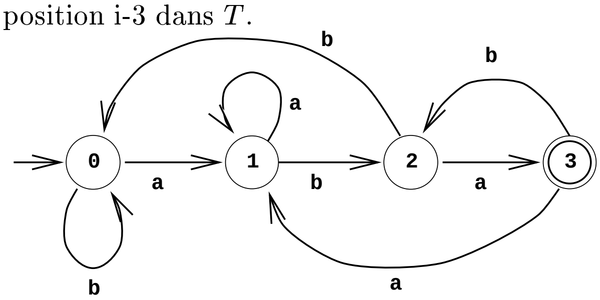
Algorithmique
Dernier TD. Semaine du 14/01/2002

Exercice 1

On considère l'automate de la figure suivante. Montrer que si on exécute cet automate sur un texte T, alors l'automate est dans l'etat 3 après la lecture des i premières lettres de T si et seulement si le motif `aba` apparait en position i-3 dans T.



Exercice 2 : Programmation de KMP en Java

On considère la méthode suivante en Java :

```
static int[] prefixe(char[] motif){
    int m = motif.length, k = -1, q;
    int[] pref = new int[m];
    pref[0] = -1;
    for (q = 1; q < m; q++) {
        while (k >= 0 && motif[k+1] != motif[q])
             k= pref[k];
        if (motif[k+1] == motif[q]) k++;
        pref[q] = k;
    }
    return pref;
}
```

1) Quel est le tableau renvoyé pour le motif `ababba` ?

Étant donnés deux tableaux de caractères `tab` et `motif`, l'algorithme de recherche de Knuth-Morris-Pratt, détectant chacune des occurences de motif dans tab, est défini par le pseudo-code suivant qui utilise la méthode prefixe[1] :

```
KMP(tab, motif)
  n := longueur(tab)
  m := longueur(motif)
  pref := prefixe(motif)
  q :=  -1
  pour i allant de 0 \'a n-1 faire :
       tant que q >= 0 et motif[q+1] != tab[i] faire
            q := pref[q]
       si motif[q+1] == tab[i] alors q := q + 1
       si q == m-1 alors :
            imprimer "occurence \'a la position" i + 1 - m
            q := pref[q]
```

2) Implémenter cet algorithme en Java.

Exercice 3

Implémenter en Java la méthode suivante

```
static int occCar(char[] motif, char c, int k)
```

renvoyant la position de la dernière occurrence de `c` dans `motif` avant la position `k` si cette occurrence existe, et -1 sinon.

Exercice 4

On considère deux tableaux de caractères `tab` et `motif`. `tab` est de longueur `n` et `motif` est de longueur `m`. On cherche à savoir si `motif` a une occurrence dans `tab`. On place la dernière lettre de `motif` face à la lettre de position `i` dans `tab`.

On suppose que les mots `tab[0..i]` et `motif[0..m-1]` ont un plus grand suffixe commun de longueur `k`. C'est à dire que les tableaux `tab[i-k+1..i]` et `motif[m-k..m-1]` sont égaux mais que `tab[i-k] != motif[m-k-1]`.

1. De combien de lettres peut-on décaler `motif` en toute sécurité si `occCar(motif, tab[i-k], m-k-1)` est égal à `d` ?

2. En déduire une version améliorée de l'algorithme naïf utilisant la fonction `occCar`.

[1] Remarquons que `prefixe(motif)` renvoie un tableau `tab` tel que `tab[j]` soit égal au plus grand $k < j$ tel que $l_0 \dots l_k$ soit suffixe de $l_0 \dots l_j$ si celui-ci existe, et -1 sinon.