

Teaching Guidelines for  
**Logic Building & Problem Solving (optional)**  
PG-DAC March 2022

---

**Duration:** 18 hours online sessions + problem solving as self study

**Objective:** This preparatory module is conducted online before the actual PG-DAC course commences for students to learn to think logically and how to solve problems. A number of problems will be given to the students to solve them logically.

**Prerequisites:** Knowledge of computer fundamentals.

**Evaluation:** No separate evaluations. Daily assignment problems are to be solved and submitted.

---

*Note: Each day comprises 3 hours' online lecture sessions followed by solving various problems using the topics learnt by the students.*

**Day 1: Building Logic and Solving Problems**

**Lecture:**

- Logical thinking
- Problem solving process

**Assignments:**

- Practice logic building on given problems

**Day 2: Decision making**

**Lecture:**

- If-else, nested if-else, switch case

**Assignments:**

- Decision making problems using if-else, nested if-else, switch case

**Day 3: Control Statements**

**Lecture:**

- Loops

**Assignments:**

- Conditional problems using loops

**Day 4: Modular Programming**

**Lecture:**

- Functions

**Assignments:**

- Problems using functions

**Day 5: Arrays**

**Lecture:**

- Arrays

**Assignments:**

- Problems using arrays

**Day 6: Strings**

**Lecture:**

Strings

**Assignments:**

Problems using strings

## Teaching Guidelines for Concepts of Programming & Operating System PG-DAC March 2022

---

**Duration:** 66 hours (36 class room hours + 22 lab hours + 8 revision/practice hours)

**Prerequisites:** Knowledge of computer fundamentals

**Evaluation:** 100 marks (Concepts of Programming – 40 marks + Operating Systems – 60 marks)

**Weightage:** Theory exam – 40%, Lab exam – 40%, Internals – 20%

---

### Concepts of Programming

**Duration:** 10 theory hours + 10 lab hours (**20 hours**)

**Objective:** To introduce the fundamental programming concepts in Java.

**Evaluation:** 40 marks (Theory exam: 12 + Lab exam: 20 + Internals: 8 marks)

**Text Book:**

- Core and Advanced Java Black Book / Dreamtech Press

**References:**

- Java The Complete Reference by Herbert Schildt / McGraw Hill
  - Core Java : Fundamentals - Volume 1 Gary Cornell, Cay S. Horstmann/ Pearson
  - Programming in Java by Sachin Malhotra, Saurabh Choudhary / Oxford University Press
- 

(Note: Each Session is of 2 hours)

**Sessions 1 & 2:**

**Lecture:**

**Getting Started**

- Setup development environment (JRE, JDK, eclipse)
- Writing your first Java program

**Variables & Methods**

- About main () method
- Java Data Types, Primitives and Binary Literals
- Data type compatibility and casting of primitive data types
- Static variables and methods
- Accessing static variables and methods of different class
- Final variables

**Operators**

- Arithmetic Operator
- Relational Operator
- Logical Operator
- Unary Operator
- Ternary Operator
- Assignment Operator

**Lab:**

Write Java programs to:

- Print Hello World
- Add two numbers/binary numbers/characters
- Calculate compound interest
- Calculate power of a number
- Swap two numbers
- Calculate area of rectangle
- Calculate area and circumference of circle using multiple classes
- Java program to find ASCII value of a character
- Print default values of primitive data type variables
- Swap two variables without using the third variable
- Print Fibonacci series till n

### **Session 3: Conditional and Looping Statements**

#### **Lecture:**

- If, else if, switch
- break & continue keyword
- for loop
- while loop
- do while loop
- Recursion

#### **Lab:**

Write Java programs to:

- Display prime numbers between 1 and 100 or 1 and n
- Find the factorial of a number
- Check if a number is palindrome or not
- Add two integer variables in 5 different ways using functions and control statement
- Find square root of a number without sqrt method
- Check Armstrong number
- Calculate grades of students using their marks
- Use switch case, recursion, print patterns, etc.

### **Session 4: Objects**

#### **Lecture:**

- Reference variables and methods
- Constructors (Default constructor, parameterised constructor)
- Static method v/s instance method
- Reference variable as instance member of the class
- String class

#### **Lab:**

- Build a class Employee which contains details about the employee and compile and run its instance.
- Build a class which has references to other classes. Instantiate these reference variables and invoke instance methods.

### **Session 5: Arrays**

#### **Lecture:**

- Initializing an Array in Java
- Two dimensional array in java
- Java Variable Arguments explained
- Add, update, read array elements
- Sorting and searching in array

- Java String Array to String
- How to copy arrays in Java

**Lab:**

Write Java programs to:

- Calculate average of numbers using Array
- Reverse an array
- Sort an array in ascending order
- Convert char Array to String
- Add two Matrix using Multi-dimensional Arrays
- Sort strings in alphabetical order
- Find out the highest and second highest numbers in an array
- Concatenate two arrays

## Concepts of Operating System

**Duration:** 26 class room hours + 12 lab hours (**38 hours**)

**Objective:** To introduce Operating System concepts with Linux environment, and to learn Shell Programming.

**Evaluation:** 60 marks (Theory exam: 28 + Lab exam: 20 + Internals: 12 marks)

**Text Books:**

- Operating Systems Principles by Abraham Silberschatz, Peter Galvin & Greg Gagne / Wiley
- Unix Concepts and Applications by Sumitabha Das / McGraw Hill

**References:**

- Modern operating Systems by Andrew Tanenbaum & Herbert Bos/ Pearson
- Principles of Operating Systems by Naresh Chauhan / Oxford University Press
- Beginning Linux Programming by Neil Matthew & Richard Stones / Wrox
- Operating System : A Design-Oriented Approach by Charles Crowley / McGraw Hill

---

(Note: Each Session is of 2 hours)

**Session 1:****Lecture:***Introduction to OS*

- What is OS; How is it different from other application software; Why is it hardware dependent
- Different components of OS
- Basic computer organization required for OS
- Examples of well known OS including mobile OS, embedded system OS, Real Time OS, desktop OS server machine OS etc. ; How are these different from each other and why
- Functions of OS
- User and Kernel space and mode; Interrupts and system calls

**(No Lab)****Session 2:****Lecture:***Introduction to Linux*

- Working basics of file system

- Commands associated with files/directories & other basic commands. Operators like redirection, pipe
- What are file permissions and how to set them
- Permissions (chmod, chown, etc); access control list; network commands (telnet, ftp, ssh, sftp, finger)
- System variables like – PS1, PS2 etc. How to set them

#### *Shell Programming*

- What is shell; What are different shells in Linux?
- Shell variables; Wildcard symbols
- Shell meta characters; Command line arguments; Read, Echo

#### **Lab:**

- Working with various OS commands
- Shell programs related to Session 2

#### **Session 3:**

##### **Lecture:**

#### *Shell Programming*

- Decision loops (if else, test, nested if else, case controls, while...until, for)
- Regular expressions; Arithmetic expressions
- More examples in Shell Programming

#### **Lab:**

- Shell Programs related to Session 3

#### **Sessions 4, 5 & 6:**

##### **Lecture:**

#### *Processes*

- What is process; preemptive and non-preemptive processes
- Process management; Process life cycle
- What are schedulers – Short term, Medium term and Long term.
- Process scheduling algorithms – FCFS, Shortest Job First, Priority, RR, Queue. Belady's Anomaly
- Examples associated with scheduling algorithms to find turnaround time to find the better performing scheduler.
- Process creation using fork; waitpid and exec system calls; Examples on process creation; Parent and child processes
- Orphan and zombie processes

#### **Lab:** (2 hours)

- Creating processes - parent and child processes
- Handling orphan and zombie processes.

#### **Session 7:**

##### **Lecture:**

#### *Signals*

- What are signals
- Generating and handling signals

#### *Threads*

- What are threads; user and kernel threads; how threads are different from processes
- Thread programming using pthread.

