



**PG-DAC - Aug 2024**  
**Syllabus & Marks Distribution**

Teaching Guidelines for  
**Logic Building & Problem Solving**  
(*Optional Preparatory Module*)  
PG-DAC August 2024

---

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

**Objective:** This preparatory module is conducted online before the actual PG-DAC course commencement 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 of online lecture sessions followed by solving various problems using the topics learned 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 Operating Systems & Software Development Methodologies**  
PG-DAC August 2024

---

**Duration:** 74 hours (50 theory hours + 24 lab hours)

**Evaluation:** 100 marks

**Weightage:** Theory exam – 40%, Lab exam – 30%, Internals – 30%

## Concepts of Operating Systems

**Duration:** 26 hours (18 theory hours + 8 lab hours)

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

**Prerequisites:** Knowledge of computer fundamentals

**Evaluation:** 35 marks (CCEE: 15 + Lab exam: 10 + Internals: 10)

**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: Introduction to OS

**Lecture:**

- 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: Introduction to Linux

**Lecture:**

- 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: (4 hours)**

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

## **Session 3: Shell Programming**

### **Lecture:**

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

### **Lab: (4 hours)**

- Shell Programs related to Session 3

## **Sessions 4 & 5: Processes**

### **Lecture:**

- What is process; preemptive and non-preemptive processes
- Difference between process and thread
- Process management; Process life cycle
- What are schedulers – Short term, Mediumterm 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

### **No Lab**

## **Sessions 6 & 7:**

### **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
- IO management

### **No Lab**

**Session 8:****Lecture:*****Virtual Memory***

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

**No Lab****Session 9:****Lecture:*****Deadlock***

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

**No Lab**

## Software Development Methodologies

**Duration: 48 hours (32 theory hours + 16 lab hours)****Objective:** To build knowledge of Software development methodologies.**Evaluation: 65 marks (CCEE: 25 + Lab exam: 20 + Internals: 20)****Reference Books:**

- Software Engineering by Chandramouli / Pearson
- Software engineering by Ian Sommerville / Pearson
- Object-Oriented Analysis and Design Using UML - An Introduction to Unified Process and Design Patterns by Mahesh P. Matha / PHI
- 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 (4 hours)

**Session 1****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 & DevOps (28 hours)**

**Sessions 2, 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

**No Lab**

**Sessions 6, 7 & 8**

**Lecture**

- Introduction to Agile development model
- Agile development components
- Benefits of Agile
- Introduction to different tools used for agile web development
- Scrum and Extreme Programming
- Introduction to AtlassianJira
  - Add Project
  - Add Tasks and sub-tasks

- Create sprints with tasks
- Case study of developing web application using agile methodology

**No Lab****Sessions 9 & 10****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 11****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 (4 hours)**

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

**Testing & Integration (16 hours)****Session 12****Lecture**

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

**Assignment**

- Read more testing concepts used in the industry

## Session 13

### 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)

### Assignment

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

## Sessions 14 & 15

### 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 16

### 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

Teaching Guidelines for  
**Object Oriented Programming with Java**  
PG-DAC August 2024

---

**Duration:** 116 hours (60 theory hours + 56 lab hours)

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

**Prerequisites:** Basic knowledge of computer programming

**Evaluation:** Total 100 marks

**Weightage:** CCEE – 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

---

(Note: Each Session is of 2 hours)

**Session 1: Introduction to Java**

**Lecture:**

- Introduction to java
- Features of java
- JVM Architecture
- JDK and its usage
- Structure of java class
- Working with data types: Primitive data types

**Sessions 2 & 3: Basic programming concepts**

**Lecture:**

- Java Tokens
- Declaring variables and methods
- Data type compatibility
- Operators
- Control statements
- Arrays 1-D and multidimensional array

**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++.

**Object Oriented Programming Concepts****Session 4: Object Oriented Programming Concepts****Lecture:**

- Introduction to OOP
- Classes and Objects
- OOP principles
- Encapsulation, Abstraction, Inheritance and Polymorphism

**Session 5:****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 6:****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.

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

- Inheritance: single & multilevel
- Inheritance: Hierarchical
- Association, Aggregation and Composition

- Polymorphism: Compile time and runtime polymorphism
- Rules of overriding and overloading of methods
- super and this keyword

**Lab 5 & 6:**

- Create a class Employee and encapsulate the data members.
- Create demo applications to illustrate different types of inheritance.

