Time Remaining: 00:03:59

Ribosome

Back to Question List

Question 1 / 1

Given a string representing DNA, return a list of strings each representing one of the proteins coded for by the DNA.

(Some details of DNA and protein are ignored here, you should ignore them too)

DNA is represented as a string whose characters are all in the set [ACGT].

Proteins are represented as a string whose characters are all in the set [GAVLIMFWPSTCYNQDEKRH].

Three adjacent characters in DNA represent a codon. DNA is read one codon at a time and translated into protein. Reading starts with a start codon and stops at a stop codon.

There is one start codon, ATG, which also codes for the protein M (which means for the purposes of these problems that all proteins begin with M). ATG only acts as a start codon if it is not in between another start/stop codon pair.

There are three stop codons, TAA, TAG, and TGA. These do not code for a character in the resulting protein, they simply terminate the protein.

For example, here is a short sequence of DNA:

ACATGGTGCACCTGACTCTCATTTGAGATATAAAAAAACCATGAGATCGATGGCGCTACGCATAATATAAAAA

It is translated as follows:

(junk) M V H L T L I STOP (junk) M R S M A L R I I STOP (junk)

ACCTCCAAC-ATG-GTG-CAC-CTG-ACT-CTC-ATT-TGA-GATATAAAAAAACC-ATG-AGA-TCG-ATG-GCG-CTA-CGC-ATA-ATA-TAA-AAAAGGCCA

So given the above protein as input your program should return:

MVHLTLI

MRSMALRII

Download sample testcases as zip ['Compile & Test' will run your code against these testcases]

Pick your language C C++ Java PHP Ruby Python Perl Haskell

```java
/*
Please write complete compilable code.
Your class should be named Solution
Read input from standard input (STDIN) and print output to standard output(STDOUT).
For more details, please check http://www.interviewstreet.com/recruit/challenges/faq/view#stdio
*/

import java.io.BufferedReader;
import java.io.DataInputStream;
import java.io.FileInputStream;
import java.io.InputStreamReader;
import java.util.Arrays;
import java.util.HashMap;
import java.util.Set;
import java.util.HashSet;

public class Solution {

        public static void main(String[] args)throws Exception{

                BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
                br.readLine(); //Skip Codon
                HashMap<String,String> map = new HashMap<String,String>();

                while(!br.readLine().equalsIgnoreCase("// DNA")){
                        String[] input = br.readLine().split(" ");
                        map.put(input[0], input[1]);
                }

                String dnaString = br.readLine();
                solve(map,dnaString);
        }

        public static void solve(HashMap<String,String> map, String dna){

                String startCodon = "ATG";
                String stopCodon1 = "TAA";
                String stopCodon2 = "TAG";
                String stopCodon3 = "TGA";

                String dnaCopy = dna;

            while(dnaCopy.length() != 0){

                int start = dnaCopy.indexOf(startCodon);
                int end = minStop(dnaCopy);
                printProtein(dnaCopy.substring(start,end),map);
                dnaCopy = dnaCopy.substring(end);
            }

        }

    public static void printProtein(String dna, HashMap<String,String> map){

        for(int i = 0; i < dna.length(); i++)
            System.out.print(map.get(dna.substring(i,i+3)));

        System.out.println();
    }

    public static int minStop(String dna){
                String stopCodon1 = "TAA";
                String stopCodon2 = "TAG";
                String stopCodon3 = "TGA";
            int[] mins = new int[3];

            mins[0] = dna.indexOf(stopCodon1);
            mins[1] = dna.indexOf(stopCodon2);
            mins[2] = dna.indexOf(stopCodon3);

            Arrays.sort(mins);
            return mins[0];
    }


        /*
        public static HashMap<String, String> proteinHash(){

        }
        */

}
```

Keyboard Shortcuts Available

Compile & Test | Submit Answer | Previous Question | Skip Question

TestCase #0

Status: Failed (Unexpected Error)

Your output:

Expected output:

```
MVHLTLI
MRAMALRII
```

TestCase #1

Status: Failed (Unexpected Error)

Your output:

```
nullCnullHInullnullnullGAPnullnullGnullYTPnullnullSALFFnullCVFSPPL
```

Expected output:

```
MHWARYPAFVPFSEG
MVIAVSCVKLLSAHNSTQHTSRKHKV
MSELTHINCVALTARFPVGKPVVPAALMNRPTRGERRFAYWALFRFLAH
M
M
MRLSKRIFT
MKF
MSKLGLTVTNA
MIPRDPRSPAPDLSAINQPAGRAERRSGPATLSASIQSINCCREARVSSSPVNSLRNVVAIATGIVVSRSSFGMASFSSGSQRSRRVT
MLCKKAVSSFGPPIVVRSKLAAVLSLMVMAALHNSLTVMPSVRCFSVTGEYSTKSF
MRRPSCSCPASIRDNTAPHSRTLKVLIIGKRSSGRKLSRILPLLRSSSM
MPQKRE
MLNTHTLPFSILLKHLSGLLSHERIHI
MYLEK
MAHYVNHHPNQVFWGRGAVKH
MRRYRARPIRHSGCATVGKGDRCGPLRYYASRP
MAYMDSCQFNMVDLDLCQFNMAYMDSCQFNMVDLDPSQFNMADLAPCQFNMADLALCQLGRGLLGTVPSLRRGLGPVPSPPY
MVPIMAAILAICQDQYIGNIQYGPMPIWLLARFNTMYWPYAI
MGFPIDVDSPSQWAVPYTIYGAS
MVSIYGLSY
MGGPIDVYGASPIDVNYGKWPAWLNAH
MGWLIAHSYPFSRPLLTSMTVNGPLGSTSISINSNLASTLLLEGRQGTLAVLPLTSMAVNGPRWLPSTSPLTSMGRGNDANGRSIDVNGR
MGGLYKQCSFREPPFCLGTSEQA
MPEPAKSAPAPKKGSKKAVTKAQKKGGKKRKRSRKESYSIYVYKVLKQVHPDTGISSKAMGIMNSFVNDIFERIAGEASRLAHYNKRSTITSREIQTAVRLLLPGELA
MIRYIDEFGQTTTRMQ
MLYL
MLLLYL
MFQVQGEVWEVF
```

Powered by interview street