BMS COLLEGE OF ENGINEERING

(Autonomous College under VTU)

Bull Temple Road, Basavanagudi, Bangalore – 560019

A project report on

"Evolutionary Closeness Analysis"

Submitted in partial fulfilment of the requirements for the award of degree

BACHELOR OF ENGINEERING IN
INFORMATION SCIENCE AND ENGINEERING

By

Gunanka D	1BM22IS081
K L Gireesh	1BM22IS094
Karan Nagraj Kulkarni	1BM22IS097
Rishi Raaj Verma	1BM22IS260

Department of Information Science and Engineering
2025-26

2025-26

BMS COLLEGE OF ENGINEERING

(Autonomous College under VTU)

Bull Temple Road, Basavanagudi, Bangalore – 560019

Department of Information Science and Engineering

CERTIFICATE

This is to certify that the project entitled “Evolutionary Closeness Analysis” is a bona-fide work carried out by Gunanka D – 1BM22IS081, K L Gireesh – 1BM22IS094, Karan Nagraj Kulkarni – 1BM22IS097 & Rishi Raaj Verma – 1BM22IS260 in partial fulfilment for the award of degree of Bachelor of Engineering in Information Science and Engineering from Visvesvaraya Technological University, Belgaum during the year 2025- 2026. It is certified that all corrections/suggestions indicated for Internal Assessments have been incorporated in the report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the Bachelor of Engineering Degree.

Signature of the Faculty

Harini S.

Assistant Professor

Table of Contents

Introduction

The study of evolutionary relationships between species, also known as phylogenetics, plays a crucial role in understanding biodiversity, species origins, and genetic divergence. One of the most effective approaches in modern biology is to compare DNA sequences to estimate how closely related different species are. With the rapid advancement in bioinformatics tools and genetic databases, researchers can now analyse genetic similarities at the molecular level using computational methods.

This project focuses on determining the evolutionary closeness between **two species** by analysing their DNA sequences—specifically, the **Cytochrome c oxidase subunit 1 (COX1)** gene. COX1 is a mitochondrial gene that is widely used in species identification and phylogenetic studies due to its **moderate rate of evolution** and presence across most animal species. It serves as a reliable genetic marker for comparing species that diverged **within the last 100 million years**.

Using **Biopython**, a powerful library for biological computation, the program fetches COX1 sequences from the NCBI database, aligns them using advanced algorithms like **MUSCLE**, and calculates the percentage of genetic similarity. This provides an estimate of how evolutionarily close two species are based on their DNA data.

Such computational approaches not only enhance our understanding of life's diversity but also offer a scalable way to perform evolutionary analysis without requiring lab-based genetic testing. This project demonstrates how bioinformatics can be applied to explore genetic relationships in a fast, accessible, and accurate manner.

Deep Dive on topic

Understanding the evolutionary relationships between different species has been a foundational objective of biological sciences for centuries. Traditionally, **evolutionary trees (phylogenies)** were constructed using morphological traits such as skeletal structures, behavioural patterns, or ecological roles. However, with the advent of **molecular biology** and **genomic sequencing** technologies, it is now possible to compare organisms at the genetic level, offering a far more accurate and objective means of determining evolutionary closeness. The branch of science that deals with understanding these relationships through genetic material is called **molecular phylogenetics**.

This project focuses on developing a Python-based tool to compute the **evolutionary similarity between two given species** using their **DNA sequences**. Specifically, it centres around the analysis of the **Cytochrome c oxidase subunit I (COX1)** gene—a widely accepted marker in molecular taxonomy and phylogenetic research. The COX1 gene is located in the mitochondrial genome and encodes a crucial enzyme involved in the respiratory electron transport chain. Due to its evolutionary stability and moderate mutation rate, COX1 is commonly used in **DNA barcoding**, species identification, and evolutionary studies, especially for animal taxa.

Why COX-1?

Chain of research that led to choosing **COX1** as our target gene.

First of all, we need to choose a target gene known as **markers**. If you randomly fetch any sequence from NCBI, it might be garbage (like viral DNA, synthetic stuff, etc.). You must fetch **orthologous genes** — same gene across species (e.g., cytochrome c, COX1, rRNA genes). Full genome comparison is ideal but impossible manually — that's why scientists use markers.

Common Genetic Markers

Most phylogenetic analyses focus on genes that are universally present and evolve at appropriate rates. **Highly conserved genes** give reliable signals across distant groups, while faster-evolving regions help with close relatives. Common choices include:

- **Ribosomal RNA genes:** The 16S rRNA gene (in bacteria/archaea) and 18S (in eukaryotes) are classic universal markers . They change slowly and are easy to align across broad taxonomic groups.

- **Mitochondrial DNA:** In animals, mitochondrial genes (like **COI**, cytochrome *b*, 12S/16S rRNA) are popular because they mutate relatively quickly and are abundant in cells. (Typical animal mtDNA carries 16S and 12S rRNAs plus protein genes such as **COI**.) Plant phylogenies often use chloroplast genes (e.g. *rbcL*, *matK*).

- **Conserved protein-coding genes:** Housekeeping genes (like RNA polymerase subunits, translation factors, tubulins, etc.) are also used. For deep analyses, researchers may select dozens or hundreds of single-copies orthologs that exist in all species of interest.

- **Nuclear ribosomal spacers:** In fungi and plants, the internal transcribed spacer (ITS) regions between rRNA genes evolve rapidly and serve as species-level barcodes (similar to COI in animals).

By combining markers (multi-gene or genome-wide alignments), one can improve accuracy. For example, a “super matrix” of many genes or concatenated alignments is common in modern studies

So, we got a list of markers.

Gene Name	Purpose / Why it’s used	Notes
COX1 (Cytochrome c oxidase subunit 1)	Mitochondrial gene, evolves **moderately**	Used in *DNA barcoding*
16S rRNA (for prokaryotes)	Ribosomal RNA gene, slow-evolving	Only for bacteria/archaea
18S rRNA (for eukaryotes)	Ribosomal RNA gene, slow-evolving	Good for all animals
Cytochrome b (cytb)	Mitochondrial, useful for mammals, birds	Very classic
Beta-globin gene	Blood oxygen transport, moderate evolution	Nice for mammals
Histone H3	Super-conserved across species	Used for deep phylogeny

How does the speed of evolution of the genes help?

The **speed of evolution** of a gene matters because:

- **Slow-evolving genes** (like 18S rRNA, Histone H3):
  - Mutate *very little* over millions of years.
  - Help you **compare distant species** (like humans' vs fish vs plants).
  - Because changes are rare → differences show *real ancient splits.*
- **Fast-evolving genes** (like Cytochrome b, mitochondrial DNA):
  - Mutate *faster*, more changes in shorter time.
  - Help you **compare close relatives** (like humans' vs chimps, different bird species).
  - Because you need *small recent differences* to detect.

What is the best Temporal range for COX-1?

COX1 (Cytochrome c oxidase subunit 1) is a **moderately evolving** mitochondrial gene.
It works **very well** for species that split **up to ~100 million years ago (MYA)**.
(especially animals like mammals, birds, reptiles, fishes)

Rough idea:

- Human vs Chimpanzee (~6 MYA) ➔ COX1 works great.
- Human vs Macaque (~25 MYA) ➔ Still good.
- Mammal vs Lizard (~300 MYA) ➔ COX1 starts struggling (too divergent).
- Mammal vs Fish (~400 MYA) ➔ Not reliable anymore for fine closeness.

Simple rule:
COX1 is best for "within ~100 MYA" comparisons — recent and medium-distant species.
For **very ancient** stuff (plants vs animals, fish vs mammals), you need **18S rRNA** or **Histone H3** (slow-evolving).

Continuation with Deep Dive

The core goal of this project is to create a user-friendly, automated pipeline that allows the comparison of **two species of interest** and outputs their **genetic closeness as a similarity percentage**. Instead of requiring a locally stored DNA database, the tool utilizes **NCBI's nucleotide database** via the Entrez API (offered by **Biopython**) to fetch up-to-date COX1 sequences for any valid scientific species name provided. This ensures a high level of flexibility and real-time sequence access without manual intervention or dataset maintenance.

Once the sequences are fetched, the program proceeds to align them using the **MUSCLE (Multiple Sequence Comparison by Log-Expectation)** algorithm. MUSCLE is one of the most trusted multiple sequence alignment tools in bioinformatics, known for its speed and accuracy. It is widely used in genetic research for both small- and large-scale sequence alignments. This alignment step is crucial, as comparing raw sequences without accounting for insertions, deletions, or mismatches can lead to misleading similarity scores. Proper alignment ensures that homologous bases are compared, and evolutionary events such as mutations are correctly reflected in the final similarity percentage.

After alignment, the similarity is computed by analysing the aligned sequences and calculating the number of matching base pairs over the total alignment length. This similarity percentage serves as a proxy for **evolutionary closeness—**a higher percentage indicates more recent divergence from a common ancestor, whereas lower percentages reflect more ancient divergence and greater evolutionary distance.

