ENSICAEN
ÉCOLE NATIONALE SUPÉRIEURE D'INGÉNIEURS DE CAEN
& CENTRE DE RECHERCHE

INTRODUCTION A LA PROGRAMMATION

Projet de programmation

Jeu du labyrinthe

Version 1.0, 18/10/2012

ECOLE NATIONALE SUPERIEURE D'INGENIEURS DE CAEN
6, boulevard Maréchal Juin - 14050 Caen Cedex - Tél. +33 (0)2 31 45 27 50 – Fax +33 (0)2 31 45 27 60 – www.ensicaen.fr

1 Présentation du projet

Le but du projet est la réalisation d'un générateur de labyrinthe, et de son utilisation dans un jeu. Ce projet ce fera en quatre étapes et en monôme. Chaque étape correspond à un niveau de difficulté et correspond à une note maximale de projet. Le passage à une étape supérieure ne se fait qu'après avoir été validé par l'enseignant de TP.

Une étape est validée lorsque toutes les exigences et fonctionnalités demandées on été prises en compte. La validation d'une étape n'assure pas une note maximale, même si elle s'en rapproche. Les critères de notation sont les suivants :

- Lisibilité du code : respect des consignes du cours d'ODL ;
- Respect des consignes de l'étape du projet ;
- Optimisation du code, les choix de conception ;
- Ergonomie.

Le rendu du projet sera :

- Une archive contenant le code source de l'application,
- Un compte-rendu présentant votre projet :
  - ce qui a été réalisé,
  - le fonctionnement du point de vue de l'utilisateur (du joueur),
  - le fonctionnement des points clés de votre projet, du point de vue du programmeur,
  - les limitations/bugs connus,
  - …

1.1 Etape 1 /10 points

Le but est de réaliser un générateur simple de labyrinthe de taille 9*23. L'algorithme de génération vous est fourni en annexe. L'affichage sera fait en console, tel que présenté à la Figure 1. Les symboles # représentent les murs du labyrinthe, l'entrée est en haut à gauche et la sortie en bas à droite. Le personnage à déplacer est représenté par la lettre o.

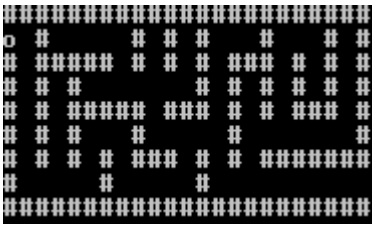
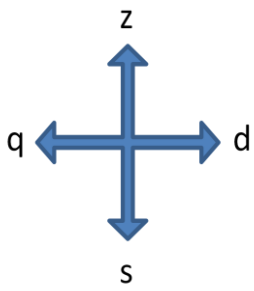


Figure 1 Rendu d'un labyrinthe en mode console

L'utilisateur peut se déplacer dans les couloirs du labyrinthe avec l'aide des touches zqsd suivie de la touche entrée :



Lorsque le joueur arrive à la fin du labyrinthe, le programme lui propose de jouer une nouvelle partie, et quitte le programme si celui refuse.

1.2 Etape 2 /14 points

Deux sortes d'objets sont éparpillées de façon aléatoire dans le labyrinthe :

- Des objets bonus (trésors), accordant un certain nombre de points fixes : les cadeaux d'une valeur de X points.
- Des objets malus (pièges), retirant un certain nombre de points fixes : les destructeurs d'une valeur de Y points.

Un score sera attribué au joueur qui sera fonction de sa rapidité à trouver la sortie et des bonus et/ou malus rencontrées lors de son parcours. Le score est affiché à chaque pas et en fin de partie.

Au démarrage du programme, un menu permet à l'utilisateur de :

- Créer un labyrinthe,
- Charger un labyrinthe,
- Jouer,
- Quitter.

Lors de la création du labyrinthe, l'utilisateur pourra choisir la taille du labyrinthe (hauteur et largeur impaire) ainsi que son nom. Le programme généré un nouveau labyrinthe et différentes sortes de bonus/malus sont créées et réparties au hasard dans le labyrinthe. Le labyrinthe est enregistré dans un fichier au format .init.

Lors du chargement, l'utilisateur indique le nom du labyrinthe auquel il souhaite jouer. Le programme doit alors charger un fichier qui recharge un précédent labyrinthe et ses malus/bonus (d'extension .init).

Lorsque l'utilisateur décide de jouer, il joue sa partie comme dans l'étape précédente. Si son score fait partie des 10 meilleurs déjà enregistrés, le programme l'invite à saisir son nom et l'enregistre dans un fichier d'extension .cfg et au nom du labyrinthe.

