1 Le jeu

Le Jeu "Connexion" a lieu sur un tableau $n \times n$, comme celui sur la Figure 1 (ici, $n = 8$) :

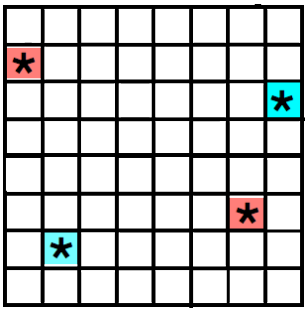

Figure 1 – Position initiale

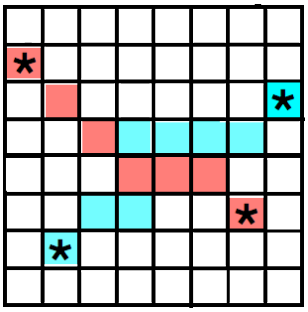

Figure 2 – Position où Rouge gagne

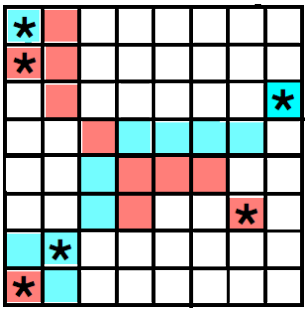

Figure 3 – Qui gagne ?

Il y a deux joueurs, R (rouge) et B (bleu). Au début du jeu, chaque joueur dispose de k cases, $k > 1$, de sa couleur, positionnées de manière aléatoire sur le tableau et marquées comme $*$. Par exemple, sur la Figure 1, $k = 2$. Tour à tour, les joueurs colorent les cases libres en leur couleur respective. Le but du joueur R est de pouvoir le plus rapidement possible relier entre elles toutes les cases $*$ de couleur rouge par des chemins de couleur rouge. Le but du joueur B est similaire, avec les cases bleues. Dans la Figure 2, R gagne. Il est toutefois possible que les joueurs ne réussissent pas à connecter toutes leurs cases $*$. Dans ce cas, le joueur ayant réussi à relier entre elles un nombre plus grand de cases gagne. En cas d'égalité, c'est la chronologie qui est prise en compte, le joueur gagnant étant celui qui a relié en premier ses cases. Ainsi, par exemple dans la Figure 3 où $k = 3$, on suppose que B a gagné s'il est le premier à avoir relié deux cases $*$ bleues, malgré le fait que R a fait la même chose (mais plus tard) avec deux cases $*$ rouges.

2 Travail à faire - Obligatoire (sur 16 points)

Le but du projet est d'abord de gérer le jeu pour deux joueurs humains, et ensuite de proposer des stratégies de jeu simples pour l'ordinateur. L'efficacité algorithmique doit être visée en permanence, y compris en choisissant les structures de données les plus efficaces.

Le programme doit implémenter a minima les fonctionnalités ci-dessous. Il doit également permettre de tester ces fonctionnalités **à tout moment du jeu** par l'utilisateur à l'aide du menu que vous devez créer. Il est impératif d'utiliser exactement les noms de méthodes indiqués ci-dessous (par contre, vous pouvez choisir vous-mêmes le nombre et le type des paramètres, ainsi que la classe dans laquelle vous définirez chaque méthode). Les cases du tableau sont définies par deux indices, la case (i, j) étant celle sur la i-ème ligne et j-ème colonne. La case $(1, 1)$ est en haut à gauche du tableau.

Fonctionnalités :

1. *colorerCase* : colorer une case (x, y) en couleur c sur le tableau de jeu
2. *afficheComposante* : afficher la composante dont la case (x, y) de couleur c appartient, c'est-à-dire l'ensemble des cases de couleur c reliés à (x, y) par un chemin de couleur c
3. *existeCheminCases* : tester s'il y a un chemin d'une couleur donnée entre deux cases données du tableau de jeu
4. *relierCasesMin* : étant données deux cases (x, y) et (z, t) de la même couleur c, indiquer le nombre minimum de cases qu'il faut colorer de cette même couleur c pour créer un chemin entre les deux cases données.
5. *nombreEtoiles* : afficher le nombre de cases $*$ que possède la composante dont la case (x, y) de couleur c fait partie
6. *afficheScores* : afficher le nombre de cases $*$ maximum que chaque joueur a réussi à connecter au sein d'une composante connexe
7. *relieComposantes* : tester si la coloration de la case (x, y) en couleur c relie deux ou plusieurs composantes de couleur c existant avant la coloration
8. *joueDeuxHumains* : lancer et gérer le jeu pour deux humains, en récupérant à chaque coup les coordonnées des cases à colorer. La partie est finie lorsque l'un des joueurs a réussi à connecter toutes ses cases $*$, l'un des joueurs a décidé d'abandonner ou bien toutes les cases ont été colorées. Dans ce dernier cas, on détermine le gagnant comme celui qui était le premier à créer une des composantes possédant le plus de ses cases $*$.