#### **Lab:**

- Assignment on signals
- Assignment on threads – Thread creation, thread synchronization

## Sessions 8 & 9:

### Lecture:

#### *Memory management*

- What are different types of memories; What is the need of Memory management
- Continuous and Dynamic allocation
- First Fit, Best Fit, worst Fit
- Compaction
- Internal and external fragmentation
- Segmentation – What is segmentation; Hardware requirement for segmentation; segmentation table and its interpretation
- Paging – What is paging; hardware required for paging; paging table; Translation look aside buffer
- Concept of dirty bit
- Shared pages and reentrant code
- Throttling

(*No Lab*)

## Session 10:

### Lecture:

#### *Virtual Memory*

- What is virtual memory
- Demand paging
- Page faults
- Page replacement algorithms

(*No Lab*)

## Session 11:

### Lecture:

#### *Deadlock*

- Necessary conditions of deadlock
- Deadlock prevention and avoidance
- Semaphore
- Mutex
- Producer consumer problem
- Dead-lock vs Starvation

### Lab:

- Semaphore, Mutex

## Sessions 12 & 13:

### Lecture:

#### *Inter process communication*

- Message queues,
- Shared memory
- Pipes
- FIFO

### Lab: (2 hours)

- IPC using shared memory
- IPC using Pipes
- IPC using FIFO

# Teaching Guidelines for Object Oriented Programming with Java

## PG-DAC March 2022

---

**Duration:** 104 hours (46 theory hours + 46 lab hours + 12 revision/practice hours)

**Objective:** To reinforce knowledge of Object Oriented Programming concepts using Core Java.

**Prerequisites:** Basic knowledge of computer programming

**Evaluation:** Total 100 marks

**Weightage:** Theory exam – 40%, Lab exam – 40%, Internals – 20%

### Text Book:

- Core and Advanced Java Black Book / Dreamtech Press

### References:

- Java 8 Programming Black Book / Dreamtech Press
- Core Java : Volume 1 - Fundamentals by Cay S. Horstmann / Prentice Hall
- Core Java : Volume 2 - Advanced Features by Cay S. Horstmann / Prentice Hall
- Programming in Java by Sachin Malhotra, Saurabh Choudhary / Oxford University Press
- Java The Complete Reference by Herbert Schildt / McGraw Hill
- Core Java 8 for Beginners by Sharanam Shah, Vaishali Shah / Shroff Publishers
- Murach's Java Programming by Joel Murach / Mike Murach
- Object-Oriented Analysis and Design with applications by Grady Booch / Pearson
- Object-Oriented Analysis and Design Using UML - An Introduction to Unified Process and Design Patterns by Mahesh P. Matha / PHI

---

(Note: Each Session is of 2 hours)

### Session 1:

#### Lecture:

Introduction to java

Features of java

JVM Architecture

JDK and its usage

Structure of java class

Working with data types: Primitive data types

### Session 2:

#### Lecture:

Operators

- Unary, binary, Arithmetic, Assignment, compound, relational, logical, equality

Control statements

- if-else-if, switch, ternary operator, for loop, while loop, do-while loop

Declaring variables and methods

Data type compatibility

**Lab 1 & 2:**

Get yourself acquainted with java environment.

Print different patterns of asterisk (\*) using loops (e.g. triangle of \*).

**Tutorial:**

Compare syntactical similarities and dissimilarities between Java and C++.

**Session 3:****Lecture:**

Static variables and methods

Accessing static variables and methods of different class

Introduction to reference data types

Reference variables and methods

Difference between reference data types and primitive data types

Difference between reference variable and static variable

**Session 4:****Lecture:**

Constructors, initializing reference variables using constructors

Pass by value v/s pass by reference

Re-assigning a reference variable

Passing reference variable to method

Initializing reference variable of different class

Heap memory and stack memory

**Lab 3 & 4:**

Print default values of static & instance variables for different data types.

Build a class Employee which contains details about the employee and compile and run its instance.

Build a class which has references to other classes. Instantiate these reference variables and invoke instance methods.

**Tutorial:**

Understand role of stack and heap memory in method invocation and object creation.

**Object Oriented Programming Concepts****Session 5:****Lecture:**

Introduction to OOP concepts

Encapsulation

Inheritance: single & multilevel

**Session 6:****Lecture:**

Inheritance: Hierarchical

Polymorphism: Compile time and runtime polymorphism

Rules of overriding and overloading of methods

super and this keywords

Upcasting & downcasting of a reference variable

**Lab 5 & 6:**

Create a class Employee and encapsulate the data members.

Create demo applications to illustrate different types of inheritance.

**Session 7:****Lecture:**

Abstract class and abstract methods

Interface (implementing multiple interfaces)

**Session 8:****Lecture:**

Final variables, final methods and final class

Functional interface

New interface features (Java 8 & above)

Arrays

Enumerations

**Lab 7 & 8:**

Create an Array of Employee class and initialize array elements with different employee objects.

Try to understand the no of objects on heap memory when any array is created.

**Session 9:****Lecture:**

Access modifiers (public, private, protected and default)

Packages and import statements

Static imports

Constructor chaining (with and without packages)

Accessing protected variables and methods outside the package

**Session 10:****Lecture:**

Garbage collection in java

Requesting JVM to run garbage collection

Different ways to make object eligible for garbage collection: (Nulling a reference variable, Re-assigning a reference variable & island of isolation)

Finalize method

**Lab 9 & 10:**

Create a demo application to understand the role of access modifiers.

Implement multilevel inheritance using different packages.

Access/invoke protected members/methods of a class outside the package.

Override finalize method to understand the behavior of JVM garbage collector.

**Sessions 11 & 12:****Wrapper Classes and String Class****Lecture:**

Wrapper classes and constant pools

String class, StringBuffer & StringBuilder class

String pool

**Lab 11 & 12:**

Create sample classes to understand boxing & unboxing.

Use different methods of java defined wrapper classes.

Create StringDemo class and perform different string manipulation methods.

**Tutorial:**

Understand the difference between String / StringBuffer / StringBuilder.

**Sessions 13 & 14:****Exception Handling****Lecture:**

Exception hierarchy, Errors, Checked and un-checked exceptions

Exception propagation

try-catch-finally block , throws clause and throw keyword

Multi catch block

Creating user defined checked and unchecked exceptions

**Lab 13 & 14:**

Create user defined checked and unchecked exceptions .

**Session 15:****java.io & java.nio Package****Lecture:**

Brief introduction to InputStream, OutputStream, Reader and Writer interfaces

NIO package

Serialization and de-serialization

Shallow copy and deep copy

**Lab 15:**

Create a Demo class to Read & write image/text files.

Create SerializationDemo class to illustrate serialization and de-serialization process.

**Session 16:****Lecture:****Object Class & java.util Package**

Date, DateTime, Calendar class

Converting Date to String and String to Date using SimpleDateFormat class

Object Class: Overriding to String, equals & hashCode method

**Collections****Session 17:****Lecture:**

Introduction to collections: Collection hierarchy

List, Queue, Set and Map Collections

List Collection:

- ArrayList, LinkedList
- Vector (insert, delete, search, sort, iterate, replace operations)

Collections class

Comparable and Comparator interfaces

Queue collection

**Lab 16 & 17:**

Create DateManipulator class to convert String to date, date to String and to find out number of days between two dates.

Create a list of java defined wrapper classes and perform insert/delete/search/iterate/sort operations.

Create a collection of Employee class and sort objects using comparable and comparator interfaces.

Implement Queue data structure using LinkedList and Queue collection.

**Sessions 18 & 19:****Lecture:**

Set Collection:

- HashSet, LinkedHashSet & TreeSet collection
- Backed set collections

Map Collection:

- HashTable, HashMap, LinkedHashMap & TreeMap classes
- Backed Map collections

Generics

Concurrent collections

**Lab 18 & 19:**

Create an Employee HashSet collection and override equals & hashCode methods to understand how the set maintains uniqueness using these methods.

Create a Sample class to understand generic assignments using "? extends SomeClass", "? super someclass" and "?".

**Session 20:****Lecture:**

MultiThreading : Thread class and Runnable Interface

sleep, join, yield, setPriority, getPriority methods

ThreadGroup class

**Lab 20:**

Create multiple threads using Thread class and Runnable interfaces.

Assign same task and different task to multiple threads.

Understand sleep, join, yield methods.

**Sessions 21 & 22:****Lecture:**

Synchronization

Deadlock

Wait, notify and notifyAll methods

Producer & Consumer problem

**Lab 21 & 22:**

Create a Deadlock class to demonstrate deadlock in multithreading environment.

Implement wait, notify and notifyAll methods.

Demonstrate how to share threadlocal data between multiple threads.

**Session 23:**

**Lecture:**

Inner Class (Regular, Method local, Anonymous & static inner class)

Lambda Expression

Reflection

**Lab 23:**

Invoke private methods of some other class using reflection.

Create multiple threads using anonymous inner classes.

Create multiple threads using lambda expressions.

# Teaching Guidelines for Algorithms and Data Structures Using Java

## PG-DAC March 2022

---

**Duration:** 82 hours (30 class room hours + 28 lab hours + 18 mini project hours + 6 revision/practice hours)

**Objective:** To reinforce knowledge of problem solving techniques, data structure concepts and analysis of different algorithms using Java.

**Prerequisites:** Knowledge of programming in C/C++ with object oriented concepts

**Evaluation:** 100 Marks

**Weightage:** Theory exam – 40%, Lab exam – 40%, Internals & Mini project – 20%

**Text Book:**

- Data Abstraction and Problem Solving with Java: Walls and Mirrors by Janet Prichard , Frank M. Carrano / Pearson

**References:**

- Problem Solving: Best Strategies to Decision Making, Critical Thinking and Positive Thinking by Thomas Richards / Kindle Edition
  - Data Abstraction and Problem Solving with Java: Walls and Mirrors by Janet Prichard , Frank M. Carrano / Pearson
  - Object-oriented Analysis and Design Using UML - An Introduction to Unified Process and Design Patterns by Mahesh P. Matha / PHI
  - Introduction to Algorithms by Cormen, Leiserson, Rivest and Stein
- 

(Note: Each Session is of 2 hours)

**Session 1:**

**Problem Solving & Computational Thinking**

**Lecture:**

- Define the problem
- Identify the problem
- Introduction to Problem Solving
- Problem solving basics

**Lab:**

- Faculty needs to assign different problems, mostly real world problems
- Students (team-wise, two students in a team) need to analyze as per the techniques learned
- Based on the above problems students need to select as per the selection criteria learned
- They need to implement the selected solution and need to do the documentations.
- Introducing the mini project ideas

**Sessions 2 & 3:**

**Algorithms & Data Structures**

**Objective:** At the end of the session students should know, what is the importance of data structure in problem solving. How stacks, queues, circular queues work. Their real world applications. How to solve problems using these data structures.

