Eksamensopgave i Bioinformatik og Programmering

Denne eksamensopgave består af to dele, som vægtes lige i bedømmelsen:

1. Et sæt programmeringsopgaver.
2. Et sæt bioinformatikopgaver.

Filer til brug ved eksamen

Udover denne PDF-fil som indeholder eksamensopgaverne, har du også downloaded tre andre filer fra Digital Eksamen, som du skal bruge til at løse eksamensopgaven:

- `progexam.py`: Det er i denne fil, du skal skrive de Python funktioner, der bedes om i eksamensopgavens programmeringsdel.
- `test_progexam.py`: Det er denne fil, du kan bruge til at teste de funktioner du skriver i `progexam.py`.
- `bioinfexam.py`: Det er i denne fil, du skriver svarene på eksamensopgavens bioinformatik-del.

Sådan løser du programmeringsopgaverne

Start med at åbne din terminal og naviger ind i den folder, som Digital Eksamen har lavet på din computer. Der er muligt at folderens navn indeholder mellemrum. For at navigere ind i en folder der indeholder et mellemrum vha. terminalen, kan man skrive starten af folder-navnet og så trykke på Tab. Så fuldendes navnet automatisk. F.eks.: Hvis folderen hedder "Digital Eksamen", kan man skrive: "cd Digital" og så trykke Tab. Så fuldendes navnet og man kan trykke Enter.

Som i programmeringsprojekterne fra kurset skriver du din kode i `progexam.py` og kører koden sådan her:

```
python progexam.py
```

Som i programmeringsprojekterne i kurset kan du teste din kode sådan her:

```
python test_progexam.py
```

Test scriptet er tilgængeligt som en hjælp til at teste din kode, men du har selv det fulde ansvar for rigtigheden af din kode.

Det er tilladt at bruge løsninger af opgaver til at løse senere opgaver. Man må altså gerne kalde tidligere definerede funktioner inde i andre funktioner, man senere bliver bedt om at skrive.

Følgende er *afgørende* for at din eksamensbesvarelse kan evalueres korrekt:

1. Hver funktion skal navngives *præcis* som angivet i opgaven. Funktioner der ikke er navngivet korrekt, regnes som ikke besvarede.
2. Det er ikke tilladt importere kode fra andre filer, du har skrevet eller installeret. Det vil sige, at du ikke må bruge `import` statements i din fil.
3. Når du afleverer `progexam.py` må den *kun* indeholde definitioner af de funktioner, der er beskrevet i eksamensopgaven. Al kode udenfor funktionsdefinitioner skal slettes inden du afleverer, så sørg for at teste i god tid inden aflevering, om dine funktioner stadig virker, når du sletter sådan ekstra kode.

Allervigtigst: Funktioner der ikke fuldstænding opfylder opgavens beskrivelse regnes som ikke besvarede. Så sørg for at lave dine funktioner færdige, så de klarer *alle* tests. Hvis

ingen af dine funktioner er *helt rigtigt besvaret* får du *ingen point*.

Sådan løser du bioinformatikopgaverne

Bioinformatikdelen af eksamensopgaven består af et sæt af opgaver, der hver dækker et emne. Hver opgave indeholder flere delopgaver. Der er tre typer delopgaver:

1. Udsagn der enten er sande eller falske og som skal besvares med `True` eller `False`.
2. Spørgsmål der skal besvares med et tal (`int` eller `float`).
3. Spørgsmål der skal besvares med en tekst streng (f.eks. `'Dette er mit bedste svar'`)

Filen `bioinfexam.py` er en Python fil og indeholder en variabel for hver delopgave. For eksempel: den variabel der hører til delopgave tre i emne syv hedder `emne_7_del_3`. Hver variabel har en default værdi som enten er `None` eller en tom streng (`' '`):

```
emne_7_del_3 = None
emne_7_del_4 = ' '
```

1. Du besvarer sandt/falsk udsagn ved at udskifte `None` med enten `True` eller `False`.
2. Du besvarer tal-spørgsmål ved at udskifte `None` med et tal.
3. Du besvarer tekst-opgaver ved at fylde tekst i den tomme streng.

Det er anført i `bioinfexam.py` om en delopgave skal besvares med `True`/`False`, et tal, eller tekst.

