100 Days Comprehensive Guide On Learning Programming languages like

Subscribe to my youtube channel to learn for free using beginner friendly tutoarials:
Click here to join

Days 1-10: Python Basics

Day 1: Introduction to Python, Installation, and Setup

Topics: Python overview, installing Python, setting up an IDE.

Project: Set up Python on your computer and write a simple "Hello, World!" program.

Day 2: Basic Syntax and Variables

Topics: Variables, data types, and basic input/output.

Project: Create a program that asks for the user's name and age, then prints a greeting.

Day 3: Basic Operations

Topics: Arithmetic operations, comparison operators, logical operators.

Project: Create a calculator that performs basic arithmetic operations.

Day 4: Strings and String Manipulation

Topics: String operations, slicing, formatting.

Project: Create a program that reverses a string and changes its case.

Day 5: Lists and Tuples

Topics: List operations, tuples, indexing, slicing.

Project: Build a program that manages a list of tasks.

Day 6: Dictionaries and Sets

Topics: Dictionary operations, sets, and their uses.

Project: Create a contact book application using dictionaries.

Day 7: Conditional Statements

Topics: if, elif, else statements.

Project: Develop a simple quiz program with conditional logic.

Day 8: Loops

Topics: For loops, while loops, and loop control statements.

Project: Create a program that prints multiplication tables.

Day 9: Functions

Topics: Defining functions, arguments, return values.

Project: Build a program to calculate the factorial of a number using a function.

Day 10: Review and Mini-Project

Project: Create a program that combines loops, conditionals, and functions to solve a small problem, such as a number guessing game.

Days 11-20: Intermediate Python

Day 11: File Handling

Topics: Reading and writing files, working with text and CSV files.

Project: Create a program that reads a text file and counts the frequency of each word.

Day 12: Error Handling

Topics: Try, except blocks, handling different exceptions.

Project: Create a program that handles user input errors gracefully.

Day 13: Lists Comprehensions and Generators

Topics: List comprehensions, generators, and their use cases.

Project: Build a program that generates Fibonacci numbers up to a certain limit.

Day 14: Modules and Packages

Topics: Importing modules, creating your own modules.

Project: Create a simple utility module and use it in a project.

Day 15: Object-Oriented Programming (OOP) Basics

Topics: Classes, objects, methods, attributes.

Project: Build a simple class to represent a bank account.

Day 16: OOP - Inheritance and Polymorphism

Topics: Inheritance, method overriding, polymorphism.

Project: Create a class hierarchy for different types of vehicles.

Day 17: OOP - Encapsulation and Abstraction

Topics: Encapsulation, getters, setters, abstraction.

Project: Implement a library management system with encapsulated classes.

Day 18: Working with Dates and Time

Topics: The datetime module, working with dates and times.

Project: Create a program that logs events with timestamps.

Day 19: Lambda Functions and Functional Programming

Topics: Lambda functions, map, filter, reduce.

Project: Build a program that processes a list of numbers using functional programming techniques.

Day 20: Review and Mini-Project

Project: Develop a small CLI application, such as a to-do list manager, that utilizes OOP principles.

Days 21-30: Advanced Python Concepts

Day 21: Decorators and Context Managers

Topics: Function decorators, context managers, the `with` statement.

Project: Create a decorator to measure the execution time of functions.

Day 22: Iterators and Generators

Topics: Iterators, custom iterators, generator functions.

Project: Build a custom iterator for a sequence of numbers.

Day 23: Regular Expressions

Topics: Regex syntax, matching, searching, replacing.

Project: Create a program to validate email addresses using regular expressions.

Day 24: Working with JSON and APIs

Topics: JSON data, parsing JSON, working with APIs.

Project: Create a program that fetches data from a public API and processes it.

Day 25: Web Scraping Basics

Topics: Introduction to web scraping, using BeautifulSoup and requests.

Project: Scrape data from a website and save it in a CSV file.

Day 26: Working with Databases

Topics: SQLite basics, CRUD operations.

Project: Build a simple database to manage a list of books.

Day 27: Multithreading and Multiprocessing

Topics: Concurrency, threading, multiprocessing.

Project: Create a program that downloads multiple files concurrently.

Day 28: Working with Data: Numpy and Pandas Basics

Topics: Introduction to Numpy, Pandas for data manipulation.

Project: Analyze a dataset using Pandas.

Day 29: Introduction to Data Visualization

Topics: Matplotlib basics, plotting data.

Project: Create visualizations from a dataset using Matplotlib.

Day 30: Review and Mini-Project

Project: Develop a small data analysis project using Numpy, Pandas, and Matplotlib.

