

## Day1\_Class,method,object

DAY1:

- - - - -
- 1. Java Introduction
- 2. Class, Method, Object
- 3. Same package and Different package
- 4. Encapsulation

### QUESTIONS(Theory):

- - - - -
- 1. What is platform independent?
- 2. What is open source?
- 3. Difference between JDK, JRE, JVM?
- 4. Why we go for java?
- 5. What is the latest version of JDK and which version you are using in your project?
- 6. What is the latest version of Eclipse and which version you are using in your project?
- 7. Difference between C++ and Java?
- 8. Features of Java?
- 9. What type of tool you are using in your project to execute Java?
- 10. Difference between class, method, object?
- 11. Where object stores?
- 12. How to access one class method in another package in different package?
- 13. What is encapsulation?
- 14. Coding standard to create project, class, method, package and object?
- 15. What gives Java its "write once and run anywhere" nature?

### QUESTIONS(Programs):

-----

#### QUESTION 1:

-----

```
Project :EmployeeDetails  
Package :org.emp  
Class :Employee  
Methods :empId(), empName(), empDob(), empPhone(), empEmail(), empAddress()
```

#### Description:

Create an object for Employee class and call above methods also follow the all coding standards.

#### QUESTION 2:

-----

```
Project :GreensAddress  
Package :org.add  
Class :GreensTech  
Methods :greensOmr(), greensAdayar(), greensTambaram(), greensVelacherry(), greensAnnaNagar()
```

#### Description:

Create an object for GreensTech class and call above methods also follow the all coding standards.

#### QUESTION 3:

-----

```
Project :CompanyDetails  
Package :org.company  
Class :CompanyInfo  
Methods :companyName(), companyId(), companyAddress()
```

#### Description:

Create an object for CompanyDetails class and call above methods also follow the all coding standards.

#### QUESTION 4:

-----

```
Project :MyPhone  
Package :org.phone  
Class :PhoneInfo  
Methods :phoneName(), phoneMieNum(), Camera(), storage(), osName()
```

**Description:**

Create an object for PhoneInfo class and call above methods also follow the all coding standards.

**QUESTION 5:**

```
-----  
Project :LanguageDetails  
Package :org.lang  
  
Class :LanguageInfo  
Methods :tamilLanguage(),englishLanguage(),hindiLanguage()  
  
Class :StateDetails  
Methods :southIndia(),northIndia()
```

**Description:**

Create an object for LanguageInfo and StateDetails inside the StateDetails class and call both classes methods also follow the all coding standards.

**QUESTION 6:**

```
-----  
Project :EmployeeInformation  
Package :org.emp  
Class :Employee  
Methods :empName()  
  
Package :org.company  
Class :Company  
Methods :companyName()  
  
Package :org.client  
Class :Client  
Methods :clientName()  
  
Package :org.project  
Class :Project  
Methods :projectName()
```

**Description:**

Create an object for all 4 classes inside the Employee class and call all classes methods also follow the all coding standards.

**QUESTION 7:**

```
-----  
Project :PhoneDetails  
Package :org.phone  
Class :ExternalStorage  
Methods :size()  
  
Class :InternalStorage  
Methods :processorName(),ramSize()
```

**Description:**

Create an object for ExternalStorage and InternalStorage inside the InternalStorage class and call both classes methods also follow the all coding standards.

**QUESTION 8:**

```
-----  
Project :CollegeInformation  
Package :org.college  
Class :College  
Methods :collegeName(),collegeCode(),collegeRank()  
  
Class :Student  
Methods :studentName(),studentDept(),studentId()  
  
Class :Hostel  
Methods :hostelName()
```

Class :Dept  
Methods :deptName()

Description:

Create an object for all 4 classes inside the College class and call all classes methods also follow the all coding standards.

QUESTION 9:

-----  
Project :VehicleInformation  
Package :org.allvehicle  
Class :Vehicle  
Methods :VehicleNecessery()

Package :org.twowheeler  
Class :TwoWheller  
Methods :bike(),cycle()

Package :org.threewheeler  
Class :ThreeWheeler  
Methods :Auto()

Package :org.fourwheeler  
Class :FourWheeler  
Methods :car(),bus(),lorry()

Description:

Create an object for all 4 classes inside the Vehicle class and call all classes methods also follow the all coding standards.

QUESTION 10:

-----  
Project :TransportInformation  
Package :org.transport  
Class :Transport  
Methods :TransportForm

Package :org.road  
Class :Road  
Methods :bike(),cycle(),bus(),car()

Package :org.air  
Class :Air  
Methods :aeroPlane(),heliCopter()

Package :org.water  
Class :Water  
Methods :boat(),ship()

Description:

Create an object for all 4 classes inside the Transport class and call all classes methods also follow the all coding standards.

QUESTION 11:

-----  
Project :NetworkInformation  
Package :org.network  
Class :Wifi  
Methods :wifiName()

Class :MobileData  
Methods :dataName()

Class :Lan

Methods :lanName()

Class :Wireless

Methods :modemName()

**Description:**

Create an object for all 4 classes inside the Wifi class and call all classes methods also follow the all coding standards.

Day2\_Inheritance,Scanner,Datatype,access specifier

DAY2:

- 
- 1.Inheritance
- 2.Access specifiers
- 3.Data types
- 4.Scanner class

QUESTIONS(Theory)

- 
- 1.What is mean by inheritance?
- 2.Types of inheritance and explain all types?
- 3.What is mean by multiple inheritance,why java won't support multiple inheritance?
- 4.Difference between hybrid and hierachical inheritance?
- 5.What is the use of access specifier and types?
- 6.Difference between public and protected?
- 7.What is mean by Wrapper class?
- 8.What is default value of String?
- 9.What is difference between primitive and non primitive datatypes?
- 10.What is default package in java?
- 11.What is the super class of all java class?
- 12.What is use of scanner class?
- 13.What are the different methods available in Scanner class?
- 14.Scanner class is under which package?
- 15.Difference between next() and nextLine()?

QUESTIONS(Programs)

-----

QUESTION 1:

-----

Description: Using Scanner class get the below details

- empId
- empName
- empEmail
- empPhoneno
- empSalary
- empGender
- empCity

QUESTION 2:

-----

Description: Using Scanner class get the below details

- studentId
- studentName
- Mark1
- Mark2
- Mark3
- Mark4
- Mark5

:Find the total and average of marks

QUESTION 3:

-----

```
package name: org.all
Project name: LanguageDetails
Class name : Languageclass
Methods     : alllanguage
```

```
package name: org.tamil
Project name: LanguageDetails
Class name : Tamil
Methods     : tamillanguage
```

```
package name: org.english
```

```
Project name: LanguageDetails  
Class name : English  
Methods     : englishlanguage
```

```
package name: org.telgu  
Project name: LanguageDetails  
Class name : Telgu  
Methods     : telgulanguage
```

Description:

create above 4 packages and call all your class methods into the Languageclass using multilevel inheritance.

QUESTION 4:

```
-----  
package name: org.india  
Project name: SouthIndia  
Class name : India  
Methods     : india
```

```
package name: org.tamilnadu  
Project name: SouthIndia  
Class name : TamiladuN  
Methods     : tamillanguage
```

```
package name: org.kerala  
Project name: SouthIndia  
Class name : kerala  
Methods     : malayalam
```

```
package name: org.andrapradesh  
Project name: SouthIndia  
Class name : AndhraPradesh  
Methods     : telugu
```

Description:

create above 4 packages and call all your class methods into the India using multilevel inheritance.

QUESTION 5:

```
-----  
Project   :CollegeInformation  
Package   :org.college  
Class     :College  
Methods   :collegeName(),collegeCode(),collegeRank()
```

```
Class     :Student  
Methods   :studentName(),studentDept(),studentId()
```

```
Class     :Hostel  
Methods   :HostelName()
```

```
Class     :dept  
Methods   :deptName()
```

Description:

create above 4 class and call all your class methods into the Student using multilevel inheritance.

QUESTION 6:

```
-----  
Project   :Computer  
Class     :Computer  
Methods   :computerModel()
```

```
Class     :Desktop  
Methods   :desktopSize()
```

Description:

create above 2 class and call all your class methods into the Desktop using single inheritance.

QUESTION 7:

```
-----  
Project :LanguageDetails  
Package :org.lang  
Class :LanguageInfo  
Methods :tamilLanguage(),englishLanguage(),hindiLanguage()  
  
Class :StateDetails  
Methods :southIndia(),northIndia()
```

Description:

create above 2 class and call all your class methods into the LanguageInfo using single inheritance.

QUESTION 8:

```
-----  
Description: Using Scanner class get the below details  
StudentId  
StudentName  
StudentEmail  
StudentPhoneno  
StudentDept  
StudentGender  
StudentCity
```

QUESTION 9:

```
-----  
Project :BankDetails  
Package :org.bank  
Class :BankInfo  
Methods :saving(),fixed()  
  
Class :AxisBank  
Methods :deposit()
```

Description:

create above 2 class and call all your class methods into the BankInfo using single inheritance.

QUESTION 10:

```
-----  
Project :CompanyDetails  
Package :org.company  
Class :Company  
Methods :companyName()  
  
Package :org.client  
Class :Client  
Methods :clientName()
```

Description:

create above 2 packages and call all your class methods into the Comapany using single inheritance.

QUESTION 11:

```
-----  
Project :EducationInformation  
Package :org.edu  
Class :Education  
Methods :ug(),pg()  
  
Class :Arts  
Methods :bsc(),bEd(),bA(),bBA()  
  
Class :Engineering  
Methods :bE(),bTech()
```

Class :Medicine  
Methods :physiyo(), dental(), mbbs()

Description:

create above 4 class and call all your class methods into the Education using multilevel inheritance.

## Day3\_Polymorphism, Abstraction

DAY3:

- - - - -
- 1. Polymorphism
- 2. Abstraction

### QUESTIONS (Theory)

- - - - -
- 1. What is mean by polymorphism?
- 2. Difference between method overloading and method overriding?
- 3. What is mean by Abstraction?
- 4. Difference between Abstract class and interface?
- 5. What is mean by abstract method?
- 6. Can we create object for abstract class?
- 7. In interface, can we make method as static?
- 8. In interface, can we make method as final?
- 9. How will achieve multiple inheritance in java, write a code for that?

### QUESTIONS (Programs)

- - - - -  
QUESTION 1:  
-----

Find the answer for below questions and tell whether it is possible or not?

I implements I  
I implements C  
I implements A  
I extends I  
I extends C  
I extends A

C implements I  
C implements C  
C implements A  
C extends I  
C extends C  
C extends A

A implements I  
A implements C  
A implements A  
A extends I  
A extends C  
A extends A

A-abstract class  
C-class  
I- interface

QUESTION 2:

- - - - -  
Project :EmployeeDetails  
Package :org.emp  
Class :Employee  
Methods :empId()

Description

You have to overload the method empId() based on different datatype in arguments.

QUESTION 3:

- - - - -  
Project :CompanyDetails  
Package :org.company  
Class :CompanyInfo  
Methods :companyName()

**Description**

You have to overload the method `companyName()` based on different Number of arguments.

**QUESTION 4:**

```
-----  
Project :MyPhone  
Package :org.phone  
Class :Phone  
Methods :phoneInfo()
```

**Description**

You have to overload the method `phoneInfo()` based on different datatype order in arguments.

**QUESTION 5:**

```
-----  
Project :GreensAddress  
Package :org.add  
Class :GreensTech  
Methods :greensOmr()
```

**Description**

You have to overload the method `greensOmr()` based on order,type,number.

**QUESTION 6:**

```
-----  
Project :BankDetails  
Package :org.bank  
Class :BankInfo  
Methods :saving(),fixed(),deposit()  
  
Class :AxisBank  
Methods :deposit()
```

**Description:**

You have to override the method `deposit` in `AxisBank`.

**QUESTION 7:**

```
-----  
Project :EducationInformation  
Package :org.edu  
Class :Education  
Methods :ug(),pg()  
  
Class :Arts  
Methods :bSc(),bEd(),bA(),bBA(),ug(),pg()
```

**Description:**

You have to override the method `ug(),pg()` in `Arts`.

**QUESTION 8:**

```
-----  
Project :UniversityInformation  
Package :org.univ  
Class :University  
Methods :ug(),pg()  
  
Class :College  
Methods :ug(),pg()
```

**Description:**

`ug(),pg()` is just a template in `University` class and You have to override the method `ug(),pg()` in `College` class.

**QUESTION 9:**

```
-----  
Project :BikeInformation  
Package :org.bike
```

```
Interface :Bike
Methods   :cost(),speed()

Class     :Ktm
Methods   :cost(),speed()

Description:
cost(),speed() is just a template in Bike Interface and You have to override the method cost(),speed() in Ktm class.
```

#### QUESTION 10:

```
-----
Project  :Computer
Interface :Hardware
Methods   :hardwareResources()

Interface :Software
Methods   :softwareResources()

Class    :Desktop
Methods   :desktopModel()
```

Description:  
create 2 Interface and achieve multiple inheritance.

## Day4\_ControlStatements

DAY4:

- - - - -
- 1.if/else if
- 2.Loopings(for,while,do-while)
- 3.Switch case
- 4.break/continue

### QUESTIONS(Theory)

- - - - -
- 1.What is difference between break and continue?
- 2.Whether we can use continue statement in switch?
- 3.What is mean by control statements and types?
- 4.What is mean by for loop?
- 5.Can you explain about for loop execution process?
- 6.What is difference between while and do-while?
- 7.What is the use of default keyword in switch?
- 8.Difference between for and while loop?

### QUESTIONS(Find the output)

QUESTION 1:

```
-----  
package org.test;  
  
public class Hello {  
    public static void main(String[] args) {  
        for (int i = 1; i <= 100; i++) {  
            if (i == 5) {  
  
            }  
            System.out.println(i);  
  
        }  
    }  
}
```

QUESTION 2:

```
-----  
package org.test;  
  
public class Hello {  
    public static void main(String[] args) {  
        for (int i = 1; i <= 10; i++) {  
            if (i == 5) {  
                break;  
            }  
            System.out.println(i);  
  
        }  
    }  
}
```

QUESTION 3:

```
-----  
package org.test;  
  
public class Hello {  
    public static void main(String[] args) {  
        for (int i = 1; i <= 10; i++) {  
            if (i == 5) {  
  
            }  
        }  
    }  
}
```

```
        continue;
    }
    System.out.println(i);
}
}
```

QUESTION 4:

```
-----
package org.test;

public class Hello {
    public static void main(String[] args) {
        for (int i = 1; i <= 3; i++) {
            for (int j = 1; j <= 3; j++) {
                System.out.println(j);
            }
        }
    }
}
```

QUESTION 5:

```
-----
package org.test;

public class Hello {
    public static void main(String[] args) {
        for (int i = 1; i <= 3; i++) {
            for (int j = 1; j <= 3; j++) {
                System.out.println(i);
            }
        }
    }
}
```

QUESTION 6:

```
-----
package org.test;

public class Hello {
    public static void main(String[] args) {
        for (int i = 1; i <= 3; i++) {
            for (int j = 1; j <= i; j++) {
                System.out.println(j);
            }
        }
    }
}
```

QUESTION 7:

```
-----
package org.test;

public class Hello {
    public static void main(String[] args) {
```

```
for (int i = 1; i <= 3; i++) {  
    for (int j = i + 1; j <= 3; j++) {  
        System.out.println(j);  
    }  
}  
}  
}
```

QUESTION 8:

```
-----  
package org.test;  
  
public class Hello {  
    public static void main(String[] args) {  
        for (int i = 1; i <= 3; i++) {  
            for (int j = i + 1; j <= i; j++) {  
                System.out.println(j);  
            }  
        }  
    }  
}
```

QUESTION 9:

```
-----  
package org.test;  
  
public class Hello {  
    public static void main(String[] args) {  
        int i=5;  
        if (i == 5) {  
            break;  
        }  
        System.out.println(i);  
    }  
}
```

QUESTION 10:

```
-----  
package org.test;  
  
public class Hello {  
    public static void main(String[] args) {  
        int i=5;  
        if (i == 5) {  
            continue;  
        }  
        System.out.println(i);  
    }  
}
```

QUESTIONS(Programs)

QUESTION 1:

Description: Write Java program to allow the user to input his/her age.  
Then the program will show if the person is eligible to vote.  
A person who is eligible to vote must be older than or equal 1 to 18 years old.

Example:

-----

Input = 10

Output = print not eligible.

