**C programming Syllabus**

**Coverage**

|  |
| --- |
|   Introduction to Programming |
|   Fundamentals in C |
|   Operators and Expressions |
|   Data types |
|   Input-Output Library Functions |
|   Control statements |
|    Function |
|    Storage class |
|    Pointer |
|    Pointer and Function |
|    Array |
|    Pointer and array |
|    Array and function |
|    Dynamic memory allocation |
|    String |
|    String and function |
|    Command line arguments |
|    Preprocessor |
|    Structure |
|    Structure and function |
|    File Handling |
|    ODBC Programming |
|    Process and threads |
|    Graphics |
|    Socket and Network programming |
|    Project  **Syllabus in Details** |

**Introduction to Programming**

* Program and Programming
* Programming Languages
* Types of software’s
* Operating Systems
* Dos commands
* Basic Linux commands and vi editor
* Compiler, Interpreter, Loader and Linker

**Fundamentals in C**

* + History of ‘C’
  + A Simple C Program
  + Program execution phases
  + Backslash character constants
  + Character set
  + Constants
  + Number systems
  + Format specifiers
  + Identifiers
  + Keywords
  + Variables
  + Data Types
  + Declaration of Variable
  + Assigning Values to Variables
  + Initialization
  + Comments
  + const Qualifier
  + Basic Structure of a ‘C’ program
  + Programming Examples

**Operators and Expressions**

* Dealing with all 45 operators
* Arithmetic operators
* Increment and decrement operators
* Relational operators
* Logical operators
* The bitwise operators
* The assignment operators
* The conditional operator
* The size of operator
* The comma operator
* Type casting operator
* Other operators
* Precedence and order of evaluation
* Programming Examples
* FAQ’s

**Data types**

* Modifiers
* format specifiers
* Dealing with each data types
* Memory representation of each type
* Programming Examples

**Input-Output Library Functions**

* Unformatted I-O Functions
* Single Character Input-Output
* String Input-Output
* Formatted I-O Functions
* printf() Width Specifier
* scanf() Width Specifier
* Programming Examples

**Control statements**

* Conditional Control Statements
* if
* if-else
* nested if-else
* else-if ladder
* Multiple Branching Control Statement
* switch-case
* Loop Control Statements
* while
* do-while
* for
* Nested Loops
* Jump Control statements
* break
* continue
* goto
* exit
* return
* Programming Examples
* FAQ’s

**Function**

* What is function ?
* Why function ?
* Advantages of using functions
* Function Prototype
* Defining a function
* Calling a function
* Return statement
* Types of functions
* Recursion
* Nested functions
* main() function
* Library Function
* Local and global variables
* Programming Examples
* FAQ’s

**Storage class**

* Types of storage class
* Scoping rules
* Dealing with all storage classes
* Programming Examples
* FAQ’s

**Pointer**

* Def of Pointer
* Declaration of Pointer Variables
* Assigning Address to Pointer Variables
* Dereferencing Pointer Variables
* Pointer to Pointer
* Pointer Arithmetic
* Pointer comparisons
* dereference and increment pointer
* pointer to const data
* const pointer
* const pointer to const data
* Void pointer or Generic Pointer
* Null pointer
* wild pointer
* Programming Examples
* FAQ’s

**Pointer and Function**

* Parameter Passing Techniques – call by value, call by address
* Using Pointers as Arguments
* Function Returning value
* Returning More than one value From A Function
* Functions Returning Address
* Function Returning Pointers
* Dangling pointer
* Pointer to a Function
* Calling A function through function pointer
* passing A function’s address as an Argument to other function
* Functions with variable number of arguments
* Programming Examples
* FAQ’s

**Array**

* One dimensional arrays
* Declaration of 1D arrays
* Initialization of 1D arrays
* Accessing element of 1D arrays
* Reading and displaying elements
* Two dimensional arrays
* Declaration of 2D arrays
* Initialization of 2D arrays
* Accessing element of 2D arrays
* Reading and displaying elements
* Programming Examples
* FAQ’s

**Pointer and Array**

* Pointer and one dimensional arrays
* Subscripting pointer variables
* Pointer to an array
* Array of pointers
* Pointers and two dimensional arrays
* Subscripting pointer To an array
* Programming Examples
* FAQ’s

**Array and Function**

* 1D array and function
* Passing individual array elements to a function
* passing individual array elements address to a function
* passing whole 1d array to a function
* 2D array and function
* Passing individual array elements to a function
* passing individual array elements address to a function
* passing whole 2d array to a function
* using arrays of function pointer
* Programming Examples
* FAQ’s

**Dynamic memory allocation**

* malloc()
* calloc()
* realloc() .
* free()
* Core dump
* Memory leak
* Dynamic 1D and 2D Arrays
* Programming Examples
* FAQ’s

**Strings**

* strings versus character arrays
* Initializing strings
* Reading string
* Displaying string
* The %s format specifier
* The gets() and puts() functions
* string handling functions
* string pointers
* Two-dimensional character arrays or array of string
* array of pointers to strings
* Programming Examples
* FAQ’s

**Strings and Function**

* passing strings individual character to a function
* passing strings individual character address to a function
* passing strings to a function
* Programming Examples
* FAQ’s

**Command line arguments**

* what is command prompt?
* why command line?
* What are command line arguments?
* Programs using command line

**Preprocessor**

* What is preprocessing?
* Macro expansions
* File inclusions
* Conditional compilation
* The stringification(# )and token passing operator( ##) operators
* Programming Examples
* FAQ’s

# Structure

* Why is structure used?
* What is structure?
* Advantages of structures
* Defining a Structure
* Declaration of Structure Variables
* Initialization of Structure Variables
* Accessing Structure Members
* Storage of Structures in Memory
* Size of Structures
* Reading and Displaying Structure Variables
* Assignment of Structure Variables
* Pointers to structures
* Array of structures
* Arrays within structures
* Nested structures
* Self-referential structures
* memory link(linked list)
* Bit fields
* Programming Examples
* FAQ’s

**Structure and Function**

* Passing structure member to a function
* Passing structure variable to a function
* Passing structure variable address to a function
* Passing array of structure to a function
* Returning a structure variable from function
* Returning a structure variable address from function
* Returning structure variable from a function
* Programming Examples
* FAQ’s

**Union and Enumeration and typedef**

* What are unions?
* Structures versus unions
* Working with unions
* Initializing unions
* Advantages of unions
* enum keyword
* typedef keyword
* Programming Examples
* FAQ’s

**File Handling**

* Using files in C
* Buffer and streams
* Working with text files and Binary Files
* File operations using std. library and system calls
* File management I/O functions
* Random Access Files
* Programming Examples
* FAQ’s

**ODBC Programming**

* ODBC rules and regulation
* Introduction to MYSQL and Oracle
* Creating, inserting and retrieving records for different Data bases.
* Programming Examples
* FAQ’s

**Process and Threads**

* What is process & Threads
* Use of fork, vfork
* Daemon process
* Programming Examples
* FAQ’s

**Graphics & Curses**

* Graphics using Glade interface with GTK+
* Working with GTK Widgets, Event handling
* Developing Application Interface
* Programming Examples
* FAQ’s

**Socket and Network programming**

**PROJECT**

**C ++ Syllabus**

**Coverage**

|  |
| --- |
|   Introduction to Programming |
|   Fundamentals in C++ |
|   Control statements |
|   Pointer array Reference |
|   Function |
|   Introduction to oops |
|   Classes and Objects |
|   Constructors and Destructors |
|   Operator Overloading |
|   Inheritance and Composition |
|   Polymorphism |
|   Exception handling |
|   Input / Output in C++: Streams |
|   File handling |
|   Working with String |
|   Command line arguments |
|   Namespace |
|   Templates |
|   Data Structures(introduction) |
|   STL |
|   Process and Threads |
|   Graphics |
|   WEB development |
|   Project |

**Syllabus in Details**

**Introduction to Programming**

* Program and Programming
* Programming Languages
* Types of software’s
* Operating Systems
* Dos commands
* Basic Linux commands and vi editor
* Compiler, Interpreter, Loader and Linker

**Fundamentals in C++**

* + History of ‘C++’
  + Migrating from procedural oriented language to object oriented languages Program
  + Keywords
  + Variables
  + Constants
  + Data type
  + Operators
  + Manipulators and uses
  + Basic Structure of a ‘C++’ program

**Control statements**

* Conditional Control Statements
* if
* if-else
* nested if-else
* else-if ladder
* Multiple Branching Control Statement
* switch-case
* Loop Control Statements
* while
* do-while
* for
* Nested Loops
* Jump Control statements
* break
* continue
* goto
* exit
* return
* Programming Examples
* FAQ’s

**Pointer array Reference**

* + pointer variable
  + Reference variable/alias variables?
  + Reference to Reference variable?
  + Reference to array?
  + Reference vs normal variable?
  + Reference vs pointer variable?
  + 1D and 2D Arrays
  + What is dynamic memory allocation?
  + The new and delete operator
  + new vs malloc
  + delete vs free
  + Dynamic 1D and 2D Arrays

**Function**

* What is function ?
* Why function ?
* Advantages of using functions
* Function Prototype
* Defining a function
* Calling a function
* Actual and Formal Arguments
* Types of functions
* Parameter Passing Techniques
* Call by Value
* Call by Reference
* Call by Pointer
* Return statement
* Returning More than one value From A Function
* Return by value mechanism
* Return by pointer mechanism
* Return by reference mechanism
* Inline Functions
* Default Arguments
* Function Overloading
* Lambda function.
* Recursion

**Introduction to oops**

* c structure vs c++ structure
* c++ class vs c++ structure
* Class
* Object
* Encapsulation
* Abstraction
* Polymorphism
* Inheritance
* Message Passing

**Classes and Objects**

* Declaring / defining classes
* Data members and member functions
* Access specifiers : public and private and protected
* Creating objects of a class
* Pointers to object
* Implicit this pointer
* Static data members
* Static member functions
* Passing objects to a member function
* Returning objects from a member function
* Friend functions
* Friend classes
* Nested classes
* Local classes
* The const member functions
* The const objects
* Array of objects
* static objects
* What are inline functions?