Days 31-50: Web Development with Python

Day 31: Introduction to Flask

Topics: Setting up Flask, creating a simple web app.

Project: Build a basic web page with Flask.

Day 32: Flask Routing and Templates

Topics: URL routing, rendering HTML templates.

Project: Create a multi-page website using Flask.

Day 33: Flask Forms and User Input

Topics: Handling forms, processing user input.

Project: Build a simple registration form.

Day 34: Flask and Databases

Topics: Integrating a database with Flask, SQLAlchemy basics.

Project: Develop a blog application with a database backend.

Day 35: User Authentication in Flask

Topics: User authentication, creating login and registration pages.

Project: Add user authentication to the blog application.

Day 36: Flask Extensions and Blueprints

Topics: Using Flask extensions, organizing your app with Blueprints.

Project: Refactor the blog application to use Blueprints.

Day 37: Introduction to Django

Topics: Setting up Django, understanding the Django project structure.

Project: Start a simple Django project and create a basic app.

Day 38: Django Models and Admin Interface

Topics: Creating models, working with the Django admin interface.

Project: Develop a product catalog with models and the admin interface.

Day 39: Django Views and Templates

Topics: Creating views, rendering templates, working with URLs.

Project: Build a product detail and listing page.

Day 40: Django Forms and User Input

Topics: Handling forms, validating user input.

Project: Create a form for adding new products to the catalog.

Day 41: Django Authentication

Topics: Implementing authentication, login, and registration.

Project: Add user authentication to your Django project.

Day 42: Django REST Framework Basics

Topics: Introduction to DRF, creating APIs.

Project: Build a simple REST API for your product catalog.

Day 43: Deploying a Flask App

Topics: Deployment basics, using platforms like Heroku or AWS.

Project: Deploy your Flask blog application to a live server.

Day 44: Deploying a Django App

Topics: Deployment basics, using platforms like Heroku or AWS.

Project: Deploy your Django project to a live server.

Day 45: Review and Web Development Mini-Project

Project: Develop a small e-commerce site or a similar web project that integrates both frontend and backend elements.

Days 51-70: Data Science and Machine Learning

Day 51: Introduction to Data Science

Topics: Overview of data science, data types, and data processing.

Project: Explore a dataset and perform basic cleaning and processing.

Day 52: Numpy and Pandas in Data Science

Topics: Advanced Numpy and Pandas techniques.

Project: Perform exploratory data analysis on a real dataset.

Day 53: Introduction to Machine Learning

Topics: Overview of machine learning, types of ML, basic concepts.

Project: Implement a simple linear regression model from scratch.

Day 54: Supervised Learning with Scikit-Learn

Topics: Introduction to Scikit-Learn, training a supervised model.

Project: Build a classification model using Scikit-Learn.

Day 55: Unsupervised Learning with Scikit-Learn

Topics: Clustering, dimensionality reduction techniques.

Project: Apply K-means clustering to a dataset.

Day 56: Data Preprocessing and Feature Engineering

Topics: Data cleaning, feature selection, scaling.

Project: Prepare a dataset for machine learning, performing feature engineering.

Day 57: Model Evaluation and Improvement

Topics: Cross-validation, hyperparameter tuning.

Project: Optimize a machine learning model using grid search.

Day 58: Introduction to Neural Networks

Topics: Basics of neural networks, forward propagation.

Project: Build a simple neural network using TensorFlow or PyTorch.

Day 59: Deep Learning with TensorFlow

Topics: Introduction to TensorFlow, building deep learning models.

Project: Create a deep learning model for image classification.

Day 60: Natural Language Processing Basics

Topics: Text processing, tokenization, basic NLP tasks.

Project: Build a sentiment analysis model.

Day 61: Data Visualization with Seaborn and Matplotlib

Topics: Advanced plotting techniques, customizing plots.

Project: Visualize the results of your machine learning models.

Day 62: Time Series Analysis

Topics: Working with time series data, forecasting.

Project: Perform time series analysis on stock market data.

Day 63: Introduction to Big Data with PySpark

Topics: Overview of big data, introduction to PySpark.

Project: Analyze a large dataset using PySpark.

Day 64: Machine Learning Deployment

Topics: Deploying ML models with Flask, creating APIs.

Project: Deploy a machine learning model as a web service.

Day 65: Review and Data Science Mini-Project

Project: Develop a comprehensive data science project, such as a recommendation system or a predictive model.

Days 71-100: Advanced Topics and Final Projects

Day 71: Introduction to Computer Vision

Topics: Image processing basics, computer vision with OpenCV.

