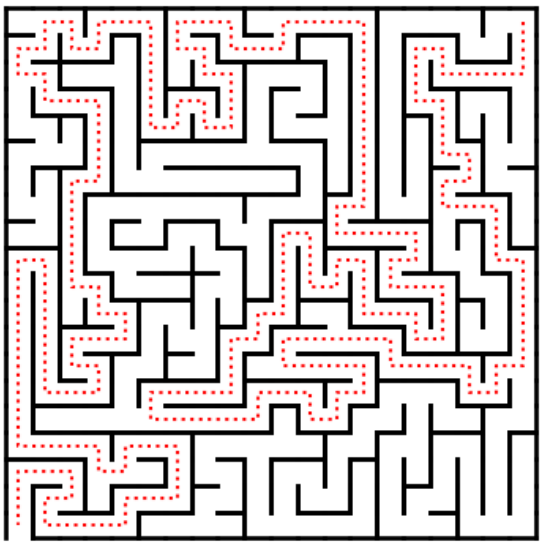
# I Sortir d'un labyrinthe

**1.**
```
def exit(Lab,depart) :
    vues = {x:False for x in Lab}
    pile = []
    case = depart
    pile.append(case)
    origine = {}
    while case != (0,0) : # on arrête dès qu'on a rejoint la sortie
        case = pile.pop()
        if not vues[case] :
            vues[case] = True
        for s in Lab[case] :
            if not vues[s] :
                origine[s] = case # on mémorise l'origine
                pile.append(s)
    chemin = [(0,0)]
    case = (0,0) # on reconstruit le chemin depuis la sortie
    while case != depart :
        case = origine[case]
        chemin.append(case)
    return chemin[::-1] # et on le retourne pour partir de la case départ
```

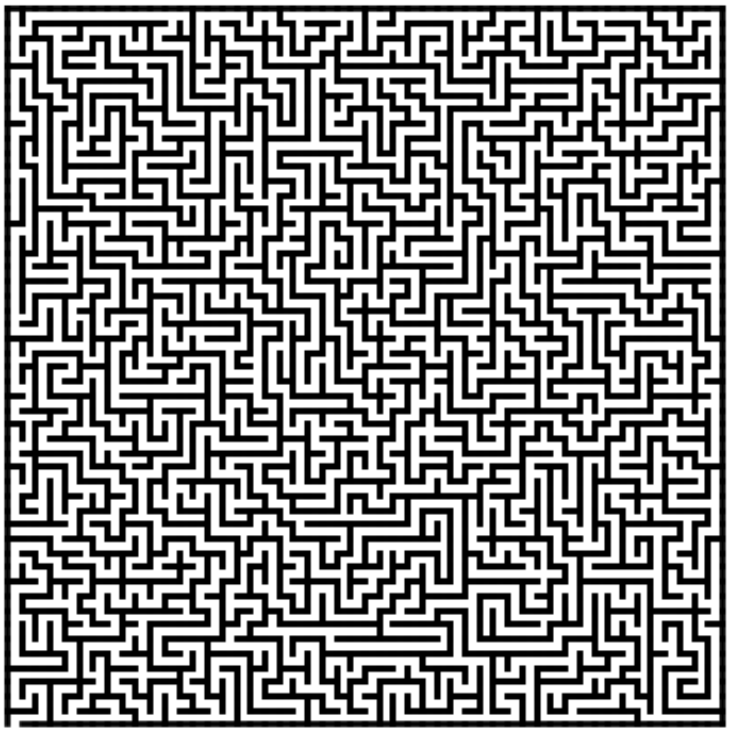
# II Création d'un labyrinthe parfait

**1.**
```
def initialiser(n) :
    d = {(i,j):[] for i in range(n) for j in range(n)}
    return d
```

**2.**
```
def maze(n) :
    d = initialiser(n)
    vues = {x:False for x in d}
    pile = []
    case = (0,0)
    pile.append(case)
    origine = {}
    while len(pile)>0 : # parcours intégral pour accéder à toutes les cases
        case = pile.pop()
        if not vues[case] :
            vues[case] = True
        L = acces(d,case)
        for s in L :
            if not vues[s] : # on ne fait rien si on est dans un cul de sac (
                toutes les cases voisines déjà visitées)
                pile.append(s)
                origine[s] = case
    for s in origine : # on fait tomber les murs sur le chemin et on crée le
        graphe non orienté
        d[s].append(origine[s])
        d[origine[s]].append(s)
    return d
```



Le chemin vers la sortie de `Lab` depuis la case en haut à droite (19,19)



Un labyrinthe parfait $50 \times 50$