Luiza Schmidt 281954 Ciência da Computação

1a) A subsequência ACCTGGTGTTCCCA contém uma mutação. Identifique o(s) potencial(is) candidatos a sequência original, informando a localização da subsequência, bem como a mutação identificada.
Entrada: sequência do cromossomo 21

```
if __name__ == '__main__':
    f = open(sys.argv[1])

    s = f.read().replace("\n","") #s contém seq do cromossomo 21
```

Saída: sequência não mutada, região de localização da subsequência, mutação existente.

Na saída do programa são impressas 13 strings, que são palíndromos, e, consecutivamente, a quantidade de vezes que eles aparecem na string de entrada, no formato "**s : n**" sendo **s** string do palíndromo e **n** número de vezes.

```
ACCTGGTTTTCCCA posição inicial: 5082790
T posição caracter original: 5082797
ACCTGGTGTTCCGA posição inicial: 6467967
G posição caracter original: 6467979
AGCTGGTGTTCCCA posição inicial: 10038319
G posição caracter original: 10038320
ACATGGTGTTCCCA posição inicial: 10399167
A posição caracter original: 10399169
ACCTGGTGATCCCA posição inicial: 20966057
A posição caracter original: 20966065
ACCTAGTGTTCCCA posição inicial: 35162825
A posição caracter original: 35162829
ACTTGGTGTTCCCA posição inicial: 35745840
T posição caracter original: 35745842
ACCTGGTGTTCCTA posição inicial: 36705952
T posição caracter original: 36705964
ACCTGGTGTTTCCA posição inicial: 37453236
T posição caracter original: 37453246
ACCTGGTGCTCCCA posição inicial: 42746544
C posição caracter original: 42746552
ACCTGGTGTTCCGA posição inicial: 43076289
G posição caracter original: 43076301
ACCTGCTGTTCCCA posição inicial: 43703887
C posição caracter original: 43703892
ACCTGGAGTTCCCA posição inicial: 44397996
A posição caracter original: 44398002
```

Para encontrar os palíndromos e suas quantidades, primeiramente foi criada uma variável onde a string da sequência que continha uma mutação foi salva, para que então fosse criado um laço *for* comparando pedaços de tamanho 14 da string de entrada (a sequência do cromossomo 21), utilizando a função "compara" para tal.

A função *compara* compara a string da mutação com uma string de tamanho 14.

Se o conteúdo das posições for igual então o contador "eq" é incrementado, se forem diferentes o caracter da string que não é fixa é salvo na variável "carac".

Após terminado o laço *for* retorna o valor *verdadeiro,* se o contador tiver valor igual a 13, e o caracter divergente, se o contador tiver valor diferente de 13 retorna o valor *falso* e uma string vazia.

```
subs = 'ACCTGGTGTTCCCA' # sequência que contém uma mutação

for i in range(len(s)-14):
    ts = s[i:i+14]
    torf, carac = compara(ts, subs)
    if torf:
        print(ts + ' posição inicial: ' + str(i)) #imprime todos os candidatos à seq original
        print(ts[carac] + ' posição caracter original: ' + str(i + carac))
```

Pseudocódigo:

S = sequência do cromossomo 21
Subs = ‘ACCTGGTGTTCCCA’