The choice of COX1 as the genetic marker is backed by its broad usage in DNA barcoding studies. COX1 sequences are widely available across most known animal species and are known to strike a balance between conservation and variability. That means they are stable enough to be found in almost all animals, but they also mutate just enough over time to be informative when comparing species. This makes COX1 ideal for inferring evolutionary relationships on the scale of **millions to tens of millions of years**, covering the divergence times of most animals.

This project is built using **Biopython**, a comprehensive open-source library that provides tools for reading, writing, and analysing biological data. Biopython's integration with NCBI databases, sequence alignment modules, and I/O tools makes it a perfect fit for developing a pipeline like this. The tool also uses standard command-line integration to run MUSCLE externally, ensuring compatibility with professionally accepted sequence alignment workflows.

One of the highlights of this project is its **modular structure**. Each major step—fetching sequences, saving them, aligning them, and computing similarity—is encapsulated in separate functions. This not only improves readability and maintainability but also allows for easier upgrades in the future. For example, if a user wishes to switch from MUSCLE to another alignment algorithm like MAFFT or Crustal Omega, they can do so with minimal code changes.

Moreover, the design is scalable. Although the current scope involves comparing two species at a time, the architecture can be easily extended to support multiple species and build phylogenetic trees using Biopython's Phylo module. Such extensions could make the tool suitable for classroom education, research exploration, or even inclusion in larger bioinformatics pipelines.

In conclusion, this project brings together computational biology and evolutionary theory into a single, interactive tool that allows users to explore genetic relationships in real time. By using publicly available genetic data, modern alignment algorithms, and clean software engineering practices, it demonstrates how bioinformatics can transform the way we study life and its history. Whether the goal is to confirm known relationships (such as the closeness of humans and chimpanzees) or to explore unknown ones, this tool provides an accessible and scientifically sound method for comparing the evolutionary similarity of species.

Literature Survey

1. **Standardized Phylogenetic and Molecular Evolutionary Analysis Applied to Species Across the Microbial Tree of Life**
Shakya, M., Ahmed, S. A., Davenport, K. W., Flynn, M. C., Lo, C.-C., & Chain, P. S. G.
This study emphasizes the use of genome-wide SNPs as robust markers for phylogenetic diversity, highlighting their utility in resolving both short and long branches in phylogenetic trees.

2. **General Properties and Phylogenetic Utilities of Nuclear Ribosomal DNA and Mitochondrial DNA Commonly Used in Molecular Systematics**
Hillis, D. M., & Dixon, M. T.
This review discusses the advantages and limitations of using nuclear rDNA and mitochondrial DNA in phylogenetic studies, providing insights into marker selection for evolutionary analyses.

3. **Common Methods for Phylogenetic Tree Construction and its Implementation in R**
Li, H., & Durbin, R.
This article outlines various methodologies for constructing phylogenetic trees, including distance-based and character-based methods, and discusses their implementation using R programming.

4. **Jukes-Cantor Distance**
MEGA Software Documentation
This resource explains the Jukes-Cantor model, a foundational method for estimating evolutionary distances by correcting for multiple substitutions at the same site.

5. **Models of DNA Evolution**
Wikipedia Contributors
This page provides an overview of various DNA substitution models, including JC69, K80, F81, HKY85, and TN93, detailing their assumptions and applications in phylogenetic analyses.

6. **Phylogenetic Convolutional Neural Networks in Metagenomics**
Asgari, E., Garakani, K., McHardy, A. C., & Mofrad, M. R. K.
This study introduces Ph-CNN, a deep learning architecture that incorporates phylogenetic information into convolutional neural networks for metagenomic data classification.

Article named ‘*Genetic Similarities: Wilson, Sarich, Sibley, and Ahlquist*’, under Understanding Evolution by Berkeley, University of California.

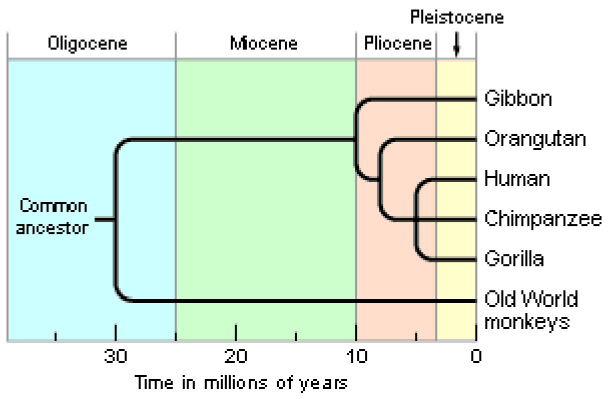
Genetic Similarities: Wilson, Sarich, Sibley, and Ahlquist

To investigate how birds are related to one another, a biologist of the 1950s would have carefully studied their anatomical similarities and differences. But today, a scientist working on the same problem could also use the very instructions from which that anatomy was built: its genetic code.
DNA sequences form the hereditary links between generations, so it is no surprise that scientists investigating evolutionary relationships have sought to get closer and closer to the DNA that underlies those relationships. However, reading the genomes of entire organisms did not fall immediately from the discovery of DNA in the 1950s. In small steps, scientists came closer to their target.

Scientists first began to zoom in on gene sequences by studying the products of DNA: proteins. After all, if two species are closely related, they should have similar gene sequences, which should then make similar proteins. So before the 1970s, proteins were used as stand-ins for genes in studying evolution.

Testing similarity using antibodies

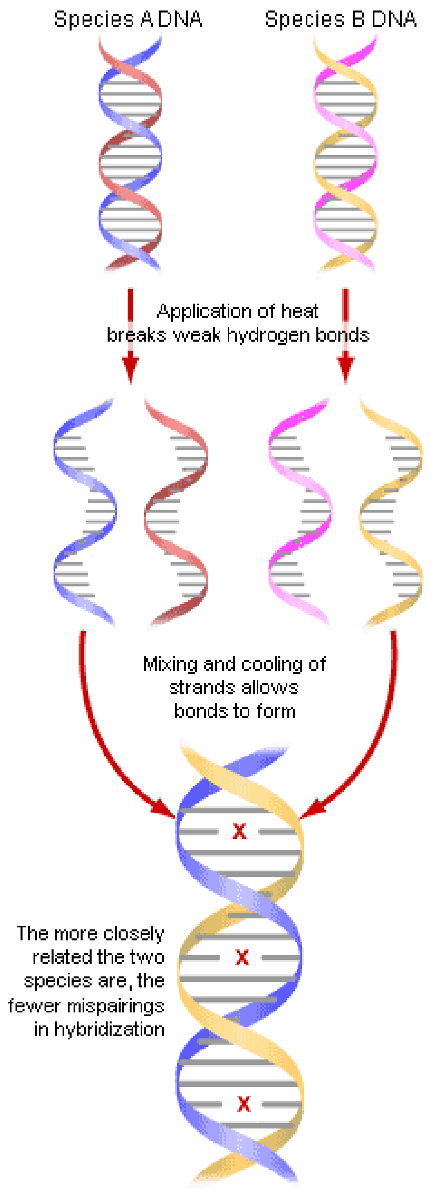
One way that researchers assessed protein similarities was by harnessing the immune system’s ability to recognize foreign proteins. For example, the immune system of a rabbit will recognize a human protein as foreign and will mount an attack against it by making antibodies specific to that protein. If those same rabbit antibodies are exposed to a similar protein — from a chimpanzee, perhaps — they will attack it as well. The more similar the proteins from the two species (human and chimpanzee) are, the stronger this second attack will be.
Although variations of this technique were being employed as early as 1904, more sensitive protocols were developed in the 1960s. These more sensitive techniques revealed the remarkable similarity between the proteins of humans and those of other great apes.
Expanding upon the work of others and making the assumption that fewer protein differences corresponded to shorter times of separation, Vincent Sarich and Allan Wilson estimated that humans, chimpanzees, and gorillas shared a common ancestor only 5 million years ago — a much shorter length of time than was commonly accepted at the time.



Times of divergence and phylogeny of hominoids, as estimated from immunological data.

Testing similarity using DNA

Scientists studying the chemistry of DNA moved even closer to actual sequences. Charles Sibley and Jon Ahlquist pioneered the use of DNA kinetics to investigate evolutionary relationships using a technique called DNA-DNA hybridization (see figure, right). Each DNA molecule is made of two strands of nucleotides. If the strands are heated, they will separate—and as they cool, the attraction of the nucleotides will make them bond back together again. To compare different species, scientists cut the DNA of the species into small segments, separate the strands, and mix the DNA together. When the two species' DNA bonds together, the match between the two strands will not be perfect since there are genetic differences between the species — and the more imperfect the match, the weaker the bond between the two strands. These weak bonds can be broken with just a little heat, while closer matches require more heat to separate the strands again.


Species A DNA
Species B DNA
Application of heat breaks weak hydrogen bonds
Mixing and cooling of strands allows bonds to form
X
X
X
The more closely related the two species are, the fewer mispairings in hybridization

DNA hybridization can measure how similar the DNA of different species is — more similar DNA hybrids "melt" at higher temperatures. When this technique was applied to primate relationships, it suggested that humans and chimpanzees carried DNA more similar to one another's than to orangutans' or gorillas' DNA.

Hypothesized evolutionary relationships between humans and their close relatives based on DNA-DNA hybridization data.

