K L Deemed to be University
Department of Computer Science & Engineering
Course Handout for Y20 Batch
A.Y.2020-21, Even Semester

Course Title	:	Object Oriented Programming
Course Code	:	19CS1203
L-T-P-S Structure	:	3-0-2-3
Credits	:	4.75
Pre-requisite	:	Basic Programming
Course Coordinator	:	Mr. G. Rama Krishna Srinivas
Course Instructors	:	Dr. D. Haritha, HOD-BES-1 Dr. Sk. Razia Dr. P. Siva Kumar Dr. S. Siva Kumar Mr. N. Sreeram Dr. E. Sreedevi Dr. N. V. S Pavan Kumar Mr. CMAK Zeelan Basha Mr. V. Uday Kumar Mr. D. Anand Dr. T. Rajesh Kumar Ms. V. Premalatha Mr. A. Krishna Mr. S. Pradeep Raj Mr. A S A L G Gopala Gupta Mr. E. Rajesh Kumar Mrs. Syed Karimunissa Mrs. V. Lakshmi Sarvani Mr. T. Ganesan Mrs. U. Harita Mr. B. Ashok Mr. M. Ram Kumar Mr. A. Srinivasa Rao Mrs. P. Gayatri Mrs. S. Harika Mr. Y. Ayyappa Mr. S. Siva Kumar Dr. Naveen Kumar Dr. M. Anusha Mrs. M. Anila Mr. G. Kalyan Chakravarthi Mrs. T. Yamini Mr. P. Neelakanteswara Rao Mrs. Shipa Itnal Mr. J. Nagaraju Mr. Ch. Naresh

Course Objective: The objective of the course is to equip the student with problem solving skills using Object Oriented Programming language – Java and details about the essential ingredients of the programming language and its fundamentals with a rich set of examples.

Course Rationale: The course takes an imperative view of problem-solving using Java programming language. This necessitates a firm foundation on the principles of Object-Oriented Programming (OOP). Student is professionally trained in OOP principles. The students are made to write Java programs on their own for sets of both mathematical and other engineering problems after exposing them to the different constructs of Java language namely abstract classes, Interfaces, packages, and multithreading. Finally, the student is acquainted with basic knowledge of the collection framework.

Course Outcomes:

CO#	**CO Description**	**BTL**	**PO/PSO**
CO1	Understand basic Concepts of OOP, fundamentals of java and apply the concepts of classes and objects through Java Language, Access control, Overloading.	3	PO3, PO5, PSO2
CO2	Apply constructors, parameter passing, String, StringBuffer and StringTokenizer.	3	PO3, PO5, PSO2
CO3	Inheritance, Packages, Exception Handling.	3	PO3, PO5, PSO2
CO4	Multithreading, Apply collection framework and event driven programming.	3	PO3, PO5, PSO2
CO5	Apply object-oriented programming concepts to write programs and Analyses requirements and design to implement lab-based project with SDLC in a group of students.	4	PO7, PO9, PO10, PSO1

Course Outcome Indicators (COIs):

CO No.	**Highest BTL**	**COI-1 (BTL-1,2)**	**COI-2 (BTL-3)**	**COI-4 (BTL-3)**
CO-1	3	Understanding the basic OOP concepts	Apply Access control and overloading	
CO-2	3	Understand String and related classes and its methods	Apply constructors, parameter passing	
CO-3	3	Understand need of inheritance, super and protected access	Apply inheritance, Exception handling	
CO-4	3	Understanding threads and lifecycle, event driven programming	Apply Multithreading and collection framework	
CO-5	4	Develop projects in java involving OOP concepts		

Program Objectives & Program Specific Objectives (POs/PSOs)

Program Objectives

PO1	An ability to apply knowledge of mathematics, science, and engineering.
PO2	An ability to identify, formulate, and solve engineering problems.
PO3	An ability to design solutions for complex engineering problems and system component or processes that meet the specified needs considering public health & safety and cultural, societal & environment.
PO4	An ability to use research-based knowledge and research methods including design of experiments, analysis and interpretation of data and synthesis of the information to obtain solutions to Mechanical engineering problems.
PO5	Ability to create, select and apply appropriate techniques, resources, and modern engineering activities, with an understanding of the limitations.
PO6	Ability to apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues, and the consequent responsibilities relevant to the professional engineering practice.
PO7	Ability to demonstrate the knowledge of engineering solutions, contemporary issues understanding their impacts on societal and environmental contexts, leading towards sustainable development.
PO8	An ability to apply ethical principles and commit to professional ethics and responsibilities and norms of engineering practice.
PO9	An ability to function effectively as an individual, and as a member or leader in diverse teams and in multi-disciplinary settings.
PO10	Ability to communicate effectively oral, written reports and graphical forms on complex engineering activities.

Program Specific Objectives

PSO1 An ability to design and develop software projects, as well as to analyse and test user requirements.

PSO2 Working knowledge on emerging software tools and technologies.

Syllabus

Introduction: Object-Oriented Programming, OOP Principles, Encapsulation, Inheritance and Polymorphism Java as a OOPs & Internet Enabled language, The Byte code, Data types, Variables, Dynamic initialization, scope and lifetime of variables, Arrays, Operators, Control statements, Type Conversion and Casting, Compiling, and running of simple Java program.

Classes and Objects: Concepts of classes and objects, declaring objects, Assigning Object Reference Variables, Methods, Constructors, Access Control, Garbage Collection, Usage of static with data and methods, usage of final with data, Overloading methods and constructors, parameter passing - call by value, recursion, Nested classes.

Inheritance: Inheritance Basics, member access rules, Usage of super key word, forms of inheritance, Method Overriding, Abstract classes, Dynamic method dispatch, Using final with inheritance, String handling functions.

Packages and Interfaces: Packages, Class path, importing packages, differences between classes and interfaces, Implementing & Applying interface.

Exception Handling: Exception Handling fundamentals, Collections Framework.

Textbooks

1. Herbert Schildt, "The Complete Reference Java", 7th edition TMH.
2. Timothy A. Budd, "An Introduction to Object-Oriented Programming", 3/e, Pearson, 2008.

Reference books

1. Deitel&Deitel, "'Java – How to program", 6th edition, PHI, 2007.
2. Cay.S.Horstmann and Gary Cornell "Core Java 2, Vol 1, Fundamentals", Seventh Edition, Pearson Education.

Tool(s) & Software: Eclipse, NetBeans

Course Delivery Plan:

Session. No.	CO	COI	Topic (s)	Book No [CH No] [Page No]	Teaching-Learning Methods	Evaluation Components
1	1	1	Introducing the course handout. Introducing Object oriented programming	T BOOK [1][2-78]	Lecturing, Discussion	SIE-1/SEE
2		1	Introduction to Java-Writing a class, Static block, variables, and methods	T BOOK [1], CH 7, Page no 141-142	Lecturing, Discussion	SIE-1/SEE
3		1	Modularization and access specifiers	T BOOK [1], CH-9, Page no:186-187	Lecturing, Discussion	SIE-1/SEE
4		1	Modularization – class and package level	T BOOK [1], CH-9, Page no:186-187	Lecturing, Discussion	SIE-1/SEE
5		1	Creating objects and accessing members through object	T BOOK [1], CH-6, Page no:105-112	Lecturing, Discussion	SIE-1/SEE
6		2	Accessor and Mutator methods and toString() method	T BOOK [1], CH-7, Page no:150-155T BOOK [1], CH-18, Page no:540-548	Lecturing, Discussion	SIE-1/SEE
7		2	Command line arguments, Console Input through Scanner class	T BOOK [3], CH-3, Page no:111-128	Lecturing, Discussion	SIE-1/SEE
8		2	Handling I/O – Files	T BOOK [3], CH-6, Page no:267-268	Lecturing, Discussion	SIE-1/SEE
9		2	Method overloading and array of references	T BOOK [1], CH-13,CH-29,CH-30.	Lecturing, Discussion	SIE-1/SEE
10	2	1	Constructors- types and this keyword	T BOOK [1], CH-7, Page no:119-121.	Lecturing, Discussion	SIE-1/SEE
11		1	Constructor overloading and Constructor chaining	T BOOK [1], CH-7, Page	Lecturing, Discussion	SIE-1/SEE

				no:125-129.		
12		1	Object as argument and return value. Call by value vs Call by reference	T BOOK [1], CH-7, Page no:130-134	Lecturing, Discussion	SIE-1/SEE
13		1	Nested and inner classes	T BOOK [1], CH-7, Page no:145-147	Lecturing, Discussion	SIE-1/SEE
14		2	String Class and methods	T BOOK [1], CH-15.	Lecturing, Discussion	SIE-1/SEE
15		2	StringBuffer Class and StringTokenizer Class	T BOOK [1], CH-15.	Lecturing, Discussion	SIE-1/SEE
16		2	References as members (Aggregation)	T BOOK [3], CH-7, Page no:308-313	Lecturing, Discussion	SIE-1/SEE
17		2	Menu driven programs for applications	T BOOK [3], CH-7.	Lecturing, Discussion	SIE-1/SEE
18		2	Menu driven programs for applications			
19	*3*	1	Inheritance – Types	T.BOOK[1],CH-8 ,Page no.157-167	Lecturing, Discussion	SIE-2/SEE
20		1	Member access – protected keyword	T.BOOK[1],CH-8 ,Page no.157-167	Lecturing, Discussion	SIE-2/SEE
21		2	Method overriding and dynamic method dispatch	T.BOOK[1],CH-8 ,Page no.171-176	Lecturing, Discussion	SIE-2/SEE
22		2	Super class variable referring sub class object	T.BOOK[1],CH-8 ,Page no.157-168	Lecturing, Discussion	SIE-2/SEE
23		2	final keyword – with methods and classes, Object class usage of super	T.BOOK[1],CH-8 ,Page no.180-181	Lecturing, Discussion	SIE-2/SEE
24		2	Abstract classes - 1	T.BOOK[1],CH-8 ,Page no.177-179	Lecturing, Discussion	SIE-2/SEE
25		2	Extending abstract classes	T.BOOK[1],CH-8 ,Page no.177-179	Lecturing, Discussion	SIE-2/SEE
26		2	Exceptional Handling – Usage of try and catch, nested try, multiple catch blocks.	T.BOOK[1],CH-9 ,Page no.183-197	Lecturing, Discussion	SIE-2/SEE