Project: Build a face detection application using OpenCV.

Day 72: Advanced Python Libraries and Tools

Topics: Overview of popular Python libraries, tools, and best practices.

Project: Explore and implement a small project using a less common library (e.g., Dask, Numba).

Day 73: Web Scraping with Selenium

Topics: Using Selenium for more complex web scraping tasks.

Project: Automate the scraping of a dynamic website.

Day 74: Working with Real-Time Data

Topics: Streaming data, working with WebSockets.

Project: Build a real-time data processing pipeline.

Day 75: Advanced Django: Middleware, Signals, and Custom Management Commands

Topics: Custom middleware, signals, and commands in Django.

Project: Implement advanced features in your Django project.

Day 76: Working with Cloud Services

Topics: Introduction to AWS, Google Cloud, or Azure with Python.

Project: Deploy a Python application on a cloud platform.

Day 77: Automation with Python

Topics: Automating tasks with Python scripts, scheduling.

Project: Create a script that automates a routine task, like data backup.

Day 78: Advanced OOP Concepts

Topics: Design patterns, SOLID principles.

Project: Refactor an existing project to use advanced OOP principles.

Day 79: Introduction to Web Development Frameworks

Topics: Overview of other Python web frameworks (FastAPI, Pyramid).

Project: Create a small project using a different web framework.

Day 80: Building and Distributing Python Packages

Topics: Creating a Python package, using PyPI.

Project: Package one of your Python projects and publish it to PyPI.

Day 81: Final Project Planning

Project: Plan your final project. Decide on the scope, technologies, and timeline.

Day 82-90: Final Project Development

Project: Work on your final project. This could be a web application, a data science project, or an automation tool that integrates multiple concepts learned.

Day 91-95: Final Project Testing and Debugging

Project: Test your project thoroughly, debug any issues, and refine the code.

Day 96-98: Final Project Documentation and Presentation

Project: Document your project, write a README file, and prepare a presentation.

Day 99: Final Project Review and Polishing

Project: Review your project, make final adjustments, and ensure everything is in order.

Day 100: Project Presentation and Reflection - **Project:** Present your final project to others, reflect on what you've learned, and plan your next steps in your Python journey.

Days 1-10: Java Basics

Day 1: Introduction to Java, Installation, and Setup

Topics: Java overview, setting up JDK and an IDE (e.g., IntelliJ IDEA, Eclipse).

Project: Write and run a simple "Hello, World!" program.

Day 2: Basic Syntax and Variables

Topics: Variables, data types, basic input/output.

Project: Create a program that asks for the user's name and age, then prints a greeting.

Day 3: Basic Operations

Topics: Arithmetic operations, comparison operators, logical operators.

Project: Create a calculator that performs basic arithmetic operations.

Day 4: Strings and String Manipulation

Topics: String class, common string methods, immutability.

Project: Create a program that reverses a string and changes its case.

Day 5: Arrays

Topics: Arrays, array operations, multidimensional arrays.

Project: Develop a program that manages an array of numbers, allowing the user to add, remove, and search for elements.

Day 6: Conditional Statements

Topics: if, else if, else statements, switch-case.

Project: Create a simple quiz program with conditional logic.

Day 7: Loops

Topics: For loops, while loops, do-while loops, break and continue.

Project: Develop a program that prints multiplication tables.

Day 8: Methods (Functions) in Java

Topics: Defining methods, method arguments, return values, method overloading.

Project: Build a program to calculate the factorial of a number using a method.

Day 9: Basic Input and Output with Scanners

Topics: Using the Scanner class for user input, basic I/O operations.

Project: Create a program that reads user input and processes it.

Day 10: Review and Mini-Project

Project: Combine loops, conditionals, and methods to develop a small game, like a number guessing game.

Days 11-20: Intermediate Java

Day 11: Object-Oriented Programming (OOP) Basics

Topics: Classes, objects, methods, attributes.

Project: Create a simple class to represent a bank account.

Day 12: Constructors and Overloading

> **Topics:** Constructors, constructor overloading.
>
> **Project:** Expand the bank account class to initialize with different types of data.

Day 13: Inheritance in Java

> **Topics:** Inheritance, the `extends` keyword, method overriding.
>
> **Project:** Create a class hierarchy for different types of employees (e.g., Manager, Engineer).

Day 14: Polymorphism and Dynamic Method Dispatch

> **Topics:** Polymorphism, method overriding, dynamic method dispatch.
>
> **Project:** Implement polymorphism in the employee hierarchy.