Sequencing DNA

Machines that automatically sequence DNA have made those sequences readily available for evolutionary research. Image courtesy of Sequenom.

The first DNA sequencing methods were invented in the late 1970s, but pure DNA, ready for sequencing, was difficult to produce — thus, making DNA sequencing labor- and time-intensive compared to other tools for making *indirect* inferences about genetic sequences. However, in the late 1980s, scientists developed a technique for producing many, many copies of a very small amount of DNA, and this invention sparked an explosion in the study of DNA

sequences. Researchers began to rely upon sequences as a crucial source of evidence for evolutionary relationships.
Sequencing genes seems to become easier every day. Ten years ago, it might have taken an hour to sequence 10 base pairs. Today a typical lab can sequence 100 base pairs in an hour and facilities with the latest technology sequence hundreds of base pairs each minute. We are now awash in genetic code — we have a basic map of the human genome and the genomes of many other organisms. However, DNA sequences alone do not answer all the questions that biologists ask, and knowing a gene's sequence is still many steps away from understanding how it actually works and what it does. DNA sequences are only one line of evidence illuminating evolutionary relationships. For example, human and chimpanzee DNA is 98% identical, and genetic sequencing can tell us exactly where in the genome those few DNA differences are — but anatomical, behavioural, and developmental studies are also crucial in deeply understanding our differences, similarities, and shared evolutionary history.

High Level Design of Implementation

Project: DNA-Based Evolutionary Closeness Analysis (COX1 Focus)

1. **Import libraries & setup MUSCLE**
   Includes:
   from Bio import Entrez, SeqIO
   import os

   from Bio.Seq import Seq
   from Bio.SeqRecord import SeqRecord
   import subprocess

   To setup muscle, download and install the .exe file inside your virtual environment.
   muscle-win64.v5.3.exe = https://github.com/rcedgar/muscle/releases

2. **Fetch COX1 Sequences from NCBI**
   Function: fetch_cox1_sequence(species_name)
   Query NCBI for the COX1 gene of the given species using Entrez.
   Fetch FASTA sequence.
   Return the sequence.

3. **Save Sequences to a Temporary FASTA File.**
   Function: save_sequences_to_fasta(seq1, seq2, file_path)
   Save both sequences into a FASTA file for MUSCLE to read.

4. **Align Sequences using MUSCLE**
   Function: align_sequences_muscle(input_fasta, output_fasta)
   Call MUSCLE via command line to align sequences.
   Output aligned FASTA.

5. **Compute Similarity**
   Function: compute_similarity(aligned_fasta)
   Read aligned sequences.
   Compare base-by-base.
   Calculate % similarity.

6. **Pipeline function**
   Fetch → Save → Align → Compute → Output similarity %.

7. **Main Program**
   Hande Inputs: Two species names.
   Call pipeline function and return the output similarity

MUSCLE

What is MUSCLE?

MUSCLE stands for **MUltiple Sequence Comparison by Log-Expectation**. It is a high-speed and high-accuracy **multiple sequence alignment (MSA)** tool developed by Robert Edgar in 2004. It is widely used in bioinformatics to align DNA, RNA, or protein sequences from multiple organisms.
Unlike simple pairwise alignment (which compares only 2 sequences at a time), **MUSCLE aligns 3 or more sequences simultaneously** in a way that attempts to maintain their biological relevance and evolutionary relationships.

Why use MUSCLE?

MUSCLE is chosen because:

- It gives **high-quality alignments**, often better than older tools like ClustalW.
- It is **fast**, even for large datasets.
- It supports **iterative refinement**, which means it repeatedly improves the alignment quality.
- It is trusted by researchers worldwide and used in many phylogenetic and comparative genomics studies.

How MUSCLE Works (Simplified Pipeline)

MUSCLE works in **3 main stages**:

1. Draft Progressive Alignment (Fast)

- MUSCLE begins by **calculating pairwise distances** (similarity scores) between all sequences using a fast method (like k-mer counting).
- From these scores, it builds a **guide tree** (rough evolutionary tree).
- It then **progressively aligns** sequences based on this tree—closely related sequences are aligned first.

2. Improved Tree Building

- Using the draft alignment, MUSCLE recalculates distances more accurately.
- A **new, improved tree** is built.
- The alignment is **reconstructed** using this new tree.

3. Refinement Iterations

- MUSCLE repeatedly splits the tree, realigns the parts, and checks if the alignment improves.
- If improvement is found, it keeps the change.
- This loop continues until no more improvements are found.

What MUSCLE does in our Project?

In our project, we're inputting **two COX1 gene sequences**, and using MUSCLE to:

1. **Align them properly**, accounting for insertions, deletions, and mutations.
2. Ensure homologous bases are lined up correctly for accurate comparison.
3. Output a **clean FASTA alignment**, which you can use to compute similarity percentage.

Even though MUSCLE is meant for multiple sequences, it works well for two as well—especially because it applies biologically informed gap penalties and tree-based logic.

Similarity Score Function Implementation.

To compute the similarity of the aligned sequences in the aligned FASTA file, we will:

1. **Read the aligned FASTA file** to get the sequences.

2. **Compare base-by-base** by calculating the number of matching bases between the two sequences.

3. **Calculate the percentage similarity** by dividing the number of matches by the total number of bases (after excluding gaps).

Steps involved:

1. **Read aligned sequences** using SeqIO.parse from BioPython.

2. **Compare base-by-base** for non-gap positions (ignoring '-').

3. **Calculate % similarity** using the formula:

Similarity = (Number of matches /Total valid bases) × 100

What this function does:

- Reads the aligned sequences from the input FASTA file.

- Compares the two sequences by checking each base (ignoring gaps) to see if they match.

- Calculates the percentage of similarity based on the total number of valid (non-gap) bases.

Complete Implementation

```
# START OF 1ST CELL

from Bio import Entrez, SeqIO
import os

# END OF 1ST CELL


# START OF 2ND CELL

# Set your email (made mandatory by NCBI to avoid spams)

Entrez.email = gunanka.is22@bmsce.ac.in

# END OF 2ND CELL


# START OF 3RD CELL

def fetch_cox1_sequence(species_name):
    """Fetches the COX1/PTGS1 gene sequence for a given species
from NCBI."""

    print(f"Fetching COX1 sequence for '{species_name}' species")
    try:
        # Querying using 3 different terms to make sure we hit
        search_terms = [
            f"{species_name}[Organism] AND (PTGS1[Gene] OR
cyclooxygenase-1[Gene Name])",
            f"{species_name}[Organism] AND (COX1[Gene] OR COX-
1[Gene] OR PTGS1[Gene])",
            f"{species_name}[Organism] AND cyclooxygenase 1"
        ]

        # try all terms
        for term in search_terms:
            print(f"Trying search term: {term}")
            handle = Entrez.esearch(db="nucleotide", term=term,
retmax=5)
            record = Entrez.read(handle)
            handle.close()

            # if record exists, get its id and sequence
            if record["IdList"]:
                # Fetch the sequence record
                seq_id = record["IdList"][0]
                print(f"SUCCESS : COX1 sequence for {species_name}
found. Found ID : {seq_id}")

                # print(f"Retrieved sequence with
header:\n{fasta_text.split('\\n')[0]}")

                # Retreive the sequence from NCBI
                print("Fetching COX-1 sequence from NCBI")
                handle = Entrez.efetch(db="nucleotide", id=seq_id,
rettype="fasta", retmode="text")
                seq_record = SeqIO.read(handle, "fasta")
                handle.close()
                print("Fetched!")

                print(f"Sequence length: {len(seq_record.seq)}
bp\n")

                return seq_record

        # None of the terms gave any records
        print(f"No COX1/PTGS1 sequence found for {species_name}")
        return None

    except Exception as e:
        print(f"Error fetching sequence: {e}")
        return None

# END OF 3RD CELL


# START OF 4TH CELL
# TEST CELL 1
# Testing fetch_cox1_sequence() ...

print("Testing fetch_cox1_sequence()...\n")

# Human beings
species_name = "Homo sapiens"
fetch_cox1_sequence(species_name)

# Chimpanzees
species_name = "Pan troglodytes"
fetch_cox1_sequence(species_name)

# Gold fish
species_name = "Carassius auratus"
```

```
fetch_cox1_sequence(species_name)

# True tuna fish
species_name = "Thunnus"
fetch_cox1_sequence(species_name)

# Yellow fin tuna fish
species_name = "Thunnus albacares"
fetch_cox1_sequence(species_name)

# Dinosaur ant
species_name = "Nothomyrmecia macrops"
fetch_cox1_sequence(species_name)

# Odorous house ant
species_name = "Tapinoma sessile"
fetch_cox1_sequence(species_name)

# Golden eagle
species_name = "Aquila chrysaetos"
fetch_cox1_sequence(species_name)

# Monarch butterfly
species_name = "Danaus plexippus"
fetch_cox1_sequence(species_name)

# Sacred Scarab (dung beetle)
species_name = "Scarabaeus sacer"
fetch_cox1_sequence(species_name)

# Blue Gourami
species_name = "Trichopodus trichopterus"
fetch_cox1_sequence(species_name)

# Silver Arowana
species_name = "Osteoglossum bicirrhosum"
fetch_cox1_sequence(species_name)

# Budgerigar
species_name = "Melopsittacus undulatus"
fetch_cox1_sequence(species_name)

print("Testing completed...")

# END OF 4TH CELL
# START OF 5TH CELL

from Bio.Seq import Seq
from Bio.SeqRecord import SeqRecord

def save_sequences_to_fasta(seq1, species_name1, seq2,
species_name2, file_path):
    """Saves 2 sequences onto a fasta file in the given file
location. Used for MUSCLE"""

    # Create SeqRecord objects
    record1 = SeqRecord(seq1.seq, id="seq1",
description=f"{species_name1} COX-1 sequence")
    record2 = SeqRecord(seq2.seq, id="seq2",
description=f"{species_name2} COX-1 sequence")

    # Write to a FASTA file
    with open(file_path, "w") as output_handle:
        SeqIO.write([record1, record2], output_handle, "fasta")

# END OF 5TH CELL


# START OF 6TH CELL

# TEST CELL 2
# Testing save_sequences_to_fasta() ...

print("Testing save_sequences_to_fasta()...\n")

# Human beings
species_name1 = "Homo sapiens"
seq1 = fetch_cox1_sequence(species_name1)

# Odorous house ant
species_name2 = "Tapinoma sessile"
seq2 = fetch_cox1_sequence(species_name2)

file_path = r"../fasta_files/species_cox1_sequences.fasta"

save_sequences_to_fasta(seq1, species_name1, seq2, species_name2,
file_path)

print("Testing completed...")

# END OF 6TH CELL

# START OF 7TH CELL

import subprocess # to use MUSCLE (multiple sequence alignment
tool)
```

