```
 1 # table de programmation dynamique, complexite: O( m m )
 2 def positions(P, T, k):
 3     m = len(P)                                                  # 1
 4     n = len(T)                                                  # 1
 5     table = []                                                  # 1
 6     results = []                                                # 1
 7     # Construit une table de programmation dynamique vide
 8     for i in range(m + 1 + 1):                                  # (m + 2) {
 9         table.append([0])                                       #            1
10         for j in range(n + m + 1):                              #            (n + m + 1) {
11             table[i].append(0)                                  #                              1
12                                                                 #                          }}
13     # remplit un tableau de programmation dynamique vide
14     for i in range(1, n + m + 1):                               # (n + m) {
15         for j in range(1, m + 1):                               #            (m +1) {
16             if T[(i-1) % n] == P[j-1]:                          #                          1
17                 table[j][i] = 1 + table[j-1][i-1]               #                          1
18                                                                 # }}
19     # trouve les séquences correspondantes dans la
20     # table de programmation dynamique
21     for i in range(m - k, m + 1):                               # (m - k) {
22         for j in range(n + m + 1):                              #       (n + m + 1) {
23             if table[i][j]>=i and table[i+1][j+1]<i+1:          #                          1
24                 print('found')                                  #                          1
25                 # déplace la cles pour le erreurs
26                 error_cor = (m-i)                               #                          1
27                 beg = ((j - m) % (n - 1)) + 1 + error_cor       #                          1
28                 end = (j % (n - 1)) + 1 + error_cor             #                          1
29                 temp = [beg, end]                               #                          1
30                 results.append(temp)                            #                          1
31                                                                 # }}
32                                                                 #time = -7km-7kn-7k+10m^2+10mn+13m+4n+8
33     for i in table:
34         print(i)
35
36     print(results)
37     return results
38
39 #unit testing
40 P = ['A', 'A', 'G', 'T']
```

```
41 T = ['G', 'A', 'T', 'A', 'A', 'G', 'T', 'A', 'A']
42 k = 1
43 positions(P, T, k)
```