QUESTION 2:

-----

Description: Write a program to find even or odd number

Example:

-----

Input = 10

Output = Even

QUESTION 3:

-----

Description: Write a program to print even number from 1 to 100

Example:

-----

Output = 2,4,...,100

QUESTION 4:

-----

Description: Find the sum of odd number 1 to 100

Example:

-----

Output = 2500

QUESTION 5:

-----

Description: Count of even number 1 to 100

Example:

-----

Output = 50

QUESTION 6:

-----

Description: Write a program to find the factorial of a number.

Example:

-----

Input = 5

Output = 120

QUESTION 7:

-----

Description: Write a program to print the fibonacci series of a number 1 to 100.

Example:

-----

Output = 0,1,1,2,3,5.....

QUESTION 8:

-----

Description: Find prime number or not.

Example:

-----

Input = 11

Output = prime number

QUESTION 9:

-----

Description : Print the below patterns using for loop.

Output:

```
-----
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
1 2 3 4 5 6 7
-----
      *
    * *
  * * *
* * * *
* * * * *
```

```
-----
*
* *
* * *
* * * *
* * * * *
```

```
-----
```

QUESTION 10:

```
-----  
Description: Find Amstrong number or not
```

Example:

```
-----  
Input = 153  
Output = Amstrong number
```

QUESTION 11:

```
-----  
Description: Reverse the number
```

Example:

```
-----  
Input = 123  
Output = 321
```

QUESTION 12:

```
-----  
Description: Count of the number
```

Example:

```
-----  
Input = 123  
Output = 3
```

QUESTION 13:

```
-----  
Description: Sum of the number
```

Example:

```
-----  
Input = 123  
Output = 6
```

QUESTION 14:

```
-----  
Description: Verify the number is palindrome number not
```

Example:

```
-----
```

Input = 141  
Output = Palindrome

DAY5:

- 1.Constructor and types  
2.this and super  
3.types of var  
4.static/final

QUESTIONS(Theory)

- 1.What is mean by constructor and types?  
2.Explain the types of variable  
3.Do constructors have any return type?  
4.Syntax for creating constructor?  
5.What are the rules for defining a constructor?  
7.Why a return type is not allowed for constructor?  
8.Can we declare constructor as 'private'?  
9.Why a compiler given constructor is called as default constructor?  
10.What is constructor chaining and how can it be achieved in Java?  
11.Can we use this() and super() in a method?  
12.What are the common uses of "this" keyword in java?  
13.Types of variable?  
14.What is meant by local variable,instance variable,class/static variable?  
15.What is mean by static keyword in java?  
16.Can we override static method in java?  
17.Can we overload static method in java?  
18.What is mean by static variable?  
19.What is mean by static method?  
20.What is mean by final keyword and what's happen when we declare final as in class,method,variable?  
21.What is difference between final and finalize keyword?  
22.Where local,static and class variables stores in jvm?

QUESTIONS(Find the below Output)

-----  
QUESTION 1:

```
-----  
package org.test;  
  
public class A {  
    public A() {  
        this("JAVA");  
        System.out.println("Default const...");  
    }  
  
    public A(int id) {  
        this(3456.5678f);  
        System.out.println(id);  
    }  
  
    public A(String name) {  
        this(12);  
        System.out.println(name);  
    }  
  
    public A(float sal) {  
        System.out.println(sal);  
    }  
  
    public static void main(String[] args) {  
        A a = new A();  
    }  
}
```

QUESTION 2:

-----

```
package org.test;

public class A extends B{
    public A() {
        System.out.println("Default const...");
    }

    public static void main(String[] args) {
        A a = new A();
    }
}

package org.test;

public class B {
    public B() {
        System.out.println("Super class");
    }
}
```

QUESTION 3:

```
-----
package org.test;

public class A extends B{
    public A() {
        System.out.println("Default const...");
    }

    public static void main(String[] args) {
        A a = new A();
    }
}

package org.test;

public class B {
    public B() {
        System.out.println("Super class");
    }

    public B(int id) {
        System.out.println(id);
    }
}
```

QUESTION 4:

```
-----
package org.test;

public class A extends B {
    public A() {
        super(12);
        System.out.println("Default const...");
    }

    public static void main(String[] args) {
        A a = new A();
    }
}
```

```
}
```

```
package org.test;
```

```
public class B {
```

```
    public B() {
```

```
        System.out.println("Super class");
```

```
    }
```

```
    public B(int id) {
```

```
        System.out.println(id);
```

```
    }
```

```
}
```

QUESTION 5:

```
-----
```

```
package org.test;
```

```
public class B {
```

```
    public B(int id) {
```

```
        System.out.println(id);
```

```
    }
```

```
}
```

```
package org.test;
```

```
public class A extends B {
```

```
    public A() {
```

```
        super(12);
```

```
        System.out.println("Default const...");
```

```
    }
```

```
    public static void main(String[] args) {
```

```
        A a = new A();
```

```
    }
```

```
}
```

QUESTION 6:

```
-----
```

```
package org.test;
```

```
public class A extends B {
```

```
    public A() {
```

```
        System.out.println("Default const...");
```

```
    }
```

```
    public static void main(String[] args) {
```

```
        A a = new A();
```

```
    }
```

```
}
```

```
package org.test;
```

```
public class B {
```

```
    public B(int id) {
```

```
        System.out.println(id);
```

```
    }
```

```
}
```



DAY6:  
-----  
1.String functions  
2.Literal String  
3.Non Literal String  
4.Mutable String  
5.ImMutable String

#### QUESTIONS(Theory)

- 1.What is mean by string?  
2.How to find length of the string?  
3.How to find particular character in string?  
4.How to split the string?  
5.What is difference between literal String and non literal string?  
6.What is mutable and immutable string?  
7.Difference between stringbulider and stringbuffer?  
8.Method name to identify memory location?  
9.What are the string functions available in java?  
10.What is difference between charAt() and contains() method?  
11.What is the return type of compareTo()  
12.Where the Literal String and non literal String stores?  
13.What is mean by ASCII value?

#### QUESTIONS(Programs)

##### QUESTION 1:

Description: Find the length of the below string

String 1: GreensTechnology  
String 2: SeleniumAutomationtool  
String 3: velmurugan  
String 4: j a v a p r o g r a m  
String 5: 9095484678

##### QUESTION 2:

Description: Find the particular character index in the given string

String 1: GreensTechnology  
Find the last index of o  
  
String 2: SeleniumAutomationtool  
Find the index of o  
  
String 3: Velmurugan  
Find the index of n  
  
String 4: j a v a p r o g r a m  
Find the last index of (emptyspace)  
  
String 5: 9095484678  
Find the index of 8

##### Question 3:

Description: Find the particular character in the given string

String 1: GreensTechnology  
print the character h  
  
String 2: SeleniumAutomationtool  
print the character o  
  
String 3: velmurugan  
print the character u

```
String 4: j a v a p r o g r a m  
          print the character p
```

```
String 5: 9095484678  
          print the character 7
```

#### QUESTION 4:

-----  
Description : Get two input from user and check the equality  
 : print in the output whether it is Equal or not

Example:

-----  
Input :

```
String 1 : Java  
String 2 : Java
```

Example:

-----  
Input :

```
String 1 : Java  
String 2 : java
```

Example:

-----  
Input :

```
String 1 : Green Technology  
String 2 : GreenTechnology.
```

Example(use equalsIgnoreCase):

-----  
Input :

```
String 1 : Java  
String 2 : java
```

Example(use equalsIgnoreCase) :

-----  
Input :

```
String 1 : Nisha  
String 2 : nisha
```

#### QUESTION 5:

-----  
QUESTION 5.1:

-----  
Description: Get the email id from the user and verify '@' is present or not?

Example:

-----  
Input = velmurugank451@gmail.com

Output = valid email id

-----  
QUESTION 5.2:

-----  
Description:Get the address from the user and verify "pincode" is present or not?

Example:

-----  
Input = 5-35-2a,venkatesh nivas,Aruppukottai

Output = invalid address

-----  
QUESTION 5.3:

-----  
Description:Get the email from the user and verify '@' is present or not and return true or false?

Example:

-----

Input = Nishakerala24@gmail.com  
Output = True/False

QUESTION 5.4:

-----  
Description: Get the phonenumber from the user and verify any character is present or not .  
If character is present return invalid number

Example:

-----  
Input = 90954a6078  
Output = False

QUESTION 6:

-----  
Description: Get the phonenumber from the user .  
If phonenumber exceeds greater than 10 then return invalid number

Example:

-----  
Input = 89034256972365  
output = invalid

Example 2:

-----  
Input = 9095484678  
Output = valid

QUESTION 7:

QUESTION 7.1:

-----  
Description: Given string as "Welcome to java class" and replace java into sql.

Example:

-----  
Input = Welcome to class java  
output = Welcome to class sql

QUESTION 7.2:

-----  
Description: Given string as "Greens Adayar"and replace Adayar into Omr.

Example:

-----  
Input = Greens Adayar  
Output = Greens Omr

QUESTION 7.3:

-----  
Description: Given String as "Welcome to java class" and Replace space into '#'

Example:

-----  
input:Welcome to java class  
output:Welcome#to#java#class

QUESTION 7.4:

-----  
Description: Get the email from the user and verify "gmail" is present or not.  
If present replace that gmail into yahoo

Example:

-----  
Input = Nishakerala24@gmail.com  
Output = Nishakerala24@yahoo.com

**QUESTION 7.5:**

-----  
Description: Get the address from the user and verify "pincode" is present or not.  
If present replace the pincode with empty space

Example:

-----  
Input = 5-35-2a,venkatesh nivas,Aruppukottai,pincode-626101  
Output = 5-35-2a,venkatesh nivas,Aruppukottai

**QUESTION 8**

-----  
**QUESTION 8.1**

-----  
Description: Get the input from the user and print that word in lowercase

Example:

-----  
Input = NISHANTHI  
Output = nishanthi

**QUESTION 8.2**

-----  
Description: Get the input from the user and print that word in Uppercase

Example:

-----  
Input = nishanthi  
Output = NISHANTHI

**QUESTION 8.3**

-----  
Description: Convert all small letter and into capital letter

Example:

-----  
Input = WelcomE  
Output = wELCOMe

**QUESTION 8.4**

-----  
Description: Find the number of uppercase count and lowercase count in the given String

Example:

-----  
Input = WelComeToJava

Output:

-----  
UpperCase=4

LowerCase=9

**QUESTION 9**

-----  
**QUESTION 9.1**

-----  
Description: Given String as "Welcome to java class" and verify whether the given string startsWith welcome

Example:

-----  
Input = Welcome to class java  
output = True

**QUESTION 9.2**

-----  
Description: Given String as "Hai i am nisha" and verify whether the given string startsWith welcome

Example:

-----  
Input = Hai i am nisha  
output = False

QUESTION 9.3

-----  
Description: Given String as "Welcome to java class" and verify whether the given string endsWith class

Example:

-----  
Input = Welcome to java class  
output = True

QUESTION 9.4

-----  
Description: Given String as "Welcome to java class" and verify whether the given string endsWith java

Example:

-----  
Input = Welcome to java class  
output = False

QUESTION 9.5

-----  
Description: Given String as "Welcome to java class" and verify whether the string is empty or not

Example:

-----  
Input = Welcome to java class  
output = False

QUESTION 9.6

-----  
Description: Given String as "" and verify whether the string is empty or not

Example:

-----  
Input = ""  
Output = False

QUESTION 10

-----  
Description : Get two input from user and Compare

Example

-----  
String 1 : Nisha  
String 2 : nisha

Example

-----  
String 1 : Nia  
String 2 : nisha

QUESTION 11

-----  
QUESTION 11.1

-----  
Description : Generate the two literal string and find the identityHashCose()

Example

-----

```
String 1 : Nisha  
String 2 : Nisha
```

#### QUESTION 11.2

-----  
Description : Generate the two non literal string and find the identityHashCose()

Example

```
String 1 : Nisha  
String 2 : Nisha
```

#### QUESTION 11.3

-----  
Description : Generate the three non literal string and find the identityHashCose()

Example

```
String 1 : Nisha  
String 2 : Rengan  
String 3 : NishaRengan
```

#### QUESTION 11.4

-----  
Description : Generate the three literal string and find the identityHashCose()

Example

```
String 1 : Nisha  
String 2 : Rengan  
String 3 : NishaRengan
```

#### QUESTION 12

##### QUESTION 12.1

-----  
Description: Given String as "Welcome to java class" and split it by space.

Example:

```
-----  
Input :Welcome to java class  
Output:  
-----  
Welcome  
to  
java  
class
```

#### QUESTION 12.2

-----  
Description: Given String as "Welcome to java class" and split it by 1

Example:

```
-----  
Input :Welcome to java class  
Output:  
-----  
We  
come to java c  
ass
```

#### QUESTION 13

##### QUESTION 13.1

-----  
Description: Given String as "Welcome to java class" and generate a substring.

Example:

-----  
Input :Welcome to java class

Output:

-----  
Welcome

QUESTION 14

-----

Example:

-----

Description: Given String as "Welcome" and the number of consonant count and vowels count

Example:

-----

Input = Welcome

output:

-----

vowels = 3

consonant = 4

QUESTION 15:

-----

Description: Find the count of caps,small,number and special character in given string

Example:

-----

Input : Welcome To Java class @123

Output

-----

caps count :3

small count :15

number count:3

Special char:5

QUESTION 16

-----

Description: Replace all vowels char into '@'

Example:

-----

Input = Welcome

Output = W@lc@m@

DAY7:

- 1.Arrays  
2.Collections Introduction  
3.List(ArrayList only)

QUESTIONS(Theory)

- 1.What do you mean by an array and How to create an Array?  
2.Can you change size of array once created?  
3.Is it legal to initialize an array int i[] = {1, 2, 3, 4, 5}  
4.Advantages and disadvantages of Array?  
5.Can we change the size of an array at run time?  
6.Can you declare an array without assigning the size of an array?  
7.What is the default value of Array?  
8.How to print element of Array?  
9.How to compare Two Arrays?  
10.How to sort an Array?  
11.Can we declare array size as a negative number?  
12.Can we add or delete an element after assigning an array?  
13.Can we use Generics with the array?  
14.What is collection and explain about types?  
15.What is the difference between ArrayList and Vector?  
16.What is the difference between ArrayList and LinkedList?  
17.How to convert Array to List and List to Array

QUESTIONS(Find the below Output)

QUESTION 1:

Description : Write a Java program to sum values of an array  
Input a[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}  
Output = 55

QUESTION 2:

Description : Write a Java program to calculate the average value of array elements.  
Input a[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}  
Output = average = 7.0

QUESTION 3:

Description : Write a Java program to remove duplicates from array  
Input a[] = {10, 10, 20, 50, 60, 80, 60, 50}  
Output a[] = {10, 20, 50, 60, 80}

QUESTION 4:

QUESTION 4.1:  
Description : Create a new ArrayListlist with values and find the length of it  
Input : List = 10, 20, 30, 90, 10, 10, 40, 50

QUESTION 4.2:

Description : Create a new LinkedListlist with values and find the length of it  
Input : List = 100, 200, 300, 400, 500, 600, 700

QUESTION 4.3:

Description : Create a new vector with values and find the length of it  
Input : List = 105, 205, 305, 405, 505, 605, 705, 805

QUESTION 4.4:

Description : Create a new LinkedListlist with values and find the size() of it.  
Input : List = 100,200,300,400,500,600,700

QUESTION 5:

-----  
QUESTION 5.1:

-----  
Description : Get the first index value of 10  
Input: List = 10,20,30,90

QUESTION 5.2:

-----  
Description : Get the last index value of 10  
Input: List = 10,20,30,90,10,10,40,50

QUESTION 5.3:

-----  
Description : Get the index value of 50  
Input: List = 10,20,30,90,10,10,40,50

QUESTION 5.4:

-----  
Description : Get the index value of 90  
Input: List = 10,20,30,90,10,10,40,50

QUESTION 5.5:

-----  
Description : Get the each index value of 10 present in below list  
Input: List = 10,20,30,90,10,10,40,50,10

QUESTION 5.6:

-----  
Description : Get the each index value of 70 present in below list  
Input: List = 10,20,30,90,10,10,40,50,10

QUESTION 6:

-----  
QUESTION 6.1:

-----  
Description : Get the value present at 2nd index  
Input: List = 10,20,30,40,50,60

QUESTION 6.2:

-----  
Description : Get the value present at 4th index  
Input: List = 100,200,300,400,500,600,700

QUESTION 6.3:

