

seance_25nov

November 26, 2025

Thèmes : recherche naïve mot texte; lecture écriture en Python, directive `with`, dictionnaire vide

1 Recherche naïve d'un mot dans un texte

```
[1]: def start(t: str, m: str, i: int) -> bool:
      """Renvoie True si m
      apparaît dans t à partir de la position i."""
      if i + len(m) > len(t):
          return False
      for j in range(len(m)):
          if t[i + j] != m[j]:
              return False
      return True
```

```
# In[32]:
```

```
texte = "abracadabra"
start(texte, "ac", 3), start(texte, "ac", 0)
```

```
def cherche_mot_naif(t: str, m: str) -> int:
    """Renvoie la position de la première
    occurrence de m dans t, ou -1 si absente."""
    for i in range(len(t) - len(m) + 1):
        if start(t, m, i):
            return i
    return -1
```

```
# In[30]:
```

```
texte = "abracadabra"
print(cherche_mot_naif(texte, "cada"))    # 4
print(cherche_mot_naif(texte, "cadabra")) # 4
```

```
print(cherche_mot_naif(texte, "xyz"))    # -1
```

```
4
4
-1
```

2 Analyse de complexité

La fonction `start`, dans le pire des cas, parcourt toute la boucle (c'est quand le mot est bien en position i dans le texte). Complexité $O(|m|)$

Meilleur cas $O(1)$: quand la 1ere lettre du mot est différente de la lettre i du texte.

La fonction de recherche a un meilleur cas lorsque la 1ere lettre du mot n'est pas dans le texte. Dans ce cas la complexité est un $O(|t| - |m|)$.

Dans le pire cas, on parcourt pour chaque lettre du texte, toute la longueur du mot et on se rend compte seulement à la fin que la dernière lettre n'est pas la bonne.

Cela arrive lorsque le mot est de la forme

```
t = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
m = xxxxy
```

La complexité est un $((|t| - |m|) \times |m|)$

3 Lecture Ecriture

```
[25]: import os
```

```
[3]: os.getcwd()
```

```
[3]: '/home/noyer/CPGE_n/CPGE_2017/Info/CoursInfo/IPT/Tris/Tris2025'
```

```
[26]: def exemple(l:list)->None:
      if l :#l non vide considéré kom true
          print("non vide")
      else:# l vide considéré comme false
          print("vide")
```

```
[27]: exemple([1,2,3])
```

non vide

```
[28]: exemple([])
```

vide

```
[31]: for e in ["toto", "", 0, 1]:
      exemple(e)
```

```
non vide  
vide  
vide  
non vide
```

```
[33]: os.listdir()
```

```
[33]: ['seance_15oct.zip',  
      'seance_5nov.zip',  
      'tri_2025.ipynb',  
      'tp_tris_2025_Enonce.pdf',  
      'fichier.txt~',  
      '.ipynb_checkpoints',  
      'seance_19nov.ipynb',  
      'tri_2025.py',  
      '.#seance_26nov.py',  
      'seance_19nov.log',  
      'seance_19nov.zip',  
      'seance_25nov.ipynb',  
      'seance_19nov.pdf',  
      'seance_12nov.zip',  
      'tp_tris_2025.tex',  
      'seance_19nov.aux',  
      'seance_5nov.py',  
      '#seance_26nov.py#',  
      'fichier.txt',  
      'seance_5nov.ipynb',  
      'seance_15oct.py',  
      'seance_19nov.tex',  
      'seance_5nov_copie.ipynb',  
      'seance_19nov.py',  
      'untitled.txt',  
      'seance_5nov.py~',  
      'README.tex',  
      'README.tex~',  
      'tp_tris_2025.tex~',  
      'seance_12nov.pdf',  
      'seance_12nov.ipynb',  
      'seance_15oct.py~',  
      'fichier2.txt',  
      'seance_19nov.out',  
      'seance_12nov.py']
```

```
[11]: "fichier.txt" in os.listdir()
```

```
[11]: True
```

```
[10]: os.getcwd()
```

```
[10]: '/home/noyer/CPGE_n/CPGE_2017/Info/CoursInfo/IPT/Tris/Tris2025'
```

```
[13]: f = open("fichier.txt", 'r')
      s = f.read()
      f.close()
      print(s)
```

```
coucou
gaga
toto et gogo
```

```
[35]: f = open("fichier.txt", 'r')
      ligne = f.readline()
      while ligne:
          print(ligne, end="")
          ligne = f.readline()
      f.close()
```

```
coucou

gaga
toto et gogo
```

4 mode écriture

```
wou a
```

```
[17]: "fichier2.txt" in os.getcwd()
```

```
[17]: False
```

```
[36]: f = open ("fichier2.txt", "w")
      f.write("un\ndeux\ntrois et quatre\n")
      f.write("cinq")
      f.close()
      f=open ("fichier2.txt", "r")
      print(f.read())
      f.close()
```

```
un
deux
trois et quatre
cinq
```

```
[37]: f = open ("fichier2.txt", "w")
      f.write("\ntoto\n")
      f.close()
      f=open ("fichier2.txt", "r")
      print(f.read())
      f.close()
```

toto

```
[38]: f = open ("fichier2.txt", "a")
      f.write("\ngogo\n")
      f.close()
      f=open ("fichier2.txt", "r")
      print(f.read())
      f.close()
```

toto

gogo

```
[39]: with open ("fichier2.txt", "a") as f:
      f.write("\nsont copains\n")

      with open ("fichier2.txt", "r") as f:
          print(f.read())
```

toto

gogo

sont copains

```
[22]: os.getcwd()
```

```
[22]: '/home/noyer/CPGE_n/CPGE_2017/Info/CoursInfo/IPT/Tris/Tris2025'
```

5 Un point sur les dictionnaires

On utilise une structure de données avec des *entrées* (ou *clés*) et des *valeurs* comme un *dictionnaire de langue française*: - les *clés* sont des mots - les *valeurs* sont les définition des mots

Autre exemple : un annuaire - entrées sont des noms - valeurs : numéros de téléphone

les listes Python peuvent être considérées abusivement des cas particuliers de dictionnaires dont les entrées sont des entiers positifs

```
[40]: d = {} #dico vide
```

```
[ ]:
```