RAMAIAH UNIVERSITY OF APPLIED SCIENCES

Faculty of Engineering & Technology			
Ramaiah University of Applied Sciences			
Department	Computer Science and Engineering	**Programme**	B. Tech.
Semester/Batch	5th /2021		
Course Code	20ISC315A	**Course Title**	Bioinformatics
Course Leader	Ms Naganandini G		

Assignment			
Name of Student	**R.KAARTHIGEYAN**	Register No	**21ETIS411028**

Sections		Marking Scheme	Max Marks	First Examiner Marks	Second Examiner Marks
Part-A	**A1-A5**	Scenario based questions to have a good insight on Bioinformatics Data, Databases, Data Format, Database Search, Data Retrieval Systems, and Genome Browsers.	2x5		
		Max Marks	**10**		
Part –B	**B1.1**	Solve using any one of the sequence alignment algorithm with all the necessary steps and the necessary outcome.	7		
	B1.2	Writing a python code and the relevant output for the solved sequence alignment algorithm using a python/biopython code.	8		
		Max Marks	**15**		

Course Marks Tabulation				
Component- CET B Assignment	**First Examiner**	**Remarks**	**Second Examiner**	**Remarks**
Marks (out of 25)				

Signature of First Examiner

Signature of Second Examiner

RAMAIAH UNIVERSITY
OF APPLIED SCIENCES

Assignment

Instructions to students:

1. The assignment consists of 2 parts –
   Part-A
   Part-B
2. Maximum marks is 25.
3. The assignment has to be neatly word processed and filed as per the prescribed format.
4. The printed assignment must be submitted to the subject leader.
5. **Submission Date: 30 March 2024**
6. **Submission after the due date is not permitted.**
7. **IMPORTANT**: It is essential that all the sources used in preparation of the assignment must be suitably referenced in the text.
8. Marks will be awarded only to the sections and subsections clearly indicated as per the problem statement/exercise/question.
9. Source that can be referenced for solving and coding
   Needleman Wunsch-https://youtu.be/ipp-pNRIp4g?si=4gt4nr6lMQCx3sBc
   Smith Waterman-https://youtu.be/lu9ScxSejSE?si=vzBfkf5w1P88EZ71
10. Refer this link for notes and to answer the questions of part-A.
    https://drive.google.com/drive/folders/1UG_IO2DkS6wIZe5OdqLiG5uk75yUahFE

Preamble:

This course encompasses fundamental concepts in gene and genome studies, molecular evolution, genomic technologies, and bioinformatics, addressing topics such as DNA structure, gene mutations, molecular evolution theories, genomic sequencing technologies, and data management techniques. Through a comprehensive exploration of biological macromolecules, molecular evolution principles, genomic technologies, and bioinformatics methodologies, readers gain insights into the intricate interplay between genetics, genomics, and computational biology, paving the way for a deeper understanding of the complexities of biological systems and their applications in modern science.

RAMAIAH
UNIVERSITY
OF APPLIED SCIENCES

Part –A:

Scenario 1: You're explaining the concept of bioinformatics to a friend who is confused about its distinction from computational biology.

A 1. How would you differentiate between bioinformatics and computational biology, providing clear examples of each?

Ans. **Bioinformatics:**

- Focuses on **developing and applying computational tools** to manage and analyze biological data.
- Leverages existing knowledge and databases to **extract meaning** from large datasets like DNA sequences.
- Requires strong skills in **programming, statistics, and data management**.

Example: A bioinformatician might use software to compare DNA sequences from different organisms to identify potential disease-causing genes.

Computational Biology:

- Emphasizes **building theoretical models and simulations** of biological systems.
- Aims to **understand complex biological processes** through mathematical and computational methods.
- Requires a strong foundation in **mathematics, physics, and computer science**.

Example: A computational biologist might develop a computer model to simulate protein folding, a crucial process for protein function.

