Rapport sur le projet INFO 2A

Semenov Egor

Mai 2024

Introduction

Dans mon programme, j'ai réussi à implémenter toutes les fonctionnalités décrites dans la première partie, ainsi qu'à écrire des fonctions supplémentaires telles que l'affichage du terrain de jeu dans NotePad++, la sauvegarde des records des précédentes courses, ainsi que la disposition des mines et des outils par probabilités.

1 Description des classes

1.1 Classe Map

La carte est une combinaison de cells.

Les valeurs contenues dans la classe Map

- Tableau bidimensionnel de cells
- Largeur et hauteur de la carte
- Nombre de dépôts d'énergie, de boîtes de grenades, de mines
- Nombre de grenades dans une boîte, quantité d'énergie dans un dépôt
- Tableau de cellules où des objets ont déjà été placés
- Tableau d'outils qui n'ont pas été placés sur la carte

Les méthodes de la classe Map permettent

- Créer une carte
- Redistribuer sur la carte les mines, les outils, les boîtes de grenades, les dépôts d'énergie
- Vérifier si une cellule avec les coordonnées spécifiées existe

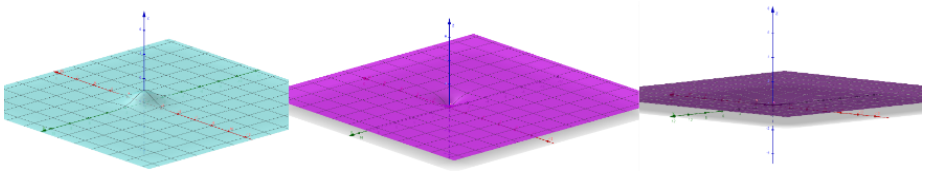

Figure 1: Fonction exponentiel, Fonction $z = \frac{x^2+y^2}{1+x^2+y^2}$, Fonction $z = \frac{x^2+y^2}{10+x^2+y^2}$

- Renvoyer la cellule avec les coordonnées spécifiées
- Afficher la répartition des mines, des outils, des objets collectés
- Afficher la répartition des probabilités de mines et d'outils.

Implémentation de la génération des mines et des outils

Pour générer des mines et des outils sur la carte, j'ai décidé d'utiliser des probabilités. Cette méthode a été choisie car je voulais implémenter une génération non homogène, à savoir que la plupart des mines devraient être situées près du bord de la carte, tandis que la plupart des outils devraient être plus près du centre. Pour atteindre cet effet, la probabilité d'apparition des mines doit augmenter progressivement du centre vers les bords. Sur l'image 1, vous pouvez voir les différents graphiques utilisés, et j'ai choisi la dernière option. Comment ça marche :

1. Lors de la génération d'une cellule, la probabilité d'apparition d'une mine et d'un outil y est créée et enregistrée.
2. Dans le constructeur de la classe Map, la probabilité est normalisée
3. La méthode generateMines() ou similaire est appelée, ensuite son fonctionnement
4. Tout d'abord, la méthode réinitialise les emplacements actuels des mines
5. Tant que le nombre de mines générées n'est pas suffisant, la méthode parcourt le tableau Cells jusqu'à ce qu'elle trouve une cellule où aucun objet n'a encore été placé, génère un nombre et, s'il est inférieur à la probabilité contenue dans la cellule, une mine apparaît dans celle-ci. Pour plus d'aléatoire, une probabilité cumulative a été ajoutée.

1.2 Classe Cell

La classe cell représente une partie constitutive de la carte.

Les valeurs contenues dans la cellule

- Coordonnées sur la carte
- Carte Map
- Objets contenus
- Visibilité propre
- Présence du joueur
- Tableau booléen, responsable de l'intégrité des murs
- Probabilités d'apparition de mines et d'outils dans la cellule

Les méthodes dans la classe Cell permettent

- Renvoyer la cellule dans une direction donnée
- Renvoyer le tableau des cellules adjacentes
- Vérifier s'il y a un mur dans la direction donnée
- Casser un mur
- Réparer un mur
- Vérifier si la cellule est visible actuellement
- Vérifier si la cellule est visible en présence d'un mur dans la direction donnée
- Vérifier si la cellule adjacente dans la direction donnée est visible
- Renvoyer le tableau de chaînes nécessaire à l'affichage de la cellule

Implémentation des methodes les plus complexes

Method breakWall() Dans la classe Cell, les données sur les murs sont stockées sous forme de tableau booléen, où true signifie que le mur est intact et false qu'il est cassé. Voyons comment la méthode casse le mur gauche de cell:

1. Tout d'abord, elle remplace la valeur du mur gauche dans le tableau par false
2. Ensuite, elle s'appelle dans la cellule à gauche, s'il y en a une, qui casse le mur droit
3. Elle définit sa visibilité sur true

Methode getLayers() J'ai misé sur une interface utilisateur agréable, ce qui m'a amené à écrire une fonction répondant à mes besoins élevés.

