Logique de développement

1. Analyse du besoin

Le programme doit résoudre le problème proposé et être appelé en ligne de commande. Un document annexe est fourni pour expliquer l'utilisation du programme.

2. Architecture

1) Stratégie de développement :

Hypothèses retenues non précisées dans l'énoncé :

- La grille en entrée doit être de forme rectangulaire
- Si le feu atteint une sortie en même temps que le personnage celle-ci ne peut pas être utilisée

Stratégie :

Le programme se base sur une matrice correspondant à la grille fournie. La logique est de faire se propager le feu et les cases atteignables par le personnage à chaque itération (ainsi que les trajets utilisés pour les atteindre).

À chaque tour on partira des cases atteintes au tour précédent (le 1er tour est la case « S »). Pour chacune de ces cases, on regarde les cases atteignables (sans feu ni arbre) qui n'ont pas encore été parcourues. On complète les cases atteignables avec l'information du trajet parcouru pour y arriver. Évidemment il y a plusieurs façons d'atteindre une case : en partant d'une case, aller en bas puis à droite revient à la même chose en termes de nombre d'itérations que d'aller à droite puis en bas (s'il n'y a pas d'obstacle). Le trajet dépendra de la façon dont on parcourt la matrice dans notre algorithme.

2) Architecture fonctionnelle

a) Vue logique :

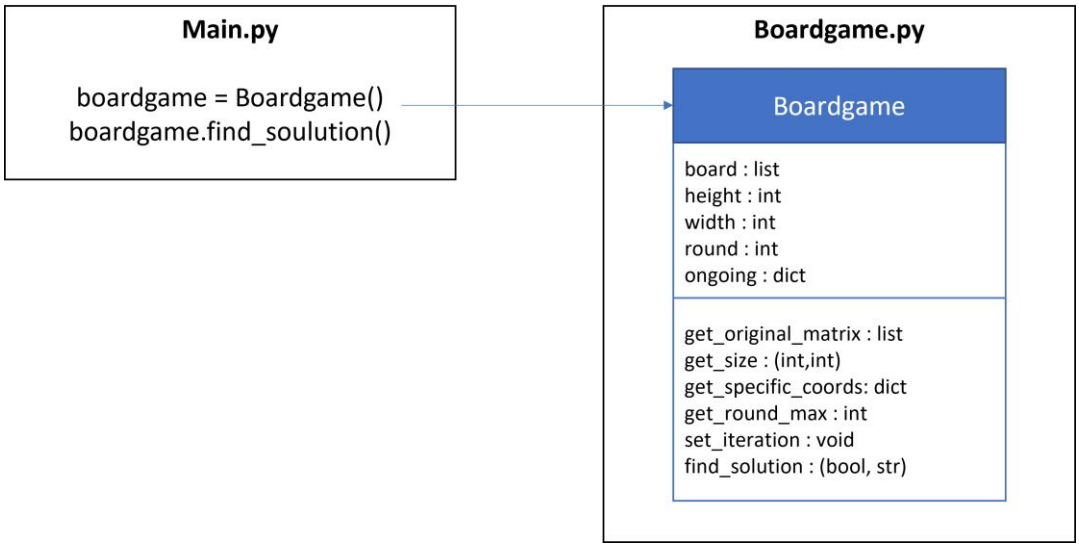

Figure 1 - Logique du programme

b) Vue de processus :

Il n'y a pas de processus/thread qui fonctionnent en parallèle. Le script principal récupère l'argument fourni dans le terminal de commande, crée une instance de la classe « Boardgame », lance la méthode « find_soulution » et renvoi le résultat.

3) Architecture technique

a) Vue de réalisation :

Environnement d'exécution : serveur Linux

Source de données : fichier texte

b) Contraintes de réalisation :

Temps d'exécution : le temps d'exécution sur les exemples fournis ne doit pas excéder quelques secondes

3. Plan de validation

Pour vérifier le bon fonctionnement du script, j'ai exécuté le code sur les 4 exemples fournis et vérifié que les solutions proposées fonctionnaient.

J'ai ensuite testé avec des grilles modifiées pour vérifier le comportement du code dans des cas limites : pas de sorties, sorties inatteignables, le feu atteint la sortie avant le personnage, format de grille non conforme.

