

**RAPPORT**

**LABYRINTHE HANTE**

**Clément SAUVARD**

**&**

**Joé DE OLIVEIRA**

Fonctionnalités implémentées et fonctionnelles

* Mode autonome
* Chargement dynamique du labyrinthe
* Positionnement correct des éléments selon le fichier .txt (Chasseur, gardiens, affiches, trésor, murs)
* Affichage dynamique des affiches (liaison dynamique entre le caractère de l’affiche et le chemin de la texture)
* Gestion des collisions (ex : les murs ne peuvent pas être traversés, que ce soit pour le chasseur ou les gardiens, la boule de feu est arrêté par les obstacles).
* Algorithme de Dikjstra pour le plus court chemin
* Mode attaque et mode défense pour les gardiens
* Gestion de la vie (Un chasseur a un capital de vie, le gardien aussi).
* Processus de régénération de la vie pour les gardiens et le chasseur (s’il manque de la vie au chasseur/gardien et qu’il n’a pas été touché récemment, il gagne un point de vie au bout de quelques secondes)
* Le gardien perd de la précision de tir quand il est blessé. Le chasseur aussi.
* Le chasseur peut tuer les gardiens
* Les gardiens peuvent tuer le chasseur, si le chasseur meurt, game over
* Gestion des collisions entre la boule et un élément (on sait lorsque la boule touche un mur, un gardien, une caisse, le trésor, ou est dans le vide)
* Fin du jeu lorsque le chasseur tir sur le trésor

Fonctionnalités non implémentées

* Jeu en réseau

Fonctionnement du mode attaque défense :

Voici comment nous avons implémenté le mode attaque : on crée un tableau résultant de notre algorithme de dikjstra, (qui contient donc une numérotation représentative du plus cours chemin) qui fabrique un chemin plus court non pas vers le trésor mais vers un point aléatoire du labyrinthe (en vérifiant que ce point ne soit pas un obstacle), le gardien va donc s’y diriger et une fois qu'il dépassera un certain seuil (c’est-à-dire, si la valeur de la case où il se trouve dans le tableau de dikjstra est trop grande en l’occurrence) il revient en mode défense et se rapproche du trésor, en utilisant cette fois ci le tableau de dikjstra concernant le trésor. Ainsi, une fois qu’il se rapproche trop du trésor, il repasse en mode attaque et un nouveau point aléatoire dans le labyrinthe est désigné et il doit s’y diriger.

A savoir : le code utilise c++ 14 notamment pour la fonctionnalité « auto » (déduction de type)