Here's an analogy: Bioinformatics is like being a data librarian in the biological world, organizing and interpreting information. Computational biology is like being a biological architect, using that information to design and understand the structures and functionalities of life.

Scenario 2: You're tasked with analyzing a large dataset of gene expression data to identify potential biomarkers for a specific disease.

A 2. What specific goals would you set for your bioinformatics analysis, and what tools or methods would you employ to achieve them?

Ans. **Bioinformatics Analysis for Disease Biomarker Discovery**

Goals:

1. **Identify Differentially Expressed Genes (DEGs):**
    - Use statistical tests to compare gene expression levels between healthy and diseased samples.
    - Focus on genes with significant changes (up or downregulation) in the disease state.

2. **Functional Enrichment Analysis:**

    - Group DEGs based on their known biological functions (e.g., using Gene Ontology).
    - Identify pathways or processes significantly enriched in the DEGs.
    - This helps narrow down the search to genes potentially involved in the disease mechanism.

3. **Prioritization and Validation:**

    - Analyze DEGs within enriched pathways for characteristics of good biomarkers (e.g., tissue specificity, ease of detection).
    - Prioritize a smaller set of promising candidates for further experimental validation using techniques like qPCR or western blotting.

Tools and Methods:

- **Differential Expression Analysis:** Tools like DESeq2, EdgeR, or Limma can be used to identify DEGs with appropriate statistical cutoffs (adjusted p-value).
- **Functional Enrichment Analysis:** Gene Ontology (GO) resources or databases like KEGG can be used to assign functions and identify enriched pathways using tools like DAVID or clusterProfiler.
- **Network Analysis:** Tools like STRING or Cytoscape can be used to explore interactions between DEGs and identify potential regulatory networks.

Additional Considerations:

- **Data Quality Control:** Ensure the gene expression data is properly normalized and filtered for outliers or technical artifacts.
- **Batch Effects:** Account for potential batch effects arising from different experimental runs to avoid misleading results.
- **Machine Learning:** Consider incorporating machine learning algorithms for more robust biomarker selection and classification.

By following these steps and using the appropriate tools, we can increase the chances of identifying reliable and informative biomarkers for further disease diagnosis and development of targeted therapies.

Scenario 3: You're working in a genomics lab and need to convert DNA sequence data from FASTA format to GenBank format for submission to a public database.
A3. Describe the process of converting sequence formats using tools like Readseq, highlighting any potential issues or considerations.

Ans. While Readseq is a great tool for general sequence manipulation, it's not ideal for FASTA to GenBank conversion. Here's a breakdown using more suitable options:

Conversion Process:

1. **Online Converters:** Websites like NCBI sequence converter: https://www.ncbi.nlm.nih.gov/genbank/fastaformat/ offer a user-friendly interface to upload your FASTA file and download it in GenBank format.

2. **Command-line Tools:**

   - **Seqret (EMBOSS package):** This powerful tool allows conversion between various sequence formats. The command would be: `seqret input.fasta output.gb` (replace with your filenames).
   - **Biopython (Python library):** If you're comfortable with Python, Biopython's SeqIO module allows programmatic conversion with more control over annotations.

Considerations:

- **FASTA Limitations:** Basic FASTA files only contain sequence and ID. GenBank requires additional information like organism source, features (genes, exons), etc.

  - For online converters, you might need to provide this information manually.
  - Command-line tools might offer options to include basic annotations.

- **Accuracy and Completeness:** Ensure the converted GenBank file adheres to the database submission guidelines. Double-check for missing information or formatting errors.

- **Large Datasets:** Online converters might have limitations on file size. For large datasets, consider command-line tools for better efficiency.

By choosing the right tool and considering these points, you can effectively convert your FASTA data to GenBank format for successful submission to a public database.

Scenario 4: You're conducting research on a gene associated with a rare genetic disorder and need to retrieve relevant information from primary and secondary databases.
A4. How would you navigate and retrieve data from these databases, and what visualization tools or genome browsers would you use to analyze the data effectively?

