

**“AÑO DEL BICENTENARIO DEL PERÚ: 200 AÑOS
DE INDEPENDENCIA”.**



**UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE
AREQUIPA**

**CIENCIA DE LA COMPUTACIÓN
BIOINFORMÁTICA**

Práctica 07

Estudiantes:

Miguel Alexander, Herrera Cooper
Milagros Celia, Cruz Mamani
Yara Jeanette, Quispe Quispe
Ingrid Sally, Espinel Quispe

Docente:

Mg. Vicente Enrique, Machaca Arceda

June 2, 2021



Contents

1	Actividad 1	2
1.1	Código	2
1.2	Pruebas	5
1.2.1	Input	5
1.2.2	Output	5
2	Actividad 2	6
2.1	Código	6
2.2	Prueba 1	8
2.2.1	Input	8
2.2.2	Output	8
2.3	Prueba 2	11
3	Actividad 3	13
3.1	Código	13
3.2	Resultados	13
3.2.1	Input	13
3.2.2	Output	13
4	Conclusiones	16
5	Repositorio	17

1 Actividad 1

Implemente el algoritmo de alineamiento local utilizando el algoritmo de Smith-waterman. Evalúe sus resultados con las secuencias::

- S_1 : AGG
- S_2 : AAG

Utilice gapOpen = gapEXTEND = -5 y la siguiente matriz de sustitución:

	A	C	G	T
A	2	-7	-5	-7
C	-7	2	-7	-5
G	-5	-7	2	-7
T	-7	-5	-7	2

Table 1: Matriz de Similitud

La salida debe incluir el score matrix y todos los alineamientos posibles.

Resolución

1.1 Código

A continuación, mostramos el algoritmo implementado en Python.

```

1 import sys
2 import numpy as np
3
4 def Matriz_Sustitucion(archivo):
5     next(archivo)
6     r_c = {}
7     s = []
8     i = 0
9
10    for line in archivo:
11        if len(line.strip())!=0:
12            line=line.rstrip('\n')
13
14            r_c[line.split('\t')[0:1][0]] = i
15            i = i+1
16
17            s.append(list(map(int,line.split('\t')[1:])))
18
19    archivo.close()
20
21    return r_c, s
22
23
24 def Guardar_Resultado(F):
25     f = open(resultadoArchivo, "w")

```

```

26
27 f.write(' ')
28 for i in range(columna-1):
29     f.write(sequencial[i]+' ')
30
31 f.write('\n')
32
33 seq2_t = ' ' + sequencia2
34
35 for r in range(fila):
36     f.write(seq2_t[r])
37     for c in range(columna):
38         f.write(' ( ' + '{'}.format(F[r][c][0])+' '+'{'.format(F[r][c][1]))+')
39
40     f.write('\n')
41 f.close()
42
43 f = open(resultadoArchivo, "r")
44 print(f.read())
45
46 def Obtener_Secuencias(F, i, j, alineamiento_sequencial = "",
47     alineamiento_sequencia2 = ""):
48
49     if F[i][j][0]==0:
50         print ()
51         print (alineamiento_sequencial)
52         print (alineamiento_sequencia2)
53         return
54
55     if i > 0 or j > 0:
56
57         if (i>0 and j>0 and F[i][j][1] == 'D'):
58             alineamiento_sequencial = sequencial[j-1] + alineamiento_sequencial
59             alineamiento_sequencia2 = sequencia2[i-1] + alineamiento_sequencia2
60             i = i-1
61             j = j-1
62
63         elif (i>0 and F[i][j][1]=='U'):
64             alineamiento_sequencial = "-" + alineamiento_sequencia2
65             alineamiento_sequencia2 = sequencia2[i-1] + alineamiento_sequencia2
66             i = i-1
67
68         else:
69             alineamiento_sequencia2 = "-" + alineamiento_sequencia2
70             alineamiento_sequencial = sequencial[j-1] + alineamiento_sequencial
71             j = j-1
72
73         Obtener_Secuencias(F, i, j, alineamiento_sequencial,
74             alineamiento_sequencia2)
75
76 def Alineamiento_Local(F, i, j, maximo_total):
77
78     diag = F[i-1][j-1][0] + s[r_c[sequencia2[i-1]]][r_c[sequencial[j-1]]]
79     up = F[i-1][j][0] + d

```

```

80 left = F[i][j-1][0] + d
81 diag2 = 0
82
83 entrada_max = [diag, up, left, diag2]
84 maximo = max(entrada_max)
85
86 if maximo > maximo_total:
87     maximo_total = maximo
88
89 F[i][j][0] = maximo
90
91 if F[i][j][0]==diag:
92     F[i][j][1] = 'D'
93
94 if F[i][j][0]==up:
95     F[i][j][1] = 'U'
96
97 if F[i][j][0]==left:
98     F[i][j][1] = 'L'
99
100 if i==fila-1 and j==columna-1:
101     Guardar_Resultado(F)
102
103     major = -1000
104     s_i = 0
105     s_j = 0
106     for r in range(1, fila):
107         for c in range(1, columna):
108             if F[r][c][0]>=major:
109                 s_i = i
110                 s_j = j
111                 i = r
112                 j = c
113                 major = F[r][c][0]
114
115     Obtener_Secuencias(F, i, j)
116     Obtener_Secuencias(F, s_i, s_j)
117     print("Maximo Total", maximo_total)
118     return
119
120 if j<columna-1:
121     Alineamiento_Local(F, i, j+1, maximo_total)
122 else:
123     Alineamiento_Local(F, i+1, 1, maximo_total)
124
125
126 if __name__ == "__main__":
127
128     archivoNombre = sys.argv[1]
129     d = int(sys.argv[2])
130
131     resultadoArchivo = "resultado.txt"
132
133     archivo = open(archivoNombre, "r")
134
135
136     r_c, s = Matriz_Sustitucion(archivo)

```

```

137
138   secuencia1 = "AAG"
139   secuencia2 = "AGC"
140
141
142   columna = len(secuencia1)+1
143   fila = len(secuencia2)+1
144
145   F = np.zeros([fila, columna], dtype='i,O')
146
147   maximo_total = 0
148
149   Alineamiento_Local(F, 1, 1, maximo_total)
150
151

```

1.2 Pruebas

Ahora veremos la prueba realizada según lo solicitado en el ejercicio

1.2.1 Input

Evaluamos los resultados con las secuencias

```

1   secuencia1 = "AAG"
2   secuencia2 = "AGC"

```

1.2.2 Output

Seguidamente pasamos a ver el resultado generado :

```

D:\INGRID\unsa\2021-1\bio\practica7\prueba>python ejercicio1.py substitution_matrix.txt -5
      A      A      G
( 0 0) ( 0 0) ( 0 0) ( 0 0)
A ( 0 0) ( 2 D) ( 2 D) ( 0 0)
G ( 0 0) ( 0 0) ( 0 0) ( 4 D)
C ( 0 0) ( 0 0) ( 0 0) ( 0 0)

AG
AG

A
A
Maximo Total 4

```

Figure 1: Matriz Final - Alineaciones - Puntajes

2 Actividad 2

Utilice el algoritmo implementado anteriormente, pero esta vez con secuencias reales. Se recomienda utilizar las secuencias de ADN.