27		2	Exceptional Handling - throw, throws, and finally blocks	T.BOOK[1],CH-9	Lecturing, Discussion	SIE-2/SEE
28		1	Exceptional Handling – User defined Exception	T.BOOK[1],CH-9 ,Page no.192-202	Lecturing, Discussion	SIE-2/SEE
29	4	1	Interfaces and implementing interfaces	T.BOOK[1],CH-9 ,Page no.192-202	Lecturing, Discussion	SIE-2/SEE
30		1	Extending interfaces and multiple inheritance	T.BOOK[1],CH-10,Page no.209-211	Lecturing, Discussion	SIE-2/SEE
31		2	Event driven Programming – UI components and Layouts	T.BOOK[1],CH-10,Page no.213-222	Lecturing, Discussion	SIE-2/SEE
32		2	Event driven Programming			
33		2	Multithreading - 1	T.BOOK[1],CH-11,Page no.222-254	Lecturing, Discussion	SIE-2/SEE
34		2	Multithreading - 2	T.BOOK[1],CH-11,Page no.222-254	Lecturing, Discussion	SIE-2/SEE
35		2	Introduction to Collection Framework – List, Set	T.BOOK[1],CH-17,Page no.448-452	Lecturing, Discussion	SIE-2/SEE
36		2	Introduction to Collection Framework – Map	T.BOOK[1],CH-17,Page no.464-472	Lecturing, Discussion	SIE-2/SEE

Session Wise Teaching/Learning Plan:

Session: 01

Session Outcome: At the end of this session, Students will be able to understand,

1. Paradigm shift from Procedural oriented towards Object Oriented Programming.
2. Object Oriented Principles

Time (Min)	Topic	BTL	Teaching/Learning Methodology	Active Learning Methods
15	Course handout: Regarding Syllabus, Textbooks, References, MOOC's, Evaluation Pattern, Division of course Competencies.			
5	Introduction/Recap of Procedural Oriented Programming	**2**	Lecturing, Discussion	
20	Introduction to Object oriented Paradigm-Principles, Bytecode	**2**	Lecturing, Discussion	
10	Differences between Procedure oriented & Object-Oriented Programming.	**2**	Lecturing, Discussion	

SESSION NUMBER: **02**

Session Outcome: At the end of this session, Students will be able to understand,

1. The use of static block, variables and methods and learn to access them.

Time (Min)	Topic	BTL	Teaching/Learning Methodology	Active Learning Methods
5	Recap/Attendance			
25	Naming conventions for Class names, methods, and data members. Drawing class Diagram. Static variables and static methods and static block. Develop code for finding the factorial of a number.	**2**	Lecturing	
	Predict the output of the following: `class JavaExample{` `  static int num;` `  static String mystr;` `  static{` `  num = 97;` `  mystr = "Static keyword in Java";` `  }` `  public static void main(String args[])` `  {` `  System.out.println("Value of num: "+num);` `  System.out.println("Value of mystr: "+mystr);` `  }` `}` Multiple static blocks `class JavaExample2{` `  static int num;` `  static String mystr;` `  //First Static block` `  static{`			

15	`System.out.println("Static Block 1");` `num = 68;` `mystr = "Block1";` `}` `//Second static block` `static{` `System.out.println("Static Block 2");` `num = 98;` `mystr = "Block2";` `}` `public static void main(String args[])` `{` `System.out.println("Value of num: "+num);` `System.out.println("Value of mystr: "+mystr);` `}` `}` **Reference:** https://beginnersbook.com/2013/04/java-static-class-block-methods-variables/	**2**	Discussion	
5	Conclusion and Summary			

SESSION NUMBER: **03**

Session Outcome: At the end of this session, Students will be able to

1. Apply modularization – Method and class level Modularization.
2. Understand the usage of access specifiers – public, private and default.

Time (Min)	**Topic**	**BTL**	**Teaching/Learning Methodology**	**Active Learning Methods**
5	Recap/Introduction			
10	Write a Cuboid class with 3 static variables length, breadth and height of type double, and a static method volume (), access them using main () method within the same class.	**3**	Lecture/Discussion	
10	Explain the need of a class as a template (Encapsulate data and methods) Syntax – Define a class	**2**	Lecture/Discussion	
10	Modularize the above Cuboid class. Write a Cuboid class with 3 static variables length, breadth and height of type double, and a static method volume (), access them using main () method within another class Demo.	**3**		Group Discussion
5	Explain the usage of access modifiers – private and public	**2**	Lecture/Discussion	
10	Rewrite the Cuboid class with appropriate access specifiers	**3**	Lecturing, Discussion	

SESSION NUMBER: **04**

Session Outcome: At the end of this session, Students will be able to

1. Apply modularization – Class and package level.
2. Understand the usage of access specifiers – public, private and default.

Time (Min)	Topic	BTL	Teaching/ Learning Methodology	Active Learning Methods
5	Recap/Introduction			
20	Modularize the Cuboid class to a package level with appropriate access specifiers	**3**	Lecturing, Discussion	
10	To the above modularized code, add a method isCube () that returns true if all dimensions are same, else returns false.	**3**	Lecturing, Discussion	
10	Predict the Output of the following: `class Access` `{` `public static int x;` `private static int y;` `public static void cal(int a,int b)` `{` `x = a + 1;` `y = b;` `}` `}` `public class Access_Specifier` `{` `public static void main(String args[])` `{` `Access.cal(2, 3);` `System.out.println(Access.x + " " + Access.y);` `}` `}`	**3**		Group Discussion
5	Conclusion & summary			

Homework:

1. Create a class Student with id, marks for 3 subjects, total as static variables, static method to compute the total, main method to print the Student information. Modularize the code to class and package levels.

SESSION NUMBER: **05**

Session Outcome: At the end of this session,

1. Students will understand the reference variables, creation of objects using new.
2. The students will know how to access the instance members using objects.

Time (Min)	Topic	BTL	Teaching/ Learning Methodology	Active Learning Methods
5	Recap/Introduction			
15	Create a Cuboid class with 3 instance variables length, breadth and height of type double, and a method volume (). Create 2 objects with different values and print the volume.	**3**	Lecturing, Discussion	
15	Discuss - Assigning Object reference variables.	**3**		

	Predict the output of the following code snippets assuming the above definition of Box class. 1. class BoxDemo2 { public static void main(String args[]) { Box mybox1 = new Box(); Box mybox2 = mybox1; mybox1.width=10; System.out.println(mybox2.width); } } 2. class BoxDemo2 { public static void main(String args[]) { Box mybox1 = new Box(); mybox1.width=10; mybox1.height=20; Box mybox2 = mybox1; System.out.println(mybox2.width); mybox2=null; System.out.println(mybox2.width); System.out.println(mybox1.width); } }			Think/Pair/Share
10	Differentiate between instance members and static members and rules of accessing	**3**	Lecture	
5	Conclusion & summary			

SESSION NUMBER: **06**

Session Outcome:

1. Students will understand the usage of Accessor and mutator methods.
2. Students will understand and use the toString() method.

Time (Min)	**Topic**	**BTL**	**Teaching/ Learning Methodology**	**Active Learning Methods**
5	Recap/Introduction			
10	Need for accessors and mutators. Create a Cuboid class with 3 private instance variables length, breadth and height of type double, and a public method volume of return type double (). Add 3 setter methods with return type boolean (the instance variable is set when the argument is +ve) for each of the 3 instance members	**3**	Lecture	
10	Also add 3 getter methods of return type, which return the value appended with m (meters).	**3**	Lecture	
5	Use toString() method to print the details of Cuboid as Length : 10.0 m Breadth : 10.0 m Height: 10.0 m Volume : 100.0 cu.m	**3**	Lecture	

15	Access each of these methods through an object of Cuboid from the main() method of Demo class. Set the instance variables by obtaining input from console. (Use Scanner)	**3**	Lecture	
5	Conclusion & Summary			

Homework:

1. Write a Java Program to create Student class with ID, name, gender and branch. Use getter and setters. The ID must be 9-digit number, name must not have special characters and digits, gender must be either M/F and branch must be either ECE/CSE/ME/ECSE/CE/BT/EEE. Use toString() to format the details of Student.
2. Read data from console and create 2 student objects and print the data of each student.

SESSION NUMBER: **07**

Session Outcome: At the end of this session,

1. Students will understand how to obtain input through command line arguments and Scanner class.
2. Students will understand and use the Wrapper classes and methods in them.

Time (Min)	**Topic**	**BTL**	**Teaching/ Learning Methodology**	**Active Learning Methods**
5	Recap/Introduction			
10	Write a Java Program to read your name through command line arguments.	3	Lecture	
10	Explain about Wrapper classes – Byte, Short, Integer, Float, Double, Boolean, Character. Ask students to explore the Java API and Math class in java.lang package https://docs.oracle.com/javase/7/docs/api/	2	Lecture	
10	Introduce the Scanner Class in Java with the following example. import java.util.Scanner; class Input { public static void main(String[] args) Scanner input = new Scanner(System.in); // Getting float input System.out.print("Enter float: "); float myFloat = input.nextFloat(); System.out.println("Float entered = " + myFloat); // Getting double input System.out.print("Enter double: "); double myDouble = input.nextDouble(); System.out.println("Double entered = " + myDouble); // Getting String input System.out.print("Enter text: "); String myString = input.next(); System.out.println("Text entered = " + myString); } }	**3**	Lecture	

10	Write a Java Program to read Student ID, name, marks of 3 subjects through Scanner, and display the details along with total and percentage obtained.	**3**	Lecture	
5	Conclusion & Summary			
Homework: 1. Create a Cuboid class with 3 public instance variables length, breadth and height of type double, and a method volume (). Create 2 objects with different values obtained by command line arguments and print the volume of each. (The program must take 6 values as input) 2. Redo the above program using Scanner.				

