# Codage d’une fonction *identiqueDG\_1*

Dans un fichier nommé *algoHorspool.py*, écrire la fonction :

*identiqueDG\_1(texte: str, motif: str, i: int) :*

La fonction teste si le motif de longueur k est présent dans le texte à la position i, c’est-à-dire si *texte[i:i+k] == motif*, en comparant les caractères de *texte[i:i+k]* et de *motif* un par un de droite à gauche jusqu’à trouver un caractère différent.

Cette fonction identiqueDG\_1 renvoie un booléen : True si le motif est présent à cette position du texte, et False sinon.

# Codage d’une fonction *identiqueDG\_2 améliorée*

Toujours dans le fichier nommé *algoHorspool.py*, écrire la fonction :

*identiqueDG\_2(texte: str, motif: str, i: int) :*

Cette fonction est similaire à identiqueDG\_1 mais renvoie à la place du False un entier égal au décalage vers la droite à appliquer au motif par rapport au texte en cas d’échec lors de la comparaison. Si la première lettre lue dans le texte (la plus à droite) n’apparait pas dans le motif, ce décalage est égal à la longueur du motif. Dans le cas contraire, ce décalage est de 1.

# Premier pas vers l’algorithme de recherche textuelle de Boyer-Moore-Horspool (dit Horspool)

Nous allons ici prétraiter le motif afin d’établir un dictionnaire de décalages pertinents lié au motif. Ce dictionnaire est en fait appelé « table de saut du mauvais caractère » (table MC).

Cette table (dictionnaire) obtenue par prétraitement du motif, donne **le décalage brut** du motif à effectuer vers la droite, en cas de non correspondance de son dernier caractère avec le texte.

Aide : La formule donnant la valeur du décalage brut est : **m – j – 1**

Avec m : longueur motif j : position du caractère en partant de la gauche à 0

S’il y a plusieurs caractères identiques dans le motif le calcul se fait pour celui qui est le plus à droite, et le dernier caractère du motif n’est pas pris en compte.

Exemples : Pour le motif "TABLE"

dicoMC = {'T' :4, 'A' : 3, 'B' :2, 'L' :1}

Pour le motif "RECHERCHE", le dictionnaire de décalages brut sera :

dicoMC = {'R' :3, 'E' : 4, 'C' :2, 'H' :1}

Toujours dans *algoHorspool.py*, écrire la fonction *decalageMC(motif: str)*

Cette fonction renvoie un dictionnaire *dicoMC* représentant la table de saut MC (Mauvais Caractère), lié au motif.

Exemple : *decalageMC ('recherche') ->* {'R' :3, 'E' : 4, 'C' :2, 'H' :1}

# Codage d’une fonction *identiqueDG\_3*

Toujours dans le fichier nommé *algoHorspool.py*, écrire la fonction :

*identiqueDG\_3(texte: str, motif: str, dicoMC: dict, i: int) :*

Cette fonction est similaire à identiqueDG\_3 mais **calcule et renvoie le décalage du motif vers la droite à appliquer** (Grâce à la table de saut MC et la position du caractère du motif qui échoue la comparaison en partant de la fin du motif).

La fonction renverra un décalage de 1 si le décalage calculé est négatif.

**3 cas sont à envisager :**

1er cas : Lettre du texte qui échoue absente du motif

-> Décalage de la longueur du motif

Texte : À vaincre sans péri**l**, on triomphe sans gloire.

Motif : bari**l**  dicoMC = {'b': 4, 'a': 3, 'r': 2, 'i': 1}

4 3 2 1 0

baril

Décalage = 5 (longueur du motif puisque 'é' n’est pas dans la table de saut Mauvais Caractère)

2ème cas : Lettre du texte qui échoue présente dans le motif

-> Décalage = valeur du décalage dans dicoMC - position de la comparaison par rapport à la droite du motif (nb de lettres qui ont matché)

Texte : À vaincr**e** sans péril, on triomphe sans gloire.

Motif : crim**e**  dicoMC = {'c': 4, 'r': 3, 'i': 2, 'm': 1}

4 3 2 1 0

crime

Décalage = décalage pour r (caractère qui échoue à la comparaison) – position de la comparaison par rapport à la droite du motif (nb de lettres qui ont matché)

= 3 – 1 = 2

3ème cas : Lettre du texte qui échoue présente dans le motif mais décalage calculé négatif donc incohérent -> on

décale de 1

Texte : Il frictio**n**ne sa jambe.

Motif : fonctio**n** dicoMC = {'f': 7, 'o': 1, 'n': 5, 'c': 4, 't': 3, 'i': 2}

*7 6 5 4 3 2 1 0*

fonction

Décalage = décalage pour i (caractère qui échoue à la comparaison) – position de la comparaison par rapport à la gauche

= 2 – 5 = -3 on prend dans ce cas un décalage de 1

# Codage de l’algo de recherche textuelle Boyer-Moore-Horspool (dit Horspool)

Toujours dans le fichier nommé *algoHorspool.py*, écrire la fonction :

*horspool(texte: str, motif: str) :*

Cette fonction devra renvoyer le nombre de comparaisons effectuées, ainsi que le nombre d’occurrences du motif dans le texte avec leurs positions sous forme d’une liste.

Exemple 1:

texte = "Il y a deux histoires : l'histoire officielle, menteuse, qu’on enseigne, puis l'histoire secrète, où sont les véritables causes des événements."

motif = "histoire"

print(horspool(texte,motif))

-----> 25, 3, [12, 26, 80]

Ce qui veut dire 25 comparaisons, 3 occurrences du motif trouvées, positions des motifs trouvés dans le texte : 12, 26, 80

Exemple 2:

texte = "Rien de grand ne s'est accompli dans le monde sans passion."

motif = "savant"

print(horspool(texte,motif))

-----> 11, 0, []

Ce qui veut dire 11 comparaisons, 0 occurrence du motif trouvée, pas de positions de motifs trouvés dans le texte

# Comparaison algorithmes Naïf et Horspool

Ouvrir le fichier nommé *03\_ComparaisonAlgosNaifHorspool.py*, puis adapter les noms des fichiers importés ainsi que les noms des fonctions avec les vôtres (attention les \_ ne sont pas acceptés dans un import !).

Exécuter ce fichier, et indiquer quel est le gain en pourcentage de temps avec l’algorithme d’Horspool ?

Gain = ……………………….. %