P13

RECHERCHE DANS UNE LISTE - DICHOTOMIE

Exercice 1 Écrire une fonction d'en-tête `def dichotomie(L,x)` qui renvoie True si x appartient à la liste L et False sinon.

```
```

Exercice 2

1. On va mesurer le temps d'exécution de notre recherche dichotomique dans un cas peu favorable. Prenons L=[1,2,3,..,10000] et x=0.

```
import time

L = range(        )
x = 0

debut = time.time() # on note l'heure précise
dichotomie(L,x) # on fait tourner la fonction
fin =  time.time() # on note l'heure précise
duree =
print(duree)
```

2. On souhaite tracer le graphique du temps d'exécution en fonction de $n \in \mathbb{N}^*$ quand on recherche 0 dans $L = [1, 2, \ldots, n]$. Completer le programme.

```
import matplotlib.pyplot as plt

t = [] # liste des temps d'exécution
N = range(10000,200000,1000) # abscisses n

for n in N:
    L = range(        )
    debut = time.time() # on note l'heure précise
    dichotomie(L,x) # on fait tourner la fonction
    fin =  time.time() # on note l'heure précise
    t.append(               )

plt.plot(N,t)
plt.show()
```

Exercice 3 La mesure de la durée n'étant pas suffisamment précise, nous allons compter le nombre d'itérations réalisées dans la boucle. Nous comparerons la dichotomie à l'algorithme naïf.

1. Écrire une fonction `dichotomie_iterations` qui renvoie le nombre d'itérations dans la boucle pour obtenir le résultat.

2. Écrire une fonction `naif_iterations` qui renvoie le nombre d'itérations réalisées avec une recherche naïve (où on parcourt la liste dans l'ordre des éléments).

3. Graphique (sur notebook).