Dinamik programlama ile biyolojik dizilerde hizalamalar

```
f = open('katG.fasta', 'r')

l = f.readline()
while(l):
    if '>' in l:
        print('Yupppiii.. yeni sekans:')
        print(l)
    else:
        print(l)
    l = f.readline()
```

```
Yupppiii.. yeni sekans:
>KP746928.1 Mycobacterium tuberculosis H37Rv isolate INH23.2-13MGT-98 KatG (katG) gene, complete cds

GTGCCCGAGCAACACCCACCCATTACAGAAACCACCACCGGAGCCGCTAGCAACGGCTGTCCCGTCGTGG

GTCATATGAAATACCCCGTCGAGGGCGGCGGAAACCAGGACTGGTGGCCCAACCGGCTCAATCTGAAGGT

ACTGCACCAAAACCCGGCCGTCGCTGACCCGATGGGTGCGGCGTTCGACTATGCCGCGGAGGTCGCGACC

ATCGACGTTGACGCCCTGACGCGGGACATCGAGGAAGTGATGACCACCTCGCAGCCGTGGTGGCCCGCCG
```

```python
f = open('katG.fasta', 'r')
sequences = {}
l = f.readline()
curSeqName = ''
while(l):
    if '>' in l:  # sequence name
        curSeqName = l.strip()
        sequences[curSeqName] = ''
    else:         # body of sequence
        sequences[curSeqName] = sequences[curSeqName] + l.strip()
    l = f.readline()

print(sequences)
```

```
{'>KP746928.1 Mycobacterium tuberculosis H37Rv isolate INH23.2-13MGT-98 KatG (katG) gene, complete cds': 'GTGCCCGA
GCAACACCCACCCATTACAGAAACCACCACCGGAGCCGCTAGCAACGGCTGTCCCGTCGTGGGTCATATGAAATACCCCGTCGAGGGCGGCGGAAACCAGGACTGGTGGCCCAA
CCGGCTCAATCTGAAGGTACTGCACCAAAACCCGGCCGTCGCTGACCCGATGGGTGCGGCGTTCGACTATGCCGCGGAGGTCGCGACCATCGACGTTGACGCCCTGACGCGGGA
CATCGAGGAAGTGATGACCACCTCGCAGCCGTGGTGGCCCGCCGACTACGGCCACTACGGGCCGCTGTTTATCCGGATGGCGTGGCACGCTGCCGGCACCTACCGCATCCACGA
GAAGTGGACCGGCAGCCGCGTGGACCTGGTCTTCGGGTCCAACTCGGAGTTGCGGGCGCTTGTCGAGGTCTATGGCGCCGATGACGCGCAGCCGAAGTTCGTGCAGGACTTCGT
CGCTGCCTGGGACAAGGTGATGAACCTCGACAGGTTCGACGTGCGCTGA', '>KP746927.1 Mycobacterium tuberculosis H37Rv isolate INH23.2-
13MGT-97 KatG (katG) gene, complete cds': 'GTGCCCGAGCAACACCCACCCATTACAGAAACCACCACCGGAGCCGCTAGCAACGGCTGTCCCGTCGTGGG
TCATATGAAATACCCCGTCGAGGGCGGCGGAAACCAGGACTGGTGGCCCAACCGGCTCAATCTGAAGGTACTGCACCAAAACCCGGCCGTCGCTGACCCGATGGGTGCGGCGTT
CTGACCAACGACTTCTTCGTGAACCTGCTCGACATGGGTATCACCTGGGAGCCCTCGCCAGCAGATGACGGGACCTACCAGGGCAAGGATGGCAGTGGCAAGGTGAAGTGGACC
GGCAGCCGCGTGGACCTGGTCTTCGGGTCCAACTCGGAGTTGCGGGCGCTTGTCGAGGTCTATGGCGCCGATGACGCGCAGCCGAAGTTCGTGCAGGACTTCGTCGCTGCCTGG
GACAAGGTGATGAACCTCGACAGGTTCGACGTGCGCTGA'}
```

```
f = open('a.fasta', 'r')
sequences = {}
l = f.readline()
curSeqName = ''
while(l):
    if '>' in l:  # sequence name
        curSeqName = l.strip()
        sequences[curSeqName] = ''
    else:         # body of sequence
        sequences[curSeqName] = sequences[curSeqName] + l.strip()
    l = f.readline()

print(sequences)
```

```
{'>s1': 'ATGGATGATGGGATGTCGTTAGT', '>s2': 'ATGGATGCTCGGGATGTAGTTAGT'}
```

```
if sequences['>s1'] == sequences['>s2']:
    print('FIT!!!')
else:
    print(':/')
```

```
:/
```

```python
def seqCompare(s1, s2):
    score = 0
    print(s1)
    for i in range(0, len(s1)):
        if s1[i] == s2[i]:
            score = score + 1
            print('|', end='')
        else:
            print('.', end='')
    print('\n')
    print(s2)
    return score / len(s1)
```

```python
s = seqCompare(sequences['>s1'], sequences['>s2'])
print(s)
```

```
ATGGATGATGGGATGTCGTTAGT
|||||||.|.||.......|...

