KIDS & TEENS FREELANCING TRAINING INSTITUTE

Object Oriented Programming(OOP) Course Outline

Module 1: What is OOP (Object-Oriented Programming)?

- What does “Object-Oriented” mean in simple words?
- Real-life example: building a LEGO city with different buildings (objects)
- Why OOP makes coding easier, organized, and fun
- Main parts of OOP: Classes and Objects
- OOP vs. regular coding: imagine using templates instead of rewriting
- Activity: Think of real-life objects and their properties (e.g., a dog has a name, color)
- Visual task: Draw an object (like a car) with its features and actions

---

Module 2: Classes & Objects – Blueprints and Real Things

- What is a class? A blueprint for making objects
- What is an object? A real thing created from a class
- Real-life example: Class = Cookie Cutter, Object = Cookie
- Activity: Create a class called `Animal`, then make objects like Dog, Cat, Elephant
- Fun challenge: Add different features (color, sound) to each object
- Visual: Build a “class tree” of your favorite things

Module 3: Properties & Methods – What an Object Has and Does

- What are properties (also called attributes)?
- What are methods (actions an object can do)?
- Real-life example: A Robot has a name (property) and can dance (method)
- Activity: Create a `Robot` class with 2 properties and 2 actions
- Bonus: Make your robot greet you with a custom message
- Drawing activity: Create your own superhero with powers (methods) and features (properties)

Module 4: Encapsulation – Hiding and Protecting Data

- What is encapsulation in simple terms?
- Real-life example: Remote control – you don't need to know how it works inside
- Why we keep some data private (using `private` and `public`)
- Activity: Create a class with a private secret and a method to reveal it
- Visual activity: Draw a locked box (data) and a key (method) to open it

Module 5: Inheritance – Passing Traits to New Classes

- What is inheritance in OOP?
- Real-life example: Kids inherit traits from parents (height, hair color)
- How one class can "borrow" from another
- Activity: Create a `Vehicle` class and make `Car` and `Bike` inherit from it
- Bonus: Add special features to each subclass
- Fun chart: Draw a family tree of classes!

Module 6: Polymorphism – One Action, Different Forms

- What is polymorphism? (One name, many behaviors)
- Real-life example: "Play" means something different to a baby, teen, and adult
- Using the same method name in different classes with different results
- Activity: Create a `Draw()` method that acts differently in `Circle`, `Square`, and `Triangle` classes
- Bonus challenge: Add sound to each shape when it's drawn
- Visual: Make a table showing different outcomes from the same method name

Module 7: Abstraction – Focusing on What Matters

- What is abstraction in OOP?
- Real-life example: Driving a car without knowing how the engine works
- Why abstraction keeps things simple and neat
- How to create abstract classes and methods
- Activity: Make an abstract class `Animal` with a method `MakeSound()`, and then build specific animals that sound different
- Drawing task: Show the difference between what's hidden and what's shown

Module 8: OOP in the Real World & Final Project

- Where is OOP used? (Games, apps, websites, robots, more!)
- Real-life example: Video games like Minecraft use OOP to create everything!
- Jobs that use OOP: Game Developer, App Creator, Software Engineer
- Activity: Plan your own mini project (like a game, pet simulator, or robot builder)
- Final challenge: Build your own OOP-based program using everything you've learned
- Share and celebrate: Present your OOP idea to family or friends

---

Bonus Materials:

- Interactive OOP quizzes and code puzzles
- Beginner tools: Replit, Visual Studio Code, Scratch with OOP-like logic
- Printable OOP cheat sheet (with visuals)
- Team project idea: Create a zoo or game using objects and classes
- Official Certificate of Completion