Day 15: Encapsulation and Access Modifiers

> **Topics:** Encapsulation, private and public access modifiers, getters and setters.
>
> **Project:** Refactor the bank account class to use encapsulation principles.

Day 16: Abstract Classes and Interfaces

> **Topics:** Abstract classes, interfaces, multiple inheritance in Java.
>
> **Project:** Create an interface for a payment system and implement it in different classes.

Day 17: Working with Packages and Access Control

> **Topics:** Creating and using packages, understanding package-level access.
>
> **Project:** Organize a small project into different packages for better structure.

Day 18: Static Members and the Singleton Pattern

> **Topics:** Static variables, static methods, the `final` keyword, Singleton design pattern.
>
> **Project:** Implement a Singleton class to manage a global configuration.

Day 19: Exception Handling

> **Topics:** Try-catch blocks, throwing exceptions, creating custom

exceptions.

Project: Create a program that handles different types of user input errors.

Day 20: Review and Mini-Project

Project: Develop a small application (e.g., a simple library system) that integrates OOP concepts, encapsulation, and exception handling.

Days 21-30: Java Collections Framework

Day 21: Introduction to Collections

Topics: Overview of the Java Collections Framework, List, ArrayList.

Project: Create a program that manages a list of tasks using ArrayList.

Day 22: Sets and HashSet

Topics: Set interface, HashSet, TreeSet, uniqueness.

Project: Build a program that tracks unique items using a Set.

Day 23: Maps and HashMap

Topics: Map interface, HashMap, TreeMap.

Project: Implement a simple contact book using HashMap.

Day 24: LinkedList and Iterators

Topics: LinkedList, ListIterator, differences between ArrayList and LinkedList.

Project: Develop a program that simulates a waiting list using LinkedList.

Day 25: Queue and Stack

Topics: Queue interface, LinkedList as Queue, Stack class.

Project: Create a program that simulates undo/redo functionality using Stack.

Day 26: Collections Utility Methods

Topics: Collections utility class, sorting, searching, and shuffling collections.

Project: Implement a program that sorts and searches a list of students.

Day 27: Generic Classes and Methods

Topics: Generics, creating generic classes and methods.

Project: Create a generic class to manage a pair of objects.

Day 28: Comparable and Comparator

Topics: Implementing `Comparable`, using `Comparator` for custom sorting.

Project: Sort a list of employees by different criteria using Comparable and Comparator.

Day 29: Multithreading Basics

Topics: Creating threads, Runnable interface, thread life cycle.

Project: Create a simple multithreaded program that prints numbers in parallel.

Day 30: Review and Mini-Project

Project: Develop a small inventory management system that uses collections and multithreading.

Days 31-50: Advanced Java Concepts

Day 31: File Handling in Java

Topics: Reading and writing files, FileReader, FileWriter, BufferedReader, BufferedWriter.

Project: Create a program that reads a file and counts the frequency of words.

Day 32: Serialization and Deserialization

Topics: Serializable interface, ObjectOutputStream, ObjectInputStream.

Project: Build a program to save and load the state of an object to/from a file.

Day 33: Working with Dates and Time

Topics: Date class, Calendar class, LocalDate, LocalTime, and DateTimeFormatter.

Project: Develop a program that schedules events and displays them in different formats.

Day 34: Java Streams API

Topics: Streams, filter, map, reduce, and collectors.

Project: Process a collection of data using Java Streams.

Day 35: Regular Expressions in Java

Topics: Regex patterns, Matcher and Pattern classes.

Project: Create a program that validates email addresses using regular expressions.

Day 36: Reflection API

Topics: Introduction to Reflection, using Reflection to inspect classes at runtime.

Project: Build a program that dynamically loads and inspects a class's methods and fields.

Day 37: Annotations and Meta-Programming

Topics: Custom annotations, processing annotations.

Project: Create custom annotations and apply them in a project.

Day 38: Introduction to JavaFX

Topics: Setting up JavaFX, basic GUI development.

Project: Create a simple JavaFX application with a basic user interface.

Day 39: JavaFX Controls and Layouts

Topics: JavaFX controls (Buttons, TextFields), layouts (VBox, HBox, GridPane).

Project: Build a basic calculator using JavaFX.

Day 40: Event Handling in JavaFX

Topics: Handling events, event listeners, and handlers.

Project: Expand the JavaFX calculator with event handling for all buttons.

Day 41: Multithreading and Concurrency in Java

Topics: Concurrency utilities, Thread pools, Executors.

Project: Develop a program that performs parallel processing using a thread pool.

Day 42: Networking Basics in Java

Topics: Sockets, ServerSocket, and basic networking concepts.

