A Complete Workbook on Core Java Programming

Compiled By Er. Brijesh Mishra

Under the Expert Guidance of
Er. Ajay Chaudhary
MD & Chairman, Softpro Group

About the Workbook….

The "**Workbook on Core Java**" has been written keeping in mind the requirements of a learner of Java on practical grounds. The Workbook carries the multiple question modules like Long Answers, Short Answers, Multiple Choice Type & even Technical Tasks carrying programs for complete learning of a student.

"Whether you're a final year student with a challenging project in mind; a pre-final year student with plans for planting roots in Java Technology, or an enthusiastic second year student with zeal to learn a new technology, or a fresher seeking job in Java, this Workbook has everything for you."

Team Softpro always is fond of coming up with some or the other new ventures which are helpful & beneficial for the students. We hope that this Workbook will also play a vital role in your journey of Learning Java Technology.

You can always give us suggestions & feedback for more improvements & enhancements. Write down to us on: - hr@softproindia.in

Thanks,
Team Softpro

Message by the Director

Dear Readers,

Softpro Group has been growing exponentially since years and it is your trust in us that gives us the power and encouragement to take bigger and better steps. I feel immense happiness that now we are presenting you with this **Workbook on Java** and are able to assure you with the quality of its content.

The Workbook delivers the enrich question bank prepared by the author which he has earned with the years of experience working on Java and hence the "**Workbook on Core Java"** will help you to become an expert Java Developer yourself.

I wish you the best for the future.

Ajay

Er. Ajay Chaudhary
Director
Softpro India

From the Desk of the CEO

Dear Readers,

Our ongoing endeavor to excel and to succeed has penned a saga of success in I.T. Training. Now, Softpro India is more than happy to present you with this **"Workbook"** which will provide you enough knowledge & will a make you practice on Java to become an expert in Java.

I thank the author Mr. Brijesh Mishra, Consultant, Softpro India for presenting a full decade of his experience in the form of this Workbook and hope you will make the most out of the sacred knowledge it gives.

Peace and joy.

Yashi

Yashi Asthana
Chief Executive Officer
Softpro India

Acknowledgement

Firstly, I pay my regards & thank Lord Shri Rama for bestowing me with His blessing which enabled me to successfully pen down this Workbook on Java. It would have been impossible without His grace.

Secondly I will thank the Director Sir Ajay Chaudhary for giving me this golden opportunity of sharing my years of experience in Java to all the Readers.

I will also take this opportunity to thank the staff of Softpro India who has been a constant support throughout my journey.

Thank you,

Er. Brijesh Mishra
Consultant
Softpro India

About the Writer

Brijesh Mishra is having a fulfilling Experience of more than 15 Years working in the Software Development field and have a great command in Programming Concepts & Database Administration. He has worked on ERPs, customized software applications, cloud-based applications, PLC devices and many more. Currently he is working as **Senior Consultant** in Softpro India Computer Technologies P Limited.

COMPETENCY IN TECHNOLOGIES

- **Languages:** C, C++, C#, Python, Java, Assembly, R, Go
- **Microsoft Based:** VB /ASP /.NET/SQL Server
- **Database Applications:** Oracle Enterprise Edition & Oracle Developer 2K
- **Platforms:** Android, Windows, Linux
- **Frameworks:** Struts, Hibernate, Spring
- **Business Domains:** Finance, Retail, Mobile Applications, ERP, Telecom.
- **Others:** Machine Learning, IOT, Data Science, Embedded System, PLC Programming with SCADA for Industry Automation

Thank you,

Table of Contents

5.	String in Java What is a string? Built-in functions in string Work on built-in functions
6.	Method in Java Importance of method in Java Types of methods Difference between static and nonstatic methods Recursion
7.	Object-Oriented Programming System Pillars of OOPS Class and object Constructor in java Types of constructor
8.	Inheritance in Java Importance of Inheritance Types of Inheritance
9.	Polymorphism in java Types of Polymorphism in Java Method Overloading Method Overriding Difference between method overloading and overriding
10.	Interface in Java Importance of Interface in Java Interface, Abstract class and class – A discussion Difference between interface and abstract class
11.	Exception in Java Types of exceptions in Java Exception handling in java Types of exception handling in Java

	Explanation of try, catch and finally Difference between throw and throws
12.	Multithreading in Java Difference between Multithreading and Multitasking Thread life cycle Creation of thread by extending Thread class Creation of thread by implementing Runnable interface
13.	Package in Java Types of packages in Java Creation of Package
14.	Collection framework in Java Importance of Collection Framework ArrayList class LinkedList class Iterator interface
15.	ListIterator Interface HashSet Class LinkedHashSet Class TreeSet Class

Java Lecture – 01

(Introduction to Java)

Long Answer Questions: -

1. **What is Java? Write its features with an explanation.**

Ans: - Java is a powerful object-oriented programming language developed by **James Gosling** in the year **1995**.

Java Features: -

1. Simple
2. Object-Oriented
3. Platform Independent
4. Architectural Neutral
5. Portable
6. Robust
7. Secure
8. Dynamic
9. Distributed
10. Multithreaded
11. Interpretive
12. High Performance

1. Simple: -

Java is a simple programming language because:

- Java technology has eliminated all the difficult and confusion-oriented concepts like pointers & multiple inheritances.
- The C, CPP syntaxes are easy to understand and easy to write. Java maintains C and CPP syntax mainly hence Java is a simple language.
- Java technology takes less time to compile and execute the program.

2. Object-Oriented: -

Java is an object-oriented technology as it represents total data in the form of objects. By using object reference, we are calling all the methods, variables which is present in that class.

3. Platform Independent: -

- Compile the Java program on one OS (operating system). That compiled file can be executed on any OS (operating system). This phenomenon is called Platform Independent Nature.
- Java is a platform-independent language. The Java applications allow its applications compilation at one operating system which can be compiled (.class files) and can be executed on any other operating system.

4. Architectural Neutral: -

Java Tech Applications that are compiled on one Architecture (Hardware----RAM, Hard Disk) and then that Compiled program runs on any other hardware architecture (hardware). This is called Architectural Neutral.

5. Portable: -

In Java tech, the applications are compiled and executed on any OS (Operating System) and in any Architecture (hardware) hence we can say Java is a portable language.

6. Robust: -

Any technology which is good in the below mentioned areas, is said to be ROBUST.

1. Exception Handling
2. Memory Allocation

JAVA is Robust because

- JAVA is having very good predefined Exception Handling Mechanism. If we get any exception, we will get meaningful information.
- JAVA is having a very good memory management system that is Dynamic Memory Allocation (memory is allocated at runtime) which allocates and deallocates memory for objects at runtime.

7. Secure: -

- To provide implicit security feature, Java provides one component inside JVM which is called Security Manager.
- To provide explicit security for the Java applications, we are having a very good predefined library in the form of java.security package.

8. Dynamic: -

Java is a dynamic technology as it follows dynamic memory allocation (at runtime the memory is allocated) and dynamic loading to perform the operations.

9. Distributed: -

By using JAVA technology, we prepare Standalone Applications and Distributed Applications.

- Standalone Applications are those Java Applications that doesn't need Client-Server Architecture.
- Web Applications are those Java Applications which need Client-Server Architecture.
- Distributed applications are those Java Applications where the project code is distributed in multiple numbers of JVM's.

10. Multithreaded: -

- Thread is a light weight process and a small task in a large program.
- If any tech allows executing a single thread at a time such type of technologies are called single threaded technology.
- If any technology allows creating and executing more than one thread at a time, they are called as multithreaded technology e.g. JAVA.

11. Interpretive: -

Java tech is both Interpretive and Completive. By using Interpreter, we convert source code into byte code. The interpreter is a part of JVM.

12. High Performance: -

If any technology has features like Robust, Security, Platform Independent, Dynamic and so on, that technology is a high performance technology.

2. Explain JDK, JRE and JVM.

Ans: - JDK : The Java Development Kit (JDK) is one of the three core technology packages used in Java programming along with the JVM (Java Virtual Machine) and the JRE (Java Runtime Environment).

JRE: A Java Runtime Environment (JRE) is a set of components used to create and run a Java application. A JRE is part of a Java Development Kit (JDK).

JVM: - JVM (Java Virtual Machine) is an abstract machine. It is a specification that provides runtime environment in which java byte code can be executed.

3. Explain coding convention in java.

Java coding conventions: -

Classes: -

- Class name starts with an upper case letter and every inner word also starts with an upper case letter.
- This convention is also known as **camel** case convention.
- The class name should be noun.

Ex: - **S**tring, **S**tring**B**uffer, **I**nput**S**tream**R**eader ………………etc.

Interfaces: -

- Interface name starts with an upper case and every inner word also starts with an upper case letter.
- This convention is also known as **camel** case convention.
- The Interface name should be noun.

Ex: - **S**erializable, **C**loneable, **R**andom**A**ccess…………….etc.

Methods: -

- Method name starts with lower case letter and every inner word starts with upper case letter.
- This convention is known as mixed case convention.
- Method names should be verb.

Ex: - post(), char**A**t(), to**U**pper**C**ase(), compare**T**o**I**gnoreCase()………………etc.

Variables: -

- Variable name starts with lower case letter and every inner word starts with upper case letter.
- This convention is also known as mixed case convention.

Ex: - out, in, page**C**ontext ………etc.

Package: -

- Package name must be written in lower case letter.

Ex: - java.long, java.io, java.util ……………….etc.

Constants: -

- While declaring a constant, all the words are uppercase letters.

Ex: MAX_PRIORITY MIN_PRIORITY NORM_PRIORITY

NOTE: -The coding standards are applicable only for predefined library and not for user defined library but it is recommended to follow the coding standards for user defined library also.

Java Identifiers: - Any name in the java program like variable name, class name, method name or interface name is called identifier.

4. **How to take user input in Java? Explain step by step.**

Ans: - Java.util.Scanner (Dynamic Input): -

1. **Scanner** class present in **java.util** package is introduced in 1.5version.

2. **Scanner** class is used to take dynamic inputs from the keyboard.

```
Scanner s = new Scanner(System.in);

to get int value  ----> s.nextInt()

to get float value  ---> s.nextFloat()

to get byte value  ---> s.nextbyte()

to get String value  ---> s.next()

to get single line ---> s.nextLine()

to close the input stream ---> s.close()
```

Short Answer Questions: -

1. **Write the examples of Editor and IDE.**

Ans: - Editor: - Editor is a tool or software which provides a very good environment to develop java applications. Ex: - Notepad, Notepad++, edit Plus…..etc.

IDE: - IDE provides a very good environment to develop the application and is a real-time standard. However it is recommended not to use IDE to develop Core Java Applications.

2. **What is OOPS? Write its pillars.**

Ans: - OOPS: - OOPS stands for Object Oriented Programming System. It is a mechanism of software development. It has four pillars: -

i. Abstraction
ii. Encapsulation
iii. Inheritance
iv. Polymorphism

3. **Explain program structure in Java.**

Ans: - Program structure of a java program is given below: -

```
import java.lang.System;
import java.lang.String;
 class Test //class declaration
{
 //class starts
 public static void main(String[] args) //program starting point
 {
 //main starts
 System.out.println("Hello World");  //printing statement
} //main ends
} //class ends
```

4. **What is Class? Explain it.**

Ans: - Class is a collection of variables and methods. Class is declared by using class keyword followed by class name. We create variables and methods within the body of class.

5. **What is package? Explain it.**

Ans: - Package is the collection of classes, interfaces and sub-packages.

6. **What is the work of Java Compiler?**

Ans: - Java compiler is used to convert source code to byte code.

7. **What is the work of JVM?**

Ans: - JVM stands for Java Virtual Machine. JVM converts byte code to machine code.

8. **What is the work of JRE?**

Ans: - JRE stands for Java Runtime Environment. The Java Program executes under JRE.

9. **What is Multithreading?**

Ans: - The concept of multithreading is taken from multitasking. In Multitasking, CPU switches between multiple processes whereas in multithreading, CPU switches between multiple sub-processes.

10. **Why Java is secured?**

Ans: - In Java programming language, when you run a java program, JVM converts byte code to machine code. JVM verifies byte code first then convert it into machine code therefore java is called a secured language.

Technical Tasks: -

1. **Develop a Java program to print a message “Welcome to the World of Java” on screen.**

```
import java.lang.System;
import java.lang.String;
 class Test //class declaration
{
 //class starts
 public static void main(String[] args) //program starting point
 {
 //main starts
 System.out.println("Welcome to the World of Java");  //printing statement
} //main ends
} //class ends
```

2. **Develop a Java program to find the volume and surface area of cuboid.**
   v=l*b*h;
   sa=2*(l*b+b*h+h*l);

```
import java.util.Scanner;
class Cuboid
{
public static void main(String [] args)
{
int l,b,h;
Scanner sc=new Scanner(System.in);
System.out.print(“Enter length of cuboid : ”);
l=sc.nextInt();
System.out.print(“Enter breadth of cuboid : ”);
b=sc.nextInt();
System.out.print(“Enter height of cuboid : ”);
h=sc.nextInt();
int v=l*b*h;
int sa=2*(l*b+b*h+h*l);
System.out.println(“Volume of cuboid : ”+v);
System.out.println(“Surface Area of cuboid : ”+sa);
}
}
```

3. **Develop a Java program to calculate simple interest.**

```
import java.util.Scanner;
class SimpleInterest
{
public static void main(String [] args)
{
float p,n,r;
double si;
Scanner sc=new Scanner(System.in);
System.out.print("Enter principle amount : ");
p=sc.nextFloat();
System.out.print("Enter rate : ");
r=sc.nextFloat();
System.out.print("Enter time : ");
n=sc.nextFloat();
si=(p*n*r)/100;
System.out.println("Simple Interest : "+si);
}
}
```

4. **Develop a Java program to find area and perimeter of circle.**

```
import java.util.Scanner;
class Test
{
public static void main(String [] args)
{
float r;
double a,p;
Scanner sc=new Scanner(System.in);
System.out.print("Enter radius of circle : ");
r=sc.nextFloat();
a=3.14*r*r;
p=2*3.14*r;
System.out.println("Area of circle : "+a);
System.out.println("Perimeter of circle : "+p);
}
}
```

5. **Write a Java program to convert given number of days to a measure of time given in years, weeks and days. For example 375 days is equal to 1 year 1 week and 3 days (ignore leap year).**

```
import java.util.*;
class Test
{
        public static void main(String [] args)
        {
                int dayno,year,week,day;
                Scanner sc=new Scanner(System.in);
                System.out.print("Enter no of days : ");
                dayno=sc.nextInt();
                year=dayno/365;
                week=(dayno%365)/7;
                day=(dayno%365)%7;
                System.out.println(year+" Years "+week+" weeks "+day+" days");
        }
}
```

Interview Question: -

1. **What is present version of Java and initial version of Java?**
   Ans: - Initial Version: Java 1.0   Present Version: Java 17

2. **How many modifiers are there in Java and how many keywords are there in Java?**
   Ans: - In Java programming language, there are 11 modifiers and 50 keywords.

3. What was the initial name of Java and present name of Java?
   Ans: - Initial Name: OAK   Present Name: Java

4. Can we have multiple public classes in single source file?
   Ans: - No, we can't create multiple public classes in single source file.

5. Can we create multiple objects for single class?
   Ans: - Yes, we can create multiple objects for single class.

6. What do you mean by token and literal?
   Ans: - Java tokens are smallest elements of a program which are identified by the compiler. Tokens in Java include identifiers, keywords, literals, operators and separators.

7. What do you mean by identifier?
   Ans: - Any name in the Java program like variable name, class name, method name, interface name is called identifier.

8. In Java, program starts from which method and who calls that method?
   Ans: - Java program starts from main method. When the Java interpreter executes an application (by being invoked upon the application's controlling class), it starts by calling the class's main method.

9. What are the commands required for compilation and execution?
   Ans: - For compilation: - javac   filename   For execution: - java   classname

10. The compiler understandable file format and JVM understandable file format?
    Ans: - Compiler understandable file - .java   JVM understandable file - .class

11. What is the difference between path and class path?
    Ans: - Once you install Java on your machine, it is required to Set the PATH environment variable to conveniently run the executable (javac.exe, java.exe, javadoc.exe, and so on) from any directory without having to type the full path of the command. Classpath is a system environment variable used by the Java compiler and JVM. Java compiler and JVM use classpath to determine the location of required class files.

12. What do you mean by open source software?
    Ans: - The source code of open source software is available for all software vendors. The software engineer can modify open source software.

13. What operations are done at compilation time and execution time?
    Ans: - Java compiler convert source code to byte code at compilation time and JVM converts byte code to machine code at run time.

14. What is the purpose of JVM?
    Ans: - JVM converts byte code to machine code.

15. JVM is platform dependent or independent?
    Ans: - JVM is platform dependent.

16. In Java, program execution starts from?
    Ans: - Java program execution starts from main() method.

17. What does a .class file contains?
    Ans: - .class file contains byte code.

18. Is null is a keyword or not?
    Ans: - null is not a keyword, it is a literal.

19. Is it possible to declare multiple classes with main method?
    Ans: - Yes, it is possible to declare multiple classes with main method.