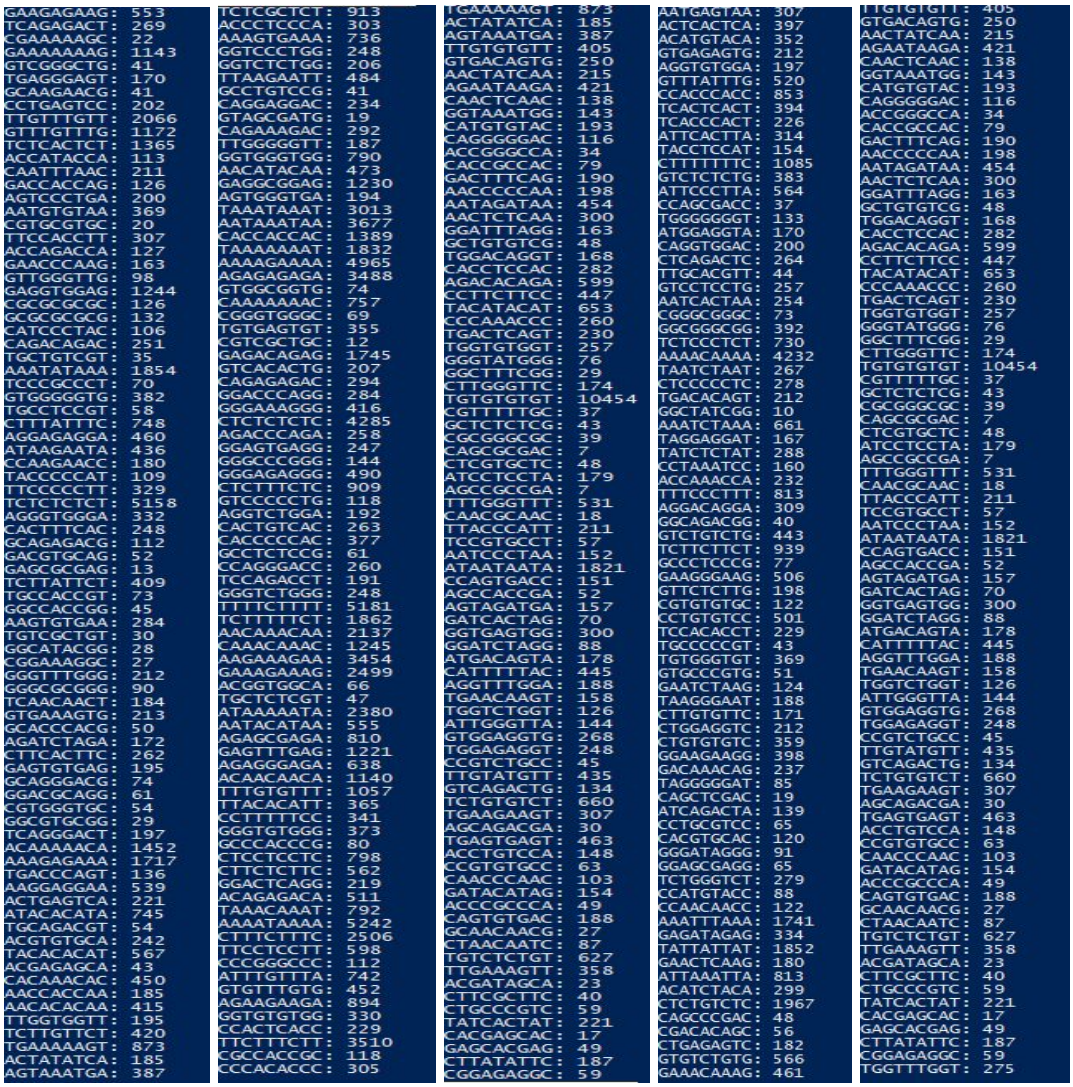
Do início de **S** até o final de **S**
    Divide **S** em janelas de 14 em 14, avançando de uma em uma unidade
    Compara com **subs**, caractere a caractere
        Se o caractere da vez for igual
            Incrementa o contador
        Senão
            Salva o caractere diferente
    Verifica se o contador é igual a 13
        Se sim
            Retorna Verdadeiro e o caractere diferente
        Senão
            Retorna Falso e uma string vázia
Imprime a sequência original e a posição inicial

1b) Um palíndromo é uma palavra, frase ou qualquer outra sequência de unidades (como uma cadeia de DNA, Enzima de Restrição, etc.) que tenha a propriedade de poder ser lida da mesma maneira da direita para a esquerda e vice-versa. Determine quantos diferentes palíndromos de tamanho 9 podem ser encontrados.
Entrada: sequência do cromossomo 21

```
if __name__ == '__main__':
    f = open(sys.argv[1])
    s = f.read().replace("\n","") #s contém seq do cromossomo 21
```

