Projet Simulateur de déplacement de foule

L'objectif du projet est de simuler le déplacement d'une foule à l'intérieur d'un terrain prédéfini. Le terrain contient des zones de déplacement autorisées, des zones de déplacements interdits ainsi que des points d'apparition des personnages constituant la foule et des objectifs de destination.

Le terrain est construit à partir d'un fichier texte qui suit la structure suivante :

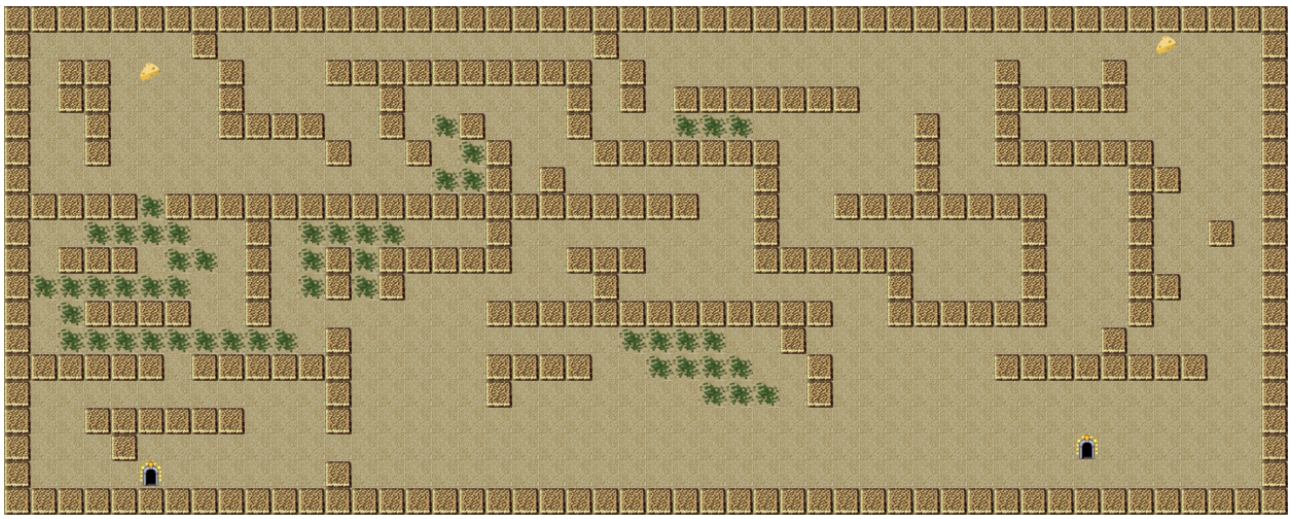
```
**************************************************
*       *                 *                    A   *
* ** A  *    ********** *              *    *      *
* **     *      *       * * *******      *****      *
*  *     ****   * G*    *    GGG       *   *         *
*  *          *   * G*    *******      *   ******    *
*                 GG* *         *      *        **   *
*****G*********************   *   ********    *     *
*  GGGG   * GGGG    *           *           *    *   * *
* *** GG * G*G*****   ***     ******     *    *     *
*GGGGGG   * G*G*         *             *     *    **   *
* G****   *          *************   ******    *     *
* GGGGGGGGG *    P       GGGG  *             *      *
****** ******      ****  GGGG  *       ********   *
*             *      *       GGG *                 *
*   ******    *                                    *
*    *              P                        D      *
*     D       *                                    *
**************************************************
```

Sémantique des symboles :

- * : un mur
- G : une zone d'herbe
- ' ' : une zone de déplacement
- D : les points d'apparition des personnages
- A : les points d'arrivée des personnages.

Le terrain doit obligatoirement être fermé (entouré de mur)

