LAPT

LONDON ACADEMY OF PROFESSIONAL TRAINING

CERTIFICATE

Certificate in Programming Fundamentals (Language Neutral)

Foundation Level

LAPT — London Academy of Professional Training

Address 85 Great Portland Street, W1W 7LT, London, United Kingdom

Email info@lapt.org

Telephone +44 7513 283044

Web lapt.org

Reference IT-SDA-F • IT Centres

UKPRN 10067351 • Registered in England and Wales, company 4984798

6	24	110	440	60%	Lifetime
SUBJECTS	CHAPTERS	CLASSES	TOTAL MARKS	PASS MARK	VALIDITY

AT A GLANCE

REFERENCE	**IT-SDA-F**
AWARD	**Certificate**
ASSESSMENT	**440 marks — 264 to pass (60%)**
PREREQUISITES	**None**
VALIDITY	**Lifetime**
AWARDING BODY	**LAPT — London Academy of Professional Training**

CURRICULUM

1 Algorithms and Logic

4 chapters · 20 classes · 90 marks

- **1 Understanding Basic Algorithmic Concepts** — 5 classes
  - 1.1 Exploring What Algorithms Are
  - 1.2 Understanding Algorithm Components: Inputs and Outputs
  - 1.3 Identifying and Describing Basic Control Structures
  - 1.4 Applying Pseudocode to Design Simple Algorithms
  - 1.5 Analyzing Algorithm Efficiency and Effectiveness
- **2 Core Logic Structures and Constructs** — 5 classes
  - 2.1 Understanding Basic Logical Constructs
  - 2.2 Implementing Conditional Statements
  - 2.3 Exploring Loop Structures
  - 2.4 Applying Logical Operators in Algorithms
  - 2.5 Constructing and Analyzing Control Flow
- **3 Introduction to Algorithmic Efficiency** — 5 classes
  - 3.1 Understanding Algorithmic Efficiency
  - 3.2 Exploring Time Complexity Concepts
  - 3.3 Analyzing Space Complexity in Algorithms
  - 3.4 Comparing Algorithms: Best, Average, and Worst Case Scenarios
  - 3.5 Applying Efficiency Principles to Optimize Code
- **4 Designing and Analyzing Data Structures** — 5 classes
  - 4.1 Understanding Data Structures: Core Concepts and Types
  - 4.2 Comparing Data Structures: Strengths and Weaknesses
  - 4.3 Creating Data Structures: Basic Design Techniques
  - 4.4 Analyzing Data Structures: Big O Notation and Complexity
  - 4.5 Implementing Solutions: Choosing the Right Data Structure

2 Data Management Fundamentals

4 chapters · 20 classes · 65 marks

- **1 Understanding Data Types and Structures** — 5 classes
  - 1.1 Exploring Primitive Data Types
  - 1.2 Analyzing Composite Data Structures
  - 1.3 Comparing Mutable and Immutable Types
  - 1.4 Classifying Data Structures by Use Case
  - 1.5 Implementing Data Structures in Applications
- **2 Data Storage and Retrieval Techniques** — 5 classes
  - 2.1 Understanding Data Storage Concepts
  - 2.2 Exploring Data Retrieval Methods
  - 2.3 Implementing Basic Data Structures
  - 2.4 Comparing Data Storage Solutions
  - 2.5 Applying Efficient Data Retrieval Techniques
- **3 Data Validation and Integrity Constraints** — 5 classes
  - 3.1 Understanding Data Validation Principles
  - 3.2 Exploring Common Data Integrity Constraints
  - 3.3 Implementing Data Validation Techniques
  - 3.4 Analyzing the Impact of Integrity Constraints on Data
  - 3.5 Applying Data Validation and Integrity in Practice
- **4 Basic Data Analysis and Manipulation** — 5 classes
  - 4.1 Understanding Data Types and Structures
  - 4.2 Extracting and Loading Data for Analysis
  - 4.3 Performing Basic Data Cleaning Techniques
  - 4.4 Summarizing Data with Descriptive Statistics
  - 4.5 Visualizing Data for Insightful Interpretation