20. What is the default package in java?
    Ans: - java.lang is the default package in java.

21. Is empty java source file is valid or not?
    Ans: - Yes, empty java source file is valid.

22. Is it java file contains more than one class?
    Ans: - Yes, java file can contain more than one class.

23. What is the purpose of variables in java?
    Ans: - Variables are value containers.

24. How many types of variables are there in java and what are those variables?
    Ans: - In java programming language there are four types of variables: -

i.)	static variables	iii.) non-static variables
ii.)	local variables	iv.) parameters

Multiple Choice Questions: -

1. Which component is used to compile, debug and execute a Java program?
   a) JVM
   b) JDK
   c) JIT
   d) JRE

2. Which component is responsible for converting bytecode into machine specific code?
   a) JVM
   b) JDK
   c) JIT
   d) JRE

3. Which component is responsible to run java program?
   a) JVM
   b) JDK
   c) JIT
   d) JRE

4. Which statement is true about java?
   a) Platform independent programming language
   b) Platform dependent programming language
   c) Code dependent programming language
   d) Sequence dependent programming language

5. Which of the below is invalid identifier with the main method?
   a) public
   b) static
   c) private
   d) final

6. What is the extension of java code files?
   a) .class
   b) .java
   c) .txt
   d) .js

7. What is the extension of compiled java classes?
   a) .class
   b) .java
   c) .txt
   d) .js

Answer Key: -

1. b	2. a	3. d	4. a
5. c	6. b	7. a	

Fill in the blanks questions: -

1. The output of the Java compiler is known as -----------------.
   Ans: - Byte Code.

2. The ---------- statement is used to include another Java package in a Java source file.
   Ans: - import

3. Java supports -------------------programming.
   Ans: - Multithreaded

4. The output of the Java compiler is executed by the ----------.
   Ans: - JVM

5. Java byte code output from the JDK compiler will be placed into a file with ----------extension.
   Ans: - .class

Java Lecture – 02

(Decision Controls In Java)

Long Answer Questions: -

1. What is Decision Control and what is the purpose of Decision Control?

   Ans: - Decision Controls are used for decision making. If you have a block of code which you want to execute based on some condition, then you can use decision control. In Java programming language there are following types of decision controls: -

   i.) if statement
   ii.) if-else statement
   iii.) nested if – else statement
   iv.) ladder if – else statement
   v.) switch statement

2. Write short notes on the following: -

   i. if statement
   ii. if-else statement
   iii. nested if – else statement
   iv. ladder if – else statement
   v. switch statement

   Ans: -

   i.) **if statement: -** if is a keyword which works as decision control. We attach a condition with if statement. If given condition is true, then the code will be executed and if given condition is false then the code will not be executed. The syntax of if statement is given below: -

```
if(condition)
{
//code
}
```

   ii.) **if – else statement: -** if-else is the variation of if statement. We attach a condition with if statement, if given condition is true then the if block code will be executed and if given condition is false then the else block code will be executed.
   The syntax of if – else statement is given below: -

```
if(condition)
{
//if block code
}
else
{
//else block code
}
```

iii.) **nested if – else: -** if you use if – else construct inside if block or else block or both blocks then it is called nested if – else.

iv.) **ladder if – else statement: -** If you have many conditions and you want to execute code based on those conditions , then you can use ladder if – else. Syntax of ladder if – else is given below: -

```
if(condition1)
{
//code1
}
else if (condition2)
{
//code2
}
else
{
//code3
}
```

v.) **switch statement: -** switch is a keyword which works as case control. It is used to make a menu based program.

3. What is the purpose of switch statement? Write the syntax of switch statement.
Ans: - switch statement: - switch is a keyword which works as case control. It is used to make a menu based program. The Syntax of switch statement is given below: -

```
switch(expression)     //int or char or String
{
case 1:
//code1
break;
case 2:
//code2
break;
default:
//code
break;
}
```

Short Answer Questions: -

1. What is the difference between if and if-else?
   Ans: - In if statement, we attach a condition, if given condition is true then the code will be executed and if given condition is false then the code will not be executed. Whereas in if – else statement we attach a condition with if statement, if given condition is true then if block code will be executed and if given condition is false then else block code will be executed.

2. What is the purpose of conditional operator?
   Ans: - The conditional operator is alternate of if – else statement. The syntax of conditional operator is given below: -
   (expression1)?(expression2): (expression3);
   If expression1 is true then expression2 will be executed and if expression1 is false then expression3 will be executed.

3. What is the difference between ladder if – else and switch?
   Ans: - In else if ladder, the control goes through every else if statement until it finds true value of the statement or it comes to the end of the else if ladder. In case of switch case, as per the value of the switch, the control jumps to the corresponding case.

4. What is the difference between conditional statement and if – else?
   Ans: - The java ternary operator or conditional operator is supported in languages such as Java, Javascript, Python, C / C++, C# etc. The java ternary operator or conditional operator takes three operands, one condition followed by a question mark (?), then an expression to be executed if the condition is true followed by a colon (:), and the expression to be executed if the condition is false. It is similar to "if else" statement.

5. Write syntax of ladder if – else.
   Ans: - Syntax of ladder if – else is given below: -

```
if(condition1)
{
//code1
}
else if (condition2)
{
//code2
}
else
{
//code3
}
```

Technical Tasks: -

1. Develop a program in Java to check whether the given number is Even or Odd.

```
import java.util.Scanner;
class Test
{
public static void main(String [] args)
{
int n;
Scanner sc=new Scanner(System.in);
System.out.print("Enter a number : ");
n=sc.nextInt();
if(n%2==0)
{
System.out.println("Number is even");
}
else
{
System.out.println("Number is odd");
}
}
}
```

2. Develop a program in java to find roots of quadratic equation $ax^2+bx+c=0$.
   Hint: - To find square root use Math.sqrt() method. Math is a built-in class in java.util package.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
double a,b,c,d,r1,r2;
Scanner sc=new Scanner(System.in);
System.out.println("Enter value for a,b and c");
a=sc.nextDouble();
b=sc.nextDouble();
c=sc.nextDouble();
d=(b*b-4*a*c);
if(d<0)
{
System.out.println("Roots are imaginary");
}
else
{
r1=(-b+Math.sqrt(d))/(2*a);
r2=(-b-Math.sqrt(d))/(2*a);
System.out.println("Root1 : "+r1);
System.out.println("Root2 : "+r2);
}
}
}
```

3. Develop a program in Java to accept basic salary from user and calculate gross salary on following basis: -

BASIC	**HRA**	**DA**
1 – 4000	10%	50%
4001 – 8000	20%	60%
8001 - 12000	25%	70%
12000 and above	30%	80%

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
double bs,hra,da,gs;
Scanner sc=new Scanner(System.in);
System.out.print("Enter basic salary : ");
bs=sc.nextDouble();
if(bs<=4000)
{
hra=bs*10/100;
da=bs*50/100;
}
else if(bs>4000 && bs<=8000)
{
hra=bs*20/100;
da=bs*60/100;
}
else if(bs>8000 && bs<=12000)
{
hra=bs*25/100;
da=bs*70/100;
}
else
{
hra=bs*30/100;
da=bs*80/100;
}
gs=bs+hra+da;
System.out.println("Gross Salary="+gs);
}
}
```

4. Develop a Java program to accept a coordinate point in an XY coordinate system and determine its quadrant.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
int x,y;
Scanner sc=new Scanner(System.in);
System.out.print("Enter value for x and y : ");
x=sc.nextInt();
y=sc.nextInt();
if(x>0 && y>0)
{
System.out.println("First Quadrant");
}
else if(x<0 && y>0)
{
System.out.println("Second Quadrant");
}
else if(x<0 && y<0)
{
System.out.println("Third Quadrant");
}
else if(x>0 && y<0)
{
System.out.println("Fourth Quadrant");
}
}
}
```

5. Develop a Java program to accept number of units consumed and calculate electricity bill: -

Unit	Bill/Unit
1-150	2.40
For next 151-300	3.00
For next more than 300	3.20

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
double unit,bill;
Scanner sc=new Scanner(System.in);
System.out.print("Enter number of units consumed : ");
unit=sc.nextDouble();
if(unit<=150)
{
bill=unit*2.40;
}
else if(unit>150 && unit<=300)
{
bill=(150*2.40)+(unit-150)*3.00;
}
else
{
bill=(150*2.40)+(150*3.00)+(unit-300)*3.20;
}
System.out.println("Your bill="+bill);
}
}
```

6. Develop a Java program to make a simple calculator using switch.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
int a,b,res=0,ch;
Scanner sc=new Scanner(System.in);
System.out.println("Enter two numbers");
a=sc.nextInt();
b=sc.nextInt();
System.out.println("1-> Addition");
System.out.println("2-> Subtraction");
System.out.println("3-> Multiplication");
System.out.println("4-> Division");
ch=sc.nextInt();
switch(ch)
{
case 1:
res=a+b;
break;
case 2:
res=a-b;
break;
case 3:
res=a*b;
break;
case 4:
res=a/b;
break;
default:
System.out.println("Invalid choice");
break;
}
System.out.println("Result="+res);
}
}
```

Interview Questions: -

1. What is Decision Control?
   Ans: - Decision controls are used for decision making.

2. How many decision control statements are there in Java?
   Ans: - In Java, following types of decision controls are present: -

   i.) if
   ii.) if-else
   iii.) nested if – else
   iv.) if – else ladder
   v.) switch

3. What are the allowed arguments of switch?
   Ans: - In switch int, char & string arguments are allowed.

4. Does the Switch allow String argument or not?
   Ans: - Yes, switch allows String argument.

5. Inside the switch statement, how many cases are possible and how many default declarations are possible?
   Ans: - Inside the switch statement, any number of cases are possible but only one default declaration is possible.

6. We are able to use break statements in how many places and what are those places?
   Ans: - The break statement has two separate and distinct uses: exiting a loop and exiting a switch statement. You cannot use break anywhere else except inside a loop or a switch statement.

7. What do you mean by transfer statements and what transfer statements are present in java?
   Ans: - Java provides six language constructs for transferring control in a program:

   i.) break
   ii.) continue
   iii.) return
   iv.) try-catch-finally
   v.) throw
   vi.) assert

Multiple Choice Questions: -

1. Which of these selection statements test only for equality?
   a) if
   b) switch
   c) if & switch
   d) none of the mentioned
2. Which of these are selection statements in Java?
   a) if()
   b) for()
   c) continue
   d) break
3. Which of the following is used with the switch statement?
   a) Continue
   b) Exit
   c) break
   d) do
4. What is true about a break?
   a) Break stops the execution of entire program
   b) Break halts the execution and forces the control out of the loop
   c) Break forces the control out of the loop and starts the execution of next iteration
   d) Break halts the execution of the loop for certain time frame
5. Which of the following is not a decision making statement?
   a) if
   b) if-else
   c) switch
   d) do-while
6. Which of the following is not a valid jump statement?
   a) break
   b) goto
   c) continue
   d) return
7. Which of the following is not a valid flow control statement?
   a) exit()
   b) break
   c) continue
   d) return

Answer Key: -

1. b	2. a	3. c	4. c
5. d	6. b	7. a	

Fill in the blanks Questions: -

1. An if or else if statement accepts ____________ as input before branching.
   Ans: - boolean

2. An if statement in Java is also a ____________ statement.
   Ans: - conditional

3. Java style if-else statements are similar to ______________.
   Ans: - Both C and C++ style

4. An else statement must be preceded by ______________ statement in Java.
   Ans: - if and else if both

5. _____________statements test only for equality.
   Ans: - switch

Java Lecture -03

(Loop Controls In Java)

Long Answer Questions: -

1. What is the purpose of Loop Controls? How many types of loop controls are there in Java?
   Ans: - Loop Controls: - If you have a block of code which you want to execute repeatedly up to the given condition is true, then you can use a loop control. There are four types of loop controls in java: -

   i.) while loop
   ii.) for loop
   iii.) do-while loop
   iv.) for each loop

2. What is the difference between while and do – while loop?
   Ans: - While is an entry loop control whereas do-while is an exit loop control. In while loop, first condition is tested then code is executed whereas in do-while loop, condition is tested after execution of code.

   Syntax of while loop: -
   ```
   Initialization of loop counter;
   while(condition)
   {
   //code
   Updation of loop counter
   }
   ```

   Syntax of do-while loop: -
   ```
   Initialization of loop counter;
   do
   {
   //code
   Updation of loop counter;
   }
   while(condition);
   ```

3. What is the difference between for loop and for each loop?
   Ans: - For loop: - for is a keyword which works as loop control. The working of for loop is same as while loop but syntax is different. The for loop is an entry control. The syntax of for loop is given below: -
   ```
   for(Initialization of loop counter; Condition; Updation of loop counter)
   {
   ```

```
//code
}
```

For each loop: - for each loop is a special loop control. It is used to traverse the elements of a collection. E.g. if you want to traverse the elements of an array you can use for each loop.

```
int [] x={10,20,30,40,50};
for(int n:x)
{
System.out.println(n);
}
```

Short Answer Questions: -

1. What is while loop? Describe with syntax.
   while loop: - while is a keyword which works as loop control. while is an entry loop control. The syntax of while loop is given below: -

   ```
   Initialization of loop counter;
   while(Condition)
   {
   //code
   Updation of loop counter;
   }
   ```

2. Explain for each loop with an example.
   For each loop: - for each loop is a special loop control. It is used to traverse the elements of a collection.
   E.g. if you want to traverse the elements of an array you can use for each loop.

   ```
   int [] x={10,20,30,40,50};
   for(int n:x)
   {
   System.out.println(n);
   }
   ```

3. for (; ;) represents?
   Ans: - It's an infinite loop, equivalent to while(true). When no termination condition is provided, the condition defaults to true.

4. Explain do-while loop with example.

   Ans: - do-while is an exit loop control. In do-while loop condition is tested after execution of code. If you have a code which you want to execute at least one time either condition is true or false then you can use do-while loop.

   E.g.

```
int i;
i=1;
do
{
System.out.println(i);
i++;
}
while (i<=10);
```

   The above code will display numbers from 1-10.

5. Explain for loop with example.

   Ans: - For Loop: - for is a keyword which works as loop control. The working of for loop is same as while loop but syntax is different.

   The example of for loop is given below: -

```
for(int i=1;i<=10;i++)
{
System.out.println(i);
}
```

   The above code will display numbers from 1-10.

6. Explain nested for loop with an example.

   Ans: - Nested for loop: - If we use a for loop inside another for loop. It is called nested for loop.

Technical Tasks: -

1. Develop a java program to print the table of a given number in given format like: -
   2*1=2
   2*2=4
   2*3=6
   ……….

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
int n,i;
Scanner sc=new Scanner(System.in);
System.out.print("Enter a number to print table : ");
n=sc.nextInt();
for(i=1;i<=10;i++)
{
System.out.println(n+"*"+i+"="+(n*i));
}
}
}
```

2. Develop a java program to find factorial of given number.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
int n,f=1;
Scanner sc=new Scanner(System.in);
System.out.print("Enter a number to find factorial : ");
n=sc.nextInt();
while(n>0)
{
f=f*n;
n--;
}
System.out.println("Factorial="+f);
}
}
```

3. Develop a java program to find sum of digits of given number.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
int n,r,s=0;
Scanner sc=new Scanner(System.in);
System.out.print("Enter a number to find sum of digits : ");
n=sc.nextInt();
while(n>0)
{
r=n%10;
s=s+r;
n=n/10;
}
System.out.println("Sum of digits="+s);
}
}
```

4. Develop a java program to reverse the digits of given number.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
int n,r,rev=0;
Scanner sc=new Scanner(System.in);
System.out.print("Enter a number : ");
n=sc.nextInt();
while(n>0)
{
r=n%10;
rev=rev*10+r;
n=n/10;
}
System.out.println("Reverse of digits="+rev);
}
}
```

5. Develop a java program to convert a binary number to its decimal equivalent.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
int num,bnum,dec=0,rem,n=0;
Scanner sc=new Scanner(System.in);
System.out.print("Enter a binary number : ");
num=sc.nextInt();
bnum=num;
while(num>0)
{
rem=num%10;
dec=dec+rem*(int)Math.pow(2,n);
n++;
num=num/10;
}
System.out.println("Reverse of digits="+dec);
}
}
```

6. Develop a java program to generate Fibonacci Series up to N terms, where value of N is entered by user.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
int n1=0,n2=1,n3,i,n;
Scanner sc=new Scanner(System.in);
System.out.print("How many terms you want in series? ");
n=sc.nextInt();
System.out.print(n1+" "+n2+" ");
for(i=1;i<=n-2;i++)
{
n3=n1+n2;
System.out.print(n3+" ");
n1=n2;
n2=n3;
}
}
}
```

7. Develop a java program to check whether the given number is prime or not.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
int n,i,c=0;
Scanner sc=new Scanner(System.in);
System.out.print("Enter a number to check prime or not : ");
n=sc.nextInt();
for(i=1;i<=n;i++)
{
if(n%i==0)
{
c++;
}
}
if(c==2)
{
System.out.println(n+" is prime");
}
else
{
System.out.println(n+" is not prime");
}
}
}
```

