RAPPORT EXPLICATIF

UBFC
UNIVERSITÉ
BOURGOGNE FRANCHE-COMTÉ

du projet d'Info 4A - Labyrinthe en C et C++
Partie 2 - Génération d'un labyrinthe intéressant

Par : Evan PETIT - IE4 I912
Professeur de TP : M A. OUAZZANI

1. Prérequis :

On suppose être déjà muni d'une fonction *bool connexe()* renvoyant 1 si le labyrinthe est connexe, et 0 si elle ne l'est pas. On appelle "case blanche" une case n'étant pas un mur.

Alors l'algorithme de génération (fonction *void genLaby(double densite)*) fonctionne ainsi :

On initialise plusieurs déjà plusieurs variables :
int duree **:** (constante définie dans le programme) La boucle générant le labyrinthe ne pourra pas tourner plus de *duree* fois afin d'éviter le bouclage infini.
int nbCases : Il s'agit du nombre de cases blanche sque l'on souhaite.
Il vaut la taille de la grille x *densite* (le paramètre passé à la fonction *genLaby*)
double marge **:** (constante définie dans le programme) On considère que le labyrinthe est correctement généré quand il y a *nbCases* cases blanches dans la grille,
avec une marge d'erreur de + ou - *nbCases* x *marge*/2

La grille est complètement vide de murs. Avec une boucle while, à chaque étape on :

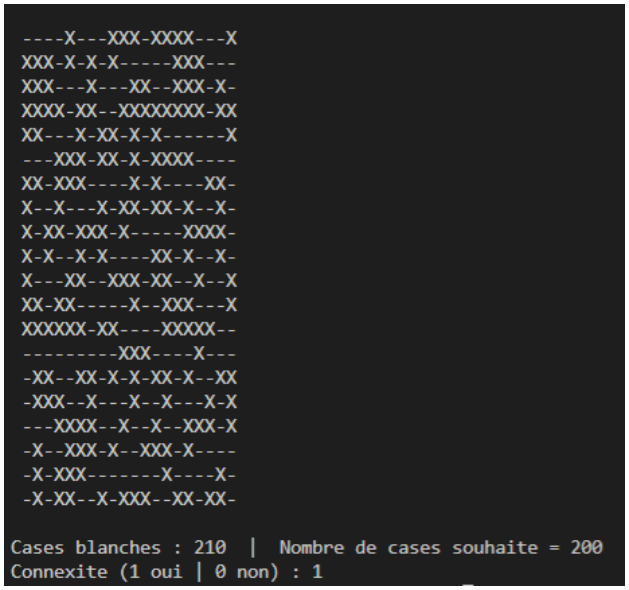
- Remplace une case blanche choisie aléatoirement dans l'ensemble des cases blanches de la grille par un mur
- On vérifie si le labyrinthe est toujours connexe malgré cet ajout.
    - S'il ne l'est plus, on remplace ce mur par une case blanche
    - S'il l'est, on vérifie si le labyrinthe possède le nombre requis de case blanche
        - Si oui, on lève un drapeau pour sortir de la boucle
        - Si non, on continue de boucler (pour un maximum de *duree* fois)

Puisqu'on ne garde jamais un mur qui rend le labyrinthe non connexe, que la durée soit dépassée ou non, le labyrinthe rendu sera TOUJOURS connexe. Cependant, si on dépasse la durée d'execution, le labyrinthe rendu ne respectera pas le nombre de cases blanches qu'on lui a demandé - il s'en approchera tout de même fortement.

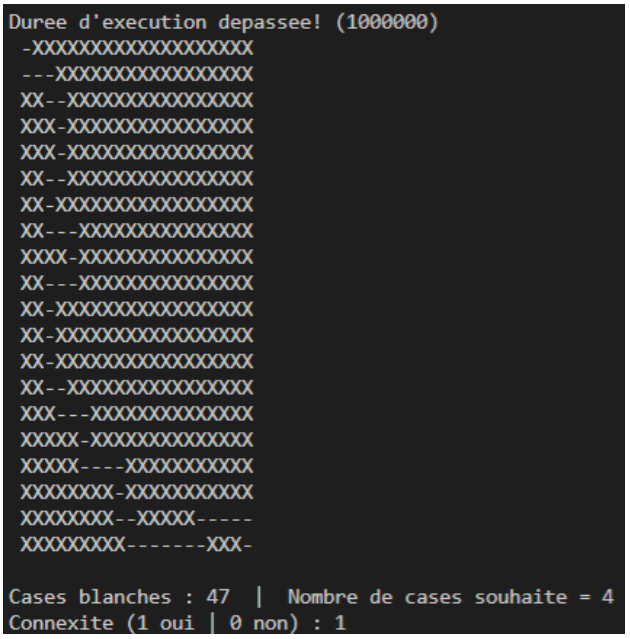
En particulier, si tant est que le labyrinthe tourne assez longtemps, il génèrera presque tout le temps un labyrinthe constitué d'un seul couloir reliant l'entrée et la sortie.