Un affichage à l'écran est nécessaire pour chacune des fonctionnalités citées, mais la partie "graphique" ne doit pas vous prendre beaucoup de temps (elle ne sera pas notée). Vous pouvez si vous le souhaitez :

- soit récupérer le code de la construction du tableau sur Internet en vous assurant que l'auteur vous en accorde le droit par une licence ou par une notice sur sa page. Bien indiquer la source et donner le copyright à son auteur.
- soit le réaliser sous forme "texte" (affichage au terminal), en remplaçant les cases vides par 0, les cases rouges par 1 et les cases bleues par 2. Les cases $*$ par $*$ et $**$, respectivement.

3 Travail à faire - Optionnel (sur 4 points)

Dans cette partie, nous vous proposons d'implémenter une ou plusieurs stratégies de jeu, afin de pouvoir faire jouer l'ordinateur contre un humain. Il s'agit de stratégies de jeu assez simples, utilisant l'analyse du tableau de jeu, et non pas de méthodes sophistiquées. Pour cela, vous devez implémenter :

9. une ou plusieurs méthodes *evaluerCaseZ*, où Z est un entier $1, 2, 3, \ldots k$ (où k est le nombre de méthodes d'évaluation proposées), qui évalue l'intérêt de colorer une case (x, y) en couleur c.
   9.1 Soit (x_1, y_1), (x_2, y_2), ..., (x_n, y_n) les cases $*$ colorées en rouge (resp. bleu). Alors, le centre de masse du jouer R (resp. B) est la case (x^*, y^*), où $x^* = (x_1 + x_2 + \ldots + x_n)/n$ et $y^* = (y_1 + y_2 + \ldots + y_n)/n$. Par exemple, la méthode *evaluerCase1* doit évaluer la proximité entre la case (x, y) qu'on compte colorer en c et son centre de masse.
10. une méthode *joueOrdiHumain* qui lance le jeu pour l'ordinateur et un humain. Les conditions d'arrêt sont les mêmes comme pour *joueDeuxHumains*.

4 Rendu TP

Le programme doit fonctionner parfaitement sur les machines du CIE, sous Linux. Un rapport d'au maximum 10 pages doit être fourni, comprenant (entre autres) :

- les commandes de compilation et exécution en mode console
- la description des structures de données choisies
- pour chaque méthode des points 1 à 10 :
  - le détail de la méthode, sous forme algorithmique (c'est-à-dire en pseudo-code) ;
  - l'explication de son fonctionnement, en français ;
  - les raisons pour lesquelles l'algorithme proposé est correct ;
  - sa complexité ;
  - les raisons pour lesquelles vous considérez cet algorithme aussi efficace que possible
- le cas échéant, la description en français des méthodes d'évaluation définies au point 9.
- des jeux de données commentés

Important

- Les fichiers du programme, ainsi que les jeux de données, seront mis dans un répertoire Nom1Nom2 (où Nom1 et Nom2 sont les noms des deux étudiants du binôme). Ce répertoire sera ensuite archivé sous le nom Nom1Nom2.zip. L'archive sera déposée sur Madoc, au plus tard **le vendredi 9 décembre 2016 à 23h59** (Madoc n'acceptera pas de retard).
- La dernière séance est consacrée à une démo de 10 minutes par projet. Cette dernière séance n'est donc pas une séance de travail sur le projet.
- Tout est important pour la notation. En particulier, il sera accordé beaucoup d'attention au respect des consignes et à la recherche d'une complexité minimum - garantie d'une efficacité maximum - pour vos méthodes, via la mise en place des structures de données les plus adaptées.