**Constructors and Destructors**

* Defining Constructor
* Defining Destructor
* Comparing Constructor Member Function
* Default Constructor
* Argument Constructor
* Copy Constructor
* Constructor Overloading
* Default Argument in Constructor
* Anonymous object
* Private Constructor and Destructor
* Local vs Global object

**Operator Overloading**

* Need of Overloading
* Defining Operator Overloaded Function
* Operator Overloading Rules
* Overloading Unary Operators
* Overloading Unary Operators using Friend
* Overloading Binary Operators
* Overloading Binary Operators using Friend
* Overloading Other Operators

**Inheritance and Composition**

* What is inheritance?
* The is-a relationship
* Single Level Inheritance
* Multilevel Inheritance
* Multiple Inheritance
* Name ambiguities under multiple inheritance
* Hierarchical Inheritance
* Hybrid Inheritance
* Multipath Inheritance
* Why virtual base classes?
* Constructor and Destructor in Inheritance.
* Common constructor.
* Delegation
* What is composition?
* The has-a relationship

**Polymorphism**

* About polymorphism
* Compile time and runtime polymorphism
* Virtual functions.
* Pure virtual function and abstract base class.
* What is RTTI (Run-time Type Information)?
* VPTR and VTABLE.
* Difference between member Function Overloading and Overriding
* Object slicing.
* Constructor and virtual function.
* Virtual destructor.
* destructor with virtual function.

**Exception handling**

* What is an exception?
* Throwing an exception
* Catching an exception
* Trying for an exception
* Order of catch blocks
* Catching all exceptions
* Nested try blocks
* Rethrowing an exception
* Exception specifications
* What is stack unwinding?
* Exceptions in ctors and dtors
* The unexpected() function
* The terminate() function
* The standard exceptions
* Creating our own exception classes

**Input / Output in C++: Streams**

* Hierarchy of I/O Streams
* Fundamental stream classes and objects
* Standard input and output functions
* Formatting flags and manipulators

**File handling**

* Hierarchy of File Streams
* Using constructor method
* Using open and close member function method.
* Object as file stream reader and writer
* Both sequential and random file accessing mechanism.
* Different error handling mechanism in files

**WORKING WITH STRING**

* Different C string handling library
* string handling using relational operator
* Different string handling function

**Command line arguments**

* what is command prompt?
* why command line?
* What are command line arguments?
* Programs using command line

**NAMESPACE**

* Creating name space
* Using name space
* Nested namespace and anonymous namespace

**Templates**

* What is generic programming?
* Need of Template
* What are function templates?
* Argument deduction and template parameters
* Overloading function templates
* What are class templates?
* Specializations of class templates

**Data Structures**

Introduction

Single Linked List

Circular Linked List

Doubly Linked List

Stacks

Queues

**STL**

* STL Components
* Containers
* Iterators
* Algorithms
* Common container operations
* Vectors
* Deques
* Lists
* Sets and multisets
* Maps and multimaps
* Implementing reference semantics
* When to use which container?
* Special STL Containers
* Stacks
* Queues
* Priority Queues
* Bitsets
* STL Iterators
* Input iterators
* Output iterators
* Forward iterators
* Bidirectional iterators
* Random access iterators

**Database operation**

* What is database?
* SQL for relational database.
* About API connect to database.
* Database connectivity MySQL.
* Database manipulation using c++

**Process and Threads**

**Graphics**

**WEB development**

Web basics.(HTML, java script, CSS).

**PROJECT**

**Data Structure Syllabus**

**Coverage**

|  |
| --- |
|   Introduction |
|   Array |
|   Pointer |
|   Function |
|   Structure |
|   Pointer, Structure with Function |
|   Stack |
|   Applications of Stack |
|   Linear Queue & its Operations |
|   Circular Queue & its Operation |
|   Linked List (Single ,Double ,Circular, Header) |
|   Tree |
|   Graph |
|   Hashing & Searching |
|   Sorting |

**Syllabus in Details**

**Introduction**: Concept Data Structure, Example, Need of Data structure , Advantages of using DS,

**Algorithm & Pseudocode**: Algorithm Definition, Characteristics of algorithm, Elements of algorithm, Pseudocode example, difference of Algorithm & Pseudocode.

**Function:** What is function, Types of function, How function works, function recursion and how it works.

**Array:** Concept of Array, Types of array, Basic Programs, Array with Functions, Single & 2-dimensional array in function argument.

**Pointer:** Pointer Basics, Pointer with functions, call by reference, array of pointers & pointer to array & Programs.

**Structure**: Understanding about Structure, Pointer structure variable, Structure as function argument using call by member value, whole structure and call by passing reference of structure.

**Stack:** Operations on Stack, Array & Linked Representation, Programs on stack, Push & Pop operations, traversing.

**Applications of Stack :**Arithmetic Expression Evaluation, Notations, Infix, Postfix, Prefix, Conversion infix to post fix, Conversion postfix to infix, Evaluation of Postfix and Pre fix using stack.

**Queue:** Operations on Queue, Array & Linked Representation, Programs on stack, Insert & Delete operations, Circular queue, representation, Deque, Priority Queue, Application of queue.

**LinkedList:** Concept of Linked List, Difference of LinkList & Array**,** Single Linked List, Representation, Operations, Traversing, Insertion(first node, last node, at a position, after a node value), Deletion(first node, last node, at a position, after a node value)

Double Linked List, Representation, Operations, Traversing, Insertion (first node, last node, at a position, after a node value), Deletion (first node, last node, at a position, after a node value),Circular Link List & header Link List example

**Tree:** Tree terminology, Binary tree, Complete Binary Tree, Binary search tree, Tree Traversals. Creation of Binary Tree from traversal methods, Expression Tree & expression Manipulation, Binary Search Tree, Insertion & deletion in BST(Program), AVL Tree, M-way Search Tree ,B+ tree, Insertion & deletion.

**Graph:** Graph terminology, Representation of graphs, path matrix, Graph Traversal, BFS (breadth first search), DFS (depth first search), Minimum spanning Tree, Kruskal’s Algorithm & Prim’s Algorithm. Warshall’s algorithm(shortest path algorithm).

**Hashing & Searching**: Linear and binary search methods, Hash functions, Hashing techniques & Chaining.

**Sorting**: Bubble sort, selection sort, Insertion sort, Quick sort, merge sort, Heap sort, Radix sort

**Core Java Syllabus**

**Coverage**

|  |
| --- |
|  Introduction. |
|  OOPS |
|  Package |
|  Exception Handling. |
|  Multithreading |
|  Applet, AWT, Event Handling |
|  Using NetBean, Ecllipse. |
|  Input Output Streams, Serialization |
|  Networking |
|  Collection Framework, classes & interfaces of java.util, generics |
|  Introduction to Swing (Java Foundation Classes). |
|  Remote Method Invocation, Implementation of RMI. |
|  JDBC (Java Data Base Connection), Types of Driver |
|  Project |

* It provides knowledge for foundation of Java programming.
* This says “how to start writing programs in Java and how to become a Java professional.
* A trainee of this module should complete classroom task as well as home task.
* After the completion of the course one can appear and clear Oracle’s OCJP exam.

**Syllabus in Details**

**1: > First Meeting with Java**

In this section we will know Java through an introduction session. We will discuss about following topics in related to Java:

**1.1 History**

* Computers... How a Program uses Computers?
* Java... Why? What? How(Green Project)? When? Where?
* Different Java Versions.
* How Java is different from other Technologies

**1.2 Introduction to Java Programming Environment**

* How to Install & set Path.
* A Simple Java Program
* Compiling & executing Java Program
* Phases of Java Program
* Analysis of a Java Program
* Understanding Syntax and Semantic Error, Runtime Exception
* Name of a Java Source File
* Platform Independency
* Java Technology( JDK, JRE, JVM, JIT)
* Features of Java
* Text Editors
* Consoles
* Problem Solving.