De plus, pour faciliter le développement et identifier de potentielles erreurs, j'ai créé une méthode pour afficher la matrice à chaque itération. Cette méthode a été mise en commentaire et n'est pas utilisée dans la version fournie :

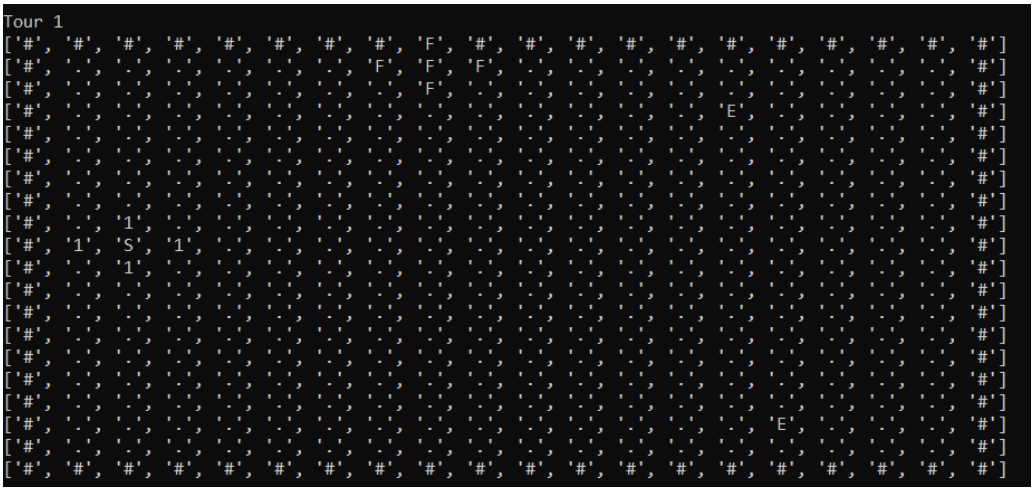

Figure 2- Matrice après une itération

```
Tour 2
['#', '#', '#', '#', '#', '#', '#', 'F', 'F', 'F', '#', '#', '#', '#', '#', '#', '#', '#', '#', '#']
['#', '.', '.', '.', '.', '.', 'F', 'F', 'F', 'F', 'F', '.', '.', '.', '.', '.', '.', '.', '.', '#']
['#', '.', '.', '.', '.', '.', '.', 'F', 'F', 'F', '.', '.', '.', '.', '.', '.', '.', '.', '.', '#']
['#', '.', '.', '.', '.', '.', '.', '.', 'F', '.', '.', '.', '.', '.', 'E', '.', '.', '.', '.', '#']
['#', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '#']
['#', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '#']
['#', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '#']
['#', '.', '2', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '#']
['#', '2', '1', '2', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '#']
['#', '1', 'S', '1', '2', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '#']
['#', '2', '1', '2', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '#']
['#', '.', '2', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '#']
['#', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '#']
['#', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '#']
['#', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '#']
['#', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '#']
['#', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '#']
['#', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', 'E', '.', '.', '.', '#']
['#', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '.', '#']
['#', '#', '#', '#', '#', '#', '#', '#', '#', '#', '#', '#', '#', '#', '#', '#', '#', '#', '#', '#']
```

Figure 3 - Matrice après 2 itérations

```
Tour 21
['F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F']
['F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F']
['F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F']
['F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F']
['F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F']
['F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F']
['F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F']
['F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F']
['F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F']
['F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F']
['F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F']
['F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F']
['F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', '#']
['F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', '20', '#']
['F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', '20', '21', '#']
['#', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', '20', '21', '.', '#']
['#', '8', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', '20', '21', '.', '.', '#']
['#', '9', '8', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', '20', 'E', '.', '.', '.', '#']
['#', '10', '9', '10', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', 'F', '20', '21', '.', '.', '.', '.', '#']
['#', '#', '#', '#', '#', 'F', 'F', 'F', 'F', 'F', 'F', 'F', '#', '#', '#', '#', '#', '#', '#', '#']
RRRRRRRRRRRRDDDDDDDDR
```

Figure 4 - Matrice à la dernière itération

Les nombres dans les cases correspondent aux nombres d'itérations nécessaires pour atteindre lesdites cases. Un autre tableau contient les chemins pour chacune des cases de la dernière itération.