nom du joueur qui est avance (celui dont les points sont le plus élevés) doit être affiché.

Critères de correction et répartition des points

- Application des règles de la programmation objet
- Respect du standard de programmation et des bonnes pratiques de programmation
- Utilisation adéquate des classes et des objets
- Ajout des méthodes nécessaires dans les différentes classes
- Fonctionnement selon les règles énoncées
- Respect des exigences demandées
- Surcharge et utilisation des opérateurs nécessaires
- Utilisation des bons algorithmes pour obtenir le résultat souhaité
- Répartition des points :
    - Génération et affichage des cartes du joueur **25 pts**
    - Affichage joueur en avance **10 pts**
    - Total des cartes **15 pts**
    - Cartes pareilles **15 pts**
    - Carte et valeurs chanceuses **20 pts**
    - Trouver les grandes valeurs **15 pts**