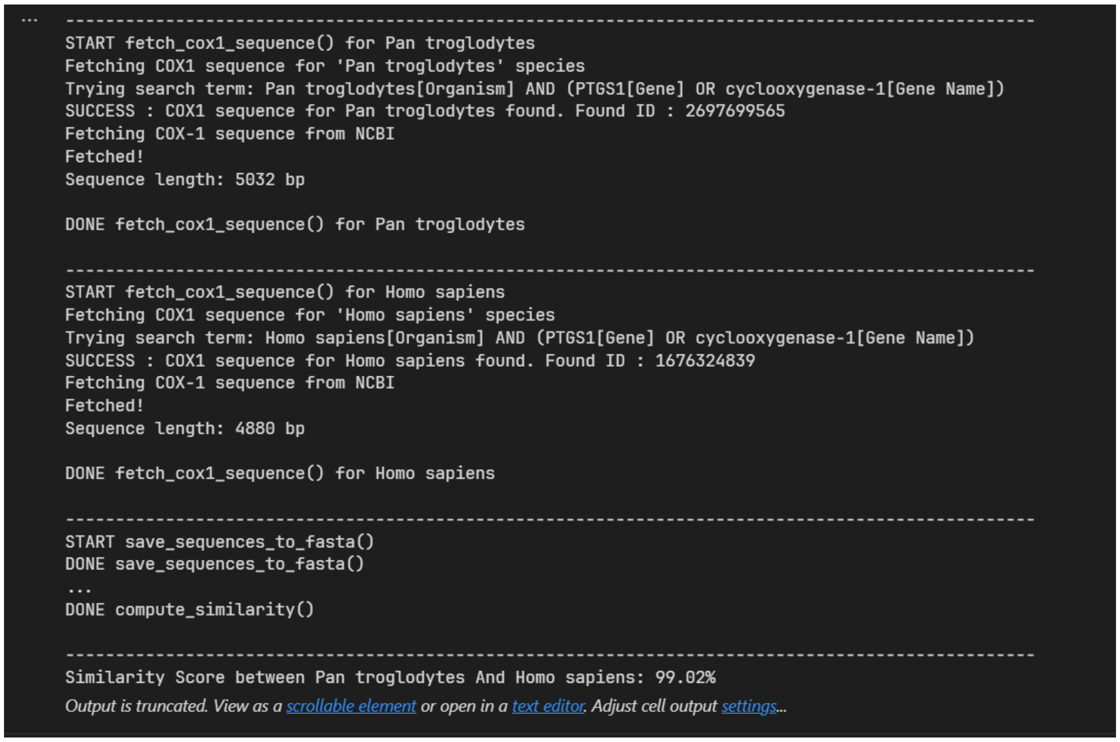
```
def align_sequences_muscle(input_fasta, output_fasta):
    """Aligns sequences using MUSCLE and saves the output to a
FASTA file."""

    muscle_path = r"../tools/muscle-win64.v5.3.exe"

    # Ensure the output directory exists
    output_dir = os.path.dirname(output_fasta)
    if not os.path.exists(output_dir):
        os.makedirs(output_dir)

     # For MUSCLE v5.3, the command line syntax has changed
    muscle_cmd = [
        muscle_path,
        "-align", input_fasta,
        "-output", output_fasta
    ]

    # Full command that is being executed
    print("Running command:", " ".join(muscle_cmd))

    # print("Starting Subprocess")
    result = subprocess.run(muscle_cmd, capture_output=True,
text=True)

    # DEBUG : Logs subprocess
    # print(f"Result stdout: {result.stdout}")
    print(f"\nSTATUS: {result.stderr}")

    if result.returncode != 0:
        print(f"Error: MUSCLE failed with exit code
{result.returncode}")

# END OF 7TH CELL


# START OF 8TH CELL

# TEST CELL 3
# Testing align_sequences_muscle() ...

print("Testing align_sequences_muscle()...\n")

input_fasta = r"../fasta_files/species_cox1_sequences.fasta"
output_fasta = r"../fasta_files/aligned_cox1_sequences.fasta"

align_sequences_muscle(input_fasta, output_fasta)
```

```
print("Testing completed...")

# END OF 8TH CELL


# START OF 9TH CELL

def compute_similarity(aligned_fasta):
    """Computes the similarity between two aligned sequences in a
FASTA file."""

    # Read the aligned sequences from the FASTA file
    aligned_sequences = list(SeqIO.parse(aligned_fasta, "fasta"))

    # Check if there are exactly two sequences
    if len(aligned_sequences) != 2:
        raise ValueError("The FASTA file must contain exactly two
sequences.")

    # Get the sequences
    seq1 = str(aligned_sequences[0].seq)
    seq2 = str(aligned_sequences[1].seq)

    # Initialize counters for matches and total length
    matches = 0
    total_bases = 0

    # Compare the sequences base-by-base
    for base1, base2 in zip(seq1, seq2):
        if base1 != '-' and base2 != '-':  # Ignore gaps ('-')
            total_bases += 1
            if base1 == base2:  # Count matches
                matches += 1

    # Calculate similarity percentage
    if total_bases == 0:
        return 0  # To avoid division by zero if there are no
valid bases (i.e., both are gaps)

    similarity_percentage = (matches / total_bases) * 100
    return similarity_percentage

# END OF 9TH CELL

# START OF 10TH CELL

# TEST CELL 4
# Testing compute_similarity()...
print("Testing compute_similarity()...\n")

aligned_fasta = r"../fasta_files/aligned_cox1_sequences.fasta"
similarity = compute_similarity(aligned_fasta)

print(f"Similarity Score: {similarity:.2f}%")
print("testing completed...")

# END OF 10TH CELL


# START OF 11TH CELL

def get_similarity_score(species1, species2):
    """Takes in 2 Species' Scientific names and returns their
similarity score based on COX-1 Genome sequence"""

    # STEP 1: Fetch COX-1 Sequences from NCBI using Entrez
    print("-------------------------------------------------------------
------------------------------------------")

    print(f"START fetch_cox1_sequence() for {species1}")
    seq1 = fetch_cox1_sequence(species1)
    print(f"DONE fetch_cox1_sequence() for {species1}\n")

    print("-------------------------------------------------------------
------------------------------------------")

    print(f"START fetch_cox1_sequence() for {species2}")
    seq2 = fetch_cox1_sequence(species2)
    print(f"DONE fetch_cox1_sequence() for {species2}\n")

    print("-------------------------------------------------------------
------------------------------------------")

    # STEP 2: Save these two sequences into a single fasta file
    print("START save_sequences_to_fasta()")
    cox_sequences_file_path =
r"../fasta_files/species_cox1_sequences.fasta"
    save_sequences_to_fasta(seq1, species1, seq2, species2,
cox_sequences_file_path)
    print("DONE save_sequences_to_fasta()\n")

    print("-------------------------------------------------------------
------------------------------------------")
```

```
    # STEP 3: Process above cox sequences fasta file using MUSCLE
and get aligned_fasta_file
    print("START align_sequences_muscle()")
    aligned_fasta_file_path =
r"../fasta_files/aligned_cox1_sequences.fasta"
    align_sequences_muscle(input_fasta=cox_sequences_file_path,
output_fasta=aligned_fasta_file_path)
    print("DONE align_sequences_muscle()\n")

    print("--------------------------------------------------------------
------------------------------------------")

    # STEP 4: Process above aligned sequences fasta file to get
similarity score
    print("START compute_similarity()")
    score = compute_similarity(aligned_fasta_file_path)
    print("DONE compute_similarity()\n")

    print("--------------------------------------------------------------
------------------------------------------")

    return score

# END OF 11TH CELL


# START OF 12TH CELL

# species1 = input("Enter first species' Scientific name: ")
# species2 = input("Enter second species' Scientific name: ")

species2 = "Homo sapiens"
species1 = "Pan troglodytes"

similarity_score = get_similarity_score(species1, species2)

print(f"Similarity Score between {species1} And {species2}:
{similarity_score:.2f}%")


# END OF 12TH CELL
```