8. Develop a java program to print the series of prime number in given range.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
int l,u,i,j,c=0;
Scanner sc=new Scanner(System.in);
System.out.print("Enter lower limit : ");
l=sc.nextInt();
System.out.print("Enter upper limit : ");
u=sc.nextInt();
for(i=l;i<=u;i++)
{
c=0;
for(j=1;j<=i;j++)
{
if(i%j==0)
{
c++;
}
}
if(c==2)
{
System.out.print(i+" ");
}
}
}
}
```

Interview Questions: -

1. What is the purpose of looping statements?
   Ans: - If you have a block of code which you want to execute repeatedly up to given condition is true then you can use a loop control.

2. What is the default condition of for loop?
   Ans: - The default condition of for loop is true.

3. Inside "For Loop", initialization, condition & increment/decrement parts are optional or mandatory?
   Ans: - Inside for loop, initialization & condition & increment / decrement parts are optional.

4. What do you mean by transfer statements and what transfer statements are present in java?
   Ans: - The transfer statements are the control statements which transfer the program execution control to a specific statement. Java provides six language constructs for transferring control in a program:
   i. break
   ii. continue
   iii. return
   iv. try-catch-finally
   v. throw
   vi. assert

5. When will we get compilation error like "unreachable statement "?
   Ans: - An unreachable Statement is an error raised as part of compilation when the Java compiler detects code that is never executed as part of the execution of the program.

6. Is it possible to declare while loop without condition yes? If yes, what is the default condition and if no, what is the error?
   Ans: - No, it must have a valid boolean expression inside the () parentheses.

Multiple Choice Questions: -

1. Which of the following loops will execute the body of loop even when condition controlling the loop is initially false?
   a) do-while
   b) while
   c) for
   d) none of the mentioned

2. Which of these jump statements can skip processing the remainder of the code in its body for a particular iteration?
   a) break
   b) return
   c) exit
   d) continue

3. What is true about a break?
   a) Break stops the execution of entire program
   b) Break halts the execution and forces the control out of the loop
   c) Break forces the control out of the loop and starts the execution of next iteration
   d) Break halts the execution of the loop for certain time frame

4. What is true about do statement?
   a) do statement executes the code of a loop at least once
   b) do statement does not get execute if condition is not matched in the first iteration
   c) do statement checks the condition at the beginning of the loop
   d) do statement executes the code more than once always

5. Which of the following is not a valid jump statement?
   a) break
   b) goto
   c) continue
   d) return

Answer Key: -

1. a	2. d	3. b	4. a	5. b

Fill in the blanks: -

1. Java provides __________ basic looping control structures.
   Ans: - four

2. The __________ first executes the block of statements and then checks the condition.
   Ans: - do-while loop

3. The __________ statements cause the program control to be transferred to a specific location depending upon the outcome of the conditional expression.
   Ans: - selection

4. The for loop is also called an __________.
   Ans: - Entry controlled loop.

5. The __________ determines the increment or decrement of the loop control variable unless the test condition becomes false.
   Ans: - The step value

6. The __________ statements cause the program control to be transferred to a specific location depending upon the outcome of the conditional expression.
   Ans: - selection

Java Lecture – 04

(Concept of Array)

Long Answer Questions: -

1. What is an array? What is the need of array?

   Ans: - An Array is the collection of similar data types that means an array can store multiple values of similar data types.

   Need of Array: - Arrays are used when there is a need to use many variables of the same type. It can be defined as a sequence of objects which are of the same data type. It is used to store a collection of data, and it is more useful to think of an array as a collection of variables of the same type. Arrays can be declared and used. A programmer has to specify the types of elements and the number of elements that are required by an array.

   Declaration of Array: -

   datatype [] arrayname=new datatype[size];

   E.g.

   int [] x=new int[10];

   The above array can store 10 numbers of int type.

   Initialization of Array: -

   int [] x={10,20,30,40,50};

   The above array stores elements in following manner: -

   x[0]=10
   x[1]=20
   x[2]=30
   x[3]=40
   x[4]=50

2. How many types of array can be declared? How to take input from user for an array?

   Declaration of array: -

   i.) First way
   int [] x; //Declaration
   x=new int[5]; //Instantiation

   ii.) Second way
   int [] x=new int[5]; //Declaration and Instantiation

   Taking input from user for an array in Java: -

   Code Segment: -

```
int [] x=new int[5];
int i;
Scanner sc=new Scanner(System.in);
```

```
System.out.println("Enter five numbers");
for(i=0;i<5;i++)
{
x[i]=sc.nextInt();
}
```

3. How many types of array are there in Java? Explain it.
   Ans: -There are two types of arrays in Java, they are −
   Single dimensional array − A single dimensional array of Java is a normal array where the array contains sequential elements (of same type) −
   int[] myArray = {10, 20, 30, 40};

   Multi-dimensional array − A multi-dimensional array in Java is an array of arrays. A two dimensional array is an array of one dimensional arrays and a three dimensional array is an array of two dimensional arrays.

Short Answer Questions: -

1. What is the importance of array?
   Ans: - Importance of Array: -
   - Arrays represent multiple data items of the same type using a single name.
   - In arrays, the elements can be accessed randomly by using the index number.
   - Arrays allocate memory in contiguous memory locations for all its elements. Hence there is no chance of extra memory being allocated in case of arrays. This avoids memory overflow or shortage of memory in arrays.
   - Using arrays, other data structures like linked lists, stacks, queues, trees, graphs etc can be implemented.
   - Two-dimensional arrays are used to represent matrices.

2. What is initialization of array?
   Ans: - Initialization of array: - Initialization means to store the values into array with a declaration.
   E.g.
   int [] list={10,20,30,40,50};
   list[0]=10;
   list[1]=20;
   list[2]=30;
   list[3]=40;
   list[4]=50;

3. Describe memory allocation in array.

   Ans: - Memory Allocation in Java is the process in which the virtual memory sections are set aside in a program for storing the variables and instances of structures and classes. However, the memory isn't allocated to an object at declaration but only a reference is created. For the memory allocation of the object, new() method is used, so that the object is always allocated memory on the heap.

4. How to declare two dimensional array?

   Ans: -

   datatype [][] arrayname=new datatype[rows][columns];

   E.g.

   int [][] A=new int[3][3];

5. How to take input from user for two dimensional array?

   Ans: - Code Segment: -

```
int [][] A=new int [3][3];
int i,j;
Scanner sc=new Scanner(System.in);
for(i=0;i<3;i++)
{
        for(j=0;j<3;j++)
        {
                A[i][j]=sc.nextInt();
        }
}
```

6. What are the different ways of copying an array into another array?

   Ans: - You can copy one array from another in several ways −

   Copying element by element − One way is to create an empty array with the length of the original array and copy each element (in a loop).

   Using the clone() method − The clone() method of the class java.lang.Object accepts an object as a parameter, creates and returns a copy of it.

   Using the System.arraycopy() method − The copy() method of the System class accepts two arrays (along with other details) and copies the contents of one array to other.

Technical Tasks: -

1. Develop a Java program to find sum and average of ten numbers using array.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
int [] x=new int[10];
int i,sum=0;
double avg;
Scanner sc=new Scanner(System.in);
System.out.println("Enter ten numbers to the list");
for(i=0;i<10;i++)
{
x[i]=sc.nextInt();
sum=sum+x[i];
}
avg=(float)sum/10;
System.out.println("Sum="+sum);
System.out.println("Average="+avg);
}
}
```

2. Develop a Java program to take five names as input and display names in alphabetical order.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
String [] name=new String[5];
int i;
Scanner sc=new Scanner(System.in);
System.out.println("Enter five names");
for(i=0;i<5;i++)
{
name[i]=sc.nextLine();
}
Arrays.sort(name);
System.out.println("Names in alphabetical order");
for(String n:name)
{
```

```
System.out.println(n);
}
}
}
```

3. Develop a Java program to take ten numbers as input for an array AR. Now copy even numbers in array EAR and odd numbers in array OAR. Now display elements of EAR and OAR.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
int [] AR=new int[10];
int [] EAR=new int[10];
int [] OAR=new int[10];
int i,n1=0,n2=0;
Scanner sc=new Scanner(System.in);
System.out.println("Enter ten numbers to the list");
for(i=0;i<10;i++)
{
AR[i]=sc.nextInt();
}
for(i=0;i<10;i++)
{
if(AR[i]%2==0)
{
EAR[n1]=AR[i];
n1++;
}
else
{
OAR[n2]=AR[i];
n2++;
}
}
System.out.println("Even numbers");
for(i=0;i<n1;i++)
{
System.out.println(EAR[i]);
}
System.out.println("Odd numbers");
for(i=0;i<n2;i++)
{
System.out.println(OAR[i]);
}
}
}
```

4. Develop a Java program to search a number in array of ten numbers using linear search.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
int [] list=new int[10];
int i,f=0,item;
Scanner sc=new Scanner(System.in);
System.out.println("Enter ten numbers to the list");
for(i=0;i<10;i++)
{
list[i]=sc.nextInt();
}
System.out.print("Enter the number to be search : ");
item=sc.nextInt();
//Code for searching
for(i=0;i<10;i++)
{
if(list[i]==item)
{
f=1;
break;
}
}
if(f==1)
{
System.out.println("The number "+item+" is found at location "+(i+1));
}
else
{
System.out.println("The number is not found");
}
}
}
```

Interview Questions: -

1. Can you pass a negative number as an array size?
   Ans: - No, you can't pass a negative number as an array size because array size contains number of elements.

2. Can you change the size of the array once you define it or can you insert or delete the elements after creating an array?
   Ans: - No, you can't change the size of the array once you define it. You can insert or delete the elements after creating an array.

3. What is the difference between int[] a and int a[] ?
   Ans: - There is no such difference in between these two types of array declaration. It's just what you prefer to use, both are integer type arrays.

4. "int a[] = new int[3]{1, 2, 3}" – is it a legal way of defining the arrays in java?
   Ans: - This is invalid way to initialize an Array in Java. You cannot provide the size of the Array when you are declaring the elements in it.

5. What are jagged arrays in java? Give example.
   Ans: - A jagged array is an array of arrays such that member arrays can be of different sizes, i.e., we can create a 2-D array but with a variable number of columns in each row.

6. How do you check the equality of two arrays in java? OR How do you compare the two arrays in java?
   Ans: - The Arrays.equals() method can be used to check if two arrays are equal.

7. What is ArrayIndexOutOfBoundsException in Java? When it occurs?
   Ans: - ArrayIndexOutOfBoundsException is a type of unchecked exception in java. It occurs when you use an array index of more than its size.

8. What value does array elements get, if they are not initialized?
   Ans: - Everything in a Java program not explicitly set to something by the programmer, is initialized to a zero value.
   For references (anything that holds an object), it is null. For int/short/byte/long, it is 0. For Booleans, it is false.

9. What are the drawbacks of the arrays in Java?
   Ans: - Deleting or inserting − You cannot insert a new element at the middle of the array. In the same way you cannot delete elements from the middle of the array. You can only insert/delete from the end of the array.

Increasing size − You cannot increase the size of the arrays in Java, if you want to add new elements you need to create new array with extended size and assign to the array reference. This leaves the original object for garbage collection and thus wastage of memory occurs.

Storing Objects − You can store objects in an array but you cannot store objects of different types.

Processing Elements − Except some operations provided by the Array class, you cannot process the contents of an array.

Modifying elements − To delete or change the elements of an array you need to traverse throughout the array which increases the time complexity.

Multiple Choice Questions: -

1. Which of these operators is used to allocate memory to array variable in Java?
   a) malloc
   b) alloc
   c) new
   d) new malloc

2. Which of these is an incorrect array declaration?
   a) int arr[] = new int[5]
   b) int [] arr = new int[5]
   c) int arr[] = new int[5]
   d) int arr[] = int [5] new

3. Which of these is necessary to specify at the time of array initialization?
   a) Row
   b) Column
   c) Both Row and Column
   d) None of the mentioned

4. How to sort an array?
   a) Array.sort()
   b) Arrays.sort()
   c) Collection.sort()
   d) System.sort()

5. An array elements are always stored in ________ memory locations.
   a) Sequential
   b) Random

c) Sequential and Random
d) Binary search

Answer Key: -

1. c	2. d	3. a	4. b	5. a

Fill in the blanks: -

1. _______ is the collection of similar data types.
   Ans: - Array

2. ______ keyword is used to obtain memory of array.
   Ans: - New

3. __________________method is used to sort array elements in ascending order.
   Ans: - Arrays.sort()

4. Two dimensional array has two subscripts one for ________ another one for __________.
   Ans: - rows, columns

5. ____________ property is used to find the length of array.
   Ans: - Length

Java Lecture – 05

(String In Java)

Long Answer Questions: -

1. What is String in Java? How to take input from user for a string?
   Ans: - String: - Technically string is a sequence of characters. In Java, string is a class in java.lang package. Object of String class is used to store string.
   Taking input from user for string: -
   Scanner sc=new Scanner(System.in);
   System.out.print("Enter a string : ");
   String name=sc.nextLine();

2. What is the difference between String and StringBuffer class?
   Ans: - Strings, which are widely used in Java programming, are a sequence of characters. In Java programming language, strings are treated as objects. The Java platform provides the String class to create and manipulate strings.
   Whereas, StringBuffer class is a thread-safe, mutable sequence of characters.
   - A string buffer is like a String, but can be modified.
   - It contains some particular sequence of characters, but the length and content of the sequence can be changed through certain method calls.
   - They are safe for use by multiple threads.
   - Every string buffer has a capacity.

Short Answer Questions: -

1. What is the difference between
   a. String str="softpro";
   b. String str = new String("softpro")

   Ans: - Both expressions give you a String object, but there is a subtle difference between them. When you create a String object using the new() operator, it always creates a new object in heap memory.
   On the other hand, if you create an object using String literal syntax e.g. "softpro", it may return an existing object from String pool (a cache of String object in Perm gen space, which is now moved to heap space in recent Java release), if it already exists.

2. What is the difference between equals() and == operator?
   Ans: - Difference between equals() and == operator: -
   - The main difference between the .equals() method and == operator is that one is a method and the other is the operator.

- We can use == operators for reference comparison (address comparison) and equals() method for content comparison. In simple words, == checks if both objects point to the same memory location whereas equals() evaluates to the comparison of values in the objects.
-

3. What is the difference between length vs length()?
   Ans: -

length	**length()**
length is an attribute i.e. a data member of array.	length() is a member method of String class.
It gives the length of an array i.e. the number of elements stored in an array.	It gives the number of characters present in a string.

4. What is the difference between equals() and equalsIgnoreCase()?
   Ans: - equals() in Java is used to check for equality between two strings.
   equalsIgnoreCase() in Java to check for equality between two strings ignoring the case.

5. What is purpose of trim() method?
   Ans: - The trim() method in Java String is a built-in function that eliminates leading and trailing spaces.

Technical Tasks: -

1. Develop a Java program to take username as input and display name in upper case and lower case.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
String name;
Scanner sc=new Scanner(System.in);
System.out.print("Enter your name : ");
name=sc.nextLine();
System.out.println("Name in upper case="+name.toUpperCase());
System.out.println("Name in lower case="+name.toLowerCase());
}
}
```

2. Develop a Java program to compare two strings for equality.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
String str1,str2;
Scanner sc=new Scanner(System.in);
System.out.print("Enter first string : ");
str1=sc.nextLine();
System.out.print("Enter second string : ");
str2=sc.nextLine();
if(str1.equals(str2))
{
System.out.println("Both strings are equal");
}
else
{
System.out.println("Both strings are not equal");
}
}
}
```

3. Develop a Java program to count number of words in a sentence.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
String sen;
Scanner sc=new Scanner(System.in);
System.out.print("Enter a sentence : ");
sen=sc.nextLine();
String [] words=sen.split(" ");
System.out.println("Number of words="+words.length);
}
}
```

4. Develop a Java program to take a sentence as input, in sentence replace a word with another word.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
String sen,fw,rw;
Scanner sc=new Scanner(System.in);
System.out.print("Enter a sentence : ");
sen=sc.nextLine();
System.out.print("Find what? ");
fw=sc.nextLine();
System.out.print("Replace with : ");
rw=sc.nextLine();
System.out.println("Modified sentence="+sen.replace(fw,rw));
}
}
```

5. Develop a Java program to check given string is palindrome or not.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
String str,revstr="";
int i;
Scanner sc=new Scanner(System.in);
System.out.print("Enter a string : ");
str=sc.nextLine();
for(i=str.length()-1;i>=0;i--)
{
revstr=revstr+str.charAt(i);
}
if(str.equalsIgnoreCase(revstr))
{
System.out.println("String is palindrome");
}
else
{
System.out.println("String is not palindrome");
}
}
}
```