**Lecture:**

- Introductory Concepts
- Algorithm Constructs
- Complexity analysis of algorithms (Big O notation )
- O O design: Abstract Data Types (ADTs)
- Basic Data Structures
  - Arrays
  - Stacks
  - Queues
  - Circular Queues

**Lab:**

- Implement stack through array
- Complexity analysis of loops and recursive algorithms.
- Implement queues with inserting element at different location (First, Last)
- Implement circular queue

**Sessions 4 & 5:**

**Linked List Data Structures**

**Objective:** At the end of the session students should know, what are applications of Linked List, different types of link list. Comparison with arrays as when to use linked list and when to use array.

**Lecture:**

- Linked lists
  - Singly linked lists
  - Doubly linked lists
  - Circular linked lists
  - Node-based storage with arrays

**Lab:**

- Implement circular queue using linked list
- Implement stack using linked list

**Session 6:**

**Recursion**

**Objective:** At the end of the session students should know what is recursion, type of recursion, local variable in recursion, stack manipulation during recursion, function complexity

**Lecture & Lab:**

- What is recursion?
- What is the base condition in recursion.
- Direct and indirect recursion.
- Memory is allocated to different function calls in recursion.
- Pro and cons of recursion
- Function complexity during recursion

**Sessions 7 & 8:**

**Trees & Applications**

**Objective:** At the end of the session students should know what is the use of binary trees, how to create binary search trees. Different tree traversals. What are the applications of binary trees? How to calculate search complexity in binary search trees? What are the limitations of binary search trees? What are different options to overcome the binary search tree limitations.

**Lecture:**

- Introduction to trees
- Trees and terminology
- Tree traversals
- Binary trees
- Complete binary trees / Almost complete binary tree (ACBT)
- Array implementation of ACBT
- Binary search trees
- AVL tree
- Multi-way tree
- Brief introduction and uses cases of B-Tree /B+Tree.

**Lab:**

- Write a program to implement a binary search tree and the following operations on it:
  - Create()
  - Tree traversals - Breadth First Search, Depth First Search, Inorder(), Preorder(), Postorder()
  - Delete()

**Sessions 9 & 10:****Searching & Sorting Algorithms**

**Objective:** At the end of the session students should know what are the different types of sorting and searching algorithms, why all the sorting algorithms are equally important despite different time/space complexity of the algorithms. How the complexity is calculated for each of them. How to pick a sorting algorithm given the nature of the data to be sorted.

**Lecture:**

- Objectives of Searching
  - The Sequential Search
  - Analysis of Sequential Search
  - The Binary Search
- Analysis of Binary Search
- Introduction to sorting
  - Selection sort
  - Insertion sort
  - Bubble sort
  - Heapsort
  - Mergesort
  - Quicksort
- Analysis of sorting algorithms

**Lab:**

- Writing program to search an item through sequential search technique.
- Implement to find an item in a list through binary search
- Implement sorting algorithm for: insertion sort, Quicksort

**Session 11:****Hash Functions and Hash Tables**

**Objective:** At the end of the session students should know what is hashing, what is the importance of hashing, comparative complexity of hashing with other search techniques. Problems (collision) with hashing and what are the different solutions of that.

**Lecture:**

- Hashing & Introduction to hash tables

- Hash Functions
- Different type of hash functions
- Collision Resolution
- Linear Probing
- Quadratic Probing
- Double Hashing
- Inserting and Deleting an element from a hash table

**Lab:**

- Implement hashing techniques in different programs solved earlier
- Write a program to implement Hash table
- Fibonacci recursive algorithm improvement using hash table

**Sessions 12 & 13:**

**Graphs & Applications**

**Objective:** At the end of the session students should know what is graph? Why is graph the most generic data structure? Different types of graphs. Different representation of graph? Graph traversals (Breadth First Traversal, Depth First Traversal). Different applications which can be solved with graphs, real world and programming problems with graphs.

**Lecture:**

- Introduction to graph theory
- Graph Terminology
- Different types of Graphs
- Representation of Graphs
  - Adjacency Matrix
  - Adjacency List
  - Graph Traversal Algorithms ( Breadth First Search, Depth First Search)
- Shortest Path
  - Level Setting : Dijkstra's algorithm
  - Level Correcting: All-pairs shortest path, Floyd-Warshall algorithm
- Spanning Trees
  - Minimum spanning tree algorithms,
  - Prim's algorithm
  - Kruskal's Algorithm

**Lab:**

- Implement a graph using adjacency Matrix and traverse using Depth First Search.
- Implement a graph and do traversal using stack and queue.

**Sessions 14 & 15:**

**Algorithm Designs**

**Objective:** At the end of the session students should know what are different classes of algorithms. What is the nature of each class of algorithms? How to pick an algorithm for a particular problem. What problems fall under each class of algorithms. What are the worst case, average case and the best case for algorithms?

**Lecture:**

- What are the different class of algorithms
- How to write efficient Algorithm
- Introduction to algorithm design techniques
- Algorithm Design techniques
- Analysis of an Algorithm

- o Asymptotic Analysis
- o Algorithm Analysis
- Analysis of different type of Algorithms
  - o Divide and Conquer Algorithm
  - o Greedy algorithm
  - o Dynamic Programming algorithm
  - o Brute force algorithm
  - o Backtracking algorithms
  - o Branch-and-bound algorithms
  - o Stochastic algorithms
- Complexity
  - o Complexity Analysis
  - o Space complexity of algorithm
  - o Time complexity of algorithm
- Case study on Algorithm Design techniques
- Application of Data structures

**Lab + Assignment:** (2 hours)

- Study on different Algorithms
- Compare different Algorithms previously programmed and do the analysis
- Mini project implementation guidelines

### **Mini Project (3 days)**

#### **Data Structure Implementation and Applications**

**Objective:** Implementation hands-on of data structures, implementation and design of a mini project covering the non-linear data structure and a shortest algorithm.

- Stack/Queue, Link List
- Merge Sort/Heap Sort
- Binary Search Tree
- Graph
- Dijkstra Algorithm

## Teaching Guidelines for Database Technologies

PG-DAC March 2022

---

**Duration:** 72 hours (32 classroom hours + 32 lab hours + 8 revision/practice hours)

**Objective:** To introduce students to RDBMS and NoSQL Databases and facilitate hands-on experience on SQL (using MySQL) and MongoDB.

**Prerequisites:** Working knowledge of Windows and Linux, familiarity with programming.

**Evaluation:** 100 Marks

**Weightage:** Theory Exam – 40%, Lab exam – 40%, internals – 20%

### Text Book:

- Murach's MySQL by Joel Murach / Shroff Publisher

### References:

- Database System Concepts by Abraham Silberschatz, Henry Korth and S. Sudarshan / McGraw Hill
  - Database Design and Relational Theory: Normal Forms and All That Jazz by C. J. Date (Author) / O'Reilly
  - Fundamentals of Database System by Shamkant B. Navathe, Ramez Elmasri / Pearson
  - MySQL: The Complete Reference by Vikram Vaswani / McGraw Hill
  - SQL & NoSQL Databases: Models, Languages, Consistency Options and Architectures for Big Data Management by Andreas Meier and Michael Kaufmann / Springer
  - MongoDB: The Definitive Guide by Shannon Bradshaw, Eoin Brazil and Kristina Chodorow / O'Reilly
  - <http://bigdata.stratebi.com/?language=en>
- 

(Note: Each Lecture and Lab Session is of 2 hours)

### Session 1:

#### Lecture

Introduction to DBMS, Basic Database Terminology

Types of DBMS: Relational, Object Relational and NoSQL Databases

Introduction to MySQL, MySQL Clients (Monitor, Shell, Workbench)

#### Lab

Using MySQL Monitor, Shell, and Workbench

### Session 2:

#### Lecture

Data Models (Conceptual, Logical, Physical)

Database Design, Entity-Relationship Diagram (ERD)

Codd's 12 rules for RDBMS

Introduction to SQL, Categories of SQL Commands: DDL, DML, DCL, DTL/TCL

DDL (CREATE/ALTER/DROP/TRUNCATE)

#### Lab

Performing basic CREATE, ALTER, DROP Commands

**Session 3:****Lecture**

Data Redundancy, Data Anomalies, Functional Dependency

Normalization, Need for Normalization

Normal Forms (1st NF, 2nd NF, 3rd NF, BCNF) with examples, Introduction to 4th and 5th NF

DML (INSERT/UPDATE/DELETE)

**Lab**

DML (INSERT/UPDATE/DELETE), TRUNCATE

**Session 4:****Lecture**

MySQL Data Types, Database Constraints (Primary Key, Unique, Not Null, Foreign Key, Default, Check\*)

Aggregate Functions, Grouping Things Together (Group By, Having)

LIKE Operator, DISTINCT, Sorting (Order by clause)

