2024

PROJET DROLE DE ZEBRE

L2 SCIENCES COGNITIVES

MAE DUGOUA-JACQUES, JONATHAN ADAM
IDMC
Université de Lorraine

Table des matières

Préambule

Ce projet nous a été demandé dans le cadre du cours de Programmation orientée objet à l'IDMC, université de lorraine pour notre validation de deuxième année en filière MIASHS Sciences Cognitives.

Le but était de réaliser un jeu du Drôle de Zèbre. Pour cela nous avons tout d'abord réalisé une version du jeu sur console avant de passer à la version interface graphique.

Pour le jeu console :

- Le Joueur choisi d'abord sa carte par 1 ou 2
- Ensuite les deux joueurs indiquent leur pseudo, l'un après l'autre.
- Le premier joueur choisi aléatoirement doit alors choisir où mettre impala pour le premier tour. Il indique donc d'abord la ligne où il veut placer impala entre 1 et 6 puis la colonne entre 1 et 7 en prenant en compte que l'impala doit être au bord du plateau et pas dans les coins.
- Des lors le tour commence, on peut commencer par indiquer si on veut voir la carte par o ou n majuscule et minuscule confondue.
- Ensuite on sélectionne le pion que l'on veut jouer en choisissant le numéro indiqué de ce pion
- Ensuite on sélectionne la case où on veut le placer sachant que la case en haut à gauche est la case (1 ;1).
- Si un secteur est plein par le dernier pion posé et que le point inauguration n'a pas encore été attribué, un massage s'affiche pour indiquer au joueur qu'il l'a remporté.
- [Déplacer Impala]
- Ensuite c'est la tour de l'autre joueur et on recommence jusqu'à ce que le plateau soit plein.
- Enfin le programme compte les points et désigne soit le gagnant soit l'égalité.

Nous avons tenté de travailler avec git et github et des drives pour nous partager le projet.

Nous avons tout d'abord imaginé le code sur papier en pseudo code et en schéma (notamment pour les relations d'extensions).

Nous nous sommes ensuite partagé le travail de manière chronologique, rédigeant en premier temps la méthode jeu de la classe Jeu pour suivre les méthodes nécessaires à sa réalisation.

Classes retenues

Classe Pion

Description

Classe abstraite qui est classe mère de la classe Animal et ImpalaJones.

Elle a comme attributs :

Int indicateur

Cette variable sert à identifier le pion comme étant un impala, une Gazelle ect…

String couleur

Cette variable sert à attribuer une couleur et donc un joueur à qui appartient ce pion

Elle sert à généraliser ces attributs, communs à toutes ses classes filles.

Fonctions et Méthodes

Dans un premier temps nous retrouvons les getters :

public String getIndicateur()

public String getCouleur()

Puis nous retrouvons les Setters :

Public void setCouleur(String v)

Classe ImpalaJones

Description

ImpalaJones est une classe fille de Pion et une classe finale car elle n'a aucun fils.

Elle n'a pas d'attributs car elle utilise ceux de sa classe mère.

Elle a un constructeur car ce n'est pas une classe abstraite, celui-ci ne prend aucun paramètre et créé un Impala reconnaissable à son indicateur « imp » et de couleur null car il n'appartient à aucun joueur.

Fonctions et Méthodes

Constructeur :

Public ImpalaJones()

Classe Animal

Description

Animal est une classe fille de la classe Pion. C'est une classe abstraite, elle sert à généraliser la méthode toString pour tous les animaux différents

Elle a comme attribut :

Int pts : qui est le nombre de points que rapporte ce Pion Animal.

Fonctions et Méthodes

Dans un premier temps nous retrouvons les getters :

public int getPts()

Et la redéfinition de la méthode toString() qui se construit ainsi :

Elle indique d'abord les points que valent le Pion, l'indicateur, et la couleur.

Elle est utilisée par exemple dans l'affichage du plateau

Classe Zebre

Description

C'est une classe fille de la classe Animal. C'est aussi une classe finale car il n'y a plus de fils derrière elle.

Elle représente le pion Zebre et a un attribut pts qu'elle récupère de sa classe mère et qui est égal à 6 d'après le constructeur car le Zebre rapporte 6 points. L'indicateur est aussi initialisé à Z pour permettre d'identifier le Zebre.

Il a aussi un attribut :

Boolean cache : qui sert à spécifier si le Zebre est caché ou non et si dans ce cas il ne vaut plus aucun point.

Fonctions et Méthodes

Constructeur :

public Zebre()

Initialise le nombre de pts et l'indicateur comme dit précédemment et initialise cache à faux.

Getter :

public boolean getCache()

Setter :

public void setCache(boolean a)

Classe Gazelle

Description

C'est une classe fille de la classe Animal. C'est aussi une classe finale car il n'y a plus de fils derrière elle.