6. Develop a Java program to take username as input and display its short name. E.g. user has entered Ajay Pratap Singh then output should be A.P.Singh.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
String name;
int i;
Scanner sc=new Scanner(System.in);
System.out.print("Enter your full name : ");
name=sc.nextLine();
String [] shortname=name.split(" ");
System.out.print("Your shortname is : ");
for(i=0;i<shortname.length-1;i++)
{
System.out.print(shortname[i].charAt(0)+".");
}
System.out.print(shortname[shortname.length-1]);
}
}
```

Interview Questions: -

1. equals() method is present in which class?
   Ans: - equals() method is present in String class.

2. What is the purpose of equals() method in String class?
   Ans: - equals() method is used to compare two strings for equality.

3. String & StringBuffer & StringBuilder & StringTokenizer are present in which package name?
   Ans: -String & StringBuffer & StringBuilder & StringTokenizer are present in java.lang package.

4. What is the purpose of String class equals() &StringBuffer class equals()?
   Ans: - equals() method has been overridden in String class to check and match the content of two different Strings. The simple answer is that StringBuffer (and StringBuilder) do not override the base semantics of Object. equals() so equals on a StringBuffer will simply compare object references.

5. What is the difference between concat() method & append()?
   Ans: -
   1. Concat is used to add a String at the end of another String. Append adds a String or character sequence to StringBufffer.
   2. Concat creates a new String object whereas StringBuffer append doesn't.
   3. Append is more efficient than concat.

6. What is the difference between compareTo() vs eqauls()?

compareTo	equals
It compares two strings lexicographically.	It checks if contents of two strings are same or not.
The result is a negative, positive or zero integer value depending on whether the String object precedes, follows or is equal to the String argument.	The result is true if the contents are same otherwise it is false.

7. What is the purpose of contains() method?
   Ans: - contains() method searches the sequence of characters in the given string. It returns true if sequence of char values are found in this string otherwise returns false.

8. String is a final class or not?
   Ans: - A String is a final class & its immutable because it can't be changed but can be referred to another object.

Multiple Choice Questions: -

1. Which of these classes is superclass of String and StringBuffer class?
   a) java.util
   b) java.lang
   c) ArrayList
   d) None of the mentioned

2. Which of this method of class String is used to obtain a length of String object?
   a) get()
   b) Sizeof()
   c) lengthof()
   d) length()

3. Which of these methods of class String is used to extract a single character from a String object?
   a) CHARAT()
   b) chatat()
   c) charAt()
   d) ChatAt()

4. Which of these methods of class String is used to compare two String objects for their equality?
   a) equals()
   b) Equals()
   c) isequal()
   d) Isequal()

5. Which of these data type value is returned by equals() method of String class?
   a) char
   b) int
   c) boolean
   d) all of the mentioned

6. Which of these methods of class String is used to remove leading and trailing whitespaces?
   a) startsWith()
   b) trim()
   c) Trim()
   d) doTrim()

7. Which of this method of class String is used to extract a substring from a String object?
   a) substring()
   b) Substring()
   c) SubString()
   d) None of the mentioned

Answer Key: -

1. b	2. d	3. c	4. a
5. c	6. b	7. a	

Fill in the blanks Questions: -

1. String class is available in _____________ package.
   Ans: - java.lang

2. ___________ method of String class is used to find length of string.
   Ans: - length()

3. ___________ is return type of equals() method.
   Ans: - boolean

4. ___________ method of String class is used to remove white spaces of starting and ending of string.
   Ans: - trim()

Java Lecture – 06
(Method in Java)

Long Answer Questions: -

1. What is method in Java? What is the importance of method in Java?
   Ans: - Method is a named block of code which performs specific task.

   Importance of method: - If you have a block of code which is required at different locations of program, then you can create a method of that code and call it from desired locations. By using method you can avoid to write same code over and over.

   Creation of method: -
   ```
   <Method Modifiers> <Method Return Type> <Method_Name>(Parameters)
   {
           //code
   }
   ```

2. How many types of methods are there in java?
   Ans: - In Java programming language, there are two types of methods: -

   Static Method: - static methods are created by using static modifier. There is no need of object to call static methods.

   Non-static Method: - non-static methods are not created by using static modifier. These methods are call by creation of object.

3. What is the difference between static and non-static methods in Java?
   Ans: - A static method is a method that belongs to a class but it does not belong to an instance of that class and this method can be called without the instance or object of that class. Every method in java defaults to a non-static method without static keyword preceding it. Non-static methods can access any static method and static variable, without creating an instance of the object.

Short Answer Questions: -

1. How to create a method in java, write its syntax?
   Ans: -
   Creation of method: -
   ```
   <Method Modifiers> <Method Return Type> <Method_Name>(Parameters)
   {
           //code
   }
   ```

2. What is access modifier?
   Ans: - Access modifiers are keywords in Java that are used to set accessibility. An access modifier restricts the access of a class, constructor, data member and method in another class. Java language has four access modifiers to control access level for classes and its members.

3. Why the main() method is static in Java?
   Ans: - The main() method is static so that JVM can invoke it without instantiating the class. This also saves the unnecessary wastage of memory which would have been used by the object declared only for calling the main() method by the JVM.
   Void: It is a keyword and used to specify that a method doesn't return anything.

4. How to call a non-static method from main in Java?
   Ans: - Non-static method is called by creating the object of class from main in java.

5. What is Recursion? Explain with an example.
   Ans: - Recursion: - When a method calls itself, then it is called Recursion. Let's consider an example of factorial: -
   5!=5*4*3*2*1=120
   5!=5*4!
   .
   .
   n!=n*(n-1)!
   fact(n)=n*fact(n-1)

Technical Tasks: -

1. Develop a Java program to make a temperature convertor by creating user defined methods. In this program create two methods ctof() and ftoc().
   In ctof() method convert temperature from Centigrade to Fahrenheit and in ftoc() method convert temperature from Fahrenheit to Centigrade.

```
import java.util.*;
class Test
{
public double ctof(double c)
{
double f;
f=(9*c)/5+32;
return f;
}
public double ftoc(double f)
{
double c;
c=(f-32)*5/9;
return c;
}
public static void main(String [] args)
{
double c,f;
int ch;
Scanner sc=new Scanner(System.in);
Test t=new Test();
System.out.println("Enter 1 for c to f");
System.out.println("Enter 2 for f to c");
ch=sc.nextInt();
switch(ch)
{
case 1:
System.out.print("Enter temperature in c : ");
c=sc.nextDouble();
f=t.ctof(c);
System.out.println("Temperature in f="+f);
break;
case 2:
System.out.print("Enter temperature in f : ");
f=sc.nextDouble();
c=t.ftoc(f);
System.out.println("Temperature in c="+c);
break;
default:
System.out.println("Invalid choice");
break;
}
}
}
```

2. Develop a Java program to find factorial of given number using ‘Recursion’.

```
import java.util.*;
class Test
{
public int fact(int n)
{
if(n==0 || n==1)
{
return 1;
}
else
{
return n*fact(n-1);
}
}
public static void main(String [] args)
{
int n,f;
Scanner sc=new Scanner(System.in);
Test t=new Test();
System.out.print("Enter a number to find factorial : ");
n=sc.nextInt();
f=t.fact(n);
System.out.println("Factorial="+f);
}
}
```

3. Develop a Java program to find area and perimeter of rectangle using user-defined static methods.

```
import java.util.*;
class Test
{
public static int area(int l,int b)
{
return (l*b);
}
public static int perimeter(int l,int b)
{
return 2*(l+b);
}
public static void main(String [] args)
{
int l,b,a,p;
Scanner sc=new Scanner(System.in);
System.out.println("Enter length and breadth of rectangle");
l=sc.nextInt();
b=sc.nextInt();
a=area(l,b);
p=perimeter(l,b);
System.out.println("Area of rectangle="+a);
System.out.println("Perimeter of rectangle="+p);
}
}
```

Interview Questions: -

1. What is method in Java?
   Method: - Method is a named block of code which perform specific task. If you have a block of code which is required in different locations of program, then you can create a method of that code and call it from desired locations. By using method, you can avoid to write same code over and over.
2. How many types of methods are there in java?
   Ans: - In Java Programming language, there are two types of methods: Static method and Non-static method.

3. What do you mean by method signature?
   Ans: - In Java, a method signature is part of the method declaration. It's the combination of the method name and the parameter list.

4. Can you overload main() method?
   Ans: - Yes, we can overload the main method in java but JVM only calls the original main method, it will never call our overloaded main method.

5. What is static method?
   Ans: - Static Method is declared by using static keyword. There is no need of object to call static method.

6. What is non-static method?
   Ans: - Non-static method is not declared by using static keyword. Non-static methods are also called instant methods. Non-static methods can be called by using object only.

Multiple Choice Questions: -

1. What is the return type of a method that does not return any value?
   a) int
   b) float
   c) void
   d) double
2. Which method can be defined only once in a program?
   a) main method
   b) finalize method
   c) static method
   d) private method
3. Which of these is the method which is executed first before execution of any other thing takes place in a program?
   a) main method
   b) finalize method
   c) static method
   d) private method
4. What is the process of defining more than one method in a class differentiated by method signature?
   a) Function overriding
   b) Function overloading
   c) Function doubling
   d) None of the mentioned
5. Which of the following is a method having same name as that of it's class?
   a) finalize
   b) delete
   c) class
   d) constructor

Answer Key: -

1. c	2. a	3. c	4. b	5. d

Fill in the blanks Questions: -

1. ___________ keyword is used to create a static method.
   Ans: - static

2. When a method calls itself, it is called _____________.
   Ans: - Recursion

3. Static method is also called _____________ method.
   Ans: - class

4. Non-static methods are also called ______________methods.
   Ans: - Instance

Java Lecture – 7

(Object Oriented Programming)

Long Answer Questions: -

1. What is Object Oriented Programming System? Describe its pillars.
   Ans: - OOPS stands for Object Oriented Programming System, it is a mechanism of software development. The OOPS has following pillars: -

   i.) **Abstraction: -** Abstraction is a mechanism to show essential functionalities and hide all other functionalities of an object.
   ii.) **Encapsulation: -** Encapsulation is a mechanism to wrap properties and functionalities in a single unit which is called object.
   iii.) **Inheritance: -** In Inheritance, you can create a new product by using existing product.
   iv.) **Polymorphism: -** The term polymorphism means one thing many forms.

2. What is Class and Object?
   Ans: -
   Class: - Class is the collection of variables and methods. Class is also called as blue print of object. We always create the object of a class.
   Object: - Object is an entity which has its states and behaviors.

3. What is constructor? Write its importance.
   Constructor: - Constructor is a special method which is used to initialize final variables. The Constructor has following properties: -

   i.) Constructor name is same as class name.
   ii.) Constructor has no return type.
   iii.) Constructor is called automatically as soon as an object is created.

Short Answer Questions: -

1. What is abstraction?
   Ans: - Abstraction is a mechanism to show essential functionalities and hide all other functionalities of an object.

2. What is encapsulation?
   Ans: - Encapsulation is a mechanism to wrap properties and functionalities in a single unit. That single unit is called object.

3. What is inheritance?
   Ans: - In Inheritance, you can create a new product by using existing product.

4. What is polymorphism?
   Ans: - The term polymorphism means one thing many forms.

5. What is class?
   Ans: - Class is the collection of variables and methods; class is also called as blueprint of object. We always create the object of a class.

6. What is an object?
   Ans: - Object: - Object is a real world entity, which has its states and behavior.

7. What is constructor?
   Ans: - Constructor is a special method which is used to initialize final variables. The constructor has following properties: -
   i.) Constructor name is same as class name.
   ii.) Constructor has no return type.
   iii.) Constructor is called automatically as soon as an object is created.

8. How many types of constructors are there?
   Ans: - In Java there are two types of constructor: -
   i.) Default Constructor
   ii.) Parameterized Constructor

Technical Tasks: -

1. Create a class named TestClass with a method sayHello(). In sayHello() method display "Hello World" message. Now call sayHello() method by creating anonymous object.

```
import java.util.*;
class MyClass
{
public void sayHello()
{
System.out.println("Hello World");
}
}
class Test
{
public static void main(String [] args)
{
new MyClass().sayHello();
}
}
```

2. Create a class named Employee. In Employee class, take three private data members empid, empname and salary. Now create a public method setEmployee() to initialize private data members. And also create a method getEmployee() to display employee's details. Now test the class employee.

```
import java.util.*;
class Employee
{
private int empid;
private String empname;
private double salary;
public void setEmployee(int eid,String ename,double sal)
{
empid=eid;
empname=ename;
salary=sal;
}
public void getEmployee()
{
System.out.println("Employee Id : "+empid);
System.out.println("Employee Name : "+empname);
System.out.println("Employee Salary : "+salary);
}
}
class Test
{
public static void main(String [] args)
{
Employee emp=new Employee();
emp.setEmployee(1001,"Brijesh Mishra",40000.0);
emp.getEmployee();
}
}
```

3. Develop a class named Rectangle with final data members length and width. Make a parameterized Constructor to initialize data members. Now make two methods rectarea() and rectperi() to calculate area and perimeter. Test the class Rectangle.

```
import java.util.*;
class Rectangle
{
final int l,b;
Rectangle(int x,int y)
{
l=x;
b=y;
}
public int rectArea()
{
return (l*b);
}
public int rectPeri()
{
return 2*(l+b);
}
}
class Test
{
public static void main(String [] args)
{
int x,y;
Scanner sc=new Scanner(System.in);
System.out.println("Enter length and breadth of rectangle");
x=sc.nextInt();
y=sc.nextInt();
Rectangle r=new Rectangle(x,y);
int a=r.rectArea();
int p=r.rectPeri();
System.out.println("Area of rectangle="+a);
System.out.println("Perimeter of rectangle="+p);
}
}
```

4. Develop a class named Interest with private data members' p, n & r of double type. Make a parameterized Constructor to initialize data members. Now make a method simpleInterest() to calculate simple interest. Now test the class Interest.

```
import java.util.*;
class Interest
{
private double p,n,r;
Interest(double p,double n,double r)
{
this.p=p;
this.n=n;
this.r=r;
}
public double simpleInterest()
{
return (p*n*r)/100;
}
}
class Test
{
public static void main(String [] args)
{
double p,n,r;
Scanner sc=new Scanner(System.in);
System.out.print("Enter principle amount : ");
p=sc.nextDouble();
System.out.print("Enter time in years : ");
n=sc.nextDouble();
System.out.print("Enter rate : ");
r=sc.nextDouble();
Interest I=new Interest(p,n,r);
double si=I.simpleInterest();
System.out.println("Simple Interest="+si);
}
}
```

Interview Questions: -

1. What are the main building blocks of OOPS?
   Ans: - Building blocks of OOPS are given below: -
   i.) Abstraction
   ii.) Encapsulation
   iii.) Inheritance
   iv.) Polymorphism

2. When do we create anonymous object? What is the advantage of anonymous object creation?
   Ans: - When you require an object only one time then you can create anonymous object. The main advantage of anonymous object is that it automatically gets destroyed after its use.

3. What is the root class for all java classes?
   Ans: - Object is the root class for all java classes.

4. What is constructor in java? Why do we use constructor.
   Ans: - Constructor is a special method which is used to initialize final data members. The properties of constructor are given below: -
   i.) Constructor name is same as class name.
   ii.) Constructor has no return type.
   iii.) Constructor is called automatically as soon as an object is created.

5. How many types of constructor are there in Java?
   Ans: - In Java, constructor is of two types: -
   i.) Default constructor
   ii.) Parameterized constructor

6. What do you mean by constructor overloading?
   Ans: - If you create many constructors in a single class with different parameters then it is called constructor overloading.

7. What do you mean by abstraction?
   Ans: - Abstraction is a mechanism to show only essential functionalities and hide all other functionalities of an object.

8. What is the difference between normal method and abstract method?
   Ans: - Abstract method contains only method declaration and no definition whereas normal method contains method definition.

Multiple Choice Questions: -

1. Which of the following is not an OOPS concept in Java?
   a) Inheritance
   b) Encapsulation
   c) Polymorphism
   d) Compilation

2. Which concept of Java is a way of converting real world objects in terms of class?
   a) Polymorphism
   b) Encapsulation
   c) Abstraction
   d) Inheritance

3. Which concept of Java is achieved by combining methods and attribute into a class?
   a) Encapsulation
   b) Inheritance

c) Polymorphism
d) Abstraction

4. Which of these keywords is used to make a class?
a) class
b) struct
c) int
d) None of the mentioned

5. Which of the following is a valid declaration of an object of class Box?
a) Box obj = new Box();
b) Box obj = new Box;
c) obj = new Box();
d) new Box obj;

6. Which of these operators is used to allocate memory for an object?
a) malloc
b) alloc
c) new
d) give

7. What would be the behavior if the constructor has a return type?
a) Compilation error
b) Runtime error
c) Compilation and runs successfully
d) Only String return type is allowed

8. What is true about constructor?
a) It can contain return type
b) It can take any number of parameters
c) It can have any non-access modifiers
d) Constructor cannot throw an exception

9. What would be the behavior if this() and super() is used in a method?
a) Runtime error
b) Throws exception
c) compile time error
d) Runs successfully

10. Which keyword is used by the method to refer to the object that invoked it?
a) import
b) catch

c) abstract
d) this

Answer Key: -

1	D	6	C
2	C	7	A
3	A	8	B
4	A	9	C
5	A	10	D

Fill in the blanks: -

1. Hiding all functionalities and showing essential functionalities is called____________________.
   Ans: - Abstraction

2. Class is known as blueprint of ____________.
   Ans: - Object

3. Object is container of __________________ and ____________________.
   Ans: - Properties and Functionalities.

4. _______________ is used to construct the object.
   Ans: - Constructor

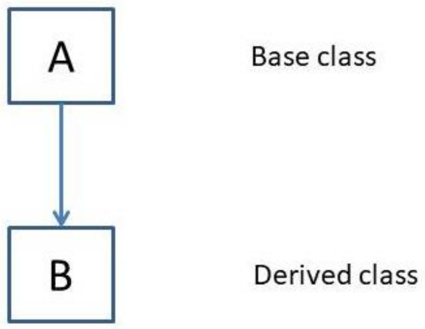
5. Creation of new product by using existing product is called _________________.
   Ans: - Inheritance

Java Lecture – 08

(Inheritance in Java)

Long Answer Questions: -

1. What is Inheritance in Java? What is its importance?
   Ans: - Inheritance: - Inheritance is a feature of object oriented programming. In Inheritance, you can create a new class by using existing class, existing class is called base class and new created class is called derived class. The main importance of inheritance is by using inheritance we can achieve 'Reusability'.



Syntax of Inheritance: -

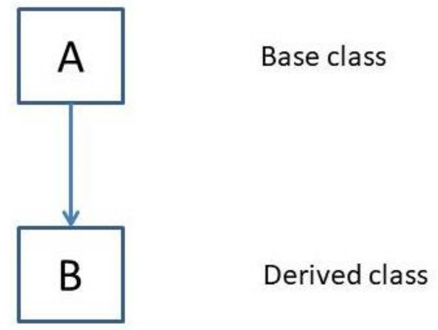
```
class A
{
//Variables and methods
}
class B extends A
{
//Variables and methods
}
```

2. How many types of Inheritance are there in Java?
   Ans: - Java programming language supports three types of inheritance: -

   i.) Single Inheritance
   ii.) Hierarchical Inheritance
   iii.) Multi-level Inheritance

Single Inheritance: -

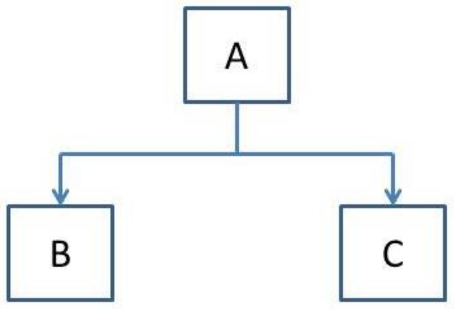
In Single Inheritance, there is a single base class and single derived class.



Syntax of Single Inheritance: -

```
class A
{
//Variables and Methods
}
class B extends A
{
//Variables and Methods
}
```

Hierarchical Inheritance: - In Hierarchical Inheritance, there is a single base class and multiple derived classes.

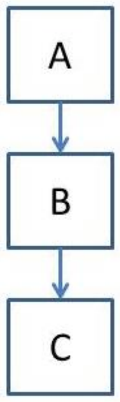


Hierarchical Inheritance

Syntax of Hierarchical Inheritance: -