SESSION NUMBER: **08**

Session Outcome: At the end of this session,

1. Students will obtain input from files.

Time (Min)	**Topic**	**BTL**	**Teaching/ Learning Methodology**	**Active Learning Methods**
5	Recap/Introduction			
15	File Class Explain how to read from file and write to a file	3	Lecture	
25	Develop the main method of Demo class such that it reads length, breadth, and height of a cuboid from a file and outputs the volume details to another file	3	Lecture/Discussion	
5	Conclusion & Summary			
Homework: 1. Write a Java Program to read details of a student from file and copy them to the other.				

SESSION NUMBER: **09**

Session Outcome: At the end of this session,

1. Students will understand method overloading.
2. Students will apply the concept of array of references.

Time (Min)	**Topic**	**BTL**	**Teaching/ Learning Methodology**	**Active Learning Methods**
5	Recap/Introduction			
15	Explain the concept of method overloading and its advantages. Add setDimensions(double l, double b, double h) in the Cuboid class which set the instance variables when all the arguments are +ve. Overload this with setDimensions(double s). Both methods return Boolean type.	3	Lecture	

10	Access the methods through an object of Cuboid from main () method of Demo class.	3		Think/Pair/Share
15	Modify the Demo class to store the details of 10 cuboid references in an array and print them	**3**	Lecture	
5	Conclusion & Summary			
Homework: 1. Write a Java Program to store details of 10 students in an array of Student references and print them.				

SESSION NUMBER: **10**

Session Outcome: At the end of this session,

1. Students will understand the need for constructors and types of constructors.
2. Students will use this keyword.

Time (Min)	**Topic**	**BTL**	**Teaching/ Learning Methodology**	**Active Learning Methods**
5	Recap/Introduction			
15	Define constructors, rules, and types. 1. Implicit vs Explicit 2. No-argument, parameterized constructor, copy constructors	2	Lecture	
20	Define the no-argument, parameterized constructor and copy constructors for the Cuboid class. The no-argument constructor will set the instance variables to 0.	3	Lecture, Discussion	
5	Explain garbage collection	2	Lecture	
5	Conclusion & Summary			

SESSION NUMBER: **11**

Session Outcome: At the end of this session,

1. Students will understand the need for constructor overloading and constructor chaining and apply them.

Time (Min)	**Topic**	**BTL**	**Teaching/ Learning Methodology**	**Active Learning Methods**
5	Recap/Introduction			
15	Overload the Cuboid class with all 3 types of constructors and create 3 objects each invoking a different type of constructor from the main method of Demo class.	3	Lecture	
15	Enhance the above code by chaining the constructors. Illustrate the importance of constructor chaining	3	Lecture	
10	Predict the output of the following. `class Temp` `{` `  Temp()` `  {`	3		

	<pre>this(5); System.out.println("The Default constructor"); } Temp(int x) { this(5, 15); System.out.println(x); } Temp(int x, int y) { System.out.println(x * y); } public static void main(String args[]) { new Temp(); } }</pre> **Reference:** https://geeksforgeeks.com <pre>class Test { Test (int w) { System.out.println(w); } static Test () { System.out.println(10); } public static void main (String args[]) { Test t=new Test(50); } }</pre> Can a constructor be static?			Group Discussion
5	Conclusion & Summary			

SESSION NUMBER: **12**

Session Outcomes: At the end of this session, Students will understand,

1. How to pass an object to method and to return objects from methods.
2. Can differentiate between call by value and call by reference.

Time (Min)	Topic	BTL	Teaching/Learning Methodology	Active Learning Methods
5	Recap			
15	Passing an object as an argument to a method Create a class Test with equals () method which compares two objects for equality and returns the result i.e., either true or false (Note: equals () methods should take an object as an argument)	**3**	Lecturing, Discussion	
15	Returning object from a method	**3**	Lecturing, Discussion	

	Create a class Test with incrByTen () method which returns an object after incrementing the value by 10 than the value in the invoking object.			
10	Call by value vs Call by reference	**3**	Lecturing, Discussion	
5	Conclusion and Summary			

SESSION NUMBER: **13**
Session Outcomes:

1. To understand and write nested and inner classes.

Time (Min)	**Topic**	**BTL**	**Teaching/Learning Methodology**	**Active Learning Methods**
5	Recap/Introduction**:**			
20	Explain about nested and inner classes with example.	**2**	Lecturing	
10	Predict the output of the following: static inner class `class Outer` `{` `  static int temp1 = 1;` `  static int temp2 = 2;` `  int temp3 = 3;` `  int temp4 = 4;` `  public static class Inner` `  {` `  private static int temp5 = 5;` `  private static int getSum()` `  {` `  return (temp1 + temp2 + temp3 + temp4 + temp5);` `  }` `  }` `  public static void main(String[] args)` `  {` `  Outer.Inner obj = new Outer.Inner();` `  System.out.println(obj.getSum());` `  }` `}` **Reference:** https://geeksforgeeks.com			Brainstorming
10	Predict the output of the following: inner class inside a method `public class Outer` `{` `  static int data = 10;` `  static int LocalClass()` `  {` `  class Inner` `  {` `  int data = 20;` `  int getData()`			

	`` { return data; } }; Inner inner = new Inner(); return inner.getData(); } public static void main(String[] args) { System.out.println(data * LocalClass()); } } ``` **Reference:** https://geeksforgeeks.com			Brainstorming
5	Conclusion and Summary			

SESSION NUMBER: **14**

Session Outcomes

1. To understand Strings and use the methods in String class.

Time (Min)	Topic	BTL	Teaching/Learning Methodology	Active Learning Methods
5	Recap			
10	String class constructors – character array as argument String class methods for Character Extraction– charAt(), getChars(), toCharArray() and length()	**2**		Think/Pair/Share
20	String Comparison methods- equals(), equalsIgnoreCase(), regionMatches(), startsWith() and endsWith(), compareTo() Difference between == and equals() method to compare Strings	**2**		Think/Pair/Share
10	Write java program to sort a set of words specified as command line arguments	**3**		Brainstorm
5	Conclusion and Summary			
Homework: 1. Write Java programs to illustrate the usage of the following String Class methods Searching Strings – indexOf(), lastIndexOf(), isEmpty() Modifying Strings – substring(), concat(), replace(), trim(), toUpperCase(), toLowerCase()				

SESSION NUMBER: **15**

Session Outcomes: At the end of this session,

1. Students will be able to understand difference between String and StringBuffer
2. Use StringTokenizer class

Time (Min)	Topic	BTL	Teaching/Learning Methodology	Active Learning Methods
5	Recap			

15	Methods in StringBuffer – length() and capacity() charAt() and setCharAt(), append(), insert(), reverse(), delete(), deleteCharAt(), replace(), substring() and other methods	**2**		15 min paper
10	Difference between String and StringBuffer	**3**		Discussion
15	Usage of StringTokenizer Class and its methods Predict the output of the following. `import java.util.StringTokenizer;` `public class App {` `public static void main(String[] args) {` `String str = "This is String , Now, created by me";` `StringTokenizer st = new StringTokenizer(str);` `System.out.println("---- Split by space ----");` `while (st.hasMoreElements()) {` `System.out.println(st.nextElement());` `}` `System.out.println("---- Split by comma ',' ------");` `StringTokenizer st2 = new StringTokenizer(str, ",");` `while (st2.hasMoreElements()){` `System.out.println(st2.nextElement());` `}` `}` `}`	**3**	Lecture	
5	Conclusion and Summary			

SESSION NUMBER: **16**

Session Outcomes: At the end of this session,

1. Students will be able use references as members.
2. Use Java.util.Date and java.util.Calendar classes.

Time (Min)	**Topic**	**BTL**	**Teaching/Learning Methodology**	**Active Learning Methods**
5	Recap			
20	Develop a student class with the following fields: ID, name, DOB (Use java.util.Date), gender. Use appropriate getters and setters, toString() method and constructors	**3**	Lecture	
10	Enhance the above code to store the marks of 6 subjects(array) as a private instance member and corresponding mutator and accessor, modify the toString() method	**3**	Lecture	

10	Usage of java.util.Date and java.util.Calendar classes	**3**	Discussion	
5	Conclusion & Summary			

SESSION NUMBER: **17**

Session Outcomes: At the end of this session, Students will be able,

1. To write menu Driven programs to perform operations.

Time (Min)	**Topic**	**BTL**	**Teaching/Learning Methodology**	**Active Learning Methods**
5	Recap			
40	Enhance the main () method of Demo class to display a menu of operations as follows: 1. Add new Student 2. Print details of all students 3. Search a student based on ID 4. Search based on name 5. Modify name based on ID 6. Exit Store details of 10 students and test the code.	**3**	Lecture	
5	Conclusion & Summary			

Homework

1. Enhance the design by taking input from file "Student.txt". The main () method reads the data from file and displays the menu with the following options:
   a) Print details of all students
   b) Search a student based on ID
   c) Search based on name
   d) Modify name based on ID
   e) Exit

SESSION NUMBER: **18**

Session Outcomes: At the end of this session, Students will be able,

1. To write menu Driven programs to perform operations.

Time (Min)	**Topic**	**BTL**	**Teaching/Learning Methodology**	**Active Learning Methods**
5	Recap			
40	Enhance the design of the Student class by including Address, which is another class that contains Street name, City, Country and Pincode as attributes. Enhance the main() method to include the following operations as well 7. Modify address based on ID 8. Count no. of students who live in a city	**3**	Lecture	
5	Conclusion & Summary			

Session: 19

Session Outcome: At the end of this session, Students will be able to understand and apply,