Elle représente le Pion Gazelle et a un attribut pts qu'elle récupère de sa classe mère et qui est égal à 2 d'après le constructeur car le Gazelle rapporte 2 points. L'indicateur est aussi initialisé à G pour permettre d'identifier la Gazelle.

Il a aussi un attribut :

boolean cache : qui sert à spécifier si le Gazelle est caché ou non et si dans ce cas il ne vaut plus aucun point.

Fonctions et Méthodes

Constructeur :

public Gazelle()

Initialise le nombre de point et l'indicateur comme dit précédemment et initialise cache à faux.

Getter :

public boolean getCache()

Setter :

public void setCache(boolean a)

Classe Lion

Description

C'est une classe fille de la classe Animal. C'est aussi une classe finale car il n'y a plus de fils derrière elle.

Elle représente le pion Lion et a un attribut pts qu'elle récupère de sa classe mère et qui est égal à 1 d'après le constructeur car le Gazelle rapporte 1points. L'indicateur est aussi initialisé à L pour permettre d'identifier le Lion.

Fonctions et Méthodes

Constructeur :

public Lion()

Class Crocodile

Description

C'est une classe fille de la classe Animal. C'est aussi une classe finale car il n'y a plus de fils derrière elle.

Elle représente le Pion Crocodile et a un attribut pts qu'elle récupère de sa classe mère et qui est égal à 0 d'après le constructeur car le Gazelle rapporte 0 points. L'indicateur est aussi initialisé à C pour permettre d'identifier le Crocodile.

Fonctions et Méthodes

Constructeur :

public Crocodile()

Classe Elephant

Description

C'est une classe fille de la classe Animal. C'est aussi une classe finale car il n'y a plus de fils derrière elle.

Elle représente le Pion Elephant et a un attribut pts qu'elle récupère de sa classe mère et qui est égal à 5 d'après le constructeur car le Gazelle rapporte 5 points. L'indicateur est aussi initialisé à E pour permettre d'identifier l'Elephant.

Fonctions et Méthodes

Constructeur :

public Elephant()

Classe Plateau

Description

La classe plateau représente le plateau de jeu. Il est représenté par un attribut :

private Case[][] plateau : de dimension 7 lignes et 8 colonnes

Fonctions et Méthodes

Getter :

public Case[][] getPlateau()

Constructeur :

public Plateau()

public void init(int carte)

Cette méthode sert à initialiser le plateau et ses terrains en fonction de la carte choisie

public String toString()

Sert à afficher correctement le plateau en fonction des différents terrains mais aussi en fonction des pions qu'il y a dessus. Utilise la toString() de la classe Animal.

public void premierePosImpala(Jeu jeu1)

Cette méthode sert à poser l'impala avant le 1er tour.

Elle demande la ligne et la colonne où impala doit être posé et change le Pion de la case retenue en « impala »

Nous avons choisi de mettre cette méthode dans la classe Plateau en suivant un point de vue sémantique : cette méthode impact directement le plateau donc nous avons trouvé plus logique de la placer dans Plateau.

public boolean inauguration(int [] coordo,Joueur joueur)

Cette méthode sert à, si le joueur arrive à poser son pion sur la dernière case vide d'un secteur, à lui changer son attribut inauguration (voir classe Joueur) à true .

Pour cela elle parcourt tout le plateau et à chaque case elle regarde si le terrain (voir classe Case) de la case correspond au terrain de la case du joueur passé en paramètre. Elle regarde ensuite s'il y a un pion dessus. S'il n'y en a pas sur un secteur, le secteur n'est pas rempli et le joueur n'a pas le point inauguration

public ArrayList<int[]> verifGazAutourCrocodile(int xCroc, int yCroc)

Méthode qui vérifie s'il y a une Gazelle autour d'un Crocodile au moment où on pose le Crocodile et retourne les cases où se trouvent cette ou ces Gazelles.

public void demanderQuelleGazchasser(ArrayList<int[]> coordoGaz, int xCroc, int yCroc)

Demande à l'utilisateur avec quelle Gazelle il veut échanger son Crocodile et s'il veut le faire. La méthode échange aussi les deux Pions

public void verifGazZebAutourLion(Jeu j1, int indiceJoueur, int xLi, int yLi)

Vérifie au moment où on pose un Lion qu'il n'y a pas un Zebre ou une Gazelle. S'il y en a un, la Gazelle est rendue à son joueur et le Zebre voie son attribut cache à true.

public void verifAnimalAgressif(int xGaZe, int yGaZe)

Sert à, quand on place une Gazelle ou un Zebre à vérifier qu'il n'y a pas un Lion autour d'eux.

