IFT 1016 – Programmation I

Devoir 2

- À faire en groupe de **deux** étudiants.
- Remise : Le 18 décembre 2019 à **23:59** au plus tard
- Il y aura une pénalité de **10%** par jour de retard.

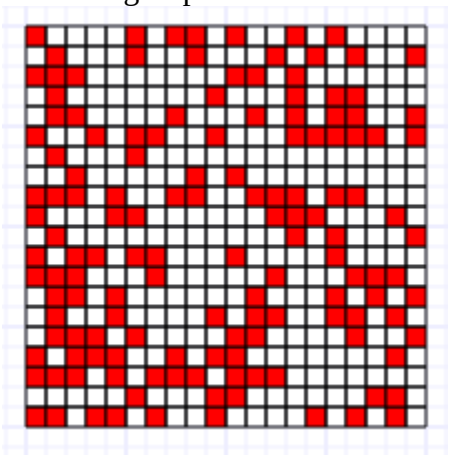
1. Objectifs du TP2

Dans ce projet, vous aurez l'occasion de pratiquer les concepts suivants:

- Les fonctions et la décomposition fonctionnelle
- Les tableaux multidimensionnels
- Les opérations de dessin (tortue)

2. Énoncé

Vous devrez écrire une implantation d'un jeu exécuté dans l'environnement CodeBoot.
Le jeu de la vie, que vous devez coder, représente un automate cellulaire imaginé par John Horton Conway en 1970 (https://fr.wikipedia.org/wiki/Jeu_de_la_vie). Le jeu de la vie est un jeu dit à "0 joueur", au sens où le déroulement du jeu dépend seulement de sa configuration initiale.
On dispose d'une grille de cellules qui peuvent être soit vivantes soit mortes et dont l'état change au fil du temps suivant des règles prédéfinies.

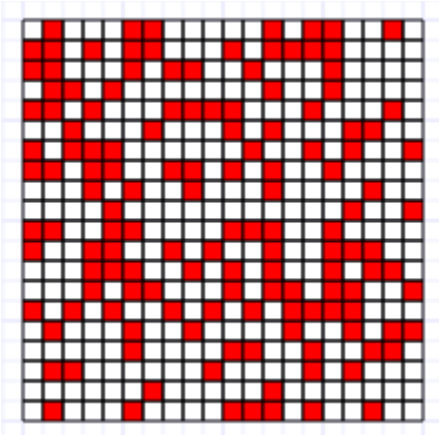


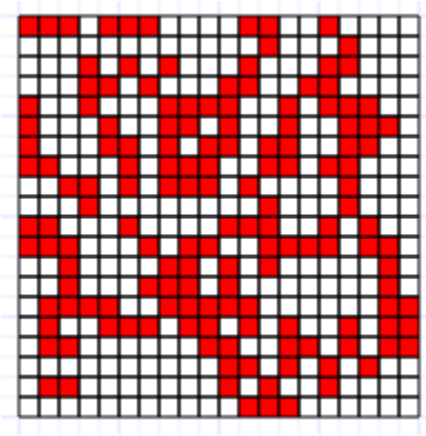
La partie se déroule de la manière suivante:

1. Laisser au hasard quelles cellules commencent vivantes et quelles commencent mortes; Le nombre de cellules vivantes devra être défini en pourcentage en utilisant la commande prompt. Donc, si le nombre des cases dans la grille de jeu est 25 et le pourcentage a été définie par un utilisateur come 35%, donc vous devriez initialiser

comme cellules vivantes, 0.35*25 = 8.75, 8 ou 9 cellules selon la méthode d'arrondie choisie.