Utilice $gapOpen = gapEXTEND = -5$, $identicalMatch = 2$ y $mismatch = -2$.

Puede utilizar estas bases de datos:

- NCBI virus
- Virus pathogen resource
- Castor

Resolución

2.1 Código

A continuación, mostramos el algoritmo implementado en Python.

```

1 import sys
2 import numpy as np
3
4 def Alineamiento_Local(secuencia_1, secuencia_2, penalidad):
5
6     for i in range(1, fila):
7         for j in range(1, columna):
8             valor = identicalMatch
9
10            if secuencia_2[i-1] != secuencia_1[j-1]:
11                valor = mismatch
12
13            Matriz[i][j] = max(Matriz[i-1][j-1] + valor, Matriz[i-1][j] +
14                               penalidad, Matriz[i][j-1] + penalidad, 0)
15
16     print ()
17     print (Matriz)
18
19     mayor = -1000
20     i = 0
21     j = 0
22     for r in range(1, fila):
23         for c in range(1, columna):
24             if Matriz[r][c] > mayor:
25                 i = r
26                 j = c
27                 mayor = Matriz[r][c]
28
29     secuencia_alineada_1 = ""
30     secuencia_alineada_2 = ""
31
32     while (i > 0 or j > 0):

```

```

33
34     valor = identicalMatch
35     if secuencia_2[i-1] != secuencia_1[j-1]:
36         valor = mismatch
37
38     if (i>0 and j>0 and Matriz[i][j] == Matriz[i-1][j-1] + valor):
39         secuencia_alineada_1= secuencia_1[j-1] + secuencia_alineada_1
40         secuencia_alineada_2= secuencia_2[i-1] + secuencia_alineada_2
41         i = i-1
42         j = j-1
43
44     elif (i>0 and Matriz[i][j]==Matriz[i-1][j]+penalidad):
45         secuencia_alineada_1= "-" + secuencia_alineada_2
46         secuencia_alineada_2= secuencia_2[i-1] + secuencia_alineada_2
47         i = i-1
48
49     else:
50         secuencia_alineada_2= "-" + secuencia_alineada_2
51         secuencia_alineada_1= secuencia_1[j-1] + secuencia_alineada_1
52         j = j-1
53
54     if Matriz[i][j]==0:
55         break
56
57
58     print ()
59     print("Secuencia Alineada 1 :\n")
60     print (secuencia_alineada_1)
61     print ()
62     print("Secuencia Alineada 2:\n")
63     print (secuencia_alineada_2)
64     print("\n")
65     print()
66
67
68 def Obtener_Secuencia(file):
69     next(file)
70     secuencia = ""
71
72     for linea in file:
73         if len(linea.strip())!=0:
74             linea=linea.rstrip('\n')
75             secuencia = secuencia + linea
76
77     file.close()
78
79     return secuencia
80
81
82 if __name__ == "__main__":
83
84     archivo_1 = sys.argv[1]
85     archivo_2 = sys.argv[2]
86
87     penalidad = int(sys.argv[3])
88
89     identicalMatch = 2

```



```

90 mismatch = -2
91
92 f1 = open(archivo_1, "r")
93 f2 = open(archivo_2, "r")
94
95 secuencia_1 = Obtener_Secuencia(f1)
96 secuencia_2 = Obtener_Secuencia(f2)
97
98 columna = len(secuencia_1)+1
99 fila = len(secuencia_2)+1
100
101 Matriz = np.zeros([fila, columna], dtype=int)
102
103 Alineamiento_Local(secuencia_1, secuencia_2, penalidad)

```

2.2 Prueba 1

2.2.1 Input

De acuerdo a lo solicitado, realizaremos las pruebas con los archivos .fasta, trabajados en la Práctica 03.

1. P21333.fasta - Filamina-A (Humano)
2. Q8BTM8.fasta - Filamina-A (Ratón)

2.2.2 Output

Seguidamente pasamos a ver el resultado generado :

```

cooper@cooper-Legion-Y545:~/Escritorio/5to/9no_SEMESTRE/BIOINFORMATICS/Practica_07$ python3 ejercicio2.py
P21333.fasta.txt Q8BTM8.fasta.txt -5
[[ 0  0  0 ...  0  0  0]
 [ 0  2  0 ...  0  0  0]
 [ 0  0  4 ...  0  0  0]
 ...
 [ 0  2  0 ... 5006 5001 4996]
 [ 0  0  0 ... 5005 5008 5003]
 [ 0  0  0 ... 5000 5003 5010]]

```

Figure 2: Matriz Final - gapOpen = gapEXTEND = -5, identicalMatch = 2 y mismatch = -2

Alineamiento para P21333.fasta - Filamina-A (Humano)