**Session 8:**

**Lecture:**

- Upcasting and downcasting of a reference variable
- Abstract class and abstract methods
- Interface (implementing multiple interfaces)

**Sessions 9 & 10:**

**Lecture:**

- Final variables, final methods and final class
- Functional interface
- New interface features (Java 8 & 11)
- Lambda Expression
- Inner Class (Regular, Method local, Anonymous & static inner class)
- Enum

**Lab 7, 8 & 9:**

- 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.
- Implementation of functional interface with anonymous class and lambda expression.

**Session 11:**

**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 12:**

**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 10 & 11:**

- 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 13 & 14:

#### Wrapper Classes and String Class

##### Lecture:

- Wrapper classes and constant pools
- String class, StringBuffer&StringBuilder class
- String pool

##### Lab 12 & 13:

- 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 15 & 16:

#### 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 14 & 15:

- Create user defined checked and unchecked exceptions.

### Sessions 17:

#### 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

### Session 18:

##### 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 methods

##### Lab 16 & 17:

- Create a Demo class to read & write image/text files.
- Create Serialization Demo class to illustrate serialization and de-serialization process.
- Create a demo class for Date, Time and Calendar

## Collections

### Sessions 19, 20, 21 & 22:

#### 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 18, 19, 20 & 21:

- 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 23, 24, 25 & 26:

#### Lecture:

- Set Collection:
  - HashSet, LinkedHashSet&TreeSet collection
  - Backed set collections.
- Map Collection:
  - HashTable, HashMap, LinkedHashMap&TreeMap classes
  - Backed Map collections.
- Concurrent collections
- Implementation of Java 8 stream API

### Lab 22, 23 & 24:

- 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 "? extendsSomeClass" , "? supersomeclass" and "?"
- Implementation of streams methods: Map, Reduce, Count, Sort, etc.

### Session 27:

#### Lecture:

- MultiThreading : Thread class and Runnable Interface
- sleep, join, yield, setPriority, getPriority methods.
- ThreadGroup class

**Lab 25:**

- Create multiple threads using Thread class and Runnable interfaces.
- Assign same task and different task to multiple threads.
- Understand sleep, join, yield methods.

**Sessions 28 & 29:**

**Lecture:**

- Synchronization
- Deadlock
- Wait, notify and notifyAll methods.
- Producer & Consumer problem

**Lab 26 & 27:**

- 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 30: Generics and Reflection API**

**Lecture:**

- Introduction to generics
- Generic classes
- Generic methods
- Wild cards (upper and lower)
- Metadata & Reflection

**Lab 28:**

- 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 August 2024

---

**Duration:** 72 hours (36 theory hours + 36 lab 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:** CCEE – 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 )
- OO 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 linkedlist and when to use array.

**Lecture:**

- Linked lists
  - Singlylinked lists
  - Doublylinked 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 & 9:**
**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

**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 10, 11 & 12:**

**Searching and 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 13:**

**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 14, 15 & 16:**

**Graphs and 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 17 & 18:**

**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
  - Asymptotic Analysis
  - Algorithm Analysis

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

**Lab+ Assignment:**

- Study on different Algorithms
- Compare different Algorithms previously programmed and do the analysis

## Teaching Guidelines for Database Technologies

PG-DAC August 2024

---

**Duration:** 72 hours (36 theory hours + 36 lab 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:** CCEE – 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

**Sessions 3 & 4:****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 (2 hours)**

DML (INSERT/UPDATE/DELETE), TRUNCATE

**Session 5:****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 6:****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 7:****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 8:****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 9:****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 10:****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 11:****Lecture**

Loop constructs (ITERATE, LEAVE)

Functions with and without parameters

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

**Lab (4 hours)**

Creating Function and returning value from it

Use of built-in functions in queries

**Session 12:****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 13:****Lecture**

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

Trigger Examples and real time use

**Lab**

Create Before Triggers

Create After Triggers

**Session 14:****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 15:****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

