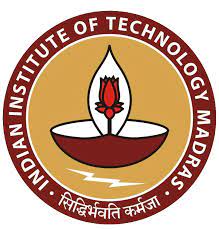
..

**BT 3040: Bioinformatics**

**Assignment 10**



Indian Institute of Technology Madras

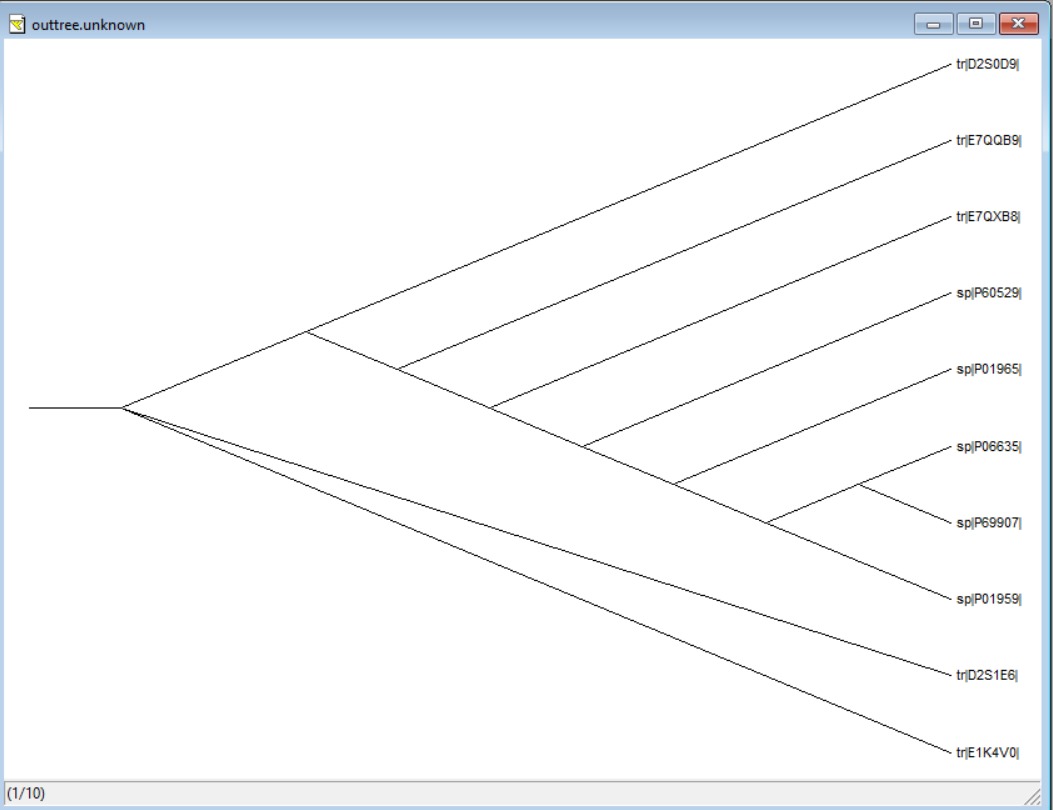
Atharva Mandar Phatak | BE21B009

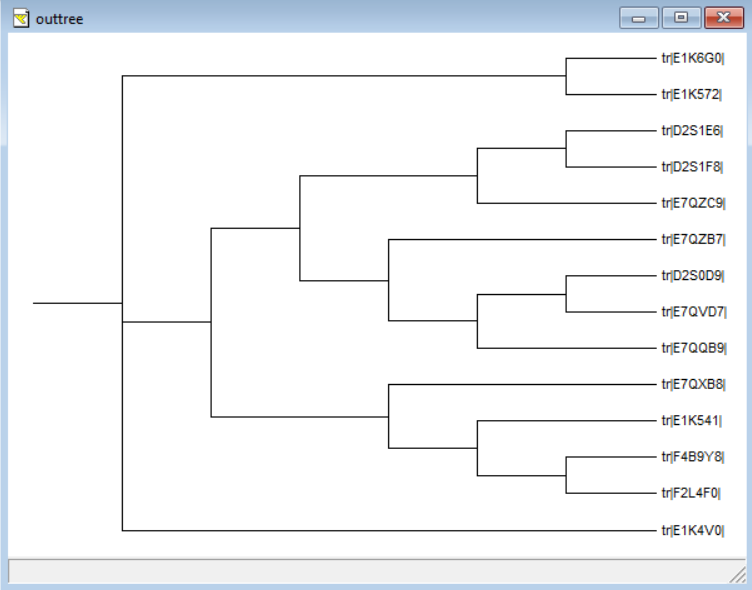
Department of Biotechnology

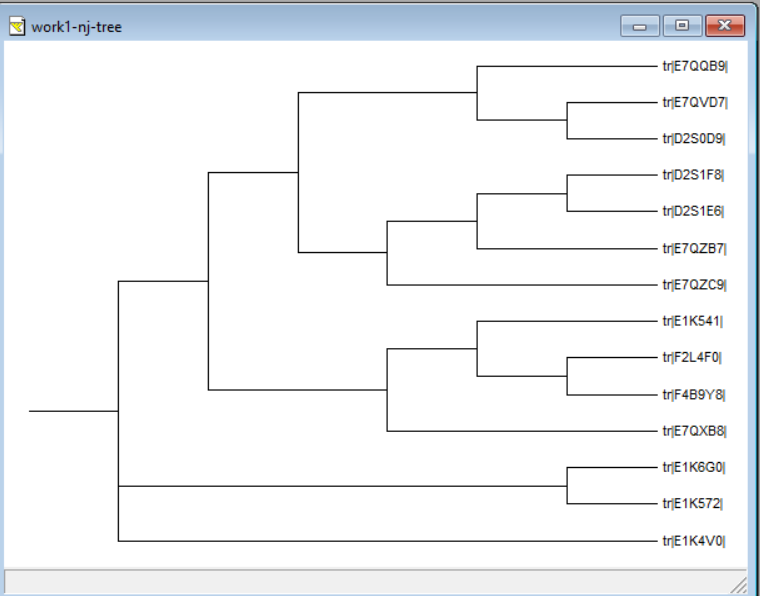
## **Q1**) **Obtain the consensus phylogenetic tree for the following two sets of sequences:**