Følgende er *afgørende* for at din eksamensbesvarelse kan evalueres korrekt: Delopgaver som ikke er besvaret betragtes som forkert besvarede, *så du er bedst tjent med at gætte fremfor ikke at svare*.

I nogle af opgaveemnerne refererer statements til en vist illustration. Alt efter størrelsen på din skærm kan du være nødt til at "zoome ind" på illustrationerne i det program du bruger til at vise denne PDF, ellers kan der være detaljer du ikke kan se.

Sådan afleverer du din eksamensopgave i Digital Eksamen

Inden du afleverer, skal du tjekke at `progexam.py` kun indeholder definitioner af de funktioner der er beskrevet i eksamensopgaven. Hvis du har skrevet yderligere kode for at teste dine funktioner, skal du slette den inden du oplader din fil.

Du afleverer din eksamensbesvarelse ved at uploade disse to filer til Digital Examen:

- `progexam.py` skal afleveres som ***hoveddokument***
- `bioinfexam.py` skal afleveres som ***bilag***.

Eksamensopgaverne starter på næste side

Programming assignments

This assignment is in English to make it most like what you are used to from the programming exercises in the course.

In the assignment, Python code will look like this:

```
print('hello world')
```

Whenever I refer to Python code inside a sentence it will be styled like `this`.

You will be analyzing protein sequences and their hydrophobicity. The hydrophobicity is measured as the hydrophobicity at PH 7 (rounded to the nearest integer). So if we have a protein sequence like this:

```
'ILVSLMYFEW'
```

then we represent the hydrophobicity score for each amino acid as values in a list of the same length, like this:

```
[99, 97, 76, -5, 97, 74, 63, 100, -31, 97]
```

If you are not familiar with hydrophobicity of amino acids, you can just think of the numbers as describing some amino acid attribute that we are interested in.

The actual assignment starts below.

Happy coding.

Problem 1

You want to find the length of a protein sequence

Write a function, `seq_length`, which takes one argument:

1. A string, which represents the protein sequence.

The function must return:

- An integer, which represents the length of the protein sequence.

Example usage: If the function is called like this:

```
seq_length('ILVSLMYFEW')
```

then it should return :

```
10
```

Problem 2

The protein sequence and the list of hydrophobicity scores need to be the same length, so you would like to check whether this is the case.

Write a function, `same_length`, which takes two arguments:

1. A string, which represents the protein sequence.
2. A list of integers, which represents hydrophobicity scores of each amino acid in the protein sequence.

The function must return:

- A boolean (`True` or `False`), representing weather the two arguments have the same length.

Example usage: If the function is called like this:

```
same_length('ILV', [99, 97, 76, -5])
```

then it should return :

```
False
```

Problem 3

You want to count the number of occurrences of each amino acid in the protein sequence:

Write a function, `aminoacid_counts`, which takes one argument:

1. A string, which represents the protein sequence.

The function must return:

- A dictionary in which keys are the different amino acids in the sequence. The value associated with each key should be the number of times that the key appears in the string.

Example usage: If the function is called like this:

```
aminoacid_counts('LVLT')
```

then it should return (not necessarily with key-value pairs in that order):

```
{'L': 2, 'V': 1, 'T': 1}
```

Problem 4

You have a list of hydrophobicities for amino acids in a protein sequence and want to compute the mean (average) hydrophobicity of the amino acids in the sequence:

Write a function, `mean_hydrophobicity`, which takes one argument:

1. A list of integers, which represents hydrophobicity scores of each amino acid in a protein sequence.

The function must return:

- A float, which represents the mean of the values in the function argument.

Example usage: If the function is called like this:

```
mean_hydrophobicity([100, 0])
```

then it should return :

```
50.0
```

Problem 5

You want to compute the hydrophobicity profile across the protein sequence. For that you need to compute the mean (average) hydrophobicity score in all overlapping windows of some length:

Write a function, `running_mean`, which takes one argument:

1. A list of integers, which represents hydrophobicity scores of each amino acid in a protein sequence.
2. An integer, which represents the size of the overlapping windows.

The function must return:

- A a list of floats, which represents the mean hydrophobicity of each overlapping window.

Example usage: If the function is called like this:

```
running_mean([1, 2, 3, 4, 5, 6], 2)
```

then it should return :

```
[1.5, 2.5, 3.5, 4.5, 5.5]
```

Problem 6

You have a protein sequence and want to extract a list of the amino acids with a hydrophobicity score larger than zero:

Write a function, `find_hydrophobic`, which takes two arguments:

1. A string, which represents the protein sequence.
2. A list of integers, which represent hydrophobicity scores of each amino acid in the protein sequence.