**No Lab****Session 16, 17, 18:****Lecture**

Introduction to MongoDB, Features of MongoDB

MongoDB command interface and MongoDB compass

MongoDB Documents & Collections

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

JSON and BSON documents

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

MongoDB – Operators, Sorting, Indexing

**Lab (8 hours)**

Using MongoDB Shell and Compass

Creating database, Connecting to a database, Creating Collections

Performing CRUD operations

MongoDB: Complex Read Using Operators, Sorting Operations, Creating Indexes

Teaching Guidelines for  
**Web Programming Technologies**  
PG-DAC August 2024

---

**Duration:** 112 hours (56 theory hours + 56 lab 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:** CCEE – 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 IriarteKoroliova / Packt

**References:**

- Internet & World Wide Web : How to Program by Paul Deitel, Henry Deitel&Abbey Deitel / Pearson Education
  - XML - How to Program by Deitelet 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 NodebyVasan 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 HTTP1.0, 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 theWeb
- 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: Responsive Web Design**

**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

**Lab:**

- Update the design of the Resume form using Bootstrap

**Session 5: 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 6 & 7: 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 8 & 9: 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 10: JSON & jQuery**

**Lecture:**

- JSON: JavaScript Object Notation (JSON)
  - Introduction and need of JSON
  - JSON Syntax Rules
  - JSON Objects, JSON Arrays, JSON Files

- JSON parsing
- jQuery: Introduction
  - jQuery selectors
  - jQuery events
  - jQuery animation effects
  - jQuery DOM traversal and manipulation
  - Data attributes and templates
  - jQuery DOM utility functions
  - jQuery plugins

**Lab:**

- Create a JSON object, array, and file to store a cricket match (or any team sport) scoreboard.
- 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 11 & 12: AJAX & Axios HTTP Client

**Lecture:**

- AJAX: Asynchronous JavaScript and XML
  - Introduction to AJAX
  - AJAX framework and its architecture
  - Web services and AJAX
  - AJAX using jQuery and jQuery
- Axios: A promise-based HTTP client
  - The Axios instance and its config
  - Handling request and response
  - Handling errors

**Lab:**

- Design and implement a webpage that displays a live scoreboard. Use AJAX (XMLHttpRequest) to retrieve and interpret JSON data from a URL provided by the faculty.
- Design and implement a webpage that displays live news headlines. Use the Axios HTTP client to retrieve and interpret JSON data from a URL provided by the faculty.

### Session 13: 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 14 & 15: 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 16: 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 17: 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 18: 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 19: 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 20, 21 & 22: 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 23 & 24: 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 25: 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 26, 27 & 28: 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.

Teaching Guidelines for  
**Web-based Java Programming**  
PG-DAC August 2024

---

**Duration:** 112 hours (58 theory hours + 54 lab 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:** CCEE – 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)

**Sessions 1 & 2:**

**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

**Session 3:**

**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 4, 5, 6 & 7:**

**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 8 & 9:**

**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 10, 11, 12 & 13:****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 14, 15, 16 & 17:****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 18, 19 & 20:

#### 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)

#### Lab:

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

### Sessions 21 & 22:

#### Lecture:

##### Spring Data Module

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

#### Lab:

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

### Session 23:

#### 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 24 & 25:

#### 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 26 & 27:

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
- JWT Authorization

Lab:

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

### Sessions 28 & 29:

Lecture:

#### Microservices

- Introduction to Microservices
- Microservices Architecture
- Fragmentation of business requirement
- Deployment pattern
- API gateway
- Service Discovery
- Database Management for Microservices

Lab (2 hours):

Demonstration of Spring Boot Microservices

**Teaching Guidelines for  
MS.Net Technologies**  
PG-DAC August 2024

---

**Duration:** 90 hours (50 theory hours + 40 lab hours)

**Objective:** To acquire the knowledge of Microsoft.NET 6.

**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 .Net6 using Visual Studio 2022*

**Evaluation:** 100 marks

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

**Text Book:**

- Pro C# 10 with .Net 6- Foundational Principles and Practices in Programming by Andrew Troelsen & Philip Japikse / Apress

**References:**

- C# 10 and .Net 6- 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.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

**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, ParameterizedThreadStart
- 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 13-19:**

**Lecture:**

**Introduction to Asp.Net MVC CORE**

- 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` & `AllowAnonymous` attributes
- Implementing Forms Based Authentication
- Preventing Forgery Attack using `AntiForgeryToken`
- Preventing Cross Site Scripting Attack