**1.3 Fundamentals of Java Programming**

* Naming convention of Java language
* Comments
* Statements
* Blocks (Static, Non-static/instance)
* Identifiers
* Keywords
* Literals
* Primitive Data Types, Range
* Reference(User defined) Data type
* Variables (Primitive, Reference)
* Type Casting, Default Value
* Operators
* Problem Solving

**1.4. Control Structures**

* Working with Control Structures
* Types of Control Structures
* Decision Control Structure (if, if-else, if else if, switch –case)
* Repetition Control Structure (do –while, while, for)
* Problem Solving

**1.5 Java Array**

* What is Array
* Array Declaration in java vs C and C++.
* Instantiation of an Array
* String vs character array.
* Accessing Array Elements, Default Value, for-each loop, varargs.
* Length of an Array (What is ArrayIndexOutOfBoundsException).
* Increasing, Decreasing the Size and Copy of an Array
* Multi-Dimensional Arrays
* Problem Solving

**1.6 Keyboard Input in Java**

* Java program inputs from Keyboard
* Methods of Keyboard inputs
* Scanner, Buffered Reader
* JOption Pane
* Problem Solving

**1.7 Command-Line Arguments**

* What is a Command-Line Argument?
* Java Application with Command-Line Arguments
* Conversion of Command-Line Arguments
* Passing Command-Line Arguments
* Using methods (Static , Non Static)
* Problem Solving

**1.7 Integrated Development Environment**

* Using various Editors.
* Program Compilation, Execution in Editor
* Using Ecllipse IDE
* Project Set Up
* Source File Generation
* Application Compilation and Run
* Difference between C and C++ with Java,
* Problem Solving
* Interview related Question and Answer.

**2: > Object Oriented Programming**

In this section, we will learn concept of OOPS, because it is the foundation of modern days programming language, we will and write Java Programs by using class and its functionality.

**2.1 Procedural Vs Object Oriented Program**

* Different type of Program Procedural Vs Object Oriented.
* Top Down Vs Bottom Up Approach.
* Introduction to Object Oriented Programming
* Abstraction, Encapsulation, Inheritance, Polymorphism.
* Introduction to Classes and Objects
* Custom Class Definition
* Instance and Static Variables
* Different ways to create Object Instance 5
* Types of Class 3-5
* Instance Variable and it’s role in a Class
* Constructors, types of Constructor, Constructor Rule, Constructor Overloading.
* Static Variable and it’s use.
* Methods and their behavior.
* Constructor vs Methods
* Constructors
* “this” Keyword.
* Java Access Modifiers ( and Specifiers)
* Explanation of psvm() , sopl()
* Problem Solving

**2.2 Inner Class**

* First View of Inner Class
* Outer Class Access
* Types of Inner Class
* Problem Solving

**2.3 Inheritance**

* A Little knowledge on Inheritance
* Sub-Classes
* Object Classes
* Constructor Calling Chain
* The use of "super" Keyword
* The use of “private” keword inheritance.
* Reference Casting
* Problem Solving

**2.4 Abstract Classes and Inheritance**

* Introduction to Abstract Methods, Abstract Classes and Interface
* Interface as a Type
* Interface v/s Abstract Class
* Interface Definition
* Interface Implementation
* Multiple Interfaces' Implementation
* Interfaces' Inheritance
* How to create object of Interface
* Problem Solving

**2.5 Polymorphism**

* Introduction to Polymorphism
* Types of Polymorphism
* Overloading Methods
* Overriding Methods
* Hiding Methods
* Final Class and Method
* Polymorphic Behaviour in Java
* Benefits of Polymorphism
* “Is-A” vs “Has-A”
* Association Vs Aggregation
* Problem Solving
* Interview related Question and Answer.

# 3: > Package

In this section, we will learn how to bundle up the .class file to deliver the software and how to use other package in a Java Program.

**3.1 Package and Classpath and its use**

* First look into Packages
* Benefits of Packages
* Package Creation and Use
* First look into Classpath
* Classpath Setting
* Class Import
* Package Import
* Role of public, protected, default and private w.r.t package
* Namespace Management
* Package vs. Header File
* Creating and Using the Sub Package
* Sources and Class Files Management

**3.2 Using Predefined Lang package & other Classes**

* Java.lang Hierarchy
* Object class and using toString(), equals(), hashCode(), clone(), finalize() etc
* Using Runtime Class, Process Class to play music, video from Java Program
* Primitives and Wrapper Class
* Math Class
* String, StringBuffer, StringBuilder Class
* String Constant Pool.
* Various usage and methods of String, StringBuffer, StringBuilder
* Wrapper Classes
* System Class using gc(), exit(), etc.

**3.3 New Concepts in package**

* Atuoboxing and Autounboxing
* Static import.
* Instanceof operator.
* Enum and its use in Java
* Working with jar

**3.4 Garbage Collection**

* Garbage Collection Introduction
* Advantages of Garbage Collection
* Garbage Collection Procedure
* Java API
* Interview related Question and Answer.

# 4: > Exception Handling

# In this section we will see the problem during the program execution and we will overcome the problematic situation by exception handling mechanism.

* Introduction to Exceptions
* Effects of Exceptions
* Exception Handling Mechanism
* Try,catch,finally blocks
* Rules of Exception Handling
* Exception class Hierarchy, Checked & Unchecked Exception
* Throw & throws keyword
* Custom Exception Class
* Chained Exception.
* Resource handling & multiple exception class.
* Problem Solving
* Interview related Question and Answer.

# 5: > Multithreading

# In this section we will see how run as well as to eat together. Here we learn how to execute more than one task simultaneously in our Java Program.

* Introduction
* Advanages
* Creating a Thread by inheriting from Thread class
* Run() and start() method.
* Constructor of Thread Class
* Various Method of Thread Class
* Runnable Interface Implementation
* Thread Group
* Thread States and Priorities
* Synchronization method, block
* Class & Object Level Lock
* Deadlock & its Prevention
* Interthread Synchronization
* Life Cycle of Thread
* Deprecated methods : stop(), suspend(), resume(), etc
* Problem Solving
* Interview related Question and Answer.

# 6: > GUI : Applet, AWT, Event Handling

# In this section we will leave the black and white output screen, rather we will make java colorful. In this module we will create graphical items in a Java Program.

**6.1 Applet(java.applet)**

* Introduction & Advantage of Applet
* How to create and run an Applet in browser and appletviewer
* Life Cycle of Applet
* Using Graphics, Color, Font and other classes in Applet to draw Shapes, String, Images
* Creating Banner in Applet
* AppletContext interface. Using AudioClip interfaces to play music.
* Problem Solving

**6.2 Abstract Window Toolkit(java.awt)**

* AWT Hierarchy
* Using Component classes like Button, TextArea, TextField, Checkbox, Label, Choice, List, etc
* Using Container classes line Applet, Panel, Frame, Window, Dialog(Open & Save Dialog)
* Using Layout Manager to organize component on a container.
* Using Borderss, Menus, Toolbars, Dialogs
* Using setBounds() to place component on Frame/Applet.
* Problem Solving

**6.3 GUI Event Handling( java.awt.event)**

* Delegation Event Model
* What is Events and stepsfor Event Handling
* Using different Event Classes to generate event
* Handling different events by respective Event Listeners
* Using Event in Applet & Event
* Handling the event in different ways.
* Adaptor Classes
* Problem Solving

**6.4 Using Applet and Frame**

* Passing Parameter from HTML to Applet
* Inter Applet Communication
* Communication between Frame.
* Customizing TextField, Frame, etc
* Using NetBean IDE
* Problem Solving
* Interview related Question and Answer.

# 7: > Input Output Streams

# In this section we will use various read and write operation required on IO device, Socket and on Web. In this module we will create graphical interface to create some window application in a Java Program.

**7.1 Java I/O Stream**

* I/O Stream - Introduction
* Types of Streams
* Stream Class Hierarchy
* Using File Class
* Copy and Paste the content of a file.
* Byte Streams vs Character Streams
* TextFile vs Binary File
* Character Reading from Keyboard by InputStreamReader
* Reading a Line/String from Keyboard by BufferedReader
* Standard I/O Streams
* Using Data Streams to read/write primitive data
* PrintStream vs PrintWriter
* Using StreamTokenizer and RandomAccessFile.
* Using nio package.
* Problem Solving

**7.2 Serialization**

* Introduction to Serialization
* Using Object Streams to read/write object
* Transient Keyword
* Serialization Process
* Deserialization Process
* Problem Solving
* Interview related Question and Answer.

# 8: > Networking

# In this section we will communicate with different Java Program by using various read and write operation on Socket and on Web. In this module we will create graphical interface to create chat application window application in a Java Program.

