Python Fundamentals for Bioinformatics

Module 1: Introduction to Python and Bioinformatics

- **Lesson 1.1: Overview of Bioinformatics and Its Importance**
    - Introduction to bioinformatics and its applications.
    - Why Python is the preferred language for bioinformatics.
- **Lesson 1.2: Setting Up the Python Environment**
    - Installing Python, Jupyter Notebook, and IDEs.
    - Overview of Command Line Python and Jupyter Notebook
- **Lesson 1.3: Basic Python Syntax**
    - Understanding variables, data types, and basic operations.
    - Writing and running your first Python script.

Module 2: Core Python Concepts

- **Lesson 2.1: Control Structures**
    - **Conditional Statements:**
        - if, else, and elif statements with examples relevant to bioinformatics.
    - **Loops:**
        - for and while loops, with examples like iterating over sequence data or lists of gene names.
        - Introduction to list comprehensions for efficient looping.
- **Lesson 2.2: Functions and Modules**
    - **Writing Functions:**
        - Defining functions, using parameters, return statements.
        - Importance of reusable code, with examples of bioinformatics functions (e.g., calculating GC content).
    - **Using Modules:**
        - Importing and using standard Python libraries (e.g., math, os).
        - Writing and importing custom modules for bioinformatics tasks.

- **Lesson 2.3: Working with Data Structures**
    - **Lists, Tuples, and Dictionaries:**
        - Creating, accessing, and modifying lists, tuples, and dictionaries.
        - Practical examples: storing sequences, gene annotation data, or feature sets.
    - **Basic Operations on Data Structures:**
        - Sorting, slicing, filtering, and iterating over data structures.
        - Examples: Filtering genes based on expression levels, working with dictionaries to map gene IDs to descriptions.

Module 3: Handling Biological Data

- **Lesson 3.1: Introduction to Biological Data Formats**
    - Overview of key biological data formats: FASTA, FASTQ, GenBank, GFF, VCF.
    - Understanding the structure and content of these formats.
    - Reading and writing biological data files using Python.
- **Lesson 3.2: String Manipulation for Sequence Analysis**
    - Working with DNA, RNA, and protein sequences as strings.
    - Basic operations: reverse complement, transcription (DNA to RNA), translation (RNA to protein).
    - Finding orfs and introns in sequences using Python's string methods and regular expressions.
- **Lesson 3.3: Parsing and Analyzing Sequence Data**
    - Parsing FASTA/FASTQ files to extract sequence information.
    - Basic sequence analysis tasks: calculating counting nucleotides, GC content etc.
    - Read and handling quality information in FASTQ file.

Module 4: Working with Bioinformatics Libraries

- **Lesson 4.1: Introduction to Biopython**
    - Overview of Biopython and its key modules for sequence analysis, structure analysis, and more.
    - Reading, writing, and manipulating sequence data using Biopython.
    - Practical examples: reading different file formats (e.g., FASTA to GenBank).

- **Lesson 4.2: Sequence Alignments**
    - **Pairwise Sequence Alignment:**
        - Introduction to global and local alignments.
        - Performing pairwise alignments using Biopython's Align module.
    - **Multiple Sequence Alignment:**
        - Overview of multiple sequence alignment and its applications.
        - Using tools like ClustalW or MUSCLE with Biopython integration.
- **Lesson 4.3: Handling Genomic Data**
    - Working with GenBank files: parsing and extracting features, annotations, and sequences.
    - Understanding the structure and content of GenBank files.

Module 5: Data Analysis and Visualization

- **Lesson 5.1: Introduction to NumPy and Pandas**
    - **Fundamentals of NumPy:**
        - Creating and manipulating arrays, performing mathematical operations.
        - Applying NumPy for matrix operations relevant to bioinformatics (e.g., working with gene expression matrices).
    - **Fundamentals of Pandas:**
        - Introduction to Pandas DataFrames for handling tabular data.
        - Loading, cleaning, and analyzing datasets such as gene expression data or SNP data.
- **Lesson 5.2: Introduction to Matplotlib and Seaborn**
    - **Fundamentals of Matplotlib:**
        - Creating basic plots: line plots, scatter plots, histograms.
        - Customizing plots: adding titles, labels, legends, and color schemes.
    - **Advanced Visualization with Seaborn:**
        - Creating box plots, violin plots, and heatmaps.
        - Visualizing complex datasets like expression data or genotype-phenotype associations.

- **Lesson 5.3: Differential Gene Expression Analysis**
    - **Loading and Preprocessing Gene Expression Data:**
        - Loading gene expression data (e.g., from RNA-Seq) into Pandas DataFrames.
        - Normalizing data and handling batch effects.
    - **Calculating Differential Expression:**
        - Calculating base mean, log2 fold change, p-values using Python (e.g., using statsmodels).
    - **Visualizing Results:**
        - Plotting volcano plots to highlight differentially expressed genes.
        - Creating heatmaps to visualize expression patterns across samples.

Module 6: Automating Bioinformatics Tasks

- **Lesson 6.1: Retrieving Data from Online Databases**
    - **NCBI:**
        - Using Biopython's Entrez module to query and retrieve data from NCBI databases.
        - Downloading sequence data, gene annotations, and literature references.
    - **Ensembl:**
        - Accessing Ensembl data using RESTful APIs or FTP.
        - Fetching genomic coordinates, sequence variants, and regulatory elements.
- **Lesson 6.2: Integrating Python with Command-Line Tools**
    - **BLAST:**
        - Running BLAST searches from within Python using subprocess.
        - Parsing and analyzing BLAST results programmatically.
    - **FastQC:**
        - Automating quality control for sequencing data using FastQC.
        - Parsing FastQC reports to identify quality issues in FASTQ files.
    - **Trimmomatic:**
        - Automating sequence trimming for NGS data using Trimmomatic.
        - Integrating Trimmomatic with other tools in a preprocessing pipeline.

Module 7: Capstone Project

- **Lesson 7.1: Project Introduction and Planning**
    - Defining a real-world bioinformatics problem.
    - Planning the project scope and objectives.
- **Lesson 7.2: Project Implementation**
    - Applying Python skills to solve the bioinformatics problem.
    - Writing code and analyzing results.
- **Lesson 7.3: Project Presentation and Review**
    - Presenting the project findings.
    - Code review and feedback.

Learning Outcomes

- **Core Python Programming**: Automate workflows and address bioinformatics challenges using Python's control structures, functions, and data structures effectively.
- **Handling Biological Data**:
    - **FASTA**: Analyze nucleotide and protein sequences, including computing composition, generating complements, transcription, translation, ORFs, motifs, and performing pairwise alignments.
    - **FASTQ**: Assess read quality scores, visualize data with box plots and GC content plots.
    - **GenBank**: Extract and analyze gene annotations and sequence features from GenBank files.
    - **PDB**: Visualize protein secondary structures.
    - **GFF/GTF**: Parse and analyze genomic features using Pandas.
    - **Phylogenetic Trees**: Construct and visualize trees from Newick and CLUSTAL formats.
    - **Count Tables**: Conduct differential expression analysis and generate volcano plots and heatmaps.
- **Data Analysis and Visualization**: Utilize NumPy, Pandas, Matplotlib, and Seaborn for comprehensive data analysis and visualization, including differential gene expression studies.

- **Automating Workflows**: Retrieve and manage data from online databases (NCBI, Ensembl), and automate bioinformatics tasks such as quality control and data trimming using command-line tools like BLAST, FastQC, and Trimmomatic.
- **Project Application**: Design and implement a bioinformatics project, analyze results, and present findings, showcasing practical Python skills and their application to real-world bioinformatics problems.