-----  
Description : Get the value present at 8th index  
Input: List = 105,205,305,405,505,605,705,805

QUESTION 6.4:

-----  
Description : Get the each value of list by using normal for loop  
Input: List = 105,205,305,405,505,605,705,805

QUESTION 6.5:

-----  
Description : Get the each value of list by using enhanced for loop  
Input: List = 105,205,305,405,505,605,705,805

QUESTION 7:

-----  
QUESTION 7.1:

-----  
Description : Remove the value present at 2nd index

Input: List = 10,20,30,40,50,60

QUESTION 7.2:

Description : Remove the value present at 10th index  
Input: List = 10,20,30,90,10,10,40

QUESTION 7.3:

Description : Remove the last value of 10 present in the list  
Input: List = 10,20,30,90,10,10,40

QUESTION 8:

QUESTION 8.1:

Description : Add a value 50 in the 2nd index and display the list after adding.  
Input : List = 10,20,30,90,10,10,40,50

QUESTION 8.2:

Description : Add a value 70 at the end of the list  
Input : List = 10,20,30,90,10,10,40,50

QUESTION 8.3:

Description : Add a value 80 at the 8th index of list  
Input : List = 10,20,30,90,10,10,40,50

QUESTION 8.4:

Description : Add a value 100 at the last index of 10 in the list  
Input : List = 10,20,30,90,10,10,40,50

QUESTION 9:

QUESTION 9.1:

Description : Replace the value 300 into 350 in the list  
Input : List = 100,200,300,400,500,600,700

QUESTION 9.2:

Description : Replace the value present in 7th index as 90  
Input: List = 10,20,30,90,10,10,40,50,10

QUESTION 9.3:

Description : Replace the 10 into 100 in List  
Input: List = 10,20,30,90,10,10,40,50,30  
Output: List = 100,20,30,90,100,100,40,50,30

QUESTION 10:

QUESTION 10.1:

Description : Create a new ArrayListlists with values and return the common values  
Input : List = 10,20,30,90,10,10,40,50  
Input : List = 30,40,50,60,80

QUESTION 10.2:

Description : Create a new ArrayListlists with values and return the common values  
Input : List = 10,20,30,90,10,10,40,50  
Input : List = 10,20,60,50,40,70,80,90

QUESTION 10.3:

-----  
Description : Create a new ArrayListlists with values and return the common values  
Input : List = 10,20,30,40,50,60,70,80  
Input : List = 100,200,300,400,500,600,700,8000

QUESTION 11:

-----  
QUESTION 11.1:

-----  
Description : Create a new ArrayListlists with values and perform removeAll() function  
Input : List = 10,20,30,90,10,10,40,50  
Input : List = 30,40,50,60,80

QUESTION 11.2:

-----  
Description : Create a new ArrayListlists with values and perform removeAll() function  
Input : List = 10,20,30,90,10,10,40,50  
Input : List = 10,20,60,50,40,70,80,90

QUESTION 11.3:

-----  
Description : Create a new ArrayListlists with values and perform removeAll() function  
Input : List = 10,20,30,40,50,60,70,80  
Input : List = 100,200,300,400,500,600,700,8000

DAY8:  
-----  
1.Userdefine List  
2.LinkedList vs Vector  
3.Set and types

Day8:  
-----  
1.Userdefine List  
2.LinkedList vs Vector  
3.Set and types

#### QUESTIONS(Theory)

- 1.Describe the Collections type hierarchy. What are the main interfaces, and what are the differences between them?  
2.Explain about user defined list?  
3.Mention what is Linked List?  
4.what is difference between set and List?  
5.What is the difference between HashSet and TreeSet ?  
6.Difference between Enumerator,Iterator and List Iterator  
7.How to convert List into Set

#### QUESTIONS:

##### QUESTION 1.1:

Description : Create a HashSet for the below values  
Input : List = 10,20,30,40,50,60,70,80,90,10,20

##### QUESTION 1.2:

Description : Create a LinkedHashSet for the below values  
Input : List = 10,20,30,40,50,60,70,80,90,10,20

##### QUESTION 1.3:

Description : Create a TreeSet for the below values  
Input : List = 10,20,30,40,50,60,70,80,90,10,20

##### QUESTION 2:

###### QUESTION 2.1:

Description : Convert the below list in to set(use addAll() method)  
Input : List = 10,20,30,90,10,10,40,50

##### QUESTION 2.2:

Description : Convert the below list in to set(use addAll() method)  
Input : List = 105,205,305,405,505,605,705,805,505,605

##### QUESTION 2.3:

Description : Convert the below list in to set(use addAll() method)  
Input : List = 100,200,300,400,500,600,700,100,300,500

##### QUESTION 3:

###### QUESTION 3.1:

Description : Get the each value of set by using enhanced for loop  
Input: List = 105,205,305,405,505,605,705,805

##### QUESTION 3.2:

Description : Create a TreeSet and iterate each value in the set by using enhanced for loop  
Input : List = 10,20,30,40,50,60,70,80,90,10,20

QUESTION 3.3:

-----  
Description : Create a HashSet and iterate each value in the set by using enhanced for loop  
Input : List = 10,20,30,40,50,60,70,80,90,10,20

QUESTION 4:

-----  
QUESTION 4.1:

-----  
Description : Create a new HashSet with values and return the common values  
Input : List = 10,20,30,90,10,10,40,50  
Input : List = 30,40,50,60,80

QUESTION 4.2:

-----  
Description : Create a new LinkedHashSet with values and return the common values  
Input : List = 10,20,30,90,10,10,40,50  
Input : List = 10,20,60,50,40,70,80,90

QUESTION 4.3:

-----  
Description : Create a new TreeSet with values and return the common values  
Input : List = 10,20,30,40,50,60,70,80  
Input : List = 100,200,300,400,500,600,700,8000

QUESTION 5:

-----  
QUESTION 5.1:

-----  
Description : Create a new HashSet with values and perform removeAll() function  
Input : List = 10,20,30,90,10,10,40,50  
Input : List = 30,40,50,60,80

QUESTION 5.2:

-----  
Description : Create a new LinkedHashSet with values and perform removeAll() function  
Input : List = 10,20,30,90,10,10,40,50  
Input : List = 10,20,60,50,40,70,80,90

QUESTION 5.3:

-----  
Description : Create a new TreeSet with values and perform removeAll() function  
Input : List = 10,20,30,40,50,60,70,80  
Input : List = 100,200,300,400,500,600,700,8000

QUESTION 6:

-----  
Description: Create a userdefine Set and insert the 10 employee details  
Input : empId,name,phone,address,dob,doj,eMail,gender,Sal  
Output: Using scanner class insert 10 employee details

QUESTION 7:

-----  
Description: Create a userdefine Set and insert the 10 Student details  
Key : stdId,stdName,stdPhone,address,dOB,eMail,gender  
Give the related values for key for each Student

DAY9:

- 1.Map and types  
2.User define Map

QUESTIONS(Theory)

- 1.What is map?  
2.What is difference between Hash Map and Hash Table?  
3.What is difference between set and Map?  
4.What are the classes implementing List interface?  
5.Which all classes implement Set interface ?  
6.Explain about user defined Map?  
7.How null allows in below maps:  
    HashMap :k?,v?  
    LinkedHashMap:k?,v?  
    TreeMap :k?,v?  
    HashTable :k?,v?  
8.How to Iterate Map ?  
9.Return type entrySet?

QUESTIONS(practical)

-----  
QUESTION 1.1:

Description : Create a HashMap with the below key and values  
    key : 10,20,30,40,50,60,10,50,40  
    values : java,sql,oops,Sql,oracle,DB,selenium,psql,Hadoop.

QUESTION 1.2:

Description : Create a LinkedHashMap with the below key and values  
    key : 10,20,30,40,50,60,10,50,40  
    Values : 10,20,30,40,50,60,10,50,40

QUESTION 1.3:

Description : Create a TreeMap with the below key and values  
    key : !,@,#,\$,%,&,\*,(,  
    Values : 10,20,30,40,50,60,10,50,40

QUESTION 1.4:

Description : Create a HashTable with the below key and values  
    Key : vel,Ganesh,Dinesh,Vengat,subash  
    Values : Selenium,framework,oracle,corejava,jira

QUESTION 2:

-----  
QUESTION 2.1:

Description : Create a HashMap with the below key and values and get(print) the key only in the map.  
    key : 10,20,30,40,50,60,10,50,40  
    values : java,sql,oops,Sql,oracle,DB,selenium,psql,Hadoop.

QUESTION 2.2:

Description : Create a LinkedHashMap with the below key and values and get(print) the key only in the map.  
    key : 10,20,30,40,50,60,10,50,40  
    Values : 10,20,30,40,50,60,10,50,40

QUESTION 2.3:

Description : Create a TreeMap with the below key and values and get(print) the key only in the map.  
key : !,@,#,\$,%,&,\*,(,  
Values : 10,20,30,40,50,60,10,50,409

QUESTION 2.4:

-----  
Description : Create a HashTable with the below key and values and get(print) the key only in the map.

Key : vel,Ganesh,Dinesh,Vengat,subash  
Values : Selenium,framework,oracle,corejava,jira

QUESTION 3:

QUESTION 3.1:

-----  
Description : Create a HashMap with the below key and values and get(print) the values only in the map.

key : 10,20,30,40,50,60,10,50,40  
values : java,sql,oops,Sql,oracle,DB,selenium,psql,Hadoop.

QUESTION 3.2:

-----  
Description : Create a LinkedHashMap with the below key and values and get(print) the values only in the map.

key : 10,20,30,40,50,60,10,50,40  
Values : 10,20,30,40,50,60,10,50,40

QUESTION 3.3:

-----  
Description : Create a TreeHashMap with the below key and values and get(print) the values only in the map.

key : !,@,#,\$,%,&,\*,(,  
Values : 10,20,30,40,50,60,10,50,409

QUESTION 3.4:

-----  
Description : Create a HashTable with the below key and values and get(print) the key only in the map.

Key : vel,Ganesh,Dinesh,Vengat,subash  
Values : Selenium,framework,oracle,corejava,jira

QUESTION 4:

QUESTION 4.1:

-----  
Description : Create a HashMap with the below key and values and iterate it using enhanced for loop.

key : 10,20,30,40,50,60,10,50,40  
values : java,sql,oops,Sql,oracle,DB,selenium,psql,Hadoop.

QUESTION 4.2:

-----  
Description : Create a LinkedHashMap with the below key and values and iterate it using enhanced for loop.

key : 10,20,30,40,50,60,10,50,40  
Values : 10,20,30,40,50,60,10,50,40

QUESTION 4.3:

-----  
Description : Create a TreeHashMap with the below key and values and iterate it using enhanced for loop.

key : !,@,#,\$,%,&,\*,(,  
Values : 10,20,30,40,50,60,10,50,409

QUESTION 4.4:

Description : Create a HashTable with the below key and values and iterate it using enhanced for loop.

Key : vel,Ganesh,Dinesh,Vengat,subash  
Values : Selenium,framework,oracle,corejava,jira

QUESTION 5:

QUESTION 5.1:

Description : Create a HashMap with the below key and values and iterate it using enhanced for loop and get the key and values combination.

key : 10,20,30,40,50,60,10,50,40  
values : java,sql,oops,Sql,oracle,DB,selenium,psql,Hadoop.

Sample Output:

10  
java  
20  
sql  
30  
oops  
40  
sql  
Description : like this you have to iterate the map

QUESTION 5.2:

Description : Create a LinkedHashMap with the below key and values and iterate it using enhanced for loop and get the key and values combination..

key : 10,20,30,40,50,60,10,50,40  
Values : 10,20,30,40,50,60,10,50,40

QUESTION 5.3:

Description : Create a TreeMap with the below key and values and iterate it using enhanced for loop and get the key and values combination.

key : !,@,#,\$,%,&,\*,(,  
Values : 10,20,30,40,50,60,10,50,409

QUESTION 5.4:

Description : Create a HashTable with the below key and values and iterate it using enhanced for loop and get the key and values combination.

Key : vel,Ganesh,Dinesh,Vengat,subash  
Values : Selenium,framework,oracle,corejava,jira

QUESTION 6:

Description: Create a userdefine Map and insert the 10 employee details

Key : empId,name,phone,address,dob,doj,eMail,gender,Sal  
Give the related values for key for each employee

QUESTION 7:

Description: Create a userdefine Map and insert the 10 Student details

Key : stdId,stdName,stdPhone,address,dOB,eMail,gender  
Give the related values for key for each Student

DAY10:

- 1.Exception  
2.throw and throws  
3.User define exception

QUESTIONS(Theory)

- 1.what is Exception?  
2.Explain about types of Exception?  
3.Difference between checked exception and unchecked exception?  
4.What are the differences between exception and error?  
5.What is the super class for Exception and Error?  
6.Exceptions are defined in which java package  
7.What is throw keyword in java?  
8.Can we have try block without catch block?  
9.Can we write multiple catch blocks under single try block?  
10.How to write user defined exception or custom exception in java?  
11.What are the different ways to print exception message on console?  
12.What are the differences between final finally and finalize in java?  
13.Can we write return statement in try and catch blocks?  
14.Can we write return statement in finally block?  
15.What are the differences between throw and throws?  
16.What are the Exception Handling Keywords in Java?  
17.Explain Java Exception Hierarchy?  
18.How to create custom Exception?

QUESTIONS(Programs)

-----  
QUESTION 1:

Description : Find the output for the program:

```
public class Test
{
    public static void main(String[] args)
    {
        try
        {
            System.out.printf("1");
            int sum = 9 / 0;
            System.out.printf("2");
        }
        catch(ArithmaticException e)
        {
            System.out.printf("3");
        }
        catch(Exception e)
        {
            System.out.printf("4");
        }
        finally
        {
            System.out.printf("5");
        }
    }
}
```

QUESTION 2:

Description : Find the output for the program:

```
public class Test
{
    private void m1()
```

```

    {
    m2();
    System.out.printf("1");
    }
    private void m2()
    {
    m3();
    System.out.printf("2");
    }
    private void m3()
    {
    System.out.printf("3");
try
{
    int sum = 4/0;
    System.out.printf("4");
}
catch(ArithmeticException e)
{
    System.out.printf("5");
}

    System.out.printf("7");
}
public static void main(String[] args)
{
    Test obj = new Test();
    obj.m1();
}
}

```

QUESTION 3:

-----  
Description : Find the output for the program:

```

public class Test
{
public static void main(String[] args)
{
    try
    {
        System.out.printf("1");
        int data = 5 / 0;
    }
    catch(ArithmeticException e)
    {
        System.out.printf("2");
        System.exit(0);
    }
    finally
    {
        System.out.printf("3");
    }
    System.out.printf("4");
}
}

```

QUESTION 4:

-----  
Description : Find the output for the program:

```

public class Test
{
public static void main(String[] args)
{
    try
    {

```

```

        System.out.printf("1");
        int data = 5 / 0;
    }
    catch(ArithmeticException e)
    {
        Throwables.throwAsUncheckedException(e);
    }
    finally
    {
        System.out.printf("3");
    }
}
System.out.printf("4");
}
}

```

QUESTION 5:

-----

Description : Find the output for the program:

```

import java.io.EOFException;
import java.io.IOException;

public class Test
{
    public static void main(String[] args)
    {
        try
        {
            System.out.printf("1");
            int value = 10 / 0;
            throw new IOException();
        }
        catch(EOFException e)
        {
            System.out.printf("2");
        }
        catch(ArithmeticException e)
        {
            System.out.printf("3");
        }
        catch(NullPointerException e)
        {
            System.out.printf("4");
        }
        catch(IOException e)
        {
            System.out.printf("5");
        }
        catch(Exception e)
        {
            System.out.printf("6");
        }
    }
}

```

DAY11:

- 1.JDBC Connection  
2.Mysql Introduction

QUESTIONS(Theory)

- 1.What is JDBC?  
2.What is JDBC Driver?  
3.What are the steps to connect to the database in java?  
4.What are the JDBC statements?  
5.What is the difference between Statement and PreparedStatement interface?  
6.what is schema?  
7.what is primary key?  
8.what is constraints?

QUESTIONS(Program)

-----  
Task 1:

Download Sql work bench and do all setup like creating table,assign primary key and retrieve all the values from Demoqa register table(manually).