Ans. **Retrieving Information for Rare Genetic Disorder Research**

Here's how to navigate and retrieve data from primary and secondary databases for your rare genetic disorder research, along with helpful visualization tools:

Databases:

4. **Primary Databases (Sequence and Variation Data):**

    - **GenBank:** (https://www.ncbi.nlm.nih.gov/genbank/) Stores DNA sequences, including your gene of interest.
        - Search by gene name or accession number. Download the sequence and associated annotations.
    - **ClinVar:** (https://www.ncbi.nlm.nih.gov/clinvar/) Provides information on human variations and their relationship to disease.
        - Search by gene name or variant ID for reported variants associated with the disorder.

5. **Secondary Databases (Literature and Functional Data):**

    - **OMIM (Online Mendelian Inheritance in Man):** (https://www.ncbi.nlm.nih.gov/omim) Catalogs human genes and genetic disorders.
        - Search by gene name or disorder name for clinical information, inheritance patterns, and associated variants.
    - **UniProt:** (https://www.uniprot.org/) Offers protein sequence and function information.
        - Search by gene name to find the protein sequence, functional domains, and potential interactions.
    - **DisGeNET:** (https://www.disgenet.org/) Links genes and genetic variants to diseases.
        - Search by gene name for known disease associations and supporting evidence from publications.

Data Navigation and Retrieval:

- Each database has its own search interface and functionalities. Utilize search filters, keywords, and browsing options to find relevant data.
- Pay attention to data types (e.g., DNA sequence, protein structure, variant classifications).
- Download data in appropriate formats (e.g., FASTA, CSV) for further analysis.

Visualization Tools and Genome Browsers:

- **UCSC Genome Browser:** (https://genome.ucsc.edu/) Allows visualization of your gene within the genome context, including surrounding genes, regulatory elements, and known variants.
- **Integrative Genomics Viewer (IGV):** (https://www.broadinstitute.org/scientific-community/software/integrative-genomics-viewer) Enables visualization of your gene sequence alongside various data tracks like RNA-seq data, methylation patterns, and ClinVar variants.

- **Cytoscape:** (https://cytoscape.org/) Helps visualize interactions between your gene of interest and other proteins involved in the disease pathway.

Effective Analysis:

- Use retrieved data to understand the gene's sequence, function, known variants, and disease associations.
- Leverage visualization tools to identify potential functional domains, variant locations within the gene, and their potential impact on protein structure or function.
- Integrate information from various databases to build a comprehensive picture of the gene's role in the rare genetic disorder.

By effectively utilizing these resources, you can gain valuable insights into the gene's function and its contribution to the rare genetic disorder you're researching.

Scenario 5: You're collaborating with a team of geneticists to annotate a newly sequenced genome and visualize genetic variations across different populations.
A5. How would you use genome browsers and map viewers to visualize and interpret the genomic data, and what features would you prioritize for analysis?

Ans. **Using Genome Browsers and Map Viewers for Collaborative Genome Annotation**

Genome browsers and map viewers are powerful tools for visualizing and interpreting genomic data, especially when collaborating with geneticists to annotate a newly sequenced genome and compare variations across populations. Here's how we can leverage them:

Visualization and Interpretation:

1. **Initial Annotation:**
    - Use a genome browser like **UCSC Genome Browser** or **Ensembl** to visualize the assembled genome alongside reference genomes of related species.
    - Identify genes, regulatory elements, and repetitive regions by looking at pre-loaded annotation tracks.
    - Collaborate with geneticists to manually annotate novel genes and features based on sequence homology or known functional motifs.
2. **Variant Visualization:**
    - Upload variant data from different populations (e.g., SNPs, indels) to the genome browser.
    - Utilize variant call format (VCF) files and filter by population, variant type, or functional impact (missense, nonsense).

- Visualize the distribution of variants across the genome and identify regions with high variation.