1. Inheritance and types of inheritance

Time (Min)	Topic	BTL	Teaching/Learning Methodology	Active Learning Methods
5	Recap/Introduction			
10	Need for Inheritance with an example and Syntax, Terminology	**3**	Lecture	
10	Types of Inheritance 1. Simple Inheritance Explain the above using GeometricShape Class with attributes borderColor (String), filled (Boolean type). This is inherited by Rectangle Class with length and width as attributes. Add mutators, accessors and toString() methods	**3**	Lecture	
10	2. Multilevel Inheritance Enhance the above design where Cuboid class with height field inherits the Rectangle class.	**3**	Lecture	
10	3. Hierarchy Inheritance Enhance the design of Simple Inheritance where Circle class with radius field also inherits from GeometricShape.	**3**	Lecture	
5	Conclusion & Summary			

Session: 20

Session Outcome: At the end of this session, Students will be able to

1. Understand member access protected keyword.
2. Apply super keyword.

Time (Min)	Topic	BTL	Teaching/Learning Methodology	Active Learning Methods
5	Recap/Introduction			
10	Explain protected keyword and usage	**2**	Lecture	
20	Enhance the GeometricShape design using protected keyword.	**3**		Think/Pair/ Share
10	Explain how to access super class fields and methods using super keyword	**2**	Lecture	
5	Conclusion & Summary			

Session: 21

Session Outcome: At the end of this session, Students will be able to

1. Override methods.
2. Distinguish between method overloading and overriding.

Time (Min)	Topic	BTL	Teaching/Learning Methodology	Active Learning Methods
5	Recap/Introduction			
5	Enhance the GeometricShape class with area() method that simply returns 0 with appropriate access specifiers.	**3**	Lecture	
15	Override the area() method in all the sub classes of GeometricShape	**3**		Think/Pair/ Share
10	Chain the constructors using super	**3**	Lecture	
10	Compare and contrast polymorphism forms – method overloading and method overriding	**3**	Lecture	

5	Conclusion & Summary			

Homework:

1. Write a Java Program that has a class "Room" which is inherited by classes" ClassRoom", "Lab" and "StaffRoom". Identify the properties and methods required to be maintained in each of the classes based on their unique and similar features. ClassRoom has some methods in it which need to be overridden by some of its subclasses based on their unique functionality. Use 'super' keyword to access variables, methods and constructors from subclasses and give the output accordingly**.**

Session: 22

Session Outcome: At the end of this session, Students will be able to

1. Understand how super class reference variable refers to subclass object and use this feature.

Time (Min)	**Topic**	**BTL**	**Teaching/Learning Methodology**	**Active Learning Methods**
5	Recap/Introduction			
40	Enhance the main () method of Demo class, define a reference of GeometricShape. Define a menu so that based on choice the sub class object is created and assigned to the super class reference	**3**		Think/Pair/ Share
5	Conclusion & Summary			

Session: 23

Session Outcome: At the end of this session, Students will be able to

1. Understand final keyword usage with class and instance members.
2. Object class

Time (Min)	**Topic**	**BTL**	**Teaching/Learning Methodology**	**Active Learning Methods**
5	Recap/Introduction			
10	Usage of final with static members and methods	**2**	Lecture	
20	Usage of final with instance members and methods and class with an example.	**2**	Lecture	
10	Usage of Object class	**2**	Lecture	
5	Conclusion & Summary			

Session: 24

Session Outcome: At the end of this session on **Abstract Method and Abstract Classes** Students will be able:

1. Understand the basic concepts of Abstract Method and Abstract Classes
2. Usage of Abstract Method and Abstract Classes

Time in Minutes	**Topic: Abstract Classes**	**BTL**	**Teaching Methodology**	**ALM**
05	**Recap / Introduction:**	1		
10	Explain the Definition and syntax of Abstract Method with example.	2	Chalk & talk	
20	Write an **abstract class** that contains **basic details** of employee namely **name** and **empid** and with a **concrete method** to display it. Include another **abstract method** signature to display confidential details.	3	Discussion	Brain Storming

	Extend the **abstract class** in another class **HR** with employee confidential details like **Salary** and **Performance** and display them in the implementation of the of the definition of abstract method. **Reference:**https://www.softwaretestinghelp.com/java/java-interfaces-abstract-classes/			
10	Explain the Definition, syntax, and applications of Abstract classes with example.	2	Lecturing and Discussion	
05	**Summary and Conclusion**			
	Home Assignments: 1) Write an class that has signature of methods (abstract methods) to accept two integers and a method to accept three integers and return their sum. Write another regular class extending the abstract class and implement/define the two methods. 1)https://beginnersbook.com/2014/01/abstract-method-with-examples-in-java/			

Session: 25

Session Outcome: At the end of this session on Abstract **Classes**, Students will be able:

1. Solve problems on Abstract Method and Abstract Classes

Time in Minutes	**Topic**	**BTL**	**Teaching Methodology**	**ALM**
05	Recap / Introduction			
30	Assume that a bank maintains two kinds of account for its customers, one called savings account and the other current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed. Create a class **Account** that stores customer name, account number and type of account. From this derive the classes **Curr-acct** and **Sav-acct** to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks: (a) Accept deposit from a customer and update the balance (b) Display the balance (c) Compute and deposit interest. (d) permit withdrawal and update the balance (e) Check for the minimum balance, impose penalty, if necessary and update the balance.	3	Discussion	Case study
10	Write a JAVA program for computing sum of two integers and floats using abstract classes.	3	Discussion	
05	Conclusion & Summary			
	Home Assignments:			

	1) Declare an abstract class, called Instrument, containing a field name and a method called play, that must be implemented by a sub-class. Define a sub-class called StringedInstrument that extends the Instrument class and adds an extra field called numberofStrings. Add two more classes that implement the functionality of a StringedInstrument, called ElectricGuitar and ElectricBassGuitar accordingly. The definition of these newly added classes. Create two different instances of an ElectricGuitar and an ElectricBassGuitar classes and we call their play methods as below. Finally, we create a new class called Execution that contains a single main method. Reference: https://examples.javacodegeeks.com/java-basics/java-abstract-class-example/ Reference: http://www.pskills.in/java/abstract-class.jsp			

Session: 26

Session Outcome: 1. To understand basics of exception handling in Java.
2. To implement Exception handling using try and catch blocks.

Time in Minutes	Topic	BT L	Teaching–Learning Method	Active Learning Methods
5	Recap / Introduction:		Lecturing, Discussion	
5	Explain Exception. Explain different types of Exceptions.	2		
15	Explain the usage of try catch bocks in exception handling with an example. And explain different ways to print exception messages in Java. Explain the concept of nested try blocks. Explain the concept of multiple catch blocks. Explain exception catching order. Explain catching Multiple Exceptions in a Single catch block.	3		
20	**a)** Modify the above code to handle the exception that may arise using try catch block? public class TryCatchExample1 { public static void main(String[] args) { int data=50/0; System.out.println("rest of the code"); } } b) Ask the students to write a programs to catch ArithmeticException, ArrayIndexOutOfBoundsException and NumberFormat Exception. Predict the output of the following programs. 3. public class Test	3		Surprise Test

```
{
        public static void main(String[] args)
        {
                try
                {
                        System.out.printf("1");
                        int sum = 9 / 0;
                        System.out.printf("2");
                }
                catch(ArithmeticException e)
                {
                        System.out.printf("3");
                }
                catch(Exception e)
                {
                        System.out.printf("4");
                }
        }
}
```

4.

```
class Excep6{
 public static void main(String args[]){
  try{
    try{
     System.out.println("going to divide");
     int b =39/0;
    }catch(ArithmeticException e){System.out.println(e);}

    try{
    int a[]=new int[5];
    a[5]=4;
    }catch(ArrayIndexOutOfBoundsException
e){System.out.println(e);}

    System.out.println("other statement);
  }catch(Exception e){System.out.println("handeled");}

  System.out.println("normal flow..");
 }
}
```

5.

```
public class MultipleCatchBlock4 {

    public static void main(String[] args) {

          try{
              String s=null;
              System.out.println(s.length());
             }
             catch(ArithmeticException e)
               {
                System.out.println("Arithmetic Exception occurs");
               }
             catch(ArrayIndexOutOfBoundsException e)
               {
                System.out.println("ArrayIndexOutOfBounds
Exception occurs");
               }
```

	catch(Exception e) { System.out.println("Parent Exception occurs"); } System.out.println("rest of the code"); } }			
5	Conclusion			

Session: 27

Session Outcome: 1. Understand the Throw and Throws, finally block

2. Apply the concept of Throw and Throws, finally block.

Time (min)	**Topic**	**BTL**	**Teaching – Learning Method**	**Active Learning Methods**
05	Recap the previous topic			
15	Explain throw, throws and finally.	3	Chalk and talk	
15	Create different cases of where java finally block can be used. 1.demonstrate the working of finally block when no exception occurs in try block. 2.working of finally block when an exception occurs in try block but is not handled in the catch block. 3.When exception occurs in try block and handled properly in catch block.	3	Chalk and talk	Group Discussion
10	1.Find the output of following code using Throw. public class Example1{ void checkAge(int age){ if(age<18) throw new ArithmeticException("Not Eligible for voting"); else System.out.println("Eligible for voting"); } public static void main(String args[]){ Example1 obj = new Example1(); obj.checkAge(13); System.out.println("End Of Program"); } }	3	talk	

	2.Find the output of following code using Throws. public class Example1{ int division(int a, int b) throws ArithmeticException{ int t = a/b; return t; } public static void main(String args[]){ Example1 obj = new Example1(); try{ System.out.println(obj.division(15,0)); } catch(ArithmeticException e){ System.out.println("You shouldn't divide number by zero"); } } }			
05	Conclusion			

Session: 28

Session Outcome: 1. Understand and create User-defined Exception.

Time (min)	**Topic**	**BTL**	**Teaching – Learning Method**	**Active Learning Methods**
05	Recap the previous topic			
15	Explain the need of creating user-defined exception	3	Chalk and talk	
25	Create a user-defined exception InvalidDimensionException which is thrown by setter methods of Cuboid class and handled in main()	3	Chalk and talk	Group Discussion
05	Conclusion			

Session: 29

Session Outcome: At the end of this session students will be able to learn about Interfaces.