Input :FirstName,LastName,MaritalStatus(Single,Married,Divorced),  
Hobby(Dance,Reading,Cricket),Country,DateOfBirth,PhoneNumber,  
Username,E-mail,AboutYourself,Password,ConfirmPassword

Output:

Table Created.  
Inserted value in table  
Deleted value in table  
Modified value in table

-----  
Task 2:

Download Sql work bench and do all setup like creating table,assign primary key and retrieve all the values from employee table.

Input :EmpId(PrimaryKey),EmpName,EmpSalary,EmpAddress  
Output:Print all the values from the table

Sample Output:

-----

EmpId	EmpName	EmpSalary	EmpAddress
101	Arun	2000	Chennai
102	Siva	3000	Madurai
103	Varun	2000	Selam
104	Tharun	1000	Erode

## File Operation:

-----

### QUESTIONS(Theory)

-----

- 1.What is mean by File? In which package it is available?
- 2.What are the methods available in File ?
- 3.While creating a file if you not mention the format then under which format it will save the file?
- 4.What are the difference between append and updating the file?
- 5.How to compare paths of two files?
- 6.How to create a new file?
- 7.How to get last modification date of a file?
- 8.How to create a file in a specified directory?
- 9.How to check a file exist or not?
- 10.How to make a file read-only?
- 11.How to renaming a file ?
- 12.How to get a file's size in bytes?
- 13.How to change the last modification time of a file ?
- 14.How to create a temporary file?
- 15.How to append a string in an existing file?
- 16.How to copy one file into another file?
- 17.How to delete a file?
- 18.How to read a file?
- 19.How to write to a file?
- 20.How to create directories recursively?
- 21.How to delete a directory?
- 22.How to get the fact that a directory is empty or not?
- 23.How to get a directory is hidden or not?
- 24.How to print the directory hierarchy?
- 25.How to print the last modification time of a directory?
- 26.How to get the parent directory of a file?
- 27.How to search all files inside a directory?
- 28.How to get the size of a directory?
- 29.How to traverse a directory?
- 30.How to find current working directory?
- 31.How to display root directories in the system?
- 32.How to search for a file in a directory?
- 33.How to display all the files in a directory?
- 34.How to display all the directories in a directory?

### QUESTIONS(Practical)

-----

#### QUESTION 1

-----

NOTE: Create a new file  
write the file with some 10 lines of text.

#### QUESTION 2

-----

NOTE: Retrieve the text from the file  
Check "java" word contains or not?

#### QUESTION 3

-----

NOTE: Find the row count from the file.

#### QUESTION 4

-----

NOTE: Print the last 5 lines from the file.

#### QUESTION 5

-----

NOTE: Print the odd lines from the file.

#### QUESTION 6

-----  
NOTE: Count the number of duplicate words available in the file.

QUESTION 7  
-----

NOTE: Check directory D:\Java is available or not.

QUESTION 8  
-----

NOTE: Check directory D:\Java is available or not.  
If not create new directory.

QUESTION 9  
-----

NOTE: Create sub directory D:\Java\Selenium\Material.

QUESTION 10  
-----

NOTE: Print all the available files from an existing folder.



**QUESTION 1:**

```
-----  
Project      :EmployeeDetails  
Package      :org.emp  
Class        :Employee  
Methods  
:empId(), empName(), empDob(), empPhone(), empEmail(), empAddress()
```

**Description:**

Create an object for employee class and call above methods also follow the all coding standards.

**QUESTION 2:**

```
-----  
Project      :GreensAddress  
Package      :org.add  
Class        :GreensTech  
Methods  
:greensOmr(), greensAdayar(), greensTambaram(), greensVelacherry(), greensAnnaN  
agar()
```

**Description:**

Create an object for GreensTech class and call above methods also follow the all coding standards.

**QUESTION 3:**

```
-----  
Project      :CompanyDetails  
Package      :org.company  
Class        :CompanyInfo  
Methods      :companyName(), companyId(), companyAddress()
```

**Description:**

Create an object for CompanyDetails class and call above methods also follow the all coding standards.

**QUESTION 4:**

```
-----  
Project      :MyPhone  
Package      :org.phone  
Class        :PhoneInfo  
Methods      :phoneName(), phoneMieNum(), Camera(), storage(), osName()
```

**Description:**

Create an object for PhoneInfo class and call above methods also follow the all coding standards.

**QUESTION 5:**

```
-----  
Project      :LanguageDetails  
Package      :org.lang  
  
Class        :LanguageInfo  
Methods      :tamilLanguage(), englishLanguage(), hindiLanguage()  
  
Class        :StateDetails  
Methods      :southIndia(), northIndia()
```

**Description:**

Create an object for LanguageInfo and StateDetails inside the StateDetails class and call both classes methods also follow the all coding standards

The screenshot shows the Eclipse IDE interface with two code editors and a shared Console view.

**Employee.java:**

```
1 package org.emp;
2
3 public class Employee {
4     private void empId() {
5         System.out.println("empid is created ");
6     }
7     private void empName() {
8         System.out.println("empName is created");
9     }
10    private void empDob() {
11        System.out.println("empDob is created"); //empPhone(),empEmail()
12    }
13    private void empPhone() {
14        System.out.println("empPhone is created");
15    }
16    private void empEmail() {
17        System.out.println("empEmail is created");
18    }
19    private void empAddress() {
20        System.out.println("empAddress is created");
21    }
22
23    public static void main(String[] args) {
24        Employee c =new Employee();
25        c.empId();
26        c.empName();
27        c.empDob();
28        c.empPhone();
29        c.empEmail();
30        c.empEmail();
31        c.empAddress();
32    }
33
34 }
```

**GreensTech.java:**

```
1 package org.add;
2
3 public class GreensTech {
4     private void greensOmr() {
5         System.out.println("greensOmr is created");
6     }
7     private void greensAdyar() {
8         System.out.println("greensAdyar is created");
9     }
10    private void greensTambaram() {
11        System.out.println("greensTambaram IS CREATED");
12    }
13    private void greensVelacherry() {
14        System.out.println("greensVelacherry is created");
15    }
16    private void greensAnnanagar() {
17        System.out.println("greensAnnanagar is created");
18    }
19
20    public static void main(String[] args) {
21        GreensTech c =new GreensTech();
22        c.greensOmr();
23        c.greensAdyar();
24        c.greensTambaram();
25        c.greensVelacherry();
26        c.greensAnnanagar();
27    }
28
29 }
30 }
```

**Console View:**

Employee (5) [Java Application] C:\Users\empid is created  
empName is created  
empDob is created  
empPhone is created  
empEmail is created  
empEmail is created  
empAddress is created

GreensTech [Java Application] greensOmr is created  
greensAdyar is created  
greensTambaram IS CREATED  
greensVelacherry is created  
greensAnnanagar is created

The screenshot shows the Eclipse IDE interface with three main panes: Package Explorer, Editor, and Console.

**Package Explorer:** Shows various Java projects and files, including AbstractionMethod, CollegeInformation5, CompanyDetails (with JRE System Library [jre] and src), CoreJavaPractice, DayList, Inheritance, JavaProjectOne, LanguageDetails1, Polymorphism, SeleniumDailyPractice, and SeleniumJan09. The CompanyInfo.java file is selected.

**Editor:** Displays the content of CompanyInfo.java:

```
1 package org.company;
2
3 public class CompanyInfo {
4     private void companyName() {
5         System.out.println("companyName is TCS");
6     }
7     private void companyId() {
8         System.out.println("companyId is 289");
9     }
10    private void companyAddress() {
11        System.out.println("company address is chennai");
12    }
13 }
14
15 public static void main(String[] args) {
16     CompanyInfo c = new CompanyInfo();
17     c.companyName();
18     c.companyId();
19     c.companyAddress();
20 }
21 }
```

**Console:** Shows the output of the CompanyInfo.java run:

```
<terminated> CompanyInfo (1) [Java]
companyName is TCS
companyId is 289
company address is chennai
```

The screenshot shows the Eclipse IDE interface with three main panes: Package Explorer, Editor, and Console.

**Package Explorer:** Shows various Java projects and files, including AbstractionMethod, CollegeInformation5, CoreJavaPractice, DayList, Inheritance, JavaProjectOne, LanguageDetails1, MyPhone (with JRE System Library [jre] and src), org.phone, Polymorphism, SeleniumDailyPractice, and SeleniumJan09. The PhoneInfo.java file is selected.

**Editor:** Displays the content of PhoneInfo.java:

```
1 package org.phone;
2
3 public class PhoneInfo {
4     private void phoneName() {
5         System.out.println("phoneName is REDMINOTE10");
6     }
7
8     private void phoneMeiNum() {
9         System.out.println("phoneMeiNum I IS 8319829189");
10    }
11    private void camera() {
12        System.out.println("PHONE CAMERA IS 15MEGAPIXEL");
13    }
14    private void storage() {
15        System.out.println("PHONE STORAGE IS 6GB RAM--128GB INTERNAL MEMORY");
16    }
17    private void osName() {
18        System.out.println("osName is Android");
19    }
20
21    public static void main(String[] args) {
22        PhoneInfo c = new PhoneInfo();
23        c.phoneName();
24        c.phoneMeiNum();
25        c.camera();
26        c.storage();
27        c.osName();
28    }
29 }
```

**Console:** Shows the output of the PhoneInfo.java run:

```
<terminated> PhoneInfo [Java Application] C:\Users\naveen\p2\pc
phoneName is REDMINOTE10
phoneMeiNum I IS 8319829189
PHONE CAMERA IS 15MEGAPIXEL
PHONE STORAGE IS 6GB RAM--128GB INTERNAL MEMORY
osName is Android
```

eclipse-workspace - LanguageDetails/src/org/lang/StateDetails.java - Eclipse IDE

```

1 package org.lang;
2 public class StateDetails {
3     private void southindia() {
4         // TODO Auto-generated method stub
5         System.out.println("southindia");
6     }
7     private void northindia() {
8         // TODO Auto-generated method stub
9         System.out.println("northindia");
10    }
11    public static void main(String[] args) {
12        StateDetails c = new StateDetails();
13        c.southindia();
14        c.northindia();
15        LanguageInfo d = new LanguageInfo();
16        d.tamilLanguage();
17        d.englishLanguage();
18        d.hindiLanguage();
19    }
20}
21

```

eclipse-workspace - LanguageDetails/src/org/lang/LanguageInfo.java - Eclipse IDE

```

<terminated> StateDetails [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.jst.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\southindia
northindia
tamillanguage
englishlanguage
hindilanguage

```

eclipse-workspace - LanguageDetails/src/org/lang/StateDetails.java - Eclipse IDE

```

1 package org.lang;
2
3 public class StateDetails {
4     public void tamillanguage() {
5         // TODO Auto-generated method stub
6         System.out.println("tamillanguage");
7     }
8     public void englishLanguage() {
9         // TODO Auto-generated method stub
10        System.out.println("englishlanguage");
11    }
12    public void hindiLanguage() {
13        // TODO Auto-generated method stub
14        System.out.println("hindilanguage");
15    }
16    public static void main(String[] args) {
17        LanguageInfo d = new LanguageInfo();
18        d.tamillanguage();
19        d.englishLanguage();
20        d.hindiLanguage();
21    }
22}
23

```

eclipse-workspace - LanguageDetails/src/org/lang/LanguageInfo.java - Eclipse IDE

```

<terminated> StateDetails [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.jst.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\southindia
northindia
tamillanguage
englishlanguage
hindilanguage

```

## QUESTION 6:

-----  
Project :EmployeeInformation  
Package :org.emp  
Class :Employee  
Methods :empName()

```
Package :org.company
Class :Company
Methods :companyName()

Package :org.client
Class :Client
Methods :clientName()

Package :org.project
Class :Project
Methods :projectName()
```

**Description:**

Create an object for all 4 classes inside the Employee class and call all classes methods also follow the all coding standards.

**QUESTION 7:**

---

```
Project :PhoneDetails
Package :org.phone
Class :ExternalStorage
Methods :size()

Class :InternalStorage
Methods :processorName(),ramSize()
```

**Description:**

Create an object for ExternalStorage and InternalStorage inside the InternalStorage class and call both classes methods also follow the all coding standards.

**QUESTION 8:**

---

```
Project :CollegeInformation
Package :org.college
Class :College
Methods :collegeName(),collegeCode(),collegeRank()

Class :Student
Methods :studentName(),studentDept(),studentId()

Class :Hostel
Methods :hostelName()

Class :Dept
Methods :deptName()
```

**Description:**

Create an object for all 4 classes inside the College class and call all classes methods also follow the all coding standards.

eclipse-workspace - EmployeeInformation/src/org/company/Company.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer    LanguageInfo... StateDetail... module-info... Employee.java Company.java

DayOneJava DayOneJava1 DayOneJava12 DayTwoJava EmployeeInformation JRE System Library [JavaSE-17] src org.client Client.java org.company Company.java org.emp Employee.java org.project Project.java module-info.java LanguageDetails JRE System Library [JavaSE-17] src org.lang LanguageInfo.java StateDetails.java module-info.java OneDayJava

```
1 package org.company;
2
3 public class Company {
4     public void companyName() {
5         // TODO Auto-generated method stub
6         System.out.println("companyName");
7     }
8 }
9
```

eclipse-workspace - EmployeeInformation/src/org/project/Project.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer    LanguageInfo... StateDetail... module-info... Employee.java Company.java Client.java Project.java

DayOneJava DayOneJava1 DayOneJava12 DayTwoJava EmployeeInformation JRE System Library [JavaSE-17] src org.client Client.java org.company Company.java org.emp Employee.java org.project Project.java module-info.java LanguageDetails JRE System Library [JavaSE-17] src org.lang LanguageInfo.java StateDetails.java module-info.java OneDayJava

```
1 package org.project;
2
3 public class Project {
4     public void projectName() {
5         // TODO Auto-generated method stub
6         System.out.println("projectName");
7     }
8 }
9
```

Problems Javadoc Declaration Console

<terminated> Employee (1) [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.1.v20211116-empName companyName clientName projectName

Writable Smart Insert 9 : 1 : 156

eclipse-workspace - EmployeeInformation/src/org/emp/Employee.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer

- DayOneJava
- DayOneJava1
- DayOneJava12
- DayTwoJava
- EmployeeInformation
- JRE System Library [JavaSE-17]
- src
  - org.client
    - Client.java
  - org.company
    - Company.java
  - org.emp
    - Employee.java
  - org.project
    - Project.java
  - module-info.java
- LanguageDetails
- JRE System Library [JavaSE-17]
- src
  - org.lang
    - LanguageInfo.java
    - StateDetails.java
    - module-info.java
- OneDayJava

LanguageInfo.java

```
1 package org.emp;
2 import org.company.Company;
3 import org.client.Client;
4 import org.project.Project;
5 public class Employee {
6     private void empName() {
7         // TODO Auto-generated method stub
8         System.out.println("empName");
9     }
10    public static void main(String[] args) {
11        Employee e = new Employee();
12        e.empName();
13        Company c = new Company();
14        c.companyName();
15        Client d = new Client();
16        d.clientName();
17        Project f = new Project();
18        f.projectName();
19    }
20 }
21 }
```

Console

```
<terminated> Employee (1) [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\bin
empName
companyName
clientName
projectName
```

eclipse-workspace - EmployeeInformation/src/org/client/Client.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer

- DayOneJava
- DayOneJava1
- DayOneJava12
- DayTwoJava
- EmployeeInformation
- JRE System Library [JavaSE-17]
- src
  - org.client
    - Client.java
  - org.company
    - Company.java
  - org.emp
    - Employee.java
  - org.project
    - Project.java
  - module-info.java
- LanguageDetails
- JRE System Library [JavaSE-17]
- src
  - org.lang
    - LanguageInfo.java
    - StateDetails.java
    - module-info.java
- OneDayJava

Client.java

```
1 package org.client;
2
3 public class Client {
4     public void clientName() {
5         // TODO Auto-generated method stub
6         System.out.println("clientName");
7     }
8 }
```

Console

```
<terminated> Employee (1) [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\bin\javaw.exe (22)
empName
companyName
clientName
projectName
```

The screenshot shows the Eclipse IDE interface with the following details:

- Java Editor:** The main window displays the code for `InternalStorage.java`. The code prints "processorName" and "ram size" to the console.
- Console View:** Below the editor, the `Console` tab is active, showing the output:

```
<terminated> InternalStorage [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\processorName
ram size
size
```
- Bottom Status Bar:** Shows the status bar with the text "Writable", "Smart Insert", and "20 · 1 · 455".