Secuencia Alineada 1 :

```

MSSSHSRAGQSAAGAAPGGVDTRDAEMPATEKDLAEDAPWKKIQNTFTRWCNEHLKCVSKRIANLQTDLSGLRLIALLEVL SQKKMHRKHNRPTFRQMQLN
VSVALEFLDRESIKLVSDSKAIVDGNLKLILGLIWTLLIHYSISMPMWDEEEDAEAKKQTPKQRLLGWIQNKLPQLPITNFSRDWQSGRALGALVDSAPGLCPD
WDSWDASKPVTNAREAMQADDWLGIQVITPEEIVDPNVDEHSVMTYLSQFPKAKLPGAPLRPKLNPKKARAYGPGIEPTGNMVKKRAEFTVETRAGQGEVLV
YVEDPAGHQEEAKVTANNDKNRTFSVMYVPEVTGTHKVTVL FAGQHIAKSPFEVVDKSGQDASKVTAQGPGLPSGNIANKTTYFEIFTAGAGTGEVEVVIQDPM
GQKGTVEPQLEARGDSTYRCSYQPTMEGVHTVHVTFAGVPIPRSPYTVTVGQACNPASACRAVGRGLQPKGVRVKETADFKVYTKGAGSGELKVTVKGPKEERVEVKQ
KDLGDGVYGFEEYPMVPGTYIVTITWGGQNIGRSPFEVKVGTTCGNQKVRANGPGLGEGVVGKSADFVVEAIGDDVGT LGFSVEGPSQAKIECDDKGGGSDVRVW
PQEAAGEYAVHVLCSNEDIRLSPFMADIRDAPODFHPRVKARGPGL EKTGVAVNKAPEFTVDAKHGGKAPLRVQVQDNEGCPVEALVKDNGNGTYSYCSYVPRKPVK
HTAMVSWGGSVSPNSPFRVNVGAGSHPNKVKVYGPVAKTGLKAHEPTYFTVDCAEAGQGDVSIIGIKCAPGVVGPAAEDIDFDIIRNDNDTFTVKYTPRGAGSYTI
MVL FADQATPTSPIRVKVEPSHDASKVKAEGPLSRTGVELGKPTHFTVNAKAAGKGLDVQFSGLTGDAVRDVDIIDHNDNTYTVKYTPVQGGPVGNNVTYGGD
PIPKSPFSVAVSPSLDLSKIKVSGLGEKVVDGKDQEFVKSAGGQGVASKIVGSPGAAPVCKVEPGLGADNSVVRFLPREEGPYEVEVTVGVPVPGSPFPLE
AVAPT KPSKVAFGPGQLQGSAGSPARFTIDTKAGTGGGLTVEGPCAEQLECLDNGDGTCSVSYPTEPGDYNINILFADTHIPGSPFKAHVVPFCFASKVKCS
GPGLERATAGEVQGFQVDCSSAGSAELTIEICSEAGLPAEVYIQDHGDGHTITITVPLCPGAYTVTIKYGGQPVNFPKQLQVEPAVDTSVGQCYGPGIEGQGVFR
EATTEFSVDARALTQTGGPHVKARVANPSGNLTETVYQDRGDGMVKEVTPYEGLHSVDVTVYDGSVPVSPFPQVPVTEGCDPSRVVRHGPQIGSGTTNKNKFTV
ETRGAGTGGLGLAVEGPSEAKMSCMDNKGSCSVEYIPYEAGTYSLVNTYGGHQVPGSPFKVPVHDVTDASKVKCSGPGLSGMVRANLPQSFQVDTSKAGVAPLQ
VKVQGPGLVPEPVDVNDAGDTQTVNYVPSREGPYISVLVYDEEVPRSPFKVKVLPDTHDASKVKASGPGLNITGVPASLPVEFTIDAKDAGEGLLAVQITDPEGK
PKKTHIQDNHDGTYTVAYVPDVTGRYTIILIKYGGDEIPFSPYRRAVPTGDASKCTVTVSIGGHGLGAGIGPTIQIGETVITVDTKAAGKGVCTCTVCTPDGSEV
DVDVVENEDGTDFIDYFAPQPGKYVICVRFGGEHVPNSPFQVTLAGDQPSVQPLRSQQLAPQYTYAQQGQQTWAPERPLVGVNGLDVTSLRPFDLVIPFTIKKG
EITGEVRMPSPGKVAOPTITDNKDGTVTVRYAPSEAGLHEMDIRYDNMHI PGSLQFYVDVYVNCGHVTAYGPGLTHGVVNKPATFTVNTKDAGEGGLSLAIEGPSKA
EISCTDNQDGTCSVSYLPVLPGDYSILVKYNEQHVPVGPFTARVTGDDSMRMSHLKVGSAADIPINISSETDL SLLTATVVPVSGREEPCLLRLRNGHVGISFVPK
ETGEHLVHVKKNGQHVASSPIPVVISQSEIGDASRVVSGQGLHEGHTFEPAEFIIDTRDAGYGGLSLIEGPSKVDINTEDLEDGTCRTVTCYPTPEGNYIINIKF
ADQHVPGSPFSVKVTGEGRVKESITRRRRAPSVANVGSCHDL SLKIPEISIQDMTAQVTSPSGKTHEAEIVEGENHTYCI RFVPAEMGHTTVSVKVKQHVPGSPF
QFTVGPLGEGGAHKVRAGGGLERAEAGVPAEFSINTREAGAGGLAIAVEGPSKAEISFEDRKDGSCGVAYVVQEPGDYEVSVKFNHHPDSPFVVPVVASPGDA
RRLTVSSLQESGLKVNQPAFVSLNGAKAIDAKVHSPSGAL EECYVTEIDQDKYAVRFPRENGVYLIDVKFNGTHIPGSPFKIRVGEPEGHGGDPGLVSAYGAG
LEGGVTGNPAEFVNTSNAGAGALSVTIDGPSKVKMDCQCEPEGYRVTYTPMAPGSYLSIKYGGPYHIGGSPFKAKVTGPRLVSNHSLHETSSVFVDSLTKATCA
PQHGA PGPGPADASKVAKGLGLSKAYVQKSSFTVDCSKAGNMMLLVGVHGPRTPEEILVKHVGSRLYSVSYLLKDKGEYTLVWKGDHPIPGSPYR

```

Figure 3: Alineamiento P21333 - gapOpen = gapEXTEND = -5, identicalMatch = 2 y mismatch = -2

Alineamiento para Q8BTM8.fasta - Filamina-A (Ratón)

Secuencia Alineada 2:

```
MSSSHSRCQSAVASPGGSDSRDAEMPATEKDLAEDAPWKKIQNTFTRWCNEHLKCVSKRIANLQTDLSGRLRLIALLEVLSQKKMHRKHNRPTFRQMQLN
VSVALEFLDRESIKLVSDSKAIVDGNLKLILGLIWLILHYSISMPMWDEEEDAAKQTPKQRLLGWIQNKLPQLPITNFSRDWQSGRALGALVDSAPGLCPD
WDSWDASKPVNNAREAMQADDWLGIQVITPEEIVDPNVDEHSMVTYLSQFPKAKLKPGAPLRPKLNPKKARAYGPGIEPTGNMVKKRAEFTVETRSAGQGEVLV
YVEDPAGHQEEAKVTANNDKNRTFSVMYVPEVTGTHKVTVLFAHQHIAKSPFEVYVDKSGQDASKVTAQGPGLPSGNIANKTTYFEIFTAGAGMGEVEVVIQDPT
GQKGTVEPQLEARGDSTYRCSYQPTMEGVHTVHVTFAGVPIPRSPYTVTVGQACNPAACRAIGRGLQPKGVRVKETADFKVYTKGAGSGELKVTVKGPKGEERVQK
KDLGDGVYGFEEYPTIPGTYTVITWGGQNIGRSPFEVKVGTGECNQKVRWANGPGLGGIVGKSADFVVEAIGDDVGTGLGFSVEGSPQAKIECDDKGDGSCDVRYW
PQEAEGYAVHVLNSEDIRLSPFMADIREAPQDFHPDRVKARGPGLKGTGAVNKPAAFTVDAKHAGKAPLRVQVQDNEGCSVEATVKDNGNGTYSCSYVPRKPVK
HTAMVSWGVSIPNSPFRVNVGAGSHPNKVKVYGPVAKTGLKAHEPTYFTVDCTEAGQGDVSIKICAPGVVGPTEADIDFDIIRNDNDFTVKYTPCGAGSYTI
MVLADQATPTSPIRVKVEPSHDAKSKVKAEGPGLNRTGVELGKPTHFTVNAKTAGKGLDVQFSGLAKGDAVRDVIDIDHNDNTYVKYIPVQGGPVGVNVTYGGD
HIPKSPFSGVSPSLDLSKIKVSGLDKVDVGKQDEFTVKSAGGQGVASKIVSPSGAAMPCKVEPGLGADNSVVRVFPREEGPYEEVTVYDGVVPVGPSPFPLE
AVAPTSPKSKAFKAGPGLQGGNAGSPARFTIDTKAGTGGGLGLTVEGPCEAQLCLDNGDGTCSVSYVPTPEPGDYNINILFADTHIPGSPFKAHVAPCFDASKVKCS
GPGLERATAGEVGQFQVDCSSAGSAELTIEICSEAGLPAEYVIQDHGDGTHITYIPLCPGAYTVTIKYGGQVPVNFPSKLQVEPAVDTSGVCYGPGEQGVFR
EATTEFSDARALTQTGGPHVKARVANPSGNLTDYVQDCGDGTGYKVEYTPYEEGVHSDVTDYDGSVPVSPFQVPTVEGCDPSRVRVHGPQISGTTNKNKFTV
ETRGAGTGGLGLAVEGPSEAKMSCMDNKGSCSVEYIPYAGTYSLNVTYGGHQPVGSPFKVPVHDVTDASKVKCSGPGLSGPMVRANLPQSFQVDTSKAGVAPLQ
VKVQGPGLVEPVDVVDNADGTQTVNYVPSREGSYSISVLYGEEVPRSPFKVKVLPHTDASKVKASGPGNLTTGVPASLPVEFTIDAKDAGEGLLAVQITDPEGK
PKKTHIQDNHGTITYVAYVPDVPGRYTIILIKYGGDEIPFSPYRVRAVPTGDASKCTVTSIGGHGLGAGIGPTIQIGEEVTITVDTKAAGKGVCTCTCTPDGSEV
DQDVVVENEDGTDFIDYFAPQPGKYVICVRFGGEHVPNSPFQVTALAGDQPTVQTPLRSQLAPQYNYPPQGSQQTWIPERPMVGVNGLDVTSLRPFDLVIFTIKKG
EITGEVRMPSGKVAQPSITDNKDGTVTVRYSPEAGLHEMDIRYDNMHIPGSPQLQFYVDYVNCGHITAYGPGTLHGTVNKPATFTVNTKDAGEGLLSLAIEGSKA
EISCTDNQDGTCSVSYLPVLPDGYSLVKYNDQHIPGSPFTARVTGDDSMRMSHLKVGSAADIPINISSETDLSTATVVPVPSGREEPCLLRLRNGHVGISFVVK
ETGEHLVHVKKNGQHVAASSPIPVVISQSEIGDASRVVSGQLHEGHTFPAEFIDTRDAGYGGLSLSEGPSKVDINTEDLEDGTCRVTYCTPEPGNYIINIKF
ADQHVPGSPFSVKVTGEGRVKESITRRRRAPSVANIGSHCDLSLKIPEISIQDMTAQVTSPSGKTHEAEIVEGENHTYICIRFVPAEMGMHTVSVKYKGQHVPGSPF
QFTVGLPGEAGHVRAGGPGGLERAEGVPAEFGIWTREAGAGGLAIAVEGSKAEISFEDRKDGSCGVAYVVQEPGDYEVSVKFNEEHIPDSPFVVPVSPSGDA
RRLTVSSQLQESGLKVNQPAFAVSLNAGAKAIDAKVHSPSGALEECYVTEIDQKYAVRIPRENGIYLDVKNFNGTHIPGSPFKIRVGEHPGHGDPGLVSAYGAG
LEGGVTGSPAEIFVNTSNAGAGALSVTIDGPSKVKMDCQECPEGYRVYTPMAPGYSYLISIKYGGPYHIGGSPFKAKVTGPRLVSNHSLHETSSVFVDSLTKVATV
PQHATSGPGPADVSKVAKGLGLSKAYVQKSNFTVDCSKAGNNMLLVGVHGPRTPEEILVHKMGSRLLYSVSYLLKDKGEYTLVVKWGEHIGPSPYR
```