### **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

#### Introduction to Razor Pages

##### Lab:

Lab exercise covering the concepts covered in the class

#### Session 20:

##### Lecture:

Localization in MVC (Demo Only)

Deploying ASP.NET MVC application (Demo only)

##### No Lab

#### Sessions 21, 22 & 23

##### Lecture:

##### Web APIs

- Creating ASP.NET MVC Web API
- Configuring for CORS
- Different Verbs
- Consuming using a client
- Using Newtonsoft APIs
- Integrating Web API with React App
- Configuring CORS of React App and Web API
- Sending request from React App, processing at Web API and effecting the database

##### Lab (4 hours):

Create a RESTful service using Web API.

Create a consumer.

#### Sessions 24 & 25

##### Lecture:

##### MVC integration with React

- Introduction to MVC and React
- Setting up the Project
- Integrating React with MVC Backend
  - Define Models
  - Implement Controllers
  - Use Views
- Data Management and State Handling
  - Establish data flow
  - Manage state
- Advanced Topics and Best Practices
  - Authentication and Authorization
  - Routing
  - Structuring React components

##### Lab:

Create a RESTful service using WEB API with React as the front end.

Create a consumer.

## Teaching Guidelines for **C++ Programming**

PG-DAC August 2024

---

**Duration:** 72 hours (36 theory hours + 36 lab hours)

**Objective:** To learn object oriented programming using C++

**Prerequisites:** Knowledge of computer fundamentals

**Evaluation:** 100 marks

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

**Text Book:**

- C++ Primer Plus by Stephen Prata /Pearson

**References:**

- Thinking in C++ by Bruce Eckel
  - The C++ Programming Language, Bjarne Stroustrup
- 

(Note: Each Session is of 2 hours)

### Session 1: Getting Started

**Lecture:**

- Installation and Setup development environment
- The need of C++
- Features of C++
- C++ versus C
- History of C++
- Writing your first C++ program

**Lab:**

Write different C++ 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

### Sessions 2 & 3: Beginning with C++

**Lecture:**

- C++ Program structure
- Introduction of advanced C++ concepts and feature of C++ 17
- C++ Tokens
- Initialization
- Static Members
- Constant Members
- Expressions

**Operators**

- Arithmetic Operator

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

**Lab:**

- Write a Student class and use it in your program. Store the data of 10 students and display the sorted data according to their roll numbers, dates of birth, and total marks.
- Implement all C++ operators
- Declare members and implement in your programs.

#### **Session 4: Conditional and Looping Statements**

**Lecture:**

- If, else if, switch
- for loop
- while loop
- do while loop
- Jump statement (break, continue & return keyword)
- Arrays
- Declaration and initialization of an array
- 1-D and 2-D arrays

**Lab:**

- Implement all control structures through your program
- Implement a program which accepts command line arguments from main function.

#### **Session 5: Functions in C++**

**Lecture**

- Different forms of functions
- Function prototyping
- Call by Reference
- Inline Functions
- Math library functions etc.

**Lab:**

- Implement functions through your program
- Declare function and call it by reference and note the observations
- Implement Inline functions in your program

#### **Sessions 6 & 7: Memory Management and Pointers**

**Lecture**

- Introduction to memory management in C++
- Pointers in C++
- Arrays using pointers
- Enumeration
- Typedef
- Using New operator
- Class pointer
- this pointer
- Comparison of new over malloc, calloc and realloc, etc.
- Memory freeing using Delete operator

**Lab:**

- Assignments using pointers, arrays of pointers

- Assignments on passing pointers in functions
- Using pointers, write your own functions for the following:
  - String comparison
  - String concatenate
  - String copy
  - String length

*Note:* Do not include <string.h> in your program and implement Delete operator in your program.

### Session 8: OOP Concepts

#### Lecture

- Discussion on object oriented concepts
- Classes and Objects, Access Specifiers, Overloading, Inheritance, Polymorphism
- Namespaces

#### Lab:

- Write a student class and use it in your program. Store the data of 10 students and display the sorted data according to their roll numbers, dates of birth, and total marks.