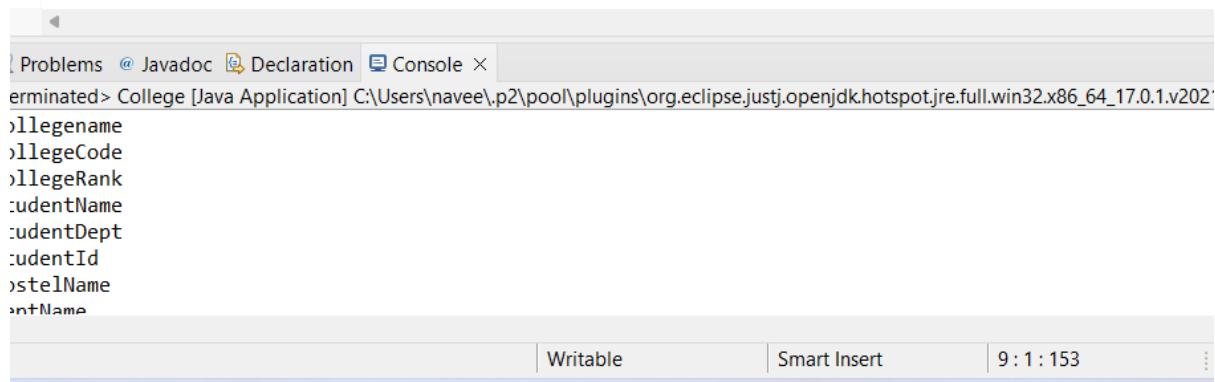
The screenshot shows the Eclipse IDE interface with the following details:

**Top Bar:** Shows tabs for LanguageInfo..., StateDetail..., module-info..., Project.java, module-info..., ExternalSto..., InternalSto..., and a file named "16".

**Editor Area:** Displays the Java code for `ExternalStorage.java`:1 package org.phone;
2
3 public class ExternalStorage {
4 public void size() {
5 // TODO Auto-generated method stub
6 System.out.println("size");
7 }
8 public static void main(String[] args) {
9 ExternalStorage c=new ExternalStorage();
10 c.size();
11 }
12 }
13

**Console Area:** Shows the output of the application's execution:<terminated> InternalStorage [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.1.v20211116-1657
processorName
ram size
size

```
1 package org.college;
2
3 public class Hostel {
4@ 4 public void hostelName() {
5     // TODO Auto-generated method stub
6 System.out.println("HostelName");
7 }
8 }
9
```



```
1 package org.college;
2
3 public class Student {
4     public void studentName() {
5         // TODO Auto-generated method stub
6         System.out.println("studentName");
7     }
8     public void studentDept() {
9         // TODO Auto-generated method stub
10        System.out.println("StudentDept");
11    }
12    public void studentId() {
13        // TODO Auto-generated method stub
14        System.out.println("StudentId");
15    }
16 }
17 }
```

```
Problems @ Javadoc Declaration Console ×
<terminated> College [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211
collegeName
collegeCode
collegeRank
studentName
StudentDept
StudentId
HostelName
deptName
```

The screenshot shows the Eclipse IDE interface with three code editors and their respective Javadoc and declaration views.

**Top Editor:**

```

1 package org.college;
2
3 public class College {
4     private void collegeName() {
5         // TODO Auto-generated method stub
6         System.out.println("collegeName");
7     }
8     private void collegeCode() {
9         // TODO Auto-generated method stub
10        System.out.println("collegeCode");
11    }
12    private void collegeRank() {
13        // TODO Auto-generated method stub
14        System.out.println("collegeRank");
15    }
16    public static void main(String[] args) {
17        College c=new College();
18        c.collegeName();
19        c.collegeCode();
20        c.collegeRank();
21        Student d =new Student();
22        d.studentName();
23        d.studentDept();
24        d.studentId();
25        Hostel f = new Hostel();
26        f.hostelName();
27        Department g =new Department();
28        g.deptName();
29    }
30 }
31

```

**Javadoc View (Top Right):**

- org.college
- College
  - collegeName(): void
  - collegeCode(): void
  - collegeRank(): void
  - main(String[]): void

**Bottom Editor:**

```

1 package org.college;
2
3 public class Department {
4     public void deptName() {
5         // TODO Auto-generated method stub
6         System.out.println("deptName");
7     }
8 }
9

```

**Javadoc View (Bottom Right):**

- org.college
- Department
- deptName(): void

**Common Problems View:**

- terminated> College [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.1.v20211116-1657\jre\bin\javaw.exe (22-Dec-2021, 4:41:46 pm - 4:41:46 pm)
- collegeName
- collegeCode
- collegeRank
- studentName
- StudentDept
- StudentId
- hostelName
- deptName

QUESTION 9:

```
-----  
Project      :VehicleInformation  
Package      :org.allvehicle  
Class        :Vehicle  
Methods       :VehicleNecessery()  
  
Package      :org.twowheeler  
Class        :TwoWheller  
Methods       :bike(),cycle()  
  
Package      :org.threewheeler  
Class        :ThreeWheeler  
Methods       :Auto()  
  
Package      :org.fourwheeler  
Class        :FourWheeler  
Methods       :car(),bus(),lorry()
```

**Description:**

Create an object for all 4 classes inside the Vehicle class and call all classes methods also follow the all coding standards.

The screenshot shows the Eclipse IDE interface with the following details:

- Editor Area:** Displays the Java code for the `FourWheeler` class. The code includes three methods: `car()`, `bus()`, and `lorry()`, each printing a specific vehicle type to `System.out`.
- Console Area:** Shows the output of the code execution. The output is:

```
vehicelNecessary
bike
cycle
tyre
car
bus
lorry
```

The screenshot shows a Java application running in the Eclipse IDE. The code editor displays a file named `ThreeWheeler.java` with the following content:

```
1 package org.threewheeler;
2
3 public class ThreeWheeler {
4     public void tyre() {
5         // TODO Auto-generated method stub
6         System.out.println("tyre");
7     }
8 }
```

The console tab is active, showing the output of the application's execution:

```
vehicelNecessary
bike
cycle
tyre
car
bus
lorry
```

```
1 package org.twowheeler;
2
3 public class TwoWheeler {
4@ 4@ public void bike() {
5     // TODO Auto-generated method stub
6 System.out.println("bike");
7 }
8@ public void cycle() {
9     // TODO Auto-generated method stub
10 System.out.println("cycle");
11 }
12 }
13
```

```
Problems @ Javadoc Declaration Console X
<terminated> Vehicle [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\bin\vehicleNecessary
bike
cycle
tyre
car
bus
lorry
```

The screenshot shows the Eclipse IDE interface. The top half displays a Java code editor with the following code:

```
1 package org.allvehicle;
2 import org.allvehicle.Vehicle;
3 import org.two Wheeler.TwoWheeler;
4 import org.threewheeler.ThreeWheeler;
5 import org.fourwheeler.FourWheeler;
6 public class Vehicle {
7     private void vehicleNecessary() {
8         // TODO Auto-generated method stub
9         System.out.println("vehicleNecessary");
10    }
11    public static void main(String[] args) {
12        Vehicle c = new Vehicle();
13        c.vehicleNecessary();
14        TwoWheeler d=new TwoWheeler();
15        d.bike();
16        d.cycle();
17        ThreeWheeler f = new ThreeWheeler();
18        f.tyre();
19        FourWheeler e= new FourWheeler();
20        e.car();
21        e.bus();
22        e.lorry();
23    }
24 }
25 }
```

The bottom half shows the Eclipse Console tab with the following output:

```
<terminated> Vehicle [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\bin\javaw.exe (22-D)
vehicleNecessary
bike
cycle
tyre
car
bus
lorry
```

QUESTION 9:

```
-----  
Project :VehicleInformation  
Package :org.allvehicle  
Class :Vehicle  
Methods :VehicleNecessery()  
  
Package :org.twowheeler  
Class :TwoWheeler  
Methods :bike(),cycle()  
  
Package :org.threewheeler  
Class :ThreeWheeler  
Methods :Auto()  
  
Package :org.fourwheeler  
Class :FourWheeler  
Methods :car(),bus(),lorry()
```

Description:

Create an object for all 4 classes inside the Vehicle class and call all classes methods also follow the all coding standards.

QUESTION 10:

```
-----  
Project :TransportInformation  
Package :org.transport  
Class :Transport  
Methods :TransportForm  
  
Package :org.road  
Class :Road  
Methods :bike(),cycle(),bus(),car()  
  
Package :org.air  
Class :Air  
Methods :aeroPlane(),heliCopter()
```

Package :org.water  
Class :Water  
Methods :boat(), ship()

**Description:**

Create an object for all 4 classes inside the Transport class and call all classes methods also follow the all coding standards.

```
1 package org.air;
2
3 public class Air {
4     // TODO Auto-generated method stub
5     System.out.println("aeroplane");
6 }
7
8 public void helicopter() {
9     // TODO Auto-generated method stub
10 System.out.println("helicopter");
11 }
12 }
13 |
```

Problems @ Javadoc Declaration Console ×  
<terminated> Transport [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.1.v20211116-1657\jre\bin  
bus  
car  
aeroplane  
helicopter  
boat  
ship

Writable Smart Insert 13 : 1 : 247 :

The image displays three separate Eclipse IDE windows, each showing a Java code editor, a Problems view, a Javadoc view, a Declaration view, and a Console view.

**Top Window (org.road package):**

```
1 package org.road;
2
3 public class Road {
4     // TODO Auto-generated method stub
5     System.out.println("bike");
6 }
7
8     public void cycle() {
9         // TODO Auto-generated method stub
10    System.out.println("cycle");
11 }
12
13     public void car() {
14         // TODO Auto-generated method stub
15    System.out.println("car");
16 }
17
18     public void bus() {
19         // TODO Auto-generated method stub
20    System.out.println("bus");
21 }
```

**Middle Window (org.water package):**

```
1 package org.water;
2
3 public class Water {
4     public void boat() {
5         // TODO Auto-generated method stub
6     System.out.println("boat");
7 }
8
9     public void ship() {
10        // TODO Auto-generated method stub
11    System.out.println("ship");
12 }
```

**Bottom Window (org.road package):**

```
bus
car
aeroplane
helicopter
boat
ship
```

**Bottom Window (org.water package):**

```
bus
car
aeroplane
helicopter
boat
ship
```

QUESTION 11:

```
-----  
Project      :NetworkInformation  
Package      :org.network  
Class        :Wifi  
Methods      :wifiName()  
  
Class        :MobileData  
Methods      :dataName()  
  
Class        :Lan  
Methods      :lanName()  
  
Class        :Wireless  
Methods      :modamName()
```

Description:

Create an object for all 4 classes inside the Wifi class and call all classes methods also follow the all coding standards.

```
1 package org.network;
2
3 public class Lan {
4     public void lanName() {
5         // TODO Auto-generated method stub
6         System.out.println("lanName");
7     }
8 }
9 |
```

Problems Javadoc Declaration Console ×  
<terminated> Wifi [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.1.v20211116-1657\jre\bin\javaw.exe  
wifiName  
dataName  
lanName  
modamName

```
1 package org.network;
2
3 public class Wifi {
4     private void wifiName() {
5         // TODO Auto-generated method stub
6         System.out.println("wifiName");
7     }
8     public static void main(String[] args) {
9         Wifi c = new Wifi();
10        c.wifiName();
11        MobileData d = new MobileData();
12        d.dataName();
13        Lan f = new Lan();
14        f.lanName();
15        Wireless g = new Wireless();
16        g.modamName();
17    }
18 }
19 |
```

Problems Javadoc Declaration Console ×  
<terminated> Wifi [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.1.v20211116-1657\jre\bin\javaw.exe  
wifiName  
dataName  
lanName  
modamName

```
1 package org.network;
2
3 public class MobileData {
4     public void dataName() {
5         // TODO Auto-generated method stub
6         System.out.println("dataName");
7     }
8 }
9
```

Problems Javadoc Declaration Console ×  
<terminated> Wifi [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.1.v20211116-1657\jre\bin\javaw.exe  
wifiName  
dataName  
lanName  
modamName

```
1 package org.network;
2
3 public class Wireless {
4     public void modamName() {
5         // TODO Auto-generated method stub
6         System.out.println("modamName");
7     }
8 }
9
```

Problems Javadoc Declaration Console ×  
<terminated> Wifi [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.1.v20211116-1657\jre\bin\javaw.exe  
wifiName  
dataName  
lanName  
modamName



## CORE JAVA DAY—2 TASK

### QUESTION 1:

Description: Using Scanner class get the below details  
empId  
empName  
empEmail  
empPhoneno  
empSalary  
empGender  
empCity

The screenshot shows an IDE interface with two panes. On the left, the code for `ScannerClass.java` is displayed:

```
1 package org.wipro;
2
3 import java.util.Scanner;
4
5 public class ScannerClass {
6     public static void main(String[] args) {
7         Scanner o = new Scanner(System.in);
8         System.out.println("emp id is:");
9         int id = o.nextInt();
10        System.out.println("emp name is:");
11        String name=o.next();
12        System.out.println("empEmail is");
13        String mail=o.next();
14        System.out.println("empPhoneno is:");
15        long phone =o.nextLong();
16        System.out.println("empSalary is:");
17        int salary =o.nextInt();
18        System.out.println("empGender is:");
19        String gender=o.next();
20        System.out.println("empCity is:");
21        String city=o.next();
22
23
24
25    }
26 }
```

The right pane shows the console output:

```
<terminated> ScannerClass [Java Application]
emp id is:
1
emp name is:
NAVEEN
empEmail is
naveen.jan9@gmail.com
empPhoneno is:
8344350473
empSalary is:
56000
empGender is:
male
empCity is:
chennai
```

### QUESTION 2:

Description: Using Scanner class get the below details

studentId  
studentName  
Mark1  
Mark2  
Mark3  
Mark4  
Mark5

:Find the total and average of marks

The screenshot shows an IDE interface with two tabs: 'ScannerClass.java' and 'Console'. The code in 'ScannerClass.java' is as follows:

```
1 package org.wipro;
2
3 import java.util.Scanner;
4
5 public class ScannerClass {
6     public static void main(String[] args) {
7         Scanner o = new Scanner(System.in);
8         System.out.println(" studentId is:");
9         int id = o.nextInt();
10        System.out.println("studentName is:");
11        String name=o.next();
12        System.out.println(" Mark1 is:");
13        int m1 =o.nextInt();
14        System.out.println(" Mark2 is:");
15        int m2 =o.nextInt();
16        System.out.println(" Mark3 is:");
17        int m3 =o.nextInt();
18        System.out.println(" Mark4 is:");
19        int m4 =o.nextInt();
20        System.out.println(" Mark5 is:");
21        int m5 =o.nextInt();
22        float total;
23        float average;
24        total = m1+m2+m3+m4+m5;
25        average = (float)(total / 5.0);
26        System.out.println("the total is "+total);
27        System.out.println("the average is"+average);
28    }
29 }
30 }
```

The 'Console' tab shows the output of running the program:

```
<terminated> ScannerClass [Java]
studentId is:
1
studentName is:
NAVEEN
Mark1 is:
98
Mark2 is:
90
Mark3 is:
67
Mark4 is:
45
Mark5 is:
67
the total is 367.0
the average is73.4
```

QUESTION 3:

```
-----
package name: org.all
Project name: LanguageDetails
Class name   : Languageclass
Methods      : alllanguage

package name: org.tamil
Project name: LanguageDetails
Class name   : Tamil
Methods      : tamillanguage

package name: org.english
Project name: LanguageDetails
Class name   : English
Methods      : englishlanguage

package name: org.telgu
Project name: LanguageDetails
Class name   : Telgu
Methods      : telgulanguage
```

Description:

create above 4 packages and call all your class methods into the Languageclass using multilevel inheritance.

```

languageclass.java
1 package org.all;
2 import org.tamil.Tamil;
3 public class Languageclass extends Tamil {
4     public void alllanguage() {
5         System.out.println("all language class tamil,english,telgu");
6     }
7 }
8 public static void main(String[] args) {
9     Languageclass c = new Languageclass();
10    c.alllanguage();
11    c.tamillanguage();
12    c.englishlanguage();
13    c.telgulanguage();
14 }
15 }

Tamil.java
1 package org.tamil;
2 import org.english.English;
3 public class Tamil extends English {
4     public void tamillanguage() {
5         System.out.println("tamillanguage");
6     }
7 }
8

```

```

English.java
1 package org.english;
2 import org.telgu.telgu;
3 public class English extends Telgu {
4     public void englishlanguage() {
5         System.out.println("englishlanguage");
6     }
7 }
8

Telugu.java
1 package org.telgu;
2
3 public class Telgu {
4     public void telgulanguage() {
5         System.out.println("telgulanguage");
6     }
7 }
8

