Nova Tech Internship Task Schedule

Domain:	Web-Developement	Python Programming	Java Programming
Mentor:	Gowtham.K	Tamilselvan.M	Jayabarathi.J.S
No of sessions:	4 sessions (4 weeks)	4 sessions (4 weeks)	4 sessions (4 weeks)
No of Students:	19	11	4

Web-Developement (Schedule)

Week 1: Introduction to Web Development

Task 1: Basic HTML and CSS
Learn the fundamentals of HTML (HyperText Markup Language) and CSS (Cascading Style Sheets).
Complete simple exercises to create static web pages.
Resources: Online tutorials, code editors (e.g., VS Code, Sublime Text).
Week 2: Frontend Development

Task 2: Responsive Web Design

Understand the principles of responsive design using media queries and flexible layouts.
Develop a responsive webpage that adapts to different screen sizes.
Resources: Responsive design tutorials, CSS frameworks like Bootstrap or Flexbox/Grid.
Week 3: Backend Development

Task 3: Introduction to JavaScript and DOM Manipulation

Learn the basics of JavaScript programming language.
Practice DOM (Document Object Model) manipulation to dynamically update webpage content.
Build interactive elements like forms, buttons, and simple animations.
Resources: JavaScript tutorials, MDN web docs, online coding challenges.

Week 4: Advanced Concepts and Project

Task 4: Full Stack Web Development Project
Combine frontend and backend skills to develop a complete web application.
Utilize frameworks like React.js for frontend and Node.js or Django for backend.
Implement features such as user authentication, data storage, and API integration.
Deploy the project on a cloud platform like Heroku or Netlify.
Resources: Full-stack development tutorials, project-based learning resources, version control (Git).

Python Programing (Schedule)

Week 1: Introduction to Python Basics

Day 1-2: Introduction to Python Syntax (Beginner)
Task: Install Python and write simple programs to understand basic syntax, variables, and data types.
Day 3-4: Control Structures (Beginner)
Task: Learn about conditional statements (if, elif, else) and loops (for, while) in Python. Write programs to practice these concepts.
Day 5-7: Functions and Modules (Beginner)
Task: Understand the concept of functions and modular programming in Python. Write functions to perform simple tasks and organize code into modules.

Week 2: Intermediate Python Programming

Day 8-9: Working with Lists (Intermediate)
Task: Learn about lists in Python and practice operations like appending, slicing, and iterating over lists.
Day 10-11: Working with Strings (Intermediate)
Task: Explore string manipulation techniques such as concatenation, slicing, and formatting.
Day 12-14: File Handling (Intermediate)
Task: Understand file input/output operations in Python. Write programs to read from and write to files.

Week 3: Data Structures and Algorithms in Python

Day 15-16: Introduction to Data Structures (Intermediate)
Task: Learn about more advanced data structures such as tuples, sets, and dictionaries in Python.
Day 17-18: Searching and Sorting Algorithms (Intermediate)
Task: Implement searching and sorting algorithms (e.g., linear search, binary search, bubble sort, merge sort) in Python.
Day 19-21: Object-Oriented Programming (OOP) in Python (Intermediate)
Task: Understand the principles of OOP in Python, including classes, objects, inheritance, and polymorphism. Implement simple class hierarchies and objects.

Week 4: Advanced Python Concepts and Project

Day 22-23: Exception Handling and Debugging (Advanced)
Task: Learn about exception handling and debugging techniques in Python. Write programs to handle exceptions gracefully.
Day 24-25: Generators and Decorators (Advanced)
Task: Explore advanced Python concepts such as generators and decorators. Write programs to implement and use them.
Day 26-28: Final Project Development (Advanced)
Task: Work on a final project that integrates concepts learned throughout the internship. The project should be a Python application or script that solves a real-world problem or demonstrates a particular skill set.
Day 29-30: Project Presentation and Assessment
Task: Present the final project to mentors and peers, explaining the design choices, implementation details, and challenges faced. Conduct assessments to evaluate interns' understanding and skills in Python programming.

Java Programming (Schedule)

Week 1: Introduction to Java Basics

Day 1-2: Introduction to Java Syntax (Beginner)
Task: Write simple Java programs to understand variables, data types, and basic syntax.
Day 3-4: Control Flow and Loops (Beginner)
Task: Implement conditional statements (if-else) and loops (for, while) in Java.
Day 5-7: Methods and Functions (Beginner)
Task: Learn about methods in Java and create reusable code blocks.

Week 2: Object-Oriented Programming (OOP) Fundamentals

Day 8-9: Classes and Objects (Beginner)
Task: Define classes and create objects to understand the principles of OOP.
Day 10-11: Encapsulation and Access Modifiers (Intermediate)
Task: Implement encapsulation and use access modifiers (public, private, protected) in Java.
Day 12-14: Inheritance and Polymorphism (Intermediate)
Task: Explore inheritance and polymorphism concepts by creating class hierarchies and overriding methods.

Week 3: Data Structures and Algorithms

Day 15-16: Arrays and ArrayLists (Intermediate)
Task: Implement basic data structures like arrays and ArrayLists in Java.
Day 17-18: Linked Lists and Stacks (Intermediate)
Task: Implement linked lists and stacks data structures in Java.
Day 19-21: Sorting and Searching Algorithms (Intermediate)
Task: Implement common sorting algorithms (e.g., bubble sort, insertion sort) and searching algorithms (e.g., binary search) in Java.

Week 4: Advanced Java Concepts

Day 22-23: Exception Handling (Intermediate)
Task: Learn about exception handling in Java and implement try-catch blocks.
Day 24-25: File Handling and Input/Output (Intermediate)
Task: Work with file input/output streams and handle file operations in Java.
Day 26-28: Multithreading and Concurrency (Advanced)
Task: Learn about multithreading concepts and implement concurrent programs using threads and synchronization.
Day 29-30: Final Project and Assessment
Task: Work on a final project that combines concepts learned throughout the internship, such as implementing a small application or solving a programming challenge. Present the project and undergo assessment to evaluate understanding and proficiency in Java programming.