## **Assignment 2: Advanced Genomic Analysis and Pattern Recognition (CCA4)**

Kedar Oza 1032240040

*Total: 100 Marks*

## ***Submission Requirements:***

1. *Code Quality: Well-commented, readable Python code following PEP 8 standards*
2. *Documentation: Include docstrings for all functions and classes*
3. *Testing: Provide test cases demonstrating functionality*
4. *Performance Analysis: Include time and space complexity analysis where relevant*
5. *GitHub Repository: Submit code via version control with proper commit history*

## ***Evaluation Criteria:***

* *Correctness (40%): Algorithm implementation accuracy and output validity*
* *Efficiency (25%): Code optimization and performance considerations*
* *Code Quality (20%): Readability, documentation, and best practices*
* *Innovation (15%): Creative solutions and additional features*

## ***Resources Allowed:***

* *Python standard library and common packages (NumPy, Pandas for data analysis)*
* *Biological sequence databases for testing*
* *Course materials and rebelScience video series*
* *Standard bioinformatics references and documentation*

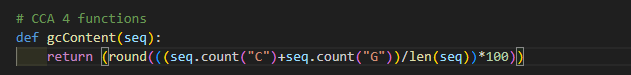
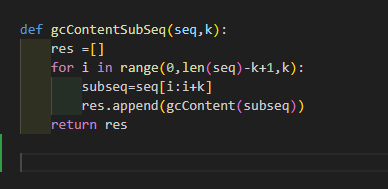
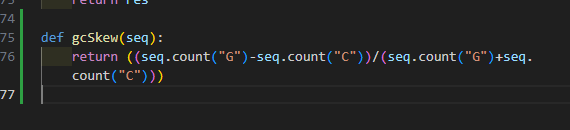
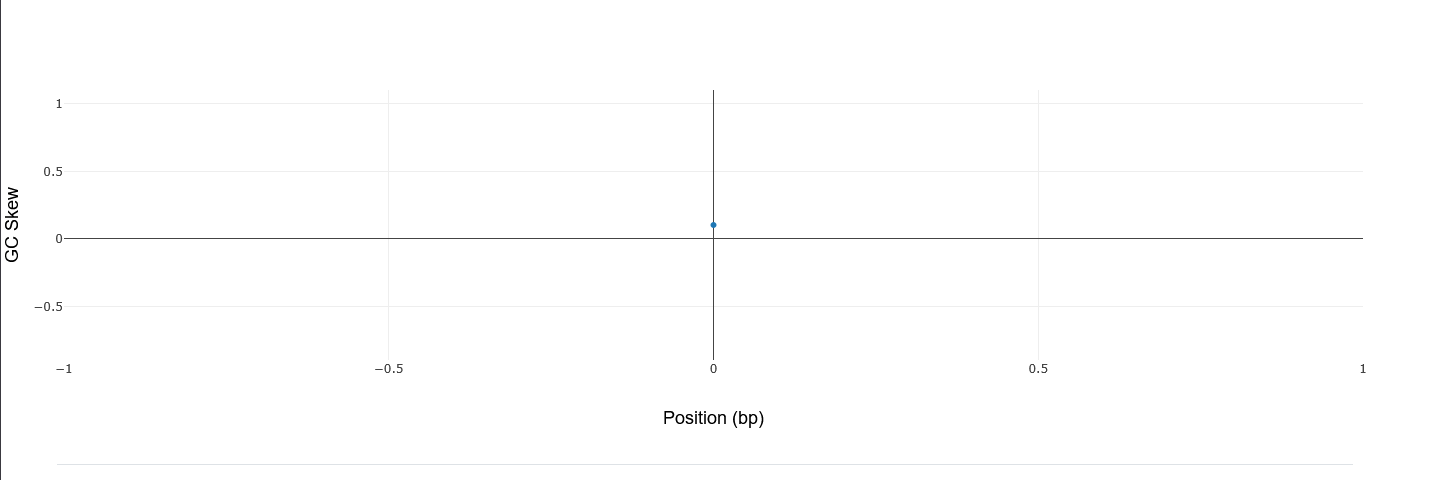
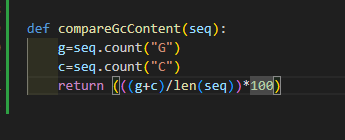
## ***Timeline:***

* ***Assignment Deadline: 28 October 2025***
* *Submission Format: Python files (.py), Jupyter notebooks (.ipynb), and documentation*
* *Presentation: Brief demonstration of key algorithms and results*