Features for Analysis:

- **Gene Tracks:** Overlay gene annotations with variant tracks to identify potential functional consequences (e.g., variants within coding regions or regulatory elements).
- **Conservation Tracks:** Compare the newly sequenced genome with conserved regions in related species to prioritize functionally important regions less likely to tolerate variation.
- **Population Frequency Tracks:** Visualize the allele frequencies of variants across different populations. Identify variants with high frequency differences that might be population-specific adaptations.
- **ClinVar Tracks:** Check for known disease-associated variants within the newly sequenced genome to identify potential risks.

Collaboration and Communication:

- Share visualization links or screenshots with the geneticist team for real-time discussion and annotation refinement.
- Utilize annotation tools within the browser to add notes, highlight regions of interest, and share interpretations.

Additional Considerations:

- Choose a genome browser with features suitable for your specific needs (e.g., advanced variant filtering, population genetics tools).
- Ensure all collaborators are familiar with the chosen browser's interface and functionalities.
- Maintain clear documentation of annotations and interpretations for future reference.

By effectively utilizing genome browsers and map viewers with these strategies, you can collaboratively annotate the new genome, identify population-specific variations, and gain valuable insights into the genetic landscape of the studied organism.

Part –B:

Scenario: You're tasked with aligning two DNA sequences to identify similarities and differences between them. The sequences provided are:

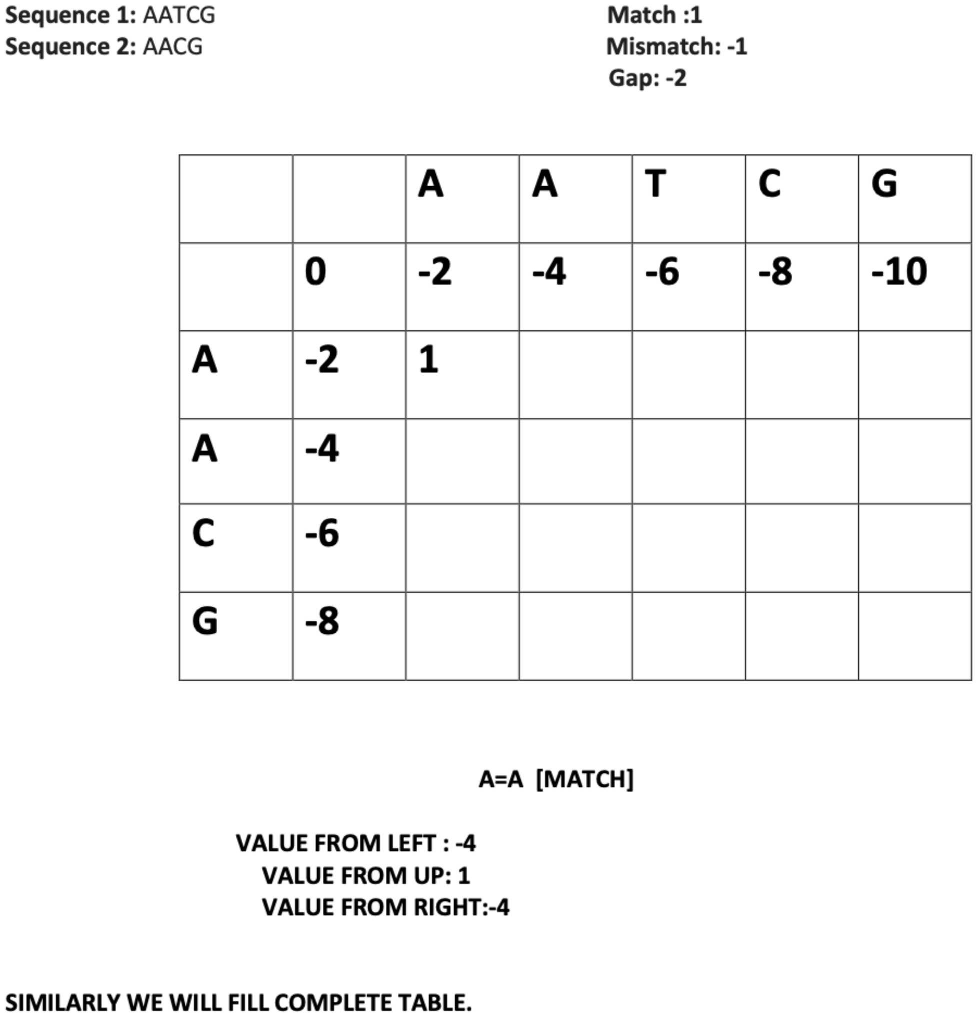
B1.1 Solve:
Sequence 1: AATCG
Sequence 2: AACG
Consider the following rewards and penalties-