```

Console

```

<terminated> Languageclass [Java Application] C:\Users\nav
all language class tamil,english,telgu
tamillanguage
englishlanguage
telgulanguage

```

#### QUESTION 4:

```

-----
package name: org.india
Project name: SouthIndia
Class name   : India
Methods      : india

package name: org.tamilnadu
Project name: SouthIndia
Class name   : TamiladuN
Methods      : tamillanguage

package name: org.kerala
Project name: SouthIndia
Class name   : kerala
Methods      : malayalam

package name: org.andrapradesh
Project name: SouthIndia
Class name   : AndhraPradesh
Methods      : telugu

```

Description:

create above 4 packages and call all your class methods into the India using multilevel inheritance.

```

Project Explorer
AbstractionMethod
CoreJavaPractice
DayList
Inheritance
JavaProjectOne
LanguageDetails1
Polymorphism
SeleniumDailyPractice
SeleniumJan09
SouthIndia5
JRE System Library [jre]
src
  org.andrapradesh
    AndhraPradesh.java
  org.india
    India.java
  org.kerala
    Kerala.java
  org.tamilnadu
    Tamilnadu.java

India.java
package org.india;
import org.tamilnadu.Tamilnadu;
public class India extends Tamilnadu {
    public void India() {
        System.out.println("all india states of india");
    }
    public static void main(String[] args) {
        India c = new India();
        c.India();
        c.tamillanguage();
        c.malayalam();
        c.telugu();
    }
}

Tamilnadu.java
package org.tamilnadu;
import org.kerala.Kerala;
public class Tamilnadu extends Kerala {
    public void tamillanguage() {
        System.out.println("tamillanguage");
    }
}

Kerala.java
package org.kerala;
import org.andrapradesh.AndhraPradesh;
public class Kerala extends AndhraPradesh {
    public void malayalam() {
        System.out.println("malayalam");
    }
}

AndhraPradesh.java
package org.andrapradesh;
public class AndhraPradesh {
    public void telugu() {
        System.out.println("telugu");
    }
}

```

#### QUESTION 5:

Project	:CollegeInformation
Package	:org.college
Class	:College
Methods	:collegeName(), collegeCode(), collegeRank()
Class	:Student
Methods	:studentName(), studentDept(), studentId()
Class	:Hostel
Methods	:HostelName()
Class	:dept
Methods	:deptName()

#### Description:

create above 4 class and call all your class methods into the Student using multilevel inheritance.

```

Student.java
1 package org.college;
2
3 public class Student extends College {
4     public void studentName() {
5         System.out.println("NAVEEN");
6     }
7     public void studentDept() {
8         System.out.println("MECHANICAL ENGINEERING");
9     }
10    public void studentId() {
11        System.out.println("98989889");
12    }
13    public static void main(String[] args) {
14        Student c = new Student();
15        c.studentName();
16        c.studentId();
17        c.studentDept();
18        c.hostelName();
19        c.deptName();
20    }
}

Collegejava.java
1 package org.college;
2
3 public class College extends Hostel {
4     public void collegeName() {
5         System.out.println("SRM UNIVERSITY -CHENNAI");
6     }
7     public void collegeCode() {
8         System.out.println("2108006");
9     }
10    public void collegeRank() {
11        System.out.println("1");
12    }
13
14 }

dept.java
1 package org.college;
2
3 public class dept {
4     public void deptName() {
5         System.out.println("mech engg");
6     }
7
8 }

HostelJava.java
1 package org.college;
2
3 public class Hostel extends dept {
4     public void hostelName() {
5         System.out.println("kctc boys hostel");
6     }
7
8 }

```

Console output:

```

<terminated> Student (1) [Java Application] C:\Users\ naveen\p
NAVEEN
98989889
MECHANICAL ENGINEERING
kctc boys hostel
mech engg
2108006
SRM UNIVERSITY -CHENNAI
1

```

#### QUESTION 6:

```

-----
Project      : Computer
Class        : Computer
Methods      : computerModel()

Class        : Desktop
Methods      : desktopSize()

```

#### Description:

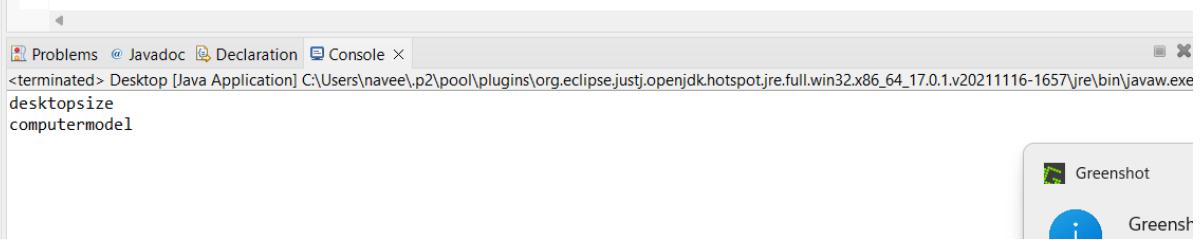
create above 2 class and call all your class methods into the Desktop using single inheritance.

```
1 package org.computer;
2
3 public class Desktop extends Computer {
4     public void deskTopSize() {
5         // TODO Auto-generated method stub
6         System.out.println("desktopsize");
7     }
8     public static void main(String[] args) {
9         Desktop c = new Desktop();
10        c.deskTopSize();
11        c.computerModel1();
12    }
13}
14
```

The screenshot shows the Eclipse IDE interface. At the top, there's a toolbar with icons for file operations like New, Open, Save, and Run. Below the toolbar is a menu bar with options like File, Edit, Select, Insert, View, Tools, Window, Help, and Preferences. The main area consists of two panes: the left pane contains the Java code for the `Desktop` class, and the right pane is the `Console`. The `Console` tab is selected, showing the output of the `main` method:

Problems @ Javadoc Declaration Console <terminated> Desktop [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_17  
desktopsize  
computermodel

```
1 package org.computer;
2
3 public class Computer {
4     public void computerModel1() {
5         // TODO Auto-generated method stub
6         System.out.println("computermodel");
7     }
8 }
9
```



**QUESTION 6:**

```
-----
Project      :CComputer
Class        :Computer
Methods      :computerModel()

Class        :Desktop
Methods      :desktopSize()
```

Description:

create above 2 class and call all your class methods into the Desktop using single inheritance.

**QUESTION 7:**

```
-----
Project      :LanguageDetails
Package      :org.lang
Class        :LanguageInfo
Methods      :tamilLanguage(),englishLanguage(),hindiLanguage()

Class        :StateDetails
Methods      :southIndia(),northIndia()
```

Description:

create above 2 class and call all your class methods into the LanguageInfo using single inheritance.

```

LanguageInfo.java
1 package org.lang;
2
3 public class LanguageInfo extends StateDetails {
4     private void tamillanguage() {
5         System.out.println("tamil language");
6     }
7     private void englishlanguage() {
8         System.out.println("english language");
9     }
10    private void hindilanguage() {
11        System.out.println("hindi language");
12    }
13
14    public static void main(String[] args) {
15        LanguageInfo c = new LanguageInfo();
16        c.tamillanguage();
17        c.englishlanguage();
18        c.hindilanguage();
19        c.southIndia();
20        c.northIndia();
21    }
22}
23
24

StateDetails.java
1 package org.lang;
2
3 public class StateDetails {
4     public void southIndia() {
5         System.out.println("southIndia --- madurai");
6     }
7     public void northIndia() {
8         System.out.println("northIndia --- delhi");
9     }
10}
11
12

Console
<terminated> LanguageInfo (Java Application) C:\Users\navee\p2\pool\plugins\org.eclipse.jdt.core\openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\bin\javaw.exe (21-Jan-2022)
tamil language
english language
hindi language
southIndia
southIndia --- madurai
northIndia
northIndia --- delhi

```

#### QUESTION 9:

Project	:BankDetails
Package	:org.bank
Class	:BankInfo
Methods	:saving(), fixed()
 Class	:AxisBank
Methods	:deposit()

#### Description:

create above 2 class and call all your class methods into the BankInfo using single inheritan

```

BankInfo.java
1 package org.bank;
2
3 public class BankInfo extends AxisBank {
4     private void saving() {
5         System.out.println("saving bank account");
6     }
7     private void fixed() {
8         System.out.println("fixed bank account");
9     }
10
11    public static void main(String[] args) {
12        BankInfo c = new BankInfo();
13        c.saving();
14        c.fixed();
15        c.deposit();
16    }
17}
18

AxisBank.java
1 package org.bank;
2
3 public class AxisBank {
4     public void deposit() {
5         System.out.println("deposit amount is ---10000");
6     }
7
8}
9

Console
<terminated> BankInfo [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.jdt.core\openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\bin\javaw.exe (21-Jan-2022, 4:00:00 AM)
saving bank account
fixed bank account
deposit amount is ---10000

```

#### QUESTION 10:

Project	:CompanyDetails
Package	:org.company
Class	:Company
Methods	:companyName()
 Package	:org.client
Class	:Client
Methods	:clientName()

#### Description:

create above 2 packages and call all your class methods into the Comapany using single inheritance.

The screenshot shows the Eclipse IDE interface with the following components:

- Package Explorer:** Shows various Java projects and files, including `AbstractionMethod`, `CollegeInformation5`, `CompanyDetails`, `JRE System Library [jre]`, `src` (containing `org.client` and `org.company`), `CoreJavaPractice`, `DayList`, `Inheritance`, `JavaProjectOne`, `LanguageDetails1`, `Polymorphism`, `SeleniumDailyPractice`, and `SeleniumJan09`.
- Editor Area:** Two tabs are open:
  - Company.java:** Contains the following code:

```
1 package org.company;
2 import org.client.Client;
3 public class Company extends Client {
4     private void companyName() {
5         System.out.println("verizon info tech india");
6     }
7     public static void main(String[] args) {
8         Company c = new Company();
9         c.companyName();
10        c.clientName();
11    }
12 }
13
```
  - Client.java:** Contains the following code:

```
1 package org.client;
2
3 public class Client {
4     public void clientName() {
5         System.out.println("sathyamootrh");
6     }
7 }
8
```
- Console:** Displays the output of the executed code:

```
<terminated> Company (7) Java Application C:\verizon info tech india
sathyamootrh
```

#### QUESTION 11:

-----

Project	:EducationInformation
Package	:org.edu
Class	:Education
Methods	:ug(),pg()
Class	:Arts
Methods	:bsc(),bEd(),bA(),bBA()
Class	:Engineering
Methods	:bE(),bTech()
Class	:Medicine
Methods	:physiyo(),dental(),mbbs()

#### Description:

create above 4 class and call all your class methods into the Education using multilevel inheritance.

**Top Left Window (Arts.java):**

```

1 package org.edu;
2
3 public class Arts extends Engineering{
4     public void bSc() {
5         System.out.println("bachelor of science");
6     }
7     public void bEd() {
8         System.out.println("bachelor of education");
9     }
10    public void bA() {
11        System.out.println("bachelor of arts");
12    }
13    public void bBa() {
14        System.out.println("bachelor of business administration");
15    }
16 }
17
18

```

**Top Right Window (Education.java):**

```

1 package org.edu;
2
3 public class Education extends Arts {
4     private void ug() {
5         System.out.println("under graduate");
6     }
7     private void pg() {
8         System.out.println("post graduate");
9     }
10    public static void main(String[] args) {
11        Education c = new Education();
12        c.ug();
13        c.pg();
14        c.bA();
15        c.bBa();
16        c.bSc();
17        c.bBa();
18        c.bf();
19        c.bTech();
20        c.dental();
21        c.mbb();
22        c.physio();
23    }
24
25
26
27

```

**Middle Left Window (Medicine.java):**

```

1 package org.edu;
2
3 public class Medicine {
4     public void physio() {
5         System.out.println("physiotherapy");
6     }
7     public void dental() {
8         System.out.println("dental");
9     }
10    public void mbbs() {
11        System.out.println("bachelor of medicine");
12    }
13 }
14
15
16

```

**Middle Right Window (Engineering.java):**

```

1 package org.edu;
2
3 public class Engineering extends Medicine {
4     public void bE() {
5         System.out.println("bachelor of engineering");
6     }
7     public void bTech() {
8         System.out.println("bachelor of technology");
9     }
10
11
12

```

**Bottom Center Window (Console):**

```

<terminated> Education (1) [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.jdt.openjdk.
under graduate
post graduate
bachelor of arts
bachelor of business administration
bachelor of sciences
bachelor of business administration
bachelor of engineering
bachelor of technology
dental
bachelor of medicine bachelor of surgiene
physiotherapy

```

## DAY 3 –JAVA TASK

### QUESTION 2:

```
-----  
Project      :EmployeeDetails  
Package      :org.emp  
Class        :Employee  
Methods       :empId()
```

#### Description

You have to overload the method empId() based on different datatype in arguments.

The screenshot shows the Eclipse IDE interface. On the left, the 'Console' view displays the output of the Java application, which includes several lines of text representing the output of different method calls. On the right, the 'JavaPracticeTask.java' file is open in the editor, showing the source code with four overloaded methods for the 'empId' function.

```
<terminated> JavaPracticeTask [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openj9  
NAVEEN  
84928498  
723|  
M  
56.77  
chennai 600049  
600034 trichy  
  
JavaPracticeTask.java  
1 package org.com;  
2  
3 public class JavaPracticeTask {  
4     private void empId(String name) {  
5         System.out.println(name);  
6     }  
7     private void empId(int i) {  
8         System.out.println(i);  
9     }  
10    private void empId(int i,char c,float f) {  
11        System.out.println(i + "\n" +c);  
12        System.out.println(f);  
13    }  
14    private void empId(String City,int Pincode) {  
15        System.out.println(City + "\t" +Pincode);  
16    }  
17    private void empId(int Pincode,String City) {  
18        System.out.println(Pincode + "\t" +City);  
19    }  
20    public static void main(String[] args) {  
21        JavaPracticeTask e= new JavaPracticeTask();  
22        e.empId("NAVEEN");  
23        e.empId(84928498);  
24        e.empId(723,'M',56.77F);  
25        e.empId("chennai",600049);  
26        e.empId(600034,"trichy");  
27    }  
28 }  
29 }
```

### QUESTION 3:

```
-----  
Project      :CompanyDetails  
Package      :org.company  
Class        :CompanyInfo  
Methods       :companyName()
```

#### Description

You have to overload the method companyName() based on different Number of arguments.

The screenshot shows an IDE interface with two windows. On the left is the code editor for `CompanyInfo.java`, which contains Java code for printing company names and details. On the right is the `Console` window showing the execution output.

```

1 package org.company;
2
3 public class CompanyInfo {
4
5
6    private void companyName(int g, String k) {
7        System.out.println(g + "\n" + k);
8    }
9    private void CompanyName(int i) {
10        System.out.println(i);
11    }
12    private void CompanyName(int i, String s) {
13        System.out.println(i + "\t" + s);
14    }
15    private void CompanyName(String j, int o) {
16        System.out.println(j + "\t" + o);
17    }
18    public static void main(String[] args) {
19        CompanyInfo r = new CompanyInfo();
20        r.CompanyName(1, "cts");
21        r.CompanyName(7877979);
22        r.CompanyName(16, "east street,karappak,navalur,chenai");
23        r.CompanyName("NAVEEN", 676768);
24    }
25 }
26

```

`<terminated> CompanyInfo [Java Application] C:\Users\nav`

```

1 cts
7877979
16 east street,karappak,navalur,chenai
NAVEEN 676768

```

#### QUESTION 4:

-----  
 Project :MyPhone  
 Package :org.phone  
 Class :Phone  
 Methods :phoneInfo()

#### Description

You have to overload the method `phoneInfo()` based on different datatype order in arguments.