S'il y en a un les deux animaux voient leur attribut cache à true.

public int[] trouverImpala()

Sert à trouver l'impala et à le supprimer de sa case afin de pouvoir le placer ailleurs

public boolean verifLigCoVide(int dx, int dy)

Sert à savoir si la ligne ou la colonne correspondant à la case donnée en paramètre contient une case sans pion.

public boolean plateauPlein()

Vérifie si le plateau contient encore une case vide et renvoie false si c'est le cas

public ArrayList<int[]> trouverCoordonneesCasesDispo()

Sert à trouver les coordonnées des cases où le Pion peut être posé.

public void poserPion(Jeu j1, int nbIndiceJoueur, int[] choixJoueur, Animal choixPion)

Permet de poser le pion aux coordonnées choisies par le joueur et de faire les vérifications pour savoir si un pion n'interagit pas avec un autre.

Pour cela on utilise les méthodes verifAnimalAgressif(), verifGazZebAutourLion(), demanderQuelleGazchasser(), verifGazAutourCrocodile().

public void voirPlateau()

Permet de demander au Joueur s'il veut voir le plateau et de le lui afficher

Classe Jeu

Description

Classe servant à commander les fonctions du jeu en lui-même.

Il a comme attributs :

private Plateau plateauDeJeu

private ArrayList<Joueur> listJoueur

Fonctions et Méthodes

Getters :

public ArrayList<Joueur> getListJoueur()

public Plateau getPlateauDejeu()

Setters:

public void setPlateauDejeu(Plateau plateau)

public void init()

Permet d'initialiser le jeu c'est-à-dire : initialiser plateauDeJeu avec sa méthode init(), initialiser les Joueurs et les ajouter à listJoueur avec la méthode init() de la classe Joueur, mélanger la liste de Joueurs et placer l'Impala pour la première fois avec premierePosImpala() de la classe Plateau

public int choixCarte()

Demande au joueur quelle carte il veut choisir

public int[] proposerChoixCase(ArrayList<int[]> tabRep)

Demande à l'utilisateur quelle Case il veut jouer parmi la liste de choix calculés précédemment

public ArrayList<Joueur> compterPoints()

Méthode qui, à la fin de la partie compte le nombre de point de chaque joueur, détermine le ou les vainqueurs et le retourne sous forme de liste

public void jeu()

Méthode qui implémente toute la mécanique du jeu. Elle permet d'initialiser le Jeu avec la méthode init(), puis de, tant que plateau n'est pas plein d'enchainer les tours de jeu. Une fois le plateau plein elle compte les points et annonce le gagnant.

Classe Joueur

Description

Implémente le Joueur avec comme attributs :

private String pseudo

private int nbPoint

private String couleur

private ArrayList<Animal> main

La main du Joueur, l'ensemble des Pions Animaux qu'il a en sa possession

private boolean inauguration

Nous avons choisi de représenter l'inauguration sous forme d'attributs booléens pour faciliter le code et le rendre moins lourd

Fonctions et Méthodes

Getters :

public boolean getInauguration()

public String getPseudo()

public String getCouleur()

public int getNbPoint()

public ArrayList<Animal> getMain()

Setters :

public void setInauguration(boolean a)

public void setNbPoint(int newPts)

Constructeurs:

public Joueur()

public Joueur(String pseudo1, boolean inauguration1, String couleur1)

public String trouverCouleur(ArrayList listJoueur)

Sert à trouver la couleur du Joueur en fonction de sa place dans la listJoueur

public ArrayList<Animal> trouverMainJoueur()

Initialise la main du Joueur en la remplissant avec les Pions demandés dans les règles du jeu

public ArrayList<Animal> trouverPionsAProposer(int[] tab)

Retourne la liste des pions disponibles à jouer dans la main du Joueur

public Animal proposerPion()

Affiche la liste de Pions que le Joueur peut jouer et lui permet de choisir et de le retourner

public void init(String pseudo, ArrayList listJoueur)

Initialise le Joueur en lui donnant un nom, une couleur et une main avec les méthodes trouverCouleur() et trouverMainJoueur().

public String trouverPseudo()

Permet au Joueur de choisir son pseudo

Classe Case

Description

Représente chaque case du plateau avec comme attributs :

private Pion pion

private int terrain

Fonctions et Méthodes

Getters : public Pion getPion()

public int getTerrain()

Setters :

public void setPion(Pion newPion)

public void setTerrain(int newTerrain)

Constructeur :

Classe DroleDeZebre(main)

Description

Lance le programme et la méthode jeu de la classe Jeu.

Classe GraphDroleDeZebre(JFrame)

Description

Classe pour l'interface graphique.

Fonctions et Méthodes

Répartition des tâches

Maé DUGOUA-JACQUES

Jonathan ADAM

Achievements réalisés

Achievement 1

Conclusion et ressentis