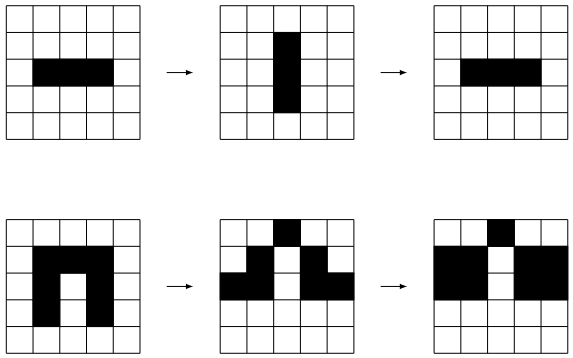
Proposition de corrigé du TPAutomates cellulaires

1 Introduction

1. On obtient



2 Implémentation

2.

```
def init_jeu_u(N) :
    grille = np.zeros((N, N), dtype=float)
    c = N//2
    # On cree une liste d'abscisses et d'ordonnees pour parcourir les points
    # (-1,-1), (-1,0), (-1,1), (0,1), (1,-1), (1,0), (1, 1)
    lX = [-1, -1, -1, 0, 1, 1, 1]
    lY = [-1, 0, 1, 1, -1, 0, 1]
    # zip permet de parcourir simultanement les listes lX et lY
    for (x,y) in zip(lX, lY):
        grille[c+x, c+y] = 1
    return grille
```

3.

```
def afficher_jeu(grille):
    X = []
    Y = []
    for i in range(1,grille.shape[0]-1):
        for j in range(1,grille.shape[1]-1):
            if (grille[i,j] != 0):
                X.append(i)
                Y.append(j)
    plt.clf()
    plt.axis([0, grille.shape[1], 0, grille.shape[0]])
    plt.scatter(X, Y,s=100, marker='s',color='black')
    plt.show()
```

4.


```
def compter_voisins(grille, i, j):
    return (grille[i-1,j-1] + grille[i-1, j] + grille[i-1,j+1] + grille[i,j-1] + grille[i,j+1] + gr
```

5.

```
def iterer_jeu_ponct(grille):
    precG = grille.copy()
    for i in range(1,grille.shape[0]-1):
        for j in range(1,grille.shape[1]-1):
            c = compter_voisins(precG, i, j)
            if ((grille[i,j] == 1 and c == 2) or c == 3):
                grille[i,j] = 1
            else:
                grille[i,j] = 0
    return(grille)
```

6.

```
def iterer_jeu(grille, nb_iter, afficher = False):
    for k in range(nb_iter):
        precG = grille.copy()
        for i in range(1,grille.shape[0]-1):
            for j in range(1,grille.shape[1]-1):
                c = compter_voisins(precG, i, j)
                if ((grille[i,j] == 1 and c == 2) or c == 3):
                    grille[i,j] = 1
                else:
                    grille[i,j] = 0
        if (afficher):
            afficher_jeu(grille)
            plt.title('${:3d}$'.format(k))
            plt.pause(10**-15)
```