J'ai commencé par définir le rapport largeur-hauteur de la future cellule, en partant de la police de caractères, et j'en ai conclu qu'il était de 5 à 9.

J'ai décidé que le résultat de cette fonction serait un tableau String[] dans lequel chaque élément serait une ligne de la salle.

La particularité de la méthode que j'ai choisie est que chaque cellule n'affiche que deux murs sur quatre, ce qui résout le problème des murs doubles.

Comment fonctionne la méthode : En cas de visibilité de la salle :

1. Construction de la couche supérieure, en tenant compte de la présence d'un mur en haut
2. Construction des trois couches du milieu, à l'aide de la méthode getContains() qui renvoie le contenu actuel de la salle
3. En cas du salle d'extrémité, une ligne et une colonne de bordure supplémentaires sont ajoutées

En cas d'invisibilité de la salle :

1. En tenant compte de la visibilité de la cellule supérieure, affichage de la ligne supérieure
2. Le contenu de la salle est remplacé par le symbole "full block"
3. En cas de cellule d'extrémité, une ligne et une colonne de bordure supplémentaires sont ajoutées

Problème des coins : J'ai consacré beaucoup de temps au problème des coins. Pour afficher une salle, j'ai utilisé des symboles Unicode. Le choix du symbole de coin dépend de la visibilité des salles adjacentes.

Pour résoudre ce problème, lors de la création de la chaîne, le paramètre de visibilité des cellules à gauche, en haut et en diagonale est pris en compte. Une fonction surchargée a également été créée, qui permet d'afficher la carte entière sans tenir compte de la visibilité.

1.3 Classe Player

La classe qui décrit le joueur, permettant au joueur d'effectuer des actions.

Les valeurs contenues dans Player

- Nom
- Carte actuelle
- Représentation de la position actuelle
- Réserve actuelle d'énergie et de grenades

- Réserve maximale de grenades et d'énergie
- Tableau d'outils disponibles
- Valeurs booléennes indiquant la mort (défaite) et la victoire
- Valeurs booléennes indiquant les possibilités du joueur de prendre une grenade, de l'énergie, des outils

Les méthodes dans la cellule permettent

- Se déplacer
- Lancer une grenade
- Prendre de l'énergie, une grenade, un outil
- Utiliser un outil

Implémentation des fonctions les plus complexes

useTool()

Cette fonction permet d'utiliser les outils.

La difficulté réside dans le fait que initialement tous les outils sont créés comme une instance de la classe ExtraTool (classe générale de tous les outils), cependant chaque outil nécessite des informations différentes pour fonctionner.

Comment fonctionne la méthode :

1. Détermination de la nécessité d'une direction pour le fonctionnement de l'outil sélectionné
2. Appel de la fonction ou de sa version surchargée (si une direction est nécessaire pour fonctionner)
3. Reformatage de l'élément requis du tableau en une classe spécifique de cet outil
4. Appel de la fonction useTool() pour l'outil sélectionné
5. Retour de la valeur String obtenue en résultat

1.4 Classe ExtraTools et ses classes dérivées

Extra Tools est la superclasse de tous les outils. Cette structure a été choisie car il était nécessaire de créer un tableau de tous les outils pour les placer correctement et les utiliser ultérieurement.

Classes héritées de ExtraTools et leur représentation sur la carte

- Discharge Unit (Du)
- Excavator (Ex)
- LRMD (Long-range mass detector) (Lm)
- LRUN (Long-range unidirectional scanner) (Lu)
- Mine detector (Md)
- ShRos (Short-range omnidirectional scanner) (Sh)

Chaque classe héritée permet d'utiliser l'outil qu'elle représente à l'aide de la fonction useTool() unique.

1.5 Classe Mine

Représente une mine et son état, avec une fonction de désamorçage.

1.6 Classe EnergySource, GrenadeSource

EnergySource représente un réservoir d'énergie. Il contient la quantité d'énergie restante. Permet au joueur de récupérer de l'énergie, en totalité ou en partie. (Noté sur la carte E)

GrenadeSource représente une boîte de grenades. Il contient la quantité de grenades restante. Permet au joueur de prendre des grenades, en totalité ou en partie. (Noté sur la carte G)

1.7 Classe Game

Lien entre toutes les parties du jeu.

Les valeurs contenues dans la classe Game

- Carte Map
- Joueur Player
- Valeur d'erreur

Méthodes de la classe Game

- play : boucle principale du jeu.
- testMode : permet de parcourir la carte sans mourir. Les murs sont automatiquement retirés.
- getDirection : fonction de demande de direction.