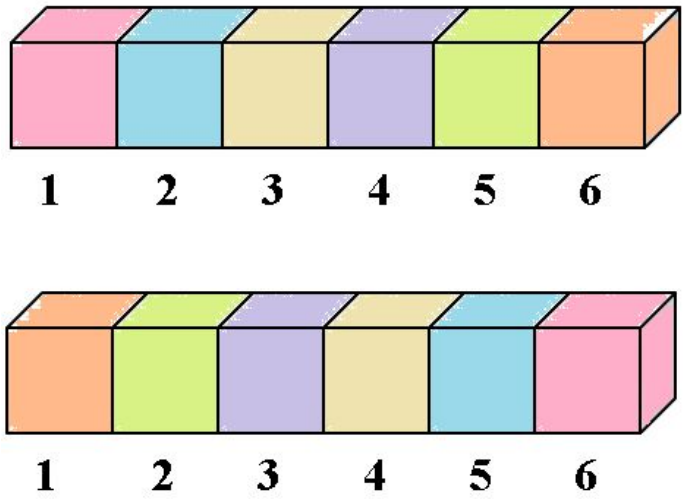
Saída: quantidade de palíndromos de tamanho 9, número de ocorrências de cada palíndromo.

Retorna 1021 strings de palíndromos diferentes e a quantidade de vezes que cada um aparece na sequência do cromossomo.



A estratégia utilizada para montar esse programa foi salvar a sequência do cromossomo em uma string, dividí-la em pedaços de tamanho 9, invertê-la/espelhá-la e comparar a invertida com sua forma original verificando se o conteúdo de suas posições inversas são iguais e criar um dicionário de palavras para salvar os palíndromos e seu número de aparições

Posições opostas equivalentes seriam comparadas de acordo com as cores na imagem abaixo, 1 com 6, 2 com 5, 3 com 4.



Pseudocódigo:

Cria dicionário de palíndromos
Enquanto a string de entrada não tiver terminado
    Divide ela em pedaço de tamanho 9, avançando de 1 em 1
    Se a string já estiver registrada no dicionário
        Incrementa sua quantidade
    Senão
        Adiciona string ao dicionário, com valor 1
Imprime todas os palíndromos do dicionário junto com seu valor (quantidade de vezes que aparece)
Imprime a quantidade total de palíndromos de tamanho 9 contidas na sequência

1c) Identifique as diferentes subsequências de tamanho 37, contabilizando o número de ocorrências de cada uma delas.
Entrada: sequência do cromossomo 21.

```
if __name__ == '__main__':
    f = open(sys.argv[1])
    s = f.read().replace("\n","") #s contém seq do cromossomo 21
```