**8.1 Networking Basics**

* What is IP Address
* What is Protocol
* What is Ports
* What is Client/Server Architecture
* What is Sockets

**8.2 Java Networking**

* INetAddress class
* ServerSocket and Socket Class
* DatagramSocket and DatagramPacket Class
* URL & URLConnection class
* MultiCastSocket class
* Creating chat application
* Problem Solving
* Interview related Question and Answer.

# 9: > Collection Framework(Java Data Structure)

# In this section we will brush up the searching, sorting algorithms that we have learned in Data Structure. In this module we will learn the different implementation of Java Data Structure in a Java Program.

**9.1 Generics( Templates)**

* What is generic
* Creating User defined Generic classes

**9.2 The java.util package.**

* Collection
* What is Collection Framework
* List, Set & Map interfaces
* Using Vector, ArrayList, Stack, LinkedList, etc.
* Using Collections class for sorting
* Using Hashtable, HashMap, TreeMap, SortedMap, LinkedHashMap etc.
* Iterator, Enumerator.
* Using Que, Deque, SortedQue, etc.
* Using HashSet, TreeSet, LinkedHashSet etc .
* Using Random class
* Using Properties in a Java Program
* Using user defined class for DataStructure
* Using Date and Formatting Date class.
* Problem Soving
* Interview related Question and Answer.

# 10: > Introduction to Swing (Java Foundation Classes)

# In this section we will brush up the GUI that we have learned in java.awt package. In this module we will learn the different classes and related event handling of javax.swing package in a Java Program.

* Introduction to Swing and MVC Architecture
* Light Weight Vs Heavy Weight Componet
* Swing Hierarchy, Swing Model
* Atomic Component JButton, JList, etc.
* Intermediate Containers JPanel, JSplitPane, etc
* Top Level container JFrame, JApplet
* Using JTree, JProgrssBar, JTable
* Problem Solving
* Interview related Question and Answer.

# 11: >  Remote Method Invocation (Distributed Application in Java)

# In this section we will learn the concept of Distributed application and its need. In this module we will learn how to write a Java Program to create distributed application.

**11.1 Need for RMI**

* RMI Introduction
* Efficiency

**11.2 RMI Architecture**

* Remote Interface
* Stub and Skeleton
* Remote Object

**11.3 RMI Communication Model**

* RMI Control Flow
* Marshaling
* Unmarshaling
* Using RRL

**11.4 Implementing RIM**

* Analyzing Remote interface, UnicastRemoteObject class.
* Running Different examples.
* Rmic, rmiregistry, etc.
* Using Examples.
* Problem Solving
* Interview related Question and Answer.

# 12: >  JDBC (Java Data Base Connection)

# In this section we will learn the basic knowledge required for the data base interaction from a java progam. In this module we will learn how to interact with different data base from a Java Program.

* Introduction to JDBC
* Databases and Drivers
* Types of Driver
* Loading a driver class file
* Estabalishing the Connection to different Database with different Driver.
* Executing SQL queries by ResultSet, Statements , PreparedStatment interface.
* Using CallableStatement.
* Transaction Management & BatchUpdate.
* Problem Solving
* Interview related Question and Answer.

**Advance Java Syllabus**

**Coverage**

|  |
| --- |
|  SQL BASICS |
|  JDBC (Java Data Base Connection) |
|  HTML and JavaScript |
|  SERVLET |
|  JSP (Java Server Pages) |
|  JSF (Java Server Faces) |
|  Facelets |
|  Web Services |
|  EJB (Enterprise Java Bean) |
|  AJAX |
|  Using IDE (Integrated Development Environment) |
|  Project |

* It makes the student to be a Web Developer in Java.
* This module trains to excel in IT Industry and how to become a Java professional.
* A trainee of this module should complete classroom task as well as home task.
* After the completion of the course one can appear and clear Oracle’s OCWCD exam.

**Syllabus in Details**

**1: SQL BASICS**

**2: JDBC**

* What is Persistency ?
* What is JDBC, ODBC ?
* JDBC architecture.
* Loading and Connecting to database.
* Using Connection, Statement, ResultSet.
* Data Fetching, Updatable & Scrollable methods of ResultSet.
* Types of Driver, Using different types of Driver.
* Using PreparedStaement
* Using Callable Statement to invoke procedure &, functions.
* Transaction Management,
* Using ResultSetMetadata.
* Connection to different data bases like MS-ACCESS, MYSQL, ORACEL etc.
* Batch Processing
* Problem Solving & Interview Question.

**3: HTML and JavaScript**

**4: SERVLET**

* Introduction to Server Side Programming..
* Web Server.
* Web Container.
* The Servlet API.
* The Servlet Interface.
* Servlet Life Cycle.
* Deployement of Web Application.
* What is Deployement Descriptor.
* Creating a servlet by inheriting from GenericServlet class.
* Creating a servlet by inheriting HttpServlet class.
* The ServletConfig Interface.
* The ServletContext Interface.
* Permanent Servlet.
* Servlet Communication.
* Servlet Chaining(Forward & Including the request).
* Send Redirect .
* Stateless and Stateful protocols.
* Session Tracking Mechanisms ( HiddenField, URL Rewriting, Cookie, Session).
* Servlet Filters.
* Listeners
* Web-Security.
* Types of Variables in Servlet.
* Problem Solving & Interview Question.

**5: JSP (Java Server Pages)**

* Introduction to Java Server Pages.
* Architecture & Anatomy of JSP Page.
* JSP life cycle, JSP with MVC Architecture.
* Dynamic webpage Creation.
* Scripting Elements.
* Database access from JSP.
* Servlet vs JSP.
* User tracking in JSP.
* Error handling in JSP.
* Types of variables.
* Implict object in JSP.
* Forwarding & including the request.
* Static include.
* Servlet-JSP interaction.
* Using Java Beans(Simple, Indexed, Boolean, Bound, Constraint Beans).
* Creating coustom tags.
* Creating a Small Mail like project.
* Problem Solving & Interview Question.

**6: JSF (Java Server Faces)**

* Introduction to Java Server Faces.
* [What Is a JavaServer Faces Application ?](https://docs.oracle.com/javaee/6/tutorial/doc/bnapk.html)
* [JavaServer Faces Technology Benefits](https://docs.oracle.com/javaee/6/tutorial/doc/bnapj.html).
* [Creating a Simple JavaServer Faces Application](https://docs.oracle.com/javaee/6/tutorial/doc/gjaam.html).
* Problem Solving & Interview Question.

**7: Facelets**

* Introduction to Facelets.
* [What Is Facelets?](https://docs.oracle.com/javaee/6/tutorial/doc/gijtu.html)
* [Developing a Simple Facelets Application](https://docs.oracle.com/javaee/6/tutorial/doc/gipob.html)
* [Creating a Facelets Application](https://docs.oracle.com/javaee/6/tutorial/doc/gipob.html#giqte)
* Problem Solving & Interview Question.

**8: Web Services**

* Introduction to Web Services.
* [Building Web Services with JAX-WS](https://docs.oracle.com/javaee/6/tutorial/doc/bnayl.html)
* [Building RESTful Web Services with JAX-RS](https://docs.oracle.com/javaee/6/tutorial/doc/giepu.html)
* JNDI (Java Naming Directory Interface)
* Interview Question.

**9: EJB (Enterprise Java Bean)**

* Introduction to Web Services.
* Enterprise JavaBeans Technology
* EJB Component Architecture
* Role of EJB & its life cycle.
* Types of Beans
* Stateless and Stateful beans
* Java Persistence API
* Security in Java EE.
* Problem Solving & Interview Question.

