

TP6 : La course de tortue

Lien du repository GitHub : <https://github.com/EC-Nantes/tp-note-2-delepine_pourias_guidon.git>

Table des matières

[1.Conception générale 2](#_Toc129798104)

[2.Conception détaillée 4](#_Toc129798105)

[a) Partie : méthode preparationDeLaPartie 4](#_Toc129798106)

[b) Partie : méthode deroulementPartie 5](#_Toc129798107)

[c) Partie : méthode deroulementPartie 6](#_Toc129798108)

# 1.Conception générale

Pour modéliser le jeu de la course de tortue, on a décidé de partir sur 4 classes principales:  
 - Joueur,   
 - Carte,   
 - Partie,  
 - Tortue,   
pour matcher le plus possible avec la réalité du jeu.  
Ces classes répondent aux contraintes suivantes :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Classe |  | | | Fonctions à réaliser | Solution |
| Partie | | FP1 | FP1.1 : Initialiser la partie | | Constructeur de partie  Méthode preparationDeLaPartie Attribut nombre de joueurs  Vecteur de joueurs |
| FP1.2 : Déroulement d’un tour | | Méthode deroulementPartie |
| FP1.3 : Gestion de la pioche | | Vecteur de cartes |
| FP1.4 : Gestion de la défausse | | Vecteur de cartes |
| FP1.5 : Appel de l’effet de la carte | | Méthode deroulementPartie |
| FP1.6 : Fin de la partie | | Méthode finDePartie |
| Joueur | | FP2 | FP2.1 : Associer le joueur à sa tortue (tuile) | | Enum COULEUR  Attribut tuile |
| FP2.2 : Gestion de la main | | Méthode piocher  Vecteur de cartes |
| FP2.3 : Choix de carte à jouer (joueur réel) | | Classe fille joueurRéel.h/.cpp  Méthode afficher  Méthode jouer |
| FP2.4 : Choix aléatoire de la carte à jouer (joueur IA) | | Classe fille joueurIA.h/.cpp  Méthode jouer |
| Carte | | FP3 | FP3.1 : Gérer la couleur des cartes | | Enum COULEUR  Attribut couleurs |
| FP3.1 : Gérer les effets des cartes | | Attribut actions |
| FP3.2 : Gérer la diversité des cartes et de leurs effets | | Classe fille cartes.h/.cpp |
| FP3.4 : Appliquer les effets des cartes | | Méthode getAction |
| Tortue | | FP4 | FP4.1 : Associer la tortue à joueur à sa tortue | | Enum COULEUR  Attribut couleur |
| FP4.2 : Connaitre la position de la tortue | | Attribut positionX  Attribut positionY  Méthode getCoordoX  Méthode getCoordoY |
| FP4.3 : Modifier la position de la tortue | | Attribut positionX  Attribut positionY  Méthode setCoordoX  Méthode setCoordoY |

Une image contenant diagramme

Description générée automatiquement

*Diagramme de classe de la conception du projet*

# 2.Conception détaillée

## a) Partie : méthode preparationDeLaPartie

Variables :

nbJoueurs : entier

vectJoueurs : tableau de joueurs

vectPioche : tableau de cartes

vectDefausse : tableau de cartes

vectTortue : tableau de tortues

Traitement :

fonction preparationDeLaPartie()

Afficher "Préparation de la partie"

Random\_device rd

mt19937 g(rd())

Mélanger les éléments de vectPioche avec shuffle(vectPioche.begin(), vectPioche.end(), g)

Répéter

Afficher "Entrez le nombre de joueurs (entre 2 et 5) : "

Lire nbJoueurs

jusqu'à ce que nbJoueurs soit compris entre 2 et 5 inclus

Afficher "Vous avez choisi " + nbJoueurs + " joueurs."

Pour i allant de 1 à nbJoueurs

vectJoueurs.ajouter(joueur(COULEUR(i)))

fin pour

Mélanger les éléments de vectJoueurs avec shuffle(vectJoueurs.begin(), vectJoueurs.end(), g)

Pour i allant de 1 à nbJoueurs

vectTortue.ajouter(tortue(i))

fin pour

Pour i allant de 0 à nbJoueurs-1

Pour j allant de 0 à 4

Afficher "Le joueur avec la tuile " + COULEUR\_NOM[vectJoueurs[i].getTuile()] + " pioche une carte"

vectJoueurs[i].Piocher(vectPioche.back())

vectPioche.retirer(vectPioche.back())

fin pour

fin pour

Afficher "Partie prête"

fin fonction

## b) Partie : méthode deroulementPartie

Initialisation :

nbTour = 0

i = 0

Traitement :

Afficher "Lancement de la partie"

Tant que checkFinPartie() est vrai

i = nbTour % (nbJoueurs +1)

Afficher "+ Le joueur avec la tuile " + COULEUR\_NOM[vectJoueurs[i].getTuile()] + " joue"

vectDefausse.ajouter(vectJoueurs[i].Jouer())

vectDefausse.back().action()

Afficher "La carte est défaussée"

vectJoueurs[i].Piocher(vectPioche.back())

vectPioche.retirer(vectPioche.back())

nbTour++

fin tant que

Afficher "Fin de la partie"

fin fonction

## c) Partie : méthode deroulementPartie

Traitement :

Fonction checkFinPartie() -> bool

finPartie <- vrai

Pour chaque tortue dans vectTortue

Afficher "La tortue " + COULEUR\_NOM[tortue.couleur] + " est sur la case " + tortue.getPosX() + " en étant à la position " + tortue.getPosY()

Si tortue.getPosX() == 10 et tortue.getPosY() == 0

Afficher "La tortue " + COULEUR\_NOM[tortue.couleur] + " a gagné !!!"

finPartie <- faux

Retourner finPartie

Fin fonction