The function must return:

- A list of strings, which represents each amino acid in the protein sequence with a hydrophobicity score above zero.

Example usage: If the function is called like this:

```
find_hydrophobic('FIPNL', [100, 99, -46, -55, 97])
```

then it should return :

```
['F', 'I', 'L']
```

Problem 7

You have a protein sequence and want to produce a version of that sequence where the characters for all amino acids with a hydrophobicity score above zero have been converted to lower-case.

Write a function, `mask_hydrophobic_aa`, which takes two arguments:

1. A string, which represents the protein sequence.
2. A list of integers, which represents hydrophobicity scores of each amino acid in the protein sequence.

The function must return:

- A string, which presents a copy of argument one, where amino acids with a hydrophobicity score above zero have been converted to lower-case.

Example usage: If the function is called like this:

```
mask_hydrophobic_aa('FIPGL', [100, 99, -46, 0, 97])
```

then it should return :

```
'fiPGl'
```

Problem 8

You want to extract the hydrophobic subsequences from a protein sequence. A hydrophobic subsequence is here defined as the consecutive sequences of amino acids with a hydrophobicity above zero.

Write a function, `hydrophobic_subseqs`, which takes two arguments:

1. A string, which represents the protein sequence.
2. A list of integers, which represents hydrophobicity scores of each amino acid in the protein sequence.

The function must return:

- A list of strings, which represents the subsequences of argument one that only contain amino acids with a hydrophobicity above zero.

Example usage: If the function is called like this:

```
hydrophobic_subseqs('ILVGLMYFEW', [99, 97, 76, 0, 97, 74, 63, 100, -31, 97])
```

then it should return :

```
['ILV', 'LMYF', 'W']
```

Problem 9

You want to find the mean (average) hydrophobicity of the amino acids next to (neighboring) each of the different kinds of amino acids in your sequence. In this example sequence: `'GVGL'`, the neighbors of the first `G` is `V` and the neighbors of the second `G` are `V` and `L`. The hydrophobicity of `V` and `L` are 76 and 97. So to compute the mean hydrophobicity of neighbors of Guanosines you would do: (76 + 76 + 97) / 3. Note that the same `V` is counted twice because it is the neighbor of two different `G`s. You want to do this for each of the different amino acids that appear in your sequence. You can assume the sequence is at least two amino acids long.

Write a function, `neighbor_hydrophobicity`, which takes two arguments:

1. A string, which represents the protein sequence.
2. A list of integers, which represents hydrophobicity scores of each amino acid in the protein sequence.

The function must return:

- A dictionary of floats. The keys in the dictionary must be the different amino acids in the sequence. The value associated with each amino acid key must be the mean hydrophobicity of all neighbors of each occurrence of that amino acid.

Example usage: If the function is called like this:

```
neighbor_hydrophobicity('GVGL', [0, 76, 0, 97])
```

then it should return (not necessarily with key-value pairs in that order):

```
{'G': 83.0, 'V': 0.0, 'L': 0.0}
```

Bioinformatikopgaver

Emne 1: Helgenoms-associationsstudier

GWAS (Genome wide association studies) betegner studier hvor man forsøger at associere genetiske varianter med f.eks. en genetisk sygdom.

Udsagn og/eller spørgsmål:

1. GWAS finder typisk association til varianter som enten har lav frekvens men stor effekt eller høj frekvens men lille effekt. (True/False)
2. Hvis en variant er signifikant associeret med en sygdom i en GWAS, så er det fordi denne variant repræsenter en skadelig mutation. (True/False)
3. Koblingsuligevægt (linkage disequilibrium) mellem loci nær ved hinanden gør det muligt at lave en GWAS uden at sekventere hele genomet. (True/False)
4. I GWAS er en association significant hvis p-værdien er mindre end 0.05. (True/False)

Emne 2: Genome assembly

Den bioinformatiske udfordring det er at samle sekvens reads til et samlet genom kaldes genome assembly.