**10: AJAX**

**11: Using IDE (Integrated Development Environment)**

**Java Framework Syllabus**

**Coverage**

|  |
| --- |
| * Introduction to Spring: |
| * Spring Core (Basic Concepts) |
| * Spring Core (Advanced Concepts) |
| * Spring Core (3.0 Annotations): |
| * Spring AOP : |
| * Spring Transaction |
| * Spring JDBC |
| * Spring MVC |
| * Spring ORM |
| * Introduction To Hibernate |
| * Hibernate Annotations vs JPA Annotations: |
| * Hibernate Mappings |
| * Hibernate Query Languages and Transactions and Caching |
| * Spring Hibernate Integrations |
| * Hibernate uses in Project: |

**Syllabus in Details**

|  |  |  |
| --- | --- | --- |
| **Objectives:** |  | * To be able to work on any enterprise level project * To facilitate the development of distributed, web-enabled applications using Spring Framework * Configure Hibernate and integrate with Spring application * Know how and when to use Spring Framework * To gain hands on experience |
|  | | |
| **Intended Audience:** |  | This training is designed for developers/fresher’s interested in learning Spring framework |
|  | | |
| **Prerequisites:** |  | * Basic understanding of JDBC * Knowledge on Servlet/JSP * Basic knowledge on any application server(Tomcat) |

1. **Introduction to Spring:**

* What is Spring,
* How Spring fits into the Enterprise world,
* Spring Modules

1. **Spring Core (Basic Concepts):**

•What is a Core Container   
• Introduction to IOC   
• Types of DI  
 Setter DI vs Constructor DI  
• Resolving Constructor Confusion  
• Collection DI  
• Bean Inheritance  
• Collection Merging  
• Inner Beans  
• Using IDRef  
• Bean Aliasing  
• Bean Scopes  
• Inner Beans  
• Null String  
• Bean Auto wiring  
• Nested Bean Factories

1. **Spring Core (Advanced Concepts):**

• P – Namespace  
• Dependency Check  
• Depends On  
• Factory Beans  
• Static Factory Method  
• Instance Factory Method  
• Aware Interfaces  
• Bean Lifecycle  
• Method Replacement  
• Lookup Method Injection  
• Property Editors  
• Bean Factory Postprocessor

1. **Spring Core (3.0 Annotations):**

• Spring VS Java Config Project annotations  
• @Required  
• @Autowire  
• @PostConstruct  
• @PreDestroy  
• @Qualifier  
• @Resource  
•@Component  
• @Service  
• @Controller  
• @Named

• Various Annotation based Post Processors  
 (AutowiredAnnotationBeanPostProcessor,  
 CommonAnnotationBeanPostProcessor,  
 PersistenceAnnotationBeanPostProcessor,  
 RequiredAnnotationBeanPostProcessor)

1. **Spring AOP :**

• AOP Concepts  
• Programmatic VS Declarative AOP  
• Programmatic AOP  
 i. Types of Advices  
 ii. Types of Pointcuts  
 iii. Working with proxies  
• Declarative AOP  
 i. Using AOP 2.0 Config element  
 ii. OGNL Expressions  
• Aspect J AOP

1. **Spring Transaction :**

• Aop 2.0 Configuration driven Transaction Management  
• Aspect J annotation based Transaction Management

1. **Spring JDBC :**

• What is DAO pattern?  
• Ways to implement Spring DAO  
• Choosing an approach for JDBC database access  
• JDBC Template  
• Executing Statements  
• Running Queries  
• SQL Parameters

1. **Spring MVC :**

• Spring 3.0 features  
• Introduction to Spring MVC  
• Handler Mapping  
• Controllers  
• Validations  
• Handler Interceptors  
• Views  
• Form tags

1. **Spring ORM :**

• Integrating with Hibernate  
• Creating and Injecting Hibernate Session Factory  
• Managing Transaction

**Hibernate Syllabus**

|  |  |  |
| --- | --- | --- |
| **Objectives:** |  | * To be able to work on any enterprise level project * To facilitate the development of distributed, web-enabled applications * Configure Hibernate and integrate with Spring application * To know the usage of hibernate in real time projects * Know how and when to use Spring Framework * To gain hands on experience |
|  | | |
| **Intended Audience:** |  | This training is designed for developer interested in using Hibernate with Spring. |
|  | | |
| **Prerequisites:** |  | * Basic understanding of JDBC * Knowledge on Servlet/JSP/Spring Core * Basic knowledge on any application server(Tomcat) and Database |

**Introduction To Hibernate :**

* Need for Hibernate
* Hibernate and ORM (Object-Relation Mapping)
* POJOs (Plain Old Java Objects) and the Data Layer
* Hibernate Over Entity Beans
* Understanding Hibernate Architecture
* Configuration
* SessionFactory
* Session
* Query
* Criteria
* Hibernate Configuration
* Hibernate Mappings
* Persistent Classes
* Working with Hibernate to perform basic **CRUD** Operations
* Configuring Mappings Using Annotations.

**Hibernate Annotations vs JPA Annotations:**

* Getting started with Hibernate 3.X
* Using JPA annotations and XML configuration
* Hibernate Domain models
* Hibernate Persistence Context
* Mapping persistence Classes
* Hibernate Inheritance

**Hibenate Mappings:**

* Component Mapping
* Inheritance Mappings
* Table Per Class Hierarchy
* Table Per Sub Class
* Table Per Concrete Class
* Association Mappings
* One -to-One
* One -to-Many
* Many -to-One
* Many -to-Many

**Hibernate Query Languages and Transactions and Caching:**

* Using queries: HQL, criteria API, native (SQL)
* Hibernate Transaction
* Mixing Hibernate and JDBC
* Hibernate second level caching

**Spring Hibernate Integrations:**

* Spring Hibernate Integration
* Data source creation
* Hibernate DAO implementation using Spring Framework
* Spring – JDBC Transaction
* Spring AOP – Integration

**Hibernate uses in Project:**

* Design Web Application using hibernate
* Hibernate in web application(case studies)
* Project Guidance

**Oracle Syllabus**

**Coverage**

|  |
| --- |
|  RDBMS Concept |
|  DDL (Data Definition Language) Statements |
|  DML (Data Manipulation Language) Statements |
|  Select Statement |
|  Single Row Function |
|  Multiple Row Functions |
|  Displaying Data from Multiple Tables |
|  Sub query |
|  Other database objects –Views, Synonym, sequence |
|  Set operator |
|  Oracle Architecure |
|  User control |
|  Advance SQL |
|  Introduction to PLSQL |
|  Anonymous PLSQL |
|  Named PLSQL Subprograms |
|  Triggers |
|  Advance PLSQL |

**Syllabus in Details**

1. RDBMS Concept

* Basic Introduction and Technology Awareness
* 3 Layer Architecture
* Data Models
* Entity types
* Types of Keys
* Types of attributes
* ER Modeling
* Normalization
* Assignments
* Industry (MNCs) standard examples

1. DDL(Data Definition Language) Statement

* Introduction to SQL
* Types of Datatype
* Create ,Alter, Drop, Trunctae commands
* Types of constraints (Primary key, Foreign key, Unique key, Not Null key, Check constraint)
* Assignments
* Industry (MNCs) standard examples

1. DML(Data Manipulation Language) statements

* Insert
* Update
* Delete
* TCl (Transaction control Statement ) – Commit, Rollback , Savepoint
* Assignments
* Industry (MNCs) standard examples

1. Select Statement

* Simple select using from clause
* Where clause
* Using like operator
* Arithmetic operators
* Distinct Keyword
* Assignments
* Industry (MNCs) standard examples

1. Single Row Function

* Character Functions
* Date Funtions
* Conversion functions
* Case Decode Function
* Assignments
* Industry (MNCs) standard examples

1. Multiple Row Functions

* Aggregate Function(Sum, Avg, Count, max, min, variance, stddev)
* Group by clause
* Having Clause
* Assignments
* Industry (MNCs) standard examples

1. Displaying Data from Multiple Table

* Inner join
* Right Outer join
* Left Outer Join
* Full Outer Join
* Assignments
* Industry (MNCs) standard examples

1. Sub query

* Independent Sub query
* Correlated Sub query
* Assignments
* Industry (MNCs) standard examples

1. Other database objects –Views, Synonym, sequence

* Assignments
* Industry (MNCs) standard examples

1. Set operator

* Union
* Union All
* Minus
* Intersect
* Assignments
* Industry (MNCs) standard examples

1. Quiz

* SQL quiz as per interview standard

1. Oracle Architecture
2. User control
3. Advance SQL

* Merge Statement
* Connect by prior
* With clause
* Rownum
* Analytic functions(Rank ,dense rank,lag ,lead etc)
* Assignments
* Industry (MNCs) standard examples

1. Introduction to PLSQL

* PLSQL Architecture
* Use of PLSQL in industry

1. Anonymous PLSQL

* Types of Data type
* PLSQL Block
* Data type declaration
* Anchored declaration
* Conditional statements ( if else)
* Looping (Loop, while, for)
* Using DML in PLSQL
* Exception Handling
* Cursor
* Assignments
* Industry (MNCs) standard examples

1. Named PLSQL Subprograms

* Local Procedure
* Local Function
* Stored Procedure
* Stored Function
* Assignments
* Industry (MNCs) standard examples

1. Triggers

* Statement Level Trigger
* Row level trigger
* Instead of trigger
* Assignments
* Industry (MNCs) standard examples

1. Advance PLSQL

* Package
* Collection variable (Nested Table, Varray, Associative array)
* Ref Cursors
* Bulk Collect
* Native SQL
* Dynamic SQL
* Assignments
* Industry (MNCs) standard examples

