

Chapitre 21 - Recherche textuelle

Dans la plupart des éditeurs de texte, il existe une fonctionnalité *Recherche* qui permet de rechercher la présence d'un motif dans un document.

Pour rechercher par exemple la présence du motif « *saperlipopette* » dans le texte ci-contre, on lance un « Ctrl F » et on saisi dans la fenêtre de dialogue :

Navigation

saperlipopette

Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae libero nesciunt illo consequatur doloribus adipisci, animi corporis molestiae ipsa perferendis ullam voluptate expedita magnam explicabo sint iste aut cumque vitae. Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae libero nesciunt illo consequatur doloribus adipisci, animi corporis molestiae ipsa perferendis ullam voluptate expedita magnam explicabo sint iste aut cumque vitae. Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae libero nesciunt illo consequatur doloribus adipisci, animi corporis molestiae ipsa perferendis ullam voluptate expedita saperlipopette magnam explicabo sint iste aut cumque vitae.

On découvre dans ce chapitre un algorithme développé par [Robert S. Boyer](#) et [J Strother Moore](#) en 1977. Il permet de réaliser une recherche avec une complexité faible. Avant de le présenter, on s'intéresse à l'algorithme de recherche naïf.

1- ALGORITHME NAÏF :

Exemple : `texte = "GCATCGCAGAGAGTATACAGTACG"`
`motif = "GCAGAGAG"`

index <i>i</i>	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
texte	G	C	A	T	C	G	C	A	G	A	G	A	G	T	A	T	A	C	A	G	T	A	C	G
Motif index <i>k</i>	G	C	A	G	A	G	A	G																
	0	1	2	3	4	5	6	7																

⇒ Code naïf avec 2 boucles for :

`len(texte) = len(motif) = len(texte) - len(motif) =`

```
def recherche_naive(texte, motif):  
    '''  
    renvoie la liste des indices (éventuellement vide)  
    des occurrences de la chaîne motif dans la chaîne texte.  
    '''  
    indices = []  
    for i in range(  
        n = 0  
        for k in range(len(motif)) :  
            if  
            if n == len(motif) :  
                indices.append(i)  
    return indices
```

Nombres de comparaisons :

index i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
texte	G	C	A	T	C	G	C	A	G	A	G	A	G	T	A	T	A	C	A	G	T	A	C	G
Motif index k	G	C	A	G	A	G	A	G																
	0	1	2	3	4	5	6	7																

⇒ Code naïf avec 1 boucle for et une boucle while :

`len(texte) =` `len(motif) =` `len(texte) - len(motif) =`

```
def recherche_naive(texte, motif):
    """
    renvoie la liste des indices (éventuellement vide)
    des occurrences de la chaîne motif dans la chaîne texte.
    """
    indices = []
    for i in range(len(texte) - len(motif) + 1):
        k = 0
        while
            k = k + 1
            if k == len(motif) :
                indices.append(i)
    return indices
```

index i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
texte	G	C	A	T	C	G	C	A	G	A	G	A	G	T	A	T	A	C	A	G	T	A	C	G
Motif	G	C	A	G	A	G	A	G																
index k	0	1	2	3	4	5	6	7																

⇒ Code naïf en effectuant les comparaisons en commençant par la fin du motif :

`len(texte) =` `len(motif) =` `len(texte) - len(motif) =`

```
def recherche_naive(texte, motif):
    """
    renvoie la liste des indices (éventuellement vide)
    des occurrences de la chaîne motif dans la chaîne texte.
    """
    indices = []
    i = 0
    while i <= len(texte) - len(motif):
        k = len(motif) - 1
        while k >= 0 and texte[i+k] == motif[k]:
            k = k-1
        if
            indices.append(i)
        i = i + 1
    return indices
```

Nombres de comparaisons :

⇒ Complexité de ce type d'algorithme :

⇒ Travail à faire :

- Ecrire ces 3 codes dans un fichier que vous nommerez rechercheTextuelleNaif.py.
- Rajouter un compteur de comparaisons effectuées.
- Traiter le cas suivant :

```
texte = "GCATCGCAGAGAGTATACAGTACG"
motif = "GCAGAGAG"
```


Et remplir le tableau ci-dessous (résultats à mettre en commentaire dans le fichier .py) :

	Script avec 2 boucles for	Script avec 1 boucle for et une boucle while	Script avec 1 boucle for et une boucle while en partant de la fin du motif
Nombre de comparaisons			

- Traiter le cas suivant :

```
fichier = open('Le_rouge_et_le_noir.txt', 'r', encoding="utf-8")
stendhal = fichier.read()
fichier.close()

nb = 0
print(recherche_naive(stendhal, "Julien"), nb)
```

Le fichier  [Le_rouge_et_le_noir.txt](#) est proposé en téléchargement sur nsibranly.fr

Et remplir le tableau ci-dessous (résultats à mettre en commentaire dans le fichier .py) :

	Script avec 2 boucles for	Script avec 1 boucle for et une boucle while	Script avec 1 boucle for et une boucle while en partant de la fin du motif
Nombre de comparaisons			