Time (Min)	**Topic**	**BTL**	**Teaching/Learning Methodology**	**Active Learning Methods**
5	Recap/Introduction**:**	1		

20	Understand the concept of Differences between classes & interfaces	**3**	Lecturing, Discussion	
20	Demonstrate an example on interface	**3**	Lecturing, Discussion	
5	Conclusion & Summary			

Session: 30

Session Outcome: At the end of this session students will be able to learn about multiple inheritance using Interfaces.

Time (Min)	Topic	BTL	Teaching/Learning Methodology	Active Learning Methods
5	Recap/Introduction**:**	1		
20	Implementation of multiple inheritance using interfaces and Execution of the program on interfaces.	**2**	Lecturing, Discussion	
20	Extending the interfaces and examples on extending interfaces	**2**	Lecturing, Discussion	
5	Conclusion & Summary			

Session: 31

Session Outcome: At the end of this session students will be able to create a static frame using UI components.

Time (Min)	Topic	BTL	Teaching/Learning Methodology	Active Learning Methods
5	Recap/Introduction**:**	1		
10	Introduce to various UI components like TextBox, ComboBox, Button, RadioButton in javax.swing package	2	Lecturing, Discussion	
15	Develop a JFrame to a simple calculator	3	Discussion	
15	Discuss various Layouts	**2**	Lecturing	
5	Conclusion & Summary			

Session: 32

Session Outcome: At the end of this session students will be able to learn about event driven programming.

Time (Min)	Topic	BTL	Teaching/Learning Methodology	Active Learning Methods
5	Recap/Introduction**:**	1		
15	Explain Event Handling mechanism. Introduce to various events, Listeners and Handlers	**2**	Lecturing, Discussion	
25	To the calculator developed add the events and text the application developed.	**3**	Discussion	
5	Conclusion & Summary			

Session: 33

Session Outcome: Understanding concept and importance of multithreading in Java.

Time(min)	Topic	BTL	Teaching – Learning Method	Active Learning Methods
05	Recap the previous topic			
15	Introduction to multithreading concept.	2	Chalk and talk	
15	Life Cycle and States of Thread in Java	2	Chalk and talk	
10	Main Thread in Java	2	Chalk and Talk	
05	Conclusion			

Session - 34

Session Outcome: 1. Understand the Thread Class and Runnable Interface
2. Apply the concept of multithreading to create multiple threads.

Time(min)	Topic	BTL	Teaching – Learning Method	ALM
05	Recap the previous topic			
15	Create thread which extends Thread class and use of start() and run() methods.	3	Chalk and talk	

15	Create thread by implementing Runnable interface.	3	Chalk and talk	seminar
10	class Test implements Runnable { public void run() { System.out.println("Run"); } } class Myclass { public static void main(String[] args) { Thread t1 = new Thread(); t1.start(); Thread t2 = new Thread(new Test()); t2.start(); System.out.println("Main"); } }	3		
05	Conclusion			

Session Number: 35
Session Outcome: Apply the concept of **Java collection framework**--Types of Lists.

.**Time(min)**	**Topic**	**BTL**	**Teaching – Learning Method**	**Active Learning Methods**
5	Attendance and Recap			
10	Understand the concept and need of **Java collection**	2	Chalk and talk	
10	Difference between Array and collection	2	Interactive	**One minute paper**

20	Demo on LinkedList and its methods Demo on Array List & its methods	2	Chalk and talk	
05	Conclusion & Summary			

Session Number: 36
Session Outcome: Student will be able to understand and apply TreeSet, HashSet, LinkedHashSet Classes and HashMap classes.

Time(min)	**Topic**	**BTL**	**Teaching – Learning Method**	**Active Learning Methods**
5	Attendance and Recap			
15	Demo on Hash Set with example	2	Chalk and talk, PPT	Role play
15	Demo on how to use LinkedHashSet, TreeSet classes and HashMap classes with example	2	Chalk and talk, PPT	
10	**Problem 1:** Ask the student to write a program that splits an arrayList to half. Eg: Input : list = {1, 2, 3, 4, 5, 6} Output : first = {1, 2, 3}, second = {4, 5, 6} Input : list = {1, 2, 3, 4, 5} Output : first = {1, 2}, second = {3, 4, 5} Ask the student to write a program that checks whether array has all identical elements using Arrays.asList() and HashSet in Java. Eg: **Input:** arr[] = {2, 2, 2, 2, 2} **Output:** Yes The given array has all identical elements i.e. 2. **Input:** arr[] = {2, 3, 3, 3, 3, 2, 2} **Output:** No			
05	Conclusion & Summary			

Lab List

Session. No	Problems to be discussed
1	1. https://www.hackerrank.com/challenges/java-loops-i/problem 2.Develop code with a method in MyFirstClass a) factorial () that computes the factorial of a given number. Modularize the design to class and package level. Note: 1. Draw the class diagram and then implement. 2. Hard code the input
2	1. Define a class Student with the following attributes. a) Name b) ID c) gender d) department Instantiate two Student objects and print the details of the students in the following format. ID: 2000030001 Name: ABC Gender: M Department: CSE 2. Design a class named QuadraticEquation for a quadratic equation $ax^2 + bx + c = 0$. The class contains: a) Attributes a, b, and c that represent three coefficients. b) Three getter methods for a, b, and c. c) A method named getDiscriminant() that returns the discriminant, which is $b^2 - 4ac$. d) The methods named getRoot1() and getRoot2() for returning two roots of the equation. r1 and r2 = $\frac{-b \pm \sqrt{b^2-4ac}}{2a}$
3	1. Rework on Lab 1 (Factorial example) by considering input through a) command line argument b) console. 2. KLEF FED office has 2 files, CSE.txt and ECE.txt. Both files have 3 columns- ID, Name and Mobile number of CSE and ECE students, respectively. The task is to read data from both files and store in a file students.txt and print the data of students.txt
4	1. Design a class named Account that contains: • A private int data field named id for the account (default 0). • A private data field named balance for the account (default 0). • A no-arg constructor that creates a default account. • A constructor that creates an account with the specified id and initial balance. • The accessor and mutator methods for id, balance. • Mutators return Boolean (If all the fields must be +ve, return true, else false) • A method named withdraw that withdraws a specified amount from the account. • A method named deposit that deposits a specified amount to the account. • A method to print the details of Account

	2. Design a class named MyPoint to represent a point with x- and y-coordinates. The class contains the data fields x and y that represent the coordinates with getter methods. ■ A no-arg constructor that creates a point (0, 0). ■ A constructor that constructs a point with specified coordinates. ■ A method named distance that returns the distance from this point to another point with the specified x- and y-coordinates. ■ A method named distance that returns the distance from this point to a specified point of the MyPoint type. Draw the UML diagram for the class and then implement the class. Write a test program that creates the two points (0, 0) and (10, 30.5) and displays the distance between them.
5	1.Write a program that reads a paragraph of text from file. Print the analytics as per the below format: Line# Number of words Data Total Count of words: 2. Define a collection of ten Strings each containing an arbitrary string of the form "month/day/year"; for example,"10/29/99" or "12/05/01". Analyze each element in the collection and output the date represented in the form 29th October 1999.
6	1. A company wants to digitalize their manual records of the employee details (Employee ID, Employee name, Employee Department). If all the three fields are given, use the given details. If not, use the default values. (Use Constructor Overloading) Default values are: ID: 0 Name: # Department: #. Write mutators and accessors and method to print Employee data in the above format. Write a menu driven main program to perform the following operations: a) Create new Employee record b) Update name based on ID c) Print All Employees d) Print Department Specific employees given department 2. https://www.hackerrank.com/challenges/java-substring/problem
7	1.Develop a program that creates a generic Shape class with attributes fillColor, borderColor, fill (Boolean type) and border width. The classes Rectangle, Circle must inherit Shape. Rectangle has length and width as attributes and circle with radius as attribute. Also add the corresponding getters and setters in each of the classes. Use appropriate access specifiers. Instantiate the objects and test the code. 2. Enhance the above Q1 hierarchy such that Shape is the general class, inherited by TwoDShape and ThreeDShape class, and Rectangle, Circle inherits the TwoDShape, Cuboid extends the ThreeDshape.
8	1. Design a class named Triangle that extends GeometricObject. The class contains: ■ Three double data fields named side1, side2, and side3 with default values 1.0 to denote three sides of the triangle. ■ A no-arg constructor that creates a default triangle.