BETWEEN... AND Operators, Comparing Nulls (IS NULL/IS Not NULL), IN/NOT IN

**Lab**

Defining Data Types for Columns

Creating, Altering, Dropping Constraints

Aggregate Functions: SUM(), AVG(), COUNT(), MAX(), MIN(), COUNT(), Group By, Having Clause

Using Like, Distinct, Order By, Between...And

Comparing Nulls, Using IN/Not-In

**Session 5:****Lecture**

Relational Algebra Operations (Selection, Projection, Union, Intersect\*, Minus\*, Cross/Cartesian)

Joins (Eqvi, Inner, Outer, Natural, Cross), SQL Standard Syntax for Joins

Copying table structure/data, Sequences (AUTO\_INCREMENT)

**Lab**

Union/Union ALL

Queries on Various type of Joins using OLD and SQL Standard Syntax

Copying table structure, Copying data from one table to another

Using AUTO\_INCREMENT

**Session 6:****Lecture**

Subquery, Correlated Subquery, EXISTS/NOT EXISTS

TCL Commands (Commit/Rollback/Savepoint), DCL Commands (GRANT/REVOKE/GRANT OPTION)

Views, Types of Views, Simple and Complex Views

**Lab**

Subqueries, Correlated Queries

Using Exists/Not-Exists

Using Commit/Rollback/Savepoint

Granting/revoking privileges on database objects

Creating Views, Querying using Views

Creating Indexes

Creating Temporary Tables

**Session 7:****Lecture**

Indexes, Benefit of Indexes, Type of Indexes, Temporary Tables

ACID Properties, Concept of Database Instance and Schema

MySQL Storage Engines (InnoDB, MyISAM and others),

**Lab**

Indexes, Temporary Tables

All other SQL Commands Revision

**Session 8:****Lecture**

Introduction to MySQL Programming, Use of MySQL Programs,

Introduction to Stored Procedures, Benefits of Stored Procedures

Procedure Parameters (IN, OUT and INOUT).

**Lab**

Creating procedure without parameters

Creating Procedure with (IN/OUT/INOUT) Parameters

**Session 9:****Lecture**

Flow Control Statements (LOOP, WHILE and REPEAT)

Using above statements in Stored Procedures/Functions

Conditional Statements (IF, IF-ELSE-THEN, SWITCH CASE)

Example of each type of statement

**Lab**

Use of flow control statement in Stored Procedure

Use of conditional statements in Stored Procedure

**Session 10:****Lecture**

Loop constructs (ITERATE, LEAVE)

Functions with and without parameters

MySQL Built-in functions (string, numeric, date etc.)

**Lab**

Creating Function and returning value from it

Use of built-in functions in queries

**Session 11:****Lecture**

Cursors (Asensitive, Insensitive, Read only, Nonscrollable)

Cursors example and real time use

**Lab:**

Writing procedures with Declare, fetch and close cursor

Example of each type of cursors

**Session 12:****Lecture**

Triggers (BEFORE, AFTER), New and Old trigger variables

Trigger Examples and real time use

**Lab**

Create Before Triggers

Create After Triggers

**Session 13:****Lecture**

Error Handling and Exceptions, Types of Handler Actions, How to write Handler

Defining and handling exceptions in Stored Procedures and Functions

**Lab**

Exception handling in Stored Procedure

Exception handling with various handler actions

**Session 14:****Lecture**

Introduction to NoSQL database, Features of NoSQL Database

Structured vs. Semi-structured and Unstructured Data

Difference between RDBMS and NoSQL databases

CAP Theorem, BASE Model

Categories of NoSQL Databases: Key-Value Store, Document Store, Column-Oriented, Graph

Introduction to MongoDB, Features of MongoDB

MongoDB command interface and MongoDB compass

**Lab**

Using MongoDB Shell and Compass

**Session 15:****Lecture**

MongoDB Documents & Collections

RDBMS & MongoDB analogies: relations/tables => collections; tuples/records => documents

JSON and BSON documents

Performing CRUD (CREATE, READ, UPDATE, DELETE) Operations, UPSERT

**Lab:**

Creating database, Connecting to a database, Creating Collections

Performing CRUD operations

**Session 16:****Lecture**

MongoDB – Operators, Sorting, Indexing

Introduction to Big Data - What is Big Data, Characteristics of Big Data, Examples

Properties of Big Data, Typical Components of a Big Data system

**Lab:**

MongoDB: Complex Read Using Operators, Sorting Operations, CreatingIndexes

Demonstration of anybig data application

# Teaching Guidelines for Web Programming Technologies

## PG-DAC March 2022

---

**Duration:** 108 hours (48 classroom hours + 48 lab hours + 12 revision/practice hours)

**Objective:** To introduce the students to HTML, CSS, JavaScript, XML, JSON, Ajax, Node.js, Express.js, React, React-Redux, and practical relevance of all these technologies.

**Evaluation:** 100 marks

**Weightage:** Theory Exam – 40%, Lab exam – 40%, Internals – 20%

### Text Books:

- Fundamentals of Web Development, 1e, by Randy Connolly, Ricardo Hoar / Pearson
- MERN Quick Start Guide – Build web applications with MongoDB, Express.js, React, and Node by Eddy Wilson Iriarte Koroliova / Packt

### References:

- Internet & World Wide Web : How to Program by Paul Deitel, Henry Deitel & Abbey Deitel / Pearson Education
- XML - How to Program by Deitel et al / Pearson Education
- Ajax in Action by Dave Crane, Eric Pascarello / Dreamtech Press
- JavaScript: The Good Parts by Douglas Crockford / O'Reilly
- Pro MERN Stack: Full Stack Web App Development with Mongo, Express, React, and Node by Vasan Subramanian / Apress
- Web Application Security: A Beginner's Guide by Bryan Sullivan & Vincent Liu / Tata McGraw Hill
- W3Schools Tutorials [ <https://www.w3schools.com/> ]
- Mozilla Developer Network Web Development Tutorials [ [https://developer.mozilla.org/en-US/docs/Learn/Getting\\_started\\_with\\_the\\_web](https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web) ]
- Curated Tutorial Links on ES6, React, etc. [ <https://github.com/markerikson/react-redux-links> ]

---

(Note: Each Session is of 2 hours)

### Session 1: Architecture of Web

#### Lecture:

- Brief history of the Internet
- How does the Internet work?
- Internet Protocol; HTTP
- Domain Names; Domain Name Service servers
- HTTP Protocols
  - Difference between HTTP 1.0, HTTP 1.1, and HTTP 2.0
  - Methods – GET, POST, HEAD, PUT, DELETE, etc.
  - Status codes
  - Stateless nature of the protocol and HTTP Session
  - HTTPS
- Architecture of the Web
- Web servers – IIS, Apache server

**Lab:**

- Exploring different browsers
  - Mozilla Firefox, Google Chrome, Safari
- Exploring different text editors
  - Windows: Notepad++, Linux: Gedit or Vim or Emacs

**Session 2: HTML****Lecture:**

- Introduction to HTML5
- Introduction to basic HTML Tags
  - Alignment, Headings, Anchor, Paragraph, Image, Lists, Tables, and iFrames
- HTML5
  - New features in HTML5
  - New elements, new attributes, link relations, microdata, ARIA accessibility, objects, events, and Canvas tags
  - HTML5 Validation
  - Audio & Video Support
  - Geo-location Support
- HTML Forms & Controls
  - Input, Text Area, Radio Button, Checkbox, Dropdown, Submit, Reset, Button, etc.
- Introduction to Document Object Model (DOM)

**Lab:**

- Create a HTML form for building your resume.

**Session 3: Cascading Style Sheets (CSS)****Lecture:**

- Introduction to CSS, Styling HTML with CSS, Structuring pages with CSS,
- Inline CSS, Internal CSS, External CSS, Multiple styles, CSS Fonts
- CSS Box Model
- id Attribute, class Attribute
- HTML Style Tags
- Linking a style to an HTML document

**Lab:**

- Apply inline, internal, and external CSS to change colors of certain text portions, bold, underline, and italics certain words in the previously created HTML resume form.

**Session 4: JavaScript****Lecture:**

- Introduction to JavaScript
- Variables in JavaScript
- Statements, Operators, Comments, Expressions, and Control Structures
- JavaScript Scopes
- Strings, String Methods
- Numbers, Number Methods
- Boolean Values
- Dates, Date Formats, Date Methods
- Arrays, Array Methods

**Lab:**

- Practice writing basic JavaScript programs for better understanding of the language constructs

## Sessions 5 & 6: JavaScript

### Lecture:

- Objects, Object Definitions, Object Properties, Object Methods, Object Prototypes
- Functions, Function Definitions, Function Parameters, Function Invocation, Function Closures
- Introduction to Object Oriented Programming in JS
  - Method, Constructor, Inheritance, Encapsulation, Abstraction, Polymorphism

### Lab:

- Write a JavaScript program to sort a list of elements by implementing a sorting algorithm.
- Write a JavaScript program to list the properties of a JavaScript object.