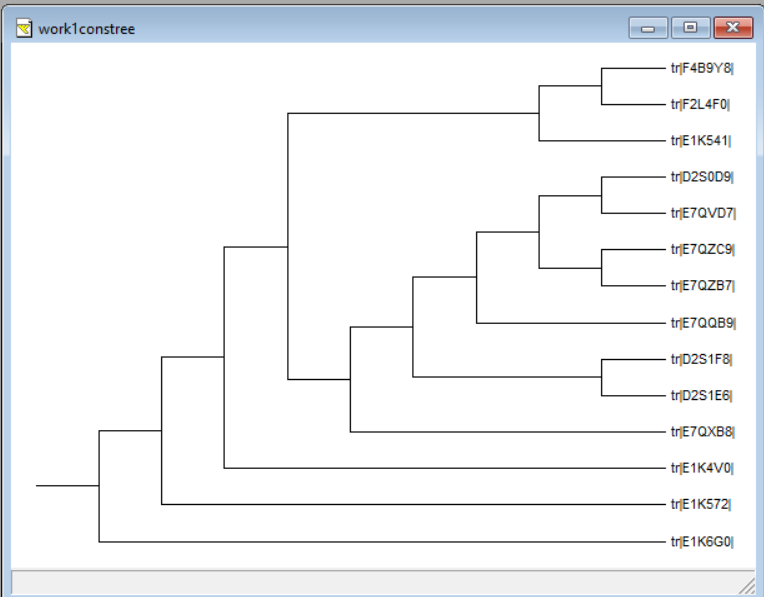
## **Set 1: tim.dat**

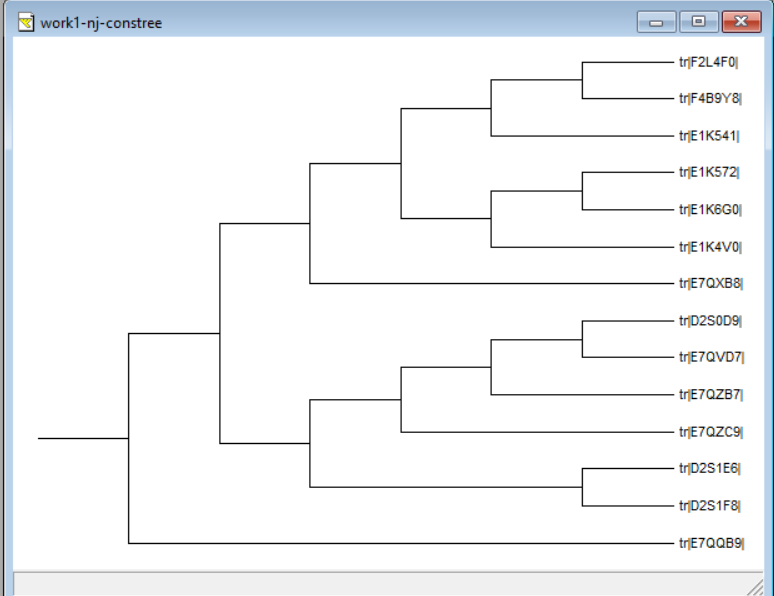
*All the files in ‘tim\_dat’ folder*





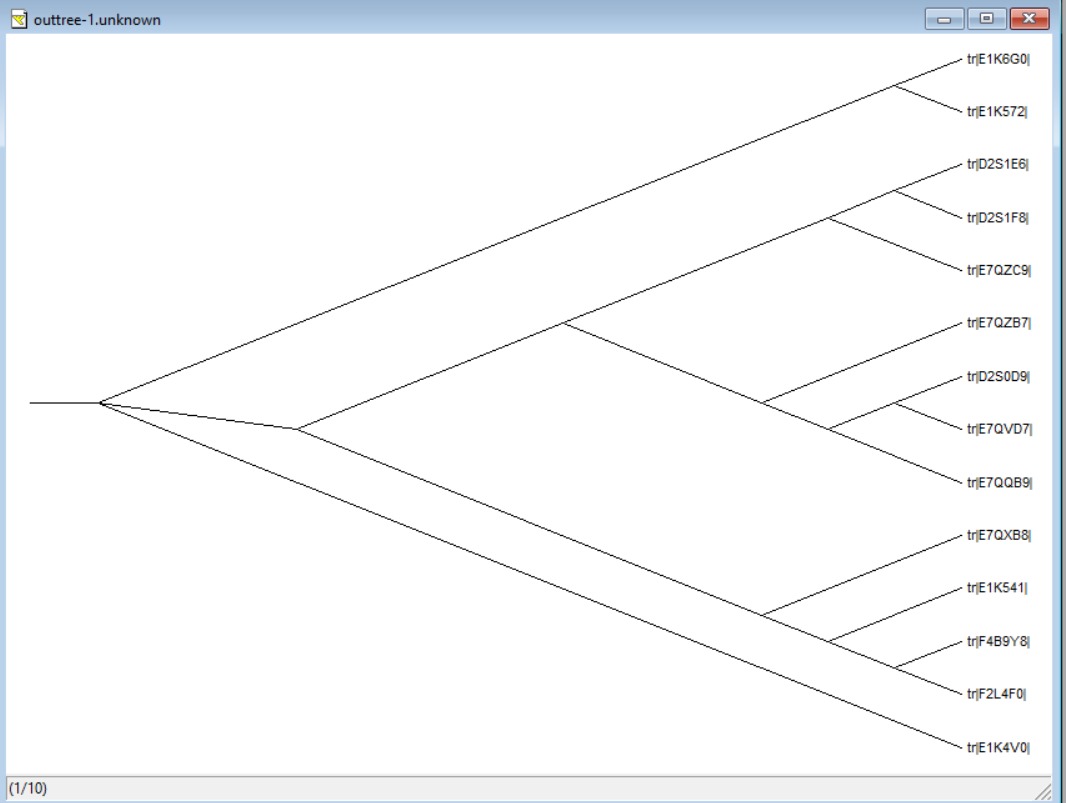


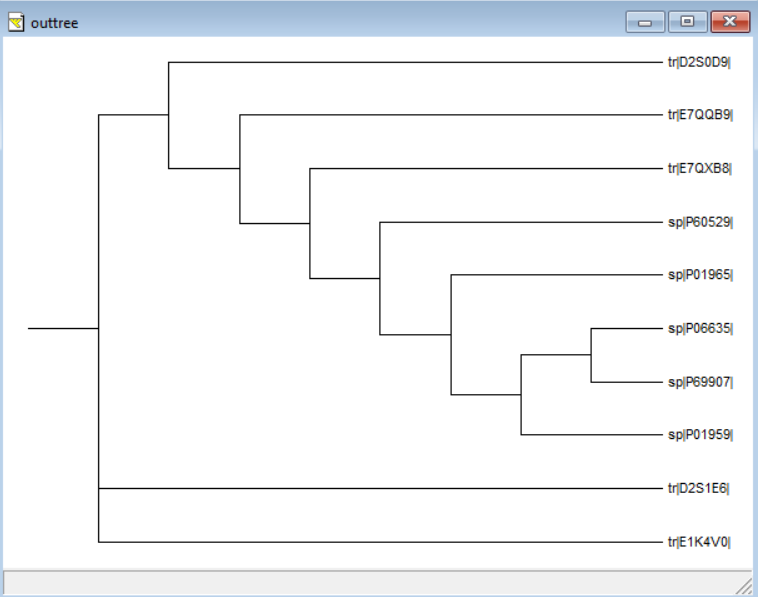


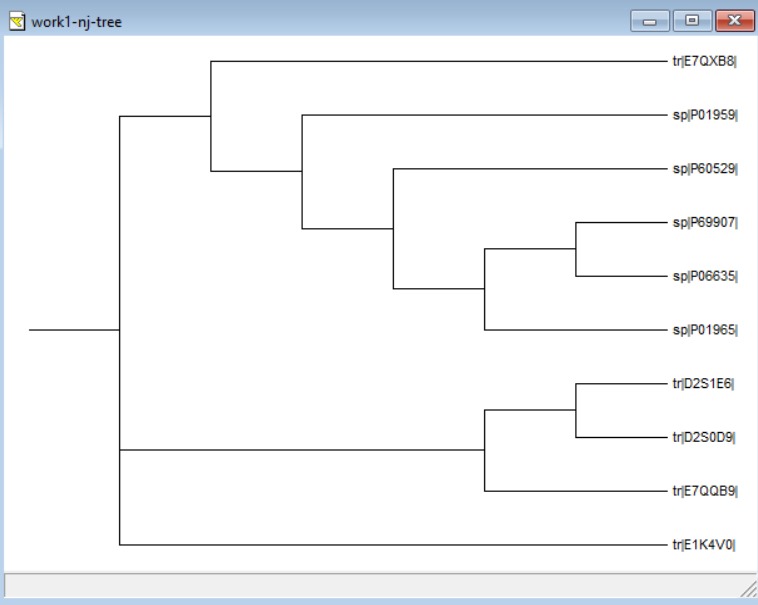


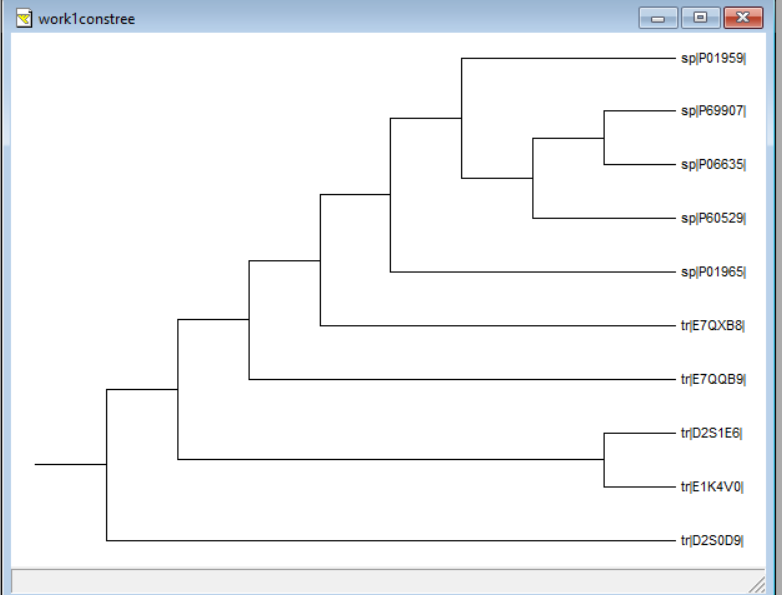
## **Set 2: tim-hemo.dat**

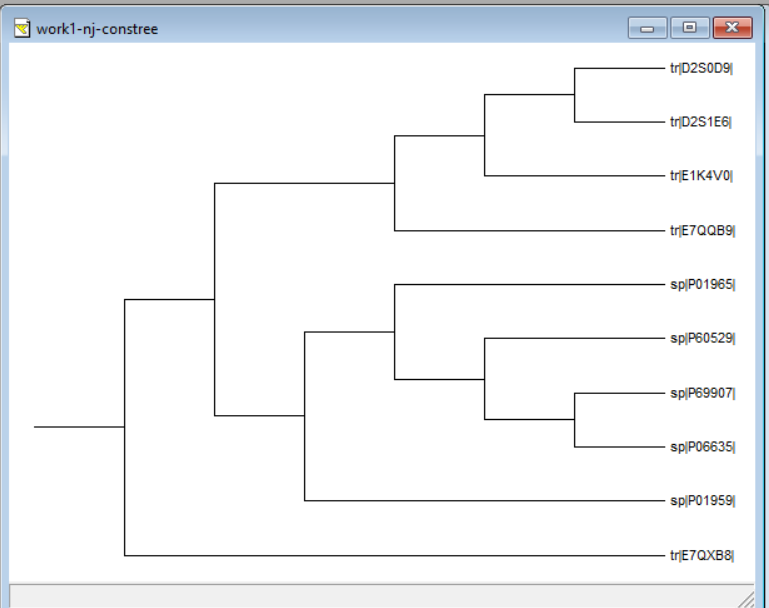
*All the files in ‘tim\_dat\_hemo’ folder*











## **Q2) Obtain the weight matrix for the following sequences**

## **MVLSPADKTNVKGKVGAHAGEYGAAAW**

## **MKRLPADPPCVKGKVKAKAGDYGATTW**

## **MALSAADKTNVKSKVGGHAGEYGAATS**

## **MVLSAADKTNVKSKAGGNAGEWWAAAW**