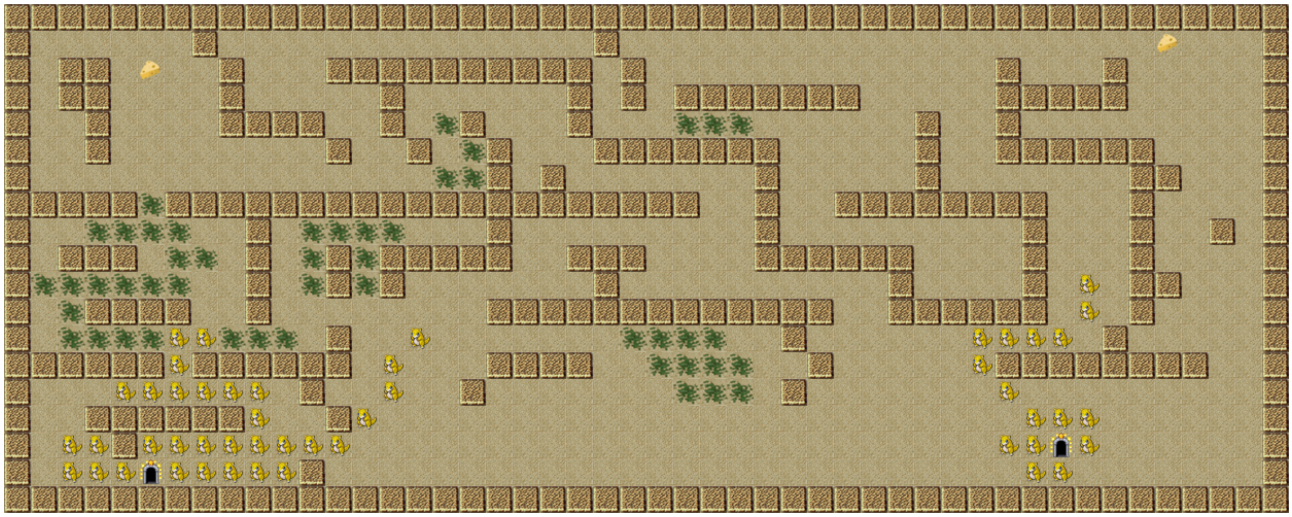
En utilisant le kit de sprites fourni, on obtient la réprésentation suivante :



Les portes représentent les points de départ, les fromages les objectifs de destination.

Les zones d'herbe nécessitent 2 unités de temps pour un déplacement, alors que les zones standards, 1 seule unité.

Les personnes constituant la foule sont symbolisées par des souris. Lors de leurs déplacements, les souris ne peuvent pas passer simultanément sur la même case, de même, elles de peuvent pas passer par dessus. Si une zone de passage est embouteillée, elle doivent attendre leur tour ou chercher un autre itinéraire plus rapide.



Les souris apparaissent à un endroit déterminée, mais elles choisissent librement leur destination.

Elles ne peuvent apparaître que sur une case contiguë à la porte. Si aucune case n'est disponible, elles doivent attendre qu'une se libère.

Votre objectif est que l'ensemble des souris rejoignent les gruyères en un nombre de tour minimum.

Le tour est l'unité de déplacement par défaut (un tour), pour la simulation, l'unité de temps est fixé à 1 seconde, mais devra être facilement paramétrable pendant la soutenance.

A chaque tour, tant qu'il y a des souris à l'intérieur de la porte, elles doivent apparaître sur la carte.

