POLYTECH LILLE

Université de Lille

# Jeu de Cavalier

Projet de programmation structurée IMA3 2018-2019

## Objectifs

Ce projet de programmation structurée, a pour objectif de réaliser en binôme, un premier logiciel avec les notions acquises au S5 (et **uniquement** celles-ci).

Dans ce projet, vous mettrez en oeuvre les notions vues en cours de Programmation Structurée (conception, algorithmique, développement, critiques et documentation). Ce projet est à rendre à votre responsable de TP pour le **VENDREDI 11 janvier 18H** (attention, délai de rigueur, pénalités de 5 points pour les retardataires pour chaque tranche de 24h de retard, une tranche entamée étant comptée comme complète), par email (à l'adresse de votre encadrant de TP), dans une archive comportant votre code, un makefile utilisable sur vos machines de TP, et un rapport au format PDF (5 pages maximum, clair et précis donnant notamment explicitement les limitations de votre solution).

**IMPORTANT**: nous disposons, tout comme vous, d'internet et des diverses sources d'informations disponibles. Nous disposons également d'une expérience et d'outils qui font que le plus souvent nous retrouvons facilement vos sources d'inspirations. Nous répétons ici l'importance d'un travail autonome et honnête. Import, maquillage, reprise (simplifiée ou pas), travail communautaires, ou tout autre variation sur le même thème ne font pas partie de ce qui est demandé ici, et seront donc notés suivant des critères tout à faire spécifiques et bien plus durs que ceux utilisés normalement.

## Descriptif global du sujet

Le jeu de chevalier est un jeu de grille (fig. 1) où le but est de contrôler un cavalier (**C**) pour arriver à la cible (**T**) avec moins d'itérations possible. Sur la grille, il se trouve également des obstacles (**X**), des bonus (**B**).

Comme dans le jeu d'échecs, le cavalier ne se déplace qu'en **L**, c'est-à-dire de deux cases dans une direction combinée avec une case perpendiculaire. Par conséquent, le cavalier peut déplacer dans 8 cases différentes si elles ne sont pas occupées par un obstacle. S'il arrive sur une case bonus, le cavalier active un pouvoir (voir 4).

Le cavalier a un champ de vue limité, il ne voit que le contenu des cases à sa proximité. Au début du jeu, une zone de 5x5 cases est visible par le cavalier.

## 1 Initialisation

Votre programme permettra d'initialiser une grille de taille 16x16 qui contient 1 chevalier, 8 obstacles, 2 bonus et 1 cible.

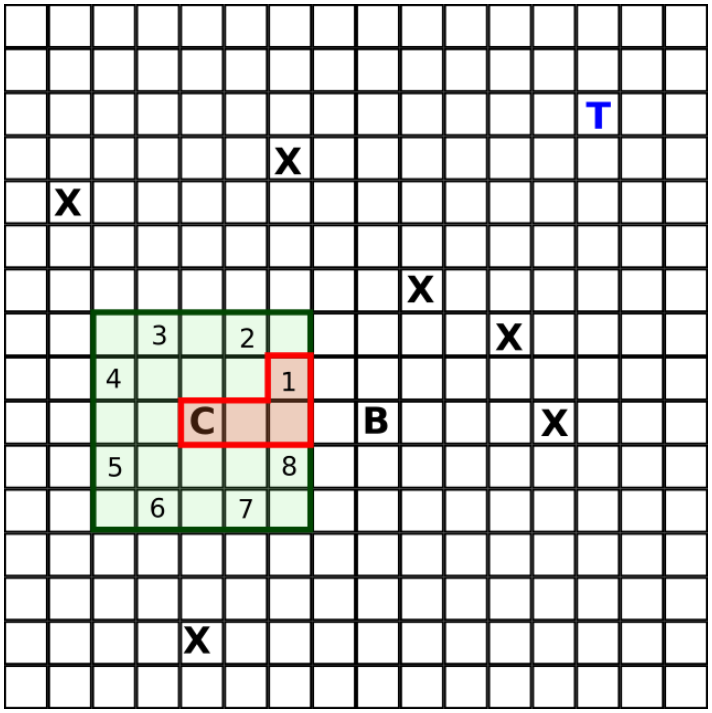


Figure 1: La grille du jeu.

**C**: Cavalier

**X**: Obstacle

**B**: Bonus

**T**: Cible

Carré vert: Champ de vue

[1....8]: Déplacements possibles

## 2 Affichage

On mettra en place l'affichage en prenant soin d'afficher la grille de manière stable et compréhensible. A chaque itération du jeu, il faut afficher le contenu des cases visibles et les numéros des cases dans lesquels le cavalier peut se déplacer.

On pourra se servir de l'appel *system ("clear");* (effectuant un effacement de la fenêtre shell où vous avez lancé votre executable).

## 3 Diriger le cavalier

Entre chaque itération du jeu, l'utilisateur doit saisir un chiffre entre 1-8 pour diriger le cavalier. S'il y a un obstacle dans la case visée, l'utilisateur gaspille un mouvement et reste dans sa position courante.

Il faudra mettre en place la procédure prenant en argument un tableau à deux dimensions (le support du jeu) et le modifiant pour qu'il corresponde à la fin de l'appel de cette procédure quand le cavalier est déplacé.

## 4 Cases bonus/malus

Il existe trois types de bonus:

**Bonus vision:** Si le cavalier arrive sur la case d'un bonus, son champ de vision augmente d'une case (6x6, 7x7). **Warp:** Repositionne le cavalier aléatoirement. **Duplicata:** Multiplie le nombre de

cibles par 2.

Vous pouvez choisir un caractère qui correspond à chaque bonus.

## 5 Fin du jeu

Le jeu termine quand le cavalier arrive sur la case de la cible. Le nombre de mouvements est affiché comme le score du joueur.

A tout moment du jeu, l'utilisateur peut appuyer sur la touche 'R' pour réinitialiser (aléatoirement) la grille.

## 6 Une variante périodique

On proposera une variante de l'implémention mise en place permettant d'utiliser la grille de manière périodique (i.e. faire en sorte que les cellules de la dernière ligne soient considérées comme voisine de celles du haut et vice-versa, que celles de la colonne de gauche soient considérées comme voisines de celles de droite et vice-versa).

## 7 Difficulté du jeu

On fera ici en sorte de permettre à l'utilisateur de donner un pourcentage de difficulté, en fonction duquel la grille est remplie du nombre de bonus et d'obstacles, mis à des positions aléatoires.

## 8 BONUS: Malus

Vous pouvez imaginer et ajouter des malus. A chaque tour les malus peuvent se rapprocher d'une case en direction du cavalier (sinon un joueur ne les prendra jamais).

## 9 BONUS: Cible mobile

On fera ici de sorte que la cible bouge aléatoirement d'une case à chaque itération du jeu (horizontalement ou verticalement). Pour attraper la cible, il faut deviner son mouvement.