	■ A constructor that creates a triangle with the specified side1, side2, and side3. ■ The accessor methods for all three data fields. ■ A method named getArea() that returns the area of this triangle. ■ A method named getPerimeter() that returns the perimeter of this triangle. ■ A method that returns a string description for the triangle as follows: "Triangle: side1 = " + side1 + " side2 = " + side2 + " side3 = " + side3; Draw the UML diagrams for the classes Triangle and GeometricObject and implement the classes. Write a test program that prompts the user to enter three sides of the triangle, a color, and a Boolean value to indicate whether the triangle is filled. The program should create a Triangle object with these sides and set the color and filled properties using the input. The program should display the area, perimeter, color, and true or false to indicate whether it is filled or not. 2. Design a class named Person and its two subclasses named Student and Employee. Make Faculty and Staff subclasses of Employee. A person has a name, address, phone number, and email address. A student has cgpa as attribute. An employee has salary, and date hired as attributes. A faculty member has designation as attribute. A staff member has hoursWorked as attribute. Draw the UML diagram for the classes and implement them. Write a test program that creates a Person, Student, Employee, Faculty, and Staff, and invokes their accessors and a method to print the data.
9	1. Define an abstract base class Point that includes protected data members for the (x, y) position of a shape, a public method to move a shape, and a public abstract method show() to output a shape. Derive subclasses- line, circle, and rectangle. You can represent a line as two points, a circle as a center and a radius, and a rectangle as two points on diagonally opposite corners. Test the classes by selecting ten random objects of the derived classes, and then invoking the show() method for each. 2. https://www.hackerrank.com/challenges/java-abstract-class/problem
10	1. https://www.hackerrank.com/challenges/java-exception-handling-try-catch/problem 2. Write a program for developing an interactive application for a simple calculator using event-driven programming.
11	1. Election committee wants to check whether the voter is eligible to vote or not. The person can vote if his age is greater than 18. Help the Election committee by developing a code which arises exception if the voter age is less than 18 then print the exception and “VOTER IS NOT ELIGIBLE TO VOTE”, otherwise print “VOTER IS ELIGIBLE TO VOTE”. (Hint: Develop user-defined Exception) 2. Rohan was asked by his teacher to develop a program which takes in a dynamic user input and prints all the prime numbers till that number and then prints all the composite numbers till that number. Help him out by developing the program to do so. Use one thread each for both prime and composite numbers and print in the format: Prime – 2

	Composite – 4 ..
12	1.Create a collection of 10 elements where each element has (key:int, value: String) and apply the following operations. a) Search for an element of a given key. If exists, print the key – value pair, otherwise, print "Search key not found". b) Remove an element from the collection for a given key. 2. Develop a program to read the country names (separated by space) from a file and store in a Set and then perform the following menu driven operations. a) Search for a country name b) Sort based on country name c) add new country name d) count the number of country names

Skilling Lesson Plan:

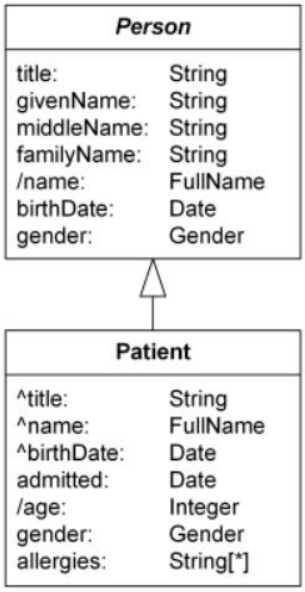
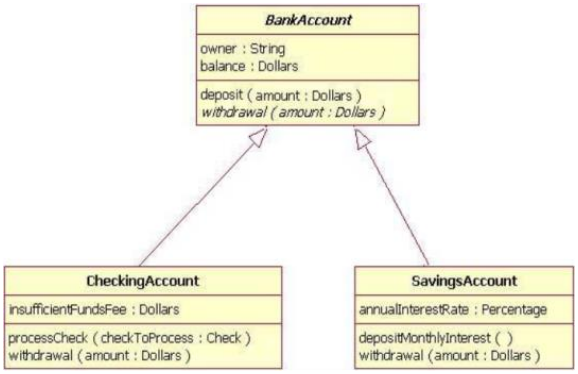
	Topics	**Skilling problems**
1	Introducing Object oriented programming	1. Write a Java Program to check if a given number is even or odd (Hardcode the value) 2. https://www.hackerrank.com/challenges/welcome-to-java/problem 3. Write a Java program to swap two numbers without using temporary variable (Hardcode the values) Explore the different ways in which the above task can be performed. 4. Write a Java program to perform the sum of digits of a number. (Hardcode the value).
2	Writing a class, variables, and methods	1. Write a class named Circle with radius as a static member and two static methods which will compute the area and the perimeter of a circle. Draw the class diagram. 2. Write a program to check if a given number is prime or non-prime. 3. Write a class named Rectangle with static members length, width and a method which will compute the area of a rectangle.

3	Modularization and access specifiers	1. Suppose you save $100 each month into a saving account the annual interest rate 5%. Thus, the monthly interest rate is 0.05/12=0.00417. After the first month the value in the account becomes100*(1+0.00417) = 100.417. After the second month the value in the account becomes (100+100.417) *(1+0.00417) = 201.252 and so on. Write a program that prompts the user to enter a monthly saving amount and display the amount value after 6 months. 2. Write a program that reads three edges for a triangle and computes the perimeter if the input is valid otherwise display the input is invalid. The input is valid if the sum of every pair of two edges greater than remaining edge. 3. Write a program to return TRUE if the year is leap year else return FALSE using static method. Modularize to method, class, and package level.
4	Modularization – class and package level	1. Modularize sum of n numbers using static variable, static method and static block to method, class, and package level. 2. Modularize the following problem to class level. Write code to display the speed of bike if speed is greater than or equal to 60 then indicate as over speed if speed is in between 40 to 60 then indicate average speed otherwise safe drive. 3. Modularize the following problem to package level. Write a code to take an integer n (hard code) and check whether n is divisible by both 3 and 7 or by neither of them nor by just one of them.
5	Creating objects and accessing members through object	1. Write a program will calculate the circumference, area of the circle. Instantiate 2 objects and use the getter and setter methods. 2. Develop the toString() method for the above Circle class and test it.
6	Accessor and Mutator methods	1.Create a class named Petrol Purchase to represent information about the petrol you purchase. The class should have five instance variables-the station's location (type String), the type of petrol (type String), the quantity (type int), the price per liter(double), the percentage discount(double). Provide accessors and mutator methods for instance variables. In addition, provide a method named getPurchase() that calculates the net purchase amount 2. Create a class named Date that includes three instance variables-a model (type String), a year (type String), and a price (double). Provide a set() and get() method for instance variables. Provide a method displayDate() that displays the month, day and year forwarded by slashes. 3. Create a class named Mytriangle contains three instance variables-side1 (type int), side2(type int), side3(type int). Write getter and setter methods for instance variables. Create two methods named isValid() and area() .First one checks the input is valid or not. If the input is not valid display as invalid. Second method computes area if the input is valid

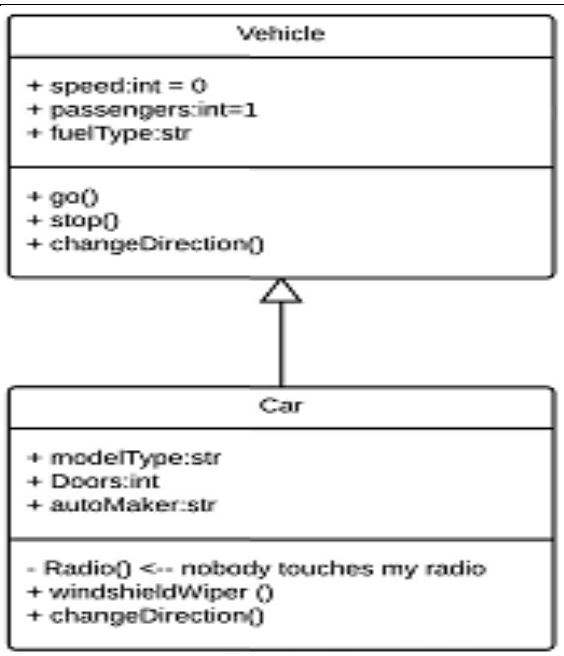
7	Command line arguments, Console Input through Scanner class	https://www.hackerrank.com/challenges/java-stdin-and-stdout-1/problem https://www.hackerrank.com/challenges/java-if-else/problem https://www.hackerrank.com/challenges/java-stdin-stdout/problem https://www.hackerrank.com/challenges/java-datatypes/problem https://www.hackerrank.com/challenges/java-output-formatting/problem
8	Method overloading Loops	1. Crate a class MethodOverload which includes two overload versions of method area()-one that calculates the area of circle by taking radius as an argument and another which computes the area of square. 2. https://www.hackerrank.com/challenges/java-output-formatting/problem 3. https://www.hackerrank.com/challenges/java-loops-i/problem 4. https://www.hackerrank.com/challenges/java-loops/problem 5.Create a class to print the area of a square and a rectangle. The class has two methods with the same name but different number of parameters. The method for printing area of rectangle has two parameters which are length and breadth respectively while the other method for printing area of square has one parameter which is side of square.
9	Online coding platform	1.Harry's File https://www.hackerearth.com/practice/basic-programming/implementation/basics-of-implementation/practice-problems/algorithm/find-the-name-if-you-can-61f601ad/ 2. Simplify Path. https://leetcode.com/problems/simplify-path/ 3.https://www.hackerrank.com/challenges/java-static-initializer-block/submissions/code/22103648 4. https://www.hackerrank.com/challenges/java-end-of-file/problem
10	Constructors- types	1. Write a program that creates an employee class with empid, name, salary as its members. Create constructors to give initial values to class members. 2. Develop a program to create a Class Vehicle with wheels and color as its member. Now create parameterized and no parameterized constructors that assign values to Vehicle class members with appropriate display messages.
11	Constructor overloading	1. Create a class named 'Rectangle' with two data members- length and breadth and a method to calculate the area which is 'length*breadth'. The class has three constructors which are: a - having no parameter - values of both length and breadth are assigned zero.

		b - having two numbers as parameters - the two numbers are assigned as length and breadth, respectively. c - having one number as parameter - both length and breadth are assigned that number. Now, create objects of the 'Rectangle' class having none, one and two parameters and print their areas. **2.** Suppose you have a Piggie Bank with an initial amount of $50 and you must add some more amount to it. Create a class 'AddAmount' with a data member named 'amount' with an initial value of $50. Now make two constructors of this class as follows: A- without any parameter - no amount will be added to the Piggie Bank B- having a parameter which is the amount that will be added to Piggie Bank Create object of the 'AddAmount' class and display the final amount in Piggie Bank.
12	Object as argument and return value. Call by value vs Call by reference	1. Create a class named 'Programming'. While creating an object of the class, if nothing is passed to it, then the message "I love programming languages" should be printed. If some String is passed to it, then in place of "programming languages" the name of that String variable should be printed. For example, while creating object if we pass "code", then "I love code" should be printed. 2. Create a class 'Student' with three data members which are name, age and address. The constructor of the class assigns default values name as "unknown", age as '0' and address as "not available". It has two members with the same name 'setInfo'. First method has two parameters for name and age and assigns the same whereas the second method takes has three parameters which are assigned to name, age and address, respectively. Print the name, age, and address of 10 students.
13	Online Coding Platform	1. Advantage Shuffle https://leetcode.com/problems/advantage-shuffle/ 2. Design Linked List https://leetcode.com/problems/design-linked-list/ https://www.hackerrank.com/challenges/java-1d-array-introduction/problem
14	String class & its methods	https://www.hackerrank.com/challenges/java-int-to-string/problem https://www.hackerrank.com/challenges/java-strings-introduction/problem https://www.hackerrank.com/challenges/java-string-compare/problem https://www.hackerrank.com/challenges/java-string-reverse/problem https://www.hackerrank.com/challenges/java-anagrams/problem
15	String Buffer and String Tokenizer.	1.Two Characters https://www.hackerrank.com/challenges/two-characters/problem 2. HackerRank in a String!

