Object-Oriented Programming: Object-oriented design, Definition of classes: fields, methods, and constructors, Subclasses, inheritance, and method overriding, Dynamic dispatch: definition of method-call, Object-oriented idioms for encapsulation, Using collection classes, iterators, and other common library components

Functional Programming: Processing structured data (e.g., trees) via functions with cases for each data variant, First-class functions (taking, returning, and storing functions), defining higher-order operations on aggregates, especially map, reduce/fold, and filter

Event-Driven and Reactive Programming: Events and event handlers, Canonical uses such as GUIs, mobile devices, robots, servers, Externally-generated events and program generated events, Separation of model, view, and controller


Language Pragmatics: Principles of language design, Evaluation order, precedence, and associativity, Eager vs. delayed evaluation, Defining control and iteration constructs, External calls and system libraries

Laboratory work: Implementing object oriented concepts, Functional Programming concepts, Event Driven Concepts, Reactive Programming concepts and formal semantic concepts.

Text Books:

Reference Books: