

# Mini-projet 2

## Algorithme de Dijkstra sur grille

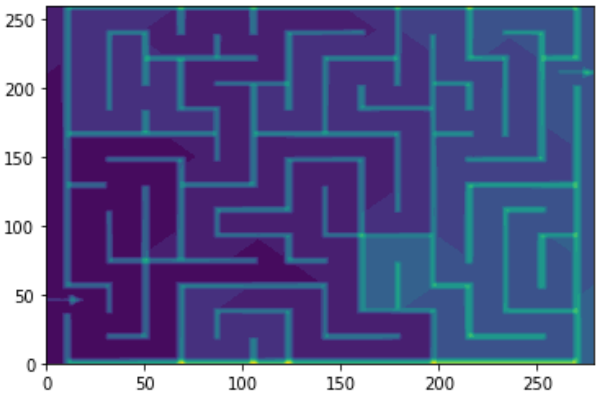
### Question 1

Algorithme	Code Python
<p><b>Data</b> : 4 Entiers <math>i, j, nx, ny</math>  <b>Result</b> : Les indices des pixels voisins du pixel <math>(i, j)</math> dans une grille de taille <math>nx \times ny</math></p> <p>Initialiser une liste de voisins vide;  Initialiser les déplacements à effectuer pour trouver les voisins de <math>(i, j)</math>;  Pour chaque déplacement faire      Calculer la position du voisin;      Si la position du voisin courant est dans la grille alors          Ajouter le voisin courant dans la liste des voisins;      FIN SI  FIN BOUCLE</p>	<pre>def voisins(i, j, nx, ny):     """     Cette fonction renvoie les indices des pixels voisins du pixel (i, j)     dans une grille de taille nx par ny      Parameters     -----     i : int         L'indice de la ligne du pixel dans la grille     j : int         L'indice de la colonne du pixel dans la grille     nx : int         La taille de la grille selon l'axe horizontal     ny : int         La taille de la grille selon l'axe vertical      Returns     -----     listeVoisins : list of tuples         Une liste contenant les indices des pixels voisins valides du pixel (i, j)     """     listeVoisins = []      # Liste de Tuples correspondant aux déplacements pour trouver les voisins     positionVoisins = [(-1, 0), (1, 0), (0, -1), (0, 1)]      for vx, vy in positionVoisins:         voisinI, voisinJ = i + vx, j + vy          # On vérifie si les voisins peuvent être dans la grille         if (0 &lt;= voisinI &lt; nx) and (0 &lt;= voisinJ &lt; ny):             listeVoisins.append((voisinI, voisinJ))      return listeVoisins</pre>

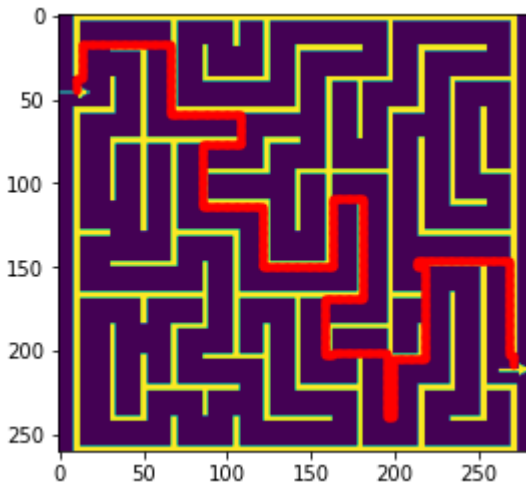
### Question 2

Voir code Python fonction **dijkstraSurGrille**

### Question 3

Code Python	Sortie graphique
<pre>u = imageio.imread('Labyrinthe002.jpg') uu = np.sum(u,axis=2) nx = np.shape(uu)[0] ny = np.shape(uu)[1] K = np.ones((nx,ny)) K[np.where(uu&lt;300)] = 500 x0=[45,10] d, px, py = dijkstraSurGrille(K, x0) plt.figure(1) plt.contourf(d,20)</pre>	

#### Question 4

Code Python	Sortie graphique
<pre>plt.figure(3) plt.imshow(K) px = np.array(px) py = np.array(py) scourant = [208,270] while not(scourant==x0):     plt.plot(scourant[1],scourant[0], 'r. ')     scourant = [px[scourant[0],scourant[1]],py[scourant[0],scourant[1]]]</pre>	 <p>The figure displays a 2D plot of a maze. The maze is represented by a grid of black and white pixels. The red path starts at the top left and winds through the maze. The axes are labeled from 0 to 250.</p>