## Sessions 7 & 8: JavaScript

### Lecture:

- Document Object Model (DOM)
  - Object hierarchy in JavaScript
  - HTML DOM, DOM Elements, DOM Events
  - DOM Methods, DOM Manipulation
- Forms, Forms API, Forms Validation
- Regular Expressions
- Errors, Debugging
- Introduction to Browser Dev Tool
- Pushing code quality via JSLint tool

### Lab:

- Write a JavaScript function to get First and Last name from the previously created Resume form
- Validate the entire Resume form using client-side JavaScript
- Write a JavaScript function to validate whether a given value is RegEx or not.

## Session 9: jQuery

### Lecture:

- Introducing to jQuery
- jQuery selectors
- jQuery events
- jQuery animation effects
- jQuery DOM traversal and manipulation
- Data attributes and templates
- jQuery DOM utility functions
- jQuery plugins

### Lab:

- Write a jQuery program to get a single element from a selection of elements of a HTML page.
- You are having sample data for the link. Write jQuery code to change the hyperlink and the text of an existing link.
- Write a jQuery program to attach a click and double-click events to all `<p>` elements.
- Write a jQuery program to hide all headings on a page when they are clicked.
  - Also find the position of the mouse pointer relative to the left and top edges of the document.

## Sessions 10 & 11: JSON & Ajax

### Lecture:

- JSON: JavaScript Object Notation (JSON)
  - Introduction and need of JSON
  - JSON Syntax Rules

- JSON Data - a Name and a Value,
- JSON Objects, JSON Arrays, JSON Files
- JSON parsing
- Ajax
  - Introduction to Ajax
  - Ajax Framework
  - Ajax Architecture
  - Web services and Ajax
  - Ajax using JSON and jQuery

**Lab:**

- Create a page showing live score/feed using Ajax and JSON from a live sport/news service endpoint given by the faculty

### Session 12: Introduction to Node.js

**Lecture:**

- Introduction to Node.js
- Browser JS vs. Node.js
- ECMAScript 2015 (ES6)
- Node.js REPL

**Lab:**

- Install Node.js 12.x.x LTS version on your machine
- Write a recursive function in Node.js
- Write a Node program that prints all the numbers between 1 and 100, each on a separate line.

A few caveats:

- if the number is divisible by 3, print "foo"
- if the number is divisible by 5, print "bar"
- if the number is divisible by both 3 and 5, print "foobar"

### Sessions 13 & 14: Node.js Asynchronous Programming

**Lecture:**

- Introduction to Asynchronous programming and callbacks
- Promises and async & await
- The Event Loop and Timers

**Lab:**

- Assignment on JavaScript callback functions
- Assignment on Timers, Promises, and Async & Await

### Session 15: Node.js Modules

**Lecture:**

- Understanding Node modules, exports, and require
- Introduction to npm
  - package.json and package-lock.json files
  - Install, update, and manage package dependencies
  - Local and global packages

**Lab:**

- Create a module and import it in other programs
- Install a module/package using npm

### Session 16: Node.js Modules – *fs* and *http*

**Lecture:**

- File I/O – Sync & Async Methods

- HTTP Module – Building an HTTP server
- Developing a Node web application

**Lab:**

- Write a program to create a new file and write some content to it in synchronous mode and read and display file contents on standard output in async mode
- Build a simple Node.js web application serving both HTTP GET and POST methods

**Session 17: Introduction to Express****Lecture:**

- Introduction to Express
- Getting started with Express
- Application, Request and Response Objects
- Routes and Middlewares
- Templates, Template Engines, and Rendering Views

**Lab:**

- Use Node and Express to write a simple web application that consists of at least 5 route implementations
- Rebuild any previous Node assignment using Express and a template engine

**Session 18: Introduction to React****Lecture:**

- Introduction to React
- Getting started with React
- React Elements and React Components
- Function and Class Components
- Working with React Components and Props
  - Compose components
  - Render components
  - Declutter components

**Lab:**

- Rebuild any previous plain HTML lab assignment using React
- Build a React Clock app showing time (hh:mm:ss) of any three countries

**Sessions 19 & 20: React****Lecture:**

- Introduction to State and Lifecycle
- Stateful components and lifecycle methods
- Props vs. State vs. Context
- Handling events
- Conditional rendering

**Lab:**

- Implement the following items in the React Clock app
  - Update the time (hh:mm:ss) using State and Lifecycle methods
  - Add a close function on each rendered clock component
  - Assign background color of rendered clock components based on AM, PM

**Session 21: React****Lecture:**

- Lists and Keys
  - Rendering Multiple Components

- Basic List Component
- Working with forms and inputs
- Refs and the DOM
- Lifting state up

**Lab:**

- Implement and integrate a new feature in the React Clock app where one can select a country time zone from dropdown list and click on “Add” button to render it.

**Session 22: React****Lecture:**

- Error Boundaries
- Composition vs. Inheritance
  - Containment
  - Specialization
- Thinking in React

**Lab:**

- Implement error boundaries at appropriate places in the React Clock app

**Session 23: Introduction to React-Redux****Lecture:**

- Introduction to Redux
- Actions, Reducers, and Stores
- Usage with React

**Lab:**

- Make necessary changes in the design and implementation of React Clock app using React-Redux to maintain the application state.

**Session 24: Responsive Web Design & Web Security****Lecture:**

- Introduction of UI Scripting
- The Best Experience for All Users
  - Desktop, Tablet, Mobile
- Bootstrap
  - Overview of Bootstrap, Need to use Bootstrap
  - Bootstrap Grid System, Grid Classes, Basic Structure of a Bootstrap Grid
  - Typography
  - Components – Tables, Images, Jumbotron, Wells, Alerts, Buttons, Button Groups, Badges/Labels, Progress Bars, Pagination, List Groups, Panels, Dropdowns, Collapse, Tabs/Pills, Navbar
  - Forms, Inputs
  - Bootstrap Themes, Templates
- Web Security
  - Introduction to Web security
  - SQL Injection, Cross-Site Scripting (XSS)
  - JSON and Security Concerns, Cross Site Request Forgery (CSRF), Injection Attacks
  - Security Standards (OWASP)

**Lab:**

- Update the design of the Resume form using Bootstrap

## Teaching Guidelines for Web-based Java Programming

PG-DAC March 2022

---

**Duration:** 100 hours (46 class room hours + 44 lab hours + 10 revision/practice hours)

**Objective:** To learn advanced concepts in java programming and perform web Programming using Java.

**Prerequisites:** Knowledge of core Java programming

**Evaluation:** 100 marks

**Weightage:** Theory exam – 40%, Lab exam – 40%, Internals – 20%

### **Text Book:**

- Core and Advanced Java Black Book / Dreamtech Press

### **References:**

- Servlet and JSP: A Tutorial by Budi Kurniawan / Brainy Software
  - Spring in Action by Craig Walls / Manning Publications
  - Advanced Java programming by Uttam K Roy / Oxford University press
  - Sun Certified Enterprise Architect for Java EE Study Guide by Mark Cade & Humphrey Sheil / Pearson Education
  - Professional Java EE Design Patterns by Murat Yener, Alex Theedom & Reza Rahman / Wrox
- 

(Note: Each Session is of 2 hours)

### **Session 1:**

#### **Lecture:**

##### J2EE Overview

- J2EE Container
- Packaging Web applications
- J2EE compliant web application
- Deployment tools.
- Web application life cycle
- Deploying web applications.
- Web Services Support

#### **No Lab**

### **Sessions 2, 3 & 4:**

#### **Lecture:**

- Servlets: Dynamic Content Generation
- Advantages of Servlets over CGI
- Servlet Life cycle
- Servlet API & Deployment
- Servlet Annotations
- The Servlet interface

- The HttpServlet, HttpServletRequest, HttpServletResponse
- Exception Handling
- Servlet, DAO, POJO DB Layers
- Session
- Session Management
- Session Tracking with
  - Cookies
  - HttpSession
- Request Dispatcher
- Page Navigation
- Complete Case study Servlet Based

**Lab:**

- Installing a servlet container (Tomcat)
- Adding Server to IDE
- Develop a structured dynamic web application(e.g. Library Management System) using servlets, deploy it in Tomcat
- Use HTTP Session in the Air Ticket Reservation System

*Reading:* Know more about the HTTP protocol at [www.w3c.org](http://www.w3c.org)

*Tutorial:* Compare which way of session tracking is better Cookies or HttpSession.