Saída: lista das diferentes subsequências e o número de ocorrências de cada uma delas.

```
TAAGGATGCAGGTTACTGTTTCTCACGTGCTTCAATC: 1
AAGGATGCAGGTTACTGTTTCTCACGTGCTTCAATCT: 1
AGGATGCAGGTTACTGTTTCTCACGTGCTTCAATCTT: 1
GGATGCAGGTTACTGTTTCTCACGTGCTTCAATCTTT: 1
GATGCAGGTTACTGTTTCTCACGTGCTTCAATCTTTA: 1
ATGCAGGTTACTGTTTCTCACGTGCTTCAATCTTTAA: 1
TGCAGGTTACTGTTTCTCACGTGCTTCAATCTTTAAA: 1
GCAGGTTACTGTTTCTCACGTGCTTCAATCTTTAAAT: 1
CAGGTTACTGTTTCTCACGTGCTTCAATCTTTAAATC: 1
AGGTTACTGTTTCTCACGTGCTTCAATCTTTAAATCA: 1
GGTTACTGTTTCTCACGTGCTTCAATCTTTAAATCAG: 1
GTTACTGTTTCTCACGTGCTTCAATCTTTAAATCAGA: 1
TTACTGTTTCTCACGTGCTTCAATCTTTAAATCAGAA: 1
TACTGTTTCTCACGTGCTTCAATCTTTAAATCAGAAA: 1
ACTGTTTCTCACGTGCTTCAATCTTTAAATCAGAAAA: 1
CTGTTTCTCACGTGCTTCAATCTTTAAATCAGAAAAT: 1
TGTTTCTCACGTGCTTCAATCTTTAAATCAGAAAATA: 1
GTTTCTCACGTGCTTCAATCTTTAAATCAGAAAATAA: 1
TTTCTCACGTGCTTCAATCTTTAAATCAGAAAATAAA: 1
TTCTCACGTGCTTCAATCTTTAAATCAGAAAATAAAC: 1
TCTCACGTGCTTCAATCTTTAAATCAGAAAATAAACT: 1
CTCACGTGCTTCAATCTTTAAATCAGAAAATAAACTC: 1
TCACGTGCTTCAATCTTTAAATCAGAAAATAAACTCT: 1
CACGTGCTTCAATCTTTAAATCAGAAAATAAACTCTT: 1
ACGTGCTTCAATCTTTAAATCAGAAAATAAACTCTTC: 1
CGTGCTTCAATCTTTAAATCAGAAAATAAACTCTTCT: 1
GTGCTTCAATCTTTAAATCAGAAAATAAACTCTTCTG: 1
TGCTTCAATCTTTAAATCAGAAAATAAACTCTTCTGC: 1
GCTTCAATCTTTAAATCAGAAAATAAACTCTTCTGCT: 1
CTTCAATCTTTAAATCAGAAAATAAACTCTTCTGCTG: 1
TTCAATCTTTAAATCAGAAAATAAACTCTTCTGCTGA: 1
TCAATCTTTAAATCAGAAAATAAACTCTTCTGCTGAA: 1
CAATCTTTAAATCAGAAAATAAACTCTTCTGCTGAAG: 1
AATCTTTAAATCAGAAAATAAACTCTTCTGCTGAAGT: 1
ATCTTTAAATCAGAAAATAAACTCTTCTGCTGAAGTA: 1
TCTTTAAATCAGAAAATAAACTCTTCTGCTGAAGTAC: 1
CTTTAAATCAGAAAATAAACTCTTCTGCTGAAGTACA: 1
TTTAAATCAGAAAATAAACTCTTCTGCTGAAGTACAT: 1
TTAAATCAGAAAATAAACTCTTCTGCTGAAGTACATT: 1
TAAATCAGAAAATAAACTCTTCTGCTGAAGTACATTA: 1
AAATCAGAAAATAAACTCTTCTGCTGAAGTACATTAA: 1
AATCAGAAAATAAACTCTTCTGCTGAAGTACATTAAC: 1
ATCAGAAAATAAACTCTTCTGCTGAAGTACATTAACA: 1
TCAGAAAATAAACTCTTCTGCTGAAGTACATTAACAA: 1
CAGAAAATAAACTCTTCTGCTGAAGTACATTAACAAA: 1
AGAAAATAAACTCTTCTGCTGAAGTACATTAACAAAG: 1
GAAAATAAACTCTTCTGCTGAAGTACATTAACAAAGC: 1
AAAATAAACTCTTCTGCTGAAGTACATTAACAAAGCT: 1
AAATAAACTCTTCTGCTGAAGTACATTAACAAAGCTC: 1
AATAAACTCTTCTGCTGAAGTACATTAACAAAGCTCC: 1
ATAAACTCTTCTGCTGAAGTACATTAACAAAGCTCCA: 1
TAAACTCTTCTGCTGAAGTACATTAACAAAGCTCCAT: 1
AAACTCTTCTGCTGAAGTACATTAACAAAGCTCCATG: 1
AACTCTTCTGCTGAAGTACATTAACAAAGCTCCATGA: 1
ACTCTTCTGCTGAAGTACATTAACAAAGCTCCATGAA: 1
CTCTTCTGCTGAAGTACATTAACAAAGCTCCATGAAC: 1
TCTTCTGCTGAAGTACATTAACAAAGCTCCATGAACA: 1
CTTCTGCTGAAGTACATTAACAAAGCTCCATGAACAT: 1
TTCTGCTGAAGTACATTAACAAAGCTCCATGAACATC: 1
TCTGCTGAAGTACATTAACAAAGCTCCATGAACATCT: 1
CTGCTGAAGTACATTAACAAAGCTCCATGAACATCTC: 1
TGCTGAAGTACATTAACAAAGCTCCATGAACATCTCA: 1
GCTGAAGTACATTAACAAAGCTCCATGAACATCTCAT: 1
CTGAAGTACATTAACAAAGCTCCATGAACATCTCATT: 1
TGAAGTACATTAACAAAGCTCCATGAACATCTCATTA: 1
GAAGTACATTAACAAAGCTCCATGAACATCTCATTAG: 1
AAGTACATTAACAAAGCTCCATGAACATCTCATTAGC: 1
AGTACATTAACAAAGCTCCATGAACATCTCATTAGCC: 1
GTACATTAACAAAGCTCCATGAACATCTCATTAGCCA: 1
TACATTAACAAAGCTCCATGAACATCTCATTAGCCAG: 1
ACATTAACAAAGCTCCATGAACATCTCATTAGCCAGA: 1
CATTAACAAAGCTCCATGAACATCTCATTAGCCAGAA: 1
ATTAACAAAGCTCCATGAACATCTCATTAGCCAGAAG: 1
TTAACAAAGCTCCATGAACATCTCATTAGCCAGAAGT: 1
TAACAAAGCTCCATGAACATCTCATTAGCCAGAAGTG: 1
AACAAAGCTCCATGAACATCTCATTAGCCAGAAGTGT: 1
ACAAAGCTCCATGAACATCTCATTAGCCAGAAGTGTT: 1
CAAAGCTCCATGAACATCTCATTAGCCAGAAGTGTTC: 1
AAAGCTCCATGAACATCTCATTAGCCAGAAGTGTTCA: 1
AAGCTCCATGAACATCTCATTAGCCAGAAGTGTTCAC: 1
AGCTCCATGAACATCTCATTAGCCAGAAGTGTTCACA: 1
GCTCCATGAACATCTCATTAGCCAGAAGTGTTCACAC: 1
```