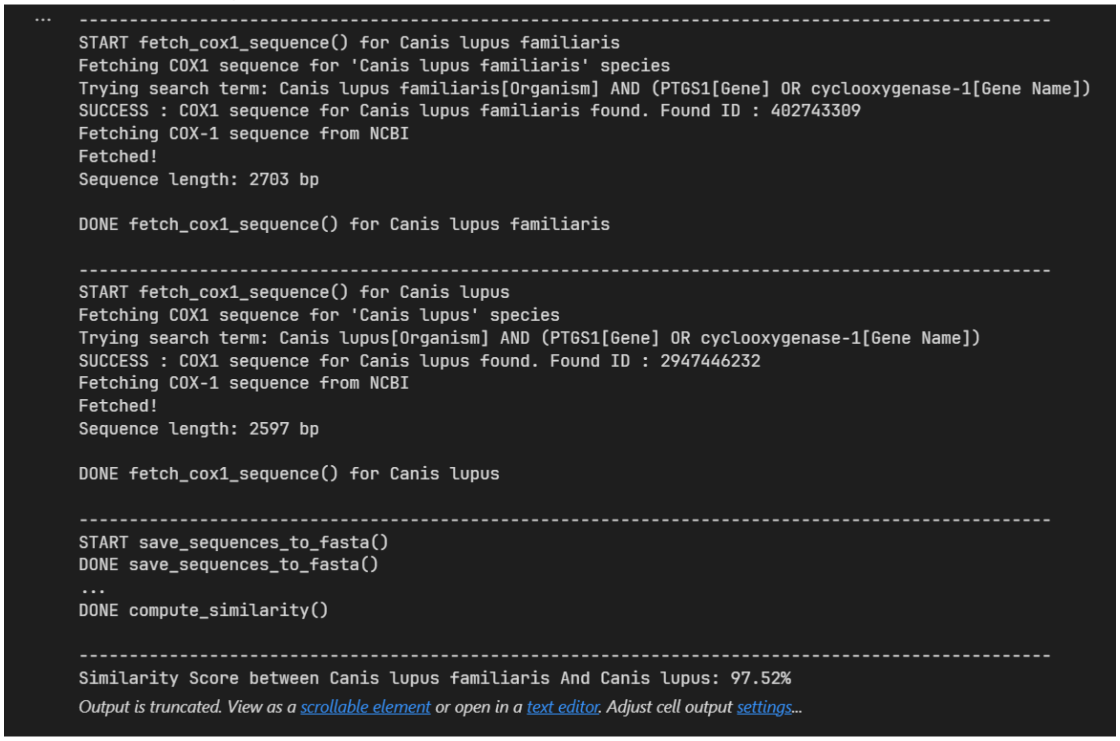
Results

1. Human being's vs Chimpanzees = 99.02%



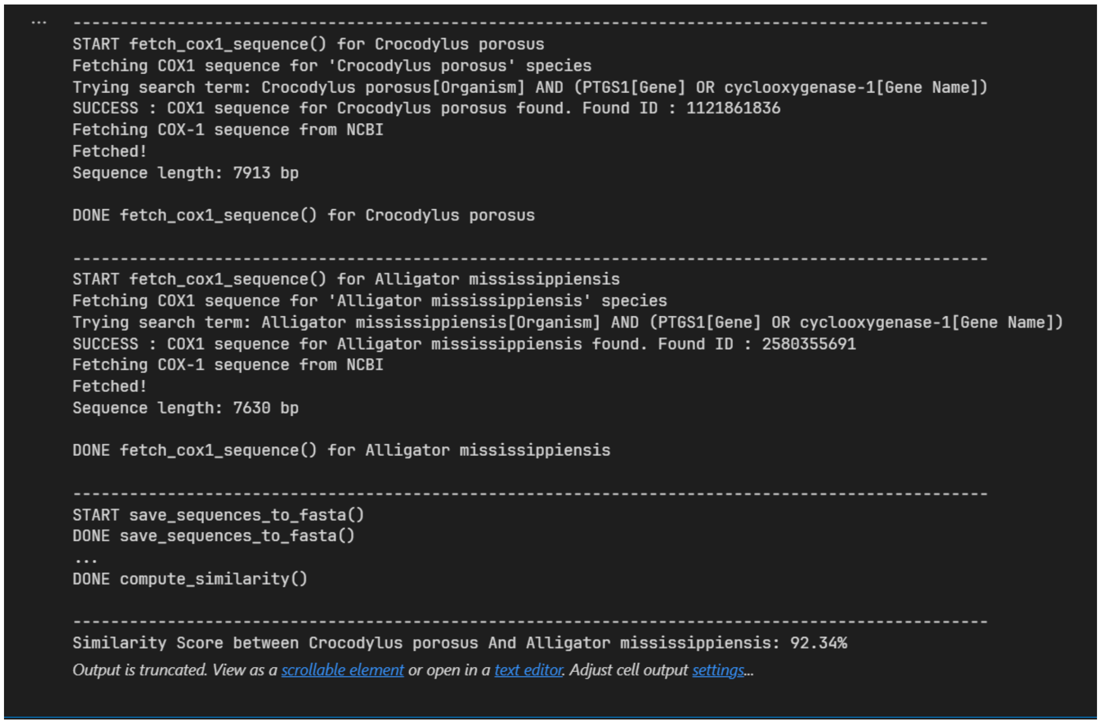
```
...  ------------------------------------------------------------------------------------------------
     START fetch_cox1_sequence() for Pan troglodytes
     Fetching COX1 sequence for 'Pan troglodytes' species
     Trying search term: Pan troglodytes[Organism] AND (PTGS1[Gene] OR cyclooxygenase-1[Gene Name])
     SUCCESS : COX1 sequence for Pan troglodytes found. Found ID : 2697699565
     Fetching COX-1 sequence from NCBI
     Fetched!
     Sequence length: 5032 bp

     DONE fetch_cox1_sequence() for Pan troglodytes

     ------------------------------------------------------------------------------------------------
     START fetch_cox1_sequence() for Homo sapiens
     Fetching COX1 sequence for 'Homo sapiens' species
     Trying search term: Homo sapiens[Organism] AND (PTGS1[Gene] OR cyclooxygenase-1[Gene Name])
     SUCCESS : COX1 sequence for Homo sapiens found. Found ID : 1676324839
     Fetching COX-1 sequence from NCBI
     Fetched!
     Sequence length: 4880 bp

     DONE fetch_cox1_sequence() for Homo sapiens

     ------------------------------------------------------------------------------------------------
     START save_sequences_to_fasta()
     DONE save_sequences_to_fasta()
     ...
     DONE compute_similarity()

     ------------------------------------------------------------------------------------------------
     Similarity Score between Pan troglodytes And Homo sapiens: 99.02%
     Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

2. Domestic dog vs Wolf = 97.52%



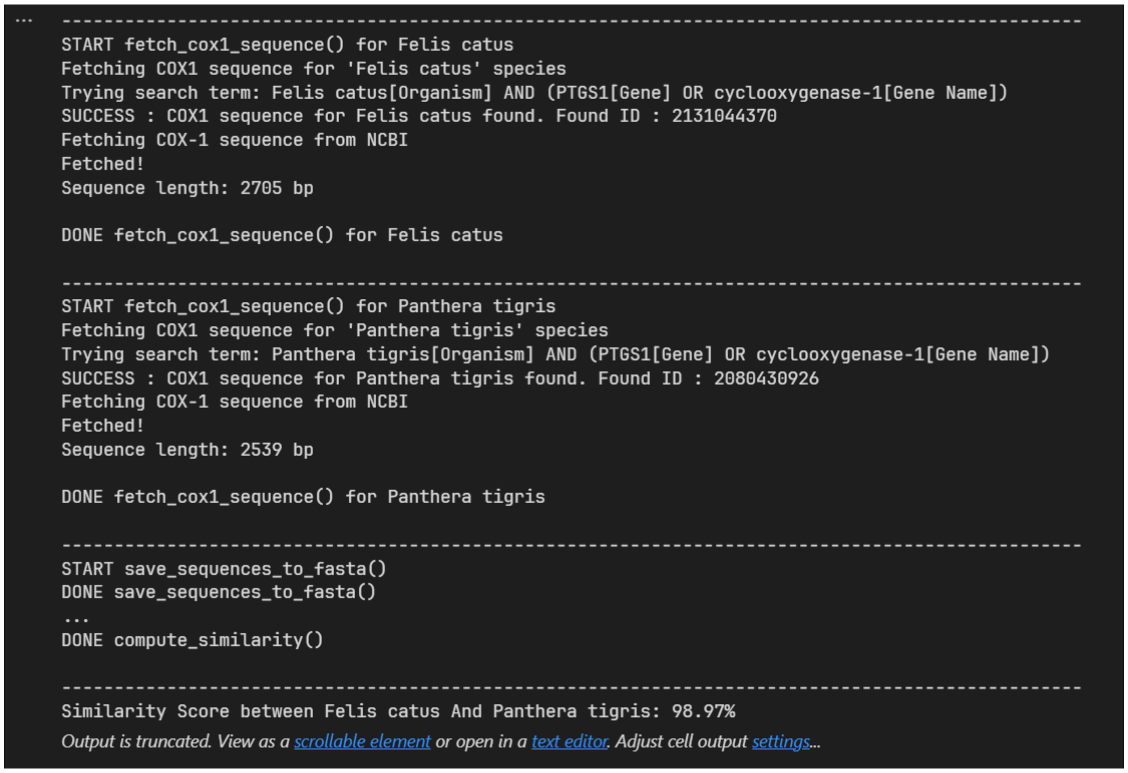
```
...  ------------------------------------------------------------------------------------------------
     START fetch_cox1_sequence() for Canis lupus familiaris
     Fetching COX1 sequence for 'Canis lupus familiaris' species
     Trying search term: Canis lupus familiaris[Organism] AND (PTGS1[Gene] OR cyclooxygenase-1[Gene Name])
     SUCCESS : COX1 sequence for Canis lupus familiaris found. Found ID : 402743309
     Fetching COX-1 sequence from NCBI
     Fetched!
     Sequence length: 2703 bp

