Programmation objet

Réalisation d'un Monopoly en C++

Quentin HAEMMERLÉ, Inès KEBBAB, Théo SAULAIS, Léa TRIQUET

Table des matières

1/ Explication des classes

a) Classe Case

La classe Case n'a qu'un seul attribut :

- nom qui est une chaine de caractère et qui correspond au nom de la case.

Avec cet attribut, on rédige les fonctions suivantes :

- Case() qui permet de créer une case vide.
- Case() qui permet de nommer une case à l'aide d'une chaine de caractère.
- Un getter et un setter pour le nom de la case.
- affiche() qui permet d'afficher le nom de la case.

b) Classe Chance

La classe Chance hérite de la classe Case et ne possède pas d'attribut :

On rédige les fonctions suivantes :

- arreterSur() qui détermine la carte piochée et l'action associée.

c) Classe Communauté

La classe Communauté hérite de la classe Case ne possède pas d'attribut :

On rédige les fonctions suivantes :

- arreterSur() qui détermine la carte piochée et l'action associée.

d) Classe Compagnie

La classe Compagnie hérite de la classe Propriété et possède un seul attribut :

- groupe qui est un pointeur vers le tableau des compagnies

Avec cet attribut, on rédige les fonctions suivantes :

- Compagnie() qui est le constructeur des compagnies
- Un getter et un setter pour le nom de la compagnie.
- arretSur() qui permet d'une part d'acheter la compagnie et d'autre part de payer le loyer correspondant.

e) Classe Dé

La classe Dé possède trois attributs :

- valeur1 qui est un entier et qui correspond à la valeur du premier dé.
- valeur2 qui est un entier et qui correspond à la valeur du second dé.
- pair qui est un booléen et qui permet de savoir si le joueur a fait un double en lançant les dés.

Avec ces attributs, on rédige les fonctions suivantes :

- Un constructeur de dés qui initialise les valeurs des dés à 0 et qui impose le booléen comme étant false.
- Un destructeur des dés.
- Deux getters qui permettent de récupérer les deux valeurs des dés.
- Un getter qui permet de récupérer la somme des deux dés.
- Double() qui permet de vérifier si les deux dés ont la même valeur.
- LancerDe() qui permet de lancer les dés (obtention de deux nombres aléatoires entre 1 et 6).

f) Classe Depart

La classe Depart hérite de la classe Case et ne possède pas d'attribut.

Elle contient les fonctions suivantes :

- Depart() qui initialise le nom de la case départ.
- arretSur() qui permet de créditer 400€ sur le solde du joueur si ce dernier s'arrête sur la case.

g) Classe Gare

La classe Gare hérite de la classe Propriete et possède un seul attribut :

- groupe qui est un tableau de Gare

Avec cet attribut, on rédige les fonctions suivantes :

- Gare() qui est le constructeur des gares
- Un getter et un setter pour obtenir le tableau des gares
- arretSur() qui permet d'une part d'acheter la gare et d'autre part de payer le loyer correspondant.

h) Classe jeu

La classe Jeu possède les attributs suivants :

- tourdejeu qui est un entier et qui correspond au n-ème tour de jeu.
- nbJoueurs qui est un entier et qui correspond au nombre de joueurs.
- plateau qui est un plateau et qui correspond au plateau de jeu.
- joueur[] qui est un Joueur et qui correspond à la liste des joueurs engagés.
- de qui est un Dé et qui correspond aux deux dés de jeu.

Avec ces attributs, on rédige les fonctions suivantes :

- Jeu() qui est le constructeur de jeu et qui est initialisé à 2 joueurs par défaut.
- 3 setters pour le tour de jeu, le plateau et les joueurs.
- Compteur() qui renvoie le tour de jeu actuel.
- lancePartie() qui permet de lancer la partie avec les joueurs et les pions associés.
- tourDeJeu() qui définit un tour de jeu : lancement des dés, déplacement du pion, gestion des doubles lors du lancement des dés.
- finPartie qui permet d'arrêter la partie s'il ne reste plus qu'un seul joueur.

i) Classe Joueur

La classe Joueur est liée aux classes Pions, Propriété et Jeu. Elle possède les attributs suivants :

- nom qui est une chaine de caractère et qui correspond au nom du joueur.
- pion qui est un Pion et qui correspond au pion choisit par le joueur.
- jeu qui est un Jeu et qui correspond à la partie associée au joueur
- prison qui est un entier initialisé à 0 et qui correspond au nombre de tours consécutifs passés en prison par le joueur
- solde qui est un entier et qui correspond au solde du joueur

Avec ces attributs, on rédige les fonctions suivantes :

- Un constructeur de joueur vide.
- Un constructeur par défaut qui prend en entrée le nom du joueur, son pion, le jeu associé, et qui définit le solde à 1500€.
- 6 getters qui renvoient le nom du joueur, son pion, son solde, la partie associée, le nombre de tours en prison et les possessions du joueur.
- 6 setters associés.
- Jouer() qui prend la valeur des dés en entrée et qui permet au joueur d'avancer.