ATGGATGCTCGGGATGTAGTTAGT
0.4782608695652174
```

Kayan pencere yöntemi:

```
f = open('katG.fasta', 'r')
sequences = {}
l = f.readline()
curSeqName = ''
while(l):
    if '>' in l:  # sequence name
        curSeqName = l.strip()[1:]
        sequences[curSeqName] = ''
    else:         # body of sequence
        sequences[curSeqName] = sequences[curSeqName] + l.strip()
    l = f.readline()

print(sequences['s1'])
```

```
GTGCCCGAGCAACACCCACCCATTACAGAAACCACCACCGGAGCCGCTAGCAACGGCTGTCCCGTCGTGGGTCATATGAAATACCCCGTCGAGGGCGGCGGAAACCAGGACTGGT
GGCCCAACCGGCTCAATCTGAAGGTACTGCACCAAAACCCGGCCGTCGCTGACCCGATGGGTGCGGCGTTCGACTATGCCGCGGAGGTCGCGACCATCGACGTTGACGCCCTGAC
GCGGGACATCGAGGAAGTGATGACCACCTCGCAGCCGTGGTGGCCCGCCGACTACGGCCACTACGGGCCGCTGTTTATCCGGATGGCGTGGCACGCTGCCGGCACCTACCGCATC
CACGACGGCCGCGGCGGCGCCGGGGGCGGCATGCAGCGGTTCGCGCCGCTTAACAGCTGGCCCGACAACGCCAGC
```

Kayan pencere yöntemi:

```
def seqCompare(s1, s2):
    score = 0.0
    for i in range(0, len(s1)):
        if s1[i] == s2[i]:
            score = score + 1
    return score / len(s1)
```

```
def windowAlign(s1:str, s2:str, ws=10):
    sm = []
    for i1 in range(0, len(s1)-ws+1):
        w1 = s1[i1:i1+ws]
        #print(w1)
        rowScore = []
        for i2 in range(0, len(s2)-ws+1):
            w2 = s2[i2:i2+ws]
            score = seqCompare(w1, w2)
            rowScore.append(score)
        sm.append(rowScore)
    print(sm)
```

```
windowAlign(sequences['s1'], sequences['s2'])
```

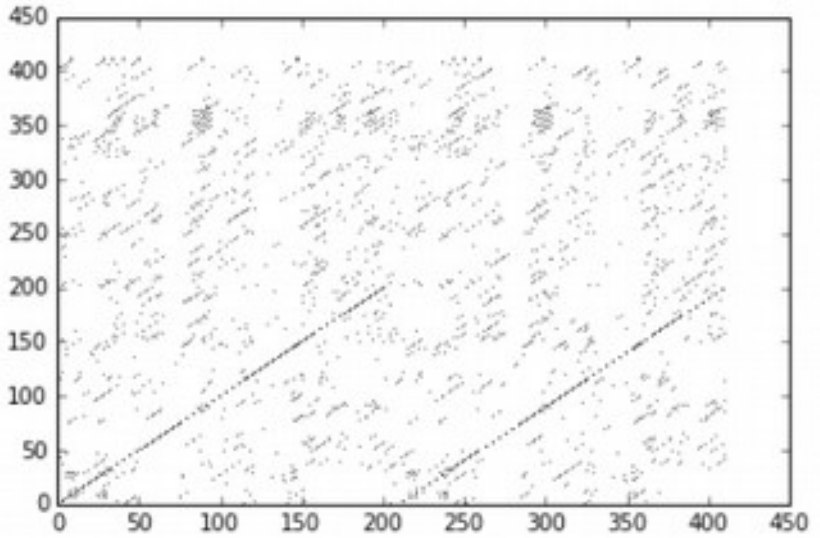
```
[[1.0, 0.2, 0.3, 0.2, 0.3, 0.2, 0.5, 0.2, 0.2, 0.3, 0.