**Sessions 5 & 6:**

**Lecture**

- JSP: Separating UI from Content generation code
- MVC architecture
- Design Pattern: MVC Pattern
- Life cycle of a JSP page
- Directives, Implicit and Explicit Objects, Scriptlets, Expressions, Expression Language
- Scope
- JSP Error Page handling
- JSTL

**Lab:**

- Separate UI code from the controller code in your Library Management System by incorporating JSP and Servlets.
- Complete the implementation of Air Ticket Reservation System.
- Implement MVC based web application using Servlet, JSP

**Sessions 7 & 8:**

**Lecture:**

JDBC & Transaction Management

- Introduction to JDBC API
- JDBC Architecture
- JDBC Drivers
- JDBC Classes& Interfaces: Driver, Connection, Statement, PreparedStatement, ResultSet and their relationship to provider implementations
- Stored procedures and functions Invocation
- SQL Injection overview and prevention

- Design Pattern: Data Access Object Pattern

**Lab:**

- Add Database CRUD operations to above MVC web application using JDBC Classes and interfaces. Use DAO and POJO Layers

**Sessions 9, 10 & 11:****Lecture:**

- Hibernate Framework
  - Introduction to Hibernate Framework
  - Architecture
- Hibernate in IDE
  - Creating web application using Hibernate API
  - Lifecycle of Hibernate Entities
- HB with annotation example
- Hibernate Mappings and Relationships
- Collection and Component Mapping
- HQL, Named Queries, Criteria Queries

**Lab:**

- Demonstrate Hibernate as standalone library in Java application
- Develop a web application (Online Bookshop) using Hibernate Persistence

*Reading:* Study Hibernate architecture from [www.hibernate.org/docs](http://www.hibernate.org/docs)

**Sessions 12, 13 & 14:****Lecture:**

- What is Spring Framework
- Overview of Spring Architecture
- Spring MVC architecture
- Spring Modules Overview
- Understanding Spring 4 annotations (Basic Introduction)
- What is IoC (Inversion of Control)
- IOC container
- Dependency Injection
- Spring Beans
- Autowiring Beans
- Bean Scopes
- Spring MVC
- Model, Model & View, HandlerMapping, ViewResolver
- Design Pattern: Front Controller Pattern
- Spring MVC Web application with JSP views (without Spring Boot)
- Using Thymeleaf as alternate View Technology (only introduction)
- Spring Validations
- Spring i18n, Localization, Properties
- File Upload example

**Lab:**

- Design and deploy Library Management System using Spring Web

## Sessions 15 & 16:

### Lecture:

- Spring Boot essentials
- Why Spring boot
- Spring Boot Overview
- Basic Introduction of MAVEN
- Building Spring Web application with Boot
- Spring Boot in detail (Use Spring Boot for all demo & assignments here onwards)
- Running a web application using Spring Boot with CRUD (with Static Data not DB)
- Spring Data JDBC

### Lab:

- Create Hello World Spring Boot Web application
- Check Libraries imported by Spring Boot
- Create Spring Boot CRUD application with Thymeleaf as View technology and Spring JDBC

## Sessions 17 & 18:

### Lecture:

#### Spring Data Module

- Spring Data JPA (Repository support for JPA)
- Crud Repository & JPA Repository
- Query methods
- Using custom query (@Query)

### Lab:

- Add CRUD operations with Spring JPA etc. to earlier Spring Web application.

## Session 19:

### Lecture:

#### Spring AOP

- AOP Overview
- Spring AOP
- AOP Terminology and annotations: Advice, Join Points, Pointcuts, Aspects

### Lab

- Modify earlier Spring MVC application to Log all the requests using AOP

## Sessions 20 & 21:

### Lecture:

#### Building REST services with Spring

- Introduction to web services
- SOAP Vs RESTful web services
- RESTful web service introduction
- Create RESTful web service in java using Spring Boot
- RESTful web service JSON example
- RESTful web service CRUD example
- Using POSTMAN client to invoke REST API's
- REST service invocation using REST Template

### Lab:

- Create REST API for Employee Management using Spring Boot

- Invoke it from POSTMAN app
- Invoke it from another Spring Boot Web application using REST Template

### Session 22 & 23:

#### Lecture:

#### Testing in Spring

- Testing in Spring
- Unit Testing of Spring MVC Controllers
- Unit Testing of Spring Service Layer
- Integration Testing of Spring MVC Applications: REST API
- Unit Testing Spring MVC Controllers with REST

#### Securing Web Application with Spring Security

- What is Spring Security
- Spring Security with Spring Boot
- Basic Authentication
- Authentication with User credentials from Database and Authorization

#### Lab:

- Design & Test Spring Application
- Secure the Spring Web application created in earlier exercise.

## Teaching Guidelines for **MS.Net Technologies**

PG-DAC March 2022

---

**Duration:** 84 hours (42 class room hours + 32 lab hours + 10 revision/practice hours)

**Objective:** To acquire the knowledge of Microsoft.NET Framework 4.7 or higher version.

**Prerequisites:** Students are expected to know any OOP. They should have undergone the Web Programming module which includes HTML, CSS, JavaScript, JSON, and XML. Knowledge of any database is required.

**Note:** Training will be carried out on .Net 4.7 or latest version of the software

**Evaluation:** 100 marks

**Weightage:** Theory exam – 40%, Lab exam – 40%, Internals – 20%

### **Text Book:**

- Pro C# 8 with .Net Core - Foundational Principles and Practices in Programming by Andrew Troelsen & Philip Japikse / Apress

### **References:**

- C# 8 and .Net Core 3.0 - Modern Cross-Platform Development by Mark J. Price / Packt
- 

(Note: Each Session is of 2 hours)

### **Session 1:**

#### **Lecture:**

Introduction to the .Net Framework

Intermediate Language (IL)

Assemblies and their structure, EXEs/DLLs

CLR and its functions

- JIT Compilation
- Memory Management
- Garbage Collection
- AppDomain Management
- Memory Management
- CLS, CTS
- Security

#### **NO LAB**

### **Session 2:**

#### **Lecture:**

.Net Framework, .Net Core, Mono, Xamarin differences

Versions of the Framework

Managed and Unmanaged Code

Introduction to Visual Studio

Using ILDASM

#### **NO LAB**

### Session 3:

#### Lecture:

Console Applications and Class Libraries (Framework and .Net Core)

C# Basics

Project References, using

Classes

Data Types in .net and CTS equivalents

Methods

- Method Overloading
- Optional Parameters
- Named Parameters and Positional Parameters
- Using params
- Local functions

Properties

- get, set
- Readonly properties
- Using property accessors to create Readonly property

Constructors

Object Initializer

Destructors

Discussion on IDisposable. To be implemented after interfaces

#### Lab:

Create a class that has Properties, Fields, Methods, Constructors (Trainer can specify any class of his choice, e.g. Student, Employee, etc)

### Session 4:

#### Lecture:

Static Members of a Class

- Fields
- Methods
- Properties
- Constructors

Static Classes

Static local functions

Inheritance

- Access Specifiers
- Constructors in a hierarchy
- Overloading in derived class
- Hiding, using new
- override
- sealed methods
- Abstract Classes
- Abstract Methods
- Sealed Classes

#### Lab:

Create multiple classes that use Inheritance based concepts

### Session 5:

#### Lecture:

Interfaces

- Implementing an interface
- Explicitly implementing an interface
- Inheritance in interfaces
- Default interface methods

Operator overloading

**Lab:**

Create and implement interfaces for the classes created in Lab 4

Implement IDisposable, IComparable

**Session 6:**

**Lecture:**

Reference and Value Types

Value Types

- struct
- enum

out and ref

nullable types

nullable reference types

?? and ??=

Working with Arrays (single, multidim, jagged), Array Class members

Indices and ranges

Indexers

**Lab:**

Lab based on array examples.

Also create an array of the class created in Lab 1.

**Session 7:**

**Lecture:**

Generic classes

Generic methods

Generic Constraints

Collections – generic and non-generic

Collection Examples based on ICollection, IList, IDictionary (both generic and non-generic)

Iterating collections using foreach

Using Tuples to pass multiple values to a function

**Lab:**

Lab based on collection examples.

Also create a collection of the class created in Lab 1.

**Session 8:**

**Lecture:**

Delegates

- Calling methods using delegates
- Uses of delegates
- Multicast delegates
- Action, Func, Predicate delegates

Anonymous methods

Lambdas

**Lab:**

Lab based on delegates examples.

**Session 9:****Lecture:**

Error Handling (Exceptions Handling)

- Checked & Unchecked Statements
- The try, catch, finally
- Dos & Don'ts of Exception Handling

User Defined Exception classes

Declaring and raising events

Handling events

Async calls using delegates

**Lab:**

Lab based on exceptions and events examples.

**Session 10:****Lecture:**

Anonymous types

Extension methods

Partial classes

Partial methods

LINQ to objects

Writing LINQ queries

Deferred execution

LINQ methods

PLINQ

**Lab:**

Lab based on LINQ examples

Students to try tutorial for 101 LINQ Queries

**Session 11:****Lecture:**

Creating a shared assembly

Creating Custom Attributes

Using Reflection to explore an Assembly

Using Reflection to load an Assembly dynamically

Files I/O and Streams

- Working with drivers, Directories, and Files
- Reading and Writing files

**NO LAB****Session 12:****Lecture:**

Threading

- ThreadStart, Parameterized ThreadStart
- ThreadPool
- Synchronizing critical data using lock, Monitor and Interlocked

Working with Tasks

- Calling functions with and without return values
- Using async, await

Using the Task Parallel Library

**Lab:**

Threading related examples

Task related examples

### Sessions 13-18:

#### Lecture:

#### Introduction to Asp.Net MVC

- Architecture of an ASP .Net MVC application
- Understanding Folder structures and configuration files

#### Understanding Controllers and Action

- Create a controller
- How actions are invoked
- `HttpGet` , `HttpPost` , `NoAction` Attributes
- Running Action result.