- Créditer() qui permet de créditer de l'argent au joueur.
- Débiter() qui permet de débiter de l'argent au joueur.
- Perdu() qui permet de définir qu'un joueur a perdu en retirant toutes ses possessions et en mettant son solde en dessous de 0.
- AfficheSolde() qui permet d'afficher le solde du joueur.
- AffichePion() qui permet d'afficher le pion du joueur

j) Classe Parc

La classe Parc hérite de la classe Case. Elle ne possède pas d'attributs.

Elle possède les fonctions suivantes :

- Parc() qui correspond à son constructeur.
- arretSur() qui permet de gérer l'arret sur la case.

k) Classe Pion

La classe Pion est liée aux classes Case et Joueur. Elle possède les attributs suivants :

- joueur qui est un Joueur et qui correspond au joueur associé au pion.
- position qui est une Case et qui correspond à la position du pion sur le plateau.
- nom qui est une chaine de caractères et qui correspond au nom du pion du joueur.

Avec ces attributs, on rédige les fonctions suivantes :

- 3 getters et setters pour le obtenir/définir le joueur, la position du pion et son nom.
- Deplacer() qui permet de déplacer le pion sur sa nouvelle case et d'afficher le nom de la case en question.
- goToPrison() qui permet d'envoyer le joueur en prison.

l) Classe Plateau

La classe Plateau possède les attributs suivants :

- lesTerrains qui renvoient aux 22 terrains du plateau.
- lesGares qui renvoient aux 4 gares du plateau.
- lesCompagnies qui renvoient aus 2 compagnies du plateau.
- lesCommunaute qui renvoient aux 3 caisse de communauté du plateau.
- LesChances qui renvoient aux 3 cases chance du plateau.
- Liste_case qui renvoie aux 40 cases du plateau.
- lesTaxes qui renvoient aux 2 cases de taxes du plateau.
- LeDepart qui renvoie à la case départ.

- lesPrisons qui renvoient aux 2 cases liées à la prison sur le plateau.
- leParc qui renvoie à la case du parc gratuit.

m) Classe Prison

La classe Prison hérite de la classe Case. Elle ne possède pas d'attributs :

Elle contient les fonctions suivantes :

- Prison() qui définit le nom de la case prison.
- arretSur() qui correspond à l'arrêt sur la case « Aller en prison » et qui envoie le joueur en prison.

n) Classe Propriété

La classe Propriété hérite de la classe Case. Elle possède les attributs suivants :

- Propriétaire qui est un Joueur et qui renvoie au propriétaire de la propriété.
- prixAchat qui est un entier et qui correspond au prix d'achat de la propriété
- hyp qui est un booléen et qui correspond au statut d'hypothèque de la propriété.

Avec ces attributs, on rédige les fonctions suivantes :

- Propriete() qui définit le nom de la propriété, son propriétaire, son état d'hypothèque et son prix d'achet
- 3 setters et getters pour le propriétaire, l'état d'hypothèque et le prix d'achat.
- hypotheque() qui permet d'hypothéquer la propriété et de récupérer 50% de sa valeur.
- arretSur() qui permet d'acheter la propriété si cette dernière ne possède pas de propriétaire.

o) Classe Taxes

La classe Taxes hérite de Case et possède l'attribut suivant :

- montant qui est un entier et qui correspond au montant de la taxe

Avec cet attribut, on définit les fonctions suivantes :

- Taxes() qui définit le nom de la taxe et son montant.
- arretSur() qui débite le joueur lorsque ce dernier s'arrête dessus.

p) Classe Terrain

La classe Terrain hérite de la classe Propriété. Elle possède les attributs suivants :

- maison qui est un entier et qui correspond au nombre de maison sur le terrain.
- hôtel qui est un booléen qui permet de savoir si le joueur possède un hôtel.
- quartier qui est un pointeur vers la liste des terrains
- loyers qui est une liste d'entiers qui correspond à la liste des différents loyers du terrain.

Avec ces attributs, on rédige les fonctions suivantes :

- Terrain() qui permet d'initialiser le terrain.
- 4 getters et setters qui permettent d'obtenir et définir les maisons, hôtels, quartiers et loyers associés aux terrains
- achatImmo() qui permet de tenir les comptes quant à l'achat de l'ensemble du quartier. La fonction permet également de construire des maisons et des hôtels sur les terrains.
- arretSur() qui permet d'une part d'acheter la gare et d'autre part de payer le loyer correspondant.
- Hypothèque() qui permet de gérer les hypothèques lorsque le joueur possède des maisons ou hôtels sur les propriétés concernées.