Construction d'un labyrinthe parfait

Je rappelle le principe de construction vu en cours :

- *le cadre " abstrait " du labyrinthe est un tableau carré, dont les éléments sont des booléens :* $True$ *si la case a été visitée,* $False$ *sinon ;*
- *le cadre " physique " du labyrinthe correspond au tableau des indices du tableau précédent, dont les éléments sont des couples* (x, y) *appelés cases.*
- *le cheminement dans ce tableau est une pile : on empile la case qu'on visite, on la dépile si on ne peut plus avancer depuis cette case ;*
- *pour avancer on choisit de manière équiprobable parmi les cases adjacentes pas encore visitées ;*
- *on ne s'arrête que quand toutes les cases ont été visitées ;*
- *pour tracer le labyrinthe, on relie les cases par un segment au fur et à mesure qu'on avance.*

Exercice 1

a. Ecrire une fonction (en fait procédure) *lab* qui prend en argument un entier n, et crée le labyrinthe logique. Toutes les cases doivent être initialisées à $False$ (pas encore visitées), sauf la case $(0, 0)$ initialisée à $True$ (car on y est au départ).
Attention : ce labyrinthe logique devra pouvoir être visible par les autres fonctions qu'on va écrire ensuite, ce qui pratiquement signifie qu'il doit être déclaré en *variable globale*.

b. Tester que tout fonctionne comme prévu pour $n = 5$.

Exercice 2

Ecrire les fonctions élémentaires suivantes :

a. *cocher* qui prend en argument une case et la marque $True$ si elle est effectivement dans le tableau, sinon ne fait rien ;

b. *visitee* qui prend en argument une case et renvoie le booléen correspondant si elle est dans le tableau, sinon $True$ pour être sûr de ne pas y aller ;

c. *options* qui prend en argument une case et renvoie la liste des cases adjacentes disponibles (pas encore visitées) ;

d. *choix* qui prend en argument une case et renvoie celle qui a été choisie aléatoirement parmi les options.

e. Tester évidemment toutes ces fonctions. Puis réinitialiser le labyrinthe logique.

Exercice 3

Construction de la pile de cheminement dans le tableau.

a. Ecrire les instructions qui permettent de passer d'une étape à la suivante : empilement d'une nouvelle case si on peut avancer, dépilement sinon.

b. Ecrire la boucle qui permet le cheminement complet, c'est à dire d'assurer qu'on a bien visité toutes les cases du tableau.

c. Insérer une instruction qui imprime l'état de la pile à chaque étape, puis contrôler que tout va bien avec $n = 5$. Ensuite, supprimer cette instruction et réinitialiser le labyrinthe logique.

Exercice 4

Construction graphique du cheminement.

a. Ecrire les instructions de préparation du graphique : effacement des graphiques précédents, repère orthonormé, axes pas apparents, taille de la fenêtre d'affichage, fond gris.

b. Ajouter aux instructions de l'exercice 3, au bon endroit, les instructions qui permettent de relier par un trait une case et la suivante. On prendra un trait clair de largeur inversement proportionnelle à n (pourquoi ?)

c. Placer le tout, dans le bon ordre, dans une fonction *labyr* d'argument un entier.

d. La tester ! On prendra au début de petites valeurs de n afin de bien vérifier que tout fonctionne comme prévu ; puis $n = 50$ ou plus afin d'obtenir un tracé intéressant. Rectifier au besoin les instructions, en particulier ajuster la largeur du trait pour que ce soit joli.

Exercice 5

Maintenant que le labyrinthe est construit, vous pouvez essayer d'écrire une fonction $solution$ qui prend en argument une case de fin, et renvoie le graphique qui montre le chemin à parcourir dans le labyrinthe pour aller de $(0, 0)$ à cette case. Ce graphique devra bien sûr être superposé sur celui du labyrinthe, et dans une autre couleur (vert par exemple).
L'idée est de garder en mémoire, pour chaque case visitée, celle qui a permis d'y accéder. A vous de la développer correctement.