### Session 9: Constructors and Destructor

#### Lecture

- Constructors
- Parameterized constructors
- Multiple constructors in class
- Dynamic initialization of objects
- Copy Constructors
- Destructor

#### Lab:

- Implement constructor and destructors through your program
- Write a program to implement inner class in C++

### Session 10: Inheritance – extending class

#### Lecture

- Types of inheritance
- Single inheritance
- Multiple inheritance
- Multilevel inheritance
- Hierarchical inheritance
- Hybrid inheritance, etc.
- Virtual base class
- Constructors in derived class

#### Lab:

- Design a hierarchy of computer printers. Use multiple inheritance in your hierarchy. Also use friend functions and classes in your program.

### Session 11: Polymorphism

#### Lecture

- Types of Polymorphism
- Overloading functions
- Overloading Operators
- Friend functions
- Constant functions

**Lab:**

- Write Date and Time classes that allow you to add, subtract, read and print simple dates in dd/mm/yyyy and time in hh:mm:ss formats. Use function overloading in your program.
- Assignments to overload =, ==, +, ++, --, <<, >> and [ ]operators.

**Session 12: Virtual Functions and Abstract Class****Lecture**

- Run Time Polymorphism
- Virtual Functions and Pure virtual functions
- dynamic\_cast, static\_cast, const\_cast, reinterpret\_cast
- Interfaces
- Abstract class

**Lab:**

- Implement Abstract classes in your program
- Using virtual and pure virtual functions implement hierarchy of computer printers
- Implement diamond problem with real life example

**Session 13: Exception Handling****Lecture**

- Exception Handling Introduction
- Exception handling – throwing, catching, re-throwing an exception
- Specifying exceptions etc.

**Lab:**

- Implement exceptions and do proper management through your program
- Implement Custom exception class

**Session 14: Managing Console I/O Operations****Lecture**

- Introduction
- C++ streams
- C++ stream classes
- Unformatted I/O operations
- Formatted I/O operations
- Managing output with manipulators

**Lab:**

- Implement console I/O operations through your program.

**Session 15: File Handling in C++****Lecture**

- Definition of file
- File handling in C++
- Doing read, write operation in files

**Lab:**

- Assignments on files doing different operations

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

- Introduction to Templates
- Function Templates
- Class Templates

**Lab:**

- Assignments on templates

**Sessions 17 & 18: STL and RTTI**

**Lecture**

- Introduction to C++ Standard Library
- Working with Stack, Vector, Queue, Map
- Introduction to RTTI (Run-Time Type Information) in C++

**Lab:**

- Assignments on STL Library

Teaching Guidelines for  
**General Aptitude & Communication**  
PG-DAC August 2024

---

**Duration:** 90 hours (Classroom hours + Practice sessions)

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

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

**Weightage:** Aptitude - 40% (through CCEE) ; Communication - 60% (through internals)

---

## General Aptitude

**Duration:** 40 Class room hours + Practice sessions

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

**Evaluation:** Centralised Course-End Exam (CCEE) (40 marks)

**Reference Books:**

- Quantitative Aptitude by RS Aggarwal / S Chand
- Verbal & Non-Verbal Reasoning: RS 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)

---

**Session 1: (2.5 hours)**

- Percentage
- Profit & Loss

**Session 2: (2.5 hours)**

- Ratio & Proportion
- Partnership

**Session 3: (2.5 hours)**

- Average
- Mixture & Alligation

**Session 4: (2.5 hours)**

- Simple Interest & Compound Interest

**Session 5: (2 hours)**

- Cyclicity & Remainders

**Session 6: (1.5 hours)**

- Coding & Decoding

**Session 7: (1.5 hours)**

- Syllogism

**Session 8: (1.5 hours)**

- Blood Relations

**Session 9: (2 hours)**

- Seating Arrangements
  - Linear & Circular

**Session 10: (2 hours)**

- Puzzles
- Ordering and Ranking

**Sessions 11 & 12: (3 hours)**

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

**Sessions 13 & 14: (3 hours)**

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

**Session 15: (1.5 hours)**

- Directions

**Session 16: (2 hours)**

- Surds and Indices
- Problems on Ages

**Session 17: (2 hours)**

- Data Interpretation
- Data Sufficiency

**Session 18: (3 hours)**

- Permutations & Combinations

**Session 19: (2 hours)**

- Probability

**Session 20: (3 hours)**

- Revision
- Getting ready for placement tests
- Practice tests

## Effective Communication

**Duration:** 50 Classroom 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

- First impressions
- Greeting
- Formal dressing & etiquettes
- Body language
- Ethics

