# **CEN0336 - Introdução a Programação de Computadores Aplicada a Ciências Biológicas**

# **Prova II**

Nome: Carolina Pacchioni Monteiro

No USP: 10782878

## **Instruções Gerais**

Vocês terão desde o dia 7 de dezembro as 18:00 e até o dia 21 de dezembro as 23:59 para resolver as atividades da prova.

São 4 perguntas/atividades, cada uma precisa de uma resposta neste arquivo de word, que tem que ser carregado no sistema e-disciplinas até o dia e horário limite. Além disso algumas perguntas precisam que vocês desenvolvam scripts e carregam num repositório GitHub específico para a prova (indicado na pergunta 1). No dia 22 de dezembro as 00:01 o professor descarregará todos os repositórios para avaliar seus scripts. Os critérios de avaliação dos scripts são:

Eles têm que rodar sem erros de sintaxe (15%)

Eles têm que ser logicamente corretos (20%)

Eles têm que ter comentários explicando o que é feito em cada passo (20%)

Eles têm que ter controle da entrada dos usuários (10%)

Eles têm que ter controle de erros usando blocos try/except (20%)

Eles têm que implementar o uso de funções (15%)

## **Perguntas/Atividades**

1. Crie um repositório no GitHub para esta prova. Escreva o endereço completo do seu repositório GitHub como resposta desta pergunta.

Lembre-se que o repositório tem que ter o nome segundo o padrão:

P2\_NNNNNNN-CEN0336

Onde NNNNNNN é seu número USP.

A resposta tem que ter o endereço completo, iniciando com https://"

O professor irá clonar seu repositório, com as respostas para os outros pontos da prova.

**RESPOSTA:** https://github.com/kazinha21/P2\_10782878\_CEN0336.git

1. Crie um script de python com o nome script\_getORF.py que realize as seguintes operações:
2. Ler um arquivo multifasta da linha de comandos, usando o modulo sys.
3. Para cada registro fasta identificar o Open Reading Frame (Fase de Leitura Aberta: <https://pt.wikipedia.org/wiki/Fase_de_leitura_aberta> ou <https://en.wikipedia.org/wiki/Open_reading_frame>) mais comprido dentre as 6 fases de leitura possíveis. Os ORFs neste caso não têm íntrons. Repare que uma fase de leitura aberta tem que iniciar com um códon de início da tradução e terminar com um códon de terminação da tradução (<https://pt.wikipedia.org/wiki/C%C3%B3digo_gen%C3%A9tico>), é seu comprimento tem que ser múltiplo de três (3)
4. Daquele ORF mais comprido identificado, gerar o peptídeo codificado.
5. Imprimir o ORF mais comprido no arquivo ORF.fna. O formato da sequência tem que ser fasta, e seu identificador, tem que ser o identificador original, adicionando o string: “\_frameX\_START\_END”, onde X é a Fase de leitura (1,2,3,4,5 ou 6), e START e END são as coordenadas na sequencia
6. Imprimir o peptídeo correspondente ao ORF mais comprido no arquivo ORF.faa. O formato da sequência tem que ser fasta, e seu identificador, tem que ser o identificador original, adicionando o string: “\_frameX\_START\_END”, onde X é a Fase de leitura (1,2,3,4,5 ou 6), e START e END são as coordenadas na sequencia
7. Não esquece que tem que processar todos os registros do arquivo fasta original, i.e., tem que ter o mesmo número de entradas nos arquivos ORF.faa, ORF.fna que no arquivo original.
8. Carregue seu script, e os arquivos ORF.faa e ORF.fna no seu repositório.
9. Calcule o md5sum do seu script e coloque esse número como resposta desta pergunta.

**RESPOSTA: md5sum:** 6f1f839a07b3ea478374a6cdb5ee2feb

**Link para script\_getORF.py:** https://github.com/kazinha21/P2\_10782878\_CEN0336/blob/main/script\_getORF.py

**Link para ORF.fna:** https://github.com/kazinha21/P2\_10782878\_CEN0336/blob/main/ORF.fna

**Link para ORF.faa:**

https://github.com/kazinha21/P2\_10782878\_CEN0336/blob/main/ORF.faa

1. Explique a diferença entre os dois sets:

set1=set('accggttaata')  
set2={'accggttaata'}

**RESPOSTA:** Ao rodar um script em python que contenha esses comandos a resposta será a seguinte, respectivamente: {'a', 'c', 'g', ‘t’} e {'accggttaata'}. Isso porque a função set é responsável por eliminar os elementos duplicados na criação de um conjunto. Dessa forma, no primeiro caso, como essa função foi utilizada, cada caractere da lista será considerado como um item e será analisado para incorporar o conjunto final, o que elimina as duplicatas do conjunto de dados e explicita o resultado: {'a', 'c', 'g', ‘t’}. Já no segundo caso, como a função set não foi utilizada, todos os caracteres dentro das aspas serão considerados como um item único, permitindo a adição de itens duplicados ao conjunto final, explicitando o resultado: {'accggttaata'}.

A diferença entre esses dois sets demonstra que as possíveis formas de se criar um conjunto são de grande importância e refletem diretamente em seu resultado final.

1. Descreva com suas próprias palavras o que o seguinte pseudocódigo faz. Faz uma avaliação desse pseudocódigo e relate quaisquer problemas que possa ter.

Defina TOTAL com zero

Defina CONTADOR NOTAS como zero

While CONTADOR NOTAS é menor ou igual a dez

      Entre (input) a seguinte nota

      Some a nota ao TOTAL

Defina a média da disciplina como o TOTAL dividido por dez (TOTAL/10)

Imprima na tela a média da disciplina

Implemente esse pseudocódigo num script de python e carregue ele no seu repositório da prova com o nome notas.py. Calcule o md5sum do seu script e coloque esse número também como resposta desta pergunta.

**RESPOSTA:** Descrevendo o script: Esse script pede 10 inputs (notas) a serem inseridas pelo usuário através do terminal para cálculo da média final das 10 notas.

Possíveis erros que podem ser relatados: 1. Não foi utilizada nenhuma transformação do tipo de input a ser inserido (número inteiro, número flutuante ou string). Dessa forma, ocorrerá um erro no cálculo da média final, pois qualquer tipo inserido será considerado uma string em python. Portanto, deveria ser acrescentado no script um comando para transformação do tipo de input para número flutuante (assim somente ocorrerá um erro se for inserido uma string).

2. O loop definido é infinito, uma vez que a variável CONTADOR NOTAS sempre terá o mesmo valor (0). Além disso, o loop aceita valores menores ou iguais a 10, somando um total de 11 valores. Dessa forma, como a média é calculada somente para 10 valores, ocorrerá um erro no cálculo da média final.

**md5sum:** bc5f3d1d2748783bb2cc4589f5d09019

**Link para notas.py:** https://github.com/kazinha21/P2\_10782878\_CEN0336/blob/main/notas.py