Match :1
Mismatch: -1
Gap: -2
Using the Needleman-Wunsch algorithm, compute the optimal global alignment score and the corresponding alignment for these sequences.

OR

Sequence 1: ATGCT
Sequence 2: AGCT
Consider the following rewards and penalties-
Match :1
Mismatch: -1
Gap: -2
Using the Smith-Waterman algorithm, identify the optimal local alignment score and the corresponding local alignment for the same sequences.

Ans. B 1 .1. Needleman-Wunsch Algorithm



Sequence 1: AATCG
Sequence 2: AACG

Match :1
Mismatch: -1
Gap: -2

		A	A	T	C	G
	0	-2	-4	-6	-8	-10
A	-2	1				
A	-4					
C	-6					
G	-8					

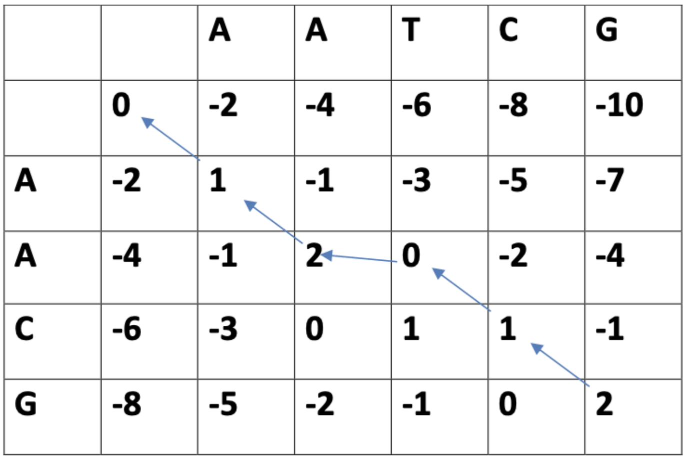
A=A [MATCH]

VALUE FROM LEFT : -4
VALUE FROM UP: 1
VALUE FROM RIGHT:-4

SIMILARLY WE WILL FILL COMPLETE TABLE.

		A	A	T	C	G
	0	-2	-4	-6	-8	-10
A	-2	1	-1	-3	-5	-7
A	-4	-1	2	0	-2	-4
C	-6	-3	0	1	1	-1
G	-8	-5	-2	-1	0	2

WE WILL DRAW THE CORRESPONDING SLINMENT FOR THESE SEQUENCES.