     DONE fetch_cox1_sequence() for Canis lupus familiaris

     ------------------------------------------------------------------------------------------------
     START fetch_cox1_sequence() for Canis lupus
     Fetching COX1 sequence for 'Canis lupus' species
     Trying search term: Canis lupus[Organism] AND (PTGS1[Gene] OR cyclooxygenase-1[Gene Name])
     SUCCESS : COX1 sequence for Canis lupus found. Found ID : 2947446232
     Fetching COX-1 sequence from NCBI
     Fetched!
     Sequence length: 2597 bp

     DONE fetch_cox1_sequence() for Canis lupus

     ------------------------------------------------------------------------------------------------
     START save_sequences_to_fasta()
     DONE save_sequences_to_fasta()
     ...
     DONE compute_similarity()

     ------------------------------------------------------------------------------------------------
     Similarity Score between Canis lupus familiaris And Canis lupus: 97.52%
     Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

3. Saltwater Crocodile vs American Alligator = 92.34%



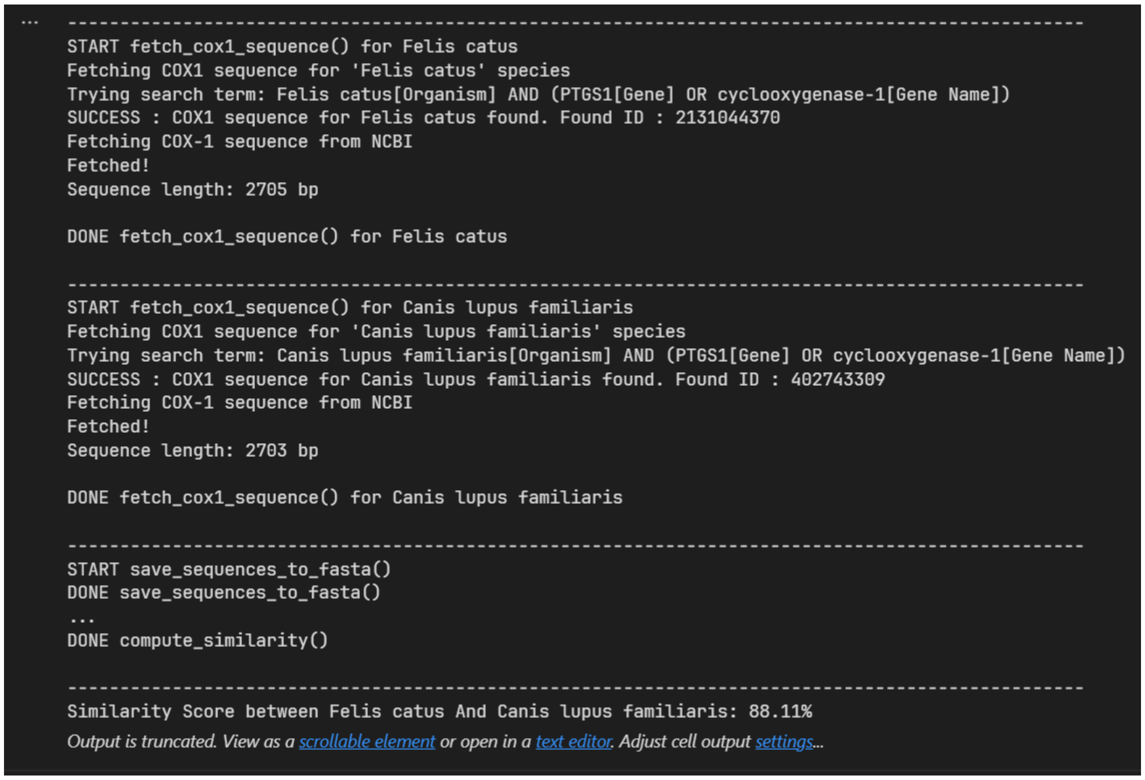
```
...  ---------------------------------------------------------------------------------------------
     START fetch_cox1_sequence() for Crocodylus porosus
     Fetching COX1 sequence for 'Crocodylus porosus' species
     Trying search term: Crocodylus porosus[Organism] AND (PTGS1[Gene] OR cyclooxygenase-1[Gene Name])
     SUCCESS : COX1 sequence for Crocodylus porosus found. Found ID : 1121861836
     Fetching COX-1 sequence from NCBI
     Fetched!
     Sequence length: 7913 bp

     DONE fetch_cox1_sequence() for Crocodylus porosus

     ---------------------------------------------------------------------------------------------
     START fetch_cox1_sequence() for Alligator mississippiensis
     Fetching COX1 sequence for 'Alligator mississippiensis' species
     Trying search term: Alligator mississippiensis[Organism] AND (PTGS1[Gene] OR cyclooxygenase-1[Gene Name])
     SUCCESS : COX1 sequence for Alligator mississippiensis found. Found ID : 2580355691
     Fetching COX-1 sequence from NCBI
     Fetched!
     Sequence length: 7630 bp

     DONE fetch_cox1_sequence() for Alligator mississippiensis

     ---------------------------------------------------------------------------------------------
     START save_sequences_to_fasta()
     DONE save_sequences_to_fasta()
     ...
     DONE compute_similarity()

     ---------------------------------------------------------------------------------------------
     Similarity Score between Crocodylus porosus And Alligator mississippiensis: 92.34%
     Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

4. Domestic Cat vs Tiger = 98.97%



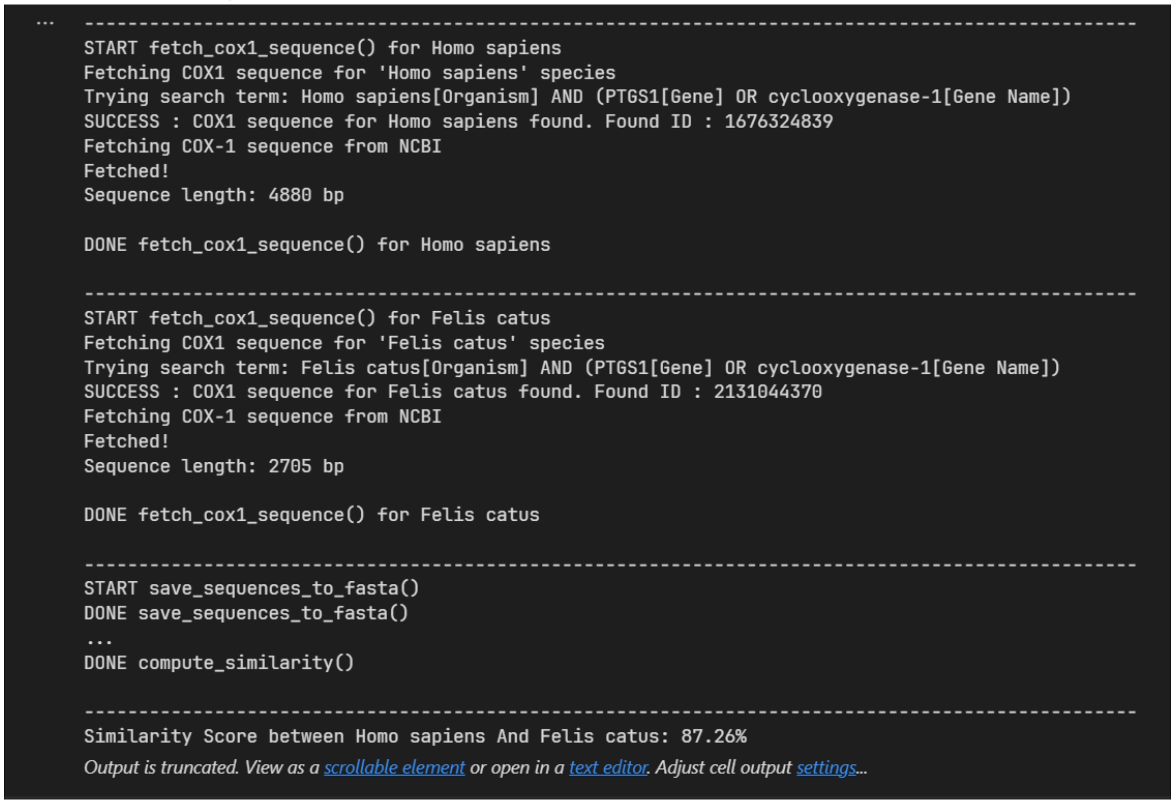
```
...  ---------------------------------------------------------------------------------------------
     START fetch_cox1_sequence() for Felis catus
     Fetching COX1 sequence for 'Felis catus' species
     Trying search term: Felis catus[Organism] AND (PTGS1[Gene] OR cyclooxygenase-1[Gene Name])
     SUCCESS : COX1 sequence for Felis catus found. Found ID : 2131044370
     Fetching COX-1 sequence from NCBI
     Fetched!
     Sequence length: 2705 bp