Figure 4: Alineamiento Q8BTM8 - gapOpen = gapEXTEND = -5, identicalMatch = 2 y mismatch = -2

- Se debe tomar en cuenta que la matriz mostrada en la Figura2 está comprimida debido a la volumen de datos en los archivos fasta.
- En consecuencia podemos visualizar el resultado del alineamiento de ambas secuencias de Filamina A, tanto para el Humano como para el ratón en la Figura11 y la Figura12 respectivamente.

2.3 Prueba 2

- Ahora probaremos con un cambio de valores

```
C:\Users\51931\Downloads>python "ejercicio2 (1).py" P21333.fasta Q8BTM8.fasta -3

[[ 0 0 0 ... 0 0 0]
 [ 0 3 0 ... 0 0 0]
 [ 0 0 6 ... 0 0 0]
 ...
 [ 0 3 0 ... 7654 7651 7648]
 [ 0 0 2 ... 7655 7657 7654]
 [ 0 0 0 ... 7652 7654 7660]]
```

Figure 5: Matriz con gapOpen = gapEXTEND = -3

- Ahora las secuencias alineadas

```
Secuencia Alineada 1 :
MSSSHSRAGQSAAGAAPGGVDTRDAEMPATEKD LAEDAPWKKIQNTFTWNCNEHLKCVSKRIANLQTDLS DGLRLIALLEVL SQQKMKHRKHNRPTFRQMQL ENVSVALEFLDRESIK
LVSIDSKAIVDGNLKLILGLIWLILHYSISMPPMDEEEDAEAKKQTPKQRL LGWIQNKLPQLPTNF SRDWSGRALGALVDS CAPGLCPDWDSWDASKPV TNAREAMQQADDWLGIPO
VITPEEIVDPNVDEHSVMTYLSQFPKAKLKPGAPLRPKLNPKKARAYGPGIEPTGNMVKKRAEFTVETR SAGQGEVLVYVEDPAGHQEEAKVTANNDKNRTFSVWYYPEVTGTHKVTVL F
AGQHIKASPFEEVYVDKSGQDASKVTAQGPGL EPSGNIANKTTYFEIFTAGAGTGEVEVVIQDPMGQKGTVEPQLEARGDSTYRCSYQPTMEGVHTVHVTFAGVPIPRSPYTVTVGQACNP
SACRAVGRGLQPKGVRVKETADFKVYTKGAGSGELKVTVKGPKEERVVKQKDLGDGVYGEYYPMPGTIYITITWGGQNIGRSPFEVKVGTCEGNQKVRAMGPGLEGGVVGKSADFVVE
AIGDDVGT LGSVEGPSQAKIECDDKGDGSCDVRYWPQEAAGEYAVHVL CNSEDIRLSPFMADIRAPQDFHPRVKARGPGL EKTGVAVNKP AEFTVD AKHGGKAPLRVQVDNEGCPVE
ALVKDNGNGTYSYVPRKPVKHTAMVSWGGVSIPNSPFRVNVGAGSHPNKVKYVGPVAKTGLKAHEPTYFTVDCAEAGQGDVSI GIKCAPGVVGP AEADIDFDIIRNNDTFTVKYTP
RGAGSYTIMVLFADQATPTSPIRVKVEPSHDASKVKAEGPGLSRTGVELGKPTHTVNAKAAGKGLDVQFSGLTKGDAVRDVIDIDHNDNTYTVKYTPVQGGPVGVNVTVGGDPIKSP
FSVAVSPSLDLSKIKVSGLGEKVDVGKQDEF TVKSKGAGGQGVASKIVSGPSGAAPCKVEPGLGADNSVVRFLPREEGPYEVEVTVYDGVVPVGPSPFPL EAVAPT KPSKVKA FGPGLQGG
SAGSPARFTIDTKGAGTGGLGLTVEGPEAQL ECLDNGDGTCSVSYPTEPGDYNINILFADTHIPGSPFKAHVVP CFDA SKVKCSGPGLERATAGEVGQFQVDCSSAGSAELTIEICSE
AGLPAEVYIQDHGDGHTITVIPLCPGAYTVTIKYGQGPVPNPSKLVQEPADVTSGVQCYGPGIEGQGVFREATT EFSVDARALQTGGPHVKARVANPSGNLTETVYQDRGDGMKYVE
YTPYEGLHSDVTVYDGSVPVSPFPQVPTEGCDPSRVRVHGPQISGTTNKNPKFTVETRGAGTGGLGLAVEGPEAKMSCMDNKGSCSVEYIPYEAGTYSLVN TVYGGHQVPGSPFKV
PVHDVTDASKVKCSGPGLSPGMVRANLPQSFOVDTSKAGVAPLQVKVQGPGLVEPVDVNDADGTQTVNYPVSRGPGYSISVLYGDEEVPRSPFKVKVLP THDASKVKASGPG LNTTGV
PASLPVEFTIDAKDAGEGLLAVQITDPEGPKPKTHIQDNHDGTYTVAYVPDVTGRYTIILKYGGDEIPFSPYRVRAVPTGDASKCTVTVS IGGHGLGAGIGPTIQIGEEVTITVDTKAAG
KGVKTVCTCTPDGSEVDVVDVNEDEGTDFIYFAPQPGKYVICRVFGEHVNSPFPQV TALAGDQPSVQPLRSQQLAPQYTAQGGQQTWAPERPLVGVNGLDVTSLRPFDLVIPFTIK
KGEITGEVRMPSGKVAQPTITDNKDGTVTVRYAPSEAGLHEMDIRYDNMHIHGSPQLQFYVDVYVNCGHVTA YGPGLTHGVNKPATFTVNTKDAGEGGLSLAIEGPKSAEISCTDNQDGT C
SVSYLPVLPGDYSILVKYNEQHVPSPGFARVTDGDSMRMHLKVGSAADIPINISETDL SLLTATVVPVPSGREEPCLKRLRNHGVGISFVPKTEGEHLVHVKNQGHVASSPVPVVIS
QSEIGDASRVRSVQGLHEGHTFEPAEFTIIDTRDAGYGGLSLSEIGPSKVDINTEDL EDGTCRTVYCTEPGNYIINIKFADQHVPGSPFSVKVTGEGVRKESITRRRRAPSVANVGS HC
DLSLKTEPISIQDMTAQVTSPSGKTHEAEIVGENHNTYCI RFPVPAEMGHTTVSKYKGQHVPGSPFQFVGPLGEGGAHKVRAGGPGLEAEAGVPAEFISINTREAGAGGLAIAVEGPKS