## **MVLSAADKTNVKSKVLANAGEFGAAAW**

## **ALLPIRTTYHKKCASGHIPEEKDLNNV**

**DEASSLKGHHIKKLEADALLIPLSASS**

Code:

#BT3040| Assignemnt 10 | Q2 | Atharva Mandar Phatak | BE21B009 |

import math

def sequence\_to\_matrix(input\_sequence, sequence\_length):

    amino\_acids = ['A', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'K', 'L', 'M', 'N', 'P', 'Q', 'R', 'S', 'T', 'V', 'W', 'Y']

    matrix = []

    for i in range(0, len(input\_sequence), sequence\_length):

        subsequence = input\_sequence[i:i+sequence\_length]

        row = [0] \* len(subsequence)

        for j, aa in enumerate(subsequence):

            row[j] = aa

        matrix.append(row)

    return matrix

input\_sequence = 'MVLSPADKTNVKGKVGAHAGEYGAAAWMKRLPADPPCVKGKVKAKAGDYGATTWMALSAADKTNVKSKVGGHAGEYGAATSMVLSAADKTNVKSKAGGNAGEWWAAAWMVLSAADKTNVKSKVLANAGEFGAAAWALLPIRTTYHKKCASGHIPEEKDLNNVDEASSLKGHHIKKLEADALLIPLSASS'

sequence\_length = 27

amino\_acid\_matrix = sequence\_to\_matrix(input\_sequence, sequence\_length)

def count\_amino\_acids(matrix):

    amino\_acids = ['A', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'K', 'L', 'M', 'N', 'P', 'Q', 'R', 'S', 'T', 'V', 'W', 'Y']

    amino\_acid\_counts = [[0] \* len(matrix[0]) for \_ in range(len(amino\_acids))]

    for row in matrix:

        for i, amino\_acid in enumerate(row):

            amino\_acid\_counts[amino\_acids.index(amino\_acid)][i] += 1

    return amino\_acid\_counts

def calculate\_weight\_matrix(amino\_acid\_counts):

    num\_rows = len(amino\_acid\_counts)

    num\_cols = len(amino\_acid\_counts[0])

    weight\_matrix = [[0] \* num\_cols for \_ in range(num\_rows)]

    for i in range(num\_rows):

        for j in range(num\_cols):

            n = amino\_acid\_counts[i][j]

            weight\_matrix[i][j] = round(math.log((n + 1/20) / (8/20)),2)

    return weight\_matrix

# Count amino acids

amino\_acid\_counts = count\_amino\_acids(amino\_acid\_matrix)