Udsagn og/eller spørgsmål:

1. Repeats i en sekvens kan kun samles hvis længden af reads er større end længden af det enkelte repeat. (True/False)
2. Genome contigs er sat sammen af flere scaffolds. (True/False)
3. N50 representerer antallet af baser i et assembly der er indeholdt i de 50 længste sekvenser. (True/False)
4. Givet følgende længder af contigs 12, 13, 27, 35, 67, 80, 87, 88, 102, 107, hvad er N50? (talværdi)
5. Hvis alle vores sekventerede baser har en Phred score på 40, hvor mange baser per 1,000,000 forventer vi så er forkert identificeret? (talværdi)
6. Paried-end reads kan benyttes til at identificere hvilke scaffolds der placeret på hver side af en repetitiv region. (True/False)

Emne 3: Needleman-Wunch og Smith-Waterman algoritmerne

Du har det splicede transcript af et gen med flere exons (dvs. uden intros). Du vil gerne aligne denne sekvens til den genomiske (usplejsede) sekvens, som du tror dit splejsede transkript stammer fra.

Udsagn og/eller spørgsmål:

1. Du kan her med fordel benytte Smith-Waterman algoritmen. (True/False)
2. Du kan her med fordel benytte afine cost til at score gaps. (True/False)
3. Du kan her med fordel benytte en lav gap cost for gaps i enderne. (True/False)
4. Når der benyttes afine cost i Needleman-Wunch algoritmen, er tiden det kræver at beregne et globalt alignment proportional med produktet af de to sekvensers længde. (True/False)
5. Både Needleman-Wunch og Smith-Waterman algoritmen identificerer et parvist alignemnt, der er optimalt givet en scoringsmatrix og en gap score. (True/False)
6. Når man benytter Needleman-Wunch algoritmen er der aldrig mere end ét optimalt alignment for en given scoringsmatrix og gap score. (True/False)
7. Tiden det kræver at beregne et lokalt alignment ved hjælp af Smith-Waterman algoritmen, er proportional med summen af de to sekvensers længde. (True/False)
8. Smith-Waterman algoritmen vil i nogen tilfælde producere et globalt alignment. (True/False)

Emne 4: Globalt alignment af to korte sekvenser

Nedenfor ser du to korte DNA sekvenser:

```
sekvens 1: TGG
sekvens 2: AATG
```

Lav et globalt alignment af de to sekvenser vha. Needleman-Wunsch algoritmen. Du skal bruge en match score på 1, en mismatch score på -1 og en gap score på -2 . Du kan gøre dette ved at udfylde en tabel vha. dynamisk programmering.

Udsagn og/eller spørgsmål:

1. Hvad er den optimale alignment score? (talværdi)
2. Hvor mange optimale alignments er der? (talværdi)
3. Der findes et optimalt alignemnt, der har tre gaps. (True/False)
4. Der findes et optimalt alignment der har to gaps. (True/False)
5. Alle optimale alignments aligner de to sidste baser i hver sekvens (G) med hindanden. (True/False)

Emne 5: Lokalt alignment af to korte sekvenser

Nedenfor ser du en dynamisk programmerings matrix for Smith-Waterman alignment af følgende to sekvenser:

```
sekvens 1: CCTCA
sekvens 2: CCCG
```

		C	C	C	G
	0	0	0	0	0
C	0	1	1	1	0
C	0	1	2	2	0
T	0	0	0	1	1
C	0	1	1	1	0
A	0	0	0	0	0

Der er brugt en match score på 1, en mismatch score på -1, og en gap score på -2.

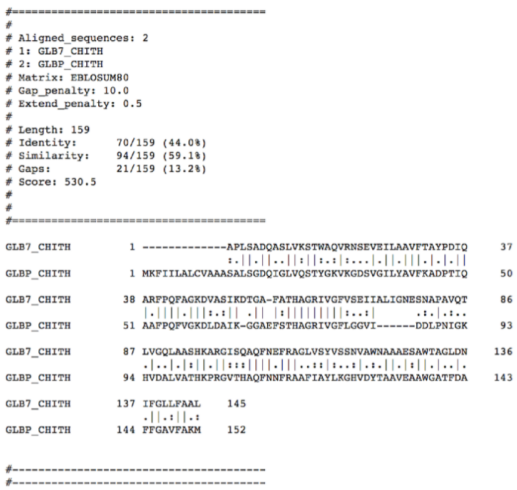
Udsagn og/eller spørgsmål:

1. Havd er den optimale alignment score? (talværdi)
2. Hvor mange optimale alignments er der? (talværdi)
3. Der findes et optimalt alignment der spænder tre baser i sekvens 1 og to baser i sekvens 2. (True/False)
4. Der findes et optimalt alignment hvor baserne AA fra sekvens 1 aligner til baserne AA fra sekvens 2. (True/False)

Emne 6: Parvist alignment ved hjælp af EBIs web tools.