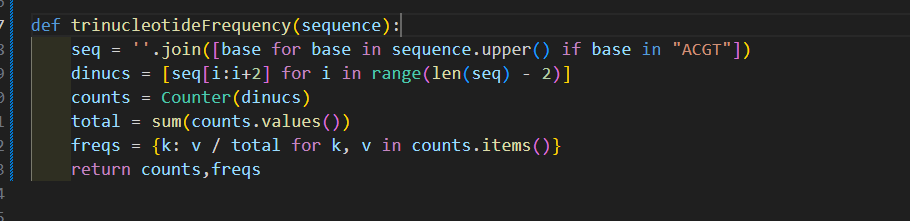
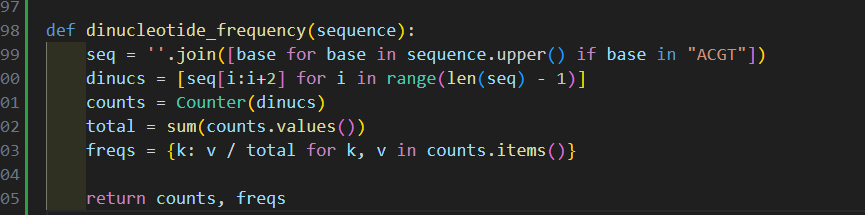
*These assignments progressively build from basic DNA manipulation to advanced genomic analysis, reflecting the curriculum outlined in the rebelScience series while providing comprehensive assessment of biological algorithm implementation skills in Python.*

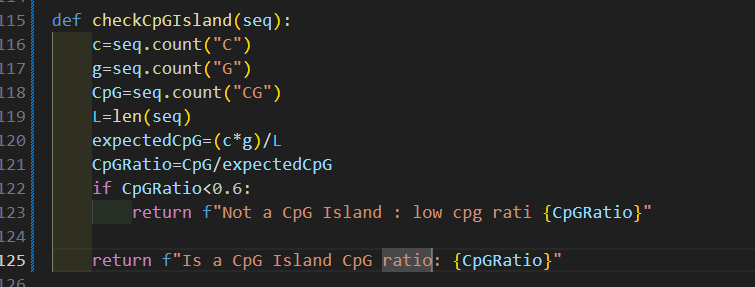
## **Part A: GC Content and Sequence Analysis (30 marks)**

Question 1: GC Content Calculation (15 marks)  
Implement comprehensive GC analysis:

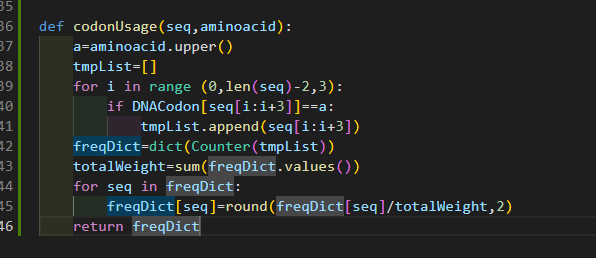
* Calculate overall GC content percentage
* 
* Analyze GC content in sliding windows
* 
* Generate GC skew plots and statistics
* 
* 
* Compare GC content across species or sequences
* 

Question 2: Sequence Composition Analysis (15 marks)  
Develop advanced composition tools:

* Calculate dinucleotide and trinucleotide frequencies
* 
* Identify CpG islands in genomic sequences



* Analyze codon usage bias in protein-coding regions

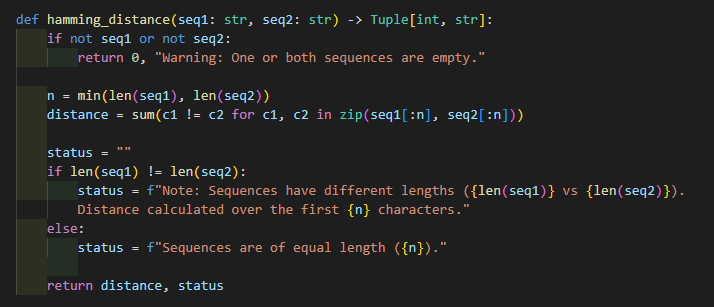


* Generate composition-based sequence classification

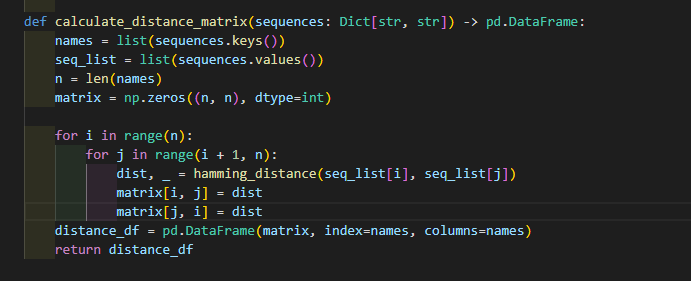
## **Part B: Pattern Matching and Motif Discovery (40 marks)**