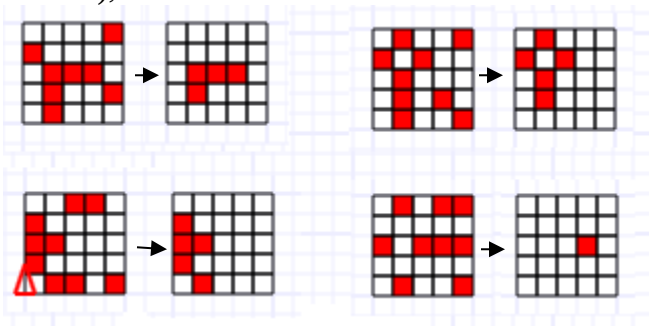
2. Démarrer le jeu et laisser la grille changer d'état toute seule. Après chaque étape redessiner la grille en faisant une pause de 0.01 secondes.
3. Apprécier les jolis motifs de l'automate qui évoluent sur l'écran.



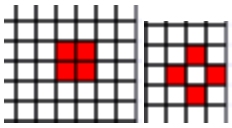


Une cellule est dite vivante lorsqu'elle est colorée. Choisissez une couleur de votre choix pour colorier les cellules vivantes. Les cellules mortes laissez en blanc. Pour effectuer les dessins de l'évaluation du jeu, utilisez la tortue. Redessinez les états de jeu chaque 0.1 seconde pour avoir une animation.
Pour colorier une case dans la grille, utiliser la méthode suivante : positionner la tortue au milieu de la case à colorier, ajuster l'épaisseur de la ligne tracée par la tortue à la valeur correspondante à la taille de la case et faites un mouvement « forward ».

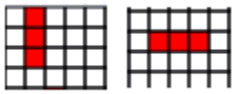
À chaque "tour", chaque cellule change d'état en fonction de ses voisins:
• Une cellule qui a moins de deux voisins vivants à une étape sera morte à la prochaine étape (sous-population);
• Une cellule qui a deux ou trois voisins vivants continuera de vivre à la prochaine étape;
• Une cellule qui a plus de trois voisins vivants sera morte à la prochaine étape (surpopulation);
• Une cellule morte qui a exactement trois voisins vivants "naîtra" à l'étape suivante et sera donc vivante (reproduction);



Étape initiale → Étape suivante



Structures qui ne changent pas



Structure qui oscille

Vous devez dessiner une grille carrée 20*20 avec les cases de 10 pixels chacune.
Le contexte du jeu peut être sauvegardé dans une variable globale représentant une grille (tableau de deux dimensions). Chaque case de la grille peut être accéder selon ses indices.
Pour ce jeu vous devez fournir un menu à l'utilisateur pour qu'il puisse spécifier le paramètre unique de jeu :

- Nombre de cellules vivantes, en pourcentage.
  On peut choisir le pourcentage approximatif de cellules vivantes en entrant un chiffre entre 1 et 99.

Après avoir spécifié le paramètre, vous devez lancer le jeu en boucle infinie.
Pour ce devoir il faudra implémenter les tests unitaires seulement pour les fonctions auxiliaires utilisées et pas pour les fonctions de dessins, car les tests de ceux derniers seront effectués visuellement.

3. Évaluation

Ce travail compte pour 15% de la note finale du cours. Vous devez faire le travail par groupes de 2 personnes. Indiquez vos noms clairement dans les commentaires au début de votre code.
Remettez le fichier `jeuVie.js` La remise doit se faire sur le site Studium du cours.
Voici les critères d'évaluation du travail :

- l'exactitude (respect de la spécification)
- l'élégance et la lisibilité du code
- la présence de commentaires explicatifs lorsque nécessaire
- le choix des identificateurs
- la décomposition fonctionnelle et le choix de tests unitaires pertinents

Indications :

- La performance de votre code doit être raisonnable.
- Chaque fonction devrait avoir un bref commentaire pour indiquer ce qu'elle fait.
- Il devrait y avoir des lignes blanches pour que le code ne soit pas trop dense (utilisez votre bon sens pour arriver à un code facile à lire)
- Les identificateurs doivent être bien choisis pour être compréhensibles (évitez les noms à une lettre, à l'exception de i, j, . . . pour les variables d'itérations des boucles for).
- Vous devez respecter le standard de code pour ce projet (soit, les noms de variables en «camelCase »).