#### Understanding Views & Models

- Creating Models & ViewModel
- Creating Razor Views
- HTML Helper Functions
- Understanding ViewBag
- Create a view using ViewBag
- Validation using Data Annotations
- Client side and server side validation
- Self validated model
- Creating Strongly Types Views
- Using Various Scaffold Templates
- CRUD operation using Model

#### MVC State Management

- `ViewBag` , `TempData` , `Session` , `Application`
- Cookies , `QueryString`

#### MVC Module

- Partial View
- Action Method and child action

#### Data Management with ADO.NET

- `Microsoft.Data.SqlClient` introduction
- Connection object, Command object, `DataReader`, `DataAdapter`, `DataSet` and `DataTable`.
- Asynchronous command Execution
- Asynchronous Connections

#### Understanding Routing & Request Life Cycle

- Routing Engine & Routing Table
- Understanding and configuring Routing Pattern in `RouteConfig` File
- Understanding 404 error and resource not found.
- Using Attributes Routing
- Understanding Request Life Cycle

#### Layouts , Bundle , Minification

- Creating Layout and using with associated views
- Understanding Bundling and Minification
- Using `BundleConfig` file
- Attaching css , js , bootstrap in bundles
- Custom Helper Function
- Asynchronous Actions
- Error Handling in MVC with Log Entry
- Filters and Custom Action Filter

### MVC Security

- Using Authorize & Allow Anonymous attributes
- Implementing Forms Based Authentication
- Preventing Forgery Attack using AntiForgeryToken
- Preventing Cross Site Scripting Attack
- Intro of OAuth (Demo only)

### Entity Framework

- Introduction to EF
- Different Approaches
- Using Code First Approach
- Using various Data Annotations
- Using Validation, Primary Key , Foreign Key etc
- Using Fluent APIs
- Database Migrations
- CRUD operation using EF

### Developing MVC application using EF Code First Approach

#### Understanding ASP.Net MVC Core

- Difference between MVC and MVC Core
- Creating a simple MVC Core Application

#### Lab:

Lab exercise covering the concept covered in the class

### Session 19:

#### Lecture:

#### Localization in MVC (Demo Only)

- Cultures and regions
- Understand Culture Type
- Server side culture declaration
- Client Side culture declaration
- Asp .Net 4.5 MVC recourse files
- Making use of local resources
- Making use of global resources
- Looking at the resource editor

#### Deploying ASP .NET MVC application (Demo only)

- Configuring IIS for ASP .Net MVC
- Bin Deploying an Asp .NET MVC application

#### NO LAB

### Session 20:

#### Lecture:

#### Windows Communication Foundation (Demo Only)

- WCF Overview
- Service Contracts
- Data Contracts
- Message Contracts
- Operation Contract and Channel Shapes
- Channel Listeners
- Channel Factories
- ChannelFactory<>

- ICommunicationObject
- Binding
- wsHttpBinding ,ws2007HttpBinding, wsDualHttpBinding
- Cross-Machine Communication Between .NET Applications
- Local Machine Communication Between .NET Applications
- Communication Using Basic Web Services
- Communication Using Advanced Web Services
- Exposing a Service Contract over Multiple End Points
- Exporting and Publishing Metadata (Service Behavior)
- Implementing Transactions (Operation Behavior )
- Hosting a Service in Windows Process Activation Services
- Hosting a Service in IIS 7
- Self-Hosting in a Managed Windows Service
- Defining Service and Endpoint Addresses
- Creating REST Service with Get & Post

**NO LAB****Session 21****Lecture:****Web APIs**

- Creating ASP.NET MVC Web API
- Different Verbs
- Consuming using a client
- Using Newtonsoft APIs

**Lab:**

Create a RESTful service using WEB API. Create a consumer.

# Teaching Guidelines for Software Development Methodologies

## PG-DAC March 2022

---

**Duration:** 84 hours (38 class room hours + 38 lab hours + 8 revision/practice hours)

**Objective:** To build knowledge of Software development methodologies.

**Evaluation:** 100 marks

**Weightage:** Theory exam – 40%, Lab exam – 40%, Internals – 20%

**Text Book:**

- Software Engineering by Chandramouli / Pearson

**References:**

- Software engineering by Ian Sommerville / Pearson
- Clean Code: A Handbook of Agile Software Craftsmanship by Robert C. Martin / Prentice Hall
- The Mythical Man-Month: Essays on Software Engineering by Frederick P. Brooks Jr. / Addison Wesley
- User Stories Applied: For Agile Software Development by Mike Cohn / Addison Wesley
- DevOps: Continuous Delivery, Integration, and Deployment with DevOps by Sricharan Vadapalli / Packt
- Git for Teams by Emma Westby / O'Reilly

---

(Note: Each Session is of 2 hours)

### Git (8 hours)

#### Sessions 1 & 2

##### Lecture

- Developing an application in a team
- Issues developers face when working in a team
- Introduction to code versioning system
- History of code versioning system
  - Different tools available for versioning
  - Software development workflow
- Introduction to git
- Introduction to git repository and git structure
- Adding code to git
- Creating and merging different git branches

##### Lab

- Create a local git repository
- Commit the initial code
- Update the code
- Use git commands to
  - Get the updated files
  - List the changes
  - Create branch
  - Merge branch

## Software Engineering (20 hours)

### Sessions 3, 4 & 5

#### Lecture

- Introduction to software engineering
  - Software Process
  - Software Process Model
  - Software Product
- Importance of Software engineering
- Software Development Life Cycles
- Requirements Engineering
  - Types of Requirements
  - Steps involved in Requirements Engineering
  - Requirement Analysis Modelling
- Design and Architectural Engineering
  - Characteristics of Good Design
  - Function Oriented vs Object Oriented System
  - Modularity, Cohesion, Coupling, Layering
  - Design Models
  - UML
- Coding
  - Programming Principles
  - Coding Conventions
- Object Oriented Analysis and Design

#### Lab

- Prepare software requirement specification for the final project
- Create the initial use-cases, activity diagram and ER diagram for the final project

### Sessions 6 & 7

#### Lecture

- Introduction to Agile development model
- Agile development components
- Benefits of Agile
- Introduction to different tools used for agile web development
- Introduction to Atlassian Jira
  - Add Project
  - Add Tasks and sub-tasks
  - Create sprints with tasks
- Case study of developing web application using agile methodology

#### Lab

- Create different sprints in Atlassian Jira for different features

## DevOps (20 hours)

### Sessions 8 & 9

#### Lecture

- Introduction to Microservices
- Microservices Architecture
- Fragmentation of business requirement
- Deployment pattern
- API gateway

- Service Discovery
- Database Management for Microservices

**Lab**

- Create Microservices

**Sessions 10 & 11****Lecture**

- Introduction to DevOps
- DevOps ecosystem
- DevOps phases
- Introduction to containerisation
- Introduction to docker
- Creating docker images using Dockerfile
- Container life cycle

**Lab**

- Install and configure docker
- Create docker image using Dockerfile
- Start docker container
- Connect to docker container
- Copy the website code to the container
- Use docker management commands to
  - List the images
  - List the containers
  - Start and stop container
  - Remove container and image

**Session 12****Lecture**

- Introduction to YAML
- Introduction to Docker Swarm and Docker Stack
- Introduction to Kubernetes
- Creating Kubernetes cluster
- Creating service in Kubernetes
- Deploying an application using dashboard

**Lab**

- Configure Kubernetes
- Configure Kubernetes Dashboard
- Setup a Kubernetes cluster
- Access application using Kubernetes service
- Deploy the website using Dashboard

**Testing & Integration (20 hours)****Session 13****Lecture**

- Introduction to software testing
- Why testing code is important
- Verification and validation
- Quality Assurance vs Quality Control vs Testing
- Principles of software testing

**Lab**

- Read more testing concepts used in the industry

**Session 14****Lecture**

- Introduction to STLC and V Model
- Types of testing: manual and automation
- Tools used for automation testing
- Introduction to testing methods: white-box, black-box and grey-box
- Introduction to functional testing: (\* students are supposed to learn the concepts)
- Introduction to non-functional testing: (\* students are supposed to learn the concepts)

**Lab**

- Create a test plan for project
- Document the use cases
- Create test case document for different sprints (designed in SE)

**Sessions 15 & 16****Lecture**

- Introduction to Selenium (use Eclipse IDE)
- Load web driver
- Create selense commands: locators: by ID, name, class, tag name, XPath
- Add interactions: text box, radio button selection, check box selection, drop down item selection, keyboard actions, mouse actions, multi select

**Lab**

- Download and configure Selenium
- Create a test suite
- Add commands and interactions

**Session 17****Lecture**

- Introduction to delivery pipeline
- Introduction to Jenkins
- Jenkins management
- Adding slave node to Jenkins
- Building a delivery pipeline
- Selenium integration with Jenkins

**Lab**

- Install and configure Jenkins
- Build a pipeline job using Jenkins
- Create a maven project for Selenium
- Add Selenium test suite in the project
- Integrate it with Jenkins

**Cloud (8 hours)****Session 18****Lecture**

- Introduction to Cloud
- Introduction to Virtualization
- Virtualization types: type1, type2

- Cloud Computing, Cloud SPI Model, Cloud Computing Types (Public, Private and Hybrid), Cloud Security (SLA and IAM).
- Virtualization, Hardware Virtualization, Para-Virtualization, Cloning, Snapshot and Template
- Containerization, Operating System Virtualization