4, 0.2, 0.0, 0.2, 0.3, 0.3, 0.3, 0.4, 0.2, 0.3, 0.3, 0
.3, 0.3, 0.4, 0.2, 0.3, 0.2, 0.2, 0.4, 0.3, 0.1, 0.3,
0.4, 0.3, 0.2, 0.2, 0.1, 0.2, 0.1, 0.4, 0.1, 0.2, 0.1,
0.4, 0.4, 0.3, 0.5, 0.4, 0.3, 0.2, 0.1, 0.4, 0.4, 0.4,
0.6, 0.3, 0.3, 0.3, 0.3, 0.4, 0.1, 0.1, 0.3, 0.5, 0.2,
0.4, 0.5, 0.1, 0.1, 0.5, 0.3, 0.5, 0.3, 0.1, 0.0, 0.2,
0.3, 0.0, 0.3, 0.4, 0.5, 0.4, 0.2, 0.2, 0.2, 0.3, 0.1,
0.3, 0.4, 0.2, 0.2, 0.2, 0.2, 0.4, 0.5, 0.3, 0.5, 0.3,
0.4, 0.0, 0.3, 0.2, 1.0, 0.2, 0.3, 0.2, 0.3, 0.2, 0.5,
0.2, 0.1, 0.4, 0.4, 0.2, 0.0, 0.2, 0.3, 0.3, 0.3, 0.4,
0.2, 0.3, 0.2, 0.3, 0.3, 0.4, 0.2, 0.3, 0.2, 0.2, 0.4,
0.0, 0.4, 0.2, 0.4, 0.3, 0.2, 0.2, 0.1, 0.2, 0.1, 0.4,
0.3, 0.2, 0.4, 0.4, 0.4, 0.3, 0.5, 0.4, 0.3, 0.2, 0.1,
0.1, 0.2, 0.7, 0.6, 0.3, 0.3, 0.3, 0.3, 0.4, 0.1, 0.1,
0.1, 0.3, 0.2, 0.4, 0.5, 0.1, 0.1, 0.5, 0.3, 0.5, 0.3,
0.3, 0.3, 0.6, 0.3, 0.0, 0.3, 0.4, 0.5, 0.4, 0.2, 0.2,
0.5, 0.1, 0.3, 0.3, 0.4, 0.2, 0.2, 0.2, 0.2, 0.4, 0.5,
0, 0.3, 0.2, 0.3, 0.3, 0.2, 0.4, 0.3, 0.1, 0.3, 0.4, 0
```

```python
def seqCompare(s1, s2):
    score = 0.0
    for i in range(0, len(s1)):
        if s1[i] == s2[i]:
            score = score + 1
    return score / len(s1)

def windowAlign(s1:str, s2:str, treshold=0.6, ws=10):
    sm = []
    for i1 in range(0, len(s1)-ws+1):
        w1 = s1[i1:i1+ws]
        #print(w1)
        rowScore = []
        for i2 in range(0, len(s2)-ws+1):
            w2 = s2[i2:i2+ws]
            score = seqCompare(w1, w2)
            if score > treshold:
                rowScore.append(score)
            else:
                rowScore.append(0)
        sm.append(rowScore)
    return sm

def readFasta(filename:str):
    f = open(filename, 'r')
    sequences = {}
    l = f.readline()
    curSeqName = ''
    while(l):
        if '>' in l:  # sequence name
            curSeqName = l.strip()[1:]
            sequences[curSeqName] = ''
        else:         # body of sequence
            sequences[curSeqName] = sequences[curSeqName] + l.strip()
        l = f.readline()
    return sequences
```

```python
s = readFasta('katG.fasta')
a = windowAlign(s['s1'], s['s2'], treshold =0.5)
colorScheme = plt.get_cmap('Greys')
plt.pcolormesh(np.array(a), cmap=colorScheme)
plt.show()
```



İhtiyacımız olan modüller:

```
from Bio import pairwise2
from Bio.SubsMat.MatrixInfo import genetic
from Bio import SeqIO
from Bio.SeqRecord import SeqRecord
from Bio.Seq import Seq
from Bio import SeqFeature
from Bio.Alphabet.IUPAC import ambiguous_dna
```

```
def scanOligo(self, target, oligo):
    '''
    :type target: Seq
    :type oligo: Seq
    '''
    consensus = []
    start = []
    end = []
    score = []
    strand = []
    for a in pairwise2.align.localds(target, oligo, genetic, -10, -0.5):
        consensus.append(self.consensus(a[0][a[3]:a[4]], a[1][a[3]:a[4]]))
        score.append(a[2])
        start.append(a[3])
        end.append(a[4])
        strand.append(0)
    for a in pairwise2.align.localds(target, oligo.reverse_complement(), genetic, -10, -0.5):
        consensus.append(self.consensus(a[0][a[3]:a[4]], a[1][a[3]:a[4]]))
        score.append(a[2])
        start.append(a[3])
        end.append(a[4])
        strand.append(1)
    return {'consensus': consensus, 'score': score, 'start': start, 'end': end, 'strand': strand}
```