On ne considère pas les pires des cas où l'aléatoire génère d'atroces labyrinthes irrattrapables : C'est possible mais très rare.

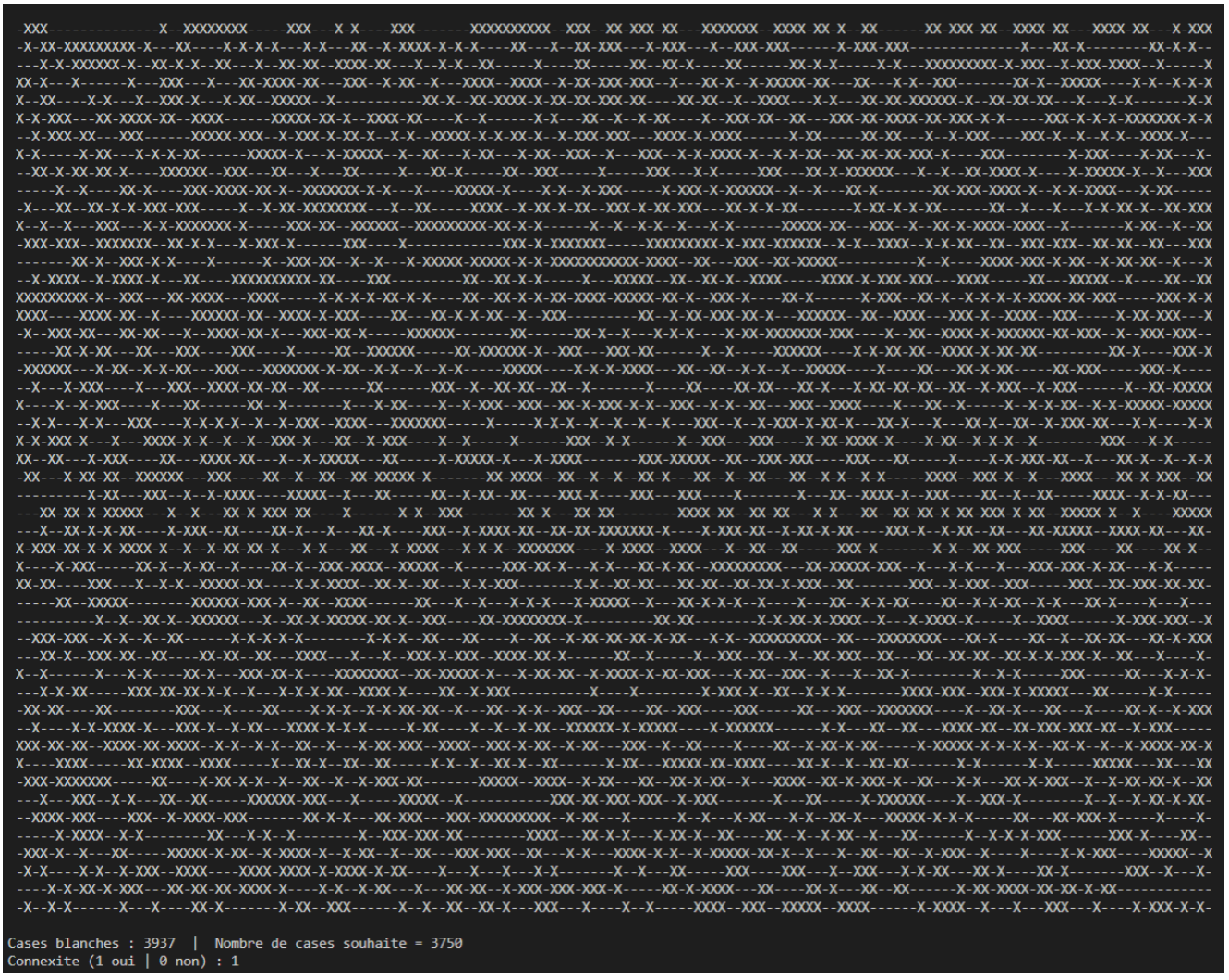
2. Exemples de labyrinthes :



Labyrinthe 20*20 de densité 0.5



Labyrinthe 20*20 de densité 0.01
20*20*0.01 = 4, c'est impossible. Le labyrinthe sorti est quand même relativement "petit".



Labyrinthe 50*150 de densité 0.5. Sorti en une poignée de secondes : L'algorithme est assez performant tant qu'on ne demande pas des densité extrêmement petites.