

Demo ticket

Session

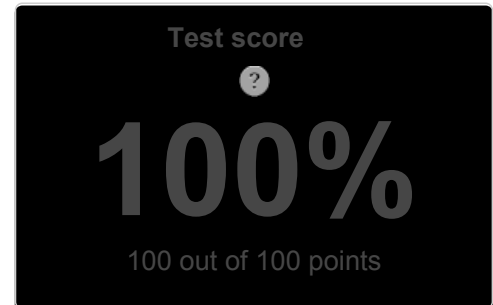
ID: demoRMTRVK-BE6
Time limit: 120 min.

Status: closed

Created on: 2014-03-17 19:22 UTC
Started on: 2014-03-17 19:22 UTC
Finished on: 2014-03-17 20:39 UTC

Tasks in test

Task score



MEDIUM

1. GenomicRangeQuery

Find the minimal nucleotide from a range of sequence DNA.

score: 100 of 100



Task description

A non-empty zero-indexed string S is given. String S consists of N characters from the set of upper-case English letters A, C, G, T . This string actually represents a DNA sequence, and the upper-case letters represent single nucleotides.

You are also given non-empty zero-indexed arrays P and Q consisting of M integers. These arrays represent queries about minimal nucleotides. We represent the letters of string S as integers 1, 2, 3, 4 in arrays P and Q , where $A = 1, C = 2, G = 3, T = 4$, and we assume that $A < C < G < T$.

Query K requires you to find the minimal nucleotide from the range $(P[K], Q[K])$, $0 \leq P[i] \leq Q[i] < N$.

For example, consider string $S = GACACCATA$ and arrays P, Q such that:

```
P[0] = 0    Q[0] = 8
P[1] = 0    Q[1] = 2
P[2] = 4    Q[2] = 5
P[3] = 7    Q[3] = 7
```

The minimal nucleotides from these ranges are as follows:

- (0, 8) is A identified by 1,
- (0, 2) is A identified by 1,
- (4, 5) is C identified by 2,
- (7, 7) is T identified by 4.

Write a function:

```
def solution(S, P, Q)
```

that, given a non-empty zero-indexed string S consisting of N characters and two non-empty zero-indexed arrays P and Q consisting of M integers, returns an array consisting of M characters specifying the consecutive answers to all queries.

The sequence should be returned as:

- a Results structure (in C), or
- a vector of integers (in C++), or
- a Results record (in Pascal), or
- an array of integers (in any other programming language).

Solution

Programming language used: Python

Total time used: 76 minutes

Effective time used: 76 minutes

Notes: correct functionality and scalability

Task timeline



Code: 20:39:17 UTC, py, final, score: 100.00

```
01. def solution(S, P, Q):
02.     N = len(S)
03.     M = len(P)
04.
05.     occurs = [[-1]*4 for _ in xrange(N)]
06.     mapping = {'A':1, 'C':2, 'G':3, 'T':4}
07.     for i, c in enumerate(S):
08.         if i == 0:
09.             occurs[i][mapping[c]-1] = i
10.             continue
11.         for j in xrange(4):
12.             occurs[i][j] = occurs[i-1][j]
13.             occurs[i][mapping[c]-1] = i
14.
15.     ret = [0] * M
16.     for i in xrange(M):
17.         for j in xrange(4):
18.             if occurs[Q[i]][j] >= P[i]:
19.                 ret[i] = j + 1
20.                 break
21.     return ret
22.
```

language).

For example, given the string S = GACACCATA and arrays P, Q such that:

P[0] = 0 Q[0] = 8
P[1] = 0 Q[1] = 2
P[2] = 4 Q[2] = 5
P[3] = 7 Q[3] = 7

the function should return the values [1, 1, 2, 4], as explained above. Assume that:

- N is an integer within the range [1..100,000];
- M is an integer within the range [1..50,000];
- each element of array P, Q is an integer within the range [0..N - 1];
- $P[i] \leq Q[i]$;
- string S consists only of upper-case English letters A, C, G, T.

Complexity:

- expected worst-case time complexity is $O(N+M)$;
- expected worst-case space complexity is $O(N)$, beyond input storage (not counting the storage required for input arguments).

Elements of input arrays can be modified.

Copyright 2009–2014 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

23.
24.

Analysis

Detected time complexity:
 $O(N + M)$

test	time	result
example example test	0.050 s.	OK
extreme_sinlge single character string	0.050 s.	OK
extreme_double double character string	0.050 s.	OK
simple simple tests	0.050 s.	OK
small_length_string small length simple string	0.050 s.	OK
small_random small random string, length = ~300	0.050 s.	OK
almost_all_same_letters GGGGGG..??..GGGGGG..??..GGGGGG	0.300 s.	OK
large_random large random string, length	0.530 s.	OK
extreme_large all max ranges	0.620 s.	OK

Training center