```
class A
{
//Variables and methods
}
class B extends A
{
//Variables and Methods
}
class C extends A
{
//variables and Methods
}
```

Multi-level Inheritance: -



Multi-level Inheritance

Syntax of Multi-level Inheritance: -

```
class A
{
//Variables and Methods
}
class B extends A
{
//Variables and Methods
}
class C extends B
{
//Variables and Methods
}
```

Short Answer Questions: -

1. Write syntax to implement inheritance.
   Ans: - Syntax to implement inheritance is given below: -

```
class A
{
//Variables and Methods
}
class B extends A
{
//Variables and Methods
}
```

2. Why multiple inheritance is not supported in java?
   Ans: - The reason behind this is to prevent ambiguity. Consider a case where class B extends class A and Class C and both class A and C have the same method display(). Now java compiler cannot decide, which display method it should inherit. To prevent such situation, multiple inheritances is not allowed in java.

3. What is Reusability?
   Ans: - Reusability: As the name specifies, reusability is a mechanism which facilitates you to reuse the fields and methods of the existing class when you create a new class. You can use the same fields and methods already defined in the previous class.

4. Write importance of inheritance.
   Ans: - Inheritance is used to achieve 'Reusability'.

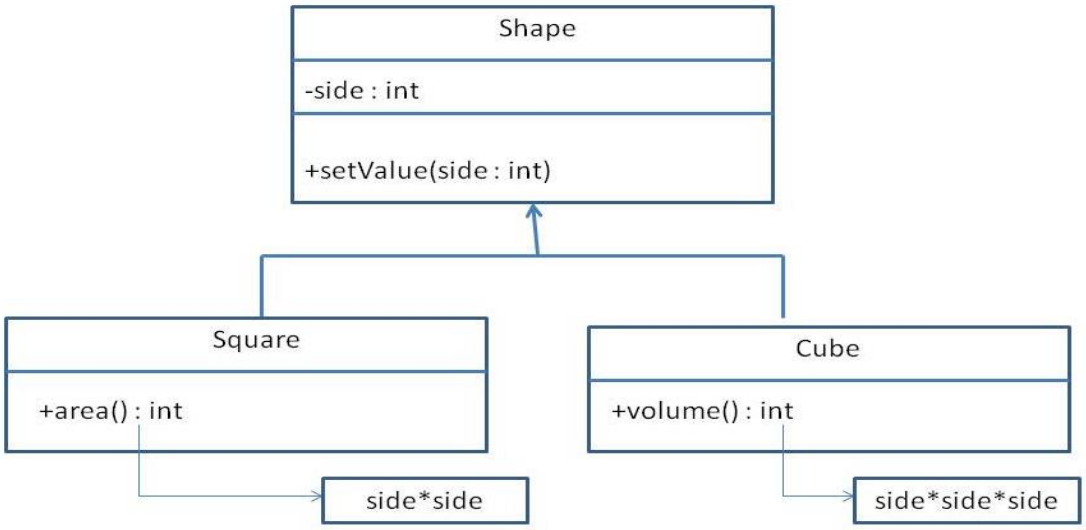
5. Which keyword is used to implement inheritance?
   Ans: - extends keyword is used to implement inheritance.

Technical Tasks: -

1. Develop a program in java to create a class Rundog. In Rundog class, make a method bark(), in bark() method display the rundog name and voice. By extending Rundog class create a new class named Bulldog. In Bulldog class, make a method grawl(), in grawl() method display bulldog name and voice.

```
class Rundog
{
public void bark()
{
System.out.println("Sheru....................");
System.out.println("Bho.........bho..........");
}
}
class Bulldog extends Rundog
{
public void growl()
{
System.out.println("Tommy....................");
System.out.println("Gurr.......gurr............");
}
}
class InDemo1
{
public static void main(String [] args)
{
Bulldog dog=new Bulldog();
dog.bark();
dog.growl();
}
}
```

2. Create the classes as following structure: -

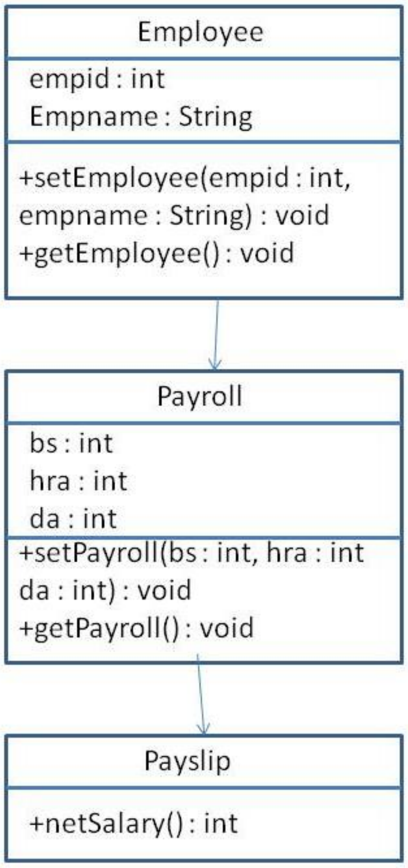


Now Test the classes.

```
import java.util.*;
class Shape
{
int side; //Instance Variable
void setValue(int s)
{
```

```
side=s;
}
}
class Square extends Shape
{
int area()
{
return (side*side);
}
}
class Cube extends Shape
{
int volume()
{
return (side*side*side);
}
}
class InDemo2
{
public static void main(String [] args)
{
int s,a,v;
Scanner sc=new Scanner(System.in);
Square sq=new Square();
System.out.print("Enter side of square : ");
s=sc.nextInt();
sq.setValue(s);
a=sq.area();
System.out.println("Area of square="+a);
Cube cu=new Cube();
System.out.print("Enter side of cube : ");
s=sc.nextInt();
cu.setValue(s);
v=cu.volume();
System.out.println("Volume of cube="+v);
}
}
```

3. Develop a program in java by using following class diagram: -



```
class Employee
{
int empid;
String empname;
void setEmployee(int eid, String ename)
{
empid=eid;
empname=ename;
}
void getEmployee()
{
System.out.println("Employee Id="+empid);
System.out.println("Employee Name="+empname);
}
}
class Payroll extends Employee
{
int bs, hra,da;
void setPayroll(int b,int h,int d)
{
bs=b;
hra=h;
da=d;
}
void getPayroll()
{
System.out.println("Basic Salary="+bs);
System.out.println("House Rent Allownces="+hra);
System.out.println("Dearness Allownces="+da);
}
}
class Payslip extends Payroll
{
void netSalary()
{
System.out.println("Net Salary="+(bs+hra+da));
}
}
class InDemo3
{
public static void main(String [] args)
{
Payslip ps=new Payslip();
ps.setEmployee(1001,"Rajat Singh");
ps.setPayroll(35000,10000,15000);
System.out.println("*********PAY SLIP**********");
ps.getEmployee();
ps.getPayroll();
ps.netSalary();
}
}
```

Interview Questions: -

1. What do you mean by inheritance?
   Ans: - Inheritance is a feature of object oriented programming. In Inheritance, you can create a new class by using existing class. The existing class is called base class and new created class is called derived class.

2. How to achieve inheritance concept and inheritance is also known as?
   Ans: - Inheritance is implemented by using extends keyword. Inheritance is also called 'Reusability'.

3. How many types of inheritance are there in java and how many types of inheritance are not supported by Java?
   Ans: - There are three types of inheritance which are supported in Java: -
   i.) Single Inheritance
   ii.) Hierarchical Inheritance
   iii.) Multi-level Inheritance

   Multiple Inheritance is not supported in Java.

4. What is the purpose of "extends" keyword?
   Ans: - The "extends" keyword is used to implement inheritance.

5. What is the difference between child class and parent class?
   Ans: - In Inheritance, existing class is called parent class and new created class is called child class.

6. What is the root class for all Java classes?
   Ans: - Object is the root class for all Java classes.

7. How to prevent inheritance concept?
   Ans: - By using private keyword, you can prevent inheritance concept.

8. How to call super class constructors?
   Ans: - Constructor of super class is called by using super keyword.

9. Is it possible to use both super and this keyword inside the method?
   Ans: - Both this() and super() cannot be used together in constructor.

10. One class is able to extends how many classes at a time?
    Ans: - One class is able to extends only one class at a time.

Multiple Choice Questions: -

1. Which of this keyword must be used to inherit a class?
a) super
b) this
c) extent
d) extends

2. Which of these is correct way of inheriting class A by class B?
a) class B + class A {}
b) class B inherits class A {}

c) class B extends A {}
d) class B extends class A {}

3. What is not a type of inheritance?
a) Single inheritance
b) Double inheritance
c) Hierarchical inheritance
d) Multiple inheritance

4. Using which of the following, multiple inheritance in Java can be implemented?
a) Interfaces
b) Multithreading
c) Protected methods
d) Private methods

5. All classes in Java are inherited from which class?
a) java.lang.class
b) java.class.inherited
c) java.class.object
d) java.lang.Object

6. In order to restrict a variable of a class from inheriting to subclass, how variable should be declared?
a) protected
b) private
c) public
d) static

Answer Key: -

1. d	2. c	3. b	4. a	5. d	6. b

Fill in the blanks Questions: -

1. ___________ keyword is used to perform inheritance.
Ans: - extends.

2. ______________________________ is not supported in Java.
Ans: - Multiple Inheritance

3. Single base class and multiple derived classes is called _________________ inheritance.
Ans: - Hierarchical

4. Concept of inheritance is also called _________________ of code.
Ans: - Reusability

Java Lecture – 09

(Polymorphism in Java)

Long Answer Questions: -

1. What is Polymorphism? Write its importance.

Polymorphism: - The term "Polymorphism" means "One Thing Many Forms".

There are two types of Polymorphism in Java: -

i. Compile Time Polymorphism (Overloading)
ii. Run Time Polymorphism (Overriding)

i. Compile time polymorphism [Overloading]: -

1) If java class allows two methods with same name but different number of arguments such type of methods are called overloaded methods.

2) We can overload the methods in two ways in java language

a. By passing different number of arguments to the same methods.

void m1(int a){}
void m1(int a,int b){}

b. Provide the same number of arguments with different data types.

void m1(int a){}
void m1(char ch){}

3) If we want achieve overloading concept one class is enough.

4) It is possible to overload any number of methods in single java class.

Method overloading: - In Java programming language you can give same name to multiple methods but their arguments should be different. Based on method arguments it is decided at compilation time that which method call from where. It is called method overloading.

Run Time Polymorphism or Method Overriding: Re-writing of base class method into derived class is called method overriding.

2. Write the difference between method overloading and method overriding.

Ans: - Difference between method overloading and method overriding is given below: -

Method Overloading	**Method Overriding**
Method overloading is used *to increase the readability* of the program.	Method overriding is used *to provide the specific implementation* of the method that is already provided by its super class.
Method overloading is performed *within class.*	Method overriding occurs *in two classes* that have IS-A (inheritance) relationship.

In case of method overloading, *parameter must be different*.	In case of method overriding, *parameter must be same*.
Method overloading is the example of *compile time polymorphism.*	Method overriding is the example of *run time polymorphism.*
In Java, method overloading can't be performed by changing return type of the method only. *Return type can be same or different* in method overloading. But you must have to change the parameter.	*Return type must be same or covariant* in method overriding.

3. What is a constructor? What is constructor overloading?
   Ans: - Constructor: - Constructor is a special method which is used to initialize final variables.

   Constructor has following properties: -
   i.) Constructor name is same class name.
   ii.) Constructor has no return type.
   iii.) Constructor is called automatically as soon as the object is created.

   Constructor overloading: - The class contains more than one constructor with same name but different arguments is called constructor overloading.

<u>Short Answer Questions: -</u>

1. What is Polymorphism?
   Ans: - The term Polymorphism means one thing many forms. In Java there are two types of Polymorphism: -
   i.) Compile time polymorphism
   ii.) Run time polymorphism

2. What is method overloading?
   Ans: - Method Overloading: - In Java, we can create multiple methods with same name but their parameters are different, this is called method overloading.

3. What is method overriding?
   Ans: - Method Overriding: - The re-writing of base class method into derived class is called method overriding.

4. What is constructor overloading?
   Ans: - The class contains more than one constructor with same name but different arguments is called constructor overloading.

Technical Tasks: -

1. Create a class with name Figure, make three method with same name area (method overloading). In first method find the area of square, second method find the area of circle and third method find the area of rectangle. Now test the class Figure.

```
import java.util.*;
class Figure
{
public int area(int s) //Area of square
{
return (s*s);
}
public int area(int l,int b) //Area of rectangle
{
return (l*b);
}
public double area(double r) //Area of circle
{
return (3.14*r*r);
}
}
class OverloadingDemo
{
public static void main(String [] args)
{
int s,l,b,a1,a2;
double r,a3;
Scanner sc=new Scanner(System.in);
Figure fig=new Figure();
System.out.print("Enter side of square : ");
s=sc.nextInt();
System.out.print("Enter length and breadth of rectangle : ");
l=sc.nextInt();
b=sc.nextInt();
System.out.print("Enter radius of circle : ");
r=sc.nextDouble();
a1=fig.area(s);
a2=fig.area(l,b);
a3=fig.area(r);
System.out.println("Area of square="+a1);
System.out.println("Area of rectangle="+a2);
System.out.println("Area of circle="+a3);
}
}
```

2. Develop a program in Java to demonstrate the concept of method overriding.

```
class A
{
public void m1()
{
System.out.println("m1 of A");
}
public void m2()
{
System.out.println("m2 of A");
}
}
class B extends A
{
 public void m1()
 {
 System.out.println("m1 of B");
 }
 public void m3()
 {
 System.out.println("m3 of B");
 }
 }
 class OverridingDemo
 {
 public static void main(String [] args)
 {
 A a1=new A();
 a1.m1();
 a1.m2();
 B b1=new B();
 b1.m1();
 b1.m2();
 b1.m3();
 }
 }
```

3. Develop a program in Java to create a class Bank. In Bank class, create a method interest() which return the bank interest as 0. Now create two classes Sbi and Pnb, these classes inherits the Bank class. In Sbi and Pnb classes, override the interest() method and return the value of interest as per bank norms. Now test the classes.

```
import java.util.*;
class Bank
{
public double interest()
{
return 0;
}
}
class Sbi extends Bank
{
public double interest()
{
return 7.5;
}
}
class Pnb extends Bank
{
public double interest()
{
return 7.0;
}
}
class Test
{
public static void main(String [] args)
{
Sbi sb=new Sbi();
System.out.println("Interest rate of sbi="+sb.interest());
Pnb pb=new Pnb();
System.out.println("Interest rate of Pnb : "+pb.interest());
}
}
```

Interview Questions: -

1. What do you mean by polymorphism?

   Polymorphism: - The term Polymorphism means one thing many forms. In Java there are two types of Polymorphism.

   i.) Compile Time Polymorphism
   ii.) Run Time Polymorphism

2. What do you mean by method overloading and method overriding?
   Method Overloading: - In Java, you can create multiple methods with same name but their parameters are different, this is called method overloading.
   Method Overriding: - The re-writing of base class method into derived class is called method overriding.

3. How many types of overloading are there in Java?
   In Java there are two types of overloading: -
   i.) Method Overloading
   ii.) Constructor Overloading.

4. What do you mean by constructor overloading?
   Constructor Overloading: - The class contains more than one constructors with same name but different arguments is called constructor overloading.

5. What are rules are to be followed while performing method overriding?
   Rules for performing method overriding: -
   i.) Class must be inherited.
   ii.) Base class method name and derived class method name must be same.
   iii.) Base class method parameters and derived class method parameters must be same.
   iv.) Base class method return type and derived class method return type can be same.
   v.) Base class method modifier and derived class method modifier can be same or derived class method modifier should be low precedence.

6. What is the purpose of final modifier in Java?
   Ans: - Java final keyword is a non-access specifier that is used to restrict a class, variable, and method. If we initialize a variable with the final keyword, then we cannot modify its value. If we declare a method as final, then it cannot be overridden by any subclasses.

<u>Multiple Choice Questions: -</u>

1. What is the process of defining two or more methods within same class that have same name but different parameters declaration?
   a) method overloading
   b) method overriding
   c) method hiding
   d) None of the mentioned above

2. Which of these can be overloaded?
   a) Methods

b) Constructors
c) All of the mentioned
d) None of the mentioned

3. Which of this keyword can be used in a subclass to call the constructor of superclass?
a) super
b) this
c) extent
d) extends

4. What is the process of defining a method in a subclass having same name & type signature as a method in its superclass?
a) Method overloading
b) Method overriding
c) Method hiding
d) None of the mentioned

5. Which of these is correct way of calling a constructor having no parameters, of superclass A by subclass B?
a) super(void);
b) superclass.();
c) super.A();
d) super();

Answer Key: -

1. a	2. c	3. a	4. b	6. d

Fill in the blanks Questions: -

1. The term _______________ means one thing many forms.
Ans: - Polymorphism

2. The _____________ name is same as class name.
Ans: - Constructor

3. Method name same, parameters are different is called _______________________.
Ans: - Method Overloading.

4. Re-writing of base class method into derived class is called ______________________.
Ans: - Method Overriding

5. _____________________ is applied in same class methods.
Ans: - Method Overloading

Java Lecture – 10

(Interface in Java)

Long Answer Questions: -

1. What is interface? What are the usages of interface in Java?

   Ans: - Interfaces: -The Interface is the container of abstract methods and public static final variables. The key points of interface are given below: -

   1. Interface is also one of the types of class which contains only abstract methods. Interfaces are not alternative for abstract class but it is extension for abstract classes.

   2. The abstract class contains at least one abstract method but the interface contains only abstract methods.

   3. For the interfaces the compiler will generate .class files.

   4. Interfaces give the information about the functionalities but do not provide the information about internal implementation.

   5. It is possible to declare any number of interfaces inside a source file. We declare the interfaces by using interface keyword.

   Syntax: -

   ```
   interface interface-name
   {
   //abstract methods
   //public static final variables
   }
   ```

2. What are the differences between interface, abstract class and class?

   Ans: - Interface, Abstract class and class, A discussion: -

   Interface: - If you have requirements, but you don't know about implementation. Then you can use interface, because the interface contain 100% of unimplemented (abstract) methods. The interface is used to achieve full abstraction.

   - The interface can extend another interface.
   - The interface can extend multiple interfaces.
   - We can't create the object of interface.

   Abstract class: - If you have requirements, you know about implementation, but not complete implementation then you can use abstract class, because abstract class contains 0-100% of implemented and unimplemented methods.

- The abstract class can implement interface.
- The abstract class can extend another abstract class.
- We can't create the object of abstract class.

Class: - If you have requirements, you know about complete implementation then you can use class, because a class contains 100% implemented methods.

- The class can implement an interface.
- The class can extend abstract class.
- The class can extend another class.
- We can create the object of class only.

Short Answer Questions: -

1. What is interface?
   Ans: - The Interface is the container of abstract methods and public static final variables.

2. What is abstraction?
   Ans: - Abstraction is a mechanism to show essential functionalities and hide all other functionalities of object.

3. How you can achieve abstraction using interface?
   Ans: - Since interface contains abstract methods and public static final variables therefore interface is used to achieve full abstraction.

4. What is the difference between normal method and abstract method?
   Ans: - Normal method contains method definition whereas abstract method contains only method declaration.

5. What is the difference between normal class and abstract class?
   Ans: - Normal class contain all implemented methods whereas, abstract class contain abstract methods and implemented methods both.

6. How to prevent object creation in Java?
   Ans: - In java we can avoid object creation in 2 ways :
   - Making the class as abstract, so we can avoid unnecessary object creation with in the same class and another class.
   - Making the constructor as private (Singleton design pattern), so we can avoid object creation in another class but we can create object in parent class.

Technical Tasks: -

1. Develop an interface named printable. In printable interface there is an abstract method print. Now implement printable interface and give the definition of print() method. In print() method give "Hello World" message.