The screenshot shows an IDE interface with two windows. On the left is the code editor for `Phone.java`, which contains Java code for overloading the `phoneInfo()` method. On the right is the `Console` window showing the execution output.

```

1 package org.phone;
2
3 public class Phone {
4    private void phoneInfo(String s) {
5        System.out.println(s);
6    }
7    private void phoneInfo(int i) {
8        System.out.println(i);
9    }
10   private void phoneInfo(int i, char c) {
11       System.out.println(i + "\t" + c);
12   }
13   private void phoneInfo(String s, int pincode) {
14       System.out.println(s + "\t" + pincode);
15   }
16   public static void main(String[] args) {
17       Phone c = new Phone();
18       c.phoneInfo("REDMI NOTE TEN");
19       c.phoneInfo(1);
20       c.phoneInfo(1, 'R');
21       c.phoneInfo("chennai", 600019);
22   }
23 }
24

```

`<terminated> Phone`

```

REDMI NOTE TEN
1
1 R
chennai 600019

```

#### QUESTION 5:

-----  
 Project :GreensAddress

```
Package :org.add
Class :GreensTech
Methods :greensOmr()
```

Description

You have to overload the method greensOmr() based on order,type,number.

The screenshot shows an IDE interface with two tabs: 'GreensTech.java' and 'Console'. The code in 'GreensTech.java' is as follows:

```
1 package org.add;
2
3 public class GreensTech {
4     private void greensOmr(String name) {
5         System.out.println(name);
6     }
7     private void greensOmr(int i) {
8         System.out.println(i);
9     }
10    private void greensOmr(int i, String C) {
11        System.out.println(i+"\n"+C);
12    }
13    private void greensOmr(String s, int pincode) {
14        System.out.println(s+"\t"+pincode);
15    }
16    public static void main(String[] args) {
17        GreensTech t = new GreensTech();
18        t.greensOmr("GREENS TECHNOLOGY");
19        t.greensOmr(1); // Line 19 highlighted in blue
20        t.greensOmr(1, "selenium");
21        t.greensOmr("chennai", 600049);
22    }
23 }
```

The 'Console' tab shows the output of the program:

```
<terminated> GreensTech
GREENS TECHNOLOGY
1
1
selenium
chennai 600049
```

QUESTION 6:

```
-----
Project :BankDetails
Package :org.bank
Class :BankInfo
Methods :saving(),fixed(),deposit()

Class :AxisBank
Methods :deposit()
```

Description:

You have to override the method deposit in AxisBank.

The screenshot shows the Eclipse IDE interface with three main panes:

- Top Left:** A code editor titled "AxisBank.java" containing the following Java code:

```
1 package org.bank;
2
3 public class AxisBank extends BankInfo {
4     private void deposit() {
5         System.out.println("100");
6     }
7 }
8 public static void main(String[] args) {
9     AxisBank c = new AxisBank();
10    c.deposit();
11 }
12 }
13 }
```

- Bottom Left:** A code editor titled "Console" showing the output of the application:

```
<terminated> AxisBank (1) [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.h
100
```

- Top Right:** A code editor titled "BankInfo.java" containing the following Java code:

```
1 package org.bank;
2
3 public class BankInfo {
4     private void saving() {
5         System.out.println("5.54");
6     }
7     private void fixed() {
8         System.out.println("8.99");
9     }
10 }
11 private void deposit() {
12     System.out.println("100");
13 }
14 }
15 }
16 }
```

#### QUESTION 7:

-----

Project	:EducationInformation
Package	:org.edu
Class	:Education
Methods	:ug(), pg()

Class	:Arts
Methods	:bSc(), bEd(), bA(), bBA(), ug(), pg()

#### Description:

You have to override the method ug(), pg() in Arts.

```
Education.java
1 package org.edu;
2
3 public class Education {
4     private void ug() {
5         System.out.println("under graduate");
6     }
7     private void pg() {
8         System.out.println("post graduate");
9     }
10 }
11
12

Arts.java
1 package org.edu;
2
3 public class Arts extends Education {
4     private void bSc() {
5         System.out.println("bachelor of science");
6     }
7     private void bEd() {
8         System.out.println("bachelor of education");
9     }
10    private void bA() {
11        System.out.println("bachelor of science");
12    }
13    private void bBa() {
14        System.out.println("bachelor of business admino");
15    }
16
17    public void ug() {
18        System.out.println("under graduate");
19    }
20    private void pg() {
21        System.out.println("post graduate");
22    }
23    public static void main(String[] args) {
24        Arts c = new Arts();
25        c.ug();
26        c.pg();
27    }
28 }
```

Console

```
<terminated> Arts (1) [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\bin\javaw.exe (20-Jan-2022, under graduate post graduate)
```

QUESTION 8:

```
-----  
Project      :UniversityInformation  
Package      :org.univ  
Class        :University  
Methods      :ug(),pg()  
  
Class        :College  
Methods      :ug(),pg()
```

Description:

ug(),pg() is just a template in University class and You have to override the method ug(),pg() in College class.

```
Education.java
1 package org.univ;
2
3 public class Education {
4     private void ug() {
5         System.out.println("UNDER GRADUATE");
6     }
7 }
8 private void pg() {
9     System.out.println("POST GRADUATE");
10}
11
12}
13
```

```
College.java
1 package org.univ;
2
3 public class College extends Education {
4     private void ug() {
5         System.out.println("UNDER GRADUATE");
6     }
7     private void pg() {
8         System.out.println("post graduate");
9     }
10    public static void main(String[] args) {
11
12
13        College c = new College();
14        c.ug();
15        c.pg();
16    }
17 }
```

```
Console
<terminated> College
UNDER GRADUATE
post graduate
```

QUESTION 9:

-----  
Project :BikeInformation  
Package :org.bike  
Interface :Bike  
Methods :cost(),speed()

Class :Ktm  
Methods :cost(),speed()

Description:

cost(),speed() is just a template in Bike Interface and You have to override the method cost(),speed() in Ktm class.

The screenshot shows the Eclipse IDE interface with the following components:

- Project Explorer:** Shows a Java project named "AbstractionMethod" containing packages like JRE System Library [jre], org.bike, and src.
- Ktm.java:** A class that implements the Bike interface. It has methods cost() and speed().
- Bike.java:** An interface with methods cost() and speed().
- Console:** Displays the output of the program: <terminated>, 560000, 50kmph.

```

Ktm.java
1 package org.bike;
2
3 public class Ktm implements Bike {
4     @Override
5     public void cost() {
6         System.out.println("560000");
7     }
8
9     @Override
10    public void speed() {
11        System.out.println("50kmph");
12    }
13}
14public static void main(String[] args) {
15    Ktm z = new Ktm();
16    z.cost();
17    z.speed();
18}
19}

Bike.java
1 package org.bike;
2
3 public interface Bike {
4     void cost();
5     void speed();
6
7 }
8

```

#### QUESTION 10:

```

Project      :Computer
Interface    :HardWare
Methods      :hardwareResources()

Interface   :Software
Methods      :softwareResources()

Class       :Desktop
Methods      :desktopModel()

```

#### Description:

create 2 Interface and achieve multiple inheritance.

The screenshot shows the Eclipse IDE interface with the following components:

- Project Explorer:** Shows a Java project named "AbstractionMethod" containing packages like JRE System Library [jre], Computer, src, and others.
- Software.java:** An interface named Software with a method softwareResources().
- Desktop.java:** A class named Desktop that implements both Hardware and Software interfaces. It overrides the softwareResources() method and calls the hardwareResources() method.
- Hardware.java:** An interface named Hardware with a method hardwareResources().
- Console:** Displays the output of the program: hardware resource laptop, windows 10, dell insipirion 3501.

```

Software.java
1
2 public interface Software {
3     void softwareResources();
4 }
5

Desktop.java
1
2 public class Desktop implements Hardware,Software {
3     private void desktopModel1() {
4         System.out.println("dell insipirion 3501");
5     }
6
7     @Override
8     public void softwareResources() {
9         System.out.println("windows 10");
10    }
11 }
12
13
14 @Override
15 public void hardwareResources() {
16     System.out.println("hardware resource laptop");
17 }
18
19 public static void main(String[] args) {
20     Desktop r = new Desktop();
21     r.hardwareResources();
22     r.softwareResources();
23     r.desktopModel1();
24 }
25 }

Hardware.java
1
2 public interface Hardware {
3     void hardwareResources();
4 }
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

```

## DAY4-

Day4\_ControlStatements

DAY4:

-----

1.if/else if  
2.Loopings(for,while,do-while)  
3.Switch case  
4.break/continue  
QUESTIONS(Find the output)

-----  
QUESTION 1:

-----

package org.test;

```
public class Hello {  
    public static void main(String[] args) {  
        for (int i = 1; i <= 100; i++) {  
            if (i == 5) {  
  
            }  
            System.out.println(i);  
  
        }  
    }  
}
```

QUESTION 2:

-----

package org.test;

```
public class Hello {  
    public static void main(String[] args) {  
        for (int i = 1; i <= 10; i++) {  
            if (i == 5) {  
                break;  
            }  
            System.out.println(i);  
  
        }  
    }  
}
```

QUESTION 3:

-----

package org.test;

```
public class Hello {  
    public static void main(String[] args) {  
        for (int i = 1; i <= 10; i++) {  
            if (i == 5) {  
                continue;  
            }  
            System.out.println(i);  
        }  
    }  
}
```

```
    }
}

}

QUESTION 4:
```

```
-----
package org.test;

public class Hello {
    public static void main(String[] args) {
        for (int i = 1; i <= 3; i++) {
            for (int j = 1; j <= 3; j++) {
                System.out.println(j);
            }
        }
    }
}
```

```
QUESTION 5:
```

```
-----
package org.test;

public class Hello {
    public static void main(String[] args) {
        for (int i = 1; i <= 3; i++) {
            for (int j = 1; j <= 3; j++) {
                System.out.println(i);
            }
        }
    }
}
```

```
QUESTION 6:
```

```
-----
package org.test;

public class Hello {
    public static void main(String[] args) {
        for (int i = 1; i <= 3; i++) {
            for (int j = 1; j <= i; j++) {
                System.out.println(j);
            }
        }
    }
}
```

```
QUESTION 7:
```

```
-----
package org.test;
```

```
public class Hello {  
    public static void main(String[] args) {  
        for (int i = 1; i <= 3; i++) {  
            for (int j = i + 1; j <= 3; j++) {  
                System.out.println(j);  
            }  
        }  
    }  
}
```

QUESTION 8:

```
-----  
package org.test;  
  
public class Hello {  
    public static void main(String[] args) {  
        for (int i = 1; i <= 3; i++) {  
            for (int j = i + 1; j <= i; j++) {  
                System.out.println(j);  
            }  
        }  
    }  
}
```

QUESTION 9:

```
-----  
package org.test;  
  
public class Hello {  
    public static void main(String[] args) {  
        int i=5;  
        if (i == 5) {  
            break;  
        }  
        System.out.println(i);  
    }  
}
```

QUESTION 10:

```
-----  
package org.test;  
  
public class Hello {  
    public static void main(String[] args) {  
        int i=5;  
        if (i == 5) {  
            continue;  
        }  
        System.out.println(i);  
    }  
}
```



0.1  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100

The screenshot shows the Eclipse IDE interface with two identical Java files, `Hello.java`, open in separate editors. Both files are located in the package `org.test`. The code in both files is as follows:

```
1 package org.test;
2
3 public class Hello {
4     public static void main(String[] args) {
5         for (int i = 1; i <= 100; i++) {
6             if (i == 5) {
7                 System.out.println(i);
8             }
9         }
10    }
11 }
12
13 }
```

The Project Explorer view on the left shows several Java projects and source files. The Console view at the bottom shows the output of the application, which prints the numbers from 1 to 100, with the number 5 printed in blue.

**Project Explorer**

- AbstractionMethod
- Collegelnformation5
- Constructor
- DayList
- Encapsulation
- FileHandling
- Inheritance
- JavaProjectOne
- JavatPoint
  - JRE System Library [jre]
  - src
    - org.test
      - Hello.java
- LanguageDetails1
- Polymorphism
- RemoteSystemsTempFiles
- SeleniumDailyPractice
- SeleniumJan09

**Console**

```
<terminated> Hello (2) [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.jdt.openjdkhotspot.jre.ful
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
break cannot be used outside of a loop or a switch
at org.test.Hello.main(Hello.java:7)
```

**Hello.java**

```
1 package org.test;
2
3 public class Hello {
4     public static void main(String[] args) {
5         int i=5;
6         if (i == 5) {
7             break;
8         }
9         System.out.println(i);
10    }
11 }
12 }
```

**Project Explorer**

- AbstractionMethod
- Collegelnformation5
- Constructor
- DayList
- Encapsulation
- FileHandling
- Inheritance
- JavaProjectOne
- JavatPoint
  - JRE System Library [jre]
  - src
    - org.test
      - Hello.java
- LanguageDetails1
- Polymorphism
- RemoteSystemsTempFiles
- SeleniumDailyPractice
- SeleniumJan09

**Console**

```
<terminated> Hello (2) [Java Application] C:\Users\navee\p2\pool\pli
1
2
3
```

**Hello.java**

```
1 package org.test;
2
3 public class Hello {
4     public static void main(String[] args) {
5         for (int i = 1; i <= 3; i++) {
6             for (int j = i + 1; j <= i; j++) {
7                 System.out.println(j);
8             }
9         }
10    }
11 }
12 }
```

**Project Explorer**

- AbstractionMethod
- Collegelnformation5
- Constructor
- DayList
- Encapsulation
- FileHandling
- Inheritance
- JavaProjectOne
- JavatPoint
  - JRE System Library [jre]
  - src
    - org.test
      - Hello.java
- LanguageDetails1
- Polymorphism
- RemoteSystemsTempFiles
- SeleniumDailyPractice
- SeleniumJan09

**Console**

```
<terminated> Hello (2) [Java Application] C:\Users\navee\p2\pool\pli
1
2
3
```

**Hello.java**

```
1 package org.test;
2
3 public class Hello {
4     public static void main(String[] args) {
5         for (int i = 1; i <= 3; i++) {
6             for (int j = i + 1; j <= 3; j++) {
7                 System.out.println(j);
8             }
9         }
10    }
11 }
12 }
```

**Project Explorer**

- AbstractionMethod
- Collegelnformation5
- Constructor
- DayList
- Encapsulation
- FileHandling
- Inheritance
- JavaProjectOne
- JavatPoint
  - JRE System Library [jre]
  - src
    - org.test
      - Hello.java
- LanguageDetails1
- Polymorphism
- RemoteSystemsTempFiles
- SeleniumDailyPractice
- SeleniumJan09

**Console**

```
<terminated> Hello (2) [Java Application] C:\Users\navee\p2\pool\pli
1
1
2
1
2
3
```

**Hello.java**

```
1 package org.test;
2
3 public class Hello {
4     public static void main(String[] args) {
5         for (int i = 1; i <= 3; i++) {
6             for (int j = 1; j <= i; j++) {
7                 System.out.println(j);
8             }
9         }
10    }
11 }
12 }
```

The image shows three separate Java code snippets, each with its own editor window, demonstrating different control flow structures.

**Top Editor:**

```
1 package org.test;
2
3 public class Hello {
4     public static void main(String[] args) {
5         for (int i = 1; i <= 3; i++) {
6             for (int j = 1; j <= 3; j++) {
7                 System.out.println(i);
8             }
9         }
10    }
11 }
12
13 }
```

**Middle Editor:**

```
1 package org.test;
2
3 public class Hello {
4     public static void main(String[] args) {
5         for (int i = 1; i <= 3; i++) {
6             for (int j = 1; j <= 3; j++) {
7                 System.out.println(j);
8             }
9         }
10    }
11 }
12
13 }
```

**Bottom Editor:**

```
1 package org.test;
2
3 public class Hello {
4     public static void main(String[] args) {
5         for (int i = 1; i <= 10; i++) {
6             if (i == 5) {
7                 continue;
8             }
9             System.out.println(i);
10        }
11    }
12 }
13
14 }
```

The screenshot shows two instances of the Eclipse IDE interface. In both instances, the Project Explorer view lists several Java projects and files. The Hello.java file is selected in both the Project Explorer and the Editor view.

**Top Instance:**

- Project Explorer:** Shows JavaProjectOne, JRE System Library [jre], and org.test.
- Console:** Displays the output of the Java application: "Hello (2) [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.jst.j.openjdk.hotspot.jre.full". The output shows the numbers 1 through 4.
- Hello.java:** The code is as follows:
 

```

1 package org.test;
2
3 public class Hello {
4     public static void main(String[] args) {
5         for (int i = 1; i <= 10; i++) {
6             if (i == 5) {
7                 break;
8             }
9             System.out.println(i);
10        }
11    }
12 }
13 }
14 }
```

**Bottom Instance:**

- Project Explorer:** Shows JavaProjectOne, JRE System Library [jre], and org.test.
- Console:** Displays the output of the Java application: "Hello (2) [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.jst.j.openjdk.hotspot.jre.full". The output shows the error message: "Exception in thread "main" java.lang.Error: Unresolved compilation problem: continue cannot be used outside of a loop at org.test.Hello.main(Hello.java:8)".
- Hello.java:** The code is as follows:
 

```

1 package org.test;
2
3 public class Hello {
4
5     public static void main(String[] args) {
6         int i=5;
7         if (i == 5) {
8             continue;
9         }
10        System.out.println(i);
11    }
12 }
13 }
```

## QUESTIONS (Programs)

---

### QUESTION 1:

---

Description: Write Java program to allow the user to input his/her age.  
 Then the program will show if the person is eligible to vote.  
 A person who is eligible to vote must be older than or equal 18 years old.