3 Introduction to Programming

4 chapters · 20 classes · 90 marks

- **1 Understanding Programming Concepts and Computational Thinking** — 5 classes
  - 1.1 Defining Core Programming Concepts
  - 1.2 Exploring Variables and Data Types
  - 1.3 Understanding Control Structures
  - 1.4 Applying Logical and Computational Thinking
  - 1.5 Solving Problems with Algorithms
- **2 Data Types, Variables, and Operators** — 5 classes
  - 2.1 Understanding Basic Data Types
  - 2.2 Exploring Variables and Their Uses
  - 2.3 Mastering Arithmetic and Logical Operators
  - 2.4 Applying String and Boolean Data Types
  - 2.5 Solving Problems with Variables and Operators
- **3 Control Structures: Sequence, Selection, and Iteration** — 5 classes
  - 3.1 Understanding Sequence in Programming
  - 3.2 Implementing Conditional Selection
  - 3.3 Utilizing Iteration Constructs
  - 3.4 Combining Sequence, Selection, and Iteration
  - 3.5 Applying Control Structures in Real-world Problems
- **4 Basic Input/Output and Error Handling** — 5 classes
  - 4.1 Understanding Standard Input and Output
  - 4.2 Implementing Input Functions in Programming
  - 4.3 Exploring Output Techniques and Methods
  - 4.4 Identifying and Managing Common I/O Errors
  - 4.5 Applying Error Handling Strategies in Code

4 Practical Programming Applications

4 chapters · 10 classes · 65 marks

- **1 Understanding Programming Concepts and Tools** — 5 classes
  - 1.1 Explore Basic Programming Concepts
  - 1.2 Identify Core Programming Tools
  - 1.3 Demonstrate Syntax and Structure
  - 1.4 Utilize Software Development Environments
  - 1.5 Implement Debugging Techniques
- **2 Problem Solving and Algorithm Development** — 5 classes
  - 2.1 Understanding Problem-Solving Frameworks
  - 2.2 Identifying and Defining Programming Problems
  - 2.3 Developing Step-by-Step Algorithms
  - 2.4 Implementing Algorithms Using Pseudocode
  - 2.5 Evaluating and Refining Algorithmic Solutions
- **3 Implementing Code Logic and Structures**
- **4 Testing, Debugging, and Optimizing Code**

5 Software Design Principles

4 chapters · 20 classes · 65 marks

- **1 Understanding Core Software Design Principles** — 5 classes
  - 1.1 Introduction to Core Software Design Principles
  - 1.2 Exploring Abstraction in Software Design
  - 1.3 Understanding Encapsulation and Its Benefits
  - 1.4 Implementing Modular Design for Flexibility
  - 1.5 Applying Principles to Real-World Scenarios
- **2 Design Patterns and Their Applications** — 5 classes
  - 2.1 Understanding Design Patterns: An Overview
  - 2.2 Exploring Creational Patterns: Building Through Instantiation
  - 2.3 Harnessing Structural Patterns: Organizing Objects and Classes
  - 2.4 Implementing Behavioral Patterns: Managing Communication and Workflow
  - 2.5 Applying Design Patterns: Real-World Examples and Best Practices
- **3 Principles of Object-Oriented Design** — 5 classes
  - 3.1 Understanding the Core Concepts of Object-Oriented Design
  - 3.2 Identifying and Defining Classes and Objects
  - 3.3 Exploring Encapsulation and Information Hiding
  - 3.4 Mastering Inheritance for Code Reusability
  - 3.5 Implementing Polymorphism for Flexible Software Design
- **4 Designing for Scalability and Maintainability** — 5 classes
  - 4.1 Understanding Scalability and Maintainability Concepts
  - 4.2 Analyzing System Requirements for Scalability
  - 4.3 Implementing Design Patterns for Maintainability
  - 4.4 Evaluating Design Decisions for Scalability
  - 4.5 Applying Modular Design for Future Growth

6 Syntax Across Languages

4 chapters · 20 classes **65 marks**

- **1 Understanding Basic Syntax Elements** — 5 classes
  - 1.1 Identifying Common Syntax Elements
  - 1.2 Exploring Variable Declarations
  - 1.3 Understanding Control Structures
  - 1.4 Utilizing Functions and Procedures
  - 1.5 Applying Syntax Rules in Practice
- **2 Control Flow Constructs** — 5 classes
  - 2.1 Understanding Conditional Statements
  - 2.2 Exploring Loop Structures
  - 2.3 Implementing Branching with Conditionals
  - 2.4 Comparing Iterative Constructs Across Languages
  - 2.5 Applying Control Flow to Solve Problems
- **3 Functions and Modular Syntax** — 5 classes
  - 3.1 Understanding Functions: Concepts and Terminology
  - 3.2 Exploring Function Syntax: Parameters and Return Values
  - 3.3 Implementing Modular Code: Breaking Down Problems
  - 3.4 Applying Functions: Real-World Examples and Uses
  - 3.5 Reviewing Function Syntax: Best Practices and Common Pitfalls
- **4 Error Handling and Exceptions** — 5 classes
  - 4.1 Understanding Error Types and Exceptions
  - 4.2 Identifying and Interpreting Error Messages
  - 4.3 Implementing Basic Try-Catch Blocks
  - 4.4 Managing Multiple Exceptions
  - 4.5 Creating Custom Error Handlers