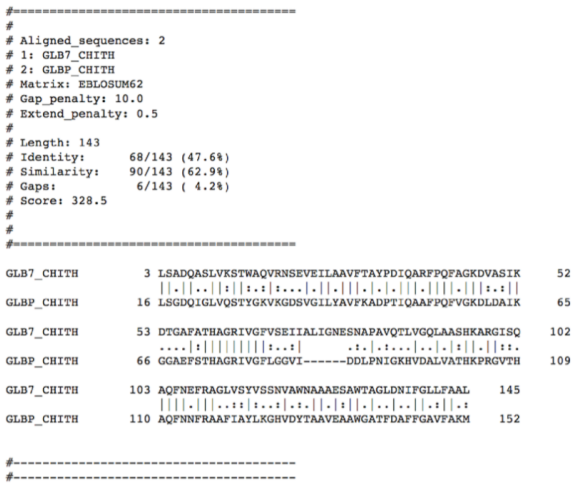
Nedenfor ser du to screen shots med output fra to parvise alignments foretaget ved hjælp af EBIs webservice.

Alignment A



```
#=======================================
#
# Aligned_sequences: 2
# 1: GLB7_CHITH
# 2: GLBP_CHITH
# Matrix: EBLOSUM80
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 159
# Identity:      70/159 (44.0%)
# Similarity:    94/159 (59.1%)
# Gaps:          21/159 (13.2%)
# Score: 530.5
#
#
#=======================================

GLB7_CHITH         1 -------------APLSADQASLVKSTWAQVRNSEVEILAAVFTAYPDIQ     37
                                  :.||.||..||:||:.:|:...|.||.|||.|.|.||
GLBP_CHITH         1 MKFIILALCVAAASALSGDQIGLVQSTYGKVKGDSVGILYAVFKADPTIQ     50

GLB7_CHITH        38 ARFPQFAGKDVASIKDTGA-FATHAGRIVGFVSEIIALIGNESNAPAVQT     86
                     |.||||.|||:.:|| .|| |:|||||||||:..:|      .:.|.:..
GLBP_CHITH        51 AAFPQFVGKDLDAIK-GGAEFSTHAGRIVGFLGGVI------DDLPNIGK     93

GLB7_CHITH        87 LVGQLAASHKARGISQAQFNEFRAGLVSYVSSNVAWNAAAESAWTAGLDN    136
                     .|..|.|:||.||:::||||.|||..:::|:.:|.:.||.|:||.|...|.
GLBP_CHITH        94 HVDALVATHKPRGVTHAQFNNFRAAFIAYLKGHVDYTAAVEAAWGATFDA    143

GLB7_CHITH       137 IFGLLFAAL    145
                     .||.:||.:
GLBP_CHITH       144 FFGAVFAKM    152


#---------------------------------------
#---------------------------------------
```

Alignment B



```
#=======================================
#
# Aligned_sequences: 2
# 1: GLB7_CHITH
# 2: GLBP_CHITH
# Matrix: EBLOSUM62
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 143
# Identity:      68/143 (47.6%)
# Similarity:    90/143 (62.9%)
# Gaps:           6/143 ( 4.2%)
# Score: 328.5
#
#
#=======================================

GLB7_CHITH         3 LSADQASLVKSTWAQVRNSEVEILAAVFTAYPDIQARFPQFAGKDVASIK     52
                     ||.||..||:||:.:|:...|.||.|||.|.|.|||.||||.|||:.:||
GLBP_CHITH        16 LSGDQIGLVQSTYGKVKGDSVGILYAVFKADPTIQAAFPQFVGKDLDAIK     65

GLB7_CHITH        53 DTGAFATHAGRIVGFVSEIIALIGNESNAPAVQTLVGQLAASHKARGISQ    102
                     ....|:|||||||||:..:|      .:.|.:...|..|.:||.||::.
GLBP_CHITH        66 GGAEFSTHAGRIVGFLGGVI------DDLPNIGKHVDALVATHKPRGVTH    109

GLB7_CHITH       103 AQFNEFRAGLVSYVSSNVAWNAAAESAWTAGLDNIFGLLFAAL    145
                     ||||.|||..::|:..:|.:.||.:||.|..|..:||.:||.:
GLBP_CHITH       110 AQFNNFRAAFIAYLKGHVDYTAAVEAAWGATFDAFFGAVFAKM    152


#---------------------------------------
#---------------------------------------
```