		https://www.hackerrank.com/challenges/hackerrank-in-a-string/problem 3. Letter Combinations of a Phone Number https://leetcode.com/problems/letter-combinations-of-a-phone-number/ 4. Find All Good Strings https://leetcode.com/problems/find-all-good-strings/
16	References as members Date & Time	https://www.w3resource.com/java-exercises/datetime/index.php
17	Menu driven programs for applications	1. Create a program using switch-case that would let the users place their order. after order is placed, menu would re-appear to let user place order again (if they want to) Also provide option to exit from the menu application to user. 2. Write a program to print series 0, 3, 8, 15, 24 and s=1/2+3/4+5/6+..+19/20 with help of Menu using switch case statement.
18	Inheritance – Types	1 Create Vehicle Interface with name, maxPassanger, and maxSpeed variables. Create Land Vehicle and Sea Vehicle Inteface from Vehicle interface. Land Vehicle has numWheels variable and drive method. Sea Vehicle has displacement variable and launch method. Create Car class from Land Vehicle, Hovercraft from Land Vehicle and Sea Vehicle interface. Also create Ship from Sea Vehicle. Provide additional methods in Hovercraft as enter Land and enter Sea. Similarly provide other methods for class Car and Ship. Demonstrate all classes in an application. 2 Method Resolution order https://www.codechef.com/problems/MRO
19	Member access – protected keyword	1.Write a program that has student class where name, id, mail is protected specifier in mypack1 package and create driver class which contains main()(ex "demopro" class) in mypack2 package which extend the student class ,try to access the member variables of student. Rerun the program by making default access specifier to the attributes in student class. 2. Write a program which creates two packages p1 and p2. Class A in p1 is made public, to access it in p2. The method display in class A is protected and class B is inherited from class A, then access the method display().
20	Method overriding and dynamic method dispatch	https://www.hackerrank.com/challenges/java-inheritance-1/problem https://www.hackerrank.com/challenges/java-inheritance-2/problem https://www.hackerrank.com/challenges/java-method-overriding/problem

		https://www.hackerrank.com/challenges/java-method-overriding-2-super-keyword/problem https://www.hackerrank.com/challenges/java-instanceof-keyword/problem
21	Inheritance	1. Write the necessary classes with given member data and appropriate methods by identifying the inherited properties from person class to patient class and display all the details of patient.  2.Write a program to implement the below design. 
22	Inheritance	1. Create a class named 'Shape' with a method to print "This is This is shape". Then create two other classes named 'Rectangle', 'Circle' inheriting the Shape class, both having a method to print "This is rectangular shape" and "This is circular shape" respectively. Create a subclass 'Square' of 'Rectangle' having a method to print "Square is a rectangle". Now call the method of 'Shape' and 'Rectangle' class by the object of 'Square' class.

		1. Write a program to calculate the average height of all the students of a class. The number of students and their heights in a class are entered by user. 2. Write a program to print the name, salary and date of joining of 10 employeesinacompany.Use array of objects. 3. Suppose you have a Piggie Bank with an initial amount of $50 and you must add some more amount to it. Create a class 'AddAmount' with a data member named 'amount' with an initial value of $50. Now make two constructors of this class as follows: 1 - without any parameter - no amount will be added to the Piggie Bank 2 - having a parameter which is the amount that will be added to Piggie Bank Create object of the 'AddAmount' class and display the final amount in Piggie Bank. 4. Create a class named 'Member' having the following members: Data members 1 - Name 2 - Age 3 - Phone number 4 - Address 5 - Salary It also has a method named 'printSalary' which prints the salary of the members. Two classes 'Employee' and 'Manager' inherits the 'Member' class. The 'Employee' and 'Manager' classes have data members 'specialization' and 'department' respectively. Now, assign name, age, phone number, address and salary to an employee and a manager by making an object of both classes and print the same. 6 Write a program to implement below design

		
23	Abstract classes - 1	1.Create an abstract class 'Animals' with two abstract methods 'cats' and 'dogs'. Now create a class 'Cats' with a method 'cats' which prints "Cats meow" and a class 'Dogs' with a method 'dogs' which prints "Dogs bark", both inheriting the class 'Animals'. Now create an object for each of the subclasses and call their respective methods. 2. We must calculate the percentage of marks obtained in three subjects (each out of 100) by student A and in four subjects (each out of 100) by student B. Create an abstract class 'Marks' with an abstract method 'getPercentage'. It is inherited by two other classes 'A' and 'B' each having a method with the same name which returns the percentage of the students. The constructor of student A takes the marks in three subjects as its parameters and the marks in four subjects as its parameters for student B. Create an object for each of the two classes and print the percentage of marks for both the students
24	Extending abstract classes	1.Write an abstract class Shape with the following. – Data members: numSides – Constructor: initialize numSides – Concrete method: get method for numSides – Abstract methods: getArea(), getPerimeter() Write a concrete subclass Rectangle with the following. – Data members: width, height Write a concrete subclass RtTriangle with the following. – Data members: width, height In another class, write a main method to define a Rectangle and a Triangle.
25	Packages and Access Control	1. Access specifiers problem Create a 4 variables and access at different level. a) Same package sub class b) Same package non sub class

		c) Another package sub class d) Another package non sub class. 2) **Private data and method access.** Write a program which creates class A with private data members as "data" . a private member function msg() and a public method hello(). Create a another class Simple which contains the main() and access the private data "data".
26	Predefined and User defined Packages	1) Static Import Write a program which creates a class "StaticImportDemo". Here import statically the predefined System class to in your class. 2) User-defined packages Write a program which creates a student class in "pack" package. Here you define id,name,section name,email as data members, have a display(). And create another package Mypack place" test" class using import statement import the student class print the details of student.
27	Exceptional Handling – Usage of try and catch	1.Exception Handling https://www.hackerearth.com/practice/basic-programming/implementation/basics-of-implementation/practice-problems/algorithm/exception-handling-2-46f67551/ **2. Exceptions** https://www.hackerrank.com/challenges/exceptions/problem 3.Exception Handling https://www.hackerrank.com/challenges/java-exception-handling/problem 4. Exception Handling (Try-catch) https://www.hackerrank.com/challenges/java-exception-handling-try-catch/problem 5. Day 3: Try, Catch, and Finally https://www.hackerrank.com/challenges/js10-try-catch-and-finally/problem
28	Multiple catch blocks, Nested try blocks	1. Exception Handling https://www.hackerearth.com/practice/basic-programming/implementation/basics-of-implementation/practice-problems/algorithm/exception-handling-2-46f67551/ 2. Nested try catch block in – Exception handling

		https://beginnersbook.com/2013/04/nested-try-catch/
29	**T**hrow, Throws and finally blocks User defined Exceptions	**2**. Read two numbers num1 and num2. Your goal is to find integer division. . If Num1 or Num2 were not an integer, the program would throw a Number Format Exception. If Num2 were Zero, the program would throw an Arithmetic Exception.
30	Multithreading - 1	1.Write a program which create a new class which implements interface and override run() method. Then we instantiate a Thread object and call start() method on this object. Like below and calculate Fibonacci series
31	Multithreading - 2	**1) Sereja and Number Division 2** Sereja has an integer number **A** that doesn't contain zeroes in its decimal form. Also he has **N** integers **B[1], B[2], ..., B[N]**. Let us first define function **f** for a number **A** as follows. $\mathrm{f(A)} = \sum_{i=1}^{N} (A \pmod{B[i]})$ Now he has to reorder the digits of **A** such that **f(A)** is minimum. Please help him in finding most optimal **A**. **URL : https://www.codechef.com/JAN15/problems/SEAND2** **2) Multi Threads** Sheldon is addicted to Facebook , and keeps refreshing the main page not to miss any changes in the "recent actions" list. He likes to read thread conversations of where each thread consists of multiple messages. Recent actions shows a list of n different threads ordered by the time of the latest message in the thread. When a new message is posted in a thread that thread jumps on the top of the list. No two messages of different threads are ever posted at the same time. Sheldon has just finished reading all his opened threads and refreshes the main page for some more messages to feed his addiction. He notices that no new threads have appeared in the list and at the i-th place in the list there is a thread that was at the ai-th place before the refresh. He doesn't want to waste any time reading old messages, so he wants to open only threads with new messages.