Project: Create a simple client-server chat application.

Day 43: Java RMI (Remote Method Invocation)

Topics: Introduction to RMI, setting up an RMI server and client.

Project: Implement a remote calculator using RMI.

Day 44: Java NIO (New I/O)

Topics: Overview of NIO, Channels, Buffers, Selectors.

Project: Develop a program to efficiently read and write large files using NIO.

Day 45: Review and Advanced Mini-Project

Project: Create a multi-threaded file search utility that uses advanced file handling and concurrency features.

Days 51-70: Web Development with Java

Day 51: Introduction to Web Development with Java

Topics: Overview of web development, setting up a Java web project.

Project: Create a simple servlet that responds with "Hello, World!".

Day 52: Servlets and JSP Basics

Topics: Introduction to Servlets, JavaServer Pages (JSP), handling requests and responses.

Project: Develop a basic web form that submits data to a servlet and displays a response.

Day 53: Working with Sessions and Cookies

Topics: Session management, using cookies in web applications.

Project: Create a user login system that uses sessions to manage state.

Day 54: JavaServer Pages (JSP) and JSTL

Topics: Using JSP, JSTL tags, MVC pattern with JSP.

Project: Build a simple blog application using JSP and Servlets.

Day 55: Introduction to Spring Framework

Topics: Overview of Spring, setting up a Spring project, dependency injection.

Project: Create a basic Spring application with dependency injection.

Day 56: Spring Boot Basics

Topics: Introduction to Spring Boot, creating Spring Boot applications.

Project: Develop a simple RESTful API using Spring Boot.

Day 57: Spring Boot RESTful Web Services

Topics: Building REST APIs, handling HTTP requests.

Project: Create a CRUD API for managing a list of products.

Day 58: Spring Data JPA

Topics: Introduction to Spring Data JPA, working with databases.

Project: Build a REST API that performs CRUD operations on a database using Spring Data JPA.

Day 59: Spring Security Basics

Topics: Introduction to Spring Security, implementing basic authentication and authorization.

Project: Add security features to the product management API.

Day 60: Building Web Applications with Spring MVC

Topics: Overview of Spring MVC, building web applications.

Project: Develop a simple e-commerce site using Spring MVC.

Day 61: Thymeleaf in Spring Boot

Topics: Introduction to Thymeleaf, integrating Thymeleaf with Spring Boot.

Project: Use Thymeleaf to create dynamic views in your Spring MVC application.

Day 62: Working with Databases in Java

Topics: JDBC, database connections, executing SQL queries.

Project: Develop a program that interacts with a database using

JDBC.

Day 63: Hibernate ORM Basics

Topics: Introduction to Hibernate, mapping entities, Hibernate queries.

Project: Create a simple application that uses Hibernate to manage data in a relational database.

Day 64: Deploying Java Web Applications

Topics: Packaging and deploying Java web applications, using Tomcat or other servers.

Project: Deploy your Spring Boot application to a server.

Day 65: Review and Web Development Mini-Project

Project: Develop a complete web application, such as a small e-commerce platform, integrating Spring Boot, Spring Data JPA, and Spring Security.

Days 71-90: Advanced Java Topics

Day 71: Introduction to Microservices with Spring Boot

Topics: Overview of microservices, building microservices with Spring Boot.

Project: Develop a simple microservice for managing user accounts.

Day 72: Working with RESTful APIs

Topics: Advanced REST concepts, API versioning, documentation.

Project: Extend your microservice to include advanced REST features.

Day 73: Working with JSON and XML

Topics: JSON processing with Jackson, XML processing.

Project: Build an API that supports both JSON and XML data formats.

Day 74: Advanced Spring Security

Topics: Role-based access control, JWT authentication.

Project: Add JWT-based security to your microservice.

Day 75: Spring Boot Actuator and Monitoring

Topics: Using Spring Boot Actuator, monitoring application health.

Project: Implement monitoring and health checks in your microservice.

Day 76: Introduction to Cloud Services

Topics: Overview of cloud computing, deploying Java applications to the cloud.

Project: Deploy your Spring Boot microservice to AWS or Google Cloud.

Day 77: Java and NoSQL Databases

Topics: Introduction to NoSQL, using MongoDB with Java.

Project: Develop a microservice that interacts with a MongoDB database.

Day 78: Reactive Programming with Spring WebFlux

Topics: Introduction to reactive programming, using Spring WebFlux.

Project: Create a reactive REST API using Spring WebFlux.

Day 79: Introduction to Apache Kafka

Topics: Overview of Kafka, producing and consuming messages with Java.