Description du fonctionnement de la méthode play

Cette méthode est la principale du programme, c'est par elle que le jeu est lancé. La méthode prend en entrée une valeur booléenne indiquant l'endroit où afficher. Le temps de début de jeu est enregistré. La boucle principale s'exécute tant que le joueur est en vie et n'a pas gagné.

1. Génération des options d'action possibles à ce moment-là.
2. Le joueur entre l'action souhaitée soit par texte, soit par indice.
3. La méthode de la classe Player ou Output, exécutant l'action, est appelée.
4. Le terrain est affiché après la mise à jour.

Après la fin du jeu, un tableau des scores avec les temps des jeux précédents est affiché à l'écran.

1.8 Classe Output

Permet de visualiser le terrain, de jouer des animations et gère l'affichage dans Notepad++. C'est la classe la plus complexe du programme.

Les valeurs contenues dans la classe Output

- Chemins des fichiers de stockage et de sortie
- Jeu à afficher

Méthodes de la classe Output

- outputMap affiche la carte dans Notepad++ en utilisant generateArray, toFinalString et output
- outputMapToConsole fait la même chose que outputMap mais affiche la carte dans la console
- generateArray crée un tableau à deux dimensions pour afficher la carte
- toFinalString convertit un tableau à deux dimensions en une seule chaîne de caractères pour l'affichage
- output affiche une chaîne dans un fichier txt ouvert dans Notepad++
- saveScore sauvegarde le score dans le fichier Score.csv
- getPreviousScore extrait les scores précédents du fichier Score.csv Liées aux animations
- readAniamtion permet de lire les lignes du fichier Animation.txt

- extract permet d'extraire une cellule du tableau généré par generateArray en fonction de ses coordonnées.
- replace permet de remplacer une partie du tableau à deux dimensions par un autre tableau à deux dimensions.
- replaceCorners remplace certains caractères par d'autres
- copyArray crée une copie indépendante du tableau à deux dimensions
- toOtherForm divise toutes les lignes du tableau à deux dimensions en tableaux de caractères
- animationBreak détermine quelle animation doit être affichée
- animationBreakDiscover animation de la découverte d'une nouvelle pièce
- animationBreakWall animation du retrait du mur entre deux cellules ouvertes
- animationKab00m animation de l'explosion d'une mine

Description d'une des méthodes d'animation (animationBreakWall)

Cette méthode est appelée depuis la méthode animationBreak, qui détermine quelle animation doit être appelée en fonction de la position actuelle du joueur et de la direction de la dernière action effectuée. Algorithme de la méthode :

1. Reconstruction du mur qui vient d'être cassé afin de créer un tableau où la cellule n'est pas encore visible et ainsi résoudre le problème des coins.
2. Création d'un tableau de la carte avec les cellules visibles à ce moment-là
3. Extraction de la cellule sur laquelle l'action est effectuée
4. Sauvegarde d'une copie du tableau pour une utilisation future
5. Suppression du mur
6. Lecture du fichier Animation.txt, contenant les animations par trame ; lors de la lecture, un tableau tridimensionnel est retourné, chaque élément du tableau étant une trame.
7. Chaque trame remplace successivement une partie du tableau de la cellule extraite
8. Le tableau passe par la procédure de remplacement de caractères.Pour faire défiler correctement l'animation, il est nécessaire de changer certains caractères en fonction de l'environnement de la cellule, certains avant la destruction du mur et d'autres après, par exemple pour gérer les coins. Pour cela, la fonction replaceCorners() a été créée pour remplacer les caractères # par ceux présents dans la cellule avant la destruction du mur, et les caractères @ par ceux présents après la destruction du mur.

9. Le tableau modifié après l'étape 7 et modifié après l'étape 8 remplace enfin la cellule du tableau formé avant la destruction du mur.

10. Attente de 120 millisecondes.

1.9 Quelques mots supplémentaires sur les stratégies choisies

Je n'ai pas créé de classe direction car je n'ai pas trouvé de méthodes qui pourraient être mises en œuvre dans une telle classe, donc dans mon programme, les fonctions de la classe direction sont remplies par des chaînes indiquant la direction ("haut", "bas", "gauche", "droite").

Si un résultat de méthode doit être affiché à l'écran, la méthode renvoie une chaîne de caractères, bien que je reconnaisse qu'il aurait été judicieux d'attribuer à de telles classes un attribut replyString qui stockerait le résultat des méthodes, puis serait demandé là où il est nécessaire.

Certaines erreurs sont également renvoyées sous forme de chaînes de caractères, bien qu'elles auraient dû être gérées avec try catch. Cependant, quand j'ai réalisé cela, le programme était déjà largement écrit et réécrire avec try catch aurait nécessité beaucoup de travail, à la fois en termes de programmation et de conception conceptuelle.