Udsagn og/eller spørgsmål:

1. Alignment B er et lokalt alignment. (True/False)
2. Alignment A er et globalt alignment. (True/False)
3. I alignment A vil et gap med længde syv bidrage til den totale score med en score på -13. (True/False)
4. Til at producere alignment B er der brugt en substitutionsmatrix, der er bedre egnet til fjernt beslægtede sekvenser end den, der er brugt til at producere alignment A. (True/False)

Emne 7: Multipelt alignment

Der findes flere forskellige programmer til multipelt alignment. Et af dem er Clustal Omega, men der er mange andre.

Udsagn og/eller spørgsmål:

1. Needleman-Wunch algoritmen kan udvides til at aligne tre sekvenser ved at fylde en tabel ud i tre dimentioner i stedet for to. (True/False)
2. Heuristiske alignment algoritmer er garanteret at producere et optimalt alignment givet en scoringsmatrix og gap score. (True/False)
3. Jo flere sekvenser man aligner, jo bedre vil man kunne identificere homologi mellem fjernt beslægtede sekvenser. (True/False)
4. Mutationer der indsætter sekvens (insertions) introducerer typisk flere gap karakterer i et multiplet alignemnt end mutationer der fjerner sekvens (deletions). (True/False)

Emne 8: Blast

BLAST programmet implementerer en algoritme til at søge med en sekvens i en stor database af sekvenser. NCBI giver online adgang til BLAST mod sekvenserne i Genbank, men man kan også lave sin egen lille database på sin egen computer og så søge mod den. BLAST har flere parametre der har indflydelse på resultatet, to af dem er ordstørrelsen W (word size) og tærskelværdien T (neighborhood word threshold score).

Udsagn og/eller spørgsmål:

1. Når BLAST identificerer homologi mellem to sekvenser begynder den med at identificere korte del-sekvenser der findes i begge sekvenser. (True/False)
2. Den score, der angives for hver identificeret databasesekvens, repræsenterer en alignment score for et lokalt alignment af en sekvens i databasen og den sekvens man søger med. (True/False)
3. E-værdien angiver sandsynligheden for at finde et match i databasen med samme eller lavere score ved rent tilfælde. (True/False)
4. E-værdien forventes at være større, hvis man spøger mod en stor database, end hvis man søger mod en lille database. (True/False)
5. BLAST analysen vil være hurtigere hvis T parameterens værdi er reduceret. (True/False)
6. BLAST analysen vil være hurtigere hvis W parameterens værdi er reduceret. (True/False)
7. Sensitiviteten af BLAST bliver højere når værdien af T parameteren reduceres. (True/False)
8. Sensitiviteten af BLAST bliver højere når værdien af W parameteren reduceres. (True/False)

Emne 9: Modeller for DNA evolution

Jukes-Cantor modellen og Kimura 2-parameter modellen er modeller for DNA evolution.

Udsagn og/eller spørgsmål:

1. Begge modeller kan bruges til at udregne den evolutionære afstand mellem to sekvenser ud fra andelen af forskelle mellem dem. (True/False)
2. Den evolutionære afstand de to modeller udregner er større end andelen af forskelle mellem de to sekvenser. (True/False)
3. Jukes-Cantor modellen lader os korrigere for at den samme positition i sekvensen har muteret flere gange. (True/False)
4. Kimura 2-parameter modellen korrigerer for, at transversioner og transitioner optræder med forskellige hyppigheder. (True/False)
5. Jukes-Cantor modellen tager højde for, at de fire baser optræder med forskellig hyppighed i DNA sekvenser. (True/False)
6. Andelen af forskelle mellem to sekvenser kan maksimalt være 1.0 (dvs. alle alignede baser er forskellige), men hvad er den maksimale evolutionare afstand som kan regnes ud vha. Jukes-Cantor modellen (angiv to decimaler)? (talværdi)

Emne 10: PAM og BLOSUM matricer

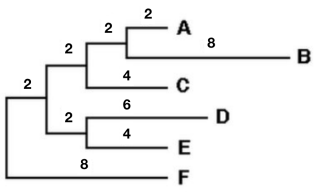
Når man konstruerer substitutionmatricer for proteiner (PAM og BLOSUM matricer), så beregnes hver score ud fra information i et sæt alignments.