AEISFEDRKDSCGVAAYVQEPGDYEVSVKFNEEHIPTDPSFVVPVAPSPGDARRLTVSSLQESGLKVNQPASFAVSLNGAKGAIDAKVHSPSGAL EECYVTEIDQKYAVRFIPRENGVY
LIDVKFNGHTHPGSPFKIRVGEPHGGDPGLVSAYGAGLEGGVTGNPAEFVWNTSNAGAGALSVTIDGPKSVKMDQCECP EGYRVTVTPMAPGSYLSIKYGGPYHIGGSPFKAKVTGPR
LVSNHSLHETS SVFVDSLTKVAT-VQPHATSGPGADVSKVAKGLGLSKAYVGQKSNFTVDCSKAGNNMLLVGVHGPRTPCEEILVKHMGSR LYSVSYLLKDKGEYTLVVKWGD EHIPG
SPYRIMVP
```

Figure 6: Secuencia alineada 1

```
Secuencia Alineada 2:
MSSSHSRGQSAAVASPGGSDSRDAEMPATEKD LAEDAPWKKIQNTFTWNCNEHLKCVSKRIANLQTDLS DGLRLIALLEVL SQQKMKHRKHNRPTFRQMQL ENVSVALEFLDRESIK
LVSIDSKAIVDGNLKLILGLIWLILHYSISMPPMDEEEDAEAKKQTPKQRL LGWIQNKLPQLPTNF SRDWSGRALGALVDS CAPGLCPDWDSWDASKPV TNAREAMQQADDWLGIPO
VITPEEIVDPNVDEHSVMTYLSQFPKAKLKPGAPLRPKLNPKKARAYGPGIEPTGNMVKKRAEFTVETR SAGQGEVLVYVEDPAGHQEEAKVTANNDKNRTFSVWYYPEVTGTHKVTVL F
AGQHIKASPFEEVYVDKSGQDASKVTAQGPGL EPSGNIANKTTYFEIFTAGAGTGEVEVVIQDPMGQKGTVEPQLEARGDSTYRCSYQPTMEGVHTVHVTFAGVPIPRSPYTVTVGQACNP
AACRAIGRGLQPKGVRVKETADFKVYTKGAGSGELKVTVKGPKEERVVKQKDLGDGVYGEYYPMPGTIYITITWGGQNIGRSPFEVKVGTCEGNQKVRAMGPGLEGGVVGKSADFVVE
AIGDDVGT LGSVEGPSQAKIECDDKGDGSCDVRYWPQEAAGEYAVHVL CNSEDIRLSPFMADIRAPQDFHPRVKARGPGL EKTGVAVNKP AEFTVD AKHAGKAPLRVQVDNEGCSVE
ATVKDNGNGTYSYVPRKPVKHTAMVSWGGVSIPNSPFRVNVGAGSHPNKVKYVGPVAKTGLKAHEPTYFTVDCTEAGQGDVSI GIKCAPGVVGP AEADIDFDIIRNNDTFTVKYTP
CGAGSYTIMVLFADQATPTSPIRVKVEPSHDASKVKAEGPGLNRTGVELGKPTHTVNAKTAGKGLDVQFSGLAKGDAVRDVIDIDHNDNTYTVKYIPVQGGPVGVNVTVGGDHPKSP
FSVAVSPSLDLSKIKVSGLGDVVDVGKQDEF TVKSKGAGGQGVASKIVSGPSGAAPCKVEPGLGADNSVVRFLPREEGPYEVEVTVYDGVVPVGPSPFPL EAVAPT KPSKVKA FGPGLQGG
NAGSPARFTIDTKGAGTGGLGLTVEGPEAQL ECLDNGDGTCSVSYPTEPGDYNINILFADTHIPGSPFKAHVAPCFDA SKVKCSGPGLERATAGEVGQFQVDCSSAGSAELTIEICSE
AGLPAEVYIQDHGDGHTITVIPLCPGAYTVTIKYGQGPVPNPSKLVQEPADVTSGVQCYGPGIEGQGVFREATT EFSVDARALQTGGPHVKARVANPSGNLTETVYQDRGDGMKYVE
YTPYEGLHSDVTVYDGSVPVSPFPQVPTEGCDPSRVRVHGPQISGTTNKNPKFTVETRGAGTGGLGLAVEGPEAKMSCMDNKGSCSVEYIPYEAGTYSLVN TVYGGHQVPGSPFKV
PVHDVTDASKVKCSGPGLSPGMVRANLPQSFOVDTSKAGVAPLQVKVQGPGLVEPVDVNDADGTQTVNYPVSRGPGYSISVLYGDEEVPRSPFKVKVLP THDASKVKASGPG LNTTGV
PASLPVEFTIDAKDAGEGLLAVQITDPEGPKPKTHIQDNHDGTYTVAYVPDVTGRYTIILKYGGDEIPFSPYRVRAVPTGDASKCTVTVS IGGHGLGAGIGPTIQIGEEVTITVDTKAAG
KGVKTVCTCTPDGSEVDVVDVNEDEGTDFIYFAPQPGKYVICRVFGEHVNSPFPQV TALAGDQPSVQPLRSQQLAPQYTAQGGQQTWAPERPLVGVNGLDVTSLRPFDLVIPFTIK
KGEITGEVRMPSGKVAQPTITDNKDGTVTVRYAPSEAGLHEMDIRYDNMHIHGSPQLQFYVDVYVNCGHVTA YGPGLTHGVNKPATFTVNTKDAGEGGLSLAIEGPKSAEISCTDNQDGT C
SVSYLPVLPGDYSILVKYNEQHVPSPGFARVTDGDSMRMHLKVGSAADIPINISETDL SLLTATVVPVPSGREEPCLKRLRNHGVGISFVPKTEGEHLVHVKNQGHVASSPVPVVIS
QSEIGDASRVRSVQGLHEGHTFEPAEFTIIDTRDAGYGGLSLSEIGPSKVDINTEDL EDGTCRTVYCTEPGNYIINIKFADQHVPGSPFSVKVTGEGVRKESITRRRRAPSVANVGS HC
DLSLKTEPISIQDMTAQVTSPSGKTHEAEIVGENHNTYCI RFPVPAEMGHTTVSKYKGQHVPGSPFQFVGPLGEGGAHKVRAGGPGLEAEAGVPAEFISINTREAGAGGLAIAVEGPKS
AEISFEDRKDSCGVAAYVQEPGDYEVSVKFNEEHIPTDPSFVVPVAPSPGDARRLTVSSLQESGLKVNQPASFAVSLNGAKGAIDAKVHSPSGAL EECYVTEIDQKYAVRFIPRENGVY
LIDVKFNGHTHPGSPFKIRVGEPHGGDPGLVSAYGAGLEGGVTGNPAEFVWNTSNAGAGALSVTIDGPKSVKMDQCECP EGYRVTVTPMAPGSYLSIKYGGPYHIGGSPFKAKVTGPR
LVSNHSLHETS SVFVDSLTKVAT-VQPHATSGPGADVSKVAKGLGLSKAYVGQKSNFTVDCSKAGNNMLLVGVHGPRTPCEEILVKHMGSR LYSVSYLLKDKGEYTLVVKWGD EHIPG
SPYRIMVP
```