1.3 Etape 3 /17 points

Le menu propose maintenant d'afficher la solution d'un labyrinthe. Pour cela, le programme doit être capable de fournir un itinéraire permettant de sortir du labyrinthe suivant le principe de la 'main droite'.

Différentes sortes de monstres jalonnent le parcours. Chaque sorte est régie par 2 paramètres :

- Paramètre de mobilité : La mobilité est définie suivant une règle régie par une fonction. Par exemple, des fantômes pourront traverser les murs, tandis que des ogres resteront à proximité des trésors qu'ils croiseront.
- Paramètre d'étendue de mobilité : Le territoire de mobilité est d'autant plus grand que leur nombre de points de pénalité est grand.

A chaque pas du parcours du joueur, tous les monstres bougent également suivant leurs propres paramètres. Vous utiliserez pour cela des pointeurs vers des fonctions.

1.4 Etape 4 /20 points

Le programme doit fonctionner aussi bien sous Windows que sous Linux.

Le programme devra être accompagné d'un jeu de tests unitaires réalisés sous Cunit.

Développement d'une IHM en SDL. Pour créer un projet SDL, il faut :

- Installer la librairie SDL pour Mingwin qui se trouve ici : http://www.libsdl.org/download-1.2.php.
- Dezipper le package obtenu dans un répertoire sous ClodeBlocks.
- Créer un nouveau projet de type SDL.
- Indiquez le dossier où vous avez décompressé la SDL dans la section Base lors du processus d'installation du nouveau projet, au moment où on vous invite à référencer la librairie.

2 Annexe

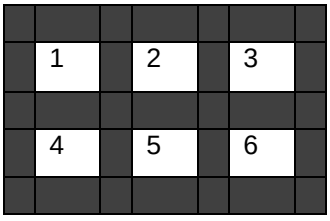
Source :
http://fr.wikipedia.org/wiki/Mod%C3%A9lisation_math%C3%A9matique_d%27un_labyrinthe

L'algorithme utilise une propriété des labyrinthes *parfaits* tel quel *chaque cellule est reliée à toutes les autres et, ce, de manière unique.* Il fonctionne en fusionnant progressivement des chemins depuis la simple cellule jusqu'à l'obtention d'un chemin unique, il suit donc une approche ascendante (bottom-up).

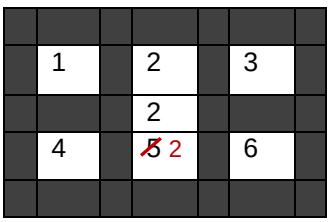
L'algorithme associe une valeur unique à chaque cellule (un numéro, par exemple) et part d'un labyrinthe où tous les murs sont fermés. À chaque itération, on choisit un mur à ouvrir de *manière aléatoire* :

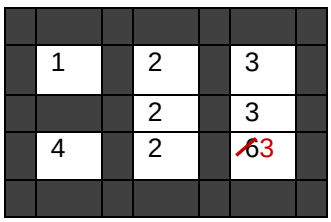
- Lorsqu'un mur est ouvert entre deux cellules adjacentes, les deux cellules sont liées entre elles et forment un chemin. L'identifiant de la première cellule est recopié dans la seconde.
- À chaque fois que l'on tente d'ouvrir un mur entre deux cellules, on vérifie que ces deux cellules ont des identifiants différents.
    - Si les identifiants sont identiques, c'est que les deux cellules sont déjà reliées et appartiennent donc au même chemin. On ne peut donc pas ouvrir le mur.
    - Si les identifiants sont différents, le mur est ouvert, et l'identifiant de la première cellule est affecté à toutes les cellules du second chemin.

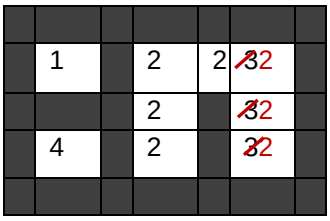
Situation initiale (taille 5*7) :

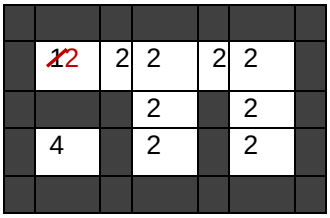


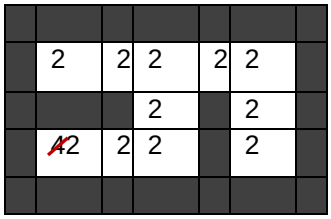
Création du labyrinthe :











Situation terminale :