```
import java.util.*;
interface printable
{
void print();
}
class Test implements printable
{
public void print()
{
System.out.println("Hello World");
}
public static void main(String [] args)
{
Test t=new Test();
t.print();
}
}
```

2. Develop an interface named Drawable. In Drawable interface create an abstract method named draw(). Now implement Drawable interface in Rectangle class and give definition of draw() method. And also implement Drawable interface in Circle class and give definition of draw() method. Now test the class Rectangle and Circle.

```
import java.util.*;
interface Drawable
{
void draw();
}
class Rectangle implements Drawable
{
public void draw()
{
System.out.println("This is rectangle");
}
}
class Circle implements Drawable
{
public void draw()
{
```

```
System.out.println("This is circle");
}
}
class Test
{
public static void main(String [] args)
{
Rectangle r=new Rectangle();
Circle c=new Circle();
r.draw();
c.draw();
}
}
```

3. Develop an interface named Bank. In Bank interface there is an abstract method rateOfInterest(). Now implement Bank interface to Sbi and Pnb classes and override the method rateOfInterest(). Now test the classes Sbi and Pnb.

```
import java.util.*;
interface Bank
{
double rateOfInterest();
}
class Sbi implements Bank
{
public double rateOfInterest()
{
return 7.5;
}
}
class Pnb implements Bank
{
public double rateOfInterest()
{
return 7.0;
}
}
class Test
{
public static void main(String [] args)
{
Sbi sb=new Sbi();
Pnb pb=new Pnb();
System.out.println("Interest rate of sbi : "+sb.rateOfInterest());
System.out.println("Interest rate of pnb : "+pb.rateOfInterest());
}
}
```

4. Develop two interfaces Printable and Showable. In Printable interface create an abstract method print() and in Showable interface create an abstract method show(). Now implement both interface in a class named TestMultipleInterface and override both methods print() and show(). Now test the class TestMultipleInterface.

```
import java.util.*;
interface Printable
{
void print();
}
interface Showable
{
void show();
}
class TestMultipleInterface implements Printable, Showable
{
public void print()
{
System.out.println("This is print() method");
}
public void show()
{
System.out.println("This is show() method");
}
}
class Test
{
public static void main(String [] args)
{
TestMultipleInterface tmi=new TestMultipleInterface();
tmi.print();
tmi.show();
}
}
```

Interview Questions: -

1. What do you mean by method hiding?

   Ans: - Method hiding can be defined as, "if a subclass defines a static method with the same signature as a static method in the super class, in such a case, the method in the subclass hides the one in the super class." The mechanism is known as method hiding. It happens because static methods are resolved at compile time.

2. What is interface?
   Ans: - Interface: - Interface is a collection of abstract methods and public static final variables.

3. What is abstract class?
   Ans: - Abstract class is a collection of abstract methods and implemented methods both.

4. What is class?
   Ans: - Class is the collection of variables and methods.

5. What is difference between interface and abstract class?
   Ans: - Interface contains abstract methods only whereas abstract class contains abstract methods and implemented methods both.

6. How many types of classes are there in Java?
   Ans: - There are seven types of classes in Java: -
   i.) static class
   ii.) final class
   iii.) abstract class
   iv.) concrete class
   v.) singleton class
   vi.) pojo class
   vii.) inner class

7. Normal class is also known as?
   Ans: - Normal class is also known as concrete class.

8. What is the difference between normal method and abstract method?
   Ans: - Normal method contains method definition whereas abstract method contains only method declaration.

9. What is the difference between normal class and abstract class?
   Ans: - Abstract class contains abstract methods and implemented methods both whereas normal class contains implemented methods only.

10. Is it possible to create an object for abstract class?
    Ans: - No, we cannot create object of abstract class.

11. Is it possible to declare main method inside the abstract class or not?
    Ans: - Yes, it is possible to declare main method inside the abstract class.

12. In Java, is abstract class reference variable able to hold child class object?
    Ans: - Yes, in java abstract class reference variable is able to hold child class object

Multiple Choice Questions: -

1. Which of these keywords is used to define interfaces in Java?
   a) interface
   b) Interface
   c) intf
   d) Intf

2. Which of these can be used to fully abstract a class from its implementation?
   a) Objects
   b) Packages
   c) Interfaces
   d) None of the Mentioned

3. Which of these access specifiers can be used for an interface?
   a) Public
   b) Protected
   c) private
   d) All of the mentioned

4. Which of these keywords are used to define an abstract class?
   a) abst
   b) abstract
   c) Abstract
   d) abstract class

5. If a class inheriting an abstract class does not define all of its function then it will be known as?
   a) Abstract
   b) A simple class
   c) Static class
   d) None of the mentioned

Answer Key: -

1. a	2. c	3. a	4. b	5. a

Fill in the blanks Questions: -

1. In Java _________________ is used to achieve abstraction
   Ans: - Interface

2. Method which has only declaration is known as __________________.
   Ans: - Abstract Method

3. Class which has abstract methods and implemented methods in known as _________________.
   Ans: - Abstract class

4. Normal class is also called __________________________.
   Ans: - Concrete class

5. In Java, multiple inheritances are possible by __________________.
   Ans: - Interface

Java Lecture – 11

(Exception Handling)

Long Answer Questions: -

1. What is Exception? How many types of exceptions in Java?

Information regarding Exception: -

- Dictionary meaning of the exception is abnormal termination.
- An expected event that disturbs or terminates normal flow of execution is called exception.
- If the application contains exception then the program is terminated abnormally & the rest of the application is not executed.
- To overcome above limitation in order to execute the rest of the application, it must handle the exception.

Types of Exceptions: - As per the Sun Micro Systems standards, the Exceptions are divided into three types:

1) Checked Exception
2) Unchecked Exception
3) Error

Checked Exception: -

✓ The Exceptions which are checked by the compiler at the time of compilation is called Checked Exceptions. **IOException, SQLException, InterruptedException**……..etc.

✓ If the application contains checked exception the code is not compiled. The checked Exception is handled in two ways
  - By using try-catch block.
  - By using throws keyword.

✓ If the application contains checked Exception the compiler is able to check it and it will give intimation to developer regarding Exception in the form of compilation error.

Unchecked Exception: -

✓ The exceptions which are not checked by the compiler at the time of compilation are called unchecked Exception.

ArithmeticException, ArrayIndexOutOfBoundsException, NumberFormatException….etc

✓ If the application contains un-checked exception, code is compiled but at runtime JVM (Default Exception handler) display exception message & program is terminated abnormally.

✓ To overcome runtime problem must handle the exception in two ways.
  - By using try-catch blocks.

- By using throws keyword.

Error: -

- ✓ Errors are caused due to lack of system resources like Heap memory full, Stack memory problem, AWT component problems…..etc
  Ex: - StackOverFlowError, OutOfMemoryError, AssertionError…………etc
- ✓ Exceptions are caused due to developers mistakes or end user supplied inputs but errors are caused due to lack of system resources.

We can handle the exceptions by using try-catch blocks or throws keyword but we are unable to handle the errors.

2. What is Exception handling in Java? How many types of exception handling are there in Java?

Exception Handling: -

- The main objective of exception handling is to get normal termination of the application in order to execute rest of the application code.

- Exception handling means just we are providing alternate code to continue the execution of remaining code and to get normal termination of the application.

- Every Exception is a predefined class present in different packages.

E.g.
java.lang.ArithmeticException
java.io.IOException
java.sql.SQLException
javax.servlet.ServletException

The exception are occurred due to two reasons -
a. Developer mistakes
b. End-user mistakes

i. While providing inputs to the application.
ii. Whenever user is entered invalid data then Exception is occur.

iii. A file that needs to be opened and is not found then Exception is occurred.

iv. Exception is occurred when the network has disconnected at the middle of the communication.

Short Answer Questions: -

1. What is exception?
   Ans: - Exception: - Dictionary meaning of exception is abnormal termination, when exception is occurred program is terminated abnormally and rest of the code is not executed.
   In Java there are three types of exceptions: -
   i.) Checked Exception
   ii.) Unchecked Exception
   iii.) Error

2. What is checked exception?
   Ans: - Checked exceptions are identified by compiler at compilation time. E.g. FileNotFoundException, ClassNotFoundException, IOException, SQLException etc.

3. What is unchecked exception?
   Ans: - Unchecked exceptions are occurred at run time.
   E.g. ArithmeticException, ArrayOutOfBoundsException, NullPointerException .. etc.

4. What is error?
   Ans: - Errors are occurred due to lack of system resources.
   E.g. IOError, AwtError, HeapMemoryFullError… etc.

5. What is exception handling?
   Ans: - Exception handling: - Exception handling is a mechanism to handle exception to achieve normal execution of program.

6. How many types of exception handling?
   Ans: - In Java, there are two types of exception handling: -
   i.) By using try-catch block
   ii.) By using throws keyword.

7. What is difference between throw and throws?

throw	**throws**
Java throw keyword is used to explicitly throw an exception.	Java throws keyword is used to declare an exception.
Checked exception cannot be propagated using throw only.	Checked exception can be propagated with throws.
Throw is followed by an instance.	Throws is followed by class.
Throw is used within the method.	Throws is used with the method signature.
You cannot throw multiple exceptions.	You can declare multiple exceptions e.g. public void method()throws IOException, SQLException.

8. Which keywords are used for exception handling in Java?
   Ans: - For exception handling we use try, catch, finally, throw and throws keywords.

Technical Tasks: -

1. Develop a program in Java to demonstrate concept of exception handling, take the example of divide by zero exception.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
int x,y,z;
Scanner sc=new Scanner(System.in);
System.out.println("Enter two numbers");
try
{
x=sc.nextInt();
y=sc.nextInt();
z=x/y;
System.out.println("Result="+z);
}
catch(InputMismatchException ex1)
{
System.out.println("Enter numbers only");
}
catch(ArithmeticException ex2)
{
System.out.println("Are you trying to / by zero?");
}
finally
{
System.out.println("This is finally block");
}
}
}
```

2. Write a program in Java to display name and roll number of students. Initialize respective array variables for 10 students. Handle ArrayIndexOutOfBoundsExeption, so that any such problem doesn't cause illegal termination of program.

```
class Test
{
public static void main(String [] args)
{
int x[]={10,20,30,40,50};
try
{
for(int i=0;i<=5;i++)
{
System.out.println(x[i]);
}
}
catch(ArrayIndexOutOfBoundsException ex)
{
System.out.println("Array index is out of bounds");
}
}
}
```

3. Develop a program in Java to handle a checked exception ClassNotFoundException.

```
class Test
{
public static void main(String [] args)
{
try
{
Class.forName("com.mysql.jdbc.Driver");
}
catch(ClassNotFoundException ex)
{
System.out.println("Jdbc driver is not available");
}
}
}
```

Interview Questions: -

1. What do you mean by Exception?
   Ans: - Dictionary meaning of exception is abnormal termination. When exception is occurred program is terminated abnormally and rest of code is not executed.

2. How many types of exceptions are there in Java?
   Ans: - In Java, there are three types of exceptions: -
   i.) checked exception
   ii.) unchecked exception
   iii.) errors

3. What is the difference between Exception and error?
   Ans: - Exceptions are generated due to users and programmers mistake whereas errors are generated due to lack of system resources.

4. What is the difference between checked Exception and un-checked Exception?
   Ans: - checked exceptions are identified by compiler at compilation time, whereas unchecked exceptions are generated at runtime.

5. Errors are caused by?
   Ans: - Errors are generated due to lack of system resources.

6. Is it possible to handle Errors in Java?
   Ans: - No, we can handle errors in Java.

7. What do you mean by exception handling?
   Ans: - Exception handling is a mechanism to handle exception to achieve normal execution of program.

8. How many ways are there to handle the exception?
   Ans: - There are two ways to handle the execption: -
   i.) By using try-catch blocks
   ii.) By using throws keyword

9. What is the root class of Exception handling?
   Ans: - Exception is the root class of exception handling.

10. Can you please write some of checked and un-checked exceptions in Java?
    Ans: -
    Checked Exceptions: - ClassNotFoundException, FileNotFoundException, IOException ..etc.
    Unchecked Exceptions: - ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException….etc.

11. What are the keywords present in Exception handling?
    Ans: - try, catch, finally, throws and throw keywords are used for exception handling.

12. What is the purpose of try block?
    Ans: - try block contain the code which you want to protect.

13. In Java, is it possible to write try without catch or not?
    Ans: - In Java, you cannot write try block without catch.

14. What is the purpose of finally block?
    Ans: - In finally block you can write code which you want to execute always.

Multiple Choice Questions: -

1. Which of these keywords is not a part of exception handling?
a) try
b) finally
c) thrown
d) catch

2. Which of these keywords must be used to monitor exceptions?
a) try
b) finally
c) throw
d) catch

3. Which of these keywords is used to manually throw an exception?
a) try
b) finally
c) throw
d) catch

4. Which of the following classes can catch all exceptions which cannot be caught?
a) RuntimeException
b) Error
c) Exception
d) ParentException

5. Which of the following is a super class of all exception type classes?
a) Catchable
b) RuntimeExceptions
c) String
d) Throwable

Answer Key: -

1. c	2. a	3. c	4. c	5. d

Fill in the blanks: -

1. The dictionary meaning of ________________ is abnormal termination.
Ans: - Exception

2. The exceptions caught at compilation time is known as ___________________ exception.
Ans: - Checked

3. The exceptions caught at run time is known as ______________________ exception.
Ans: - Unchecked

4. The exceptions raised due to lack of system resources are called _____________.
Ans: - Errors

5. _______________ is used to raise user defined exception.
Ans: - throw

Java Lecture – 12 (Multithreading)

Long Answer Questions: -

1. What is multithreading? Write difference between multithreading and multitasking.

Information about multithreading: -

1) In the earlier days, the computer's memory use to be occupied with only one program at a time and after completion of one program, it was possible to execute another program, this is called uni programming.

2) When one program execution is completed then only second program execution can be started such type of execution is called co-operative execution. This execution has a lot of disadvantages.

a. Most of the times, memory is wasted.

b. CPU utilization will be reduced because only one program is allowed to be executed at a time.

c. The program queue is developed on the basis co-operative execution.

To overcome above problem a new programming style was introduced and is called multiprogramming.

1) Multiprogramming means executing more than one program at a time.

2) All these programs are controlled by the CPU scheduler.

3) CPU scheduler will allocate a particular time period for each and every program.

4) Executing several programs simultaneously is called multiprogramming.

5) In multiprogramming a program can be entered in different states.

a. Ready state.

b. Running state.

c. Waiting state.

6) Multiprogramming mainly focuses on the number of programs.

Advantages of multiprogramming: -

1. The main advantage of multiprogramming is to provide simultaneous execution of two or more parts of an application to improve the CPU utilization.

2. CPU utilization will be increased.

3. Execution speed will be increased and response time will be decreased.

4. CPU resources are not wasted.

Thread: -

1) Thread is nothing but separate path of sequential execution.

2) The independent execution technical name is called thread.

3) Whenever different parts of the program are executed simultaneously then that each and every part is called thread.

4) The thread is light weight process because whenever we are creating thread it is not occupying the separate memory it uses the same memory. Whenever the memory is shared means it is not consuming more memory.

5) Executing more than one thread at a time is called multithreading.