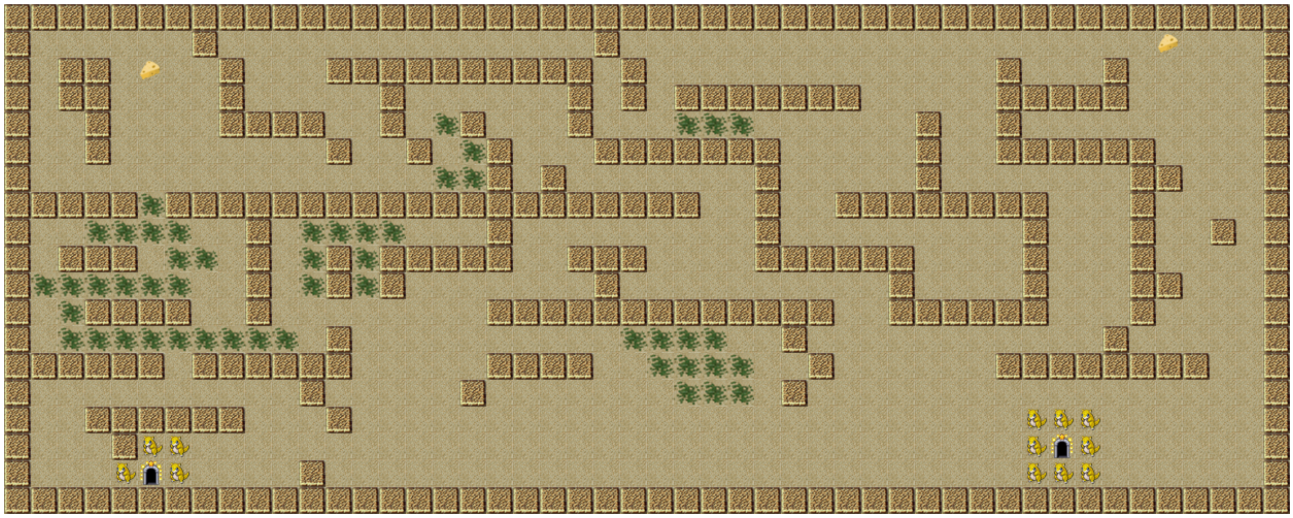
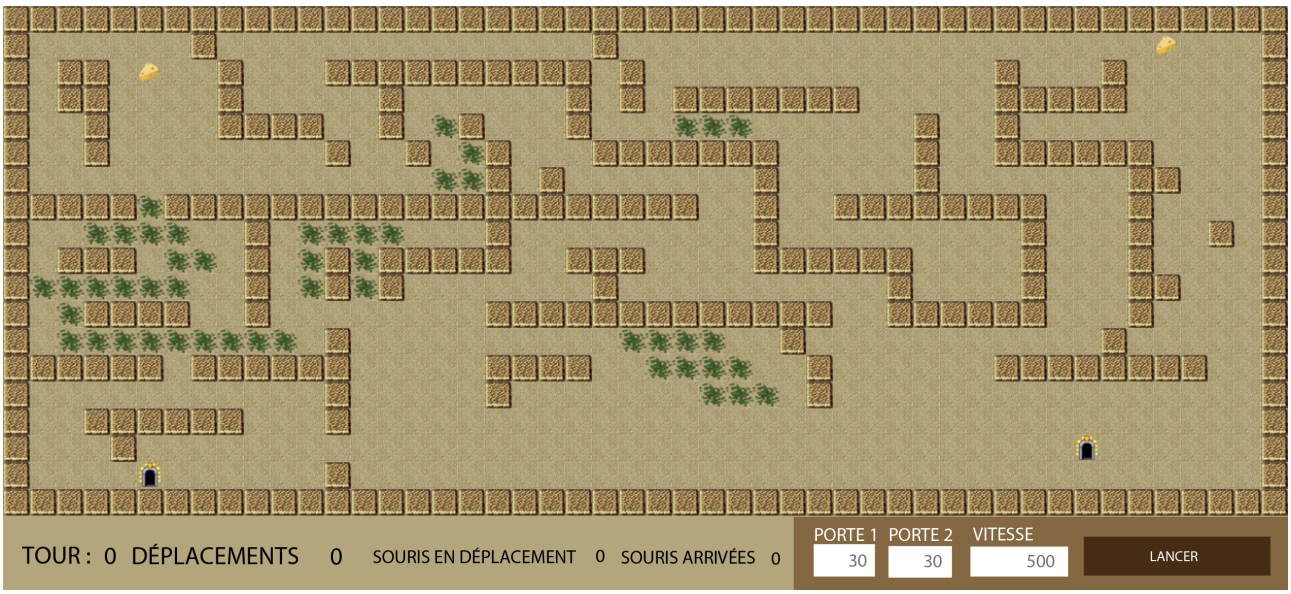


Illustration 1: Exemple de l'état initial de la carte après une unité de temps

Le nombre de souris pouvant apparaître à chaque porte devra pouvoir être déterminé dans le programme que vous avez à réaliser. Le programme pourra prendre par exemple la forme suivante :



où :

- TOUR indique le tour actuel.
- DÉPLACEMENTS, le nombre cumulés de tours réalisés par chaque souris pour atteindre l'objectif.
- SOURIS EN DÉPLACEMENT : Le nombre de souris actuellement sur le terrain.
- SOURIS ARRIVÉES : Le nombre de souris ayant bien rejoint leur destination.
- PORTE 1, PORTE n : Le nombre de souris qui sortiront à chaque porte. Une fois la simulation lancée, le nombre de souris devra décrémenter à mesure qu'elles apparaitront sur le plateau.
- VITESSE : la vitesse,exprimée en milliseconde, de chaque tour.
- LANCER : Le bouton qui permettra de lancer la simulation.

Travail attendu :

Vous devrez implémenter un programme permettant de réaliser cette simulation.

Vous pouvez choisir le langage d'implémentation parmi : le C, le javascript et le Java.

Vous n'avez pas le droit d'utiliser de librairies externes hormis les librairies d'affichage graphique. Les structures de données telles que les piles, les files, les listes chainées, les graphes devront être issues de votre travail. Vous pouvez néanmoins utiliser les implémentations fournies par les langages pour les ArrayList et les HashMap.

Concernant le calcul d'itinéraire, vous devrez implémenter une solution à partir d'un graphe et utiliser l'algorithme de Djikstra ou A*. Les optimisations tels que les calculs bidirectionnels seront les bienvenues.

Le programme devra permettre de charge n'importe quelle carte respectant le format défini.

Bon courage à tous.