Project: Implement a messaging system using Kafka and Spring Boot.

Day 80: Working with Docker and Kubernetes

Topics: Containerizing Java applications, deploying with Kubernetes.

Project: Containerize your Spring Boot application and deploy it using Kubernetes.

Day 81: Final Project Planning

Project: Plan your final project. Decide on the scope, technologies, and timeline.

Day 82-90: Final Project Development

Project: Work on your final project. This could be a microservice-based architecture, a full-stack web application, or a cloud-based solution that integrates various technologies learned.

Days 91-100: Final Project and Advanced Topics

Day 91-95: Final Project Testing and Debugging

Project: Test your project thoroughly, debug any issues, and refine the code.

Day 96-98: Final Project Documentation and Presentation

Project: Document your project, write a README file, and prepare a presentation.

Day 99: Final Project Review and Polishing

Project: Review your project, make final adjustments, and ensure everything is in order.

Day 100: Project Presentation and Reflection - **Project:** Present your final project to others, reflect on what you've learned, and plan your next steps in your Java journey.

Days 1-10: JavaScript Basics

Day 1: Introduction to JavaScript and Setup

Topics: JavaScript overview, setting up the environment (browser, editor).

Project: Write and run a simple "Hello, World!" script in the browser.

Day 2: Basic Syntax and Variables

Topics: Variables (var, let, const), data types, basic operators.

Project: Create a script that asks for the user's name and displays a personalized greeting.

Day 3: Basic Operations

Topics: Arithmetic operations, comparison operators, logical operators.

Project: Build a simple calculator that performs basic arithmetic operations.

Day 4: Strings and String Manipulation

Topics: String methods (concat, slice, substring, replace), template literals.

Project: Create a script that manipulates a string (e.g., reversing, converting to uppercase/lowercase).

Day 5: Arrays

Topics: Array creation, common methods (push, pop, shift, unshift, map, filter).

Project: Create a to-do list script that allows users to add and remove tasks.

Day 6: Conditional Statements

Topics: if, else if, else, switch-case.

Project: Develop a simple quiz program that uses conditional logic to check answers.

Day 7: Loops

Topics: for loop, while loop, do-while loop, forEach.

Project: Write a script that prints out multiplication tables up to a user-defined number.

Day 8: Functions

Topics: Function declaration, parameters, return values, arrow functions.

Project: Create a script that calculates the factorial of a number using a function.

Day 9: Objects and Arrays of Objects

Topics: Object literals, accessing properties, nested objects, arrays of objects.

Project: Develop a script that manages a list of students with their grades.

Day 10: Review and Mini-Project

Project: Combine arrays, loops, and functions to create a small game (e.g., a number guessing game).

Days 11-20: Intermediate JavaScript

Day 11: DOM Manipulation Basics

Topics: Selecting elements, manipulating content, adding/removing elements.

Project: Create a script that dynamically updates the content of a webpage (e.g., a real-time to-do list).

Day 12: Event Handling

Topics: Event listeners, event types (click, input, submit).

Project: Build a simple form that validates user input and displays the result on the page.

Day 13: Working with Forms

Topics: Form elements, getting and setting form values, form validation.

Project: Create a contact form with client-side validation.

Day 14: Advanced DOM Manipulation

Topics: Traversing the DOM, manipulating CSS styles, classes, and attributes.

Project: Build a script that dynamically applies themes to a webpage based on user selection.

Day 15: JSON and Data Storage

Topics: JSON format, parsing and stringifying, localStorage, sessionStorage.

Project: Develop a script that stores user preferences (e.g., theme, language) in localStorage.

Day 16: Functions as First-Class Citizens

Topics: Passing functions as arguments, returning functions, callbacks.

Project: Create a script that filters and sorts an array of objects using callback functions.

Day 17: Higher-Order Functions

Topics: Map, filter, reduce, forEach, find.

Project: Build a script that processes an array of numbers to calculate statistics (e.g., sum, average).

Day 18: Scope and Closures

Topics: Function scope, block scope, closures.

Project: Create a counter function that maintains its own private state using closures.

Day 19: The `this` Keyword and Arrow Functions

Topics: Understanding `this` in different contexts, arrow functions and `this`.

Project: Develop a simple object that uses methods and demonstrates how `this` works.

Day 20: Review and Mini-Project

Project: Build a small interactive quiz application that uses DOM manipulation, event handling, and localStorage.

Days 21-30: Advanced JavaScript Concepts

Day 21: Object-Oriented JavaScript

Topics: Creating objects with constructors, prototypes, and ES6 classes.