Figure 7: Secuencia alineada 2

- gapOpen = gapEXTEND = -3, identicalMatch = 3 y mismatch = -1
- Este fue el cambio de valores que se realizo

```

THIPGSPFKIRVGEPGHGGDPGLVSAYGAGLEGGVTGNPAEFVWNTSNAGAGALSVTIDGPSKVKMDCQECPEGYRVTYTPMAPGS
YLISIKYGGPYHIGGSPFKAKVTGPRLVSNHSLHETSSVFVDSLTKATCAPQHAGPGPADASKVVAKGLGLSKAYVGQKSSFTV
DCSKAGNNMLLVGVHGPRTPCCEEILVKHVGSRLYSVSYLLKDKGEYTLVVKNGDEHIPGSPYR

```

Figure 8: Secuencia alineada 1 (Original)

```

YLISIKYGGPYHIGGSPFKAKVTGPRLVSNHSLHETSSVFVDSLTK-AT-
VPQHATSGPGPADVSKVVAKGLGLSKAYVGQKSNFTVDCSKAGNNMLLVGVHGPRTPCCEEILVKHVGSRLYSVSYLLKDK-
GEYTLVVKNGDEHIPGSPYRIMVP

```

Figure 9: Secuencia alineada 1 (Valores cambiados)

- **Comentarios:**
 - Hay pequeñas diferencias, y por lo que hemos podido observar estas diferencias varian tambien dependiendo de los valores otorgados o el tamaño de la secuencia alineada

3 Actividad 3

Evalúe la pregunta anterior con otros valores de *gapOpen*, *apEXTEND*, *identicalMatch* y *mismatch*. Verifique si obtiene el mismo alineamiento y comente sus resultados.

Resolución

3.1 Código

En el código cambiaremos los valores *gapOPEN*, *gapEXTEND*, *identicalMatch*, *mismatch*.

3.2 Resultados

3.2.1 Input

De acuerdo a lo solicitado, realizaremos las pruebas con los archivos *.fasta*, trabajados en la Práctica 06.

1. P21333.fasta - Filamina-A (Humano)
2. Q8BTM8.fasta - Filamina-A (Ratón)

3.2.2 Output

Seguidamente pasamos a ver el resultado generado :

```
D:\INGRID\unsa\2021-1\bio\practica7\prueba>python ejercicio2.py Q8BTM8.fasta.txt P21333.fasta.txt -3
[[ 0 0 0 ... 0 0 0]
 [ 0 1 0 ... 1 0 0]
 [ 0 0 2 ... 0 0 0]
 ...
 [ 0 0 0 ... 2503 2502 2499]
 [ 0 0 0 ... 2500 2504 2501]
 [ 0 0 0 ... 2497 2501 2505]]
```

Figure 10: Matriz Final - *gapOpen* = *gapEXTEND* = -3, *identicalMatch* = 1 y *mismatch* = -1

Alineaminto para P21333.fasta - Filamina-A (Humano)

[illegible]

Figure 11: Alineamineto P21333 - gapOpen = gapEXTEND = -3, identicalMatch = 1 y mismatch = -1

Alineamineto para Q8BTM8.fasta - Filamina-A (Ratón)

[illegible]

Figure 12: Alineaminto Q8BTM8 - gapOpen = gapEXTEND = -3, identicalMatch = 1 y mismatch = -1

- Por ultimo probaremos con $\text{gap} = -4$ e $\text{identicalMatch} = 2$ y $\text{mismatch} = -2$

```
cooper@cooper-Legion-Y545:~/Escritorio/5to/9no_SEMESTRE/BIOINF
$ python3 ejercicio2.py P21333.fasta.txt Q8BTM8.fasta.txt -4
2648
2648

[[ 0 0 0 ... 0 0 0]
 [ 0 1 0 ... 0 0 0]
 [ 0 0 2 ... 0 0 0]
 ...
 [ 0 1 0 ... 2503 2499 2495]
 [ 0 0 0 ... 2501 2504 2500]
 [ 0 0 0 ... 2497 2500 2505]]
```