**Lab**

- Create and configure VM using VBox
- Deploy code on VM

**Session 19****Lecture**

- Cloud architecture
- Service models: IaaS, PaaS, SaaS
- Deployment models: Private, Public, Hybrid
- Services provided by Cloud (Compute, Database, Developer Tools, Storage, Media, Mobile, Web, Security, Integration etc.)
- Cloud development best practices
- Introduction to AWS
- Services provided by AWS: EC2, Lambda, S3

**Lab**

- Create AWS EC2 instance
  - Add Storage, Tag Instance, Review Instance Launch
  - Set up an Apache web server on your EC2 instance
  - Clean up your EC2 Instance

Teaching Guidelines for  
**General Aptitude & Communication**  
PG-DAC March 2022

---

**Duration:** 80 hours (Classroom hours + Practice sessions)

**Prerequisites:** Knowledge of Mathematics & English.

**Evaluation:** Grading based on combined marks of Aptitude and Communication

**Weightage:** Aptitude - 40% ; Communication - 60%

---

## General Aptitude

**Duration:** 32 Class room hours + Practice sessions

**Objective:** To reinforce knowledge of general aptitude

**Evaluation:** Internal Tests (40 marks)

**Reference Books:**

- Quantitative Aptitude by R S Aggarwal / S Chand
- Verbal & Non-Verbal Reasoning: R S Aggarwal / S Chand
- Quantitative Aptitude - Quantum CAT : Sarvesh K Verma / Arihant
- How to prepare GRE by Barron's / Galgotia
- Magic Book on Quicker Math by Manoj Tyra / BSC

Website to refer: [www.indiabix.com](http://www.indiabix.com)

---

(Note: Each Session is of 2 hours)

**Session 1:**

- Percentage
- Profit & Loss

**Session 2:**

- Ratio & Proportion

**Session 3:**

- Average
- Mixture & Alligation

**Session 4:**

- Simple Interest & Compound Interest

**Session 5:**

- Number Systems
- Series, Cyclicity & Remainders

**Session 6:**

- Data Interpretation
- Syllogism

**Session 7:**

- Coding & Decoding
- Blood Relations

**Session 8:**

- Seating Arrangements (Linear & Circular)

**Session 9:**

- Ages
- Puzzles

**Session 10 & 11:**

- Time, Speed & Distance
- Trains, Boats & Streams

**Session 12:**

- Time & Work
- Wages (Man days)
- Pipes & Cisterns

**Session 13:**

- Clocks

**Session 14:**

- Permutations & Combinations

**Session 15:**

- Probability

**Session 16:**

- Calendar

## Effective Communication

**Duration:** 48 Class room hours + Practice sessions

**Objectives:**

- To speak in English confidently
- To learn good writing and presentation skills
- To prepare for and succeed in Interviews

**Evaluation:** Internal Tests, Writings, Presentations, Activities & Sessions (60 marks)

**Reference Books:**

- Professional Communication Skills by AK Jain, PSR Bhatia & AM Shaikh / S. Chand
- Communication Skills by Sanjay Kumar & Pushp Lata / Oxford
- High School English Grammar & Composition by Wren & Martin / S. Chand
- English is Easy by Chetan Anand Singh / BSC
- Oxford Guide to English Grammar by John Eastwood / Oxford
- Business Communication by H S Mukerjee / Oxford
- Effective Business Communication by Asha Kaul / Prentice Hall

---

(Note: Each Session is of 2 hours)

**Session 1:**

Fundamentals of Communication

- Process of communication
- Types of communication
- Effective communication

**Session 2:**

The Art of Communication

- Vocabulary, spelling and grammar
- Fluency, pronunciation, intonation and accent

***Practice Sessions:***

*Practise words, spelling, intonation and correct pronunciation*

**Session 3:**

Personality Development

- Greeting
- Formal dressing & etiquettes
- Body language

**Session 4:**

Personality Development

- Developing positive attitude
- Confidence building
- Questioning techniques
- Leadership

**Practice Sessions:**

*Practise greeting, etiquettes and questioning*

**Session 5:**

English Grammar

- Nouns
- Pronouns
- Adjectives
- Articles

**Session 6:**

English Grammar

- Verbs
- Adverbs
- Prepositions
- Conjunctions

**Practice Sessions:**

*Practise sentence making*

**Session 7:**

English Grammar

- Active and passive voices
- Direct and indirect speeches

**Session 8:**

English Grammar

- Idioms
- Synonyms & Antonyms

**Practice Sessions:**

*Practise speaking in active & passive voices*

*Practise direct & indirect speaking*

*Practise idioms, synonyms & antonyms*

**Session 9:**

Correct Usage of English

**Session 10:**

Common Mistakes in English Communication

**Practice Sessions:**

*Practise General English Communication*

**Session 11:**

Listening Skills

- Importance of listening
- Techniques for effective listening

### Session 12:

#### Listening Skills

- Audio synthesis
  - Listening to audio clips
  - Question-answers based on the listened audio clips

#### *Practice Sessions:*

*Practise audio synthesis*

### Session 13:

#### Reading Skills

- Comprehension
  - Techniques

#### *Practice Sessions:*

*Comprehension exercises*

### Session 14:

#### Writing Skills

- Essay writing
  - Characteristics of a good essay
  - Types of essays
  - Structure of an essay (introduction, main body, conclusion)

### Session 15:

#### Writing Skills

- Letter writing
  - Types of letters
  - Parts of a letter
- Official emailing
  - Structure and etiquettes of email writing
  - Tips to write an impressive email

### Session 16:

#### Writing Skills

- Report writing
  - Synopsis
  - Introduction
  - Analysis (current situation, identify problems, solutions)
  - Conclusion & recommendation
  - References

#### *Practice Sessions:*

*Essay writing*

*Letter writing*

*email writing*

*Report writing*

### Session 17:

#### Public Speaking

- Managing stage fear
- Speech design
- Informative speeches
- Speeches for special occasions (Introduction, Welcome, Felicitation, Thanks, etc)
- Extempore & impromptu speeches

#### **Practice Sessions:**

*Conduct various types of speeches*

### Session 18:

#### Presentation Skills

- How to conduct effective and engaging presentations?
- Organisation & structure of presentation
- Design of slides in PPT
- Body language & voice

#### **Practice Sessions:**

*Conduct presentations using PPT*

*Feedback of presentations*

### Session 19:

#### Group Discussions

- What is a GD?
- Skills assessed in GD
- Common mistakes
- Common GD topics

#### **Practice Sessions:**

*Conduct practice GDs with video recording*

*Playing and analysis of GDs conducted*

### Session 20:

#### Interpersonal Skills

- Negotiation
- Persuasion
- Influencing

### Session 21:

#### Personal Interviews

- Preparation for Interview
  - Qualities interviewers looking for
  - Getting ready for Interviews
  - Company Research
  - Overall approach
  - Just before interview

### Session 22:

#### Personal Interviews

- Introducing yourself
  - Importance of introduction
  - Structure of introduction

#### *Practice Sessions:*

*Practise introduction*

*Analysis and feedback on introduction*

### Session 23:

#### Personal Interviews

- Facing job interviews
  - Confidence
  - Body language
  - Right mindset

### Session 24:

#### Personal Interviews

- Tips for facing Interviews
  - What to do (and not do) during interviews?
  - Best practices and common mistakes of answering questions

#### *Practice Sessions:*

*Practise common technical questions*

#### *Practice Sessions:*

*Practise common HR/behavioral questions*

#### *Practice Sessions:*

*Conduct mock interviews*

## Teaching Guidelines for Software Project

### PG-DAC March 2022

---

**Duration:** 120 hours

**Objective:** In addition to the specific subject knowledge, the Software Project module attempts to put into practice a number of things that the students have learned during the PG-DAC course, such as:

- Ability to work in a team
- Software development methodology and principles
- Good programming practices
- Technical reporting and presentation.

**Prerequisites:** Completion of the basic modules on Programming, Data Structures, and Database to start Phase I of the Project.

**Evaluation:** Grading based on the combined marks obtained in the evaluations of all the 3 phases.

**Weightage:** Phase I – 10%, Phase II – 10%, Phase III – 80% (Mid Evaluation 20% + Final Evaluation 60%)

---

### Final Project Schedule

Students, in teams, will be required to identify project topics in consultation with faculty members within the first two months of the course.

The Project module is divided in three phases.

#### I – SRS Phase:

*Tasks:* Requirements gathering, feasibility study and project thinking.

*Deliverable:* Software Requirement Specification (SRS).

*Schedule:* Right after the delivery of the basic modules. This phase will be executed along with the Software Engineering part of the Software Development Methodologies module to enable better absorption of the concepts.

#### II – Design Phase:

*Tasks:* Software design and project plan.

*Deliverable:* Students will present the design and plan on the schema of the project.

*Schedule:* This phase will be executed during the final part of the Software Development Methodologies module.

#### III – Development Phase:

*Tasks:* Coding and testing of the software system/application.

*Deliverables:* Project report, functional software system/application.

*Schedule:* This final phase will be executed during the last 15 days of the course. A mid evaluation at the middle of the project development, and a final evaluation at the end of the project will be done.