GUJARAT TECHNOLOGICAL UNIVERSITY

Program Name: Bachelor of Engineering
Level: UG
Branch: Biotechnology
Subject Code: BE03004021
Subject Name: Computational Biology

w. e. f. Academic Year:	**2025-26**
Semester:	**3**
Category of the Course:	**Professional Core Course**

Prerequisite:	Linear Algebra, Calculus and Ordinary Differential Equation, Integral and Vector Calculus, Laplace and Fourier Transforms. Programming and Data Structures Biology for Engineers
Rationale:	Computational biology integrates biology with data science, enabling analysis of complex biological systems. In the future, it will drive personalized medicine, drug discovery, and genomic research, solving global health challenges. Its interdisciplinary nature equips students with critical skills for innovation in biotechnology, healthcare, and sustainable development.

Course Outcomes:

Sr. No.	CO statement	Marks% weightage
CO-1	Understand the scripting language procedures	15
CO-2	Apply the programming structures, file handling and file management to develop programs	15
CO-3	Develop basic programs in Python	25
CO-4	Apply the basics of SQL to create database	25
CO-5	Apply the Biocomputing knowledge for any given biological problem	20

Teaching and Examination Scheme:

Teaching / Learning Scheme (in Hours per semester)					Total Credits	Assessment Pattern and Marks					Total Marks
						Theory		Tutorial / Practical			
L	T	P	PBL*	Total no of hours per semester		ESE (E)	PA / CA (M)	PA/CA (I)	PBL (I)	ESE (V)	
45	0	30	45	120	4	70	30	20	30	50	200

* Problem Based Learning (PBL) aims to accommodate learning beyond syllabus as per clause 9.4 of NBA manual.

Content:

Sr. No.	Content	Total Hrs
1	**SQL Programming:** Introduction to SQL, its role, key features, and advantages. Creating simple databases, retrieving and summarizing data, adding, deleting, and updating records, and ensuring data security. Understanding SQL basics: statements, identifiers (table names, column names), data types, constants (numeric, string, date/time, symbolic), built-in functions, handling missing data	**10**

	(null values), and executing queries.	
2	**Introduction to Python:** Overview of Python, its history, features, and development tools. Writing Python programs; understanding values and variables including numeric values, expressions, and variable assignments. Introduction to operators: arithmetic, assignment, comparison, logical, and bitwise. Exploring core data types: numbers, booleans, and strings.	**10**
3	**Functions and Flow Statements:** Sequence overview, common sequence operations, indexing, slicing, adding sequences, multiplication, length, minimum, and maximum, lists: python's workhorse, the list function, basic list operations, list methods, tuples: immutable sequences, the tuple function, basic tuple operations. control flow statements: simple if statement, if/else statement, if/elif statement, nested if statement, conditional expressions, while loop, for loop, continue and break statements.	**15**
4	**Strings and Files Handling:** Exception handling, try, except statement, functions; built-in functions, commonly used modules. strings; creating and storing strings, the str () function, basic string operations, string comparison, concatenation split, join, slice, formatting strings. files, creating and opening text files, reading and writing data and csv file concepts.	**10**
	TOTAL	**45**

Suggested Specification table with Marks (Theory): (For B.E. only)

Distribution of Theory Marks					
R Level	U Level	A Level	N Level	E Level	C Level
20	**20**	**20**	**20**	**10**	**10**

R: Remembrance; U: Understanding; A: Application, N: Analyze and E: Evaluate C: Create and above Levels (Revised Bloom's Taxonomy)

Reference Books:

1. Gowrishankar S, Veena A, Introduction to python programming, CRC Press, 2019, 1st Edition
2. Magnus Lie Hetland, Beginning Python, Apress, 2008, 1st Edition
3. James R Groff, Paul N. Weinberg and Andrew J. Oppel, SQL: The Complete Reference, McGraw-Hill Education, 2010, 3rd Edition
4. Jason M. Kinser, Python for Bioinformatics, Jones and Bartlett Publishers, 2020, 1st Edition
5. Steve Conger, Hands-On Database: An Introduction to Database Design and Development, Pearson Education, Inc., 2012, 3rd Edition

List of Experiments:

1. Create a relational SQL database for genes and proteins, and perform basic data queries.
2. Use SQL functions to summarize genomic data, including handling missing (NULL) values.
3. Write a Python script to read a DNA sequence and calculate its GC content.
4. Develop a Python program to compute the molecular weight of a protein from its sequence.
5. Import gene expression data from a CSV file, filter based on expression levels, and save results.
6. Simulate random point mutations in DNA sequences and track the number of changes.
7. Compare two biological sequences in Python to calculate similarity and identify mismatches.
8. Apply control flow statements to parse and extract biological information conditionally.
9. Use exception handling in Python for safe reading/writing of biological data files.
10. Connect Python to an SQL database to build a simple interface for querying gene/protein data.

Major Equipment: Nil
List of Open Source Software/learning website:
Open Source Software

1. Biopython – https://biopython.org
2. Clustal Omega – https://www.ebi.ac.uk/Tools/msa/clustalo/
3. MEGA – https://www.megasoftware.net
4. UCSC Genome Browser – https://genome.ucsc.edu
5. GROMACS – http://www.gromacs.org
6. PyMOL – https://pymol.org/2/
7. T-Coffee – http://tcoffee.crg.cat
8. Galaxy Project – https://usegalaxy.org
9. Open Babel – http://openbabel.org

Learning Websites

1. Coursera – https://www.coursera.org/specializations/bioinformatics
2. edX – https://www.edx.org/learn/computational-biology
3. NCBI Education – https://www.ncbi.nlm.nih.gov/education/
4. EMBL-EBI Training – https://www.ebi.ac.uk/training/
5. Rosalind – http://rosalind.info
6. NPTEL – https://nptel.ac.in
7. YouTube (e.g., StatQuest, Bioinformatics Review)

***List of suggested activities for Problem Based Learning:**

1. Literature Review: Study recent research papers on computational approaches in genomics, proteomics, and systems biology.
2. Bioinformatics Tool Exploration: Use tools like BLAST, ClustalW, and ORF Finder to analyze biological sequences.
3. Database Familiarization: Explore and document the use of key databases such as NCBI, UniProt, PDB, and KEGG.

4. Mini Project: Undertake a small project such as sequence alignment, phylogenetic tree construction, or protein structure prediction.
5. Coding Practice: Write basic Python/R scripts for sequence analysis, data visualization, or machine learning applications in biology.
6. Case Studies: Analyze real-world case studies involving computational approaches to drug discovery or disease modeling.
7. MOOCs and Online Courses: Complete modules on computational biology or bioinformatics from platforms like Coursera, edX, or NPTEL.
8. Seminar Presentation: Prepare and present on emerging topics such as AI in genomics, computational vaccinology, or synthetic biology.
9. Algorithm Design: Implement simple algorithms used in computational biology like Needleman-Wunsch or Smith-Waterman.
10. Group Discussion & Peer Learning: Conduct weekly sessions to discuss research trends, ethical implications, and innovations in the field.

****