Pseudocódigo:

Cria dicionário para sequências de tamanho 37
Enquanto a string de entrada não tiver terminado
    Divide ela em pedaço de tamanho 37, avançando de 1 em 1
    Se a string já estiver registrada no dicionário
        Incrementa sua quantidade
    Senão
        Adiciona string ao dicionário, com valor 1
Imprime todas as sequências do dicionário junto com seu valor (quantidade de vezes que aparece)

1d) Contabilize o número de ocorrências de cada um dos quatro nucleotídeos. Estime o GC%. Existe um caractere diferente na sequência?
Entrada: sequência do cromossomo 21.

```
if __name__ == '__main__':
    f = open(sys.argv[1])

    s = ''
    for line in f.readlines()[1:]:
        s += line  #pula essa linha
                   #>NC_000021.9 Homo sapiens chromosome 21, GRCh38.p7 Primary Assembly

    s = s.replace("\n","") #s contém seq do cromossomo 21
```

Saída: Número de ocorrências para cada nucleotídeo, GC%, “sim” ou “não” para caractere diferente e identificar, caso exista.

```
N: 6621361
G: 8226381
A: 11820664
T: 11856330
C: 8185244
M: 2
R: 1
16411625
percentual de GC: 35.13515515516244%
caracteres diferentes de 'A' 'T' 'C' ou 'G':
N
M
R
```

Pseudocódigo:

Cria dicionário para bases
Enquanto a string de entrada não tiver terminado
    Divide ela em pedaço de tamanho 1, avançando de 1 em 1
    Se o caractere já estivá registrado no dicionário
        Incrementa sua quantidade
    Senão
        Adiciona ao dicionário, com valor 1
Imprime cada caractere e seu número de ocorrências
Soma o número de ocorrências de G com C e divide pelo número total de bases
Imprime o percentual de GC
Imprime os caracteres diferentes