		Help Sheldon find out the number of threads that surely have new messages. A thread x surely has a new message if there is no such sequence of thread updates (posting messages) that both conditions hold: thread x is not updated (it has no new messages); The list order 1, 2, ..., n changes to a1, a2, ..., an. **URL : https://www.codechef.com/PROM2013/problems/PT2**
32	Introduction to Collection Framework	1) list Sometimes it's better to use dynamic size arrays. Arraylist can provide you this feature. Try to solve this problem using Arraylist. You are given n lines. In each line there are zero or more integers. You need to answer a few queries where you need to tell the number located in yth position of xth line. Take your input from System.in. URL:https://www.hackerrank.com/challenges/java-arraylist/problem?h_r=internal-search 2) Visitor Pattern Note: In this problem you must NOT generate any output on your own. Any such solution will be considered as being against the rules and its author will be disqualified. The output of your solution must be generated by the uneditable code provided for you in the solution template. An important concept in Object-Oriented Programming is the open/closed principle, which means writing code that is open to extension but closed to modification. In other words, new functionality should be added by writing an extension for the existing code rather than modifying it and potentially breaking other code that uses it. This challenge simulates a real-life problem where the open/closed principle can and should be applied. A Tree class implementing a rooted tree is provided in the editor. It has the following publicly available methods: getValue(): Returns the value stored in the node. getColor(): Returns the color of the node. getDepth(): Returns the depth of the node. Recall that the depth of a node is the number of edges between the node and the tree's root, so the tree's root has depth and each descendant node's depth is equal to the depth of its parent node +1. In this challenge, we treat the internal implementation of the tree as being closed to modification, so we cannot directly modify it; however, as with real-world

		situations, the implementation is written in such a way that it allows external classes to extend and build upon its functionality. More specifically, it allows objects of the TreeVis class (a Visitor Design Pattern) to visit the tree and traverse the tree structure via the accept method. There are two parts to this challenge. URL:https://www.hackerrank.com/challenges/java-vistor-pattern/problem
33	Introduction to Collection Framework	1. The Owner of a Supermarket asks the Employe to maintain a detailed record consisting Name, Id and Cost of the item available in their supermarket. He also asks him to get the information of those items in ascending order of the costs. He asks your help for writing a program which takes in the details through the Scanner class and add them to the array list. Your program should also display the details of the items compared in ascending order of the item cost. (Use list and Comparable interface). 2. Mahesh babu was asked to develop a game based on stacks. There will be two players in the game. Every player will get a turn to enter a number and that number will be pushed into a stack if the given number is divided by a number which is randomly generated by the game otherwise the top element of the stack will be popped out. The player whose is left with no numbers in the stack is the loser. Print the Winner of the game
34	Introduction to Collection Framework	1. Madhuri wants to develop an app which sorts the Student Ids based on marks (ascending). The User provides the marks of different students along with student Ids and stores them in a HahMap. Help him out writing the program to develop the app. 2. Naresh wants you to develop a program which takes in words from a list of movies names as dynamic user input and stores them in a Hashset. It should then print them in lexicographical order after converting Hashset into Treeset.
35	Event driven programming	Develop a student registration form that takes Student ID, name, DOB, Email, and mobile number using swing components.
36	Event driven programming	For the above developed form validate the inputs after the user clicks on submit button and if all the details are valid, display a pop-up message as Valid data. Validations: 1. ID and mobile are 10 digits, 2. Name is having only alphabets 3. DOB must be validated, and age should be displayed 4. Email must be validated

EVALUATION PLAN:

Evaluation Type	Evaluation Component	Weightage/Marks		Assessment Dates	Duration (Hours)	CO1	CO2	CO3	CO4	CO5
Blooms Taxonomy Level						**1**	**2**	**3**	**4**	
Semester In Summative Evaluation Total = 34 %	**Semester-in Exam-I**	Weightage	12%	Semester-in Exam-I Dates	2	6%	6%			
		Max Marks	50M			25	25			
	Semester-in Exam –II	Weightage	12 %	Semester-In Exam-II Dates	2			6%	6%	
		Max Marks	50M					25	25	
	Lab Semester-in Exam	Weightage	5 %	Lab Semester-In Exam Dates	1 ½					5%
		Max Marks	50M							50
	Skill Semester-in Exam	Weightage	5 %	Skill Semester-In Exam Dates	1 ½	1%	2%	2%		
		Max Marks	50M			10	20	20		
Formative Evaluation Total = 26 %										
	ALMs/Surprise Quiz	Weightage	8%	Continuous Evaluation		2%	2%	2%	2%	
		Max Marks	50M			12.5	12.5	12.5	12.5	
	Home Assignment	Weightage	4%	Continuous Evaluation		1 %	1 %	1%	1 %	
		Max Marks	50M			12.5	12.5	12.5	12.5	
	Lab Continuous Evaluation	Weightage	6%	Continuous evaluation						6%
		Max Marks	50M							50
	Skill Continuous Evaluation	Weightage	4%	Continuous evaluation						4%
		Max Marks	50M							50
	Benchmark Achievement in Online Coding Platform	Weightage	4%	Continuous evaluation						4%
		Max Marks	50M							50
Semester End Summative Evaluation Total = 40 %	**SE Lab Exam**	Weightage	8%	Lab External Dates	2					8%
		Max Marks	50M							50
	SE Skill (Project)	Weightage	8%	Skill External Dates	2					8%
		Max Marks	50M							50
	Semester End Exam	Weightage	24%	Semester End Exam Dates	3	6%	6%	6%	6%	
		Max Marks	100M			25	25	25	25	

ATTENDANCE POLICY

Every student is expected to be responsible for regularity of his/her attendance in classrooms and laboratories, to appear in scheduled tests and examinations and fulfil all other tasks assigned to him/her in every course. In every course, student must maintain a minimum of 85% attendance to be eligible for appearing in Semester end examination of the course, for cases of medical issues and other unavoidable circumstances the students will be condoned if their attendance is between 75% to 85% in every course, subjected to submission of medical certificates, medical case file and other needful documental proof to the concerned departments/Dean.

DETENTION POLICY

In any course, a student must maintain a minimum of 85% attendance to be eligible for appearing to the Semester End Examination, failing to fulfil these conditions will deem such student to have been detained in that course.

COURSE TEAM MEMBERS, CHAMBER CONSULTATION HOURS AND CHAMBER VENUE DETAILS:

Each instructor will specify his / her chamber consultation hours during which the student can contact him / her in his / her chamber for consultation.

S.No.	Name of Faculty	Chamber Consultation Day (s)	Chamber Consultation Timings for each day	Chamber Consultation Room No:	Signature of Course faculty
1	Dr. D. Haritha, HOD BES-1	All working days	2:00P.M to 3:20 P.M	F210	
2	Dr P.Siva Kumar	All working days	2:00P.M to 3:20 P.M	F206	
3	Mr.N.Sreeram	All working days	2:00P.M to 3:20 P.M	F206	
4	Dr.S.Siva Kumar	All working days	2:00P.M to 3:20 P.M	F206	
5	Dr.Sk.Razia	All working days	2:00P.M to 3:20 P.M	F202 (Lab)	
6	Dr. E. SreeDevi	All working days	2:00P.M to 3:20 P.M	F206	
7	Dr. N V S Pavan Kumar	All working days	2:00P.M to 3:20 P.M	F206	
8	Mr. CMAK Zeelan Basha	All working days	2:00P.M to 3:20 P.M	F206	
9	Mr V.Uday Kumar	All working days	2:00P.M to 3:20 P.M	F206	
10	Mr D.Anand	All working days	2:00P.M to 3:20 P.M	F206	

11	Mr.T.Rajesh Kumar	All working days	2:00P.M to 3:20 P.M	F206	
12	Ms. V.Premalatha	All working days	2:00P.M to 3:20 P.M	F206	
13	Mr.A.Krishna	All working days	2:00P.M to 3:20 P.M	C205	
14	Mr. Pradeep Raj Savarapu	All working days	2:00P.M to 3:20 P.M	F206	
15	Mr.ASALG Gopal Gupta	All working days	2:00P.M to 3:20 P.M	F206	
16	Mr.E.Rajesh Kumar	All working days	2:00P.M to 3:20 P.M	F206	
17	Mr.G.Rama Krishna Srinivas	All working days	2:00P.M to 3:20 P.M	F206	
18	Ms.Karimunnisa	All working days	2:00P.M to 3:20 P.M	F206	
19	Ms.V.lakshmi Sarvani	All working days	2:00P.M to 3:20 P.M	F206	
20	Mr.T.Ganesan	All working days	2:00P.M to 3:20 P.M	F206	
21	Mr.B.Ashok	All working days	2:00P.M to 3:20 P.M	F206	
22	Mr.M.Ram Kumar	All working days	2:00P.M to 3:20 P.M	C205	
23	Mr.A.Srinivasa Rao	All working days	2:00P.M to 3:20 P.M	F206	
24	Ms.P.Gayatri	All working days	2:00P.M to 3:20 P.M	F206	
25	Mr.Y.Ayyappa	All working days	2:00P.M to 3:20 P.M	F206	
26	Mr.S.Siva Kumar	All working days	2:00P.M to 3:20 P.M	C205	
27	Ms.S.Harika	All working days	2:00P.M to 3:20 P.M	F206	
28	Dr.Naveen Kumar	All working days	2:00P.M to 3:20 P.M	C205	

29	Ms.M.Anila	All working days	2:00P.M to 3:20 P.M	F206	
30	Mr.G.K. Chakravarthi	All working days	2:00P.M to 3:20 P.M	F206	
31	Ms.T. Yamini	All working days	2:00P.M to 3:20 P.M	F206	
32	Mr.P. Neelakanteswara	All working days	2:00P.M to 3:20 P.M	F206	
33	Ms.Shilpa Itnal	All working days	2:00P.M to 3:20 P.M	C205	
34	Mr.J. Nagaraju	All working days	2:00P.M to 3:20 P.M	C205	
35	Mr.Ch. Naresh	All working days	2:00P.M to 3:20 P.M	C205	
36	Dr. M.Anusha	All working days	2:00P.M to 3:20 P.M	F202 (Lab)	
37	Mrs. U. Harita	All working days	2:00P.M to 3:20 P.M	F206	

GENERAL INSTRUCTIONS

Students should come prepared for classes and carry the textbook(s) or material(s) as prescribed by the Course Faculty to the class.

NOTICES.

All notices will be communicated through the institution email.
All notices concerning the course will be displayed on the respective Notice Boards.

Signature of COURSE COORDINATOR:

Signature of Department Prof. In charge Academics & Vetting Team Member:

HEAD OF DEPARTMENT:

Approval from: DEAN-ACADEMICS
(Sign with Office Seal)