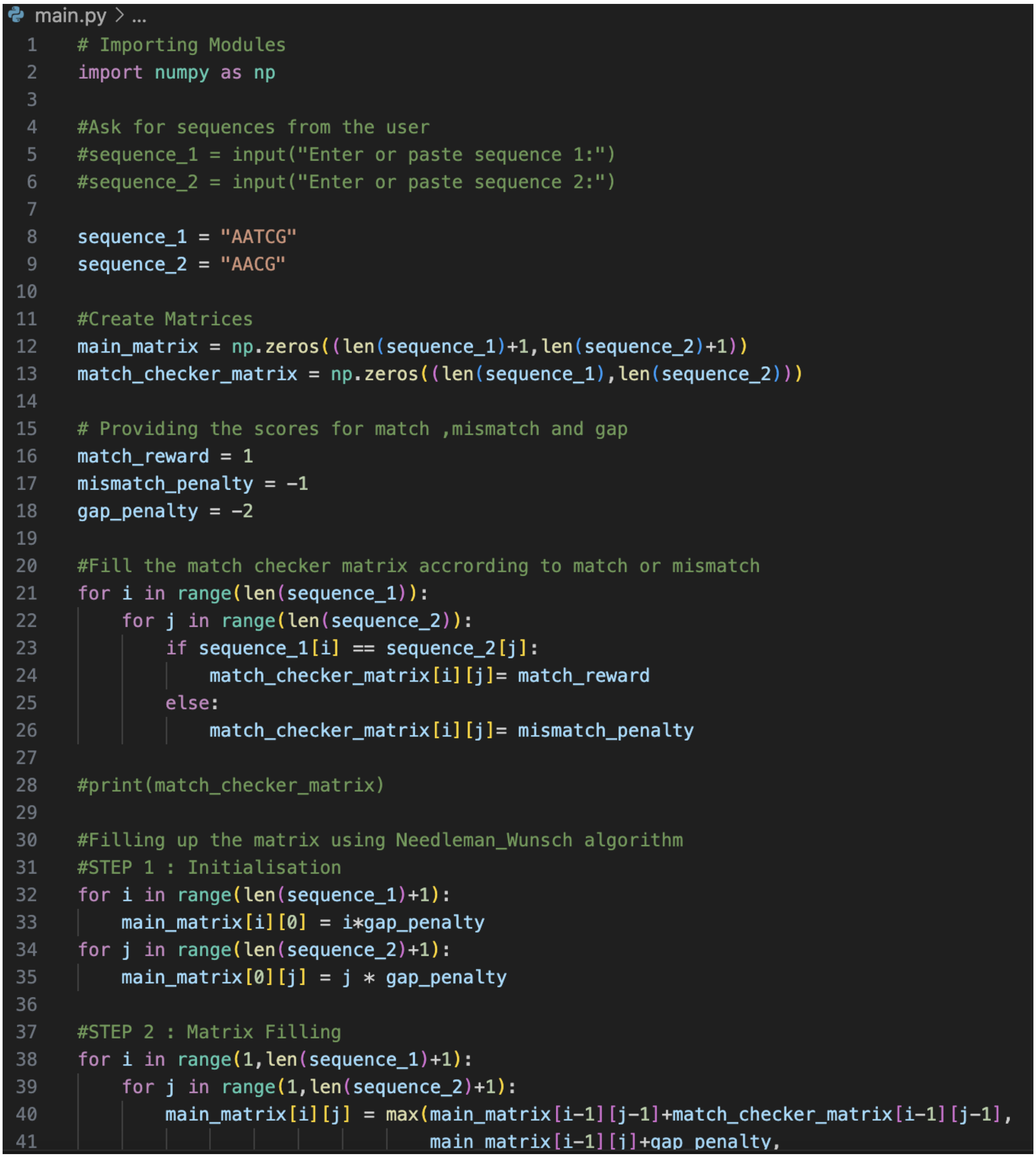
		A	A	T	C	G
	0	-2	-4	-6	-8	-10
A	-2	1	-1	-3	-5	-7
A	-4	-1	2	0	-2	-4
C	-6	-3	0	1	1	-1
G	-8	-5	-2	-1	0	2