Figure 13: Matriz con gapOpen = gapEXTEND = -4

Secuencia Alineada 1 :

```

MSSSHSRAGQSAAGAAPGGGVDTRDAEMPATEKDLAEDAPWKKIQNTFTRCNEHLKCVSKRIANLQTDLSGLRLIALLEVL SQQMKHRKHNRPTFRQMQLENVSV
ALEFLDRESIKLVSDSKAIVDGNLKLILGLIWLILHYSISMPMWDEEEDAEAKQTPKQRLLGWIQNKLPQLPITNFSRDWQSGRALGALVDSCAPGLCPDWDSWDA
SKPVTNAREAMQADDWLGIQVITPEEIVDPNVDEHSMVTYLSQFPKAKLKPAGLRLPKLNPKKARAYGPGIEPTGNMVKKRAEFTVETRAGQGEVLVYVEDPAGHQ
EEAKVTANNDKNRTFSVWVPEVTGTHKVTVLFAGQHIKSPFEVYVDSKQSGDASKVTAQGGPLESGNIANKTTYFEIFTAGAGTGEVEVVIQDPMQKGTVEPQLEA
RGDSTYRCVSYQPTMEGVHTVHTFAGVPIPRSPYTVTVGQACNPASACRAVGRGLQPKGVVRKETADFKVYTKGAGSGELKVTVKGPKEERVVKQDLGDGVYGFYVPM
VPGTYIVTITWGGQNIGRSPFEVKVGTCEGNQKVRWAGPGLGEGVVGKSADFFVEAIGDDVGTGFGSVEGSPQAKIECDDKGDGSCDVRVWPQEAAGEYAVHVLNSED I
RLSPFMADIRDPQDFHDPDRVKARGPGLKTKGAVNKPAAFTVDAKHGGKAPLRVQVQDNEGCPVEALVKDNGNGTYSYVPRKPVKHTAMVSWGGVSPNPSFRVNV
GAGSHPNKVKVYGPVAKTGLKAHEPTYFTVDCAEAGQGDVSIKICAPGVVGAPEADIDFDIIRNDNDTFTVKYTPRGAGSYTIMVLFADQATPTSPIRVKVEPSHDA
SKVKAEGPGLSRTGVELGKPTHTVNAKAAGKGLDVQFSGLTGKDAVRDVIDIDHNDNTYTVKYTPVQGGPVGVNVTYGGDPIKSPFSVAVSPSLDL SKIKVSGLGE
KVDVGKDQEFVTKSGAGGQGVASKIVGSPGAAPCKVEPGLGADNSVVRFLPREEGPYEVEVTYDGVVPVGPSPFLEAVAPT KPSVKVAFGPGGLQGGGAGSPARFTI
DTKGAGTGGLGLTVEGPCEAQLECLDNGDGTCSVSYVPTPEGDYINILFADTHIPGSPFAHVVPFCFADSKVKCSGGLERATAGEVGQFQVDCSSAGSAELTIEICS
EAGLPAEVYIQDHGDGTHITITYIPLCPGAYTVTIKYGGQVPVNFPSKLQVEPAVDTSVGVCYGPGEIQQGVFREATTESVDARALTQTGGPHVKARVANP SGNLTETY
VQDRGDMGYKVEYTPYEGLHSDVTYDGSVPVSPFPQVPTTEGCDPSRVVRVHGPGIQSGTTNKNPKFTVETRAGTGGLGLAVEGSPSEAKMSCMDNKGSCSVEYIPY
EAGTYSLVNTYGGHQPVPSPFKVPVHDVTDASKVKCSGGLSPGMVRANLPQSFQVDTSKAGVAPLQVKVQGPGLVEPVDVVDNADGTQTVNVVPSREGPYSISLVYG
DEEVPRSPFKVKVLPHTDASKVKASGGLNTTGVPASLPVEFTIDAKDAGEGLLAVQITDPEGPKPKTHIQDNHGDGTYTVAYVPDVTGRYITILIKYGGDEIPFSPYRVR
AVPTGDASKCTVTVSIGGHGLGAGIGPTIQIGEEVTITVDTKAAGKGVKTCTVCTPDGSEVDVVDVNEDEDGTDFIFYTAPQPGKYVICVRFGEHVPNSPFQVTLAAGD
QPSVQPLRLSQQLAPQYTYAQGGQQTWAPERPLVGVNGLDVTSLRPFDLVIFPTIKKGEITGEVRMPSGKVAQPTITDNKDGTVTVRVYAPSEAGLHEMDIRYDNMHPG
SPLQFYVVDYVNCGHVITAYGPGLTGCVNKPATFTVNTKDAGEGLSLAIEGSKAEISCTDNQDGTCSVSYLPVLPDGYISLVKYNEQHVPGSPFTARVTGDDSMRMSH
LVKGSAADIPINISSETDLSLLTATVVPSPGREEPCLLRLRNGHVGISFVPKETGEHLVHVKNQGHVASSPPIPVVISQSEIGDASRVRSVGGGLHEGHTFEPAEFTID
TRDAGYGLSLSTIEGPKVDINTEDLGTCTVTCPTPEGNVYIINIKFADQHVPGSPFSVKVTGEGRVKESITRRRRAPSVANVIGSHCDLSLKIPEISIQDMTAQVTS
PSGKTHEAIEVEGENHTYICIRFVPAEMGHTVSVKYKQHVPGSPFQFTVGPLGEGGAHKVRAGGGLERAEAGVPAEFSIWTREAGAGGLAIAVEGPSKAEISFEDRK
DGS CGVAYVQEPGDYEVSVKNEEHPDPSFVVPVSPSGDARRLTVSSLQESGLKVNQSPASFVSLNGAKGADAKVHSPSGAL EECYVTEIDQDKYAVRFIPRENG
IYLIDVKNFTGTHIPGSPFKIRVGEPCGHGGDPLVSAYGAGLEGGVTGSPAEIFVNTSNAGAGALSVTIDGPSKVKMDQCEQCEGYRVYTPMPAGSYLISIKYGGPYHI
GGSPFKAKVTGPRLVSNHSLHETSSVFVDSLTKATCAPQHAGPGPADASKVAKGLGLSKAYVVGQKSNFTVDCSKAGNNMLLVGVHGPRTPEEILVKHVGSRLYSV
SYLLKDKGEYTLVVKWGDHPIGSPYR

```

Figure 14: P12333 con gapOpen = gapEXTEND = -4, identicalMatch = 2 y mismatch = -2