1. Quiz

* SQL quiz as per interview standard

**.Net and Advance .Net Syllabus**

|  |
| --- |
| **Coverage**  Console Application |
|  NER Framework   Over view of OOPS |
| Language Fundamentals |
|  Arrays And Procedures |
|  Classes and Objects |
|  Inheritance/Polymorphism |
|  Delegate |
|  System Defined Function |
|  SQL Server |
|  Data structure |
|  ADO.NET Objects |
|  The Data adapter object |
|  The data reader object |
|  The dataset object |
|  Navigating through dataset |
|  Updating your database by using datasets |
|  Managing concurrency ADO.net objects to retrieve LINQ |
| Threading |
|  Exception handling   |  | | --- | | **WINDOWS APPLICATION** | |  Tools and Controls | |  Threading | |  Exception handling | | Event Handling | |  MDI and SDI | | Graphics | | Advanced Controls | | Crystal reports | |  ADO.NET Objects | | The Data adapter object | |  The data reader object | | The dataset object | | Navigating through dataset | |  Updating your database by using datasets | | Managing concurrency ADO.net objects to retrieve | |  I/O Operation | | LINQ | |

|  |
| --- |
|  |
| WEB APPLICATION |
| · IIS Server |
| · Introduction to ASP |
| · Web Controls |
| · Web application with ASP.net |
| · State Management |
| · XML |
| · Web service |
| · Ajax and Javascript |
| · Security |
| · LINQ |
| · WPF |
| · WCF |
| · Silverlight |
| · MVC |
| · CSS |
| · Deployment  **Syllabus in Details** |
| |  |  | | --- | --- | | **Module** | **Contents** | | .NET Framework | Architecture & its details | |  | Comparision between Java & .net | | Over view  of OOPS | Introduction to Object | | Concept of class, Abstract class | | Data Encapsulation | | Comparision between Java & C++ | | Function Template, method overloading and overriding | | Language Fundamentals | How to write a .net program | |  | examples with practical variable, | |  | Constants | | Arrays And Procedures | Variables | |  | Constants | |  | examples with practical | |  | programming with practical | |  | programming with practical | | Classes and Objects | Components of class | |  | Procedures, functions | |  | system defined functions | |  | Constructors | | Inheritance/Polymorphism | Over loading, overriding | | Inheritance restrictions | | Types of inheritance | | Delegate | whatis Delegate | | ststem defined functions | |  |  | | Windows Applications | Windows form Designer | |  | The controls collection | |  | multiple form application | |  | menus | |  | MDI applications | | Tools and Controls | All kinds of controls and its uses | |  | practice | | Threading | Multitasking, Understand  Thread | |  | Multithreading, creating  Threading | |  | implementing a runnable  interface | | Exception  handling | The cause of Exception  and its remedies | |  | Types of exceptions | |  | Implementation of Exception | |  | Types of Errors | |  | debugging | |  | Throwing your own exceptions | | Event Handling | Event Handlers-1 | |  | Event Handlers-1 | | Graphics and UI | Comparative study of SDI,MDI | | SQL | starting with SQL 2010 | | Implementing data integrity & Maintaining databases | | T-SQL | | Statement Procedure, Trigger | | Grouping Results | | Joins | | How to take back up and its details | | Fundamentals of database design | | database objects | | Stored procedures | | Normalization rules | | joins, constraints | | Advanced Controls | Tree View, | | Menu strip, context menu strip | | Crystal reports | Report creation | | Design the report& print Report | | ADO.NET Objects | Dataadapter,Datatable | | The datareader object | | Dataset,Command object | | Navigating through dataset | | The Dataadapter object | The Dataadapter object | | The datareader object | The datareader object | | the dataset object | the dataset object | | Navigating through dataset | Navigating through dataset | | Updating your database by using datasets | Updating your database by using datasets | | Managing concurrency ADO.net objects to retrieve | Managing concurrency ADO.net objects to retrieve | | IIS Server | Web Server mechanisms | | Introduction to ASP | What is ASP.net? | | Need of ASP.net. | | What does asp.net do? | | fundamentals of ASP.net. | | Creating your 1st application | | Web Controls | Controls with server side objects | | Code behind programming | | Program/form Layout | |  |  | | Web application with ASP.net | Coding ASP.net pages | | Inline login project | | Testing the login project | | code behind web forms | | data input with asp.net | | accessing data in asp.net | | connecting to database | | Running data in asp.net | | State Management | Working with Cookies, Caching | | | CacheDependancy | | XML | Working with XML Tag | | Adrotator | | Converting XML to Dataset and | | Dataset to Xml with example | | Web service | Working with web service | | Ajax and Javascript | Working with ajax & Jscript & HTML | | Security | web.config and machine.config | | LINQ | LINQ Introduction and implemantation | | database connection using LINQ | | WPF | Introduction | | Controls and form design and implemantation | | WCF | different Services | | Silverlight | Silverlight Application | |  |  | | MVC | Architecture and role | | Application on MVC | | CSS | Different Type of CSS and its Implemantation | | Deployment | Deployment | |

**Php Syllabus**

**Coverage**

|  |
| --- |
|  WebApplicationOverview |
|  Language Fundamental |
|  HTML |
|  Java script |
|  CSS |
|  Form Processing |
|  File Handling |
|  Session Management |
|  PHP – MySQL |
|  SQL Injection |
|  AJAX |
|  OOPS |
|  PDO |
|  Cake PHP |
|  Bootstrap |
|  Angular JS   |  |  | | --- | --- | | **Syllabus in details** |  | |  |  | | **Introduction** | **Encryption Mechanism** | | o   Web Application | o   md5 | | o   Understanding client/server roles | o   sha1 | | o   XAMPP Installation | o   crc32 | | o   Apache, PHP, MySQL | o   base64\_encode | |  | o   base64\_decode | | **PHP Language Fundamental** |  | | o   Basic syntax | **MySQL Database** | | o   Data Types | o   Introduction | | o   Variables | o   SQL(DDL, DML, DQL, TCL, DCL) | | o   Constants | o   Data types | | o   Operators | o   Functions | | o   Control Structures | o   Keys | | o   Loops | o   Join | | o   Array | o   Procedure | | o   Function |  | |  | **PHP – Database Application** | | **Form Processing** | o   Connect to MySQL Server | | o   Get and Post Method | o   CURD Operation | | o   \_GET and \_POST Array | o   Registration Application | | o   \_REQUEST Array | o   Login Application | | o   Form Validation | o   SQL Injection | | o   File Uploading |  | | o   File Downloading | **PHP – AJAX Application** | |  | o   Introduction to AJAX | | **String** | o   Features of AJAX | | o   Strings and Patterns | o   Creating First AJAX Application | | o   Matching | o   Assignment | | o   Extracting |  | | o   Searching Replacing | **PHP OOPS** | | o   Formatting | o   Introduction | | |  | | --- | |  | | o   Declaring a class | | **File Handling** | o   Objects | | o   Understanding file& directory | o   constructor | | o   Opening and closing a file | o   Destructor | | o   File Operation | o   Public ,private, protected | | o   Working with directories | o   Static properties and method | |  | o   Inheritance | | **Session Management** | o   Polymorphism | | o   URL Rewriting | o   Parent:: & self:: keyword | | o   Hidden Field | o   Instanceof operator | | o   Cookie | o   Abstract method and class | | o   Session | o   Interface | |  | o   Exception Handling | | **HTML** | **MySQLi and PHP Data objects** | | o   Introduction | o   Introduction | | o   Basic Tags | o   Installation | | o   Text Formatting Tags | o   Connection with MySQL | | o   Entities | o   Create a MySQL database | | o   Lists | o   Create a MySQL tables | | o   Images | o   Perform CURD Operation | | o   Links | o   Login Module implementation | | o   Tables | o   Registration Module implementation | | o   Forms |  | | o   Frames | **Cake PHP** | | o   Special Tags | o   Introduction | | o   Head part | o   Installation | | o   Designing website using div | o   Folder Structure | |  | o   Model | | **CSS** | o   View | | o   Style sheet Basics | o   Controller | | o   Various selectors | o   Core Libraries | | o   Properties | o   Global Constants and Function | | o   Values of each property | o   Helpers | |  | o   Plug-in | | **Java Script** |  | | o   Introduction | **Bootstrap** | | o   Variables | o   Introduction | | o   Data Types | o   Grid | | o   Operators | o   Tables | | o   Control Structures | o   Images | | o   Functions | o   Button | | o   Events | o   Button Group | | o   Document Object Model | o   Progress Bar | | o   Form Validations | o   Pagination | | o   Regular Expressions | o   Forms | |  | o   Modal | | **Angular JS** | o   Tooltip | | o   Introduction |  | | o   Expression |  | | o   Directive |  | | o   Controllers |  | | o   Filters |  | | o   Modules |  | | o   Events |  | |