**Session 4:**

Personality Development

- Developing positive attitude
- Confidence building

- Questioning techniques
- Psychometric Analysis

**Practice Sessions:**

*Practise greeting, etiquettes and questioning*

**Sessions 5 & 6:**

English Grammar

- Nouns
- Pronouns
- Verbs
- Adjectives
- Adverbs
- Prepositions
- Conjunctions
- Articles

**Practice Sessions:**

*Practise sentence making*

**Session 7:**

English Grammar

- Present Tense
- Past Tense
- Future Tense

**Practice Sessions:**

*Practise sentence making*

**Session 8:**

English Grammar

- Active and passive voices
- Direct and indirect speeches

**Session 9:**

English Grammar

- Idioms
- Synonyms & Antonyms

**Practice Sessions:**

*Practise speaking in active & passive voices*

*Practise direct & indirect speaking*

*Practise idioms, synonyms & antonyms*

**Session 10:**

Correct Usage of English

### Session 11:

Common Mistakes in English Communication

#### **Practice Sessions:**

*Practise correct English communication*

### Session 12:

Listening Skills

- Importance of listening
- Techniques for effective listening

### Session 13:

Listening Skills

- Voice & Accent (VNA) Rounds in interviews
  - Listening to audio/video clips
  - Question-answers based on the listened audio/video clips

#### **Practice Sessions:**

*Practise audio synthesis*

### Session 14:

Reading Skills

- Reading Comprehension
  - Practise proper accent and articulation
  - Techniques to answer questions based on comprehension

#### **Practice Sessions:**

*Comprehension exercises*

### Session 15:

Writing Skills

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

### Session 16:

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

#### **Practice Sessions:**

*Essay writing*

*Letter writing  
email writing  
Personalized Generative AI based writing*

### **Session 17:**

Public Speaking

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

#### ***Practice Sessions:***

*Conduct various types of speeches*

*Preparing personalized Generative AI based 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:**

Personal Interviews

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

### Session 21:

#### Personal Interviews

- Introducing yourself
  - Importance of introduction
  - Structure of introduction

### Session 22:

#### Personal Interviews

- Elevator Pitch for effective introduction
  - Importance of Elevator Pitch
  - Structure of Elevator Pitch

#### ***Practice Sessions:***

*Practise introduction*

*Analysis and feedback on introduction*

*Practise elevator pitch*

*Analysis and feedback on elevator pitch*

### 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 of answering questions
  - Common mistakes of answering questions

### Session 25:

#### Online Interviews

- Tips for online Interviews
  - Best practices for attending online interviews
  - What to do (and not do) during online interviews?

#### ***Multiple Practice Sessions:***

*Practise common technical questions*

*Practise common HR/behavioral questions*

*Conduct mock interviews*

*Conduct online interviews*

## Teaching Guidelines for Software Project

PG-DAC August 2024

---

**Duration:** 90 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 three 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:* Initiated 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.

## 6. PG-DAC Marks Distribution

The figures shown below indicate the weightage of each subject in the final performance statement for PG-DAC.

Module Code	ModuleName	Hours			Marks			
		Theory	Lab	Total	CCEE	Lab	IA	Total
DAC01	C++ Programming	36	36	72	40	40	20	100
DAC02	Concepts of Operating Systems & Software Development Methodologies	50	24	74	40	30	30	100
DAC03	Object Oriented Programming with Java	60	56	116	40	40	20	100
DAC04	Algorithms & Data Structures Using Java	36	36	72	40	40	20	100
DAC05	Database Technologies	36	36	72	40	40	20	100
DAC06	Web Programming Technologies	56	56	112	40	40	20	100
DAC07	Web-based Java Programming	58	54	112	40	40	20	100
DAC08	MS.Net Technologies	50	40	90	40	40	20	100
DAC08	Software Project	-	90	90	-	-	(100)	Grade
GAEC	General Aptitude & Effective Comm. (Aptitude 32 hrs + Comm. 48 hrs)	90		90			(100)	Grade
	<b>Total</b>	<b>472</b>	<b>428</b>	<b>900</b>	<b>320</b>	<b>310</b>	<b>170</b>	<b>800</b>