     DONE fetch_cox1_sequence() for Felis catus

     ---------------------------------------------------------------------------------------------
     START fetch_cox1_sequence() for Panthera tigris
     Fetching COX1 sequence for 'Panthera tigris' species
     Trying search term: Panthera tigris[Organism] AND (PTGS1[Gene] OR cyclooxygenase-1[Gene Name])
     SUCCESS : COX1 sequence for Panthera tigris found. Found ID : 2080430926
     Fetching COX-1 sequence from NCBI
     Fetched!
     Sequence length: 2539 bp

     DONE fetch_cox1_sequence() for Panthera tigris

     ---------------------------------------------------------------------------------------------
     START save_sequences_to_fasta()
     DONE save_sequences_to_fasta()
     ...
     DONE compute_similarity()

     ---------------------------------------------------------------------------------------------
     Similarity Score between Felis catus And Panthera tigris: 98.97%
     Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

5. Domestic Cat vs Domestic Dog = 88.11%



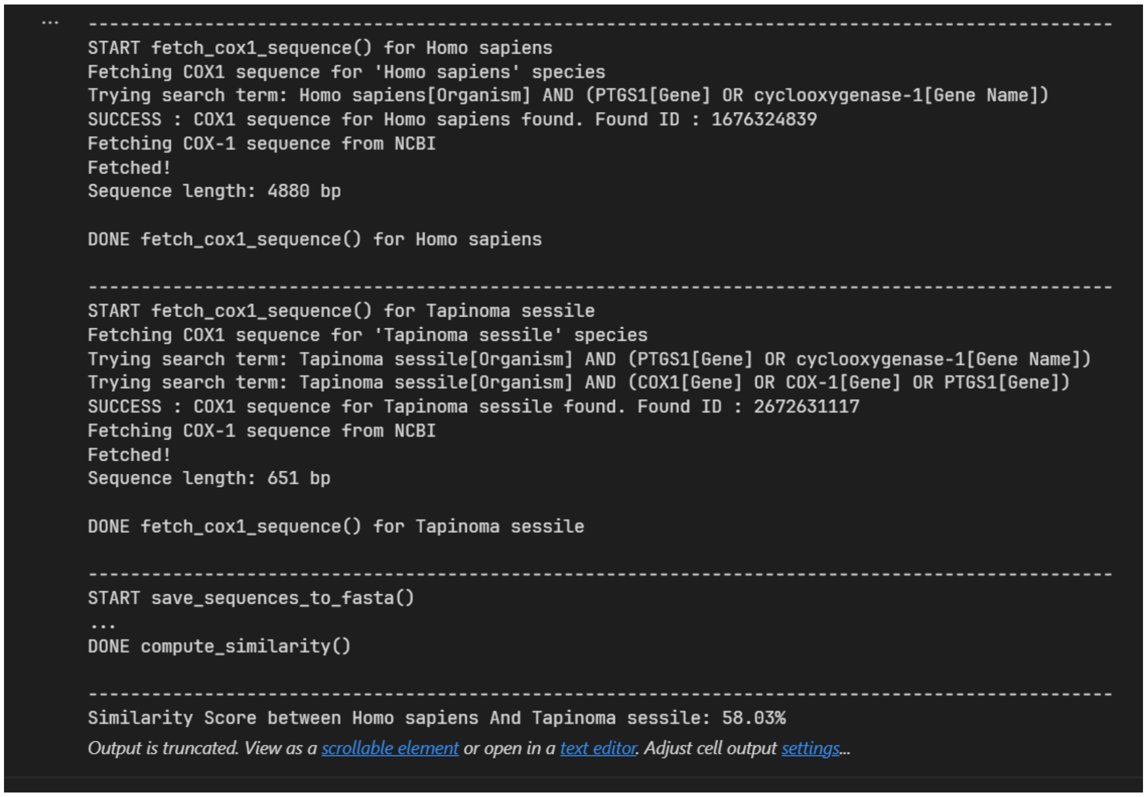
```
...  --------------------------------------------------------------------------------------------------
     START fetch_cox1_sequence() for Felis catus
     Fetching COX1 sequence for 'Felis catus' species
     Trying search term: Felis catus[Organism] AND (PTGS1[Gene] OR cyclooxygenase-1[Gene Name])
     SUCCESS : COX1 sequence for Felis catus found. Found ID : 2131044370
     Fetching COX-1 sequence from NCBI
     Fetched!
     Sequence length: 2705 bp

     DONE fetch_cox1_sequence() for Felis catus

     --------------------------------------------------------------------------------------------------
     START fetch_cox1_sequence() for Canis lupus familiaris
     Fetching COX1 sequence for 'Canis lupus familiaris' species
     Trying search term: Canis lupus familiaris[Organism] AND (PTGS1[Gene] OR cyclooxygenase-1[Gene Name])
     SUCCESS : COX1 sequence for Canis lupus familiaris found. Found ID : 402743309
     Fetching COX-1 sequence from NCBI
     Fetched!
     Sequence length: 2703 bp

     DONE fetch_cox1_sequence() for Canis lupus familiaris

     --------------------------------------------------------------------------------------------------
     START save_sequences_to_fasta()
     DONE save_sequences_to_fasta()
     ...
     DONE compute_similarity()

     --------------------------------------------------------------------------------------------------
     Similarity Score between Felis catus And Canis lupus familiaris: 88.11%
     Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

6. Human being's vs Domestic cats = 88.26%



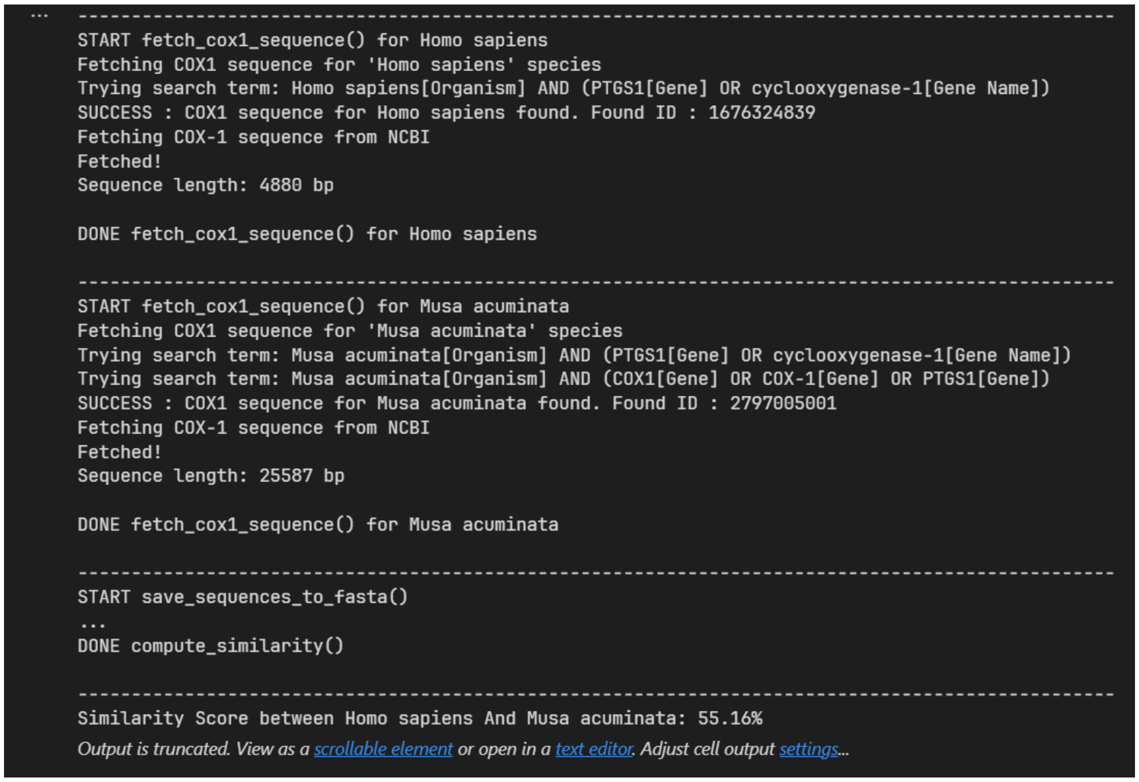
```
...  --------------------------------------------------------------------------------------------------
     START fetch_cox1_sequence() for Homo sapiens
     Fetching COX1 sequence for 'Homo sapiens' species
     Trying search term: Homo sapiens[Organism] AND (PTGS1[Gene] OR cyclooxygenase-1[Gene Name])
     SUCCESS : COX1 sequence for Homo sapiens found. Found ID : 1676324839
     Fetching COX-1 sequence from NCBI
     Fetched!
     Sequence length: 4880 bp

