Hyperion Development

Task: Capstone Project IV - Lists, Functions and String handling

www.hyperiondev.com

Introduction

Welcome to the Final Capstone Project!

Welcome to the last task and congratulations for making it this far! This task will consolidate the knowledge that you've learnt across various tasks. Now that you are comfortable with Python, we can start working on some complex applications related to interesting fields, such as Bioinformatics. This Task will focus less on teaching you Python, and more on showing you applications and writing useful programs using text files.

Connect with your mentor

CONNECT

Remember that with our courses - you're not alone! You can contact your mentor to get support on any aspect of your course.

The best way to get help is to login to www.hyperiondev.com/support to start a chat with your mentor. You can also schedule a call or get support via email.

Your mentor is happy to offer you support that is tailored to your individual career or education needs. Do not hesitate to ask a question or for additional support!

You are about to read a bit about Bioinformatics. This information covers some basic biology. Why are you being asked to read about biology if you want to become a software developer? Simply because as a programmer you will write code, not just for the sake of writing code, but to solve real-world problems. The field of Bioinformatics demonstrates very well how simple code constructs (such as the ones you have been using) can be used to solve very complex and important real-world problems. This task therefore, will give you the opportunity to apply the programming skills you have learned to solve real-world problems.

An introduction to Bioinformatics

Bioinformatics deals with methods for storing, retrieving and analysing biological data. This includes DNA and protein sequences, structures, functions, pathways and genetic interfaces. It generates new knowledge about drug design and the development of new software tools to support this.

Bioinformatics is a large field. It deals with algorithms, databases and information systems, web technologies, artificial intelligence and soft computing. It also uses information and computation theory, structural biology, software engineering, data mining, image processing and modelling and simulation. Finally, bioinformatics makes use of signal processing, discrete mathematics, control and system theory, circuit theory and statistics.

In this task, we will focus on using Python to solve a problem known as sequence alignment.

Sequence Alignment

In bioinformatics, sequence alignment is a way of arranging the sequences of DNA, RNA or protein to identify regions of similarity that may be a consequence of functional, structural or evolutionary relationships between the sequences.

Let's start with some basic definitions:

DNA – The structure of DNA was discovered in a lab at the University of Cambridge less than 100 years ago. DNA is a chemical structure found in almost every cell and contains the genetic instructions used in determining the development and functioning of all known living organisms. That's pretty impressive! Please read http://en.wikipedia.org/wiki/DNA if you want more information.

DNA is actually very simple. DNA is a long, chemical, string-like structure. A portion of DNA can be broken down into sub-particles called nucleotides (chemical elements). Nucleotides are smaller molecules that, when joined up, form the complete string of DNA.

There are only 4 different types of nucleotides in a DNA string:

Adenine which is just represented by an **A**.
Cytosine which is just represented by a **C**.
Guanine which is just represented by a **G**.
Thymine which is just represented by a **T**.

DNA is just a long String of **A**'s, **C**'s, **G**'s and **T**'s aka, nucleotides. The entire DNA strand is a PATTERN of the different combinations of these nucleotides. They are held together by a structure that makes the whole DNA structure form a double helix: the iconic DNA 'shape'.

DNA is said to be genetic 'code' because it is the 'programming language' of all living things! Within a cell, all instructions are read from that cell's particular DNA. Portions of the DNA are also directly responsible for genes. Genes control eye colour, hair colour and many of the traits that we don't even understand yet. The DNA of a human is an enormously long string of nucleotides. It has been decoded in a massive project called the human genome project. For more information on this project, you can view the details here: http://en.wikipedia/wiki/Human_genome_project. The act of determining the exact pattern of a DNA strand is known as **sequencing** and the entire string is known as the **genome**. A fully sequenced human genome would look something like this:

ACGTAAAAGGTCATACGGATCA..... (essentially a VERY long string containing the values A C G and T).

How does DNA work?

The combination of **A**'s, **C**'s, **G**'s, and **T**'s on the strand read determines the type of **amino acid** produced.

Different combinations of amino acids create different proteins which are responsible for absolutely everything in, not only the human body but in the bodies of all other living creatures too. Every group of three nucleotides (**A**, **C**, **G** or **T**) is known as a **codon** and one codon corresponds exactly to one amino acid. A table of all codons and their

corresponding amino acids can be viewed at http://www.cbs.dtu.dk/courses/27619/codon.html.

The amino acids are thus formed in an order depending on the order of the nucleotides in the base DNA sequence that was read. The order of the amino acids influences the type of protein that is created. There are many proteins, some consisting of twenty amino acids, some consisting of less.

Instructions

First read 'example.py'. Open it using IDLE or another editor of your choice.

- 'example.py' should help you understand some simple Python. Every task will have example code to help you get started. Make sure you read all of 'example.py' and try your best to understand.

- You may run example.py to see the output. Feel free to write and run your own example code before doing the Task to become more comfortable with Python.

- You are not required to read the entirety of the additional reading, it is purely for extra reference.

Note: You have reached a milestone in your Python learning, as you're now beginning to create some genuinely useful programs!

Compulsory Task 1

Follow these steps:

- Visit the website: http://www.cbs.dtu.dk/courses/27619/codon.html

- Note the 'SLC' code for each Amino Acid.

- Create a program called, 'SickleCellDisease.py'. You will simulate the effects of the Single Nucleotide Polymorphism that leads to this genetic disease.

- Write a function called 'translate' that, when given a DNA sequence of arbitrary length, the program identifies and returns the amino acid sequence of the DNA using the amino acid SLC code found in that table.
  E.g DNA Input: ATTATTATT
  Output: `III` (representing: Isoleucine, Isoleucine, Isoleucine)

- There are many different amino acids, so this may get a bit repetitive. Just do the first five Amino Acids (i.e I, L, V, F M) and make any other codon be printed as the amino acid 'X' . So basically, you would use an if - elif - elif .... else structure to translate each codon of DNA into the correct Amino Acid.

- Note that the program must be able to handle DNA sequences that are not of a length divisible by 3.

- Hint:
  len(DNA) - (Will return the length of a String)
  DNA[0:3] - (Will get the first 3 characters of the string stored in DNA num = 3)
  DNA[0:num] - (This will work too!)

Compulsory Task 2

Follow these steps:

- Add another function to the program SickleCellDisease.py called 'mutate'. This function must read the contents of the text file named 'DNA.txt'. It must then identify the first occurrence of the lowercase letter 'a' in 'DNA.txt'.
- You must then write two new text files, one named normalDNA.txt and the other named mutatedDNA.txt.
- The file normalDNA.txt must have the same DNA sequence as DNA.txt with the 'a' changed to an 'A'.
- The file mutatedDNA.txt must have the same DNA sequence as DNA.txt with the 'a' changed to a 'T'.
- Now create a new function, 'txtTranslate', that calls the translate function that you wrote in Task 1, to take in text file input.
- Call it on both mutatedDNA.txt and normalDNA.txt, and output both Amino Acid sequences to the user.

Give your thoughts..

RATE

Hyperion strives to provide internationally-excellent course content that helps you achieve your learning outcomes. Think the content of this task, or this course as a whole, can be improved or think we've done a good job?

Click here to share your thoughts anonymously.