Udsagn og/eller spørgsmål:

1. Størrelsen på hver indgang i en PAM1 scoringsmatrix repræsenterer, hvor ofte vi forventer at se en aminosyre blive erstattet med en anden, når man tager frekvensen af forskellige aminosyrer i betragtning. (True/False)
2. PAM80 og BLOSUM80 repræsenterer omtrent lige store evolutionære afstande. (True/False)
3. PAM160 og BLOSUM60 repræsenterer omtrent lige store evolutionære afstande. (True/False)
4. PAM120 er konstrueret ved ekstrapolation fra en PAM1 matrix. (True/False)
5. Jo hyppigere en aminosyre er, jo større score vil substitioner til denne aminosyre have i en PAM matrix. (True/False)

Emne 11: Afstande på et træ

Nedenfor ser du et træ for seks sekvenser: A, B, C, D, E og F. Hver grens længde er angivet.



Udsagn og/eller spørgsmål:

1. Afstanden mellem B og C er 14. (True/False)
2. A og D er tættere beslægtet end E og B. (True/False)
3. Træets totale grenlængde er 40. (True/False)
4. Afstandene på træet indikerer at substitutionsraten er den samme i hele træet. (True/False)
5. Beskriv hvilken egenskab ved molekylær evolution der skal være opfyldt for at vi kan tale om et mylekylært ur. (tekst, max 300 karakterer incl. mellemrum)

Emne 12: UPGMA

Tabellen nedenfor viser de parvise afstande mellem fire sekvenser: A, B, C og D.

	A	B	C	D
A				
B	4			
C	10	10		
D	10	10	6	

De fire sekvensers indbyrdes afstande kan repræsentres som et træ ved hjælp af clustering algoritmen UPGMA. Det konstruerede træ har ingen rod.

Udsagn og/eller spørgsmål:

1. UPGMA vil begynde med at clustre A og B. (True/False)
2. UPGMA vil slutte med at sammensmelte ("joine") et cluster af A og B med et cluster af C og D. (True/False)
3. Den korteste gren i det konstrurerede træ har længde 2. (True/False)
4. Hvis vi byggede et træ med kun sekvenserne A, B og C, så ville vi kunne bruge sekvens D som outgroup på dette træ. (True/False)

Emne 13: Fylogeni

Metoder til konstruktion af fylogenier kan opdeles i karakterbaserede metoder og metoder der clustrer sekvenser baseret på forudberegnede afstande mellem par af sekvenser.

Udsagn og/eller spørgsmål:

1. Neighbor-Joining og UPGMA benytter clustering til konstruktion af fylogeni. (True/False)
2. PhyML er et program til konstruktion af en fylogeni. (True/False)
3. Neighbor-joining algoritmen tager højde for at mutationsraten ikke er konstant. (True/False)
4. UPGMA algoritmen tager højde for at mutationsraten ikke er konstant. (True/False)

Emne 14: Felsensteins algoritme

Felsensteins algoritme benyttes i karakterbaserede metoder til konstruktion af fylogeni.

Udsagn og/eller spørgsmål:

1. Felsenteins algoritme finder det bedste træ givet et sæt sekvenser. (True/False)
2. Felsensteins algoritme beregner sandsynligheden for et givet træ med en given base på hvert blad. (True/False)
3. Felsensteins algoritme antager uafhængighed mellem mellem alignment kolonner. (True/False)
4. Den sandsynlighed, Felsensteins algoritme producerer, afhænger af hvilken model for molekylær evolution der benyttes. (True/False)
5. Felsensteins algoritme benytter rekursion. (True/False)