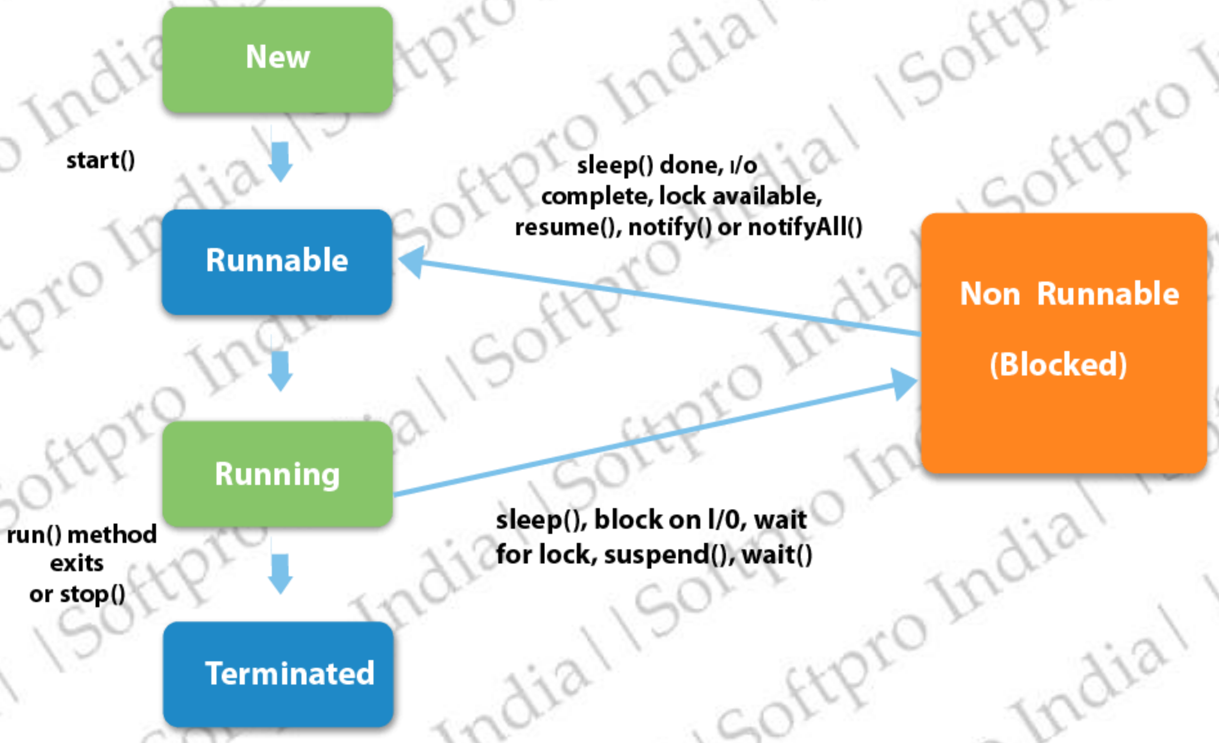
Difference between Multithreading and Multitasking: -

Multitasking	**Multithreading**
In multitasking, users are allowed to perform many tasks by CPU.	While in multithreading, many threads are created from a process through which computer power is increased.
Multitasking involves often CPU switching between the tasks.	While in multithreading also, CPU switching is often involved between the threads.
In multitasking, the processes share separate memory.	While in multithreading, processes are allocated same memory.
Multitasking component involves multiprocessing.	While multithreading component does not involve multiprocessing.
In multitasking, CPU is provided in order to execute many tasks at a time.	While in multithreading also, CPU is provided in order to execute many threads from a process at a time.
In multitasking, processes don’t share same resources, each process is allocated separate resources.	While in multithreading, each process share same resources.

2. Explain thread life cycle in Java.

Thread Life Cycle: - A thread can be in one of the five states. According to Sun, there are only 4 states in **thread life cycle in java -** new, runnable, non-runnable and terminated. There is no running state.
But for better understanding the threads, we are explaining it in the 5 states.
The life cycle of the thread in java is controlled by JVM. The java thread states are as follows:

- i.) New
- ii.) Runnable
- iii.) Running
- iv.) Non-Runnable (Blocked)
- v.) Terminated


New
start()
Runnable
sleep() done, I/o complete, lock available, resume(), notify() or notifyAll()
Non Runnable (Blocked)
Running
sleep(), block on I/0, wait for lock, suspend(), wait()
run() method exits or stop()
Terminated

1) New: -The thread is in new state if you create an instance of Thread class but before the invocation of start() method.
2) Runnable: -The thread is in runnable state after invocation of start() method, but the thread scheduler has not selected it to be the running thread.
3) Running: -The thread is in running state if the thread scheduler has selected it.
4) Non-Runnable (Blocked): -This is the state when the thread is still alive, but is currently not eligible to run.
5) Terminated: -A thread is in terminated or dead state when its run() method exits.

3. How many types of thread can be created in Java? Explain.

Creating Thread: - A thread can be created in two ways: -

1) By extending Thread class.

2) By implementing java.lang.Runnable interface.

First approach to create thread extending Thread class: -

Step 1: - Our normal java class will become Thread class whenever we are extending predefined Thread class.

```
class MyThread extends Thread {
};
```

Step 2: - override the run() method to write the business logic of the Thread(run() method present in Thread class).

```
class MyThread extends Thread {
public void run()  {
System.out.println("business logic of the thread");
System.out.println("body of the thread");
}
}
```

Step 3: - Create userdefined Thread class object.

MyThread t=new MyThread();

Step 4: - Start the Thread by using start() method of Thread class.

t.start();

Example Application: -

```
class MyThread extends Thread //defining a Thread
{
//business logic of user defined  Thread
public void run()
{
for (int i=0;i<10;i++)
{
System.out.println("userdefined Thread");
}
}
};
class ThreadDemo {
public static void main(String[] args) //main thread started
{
MyThread t=new MyThread(); //MyThread is created
t.start(); //MyThread execution started
//business logic of main Thread
for (int i=0;i<10;i++)
{
System.out.println("Main Thread");
}
}
};
```

Flow of execution: - Whenever we are calling t.start() method then JVM will search start() method in the MyThread class but since it not available so JVM will execute parent class(Thread) start() method. Thread class start() method responsibilities

a. User defined thread is registered into Thread Scheduler then only it decides new Thread is created.

b. The Thread class start() automatically calls run() to execute logics of user defined Thread.

Second approach to create thread implementing Runnable interface: -

Step 1: -our normal java class will become Thread class whenever we are implementing Runnable interface.

```
class MyClass implements Runnable {
};
```

Step2*:* override run method to write logic of Thread.

```
class MyClass implements Runnable {
public void run()  {
System.out.println("Ajay from Softpro");
System.out.println("body of the thread");
}
}
```

Step 3: - Creating an object.

MyClass obj=new MyClass();

Step 4: - Creates a Thread class object.

After new Thread is created it is not started running until we call start() method. So whenever we are calling start method that start() method call run() method then the new Thread execution started.

Thread t=new Thread(obj);

t.start();

Example Application: - Creation of Thread implementing Runnable interface .

```
class MyThread implements Runnable {
public void run()  { //business logic of user defined  Thread
for (int i=0;i<10;i++)   {
System.out.println("userdefined Thread");
}
} .
};
class ThreadDemo {
public static void main(String[] args) //main thread started
{
 MyThread r=new MyThread(); //MyThread is created
Thread t=new Thread(r);
 t.start(); //MyThread execution started
//business logic of main Thread
for (int i=0;i<10;i++)
{
System.out.println("Main Thread");
}
}
};
```

Example Application: - Creating two threads by extending Thread class using anonymous inner classes.

```
class ThreadDemo {
public static void main(String[] args)  {
Thread t1 = new Thread() //anonymous inner class
{
public void run()
{
System.out.println("user Thread-1");
}
};
Thread t2 = new Thread() //anonymous inner class
{

public void run()
{
System.out.println("user thread-2");
}
};
t1.start();
t2.start();
}
};
```

Short Answer Questions: -

1. What do you mean by Thread?
   Ans: - Thread is just like sub-process.

2. What is the difference between process and thread?
   Ans: - A program under execution is called process. A thread is just like sub-process.

Technical Tasks: -

1. Create a program in Java to print numbers from 100 to 1 with the delay of 1-1 seconds (Reverse Counter). Use the concept of multithreading.

```
class Test
{
public static void main(String [] args)
{
try
{
for(int i=100;i>0;i--)
{
System.out.println(i);
Thread.sleep(1000);
}
}
catch(InterruptedException ex)
{

}
}
}
```

2. Create a program in Java to create a thread by implementing Runnable interface.

```
class Thread1 implements Runnable
{
public void run()
{
for(int i=1;i<=50;i++)
{
System.out.println(i);
}
}
}
class ThreadDemo2
{
public static void main(String []args)
{
Thread1 t1=new Thread1();
Thread t=new Thread(t1);
t.start();
for(int i=1;i<=10;i++)
{
System.out.println("Main : "+i);
}
}
}
```

3. We create a class that extends the **java.lang.Thread** class. This class overrides the run() method available in the Thread class. A thread begins its life inside run() method. We create an object of our new class and call start() method to start the execution of a thread. Start() invokes the run() method on the Thread object.

```
class MyThread extends Thread
{
public void run()
{
for(int i=1;i<=50;i++)
{
System.out.println(i);
}
}
}
class ThreadDemo1
{
public static void main(String [] args)
{
```

```
MyThread mt=new MyThread();
mt.start();
for(int i=1;i<=20;i++)
{
System.out.println("Main : "+i);
}
}
}
```

Interview Questions: -

1. What do you mean by Thread?
   Ans: - A thread is a path of execution within a process. A process can contain multiple threads.

2. What do you mean by single threaded model?
   Ans: - Single threaded processes contain the execution of instructions in a single sequence. In other words, one command is process at a time.

3. What is the difference between single threaded model and multithreaded model?
   Ans: - Single threaded processes contain the execution of instructions in a single sequence. In other words, one command is process at a time.
   The opposite of single threaded processes are multithreaded processes. These processes allow the execution of multiple parts of a program at the same time. These are lightweight processes available within the process.

4. What do you mean by main thread and what is the importance?
   Ans: - Java provides built-in support for multithreaded programming. A multi-threaded program contains two or more parts that can run concurrently. Each part of such a program is called a thread, and each thread defines a separate path of execution.
   When a Java program starts up, one thread begins running immediately. This is usually called the main thread of our program because it is the one that is executed when our program begins.

5. What is the difference between process and thread?
   Ans: - Process-
   Processes are basically the programs which are dispatched from the ready state and are scheduled in the CPU for execution. PCB(Process Control Block) holds the concept of process. A process can create other processes which are known as Child Processes. The process takes more time to terminate and it is isolated means it does not share the memory with any other process.

   The process can have the following states like new, ready, running, waiting, terminated, suspended.

Thread-

Thread is the segment of a process means a process can have multiple threads and these multiple threads are contained within a process. A thread has three states: Running, Ready and Blocked.

Thread takes less time to terminate as compared to process but unlike process threads do not isolate.

Multiple Choice Questions: -

1. What is multithreaded programming?
   a) It's a process in which two different processes run simultaneously
   b) It's a process in which two or more parts of same process run simultaneously
   c) It's a process in which many different process are able to access same information
   d) It's a process in which a single process can access information from many sources

2. Thread priority in Java is?
   a) Integer
   b) Float
   c) double
   d) long

3. What requires less resources?
   a) Thread
   b) Process
   c) Thread and Process
   d) Neither Thread nor Process

4. What decides thread priority?
   a) Process
   b) Process scheduler
   c) Thread
   d) Thread scheduler

5. Which of these keywords are used to implement synchronization?
   a) synchronize
   b) syn
   c) synch
   d) synchronized

Answer Key: -

1. b	2. a	3. a	4. d	5. d

Fill in the blanks Questions: -

1. Thread is also called ______________________.
   Ans: - Sub process.

2. Thread can be created by extends _______________ class or implementing ______________ interface.
   Ans: - Thread, Runnable

3. _________________ method of Thread class is used to start thread.
   Ans: - start()

4. _______________method of Thread class is used to suspend thread for given time.
   Ans: - sleep()

Java Lecture – 13

(Concept of Package)

Long Answer Questions: -

1. What is package? How many types of packages are there in Java?

Ans: In Java, James Gosling maintained predefined support in the form of packages and these packages contains classes & interfaces and these classes and interfaces contains predefined methods & variables.

Java-language-----→ packages-------→classes and interfaces--------→methods and variables

Types of packages: - There are two types of packages in Java: -

1) Predefined packages

2) User defined packages

Predefined packages: The predefined packages are introduced by James Gosling and these packages contain predefined classes & interfaces and these class & interfaces contains predefined variables and methods. Example: - java.lang, java.io ,java.util…..etc

User defined packages: -

- ❖ The packages which are defined by user. These packages contain user defined classes and interfaces.
- ❖ Declare the package by using package keyword.
  syntax : package package-name;
  example : package com.softpro;
- ❖ Inside the source file it is possible to declare only one package statement and that statement must be first statement of the source file.

Example-1:valid
package com.softpro;
import java.io.*;
import java.lang.*;
	Example-3:Invalid
	import java.io.*;
	import java.lang.*;
	package com.softpro;

Example-2:Invalid
import java.io.*;
package com.softpro;
import java.io.*;
	Example-4:Invalid
	package com.sofpro;
	package com.slc;

Some predefined package and it's classes & interfaces: -

Java.lang: -The most commonly required classes and interfaces to write a sample program is encapsulated into a separate package is called java.lang package.

```
Java
        |-------→lang
                |------→String (Class)
                |------→StringBuffer (Class)
                |-----→Object (Class)
                |------→Runnable (Interface)
                |-----→Clonable (Class)
```

Note: - The default package in Java programming is java.lang package.

Java.io package: -The classes which are used to perform the input output operations are present in the java.io packages.

```
Java
        |-------→io
                |------→ FileInputStream(class)
                |------→ FileOutputStream(class)
                |-----→ FileReader(class)
                |------→ FileWriter(class)
                |-----→ Serializable(interface)
```

Java.net package: -The classes which are required for connection establishment in the network, those classes are present in the java.net package.

```
Java
        |-------→net
                |------→ Socket(class)
                |------→ ServerSocket(class)
                |-----→ URL(class)
                |------→ SocketOption(interface)
```

Short Answer Questions: -

1. What do you mean by package and what it contains?
   Ans: - Package is the collection of classes, interfaces and sub-packages.

2. What is the difference between user defined package and predefined package?
   Ans: -Pre-defined packages are available in JDK, whereas user defined packages are created by user as per their requirement.

3. Is it possible to declare multiple packages in single source file?
   Ans: - No, it is not possible to declare multiple packages in single source file.

4. What do you mean by import?
   Ans: - import is a keyword, which is used to import package in a java program.

Technical Tasks: -

1. Create a package myutil. In myutil package create a class with the name TestMyUtil. In TestMyUtil class create two methods add() and greatest(). The add() method returns sum of two numbers and greatest() method return greatest no in two nos. Now import myutil package in class TestPackage . Now test the class TestMyUtil.
   TestMyUtil.java

```
package myutil;
public class TestMyUtil
{
public int add(int x,int y)
{
return (x+y);
}
public int greatest(int x,int y)
{
if(x>y)
return x;
else
return y;
}
}
```

Test.java

```
import java.util.Scanner;
import myutil.TestMyUtil;
class Test
```

```
{
public static void main(String [] args)
{
int x,y;
Scanner sc=new Scanner(System.in);
TestMyUtil tmu=new TestMyUtil();
System.out.println("Enter two numbers");
x=sc.nextInt();
y=sc.nextInt();
System.out.println("Sum="+tmu.add(x,y));
System.out.println("Greatest No="+tmu.greatest(x,y));
}
}
```

2. Create a package mypack. In mypack package create a public class with the name TempConv. In TempConv class create two methods cToF() and fToC(). cToF() method converts the temperature from centigrade to foreignhite. fToC() method converts the temperature from foreignhite to centigrade. Now import mypack package in class TestTempConv. Now test the class TempConv.
   TempConv.java

```
package mypack;
public class TempConv
{
public double ctof(double c)
{
double f;
f=(9*c)/5+32;
return f;
}
public double ftoc(double f)
{
double c;
c=(f-32)*5/9;
return c;
}
}
```

Test.java

```
import java.util.Scanner;
import mypack.TempConv;
class Test
{
public static void main(String [] args)
{
double c,f;
int ch;
Scanner sc=new Scanner(System.in);
TempConv tc=new TempConv();
System.out.println("Enter 1 for c to f");
System.out.println("Enter 2 for f to c");
ch=sc.nextInt();
switch(ch)
{
case 1:
System.out.print("Enter temperature in c : ");
c=sc.nextDouble();
f=tc.ctof(c);
System.out.println("Temperature in f="+f);
break;
case 2:
System.out.print("Enter temperature in f : ");
f=sc.nextDouble();
c=tc.ftoc(f);
System.out.println("Temperature in c="+c);
break;
default:
System.out.println("Invalid choice");
break;
}
}
}
```

Interview Questions: -

1. Is it possible to import multiple packages in single source file?
   Ans: - Yes, you can import multiple packages in single source file.

2. Is it possible to declare multiple packages in single source file?
   Ans: - No, you can declare one package in single source file.

3. What is difference between main package and sub package?
   Ans: - A package defined inside another package is known as sub package. Sub packages are nothing different than packages except that they are defined inside another package. Sub packages are similar as sub directories which is a directory created inside another directory.

4. What do you mean by fully qualified name of class?
   Ans: - A fully-qualified class name in Java contains the package that the class originated from. Also, an inner class is a class that is another class member. So, the fully-qualified name of an inner class can be obtained using the getName() method.

5. What is the public modifier?
   Ans: - The Java access modifier public means that all code can access the class, field, constructor or method, regardless of where the accessing code is located. The accessing code can be in a different class and different package.

6. What is the default modifier in Java?
   Ans: - Default. When we don't use any keyword explicitly, Java will set a default access to a given class, method or property. The default access modifier is also called package-private, which means that all members are visible within the same package but aren't accessible from other packages.