Project: Develop a simple class-based system, such as a basic product inventory management.

Day 22: ES6+ Features

Topics: Let and const, template literals, destructuring, spread/rest operators, default parameters.

Project: Refactor previous code to use modern ES6+ syntax and features.

Day 23: Asynchronous JavaScript: Callbacks

Topics: Asynchronous operations, callback functions, error handling in callbacks.

Project: Simulate an asynchronous operation (e.g., fetching data) using callbacks.

Day 24: Promises and Fetch API

Topics: Understanding promises, chaining, handling errors, using Fetch API.

Project: Build a script that fetches data from a public API and displays it on the webpage.

Day 25: Async/Await

Topics: Converting promises to async/await, error handling.

Project: Refactor the API fetching project to use async/await for cleaner code.

Day 26: Working with Modules

Topics: Import/export syntax, organizing code with modules.

Project: Split a larger JavaScript project into modules for better structure and reusability.

Day 27: Error Handling and Debugging

Topics: Try-catch, throwing custom errors, debugging tools.

Project: Add comprehensive error handling to an existing project, including custom error messages.

Day 28: Regular Expressions

Topics: Basic regex syntax, test, match, search, replace.

Project: Create a script that validates user input (e.g., email addresses, phone numbers) using regular expressions.

Day 29: Advanced Array Methods

Topics: Some, every, includes, flat, flatMap.

Project: Develop a script that performs advanced data manipulation on nested arrays.

Day 30: Review and Mini-Project

Project: Build a small web application (e.g., a weather app) that integrates several advanced concepts, such as async/await, API calls, and DOM manipulation.

Days 31-50: Working with JavaScript Frameworks and Libraries

Day 31: Introduction to jQuery

Topics: Selecting elements, event handling, DOM manipulation with jQuery.

Project: Refactor a previous project to use jQuery for DOM manipulation and event handling.

Day 32: jQuery Plugins

Topics: Using and customizing jQuery plugins.

Project: Implement a jQuery plugin (e.g., a carousel) in a web page.

Day 33: Introduction to React.js

Topics: Setting up a React environment, understanding components, JSX.

Project: Create a simple React component that displays user information.

Day 34: React State and Props

Topics: Managing state, passing data with props.

Project: Build a React app that tracks a list of tasks with state management.

Day 35: React Event Handling and Forms

Topics: Handling events, managing form inputs in React.

Project: Create a contact form in React that validates input and displays a summary.

Day 36: React Lifecycle Methods and Hooks

Topics: Component lifecycle, using hooks like useState and useEffect.

Project: Build a React app that fetches and displays data from an API, with loading and error states.

Day 37: Introduction to Vue.js

Topics: Setting up a Vue environment, Vue instances, data binding.

Project: Create a simple Vue component that displays a greeting message.

Day 38: Vue.js Directives and Events

Topics: v-bind, v-model, v-if, v-for, event handling.

Project: Build a Vue app that filters and sorts a list of items.

Day 39: Vue.js Components and Props

Topics: Creating components, passing data with props.

Project: Develop a small Vue application that uses multiple components to display user profiles.

Day 40: Introduction to Angular

Topics: Setting up an Angular environment, components, modules.

Project: Create a simple Angular component that displays a list of products.

Day 41: Angular Data Binding and Directives

Topics: Property binding, event binding, structural and attribute directives.

Project: Build an Angular app that displays and filters a list of users.

Day 42: Angular Services and Dependency Injection

Topics: Creating services, injecting services into components.

Project: Develop an Angular application that uses a service to fetch data from an API.

Day 43: Introduction to TypeScript

Topics: TypeScript basics, types, interfaces, and classes.

Project: Convert a simple JavaScript project to TypeScript.

Day 44: Using TypeScript with React

Topics: TypeScript with React, typing props and state.

Project: Build a React application using TypeScript for type safety.

Day 45: Review and Frameworks Mini-Project

Project: Choose a framework (React, Vue, Angular) to build a small web application (e.g., a simple e-commerce site).

Days 51-70: JavaScript for Web Development

Day 51: Introduction to Node.js

Topics: Setting up Node.js, understanding the event loop, Node.js modules.

Project: Create a simple Node.js script that reads and writes to a file.

Day 52: Working with npm and Package Management

Topics: Using npm, installing and managing packages, creating a package.json file.

Project: Set up a Node.js project and manage dependencies using npm.

Day 53: Building a Simple HTTP Server with Node.js

Topics: HTTP module, creating a basic server, handling requests and responses.

Project: Develop a basic web server that serves HTML files.

Day 54: Introduction to Express.js

Topics: Setting up an Express project, routing, middleware.

