**Report on Finding a Longest Common Substring in DNA Sequences**

**Problem**

In this task, we are given several DNA sequences, each up to 1000 base pairs long. The goal is to find the longest sequence (substring) that appears in each DNA string. This is known as the "Longest Common Substring" problem. There could be multiple solutions if more than one substring has the same maximum length; in that case, returning any one of the longest common substrings is acceptable.

The solution also needs to meet the following requirements:

1. **Number of Sequences** (k): A maximum of 100 DNA sequences.
2. **Sequence Length**: Each DNA sequence must be no longer than 1000 base pairs.

**Code**

Below is the Python code that solves this problem using BioPython and Google Colab’s file upload capability for simplicity.

python

Copy code

from Bio import SeqIO

from google.colab import files

# Function to find the longest common substring in a list of DNA strings

def find\_longest\_common\_substring(dna\_strings):

if not dna\_strings:

print("No DNA sequences found.")

return ""

base\_string = dna\_strings[0]

max\_length = len(base\_string)

# Searching for the longest common substring

for length in range(max\_length, 0, -1):

for start in range(max\_length - length + 1):

substring = base\_string[start:start + length]

if all(substring in seq for seq in dna\_strings[1:]):

return substring # Return the first found longest common substring

return "none"

# Function to prompt user for input method (manual or file upload)

def get\_dna\_sequences():

choice = input("CHOOSE \n(1) Do you want to input sequences manually or \n(2) upload a FASTA file?\nEnter 1 or 2: ")

if choice == '1':

# Manual input with conditions for k ≤ 100 and length ≤ 1000

num\_sequences = int(input("Enter the number of DNA sequences (up to 100): "))

if num\_sequences > 100:

print("Error: Maximum number of sequences is 100.")

return []

sequences = []

for i in range(num\_sequences):

seq = input(f"Enter DNA sequence {i + 1} (up to 1000 bp): ").strip().upper()

if len(seq) > 1000:

print("Error: Sequence length exceeds 1000 base pairs.")

return []

sequences.append(seq)

return sequences

elif choice == '2':

# File upload

print("Please upload a FASTA file.")

uploaded = files.upload() # Open the file upload dialog in Colab

uploaded\_file = list(uploaded.keys())[0] # Get the uploaded file name

# Reading DNA sequences from the uploaded FASTA file

sequences = [str(record.seq) for record in SeqIO.parse(uploaded\_file, "fasta")]

# Check conditions for k ≤ 100 and length ≤ 1000

if len(sequences) > 100:

print("Error: FASTA file contains more than 100 sequences.")

return []

if any(len(seq) > 1000 for seq in sequences):

print("Error: One or more sequences in the file exceed 1000 base pairs.")

return []

return sequences

else:

print("Invalid choice. Please enter 1 for manual input or 2 for file upload.")

return get\_dna\_sequences()

# Main code execution

sequences = get\_dna\_sequences() # Get sequences from the user

if not sequences:

print("No sequences found. Please check the input.")

else:

# Finding and displaying the longest common substring

result = find\_longest\_common\_substring(sequences)

print("Longest common substring:", result)

**Explanation of Implementation**

The code was developed to tackle the problem by dividing the solution into three main parts:

1. **Function to Find the Longest Common Substring**:
   * This function find\_longest\_common\_substring receives a list of DNA strings as input.
   * The function starts by taking the first DNA sequence in the list (base\_string) and then iterates over all possible substrings, starting with the longest and decreasing in length.
   * For each substring, it checks if it appears in every other DNA sequence in the list.
   * When a substring is found in all sequences, it is immediately returned as the longest common substring, since we began by searching with the longest possible substrings.
2. **User Input Function**:
   * The get\_dna\_sequences function allows the user to input DNA sequences either manually or by uploading a FASTA file.
   * For manual input, the user is prompted to enter each sequence, with checks ensuring that:
     + The number of sequences does not exceed 100.
     + Each sequence does not exceed 1000 base pairs.
   * For file input, the user uploads a FASTA file. The program then reads the file using BioPython’s SeqIO library and applies the same checks for the number of sequences and their lengths.
3. **Main Program Execution**:
   * First, the program calls get\_dna\_sequences() to get the list of DNA sequences based on the chosen input method.
   * If sequences are retrieved, find\_longest\_common\_substring() is called to compute the longest common substring, which is then printed as the result.
   * If no valid sequences are found, the program outputs an appropriate message.