**Android Syllabus**

**Coverage**

|  |
| --- |
|  Android Overview and History |
|  Android Architecture |
|  Designing The User Interface |
|  Activity |
|  Intent |
|  Service |
|  Broadcast Receiver |
|  Content Provider |
|  Multimedia in Android |
|  Files |
|  SQLite database |
|  Telephony and SMS API |
|  Networking |
|  Maps, Geocoding and Location-Based Services |
|  Phone Gap   |  |  | | --- | --- | | **Syllabus in details** |  | |  |  | | **Android Overview and History** | **Intent** | | o   What is Android? | o   Introduction | | o   Importance of Android | o   Implicit Intent | | o   Version History | o   Explicit Intent | | o   Android Architecture | o   Phone call | | o   Install Android Studio | o   Send SMS | | o   Android Emulator | o   Capture Image | |  | o   Audio Recording | | **Android Building Blocks** | o   Video Recording | | o   Activities | o   Returning Results from Activities | | o   Services |  | | o   Content Providers | **Files** | | o   Broadcast Receivers | o   Shared preference | |  | o   Internal Storage | | **Developing and Running Android Application** | o   External Storage | | o   Hello World App |  | | o   Project structure | **SQLite database** | | o   The manifest file | o   Introduction to SQLite | | o   Layout resource | o   Opening and closing a database | | o   Running your app on Emulator and Device | o   Creating tables | |  | o   Cursor | | **Designing The User Interface** | o   CURD Operation | | o   Form widget |  | | o   Spinner | **Telephony and SMS API** | | o   List View | o   SMS | | o   Gallery | o   Calls | | o   Auto complete Text View | o   Emails | | o   Selection components | o   Email using Java Mail API | | o   Adapters |  | | o   Complex UI components | **Maps, Geocoding and Location Based Services** | | o   Building UI for performance | o   Introduction to Google Maps | | o   Menus and Dialogs | o   MapsView | | o   Graphics & animations | o   MapFragment | | o   Web View | o   Markers | |  | o   Get Google Places API Access | | **Android Layout Design** | o   List of search options to get data (like hospital, school, etc…) | | o   Linear Layout |  | | o   Relative Layout | **Phone Gap** | | o   Grid Layout | o   Introduction Apache Cordova Phone Gap | | o   Frame Layout | o   Advantages of Phone Gap | | o   Table layout | o   Phone Gap Components | | o   Fragment | o   How to setup Phone Gap Environment | | **Android Services** | o   Sample projects on Phone Gap | | o   Introduction |  | | o   Started Service |  | | o   Bind Service |  | | o   Assignment |  | |  |  | | **Networking** |  | | o   Connecting to an Internet Resource |  | | o   Parsing XML Using the XML Parser |  | | o   Downloading files |  | | o   Internet Services |  | |  |  | | **Content Providers** |  | | o   Registering Content Providers |  | | o   Publishing Your Content Provider’s URI Address |  | | o   Creating the Content Provider’s Database |  | | o   Implementing Content Provider Queries |  | | o   Content Provider Transactions |  | | o   Storing Files in a Content Provider |  | |  |  | | **Animation** |  | | o   Tweened View Animations |  | | o   Creating Tweened View Animations |  | | o   Applying Tweened Animations |  | | o   Using Animation Listeners |  | | o   Animating Layouts and View Groups |  | | o   Creating and Using Frame-by-Frame Animations |  | | o   Interpolated Property Animations |  | | o   Property Animations |  | | o   Property Animation Sets |  | | o   Animation Listeners |  | |  |  | |

**Embedded System Syllabus**

**Coverage**

|  |
| --- |
|  Introduction |
|  Basic Electronics Components |
|  Power supply Design |
|  Overview of Digital Electronics |
|  Semiconductor Memories |
|  Basics of Programming Language |
|  Programming in C |
|  Introduction to Microcontroller |
|  Memory Mapping |
|  Assembly language programming |
|  Software Development Tools |
|  LED Interfacing |
|  LCD Interfacing |
|  Development board testing |
|  SEVEN SEGEMENT DISPLAY interfacing |
|  Matrix keyboard interfacing |
|  FOURTEEN SEGMENT interfacing |
|  ROBOTICS basics |
|  ACTUATORS |
|  DC MOTOR interfacing |
|  New technologies and sensors |

**Syllabus in Details**

1. Introduction

* Introduction to Embedded System
* Basics Of Embedded System
* Application of ES
* Examples of ES

1. Basic Electronics Components

* Resistor
* Capacitor
* Diode
* PNP transistor
* NPN transistor
* LED
* Step down Transformer
* Rectifier
* 7805 Voltage regulator IC
* Relay
* IR sensor
* PIR sensor
* LDR
* V-POT
* Ohm’s and Watt’s Law

1. Power supply Design

* Power Supply Circuit
* Practical design

1. Overview of Digital Electronics

* Binary,Octal,Decimal,Hexadecimal
* Gate : AND,OR,NAND,NOR,XOR,INVERTER
* Gate IC’s

1. Semiconductor Memories

* Address bus
* Data bus
* RAM
* ROM
* VONN NEUMANN architecture
* HARVARD architecture

1. Basics of Programming Language

* Compiler
* Interpreter
* Stages of compilation
* Data Types
* Modifiers
* Big-endian & Little-endian
* ASCII table
* Variables
* Operator

1. Programming in C

* Examples on different concepts
* Control structure
* Functions
* Storage classes
* Pointers
* Assembly language intro

1. Introduction to Microcontroller

* MP vs MC
* 8051 MC
* 8051 architecture
* 89s52 pin diagram
* Simulator vs Emulator

1. Memory Mapping

* Memory Addressing
* Program Memory – ROM
* Data Memory – RAM
* Internal RAM
* Registers
* Bit addressable memory
* Special Function Register

1. Assembly language programming

* Instruction set and Programming
* Immediate Addressing
* Register Addressing
* Direct Addressing
* Indirect Addressing
* Indexed Addressing

1. Software Development Tools

* KEIL
* Installation of KEIL
* PROTEUS
* Installation of PROTEUS
* USBasp driver
* Installation of USBasp driver
* USB to Serial Driver
* Installation of USB to Serial Driver

1. LED Interfacing
2. LCD Interfacing
3. Development board testing

* Burning into MC

1. SEVEN SEGEMENT DISPLAY interfacing
2. Matrix keyboard interfacing
3. FOURTEEN SEGMENT interfacing
4. ROBOTICS basics

* What is robotics
* Different types of ROBOT
* Manual and Autonomous ROBOT
* Basic parts
* Application of ROBOTICS & ROBOT

1. ACTUATORS

* Different types
* Examples
* Electrical actuators
* Application of actuators

1. DC MOTOR interfacing

* Motor driver
* Manual robot making

1. New technologies and sensors

* RFID
* GSM
* GPS
* BLUETOOTH
* DTMF
* RF
* ULTRASONIC SENSOR

1. CONCLUSION : EMBEDDED SYSTEM, ROBOTICS PROJECTS & APPLICATIONS

**Vlsi Design Syllabus**

**Coverage**

|  |
| --- |
|  VLSI design flow and methodologies |
|  FPGA architecture |
|  Digital Logic Creation Techniques |
|  Real Life Event Logic Design(Traffic Light Control, Three Way Switch, Car Locking) |
|  Digital Electronics logic design |
|  Verilog coding |
|  Synthesis and Verification |
| Advanced Digital electronics circuit design using FSM |
|  CMOS fundamentals |
|  CMOS circuit design |
|  Layout Design |
|  Industry Oriented Projects |
|  IP design methodologies |
|  Next generation Transistor(TFET study)  **Syllabus in details** |

Introduction

* What is VLSI?
* Introduction to CMOS technology
* Demanding area for company
* Application of VLSI
* Indian ESDM market-Analysis of Growth & Opportunity Plan
* VLSI design methodologies
* VLSI Design Flow

Digital IC Design

* Overview of Digital Design with Verilog HDL(Day-2)
* Evolution of Computer Aided Digital Design
* VHDL vs Verilog
* Importance of Hardware Description Language(HDL)
* Popularity of Verilog HDL
* FPGA Architecture
* FPGA vs ASIC
* Hierarchical Modeling concept
* Design of 4-bit ripple carry adder
* Modules
* Instances
* Components of a simulation
* **Example: Design Block, Stimulus Block**
* Basic concepts of Verilog coding
* Lexical conventions
* Number Specification
* Strings
* Identifiers and Keywords
* Data Types
* System Tasks and Compiler directives
* Modules and Ports
* Modules
* Ports
* Port connection rules
* Connecting ports to external signals
* Gate-Level Modeling
* Gate types
* Gate Delays
* Dataflow Modeling
* Continuous Assignments
* Delays
* Expressions, operators and operands
* Operator type
* **Design: 4:1 MUX, 4-bit full adder, Ripple Counter**
* Behavioral Modeling
* Structured Procedures
* Procedural Assignment
* Timing Control
* Conditional statements
* Multiway branching
* Loops
* Sequential and parallel blocks
* Generate Blocks
* **Design: 4:1 MUX, 4-bit counter, Traffic signal controller**
* Tasks and Functions
* Difference Between task and function
* Tasks
* Automatic(Re-entrant) task
* Functions
* Automatic(Recursive ) function
* Useful Modeling Techniques
* Procedural Continuous Assignments
* Use of force and release
* Overriding Parameters
* Conditional Compilation and execution
* Time Scale
* Useful system tasks
* Switch Level Modeling
* Switch modeling Elements
* MOS switches
* CMOS switches
* **Design: CMOS NOR gate, 2:1 MUX, Simple CMOS Latch**
* Logic Synthesis and Verification
* What is logic synthesis?
* Impact of logic synthesis
* Verilog HDL synthesis
* Synthesis Design Flow
* Verification of Gate-level netlist
* Modeling tips for logic synthesis
* Design Partitioning
* Sequential Circuit Synthesis