7. What is most restricted modifier in Java?
   Ans: - private is most restricted modifier in Java.

8. What is most accessible modifier in Java?
   Ans: - public is most accessible modifier in Java.

Multiple Choice Questions: -

1. Which of these keywords is used to define packages in Java?
   a) pkg
   b) Pkg
   c) package
   d) Package

2. Which of these is a mechanism for naming and visibility control of a class and its content?
   a) Object
   b) Packages
   c) Interfaces
   d) None of the Mentioned

3. Which of these access specifiers can be used for a class so that its members can be accessed by a different class in the different package?
   a) Public
   b) Protected
   c) Private
   d) No Modifier

4. Which of the following is the correct way of importing an entire package 'pkg'?
   a) import pkg.
   b) Import pkg.
   c) import pkg.*
   d) Import pkg.*

5. Which of the following package stores all the standard java classes?
   a) lang
   b) java
   c) util
   d) java.packages

Answer Key: -

1. c	2. b	3. a	4. c	5. b

<u>Fill in the blanks Questions: -</u>

1. _______________ keyword is used to create package.
   Ans: - package

2. _____________ keyword is used to access package.
   Ans: - import

3. Package contains ________________ classes.
   Ans: - public

4. ______________is a mechanism for naming and visibility control of a class and its content.
   Ans: - Package

Java Lecture – 14

(Collection Framework Part - I)

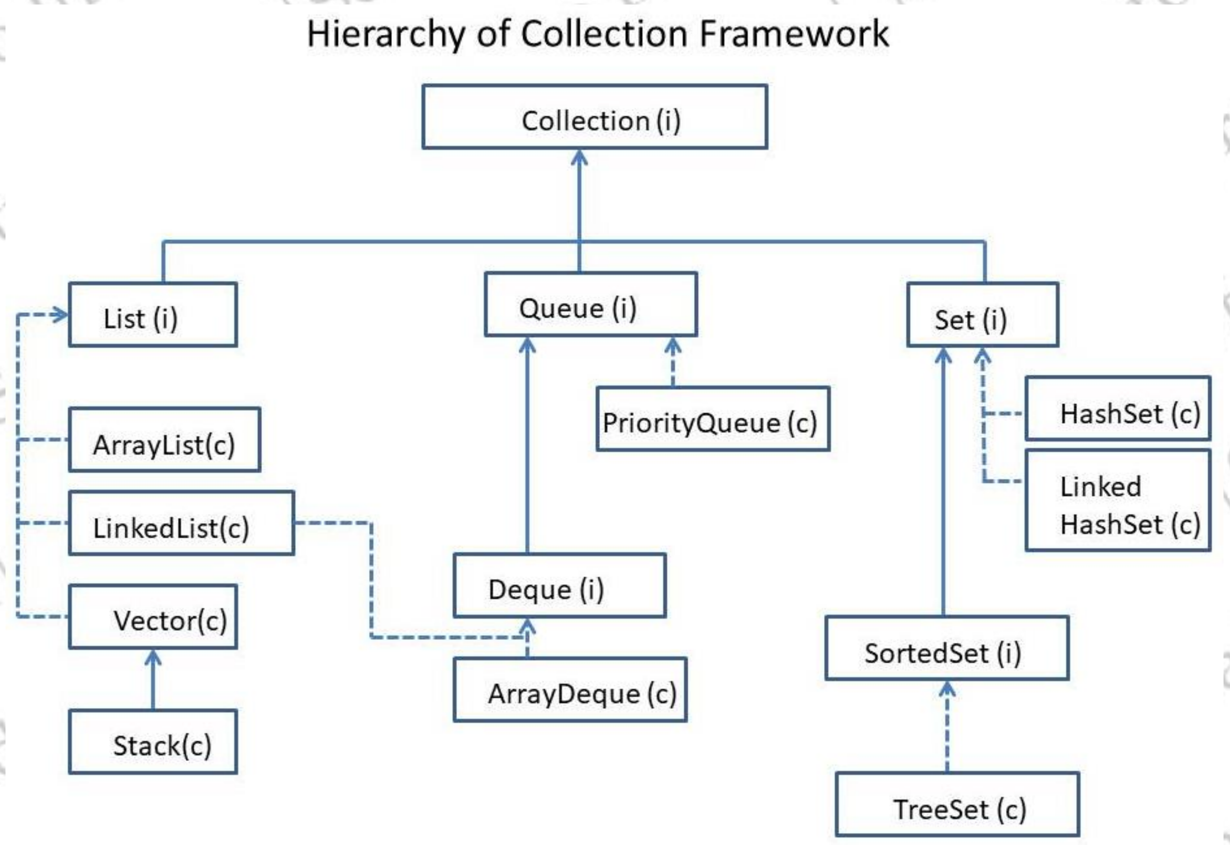
Long Answer Questions: -

1. What do you mean by Java Collection? Describe its architecture.

Ans: - Java Collection: -

- Collection framework contains group of classes and interfaces. By using these classes & interfaces, we are representing group of objects as a single entity.
- Collection is sometimes called a container. It is an object that groups multiple elements into a single unit.
- Collections are used to store, retrieve, manipulate data.

Architecture of Collection: -



2. What are the prerequisites to learn Java Collection?
   Ans: - Prerequisites to learn Java Collection are: -

   1) AutoBoxing
   2) toString() method
   3) type-casting
   4) interfaces
   5) for-each loop
   6) implementation classes
   7) compareTo() method
   8) Wrapper classes
   9) Marker interfaces advantages
   10) Anonymous inner classes

Short Answer Questions: -

1. What is ArrayList? Write is features.
   Ans: -
   ArrayList: - ArrayList is implementing List interface. It is widely used class in projects because it is providing functionality and flexibility.

   ArrayList Characteristics: -

   1) ArrayList was Introduced in 1.2 version.
   2) ArrayList stores Heterogeneous objects(different types).
   3) Inside ArrayList, we can insert Null objects.
   4) ArrayList preserved Insertion order. It means in whatever order, we inserted the data, in the same way the output is printed.
   5) ArrayList methods are non-synchronized methods.
   6) Duplicate objects are allowed.
   7) The under laying data structure is growable array.
   8) By using cursor, we are able to retrieve the data from ArrayList : Iterator , ListIterator

2. In how many types, you can iterate collections?
   Ans: - We can iterate collections in two ways: -
   i.) By using for each loop
   ii.) By using Iterator

3. What is Iterator? How it works?
   Ans: - Iterator enables you to cycle through a collection, obtaining or removing elements. ... Each of the collection classes provides an iterator( ) method that returns an iterator to the start of the collection. By using this iterator object, you can access each element in the collection, one element at a time.

Technical Tasks: -

1. Develop a program in Java to create an ArrayList of String type. Store five names of your friends into ArrayList. Display the elements of ArrayList using for-each loop.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
ArrayList<String> al=new ArrayList<String>();
al.add("Brijesh");
al.add("Yashi");
al.add("Rohit");
al.add("Rajat");
al.add("Disha");
System.out.println("List of my friends");
for(String n:al)
{
System.out.println(n);
}
}
}
```

2. Develop a program in Java to create an ArrayList of String type. Store five names of your friends into ArrayList. Display the elements of ArrayList using iterator.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
ArrayList<String> al=new ArrayList<String>();
al.add("Brijesh");
al.add("Yashi");
al.add("Rohit");
al.add("Rajat");
al.add("Disha");
System.out.println("List of my friends");
Iterator itr=al.iterator();
while(itr.hasNext())
{
System.out.println(itr.next());
}
}
}
```

3. Develop a program in Java to create an ArrayList of objects of user defined class Student. The Student class has rollno, name and fee. Store Student information in ArrayList and display Student information.

```
import java.util.*;
class Student
{
int rollno;
String name;
double fee;
Student(int rollno,String name,double fee)
{
this.rollno=rollno;
this.name=name;
this.fee=fee;
}
}
class Test
{
public static void main(String [] args)
{
ArrayList<Student> al=new ArrayList<Student>();
al.add(new Student(1001,"Brijesh",5000.0));
al.add(new Student(1002,"Yashi",5000.0));
al.add(new Student(1003,"Rohit",5000.0));
System.out.println("List of students");
for(Student s:al)
{
System.out.println(s.rollno+"\t"+s.name+"\t"+s.fee);
}
}
}
```

4. Develop a program in Java to take a LinkedList. Store five employee names in this LinkedList & display the names using Iterator.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
LinkedList<String> al=new LinkedList<String>();
al.add("Brijesh");
al.add("Yashi");
al.add("Rohit");
al.add("Disha");
al.add("Priya");
System.out.println("List of Employees");
Iterator itr=al.iterator();
while(itr.hasNext())
{
System.out.println(itr.next());
}
}
}
```

5. Develop a program in Java to store five elements in LinkedList and reverse those elements.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
LinkedList<String> al=new LinkedList<String>();
al.add("Brijesh");
al.add("Yashi");
al.add("Rohit");
al.add("Disha");
al.add("Priya");
Collections.reverse(al);
System.out.println("List of Employees in reverse order");
Iterator itr=al.iterator();
while(itr.hasNext())
{
System.out.println(itr.next());
}
}
}
```

Interview Questions: -

1. Collection frame work classes are present in which package?
   Ans: - Collection framework classes are present in java.util package.

2. What is the root interface of collections?
   Ans: - Collection interface is the root interface of collections.

3. What is the difference between heterogeneous and homogeneous data?
   Ans: - Homogeneous data means similar data types and heterogeneous data means different data types.

Multiple Choice Questions: -

1. Which of these packages contain all the collection classes?
   a) java.lang
   b) java.util
   c) java.net
   d) java.awt

2. Which of these classes is not a part of Java’s collection framework?
   a) Maps
   b) Array
   c) Stack
   d) Queue

3. Which of these methods deletes all the elements from invoking collection?
   a) clear()
   b) reset()
   c) delete()
   d) refresh()

4. What is Collection in Java?
   a) A group of objects
   b) A group of classes
   c) A group of interfaces
   d) None of the mentioned

5. Which of these standard collection classes implements a dynamic array?
   a) AbstractList
   b) LinkedList
   c) ArrayList
   d) AbstractSet

6. Which of this class can generate an array which can increase and decrease in size automatically?
   a) ArrayList()
   b) DynamicList()
   c) LinkedList()
   d) MallocList()

7. Which of these methods can be used to obtain a static array from an ArrayList object?
   a) Array()
   b) covertArray()
   c) toArray()
   d) covertoArray()

8. Which of these standard collection classes implements a linked list data structure?
   a) AbstractList
   b) LinkedList
   c) HashSet
   d) AbstractSet

9. Which of these method is used to add an element to the start of a LinkedList object?
   a) add()
   b) first()
   c) AddFirst()
   d) addFirst()

10. Which of this method is used to change an element in a LinkedList Object?
   a) change()
   b) set()
   c) redo()
   d) add()

Answer Key: -

1	B	6	A
2	A	7	C
3	A	8	B
4	A	9	D
5	C	10	B

Fill in the blanks Questions: -

1. ________________ is the root interface for all collection.

   Ans: - Collection

2. ArrayList is a class which extends _____________________class and implements ___________ interface.

   Ans: - AbstractList, List

3. Duplicate values are allowed in _______________ and _________________ both.

   Ans: - ArrayList, LinkedList

4. _____________________ method sorts the elements of a collection.

   Ans: - Collections.sort()

Java Lecture – 15

(Collection Framework Part - II)

Long Answer Questions: -

1. What is the difference between Iterator and ListIterator?
   Ans: - Iterator: - Iterators are used in Collection framework in Java to retrieve elements one by one. It can be applied to any Collection object. By using Iterator, we can perform both read and remove operations. Iterator must be used whenever we want to enumerate elements in all Collection framework implemented interfaces like Set, List, Queue, Deque and also in all implemented classes of Map interface. Iterator is the only cursor available for entire collection framework.
   Iterator object can be created by calling iterator() method present in Collection interface.

   ListIterator: -It is only applicable for List collection implemented classes like arraylist, linkedlist etc. It provides bi-directional iteration. ListIterator must be used when we want to enumerate elements of List. This cursor has more functionality (methods) than iterator.
   ListIterator object can be created by calling listIterator() method present in List interface.

2. What is the difference between HashSet, LinkedHashSet and TreeSet?
   HashSet: -
   Java HashSet class is used to create a collection that uses a hash table for storage. It inherits the AbstractSet class and implements Set interface.
   The important points about Java HashSet class are:
   - HashSet stores the elements by using a mechanism called hashing.
   - HashSet contains unique elements only.
   - HashSet allows null value.
   - HashSet class is non-synchronized.
   - HashSet doesn't maintain the insertion order. Here, elements are inserted on the basis of their hashcode.
   - HashSet is the best approach for search operations.

   LinkedHashSet: -

   LinkedHashSet maintains a linked list of the entries in the set, in the order in which they were inserted. This allows insertion-order iteration over the set.

   That is, when cycling through a LinkedHashSet using an iterator, the elements will be returned in the order in which they were inserted.

   The hash code is then used as the index at which the data associated with the key is stored. The transformation of the key into its hash code is performed automatically.

TreeSet: -

TreeSet provides an implementation of the Set interface that uses a tree for storage. Objects are stored in a sorted and ascending order.

Access and retrieval times are quite fast, which makes TreeSet an excellent choice when storing large amounts of sorted information that must be found quickly.

Short Answer Questions: -

1. What is the use of ListIterator?
   ListIterator: - It is only applicable for List collection implemented classes like arraylist, linkedlist etc. It provides bi-directional iteration. ListIterator must be used when we want to enumerate elements of List. This cursor has more functionality (methods) than iterator.
   ListIterator object can be created by calling listIterator() method present in List interface.

2. What is HashSet?
   HashSet: -
   Java HashSet class is used to create a collection that uses a hash table for storage. It inherits the AbstractSet class and implements Set interface.
   The important points about Java HashSet class are:
   - ➢ HashSet stores the elements by using a mechanism called hashing.
   - ➢ HashSet contains unique elements only.
   - ➢ HashSet allows null value.
   - ➢ HashSet class is non synchronized.
   - ➢ HashSet doesn't maintain the insertion order. Here, elements are inserted on the basis of their hashcode.
   - ➢ HashSet is the best approach for search operations.

3. What is LinkedHashSet? Write its features.

   LinkedHashSet: -

   LinkedHashSet maintains a linked list of the entries in the set, in the order in which they were inserted. This allows insertion-order iteration over the set.

   That is, when cycling through a LinkedHashSet using an iterator, the elements will be returned in the order in which they were inserted.

   The hash code is then used as the index at which the data associated with the key is stored. The transformation of the key into its hash code is performed automatically.

4. What is TreeSet?

   TreeSet: -

TreeSet provides an implementation of the Set interface that uses a tree for storage. Objects are stored in a sorted and ascending order.

Access and retrieval times are quite fast, which makes TreeSet an excellent choice when storing large amounts of sorted information that must be found quickly.

Technical Tasks: -

1. Create a LinkedList of Integer type. Store 5 numbers in LinkedList. Now display the LinkedList elements in forward direction and backward direction using ListIterator.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
LinkedList<Integer> al=new LinkedList<Integer>();
al.add(10);
al.add(20);
al.add(30);
al.add(40);
al.add(50);
ListIterator itr=al.listIterator();
System.out.println("Forward direction traversal");
while(itr.hasNext())
{
System.out.println(itr.next());
}
System.out.println("Backward direction traversal");
while(itr.hasPrevious())
{
System.out.println(itr.previous());
}
}
}
```

2. Create a program in Java to demonstrate the concept of LinkedHashSet.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{
LinkedHashSet<Integer> al=new LinkedHashSet<Integer>();
al.add(10);
al.add(20);
al.add(30);
al.add(40);
al.add(50);
al.add(40);
Iterator itr=al.iterator();
System.out.println("Elements of linked hash set");
while(itr.hasNext())
{
System.out.println(itr.next());
}
}
}
```

3. Create a TreeSet of String type. Store five names in TreeSet. Display the elements of TreeSet using Iterator.

```
import java.util.*;
class Test
{
public static void main(String [] args)
{

TreeSet<String> al=new TreeSet<String>();
al.add("Brijesh");
al.add("Ajay");
al.add("Yashi");
al.add("Rohit");
al.add("Disha");
Iterator itr=al.iterator();
System.out.println("Elements of tree set");
while(itr.hasNext())
{
System.out.println(itr.next());
}
}
}
```

Multiple Choice Questions: -

1. Which of these classes implements Set interface?
   a) ArrayList
   b) HashSet
   c) LinkedList
   d) DynamicList

2. Which of this method of HashSet class is used to add elements to its object?
   a) add()
   b) Add()
   c) addFirst()
   d) insert()

3. What is the unique feature of LinkedHashSet?
   a) It is not a valid class
   b) It maintains the insertion order and guarantees uniqueness
   c) It provides a way to store key values with uniqueness
   d) The elements in the collection are linked to each other

4. How to sort elements of ArrayList?
   a) Collection.sort(listObj);
   b) Collections.sort(listObj);
   c) listObj.sort();
   d) Sorter.sortAsc(listObj);

5. Which of these return type of hasNext() method of an iterator?
   a) Integer
   b) Double
   c) Boolean
   d) Collections Object

6. Which of these methods is used to obtain an iterator to the start of collection?
   a) start()
   b) begin()
   c) iteratorSet()
   d) iterator()

7. Which of these methods can be used to move to next element in a collection?
   a) next()
   b) move()
   c) shuffle()
   d) hasNext()

8. Which of these iterators can be used only with List?
   a) Setiterator
   b) ListIterator
   c) Literator
   d) None of the mentioned above

9. Which of this interface declares core method that all collections will have?
   a) set
   b) EventListner
   c) Comparator
   d) Collection

10. Which of this interface handle sequences?
   a) Set
   b) List
   c) Comparator
   d) Collection

Answer Key: -

1	B	6	D
2	A	7	A
3	B	8	B
4	B	9	D
5	C	10	B

Fill in the blanks Questions: -

1. _________________ contain unique values but do not maintain insertion order.
   Ans: - HashSet

2. ________________ contain unique values and maintain insertion order.
   Ans: - LinkedHashSet

3. _______________ contain unique values and maintain ascending order.
   Ans: - TreeSet

4. _______________ contain element in key and value pair.
   Ans: - HashMap