# Calculate weight matrix

weight\_matrix = calculate\_weight\_matrix(amino\_acid\_counts)

from tabulate import  tabulate

print("Alignment Matrix")

print(tabulate(amino\_acid\_counts, headers=["{}".format(i+1) for i in range(len(amino\_acid\_matrix[0]))], tablefmt="grid",  numalign="centre"))

print("")

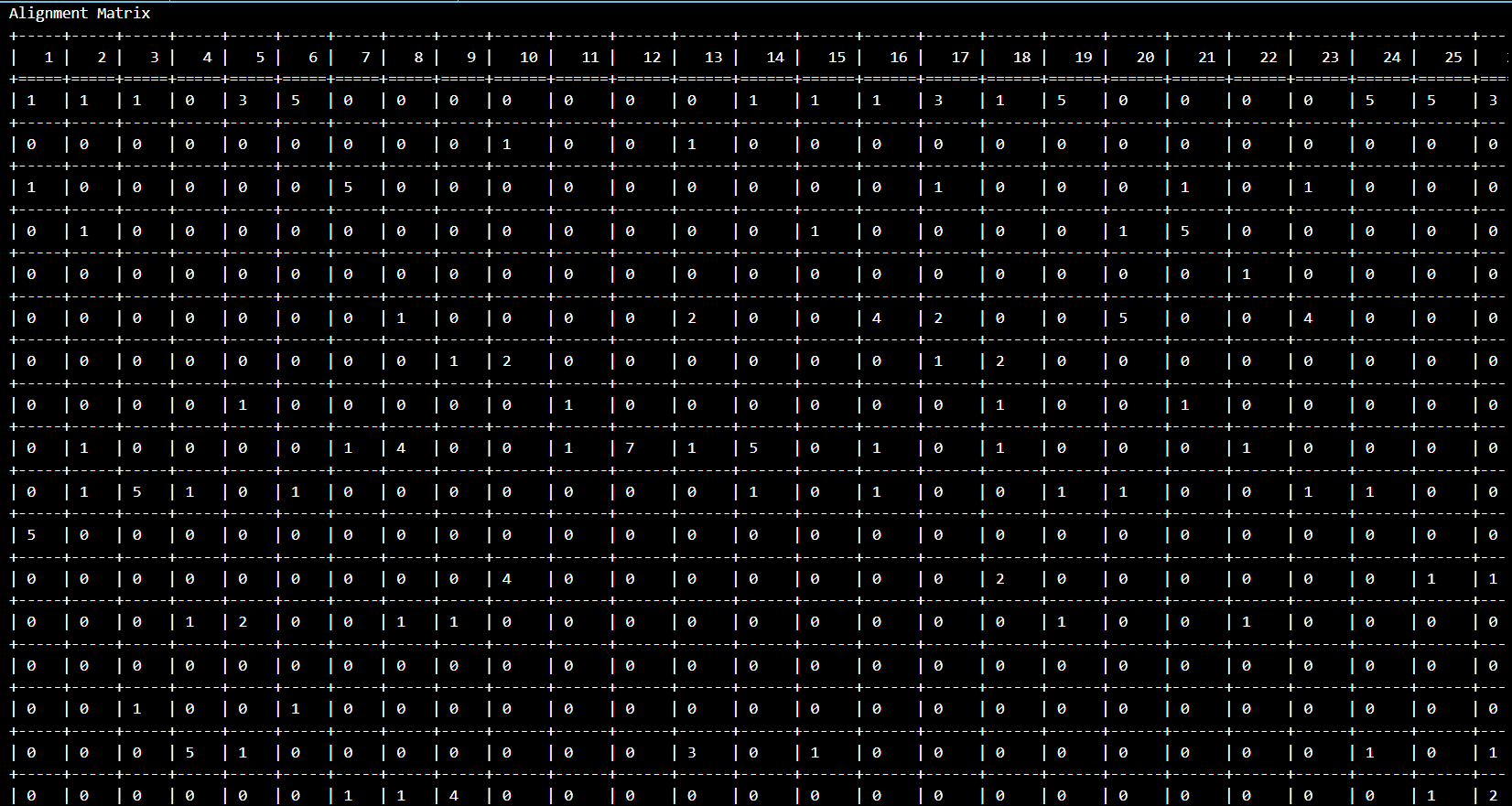
print("")