Question 3: Hamming Distance Implementation (15 marks)  
Create sophisticated distance calculations:

* Implement Hamming distance for sequence comparison

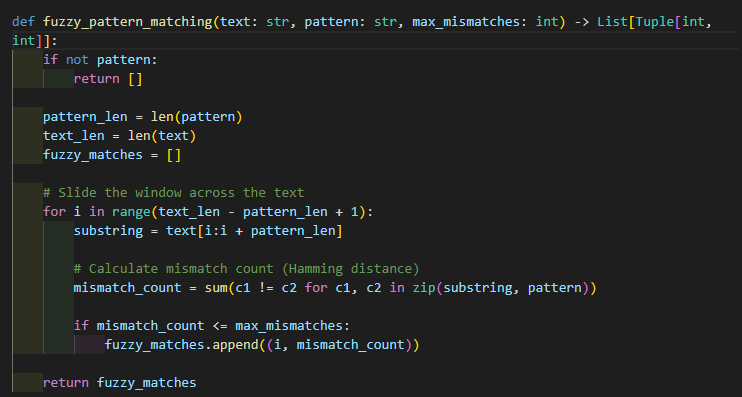


* Handle sequences of different lengths
* Calculate distance matrices for multiple sequences



* Visualize distance relationships using appropriate data structures

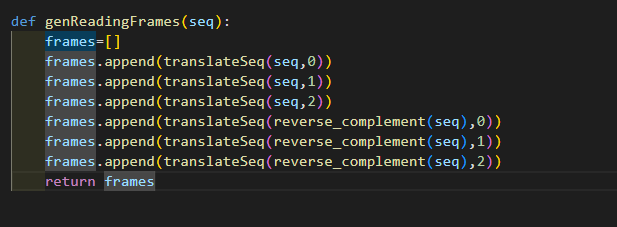
Question 4: Motif Finding Algorithms (25 marks)  
Develop pattern recognition systems:

* Implement exact pattern matching algorithms
* Create fuzzy matching with allowed mismatches
* 
* Find all occurrences of patterns in genomic sequences
* Analyze motif conservation across multiple sequences

## **Part C: Protein Sequence Analysis (30 marks)**

Question 5: Reading Frame Analysis (15 marks)  
Build comprehensive ORF detection:

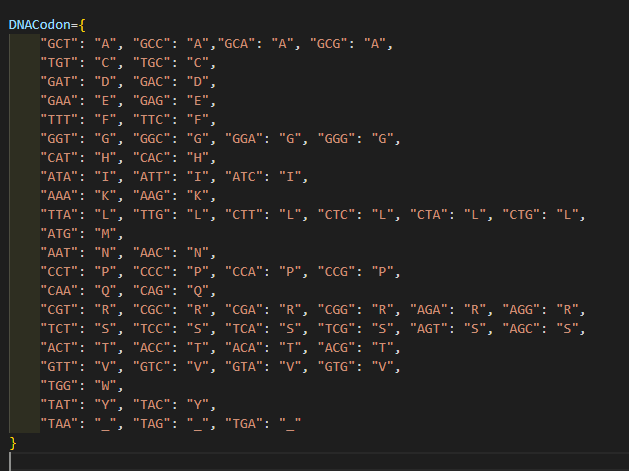
* Identify all six reading frames in DNA sequences

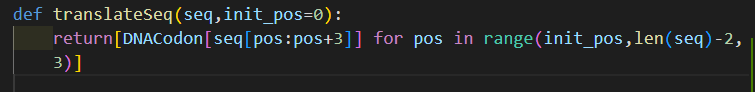


* Find start and stop codons in each frame
* Extract potential protein sequences
* Calculate ORF statistics and length distributions

Question 6: Protein Translation and Analysis (15 marks)  
Implement translation machinery:

* Convert DNA/RNA to amino acid sequences





* Handle the genetic code using dictionaries
* Identify and translate the longest ORFs
* Analyze amino acid composition and properties

Git hub link : <https://github.com/kedar1100/BioInfo.git>