Analog IC Design

* Fabrication Process of MOSFETs
* Introduction
* Basic steps
* CMOS Technology
* Layout Design Rules
* Layout design using *Microwind*
* **Design: Logic Gates**
* MOS inverter static characteristics
* Resistive Load Inverter
* CMOS inverter
* **Design: CMOS Inverter design using Dsch and Microwind**
* Importance of Time Delay and Low power

Projects

* Synthesizable FIFO Model(Day-29)
* Behavioral DRAM Model(Day-30&31)

**Tools Used**:

Xilinx ISE Design Suite(For Digital IC Design)

Microwind(For Analog IC Layout Design)

Dsch( For Circuit Design)

**Models/Logics to be designed**:

1-bit Half Adder

1-bit Full Adder

4-bit Ripple carry Full adder

4:1 MUX using logic equation

4:1 MUX using Conditional Operators

4:1 MUX using Dataflow Operators

4-bit Full adder using Dataflow Operators

Ripple Counter

Unidirectional Shift Register

Bidirectional Shift Register

T Flip-flop

Edge triggered D FF

8:1 MUX with Case statement

ALU Design

Generated Ripple Adder

Traffic Signal Controller

Three Way Switch logic

Task definition using ANSI C style Argument Declaration

D-FF with Procedural Continuous Assignment

Instantiation of a CMOS switch

Switch level logic gates

Serial Adder Design using FSM(Finite State Machine)

MOD-8 counter using FSM

Arbiter ckt using FSM

Different types of sequence detector using FSM

Synthesizable FIFO model (Project-1)

Behavioral DRAM model (Project-2)

**Matlab Syllabus**

**Coverage**

|  |
| --- |
|  MATLAB Basics |
|  Research Activities using MATLAB |
|  Error Solving Techniques |
|  Coding techniques for fast execution |
|  Interfacing with different software languages |
|  2D and 3D plotting |
|  Graphical User Interface design |
|  File export and import with MATLAB workspace |
|  Use of power system, control system & communication system tool box |
|  Use of signal processing & soft computing toolbox |
|  Use of MATLAB tools for fast computation |
|  Creating MATLAB libraries |
|  Simulation model design using MATLAB-Simulink |
|  Interactive session for VLSI domain students |

**Syllabus in Details**

Introduction

* Why MATLAB?
* History
* Its strengths
* Weaknesses
* Competitors
* Starting MATLAB, Using MATLAB as a calculator, Quitting MATLAB

Basics

* Familiar with MATLAB windows
* Basic Operations
* MATLAB-Data types
* Rules about variable names
* Predefined variables

Programming-I

* Vector
* Matrix
* Array Addressing
* Built-in functions
* Mathematical Operations
* Dealing with strings(Array of characters)
* Array of array(cell) concept

Programming-II

* Script file
* Input commands
* Output commands
* Structure of function file
* Inline functions
* Feval command
* Comparison between script file and function file

Conditional statements and Loop

* Relational and Logical Operators
* If-else statements
* Switch-case statements
* For loop
* While loop
* Special commands( Break and continue)
* Import data from large database
* Export data to own file or database

2D Plotting

* In-built functions for plotting
* Multiple plotting with special graphics
* Curve fitting
* Interpolation
* Basic fitting interface

3D Plotting

* Use of meshgrid function
* Mesh plot
* Surface plot
* Plots with special graphics

GUI

* Creating menu window for providing input
* Creating graphical user interface table
* Modifying table content
* Creating a database

Simulink

* Model design
* Simulation
* Know your MATLAB tool (Branch specific)
* Interactive session for VLSI domain students
* Create application specific IC for Xillinx

Project

++++++++++++++++++++++++++++++++

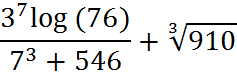
Software used: MATLAB 2011/13/14

++++++++++++++++++++++++++++++++

Sample Programs :

1. Perform simple mathematical operations
2. Calculate





1. Create specific matrix
2. Access part of matrix
3. Create array of characters
4. Addressing part of array
5. Creating array of arrays
6. WAP to generate random no.s b/w -3~3 find average, maximum, minimum, sort in ascending order.
7. WA script file to prompt user to enter 3 no.s and find the average.
8. WAP to find voltage and currents if resistances are defined in command window.
9. Electric potential of two point charges

At a point due to two particles

V=(1/4\*pi\*eps0)(q1/r1+q2/r2)

q1=2\*10^-10

q2=3\*10^-10

X1,y1=0.25,0,0

X2,y2=-0.25,0,0

Calculate potential due to two particles at points in xy-plane that are located in the domain -0.2<=x<=0.2 and 0.2<=y<=0.2

Make the plot such that the xy-plane is the plane of points and z axis is the magnitude of electric potential.

1. Heat conduction in a square plate

Three sides of a rectangular plate(a=5m,b=4m) are kept at a temp of T1=80 degreeC.

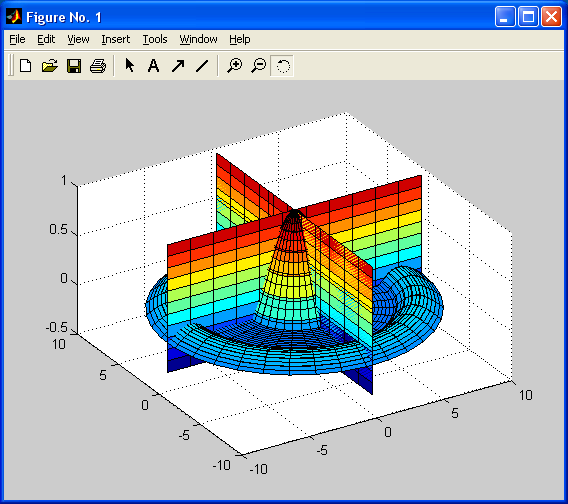
Determine and plot the temperature distribution T(x,y) in the plate.

Consider 5 terms.(Derived from Kreyszig-Mathematics)

1. Plot below figure



1. Plot below figure



1. Create a database which can be accessed by admin and user where admin can modify the database where as user can just get the information about a specific user.
2. Write a user defined function( name it FtoC) that converts temperature in degree F to degree C. Use the function to solve the following problem.

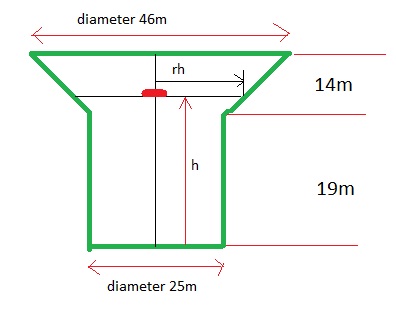
The change in the length of an object ,deltaL, due to a change in the temperature, deltaT, is given by:deltaL=alpha\*L\*deltaT

where alpha=co-efficient of thermal expanssion

Determine the change in the area of a rectangular (4.5m by 2.25m) aluminum (alpha=23\*10^-6 1/C) plate if the temperature changes from 40F to 92F.

1. The tank in a water tower has the geometry as

shown. Inside the tank there is a float that indicates the level of the water. Write a script that determines the volume of the water in the tank from the position(height h) of the float. The input is the value of h in m, and the o/p is the volume of water in m^3.



Rh=12.5+(10.5/14)(h-19)

1. Write a script file that converts quantity of energy(work) given in units of either joule,ft-lb, cal or eV to the equivalent quantity in different units specified by the user. The program asks the user to enter quantity of energy, its current units and the new desired units. The o/p is the quantity of energy in new units.

The conversion factors are:1J=0.738ft-lb=0.239cal=6.24\*10^18eV

1. Create a vector with 8 elements and calculate y using

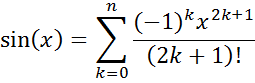


1. The following were 16 days daily maximum temp(in degree far) in Bhubaneswar during month of May 2015: 45 42 38 34 32 41 45 40 31 33 35 44 43 37 46 36. Use relational and logical operators to determine the following:
2. The number of days the temperature was above 40
3. The no. of days the temp was b/w 38 and 44.
4. The days of the month that the temperature was b/w 32 and 38
5. Use a for-end loop in a script file to calculate the sum of first n terms of the series:



Execute the script file for n=4 and n=20;

1. The function sin(x) can be written as a Taylor series by:



Write a user defined function file that calculates sin(x) by using the Taylor’s series. For function name and arguments use y=Tsin(x,n). The i/p arguments are the angle x in degree and n the number of terms in series. Use the function to calculate sin(150deg) using 3 and 7 terms.