Emne 15: Generelt om skjulte Markov modeller

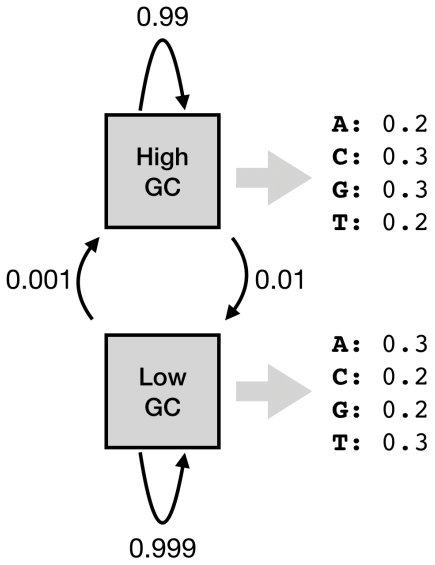
Skjulte Markov modeller (hidden Markov models, eller HMM) er en klasse af matematiske modeller, der ofte benyttes i bioinformatiske sammenhænge.

Udsagn og/eller spørgsmål:

1. En skjult Markov models transitionssandsynligheder angiver, hvor sandsynligt det er at gå fra en skjult tilstand (state) til en anden. (True/False)
2. Skjulte Markov modeller benyttes blandt andet til at identificere gener i et genom. (True/False)
3. En skjult Markov models parametre skal estimeres, før modellen kan anvendes til forudsigelse. (True/False)
4. Viterbi algoritmen beregner de optimale transitions- og emmissions-sandsynligheder. (True/False)
5. Forward algoritmen identificerer den mest sandsynlige sti gennem den skjulte Markov model. (True/False)
6. Posterior decoding identificerer en serie af skjulte tilstande, der repræsenterer en sti gennem en skjult Markov model. (True/False)

Emne 16: En simpel skjult Markov model

Herunder ses et eksempel på en simpel skjult Markov model, der kan kategorisere en genomisk sekvens i områder med henholdsvis højt og lavt GC-indhold.



Udsagn og/eller spørgsmål:

1. Hvad er transitionssandsynligheden fra "High GC" til "Low GC"? (talværdi)
2. Modellens parametre angiver, at regioner med højt GC-indhold generelt er længere end regioner med lavt GC-indhold. (True/False)
3. Hvad er emmisionssandsynligheden for basen G i en region med lavt GC-indhold? (talværdi)
4. Hvad er den forventede længde (i antal baser) af regioner med lavt GC-indhold givet modellens parametre? (talværdi)

Emne 17: Genforudsigelse med GenScan

GenScan er et stykke videnskabeligt software, der benyttes til at forudsige gener i blandt andet det menneskelige genom. Nedenfor kan du se et screenshot med output fra GenScan.

```
Gn.Ex Type S .Begin ...End .Len Fr Ph I/Ac Do/T CodRg P.... Tscr..
----- ---- - ------ ------ ---- -- -- ---- ---- ----- ----- ------

3.00 Prom +  60508  60547   40                                -6.55
3.01 Init +  69431  69483   53  1  2   76   38    53 0.390  -0.22
3.02 Intr +  73872  74051  180  0  0   74   92   121 0.561   9.16
3.03 Intr +  74690  74862  173  2  2   79   24   186 0.918   9.96
3.04 Term +  74959  75290  332  2  2  -25   42   251 0.662   3.03
3.05 PlyA +  76348  76353    6                                1.05
```

Udsagn og/eller spørgsmål:

1. Genet som GenScan har fundet ligger på "forward strand". (True/False)
2. Hvor mange exons har det gen som GenScan har fundet? (talværdi)
3. Hvad er længden af den første exon i det gen GenScan har fundet? (talværdi)
4. Hvad er længden af det splicede mRNA for det gen GenScan har fundet? (talværdi)

Emne 18: Neurale netværk

Neurale netværk er en klasse af matematiske modeller, der benyttes i bioinformatik.

Udsagn og/eller spørgsmål:

1. Neurale netværk er repræsenteret ved en graf med et sæt forbundne knuder, der hver representer en "neuron". (True/False)
2. Neurale netværk kan anvendes til klassifikation af sekvenser. (True/False)
3. Et neuralt netvært har parametre, som skal estimeres før det anvendes til forudsigelse. (True/False)
4. Et neuralt netvært har typisk et eller flere skjulte lag mellem et input og et output lag. (True/False)
5. Input værdien til hvert neuron neuron er vægtede bidrag fra neuroner i det tidligere lag af neuroner. (True/False)
6. Når man siger at et lag i et neuralt netvært er "fully connected", så betyder det at hvert neuron er forbindet med alle neuroner i laget før og laget efter. (True/False)