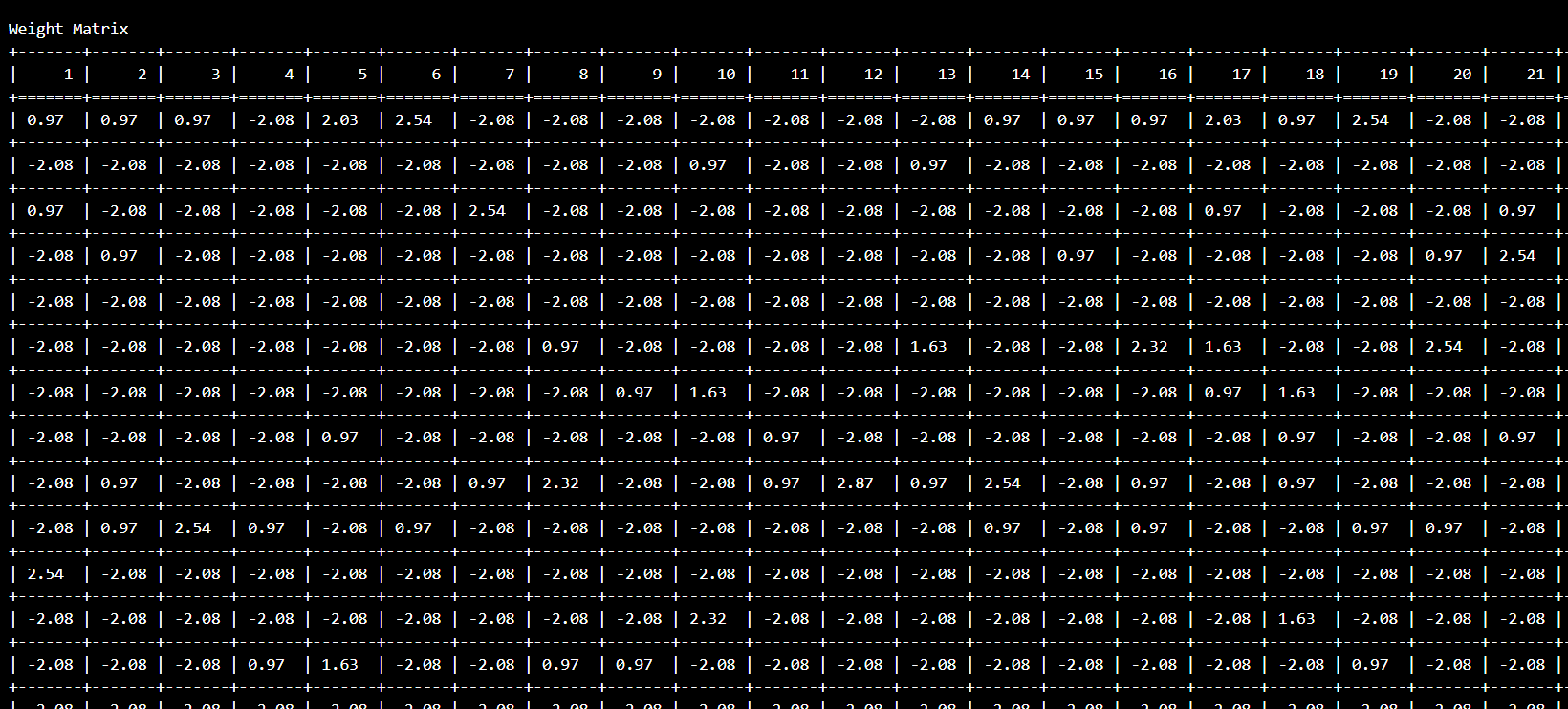
print("Weight Matrix")

print(tabulate(weight\_matrix, headers=["{}".format(i+1) for i in range(len(weight\_matrix[0]))], tablefmt="grid",  numalign="centre"))

Output:

1. Alignment Matrix



1. Weight Matrix

*Complete output in ‘Q2\_Output.txt’*