Secuencia Alineada 2:

```

MSSSHSRAGQSAAGAAPGGGVDTRDAEMPATEKDLAEDAPWKKIQNTFTRCNEHLKCVSKRIANLQTDLSGLRLIALLEVL SQQMKHRKHNRPTFRQMQLENVSV
ALEFLDRESIKLVSDSKAIVDGNLKLILGLIWLILHYSISMPMWDEEEDAEAKQTPKQRLLGWIQNKLPQLPITNFSRDWQSGRALGALVDSCAPGLCPDWDSWDA
SKPVTNAREAMQADDWLGIQVITPEEIVDPNVDEHSMVTYLSQFPKAKLKPAGLRLPKLNPKKARAYGPGIEPTGNMVKKRAEFTVETRAGQGEVLVYVEDPAGHQ
EEAKVTANNDKNRTFSVWVPEVTGTHKVTVLFAGQHIKSPFEVYVDSKQSGDASKVTAQGGPLESGNIANKTTYFEIFTAGAGTGEVEVVIQDPMQKGTVEPQLEA
RGDSTYRCVSYQPTMEGVHTVHTFAGVPIPRSPYTVTVGQACNPASACRAVGRGLQPKGVVRKETADFKVYTKGAGSGELKVTVKGPKEERVVKQDLGDGVYGFYVPT
IPGTYIVTITWGGQNIGRSPFEVKVGTCEGNQKVRWAGPGLGEGVVGKSADFFVEAIGDDVGTGFGSVEGSPQAKIECDDKGDGSCDVRVWPQEAAGEYAVHVLNSED I
RLSPFMADIRDPQDFHDPDRVKARGPGLKTKGAVNKPAAFTVDAKHGGKAPLRVQVQDNEGCPVEALVKDNGNGTYSYVPRKPVKHTAMVSWGGVSPNPSFRVNV
GAGSHPNKVKVYGPVAKTGLKAHEPTYFTVDCAEAGQGDVSIKICAPGVVGAPEADIDFDIIRNDNDTFTVKYTPRGAGSYTIMVLFADQATPTSPIRVKVEPSHDA
SKVKAEGPGLSRTGVELGKPTHTVNAKTAGKGLDVQFSGLAGDAVRDVIDIDHNDNTYTVKYIPVQGGPVGVNVTYGGDHIKSPFSVAVSPSLDL SKIKVSGLGD
KVDVGKDQEFVTKSGAGGQGVASKIVSPGAAPCKVEPGLGADNSVVRFLPREEGPYEVEVTYDGVVPVGPSPFLEAVAPT KPSVKVAFGPGGLQGGNAGSPARFTI
DTKGAGTGGLGLTVEGPCEAQLECLDNGDGTCSVSYVPTPEGDYINILFADTHIPGSPFAHVVPFCFADSKVKCSGGLERATAGEVGQFQVDCSSAGSAELTIEICS
EAGLPAEVYIQDHGDGTHITITYIPLCPGAYTVTIKYGGQVPVNFPSKLQVEPAVDTSVGVCYGPGEIQQGVFREATTESVDARALTQTGGPHVKARVANP SGNLTDTY
VQDCGDGTYKVEYTPYEEGVHSDVTYDGSVPVSPFPQVPTTEGCDPSRVVRVHGPGIQSGTTNKNPKFTVETRAGTGGLGLAVEGSPSEAKMSCMDNKGSCSVEYIPY
EAGTYSLVNTYGGHQPVPSPFKVPVHDVTDASKVKCSGGLSPGMVRANLPQSFQVDTSKAGVAPLQVKVQGPGLVEPVDVVDNADGTQTVNVVPSREGPYSISLVYG
EEVPRSPFKVKVLPHTDASKVKASGGLNTTGVPASLPVEFTIDAKDAGEGLLAVQITDPEGPKPKTHIQDNHGDGTYTVAYVPDVPGRYITILIKYGGDEIPFSPYRVR
AVPTGDASKCTVTVSIGGHGLGAGIGPTIQIGEEVTITVDTKAAGKGVKTCTVCTPDGSEVDVVDVNEDEDGTDFIFYTAPQPGKYVICVRFGEHVPNSPFQVTLAAGD
QPTVQPLRLSQQLAPQYTYAQGGQQTWAPERPLVGVNGLDVTSLRPFDLVIFPTIKKGEITGEVRMPSGKVAQPTITDNKDGTVTVRVYAPSEAGLHEMDIRYDNMHPG
SPLQFYVVDYVNCGHVITAYGPGLTGCVNKPATFTVNTKDAGEGLSLAIEGSKAEISCTDNQDGTCSVSYLPVLPDGYISLVKYNDQHVPGSPFTARVTGDDSMRMSH
LVKGSAADIPINISSETDLSLLTATVVPSPGREEPCLLRLRNGHVGISFVPKETGEHLVHVKNQGHVASSPPIPVVISQSEIGDASRVRSVGGGLHEGHTFEPAEFTID
TRDAGYGLSLSTIEGPKVDINTEDLGTCTVTCPTPEGNVYIINIKFADQHVPGSPFSVKVTGEGRVKESITRRRRAPSVANVIGSHCDLSLKIPEISIQDMTAQVTS
PSGKTHEAIEVEGENHTYICIRFVPAEMGHTVSVKYKQHVPGSPFQFTVGPLGEGGAHKVRAGGGLERAEAGVPAEFGIWTREAGAGGLAIAVEGPSKAEISFEDRK
DGS CGVAYVQEPGDYEVSVKNEEHPDPSFVVPVSPSGDARRLTVSSLQESGLKVNQSPASFVSLNGAKGADAKVHSPSGAL EECYVTEIDQDKYAVRFIPRENG
IYLIDVKNFTGTHIPGSPFKIRVGEPCGHGGDPLVSAYGAGLEGGVTGSPAEIFVNTSNAGAGALSVTIDGPSKVKMDQCEQCEGYRVYTPMPAGSYLISIKYGGPYHI
GGSPFKAKVTGPRLVSNHSLHETSSVFVDSLTKVATVPQATSGPGPADVSKVAKGLGLSKAYVVGQKSNFTVDCSKAGNNMLLVGVHGPRTPEEILVKHMGSRLYSV
SYLLKDKGEYTLVVKWGDHPIGSPYR

```

Figure 15: Q8TBM8 alinedo con gapOpen = gapEXTEND = -4, identicalMatch = 2 y mismatch = -2

4 Conclusiones

1. En el ejercicio 1 como vemos probamos el algoritmo con esas dos cadenas y hallamos alineación local con su penalidad el cual se observa su mejores alineamientos y su mejor scored el cual observamos que nuestro mejor scored siempre será mayor que cero.
2. El método experimental de alineamiento de DNA es lento, pero permite recuperar material genético a través de su alineamiento local. Debido a su lentitud se recomienda usar frameworks como Hadoop para el procesamiento distribuido de grandes cantidades de datos. Una alternativa es crear clusters en AWS o DataProc de Google, y usar el algoritmo paralelizado con Hadoop.
3. Se probó el algoritmo con secuencias reales de proteínas y nos dio como resultado nuestra matriz y las dos secuencias de alineamiento local, lo cual serviría para futuros trabajos de procesamiento genómico. Pero aun así se tiene que realizar un preprocesamiento al output del algoritmo para poder insertar en la RNN.

BASE DE COMPARACIÓN	ALINEACIÓN DE SECUENCIA GLOBAL	ALINEACIÓN DE SECUENCIA LOCAL
Técnica general	Algoritmo Needleman-Wunsch.	Algoritmo Smith-Waterman.
Descripción	Se intenta alinear toda la secuencia (alineación de extremo a extremo).	Encuentra regiones locales con el mayor nivel de similitud entre las dos secuencias.
Función	Contiene todas las letras de las secuencias de consulta y de destino.	Alinea una subcadena de la secuencia de consulta con una subcadena de la secuencia objetivo.
Dos secuencias	Si dos secuencias tienen aproximadamente la misma longitud y son bastante similares, son adecuadas para la alineación global.	Cualquiera de las dos secuencias se pueden alinear localmente ya que la alineación local encuentra tramos de secuencias con un alto nivel de coincidencias sin considerar la alineación del resto de las regiones de secuencia.
Idoneidad	Adecuado para alinear dos secuencias estrechamente relacionadas.	Adecuado para alinear secuencias más divergentes o secuencias relacionadas lejanamente.
Usar	Los alineamientos globales generalmente se realizan para comparar genes homólogos, como comparar dos genes con la misma función (en humanos frente a ratones) o comparar dos proteínas con función similar.	Se utiliza para descubrir patrones conservados en secuencias de ADN o dominios o motivos conservados en dos proteínas.

5 Repositorio

La resolución de la práctica la encuentra en el siguiente enlace [Practica_07](#)