SEQUENCE 1 : **AATCG**
SEQUENCE 2: **AA - CG**

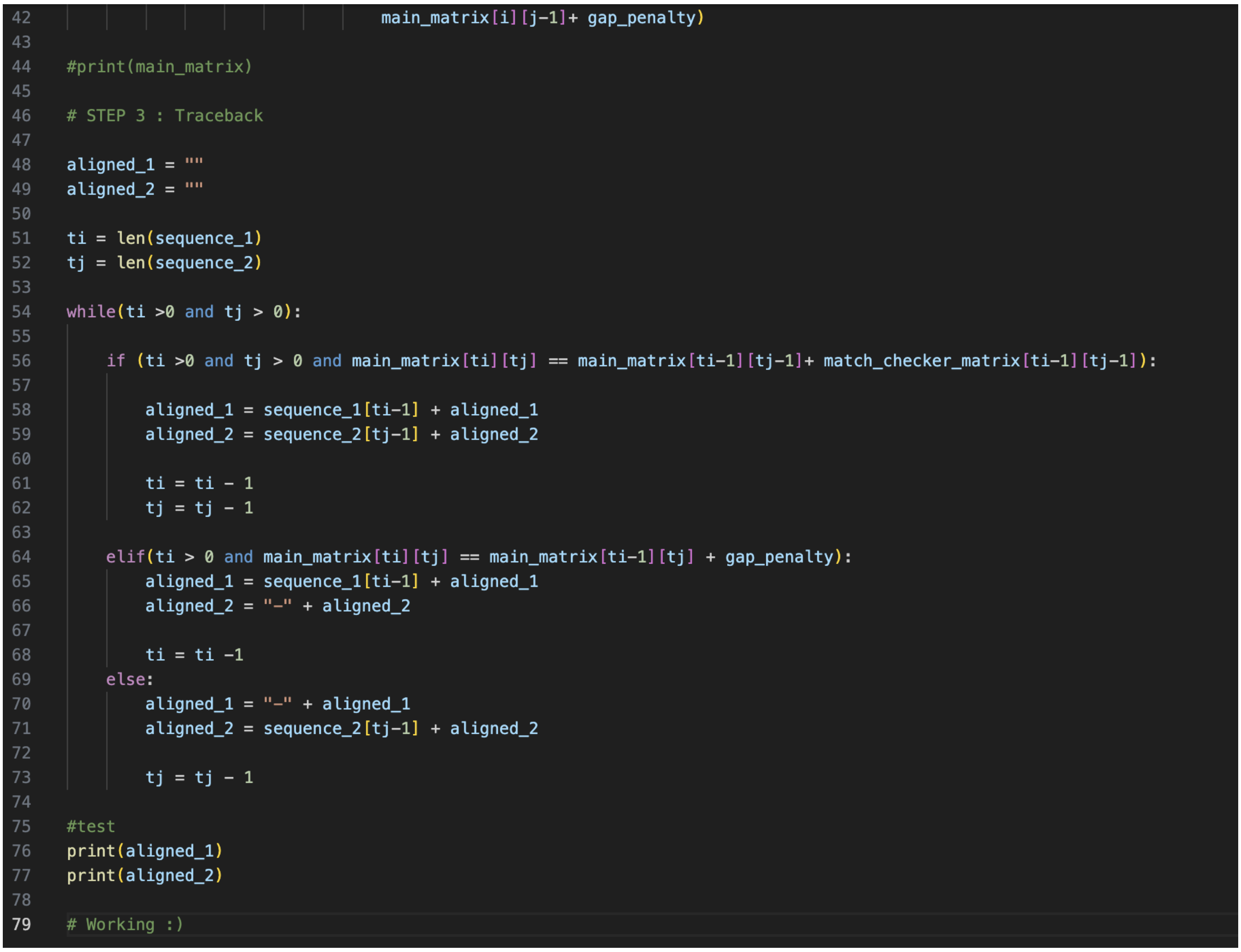
Additionally, Document the following:

B1.2: Write the Code and its relevant output for the above algorithm using biopython code.

