Universidade Federal do Rio de Janeiro
Centro de Ciências da Saúde
Instituto de Biofísica Carlos Chagas Filho

Disciplina: CFB017 - Programação para Biociências
Professor: Dr. Vitor Lima Coelho

Trabalho de Avaliação Contínua - nº 1

1 - Modifique o código 3 para ler o arquivo FASTA contendo apenas a sequência de aminoácidos do gene "TcCLB.506717.80" e imprimir apenas o nome do gene que codifica a proteína e a sequência separados por tabulação. Por exemplo, gostaríamos que a primeira coluna referente a ela tivesse apenas o identificador "TcCLB.506717.80" e na segunda coluna a sequência.

- Arquivo de entrada: arquivo FASTA contendo a sequência de aminoácidos de TcCLB.506717.80:mRNA-p1

```
# open("caminho completo para o arquivo que vai ser analisado").
refArquivo = open("/home/carol/Documents/CFB017/gabarito_gene")
# criação de variáveis.
cabeçalho = ""
sequência = ""
# o readlines() gera uma LISTA com todas as linhas.
for line in refArquivo.readlines():
# encontrar cabeçalho.
    if ">" in line:
# o split vai fatiar a lista dentro da variável 'line' com o separador ':'.
# utilizamos apenas o primeiro item da lista([0]).
        gene_nome = line.split(':')[0]
# adiciona a linha atual à variável.
# o .replace deleta o caractére '>'.
        cabeçalho = gene_nome.replace(">",'')
    else:
# adição em loop das linhas de sequencia (tem mais de uma por causa das quebras do texto).
# line.replace retira as quebras de linha/new lines(\n).
        sequência += line.replace('\n',"")
#  '\t' é o símbolo para a tabulação (tab) que age como separador das colunas.
    print(cabeçalho,'\t',sequência)
refArquivo.close()
```

2 - Modifique o código 3 para ler o arquivo multi-FASTA abaixo e imprimir cabeçalho e sequência de todas as proteínas.

- Arquivo de entrada:
  TriTrypDB-47_TcruziCLBrenerEsmeraldo-like_AnnotatedProteins.fasta

```
# open("caminho completo para o arquivo que vai ser analisado")
refArquivo = open("/home/carol/Documents/CFB017/TriTrypDB_Proteins.fasta")
# criação de variáveis
cabeçalho = ""
sequência = ""
# o readlines() gera uma LISTA com todas as linhas
# line é uma variável que a cada loop recebe como valor uma lista que contém uma linha do arquivo.
for line in refArquivo.readlines():
# busca por cabeçalhos.
   if ">" in line:
# perguntar se sequência não está vazia
      if sequência != "":
         print(cabeçalho)
         print(sequência)
# a variável sequência deve ser zerada dentro do loop para não acumular sequências de outros cabeçalhos
         sequência = ""
# sem o .strip, o print abaixo gera uma linha em branco entre o cabeçalho e a sequência
# adiciona a linha atual à variável
      cabeçalho = line.strip('\n')
   else:
# concatenação de sequências de AA
# deleção das quebras de linha (\n)
      sequência += line.strip('\n')
# precisa realizar o print novamente para imprimir os últimos itens
print(cabeçalho)
print(sequência)
refArquivo.close()
```

3 - Escreva o código 5 utilizando a biblioteca **csv**. Caso não tenha instalado, verifique o material "Configurando o ambiente de trabalho" para instalar pacotes no PyCharm. Para saber se o módulo está instalado, execute o comando import csv. Se não estiver instalado será impressa uma mensagem de erro.

- Arquivo de entrada: species.csv

```
# importar biblioteca csv.
import csv
# open("caminho completo para o arquivo que vai ser analisado").
with open("/home/carol/Documents/CFB017/species.csv") as csvfile:
# leitura e identificação do separador (já age como um "split(,)").
    all_species = csv.reader(csvfile, delimiter=',')
    for lines in all_species:
# a função .upper() deixa os caracteres da string em caixa alta.
# a função .rstrip() remove espaços em branco (default) no final da string.
        if lines[3].upper().rstrip() == "BIRD":
# o operador '*' retorna apenas os elementos da lista 'lines'.
# sep="\t" define tab como separador.
            print(*lines,sep="\t")
```

4 - Em relação o código 5:

a) qual a finalidade da função upper?

A função '.upper()' deixa todos os caracteres da string em caixa alta. O arquivo de entrada 'species.csv' possui caracteres minúsculos e maiúsculos. Para facilitar a busca pelo padrão, pode-se deixar toda a string daquele fragmento (data[3]) com todas as letras maiúsculas ou minúsculas.

b) qual a finalidade das função rstrip?

A função '.rstrip()' deleta os caracteres que estão posterior à string de interesse. O que você quer deletar deve estar especificado dentro dos parênteses. I.e., .rstrip('\t') deleta tabulações enquanto .rstrip('qacd') remove os 'q', os 'a', os 'c' e os 'd' que estiverem no final da sua string. A opção defaul exclui qualquer coisa em branco, podendo ser um espaço, uma quebra de linha ('\n'),ou uma tabulação ('\t').

c) Caso não fosse incluída a função upper, cite uma situação onde um registro do arquivo species.csv poderia não ser impresso para o usuário.

`if data[3].rstrip() == "BIRD"` ou `if data[3].rstrip() == "bird"`.Não utilizando a função .upper() - ou .casefold() pra deixar a string toda em minúsculo - podemos ver que o padrão não é encontrado. Ele só seria encontrado sem as funções citadas anteriormente, caso soubermos como a string está escrita ('Bird'). Ou seja, em `if data[3].rstrip() == "Bird"`, conseguimos obter o mesmo output que o código 5 inalterado

d) Caso não fosse incluída a função rstrip, cite uma situação onde um registro do arquivo species.csv poderia não ser impresso para o usuário.

`if data[3].upper() == "BIRD"` não retorna nenhum valor porque o arquivo species.csv possui um '\n' no final de cada linha, indicando que gera uma nova linha abaixo (como se fosse um 'enter'). Uma forma de testar isso é colocar diretamente .rstring('\n') - `if data[3].upper().rstrip('\n') == "BIRD"` - no código. Teremos o mesmo output. Se pesquisarmos pelo padrão incluindo o '\n' como em `if data[3].upper() == "BIRD"'\n'`, também obtemos o mesmo output que o código 5 inalterado.

- Arquivo: species.csv