     DONE fetch_cox1_sequence() for Homo sapiens

     --------------------------------------------------------------------------------------------------
     START fetch_cox1_sequence() for Felis catus
     Fetching COX1 sequence for 'Felis catus' species
     Trying search term: Felis catus[Organism] AND (PTGS1[Gene] OR cyclooxygenase-1[Gene Name])
     SUCCESS : COX1 sequence for Felis catus found. Found ID : 2131044370
     Fetching COX-1 sequence from NCBI
     Fetched!
     Sequence length: 2705 bp

     DONE fetch_cox1_sequence() for Felis catus

     --------------------------------------------------------------------------------------------------
     START save_sequences_to_fasta()
     DONE save_sequences_to_fasta()
     ...
     DONE compute_similarity()

     --------------------------------------------------------------------------------------------------
     Similarity Score between Homo sapiens And Felis catus: 87.26%
     Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

7. Human being's vs Odorous house ants = 58.03%



```
...
    ------------------------------------------------------------------------------------------------
    START fetch_cox1_sequence() for Homo sapiens
    Fetching COX1 sequence for 'Homo sapiens' species
    Trying search term: Homo sapiens[Organism] AND (PTGS1[Gene] OR cyclooxygenase-1[Gene Name])
    SUCCESS : COX1 sequence for Homo sapiens found. Found ID : 1676324839
    Fetching COX-1 sequence from NCBI
    Fetched!
    Sequence length: 4880 bp

    DONE fetch_cox1_sequence() for Homo sapiens

    ------------------------------------------------------------------------------------------------
    START fetch_cox1_sequence() for Tapinoma sessile
    Fetching COX1 sequence for 'Tapinoma sessile' species
    Trying search term: Tapinoma sessile[Organism] AND (PTGS1[Gene] OR cyclooxygenase-1[Gene Name])
    Trying search term: Tapinoma sessile[Organism] AND (COX1[Gene] OR COX-1[Gene] OR PTGS1[Gene])
    SUCCESS : COX1 sequence for Tapinoma sessile found. Found ID : 2672631117
    Fetching COX-1 sequence from NCBI
    Fetched!
    Sequence length: 651 bp

    DONE fetch_cox1_sequence() for Tapinoma sessile

    ------------------------------------------------------------------------------------------------
    START save_sequences_to_fasta()
    ...
    DONE compute_similarity()

    ------------------------------------------------------------------------------------------------
    Similarity Score between Homo sapiens And Tapinoma sessile: 58.03%
    Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

8. Human being's vs Cavendish Banana (*pach balehannu*) = 55.16%



```
...
    ------------------------------------------------------------------------------------------------
    START fetch_cox1_sequence() for Homo sapiens
    Fetching COX1 sequence for 'Homo sapiens' species
    Trying search term: Homo sapiens[Organism] AND (PTGS1[Gene] OR cyclooxygenase-1[Gene Name])
    SUCCESS : COX1 sequence for Homo sapiens found. Found ID : 1676324839
    Fetching COX-1 sequence from NCBI
    Fetched!
    Sequence length: 4880 bp

    DONE fetch_cox1_sequence() for Homo sapiens

    ------------------------------------------------------------------------------------------------
    START fetch_cox1_sequence() for Musa acuminata
    Fetching COX1 sequence for 'Musa acuminata' species
    Trying search term: Musa acuminata[Organism] AND (PTGS1[Gene] OR cyclooxygenase-1[Gene Name])
    Trying search term: Musa acuminata[Organism] AND (COX1[Gene] OR COX-1[Gene] OR PTGS1[Gene])
    SUCCESS : COX1 sequence for Musa acuminata found. Found ID : 2797005001
    Fetching COX-1 sequence from NCBI
    Fetched!
    Sequence length: 25587 bp

    DONE fetch_cox1_sequence() for Musa acuminata

    ------------------------------------------------------------------------------------------------
    START save_sequences_to_fasta()
    ...
    DONE compute_similarity()

    ------------------------------------------------------------------------------------------------
    Similarity Score between Homo sapiens And Musa acuminata: 55.16%
    Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

Conclusion

This project demonstrates the practical application of bioinformatics in understanding the **evolutionary relationships between species** by analyzing DNA sequence similarity. By using the mitochondrial gene **COX1** as a genetic marker, and leveraging powerful tools like **Biopython** and **MUSCLE**, we developed a pipeline that automates the process of fetching DNA sequences, aligning them accurately, and calculating their similarity percentage. This process models the kind of molecular comparison that underpins modern evolutionary biology and DNA barcoding.

One of the core outcomes of the project is the **similarity percentage** produced at the end of sequence alignment. This value is not just a mathematical score—it is a **biological signal**. A high percentage (e.g., above 95%) suggests that the species being compared have **very few genetic differences** in the chosen gene, and likely share a **recent common ancestor**. Conversely, a lower percentage indicates **greater genetic divergence**, implying that the species diverged **earlier in evolutionary time**. Thus, this single value can provide insight into **how closely or distantly two species are related** on an evolutionary timeline.

For example, when comparing species like *Homo sapiens* and *Pan troglodytes* (chimpanzees), we expect a very high similarity in their COX1 gene sequences, often above **98%**, reflecting a divergence time of around **5–7 million years ago**. On the other hand, a comparison between humans and more distant species, such as fish or insects, should yield a significantly lower similarity, matching their divergence times of hundreds of millions of years.

The value of this similarity metric becomes even more meaningful when placed in **phylogenetic context**. By aligning multiple species and comparing similarities, one could build a **phylogenetic tree** that visually illustrates the evolutionary distances between species. Though this project focuses on pairwise similarity, its modular design can be easily extended to support tree generation using Biopython's Phylo module.

From a scientific perspective, this project captures a **miniature version of what real-world molecular phylogeneticists do**, but in a simplified and accessible way. In actual research settings, scientists analyze thousands of genes or even entire genomes, but the fundamental principles remain the same—**sequence comparison, alignment, and evolutionary inference**. By focusing on COX1, which is a widely accepted barcoding gene, we ensured that the analysis remained accurate and meaningful within the scope of cross-species comparison.

Furthermore, this project highlights how computational tools can greatly reduce the barrier to entry for evolutionary research. What once required laboratory techniques, gel electrophoresis, and manual sequence matching can now be done with **a few lines of code**. This democratization of genomics opens doors for students, educators, and researchers to engage with real biological data and explore the diversity of life in a hands-on way.

In summary, the similarity percentage derived from aligned COX1 sequences offers a **quantifiable, interpretable measure of evolutionary closeness**. It reflects shared ancestry, divergence timelines, and genetic conservation—all of which are key to understanding how species have evolved and diversified over millions of years. This project not only builds technical skills in bioinformatics but also reinforces deep biological concepts about evolution, making it a valuable contribution to both education and exploratory research.

References

1. Scientific Reports – Standardized phylogenetic and molecular evolutionary analysis applied to species across the microbial tree of life
https://www.nature.com/articles/s41598-019-47530-x

2. Wikipedia – 16S ribosomal RNA
https://en.wikipedia.org/wiki/16S_ribosomal_RNA

3. Wikipedia – 18S ribosomal RNA
https://en.wikipedia.org/wiki/18S_ribosomal_RNA

4. PubMed Central – General properties and phylogenetic utilities of nuclear ribosomal DNA and mitochondrial DNA
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3417294

5. PubMed Central – Phylogenetic Resolution of Deep Eukaryotic and Fungal Relationships Using Highly Conserved Low-Copy Nuclear Genes
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3328802

6. PubMed Central – Common Methods for Phylogenetic Tree Construction and Their Implementation in R
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8591377

7. MEGA Software – Jukes-Cantor Distance
https://www.megasoftware.net/web_help_11/Distance/Jukes-Cantor_distance.htm

8. Wikipedia – Models of DNA Evolution
https://en.wikipedia.org/wiki/Models_of_DNA_evolution

9. BMC Bioinformatics – Phylogenetic Convolutional Neural Networks in Metagenomics
https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-020-3429-2

10. ChatGPT Deep Research
https://chatgpt.com/s/dr_680cd6d752508191a9db7dc058f6ec9e

11. '*Genetic Similarities: Wilson, Sarich, Sibley, and Ahlquist*'
https://evolution.berkeley.edu/the-history-of-evolutionary-thought/1900-to-present/genetic-similarities-wilson-sarich-sibley-and-ahlquist/