Ans. **PYTHON CODE:**

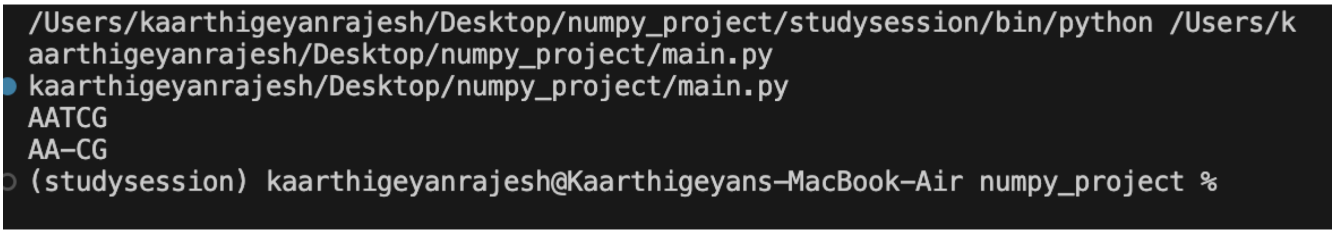


```python
# Importing Modules
import numpy as np

#Ask for sequences from the user
#sequence_1 = input("Enter or paste sequence 1:")
#sequence_2 = input("Enter or paste sequence 2:")

sequence_1 = "AATCG"
sequence_2 = "AACG"

#Create Matrices
main_matrix = np.zeros((len(sequence_1)+1,len(sequence_2)+1))
match_checker_matrix = np.zeros((len(sequence_1),len(sequence_2)))

# Providing the scores for match ,mismatch and gap
match_reward = 1
mismatch_penalty = -1
gap_penalty = -2

#Fill the match checker matrix accrording to match or mismatch
for i in range(len(sequence_1)):
    for j in range(len(sequence_2)):
        if sequence_1[i] == sequence_2[j]:
            match_checker_matrix[i][j]= match_reward
        else:
            match_checker_matrix[i][j]= mismatch_penalty

#print(match_checker_matrix)

#Filling up the matrix using Needleman_Wunsch algorithm
#STEP 1 : Initialisation
for i in range(len(sequence_1)+1):
    main_matrix[i][0] = i*gap_penalty
for j in range(len(sequence_2)+1):
    main_matrix[0][j] = j * gap_penalty

#STEP 2 : Matrix Filling
for i in range(1,len(sequence_1)+1):
    for j in range(1,len(sequence_2)+1):
        main_matrix[i][j] = max(main_matrix[i-1][j-1]+match_checker_matrix[i-1][j-1],
                                main_matrix[i-1][j]+gap_penalty,
```



```python
                                main_matrix[i][j-1]+ gap_penalty)

#print(main_matrix)

# STEP 3 : Traceback

aligned_1 = ""
aligned_2 = ""

ti = len(sequence_1)
tj = len(sequence_2)

while(ti >0 and tj > 0):

    if (ti >0 and tj > 0 and main_matrix[ti][tj] == main_matrix[ti-1][tj-1]+ match_checker_matrix[ti-1][tj-1]):

        aligned_1 = sequence_1[ti-1] + aligned_1
        aligned_2 = sequence_2[tj-1] + aligned_2

        ti = ti - 1
        tj = tj - 1

    elif(ti > 0 and main_matrix[ti][tj] == main_matrix[ti-1][tj] + gap_penalty):
        aligned_1 = sequence_1[ti-1] + aligned_1
        aligned_2 = "-" + aligned_2

        ti = ti -1
    else:
        aligned_1 = "-" + aligned_1
        aligned_2 = sequence_2[tj-1] + aligned_2

        tj = tj - 1

#test
print(aligned_1)
print(aligned_2)

# Working :)
```

OUTPUT:



```
/Users/kaarthigeyanrajesh/Desktop/numpy_project/studysession/bin/python /Users/k
aarthigeyanrajesh/Desktop/numpy_project/main.py
kaarthigeyanrajesh/Desktop/numpy_project/main.py
AATCG
AA-CG
(studysession) kaarthigeyanrajesh@Kaarthigeyans-MacBook-Air numpy_project %
```