Project: Build a simple RESTful API using Express.js.

Day 55: Working with Databases in Node.js

Topics: Connecting to a database (e.g., MongoDB, MySQL) using Node.js.

Project: Create a RESTful API that performs CRUD operations on a database.

Day 56: Authentication and Authorization in Node.js

Topics: Using Passport.js for authentication, JWT tokens.

Project: Implement user authentication in a Node.js API.

Day 57: Real-Time Applications with WebSockets

Topics: Introduction to WebSockets, using Socket.io for real-time communication.

Project: Build a simple real-time chat application using Node.js and Socket.io.

Day 58: Working with APIs and RESTful Services

Topics: Creating RESTful services, API design best practices.

Project: Build a full-featured REST API with Express.js, including CRUD operations and user authentication.

Day 59: Introduction to GraphQL

Topics: Understanding GraphQL, setting up a GraphQL server, queries and mutations.

Project: Create a simple GraphQL API and test it with queries and mutations.

Day 60: Deploying JavaScript Applications

Topics: Deploying Node.js apps, using cloud platforms (e.g., Heroku, AWS).

Project: Deploy your Node.js API to a live server.

Day 61: Progressive Web Apps (PWA)

Topics: What are PWAs, service workers, manifest files.

Project: Convert an existing web application into a Progressive Web App.

Day 62: Introduction to WebAssembly

Topics: Understanding WebAssembly, setting up a WebAssembly project.

Project: Create a simple WebAssembly module and integrate it with JavaScript.

Day 63: JavaScript Performance Optimization

Topics: Analyzing performance, optimizing code, reducing load times.

Project: Optimize the performance of an existing JavaScript project.

Day 64: Working with Webpack and Babel

Topics: Setting up Webpack, using Babel for transpiling.

Project: Set up a modern JavaScript project with Webpack and Babel for ES6+ support.

Day 65: Review and Full-Stack Mini-Project

Project: Build a full-stack application (e.g., a task manager) using Node.js, Express, a database, and a frontend framework (React, Vue, or Angular).

Days 71-90: Advanced JavaScript Topics

Day 71: Introduction to TypeScript Advanced Concepts

Topics: Generics, advanced types, decorators.

Project: Extend an existing TypeScript project with advanced features.

Day 72: JavaScript Design Patterns

Topics: Common design patterns (Singleton, Observer, Factory).

Project: Implement a design pattern in a JavaScript project.

Day 73: JavaScript and WebAssembly Advanced

Topics: More complex WebAssembly modules, integrating with existing JavaScript projects.

Project: Build a small game or performance-intensive application using WebAssembly.

Day 74: Advanced React Concepts

Topics: Context API, React Router, code splitting.

Project: Build a React application that uses the Context API and

React Router.

Day 75: State Management with Redux

Topics: Introduction to Redux, managing state in React apps.

Project: Implement Redux in a React application to manage complex state.

Day 76: Testing JavaScript Applications

Topics: Unit testing with Jest, end-to-end testing with Cypress.

Project: Write tests for an existing JavaScript project to ensure code reliability.

Day 77: Server-Side Rendering with Next.js

Topics: Introduction to Next.js, server-side rendering.

Project: Build a server-side rendered React application using Next.js.

Day 78: Advanced Vue.js Concepts

Topics: Vuex for state management, Vue Router.

Project: Build a Vue.js application with Vuex for state management and Vue Router for navigation.

Day 79: Advanced Angular Concepts

Topics: Angular services, HTTP client, RxJS.

Project: Develop an Angular application that fetches and displays data using HTTP client and RxJS.

Day 80: Building and Distributing JavaScript Libraries

Topics: Creating a reusable JavaScript library, publishing to npm.

Project: Create a small utility library and publish it on npm.

Day 81: Final Project Planning

Project: Plan your final project, decide on the scope, technologies, and timeline.

Day 82-90: Final Project Development

Project: Work on your final project. This could be a full-stack web application, a real-time communication tool, or a complex front-end project that integrates various concepts learned.

Days 91-100: Final Project and Advanced Topics

Day 91-95: Final Project Testing and Debugging

> **Project:** Test your project thoroughly, debug any issues, and refine the code.

Day 96-98: Final Project Documentation and Presentation

> **Project:** Document your project, write a README file, and prepare a presentation.

Day 99: Final Project Review and Polishing

> **Project:** Review your project, make final adjustments, and ensure everything is in order.

Day 100: Project Presentation and Reflection - **Project:** Present your final project to others, reflect on what you've learned, and plan your next steps in your JavaScript journey.