Example:

---

Input = 10

Output = print not eligible.

The screenshot shows the Eclipse IDE interface. On the left is the Project Explorer view, listing various Java projects and files. In the center is the Hello.java editor window, displaying Java code. On the right is the Console view, showing the program's output.

**Project Explorer:**

- AbstractionMethod
- CollegelInformation5
- Constructor
- DayList
- Encapsulation
- FileHandling
- Inheritance
- JavaProjectOne
- JavatPoint
  - JRE System Library [jre]
  - src
    - org.test
      - Hello.java
- LanguageDetails1
- Polymorphism
- RemoteSystemsTempFiles
- SeleniumDailyPractice
- SeleniumJan09

**Hello.java Editor:**

```
1 package org.test;
2 import java.util.*;
3 public class Hello {
4
5     public static void main(String args[])
6     {
7         int age;
8         Scanner sc=new Scanner(System.in);
9         System.out.print("input");
10        age=sc.nextInt();
11        if(age>=18)
12            System.out.println("You are eligible to vote.");
13        else
14            System.out.println("You are not eligible to vote.");
15    }
16
17
18 }
```

**Console Output:**

```
<terminated> Hello (2) [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse.jdt.core\src\org\test\Hello.java
input
10
You are not eligible to vote.
```

QUESTION 2:

-----  
Description: Write a program to find even or odd number

Example:

-----  
Input = 10  
Output = Even

The screenshot shows the Eclipse IDE interface. The Project Explorer view on the left lists various Java projects and files. The Hello.java file in the JavaProjectOne project is open in the editor. The code prints a message to the console, reads a number from the user, and then checks if it is even or odd, printing the result. The Console view at the bottom shows the output of running the program with the input '10', which outputs '10 is even'.

```

1 package org.test;
2 import java.util.Scanner;
3 public class Hello {
4
5     public static void main(String[] args) {
6
7         Scanner reader = new Scanner(System.in);
8
9         System.out.print("Enter a number: ");
10        int num = reader.nextInt();
11
12        if(num % 2 == 0)
13            System.out.println(num + " is even");
14        else
15            System.out.println(num + " is odd");
16    }
17 }

```

### QUESTION 3:

Description: Write a program to print even number from 1 to 100

Example:

Output = 2, 4, ..., 100

The screenshot shows the Eclipse IDE interface. The Project Explorer view on the left lists various Java projects and files. The Hello.java file in the JavaProjectOne project is open in the editor. The code prints a message to the console, initializes a variable 'number' to 100, and then uses a for loop to iterate from 1 to 100, printing even numbers. The Console view at the bottom shows the output of running the program, which lists even numbers from 1 to 100.

```

1 package org.test;
2 import java.util.Scanner;
3 public class Hello {
4
5     public static void main(String args[]) {
6
7         int number=100;
8         System.out.print("List of even numbers from 1 to "+number+": ");
9         for (int i=1; i<=number; i++)
10        {
11             if ((i%2==0)
12             {
13                 System.out.print(i + " ");
14             }
15         }
16     }
17 }

```

### QUESTION 4:

Description: Find the sum of odd number 1 to 100

Example:

Output = 2500

The screenshot shows the Eclipse IDE interface. On the left is the Project Explorer view, which lists various Java projects and files. In the center is the editor view showing a Java file named Hello.java. The code calculates the sum of odd numbers from 1 to 100 and prints the result. On the right is the Console view, which displays the output of the program.

```

1 package org.test;
2 import java.util.Scanner;
3 public class Hello {
4     public static void main(String args[])
5     {
6         int sum = 0;
7         for (int i = 1; i <= 100; i++)
8         {
9             if (i % 2 != 0)
10             {
11                 sum = sum + i;
12             }
13         }
14         System.out.println("The Sum Of 100 Odd Numbers are:" + sum);
15     }
16 }

```

<terminated> Hello (2) [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjd  
The Sum Of 100 Odd Numbers are:2500

#### QUESTION 5:

Description: Count of even number 1 to 100

Example:

Output = 50

The screenshot shows the Eclipse IDE interface. The Project Explorer view is identical to the previous one. In the editor view, the Hello.java file has been modified to calculate the sum of even numbers from 1 to 100. On the right is the Console view, which displays the output of the program.

```

1 package org.test;
2 import java.util.Scanner;
3 public class Hello {
4     public static void main(String args[])
5     {
6         int sum = 0;
7         for (int i = 1; i <= 100; i++)
8         {
9             if (i % 2 == 0)
10             {
11                 sum = sum + i;
12             }
13         }
14         System.out.println("The Sum Of 100 Even Numbers are:" + sum);
15     }
16 }

```

<terminated> Hello (2) [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjd  
The Sum Of 100 Even Numbers are:2550

#### QUESTION 6:

Description: Write a program to find the factorial of a number.

Example:

-----

Input = 5  
Output = 120

The screenshot shows the Eclipse IDE interface. On the left is the Project Explorer view, which lists several Java projects and source files under the 'org.test' package. In the center is the editor view showing the 'Hello.java' file with the following code:

```
1 package org.test;
2 import java.util.Scanner;
3 public class Hello {
4     public static void main(String args[]){
5         int i,fact=1;
6         int number=5;//It is the number to calculate factorial
7         for(i=1;i<=number;i++){
8             fact=fact*i;
9         }
10        System.out.println("Factorial of "+number+" is: "+fact);
11    }
12 }
```

On the right is the Console view, which displays the output of the program: "Factorial of 5 is: 120".

QUESTION 7:

-----

Description: Write a program to print the fibonacci series of a number 1 to 100.

Example:

-----

Output = 0,1,1,2,3,5.....

The screenshot shows the Eclipse IDE interface. On the left is the Project Explorer view, which lists several Java projects and source files under the 'org.test' package. In the center is the editor view showing the 'Hello.java' file with the following code:

```
1 package org.test;
2 import java.util.Scanner;
3 public class Hello {
4     public static void main(String[] args)
5     {
6         int n, a = 0, b = 0, c = 1;
7         System.out.print("Fibonacci Series:");
8         for(int i = 1; i <= 100; i++)
9         {
10             a = b;
11             b = c;
12             c = a + b;
13             System.out.print(a+" ");
14         }
15     }
16 }
```

On the right is the Console view, which displays the output of the program: "Fibonacci Series: 0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765 10946 17711 28657 46368 75025 121393 196418 317811 514229 832040 1346269 2178309 352457".

QUESTION 8:

-----

Description: Find prime number or not.

Example:

```

-----
Input = 11
Output = prime number
-----
```

The screenshot shows the Eclipse IDE interface. On the left is the Project Explorer view, listing various Java projects and files under the JavatPoint project. The Hello.java file is selected and shown in the center editor window. The code implements a prime number checker using a for loop and an if statement. Below the editor is the Console view, which shows the application's output. It prompts for a number, receives '87' as input, and then prints '87 is not a Prime Number'.

```

-----
```

```

1 package org.test;
2 import java.util.Scanner;
3 public class Hello {
4     public static void main(String args[])
5     {
6         int temp;
7         boolean isPrime=true;
8         Scanner scan= new Scanner(System.in);
9         System.out.println("Enter any number:");
10        int num=scan.nextInt();
11        scan.close();
12        for(int i=2;i<=num/2;i++)
13        {
14            temp=num%i;
15            if(temp==0)
16            {
17                isPrime=false;
18                break;
19            }
20        }
21        if(isPrime)
22            System.out.println(num + " is a Prime Number");
23        else
24            System.out.println(num + " is not a Prime Number");
25    }
26 }
```

```

-----
```

```

Console
<terminated> Hello [2] (Java Application) C:\Users\navee\p2\pool\plugins\org.eclipse.justj.c
Enter any number:
87
87 is not a Prime Number
-----
```

#### QUESTION 9:

-----  
Description : Print the below patterns using for loop.

Output:

```

1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
1 2 3 4 5 6 7
-----
```

```

*
* *
* * *
* * * *
* * * * *
```

\*  
\* \*  
\* \* \*  
\* \* \* \*  
\* \* \* \* \*  
-----

The image shows a Java development environment with three code editors and three corresponding consoles.

**Top Editor (Test.java):**

```
1 package org.test;
2
3
4 public class Test {
5     public static void main(String[] args)
6     {
7
8         for (int i = 1; i <= 7; i++)
9         {
10            for (int j = 1; j <= i; j++)
11            {
12                System.out.print(j+ " ");
13            }
14
15            System.out.println();
16        }
17    }
18 }
```

**Top Console Output:**

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
1 2 3 4 5 6 7
```

**Middle Editor (Test.java):**

```
1 package org.test;
2
3
4 public class Test {
5     public static void main(String[] args)
6     {
7
8         int rows = 5;
9
10        for (int i = 1; i <= rows; ++i) {
11            for (int j = 1; j <= i; ++j) {
12                System.out.print("* ");
13            }
14            System.out.println();
15        }
16    }
17 }
```

**Middle Console Output:**

```
*
```

```
* *
```

```
* * *
```

```
* * * *
```

```
* * * * *
```

**Bottom Editor (\*Test.java):**

```
1 package org.test;
2
3
4 public class Test {
5     public static void main(String[] args) {
6         for (int i = 0; i < 5; i++) {
7             for (int j = 0; j < 5 - i; j++)
8             {
9                 System.out.print(" ");
10            }
11            for (int k = 0; k <= i; k++)
12            {
13                System.out.print("* ");
14            }
15            System.out.println();
16        }
17    }
18 }
```

**Bottom Console Output:**

```
*
```

```
* *
```

```
* * *
```

```
* * * *
```

```
* * * * *
```

QUESTION 10:

Description: Find Amstrong number or not

Example:

Input = 153

Output = Amstrong number

The screenshot shows the Eclipse IDE interface. On the left is the Project Explorer view, listing various Java projects and files under the 'JavaProjectOne' project. In the center is the editor view, showing the code for 'Hello.java'. The code is as follows:

```
1 package org.test;
2 import java.util.Scanner;
3 public class Hello {
4     public static void main(String[] args) {
5         int num = 153, number, temp, total = 0;
6         number = num;
7         while (number != 0)
8         {
9             temp = number % 10;
10            total = total + temp*temp*temp;
11            number /= 10;
12        }
13
14        if(total == num)
15            System.out.println(num + " is an Armstrong number");
16        else
17            System.out.println(num + " is not an Armstrong number");
18    }
19 }
20 }
21 }
```

On the right is the Console view, which displays the output of the program: '153 is an Armstrong number'.

QUESTION 11:

Description: Reverse the number

Example:

Input = 123

Output = 321

The screenshot shows the Eclipse IDE interface. On the left, the Project Explorer view displays various Java projects and files. In the center, the Editor view shows the code for `Hello.java`. On the right, the Console view shows the execution results.

**Project Explorer:**

- AbstractionMethod
- Collegelnformation5
- Constructor
- DayList
- Encapsulation
- FileHandling
- Inheritance
- JavaProjectOne
- JavatPoint
  - JRE System Library [jre]
  - src
    - org.test
      - Hello.java
- LanguageDetails1
- Polymorphism
- RemoteSystemsTempFiles
- SeleniumDailyPractice
- SeleniumJan09

**Hello.java:**

```
1 package org.test;
2 import java.util.Scanner;
3 public class Hello {
4     public static void main(String[] args)
5     {
6         int number = 123, reverse = 0;
7         while(number != 0)
8         {
9             int remainder = number % 10;
10            reverse = reverse * 10 + remainder;
11            number = number/10;
12        }
13        System.out.println("The reverse of the given number is: " + reverse);
14    }
15}
```

**Console:**

```
<terminated> Hello (2) [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jri
The reverse of the given number is: 321
```

**QUESTION 12:**

-----  
Description: Count of the number

Example:

-----  
Input = 123  
Output = 3

The screenshot shows the Eclipse IDE interface. On the left is the Project Explorer view, listing various Java projects and files. In the center is the Hello.java editor window, displaying the following Java code:

```
1 package org.test;
2 import java.util.Scanner;
3 public class Hello {
4     public static void main(String[] args) {
5         int count = 0, num = 123;
6
7         while (num != 0) {
8             // num = num/10
9             num /= 10;
10            ++count;
11        }
12
13        System.out.println("Number of digits: " + count);
14    }
15
16 }
```

On the right is the Console view, showing the output of the program's execution:

```
<terminated> Hello (2) [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse.jdt.core\src\Hello.java:12: error: cannot find symbol
System.out.println("Number of digits: " + count);
                                     ^
symbol:   variable out
location: class Hello
1 error
```

The console also displays the output: "Number of digits: 3".

QUESTION 13:

-----

Description: Sum of the number

Example:

-----

Input = 123

Output = 6

The screenshot shows the Eclipse IDE interface. The Project Explorer view on the left lists various Java projects and files. The Hello.java file is open in the editor, displaying the following code:

```
1 package org.test;
2 import java.util.Scanner;
3 public class Hello {
4     public static void main(String args[])
5     {
6         int number, digit, sum = 0;
7         Scanner sc = new Scanner(System.in);
8         System.out.print("Enter the number: ");
9         number = sc.nextInt();
10        while(number > 0)
11        {
12            //finds the last digit of the given number
13            digit = number % 10;
14            //adds last digit to the variable sum
15            sum = sum + digit;
16            //removes the last digit from the number
17            number = number / 10;
18        }
19        //prints the result
20        System.out.println("Sum of Digits: "+sum);
21    }
22 }
```

The Console view at the bottom shows the output of the application:

```
<terminated> Hello (2) [Java Application] C:\Users\navee\p2\pool\plu
Enter the number: 123
Sum of Digits: 6
```

QUESTION 14:

Description: Verify the number is palindrome number not

Example:

-----

Input = 141

Output = Palindrome

The screenshot shows the Eclipse IDE interface with the following components:

- Project Explorer** (left): Lists various Java projects and files, including "AbstractionMethod", "CollegelInformation5", "Constructor", "DayList", "Encapsulation", "FileHandling", "Inheritance", "JavaProjectOne", "JavatPoint" (selected), "LanguageDetails1", "Polymorphism", "RemoteSystemsTempFiles", "SeleniumDailyPractice", and "SeleniumJan09".
- Hello.java** (center): The code for a Java application that checks if a number is a palindrome. It uses a while loop to extract digits from the number and compares them.

```
1 package org.test;
2 import java.util.Scanner;
3 public class Hello {
4     public static void main(String args[]){
5         int r,sum=0,temp;
6         int n=141;//It is the number variable to b
7
8         temp=n;
9         while(n>0){
10             r=n%10; //getting remainder
11             sum=(sum*10)+r;
12             n=n/10;
13         }
14         if(temp==sum)
15             System.out.println("palindrome number ");
16         else
17             System.out.println("not palindrome");
18     }
19 }
```

- Console** (bottom): Shows the output of the application running. The output text is "terminated> Hello (2) [Java Application] C:\Users\navee\p2\pool\plugins\c androme number"

QUESTIONS (Find the below Output)

-----

QUESTION 1:

-----

```
package org.test;

public class A {
    public A() {
        this("JAVA");
        System.out.println("Default const...");
    }

    public A(int id) {
        this(3456.5678f);
        System.out.println(id);
    }

    public A(String name) {
        this(12);
        System.out.println(name);
    }

    public A(float sal) {
        System.out.println(sal);
    }

    public static void main(String[] args) {
        A a = new A();
    }
}
```

The screenshot shows the Eclipse IDE interface. The Package Explorer view on the left lists various Java projects and files. The Java Editor view in the center contains the code for class A. The Console view on the right shows the output of the program's execution.

```
1 package org.test;
2
3 public class A {
4     public A() {
5         this("JAVA");
6         System.out.println("Default const...");
7     }
8
9     public A(int id) {
10        this(3456.5678f);
11        System.out.println(id);
12    }
13
14     public A(String name) {
15        this(12);
16        System.out.println(name);
17    }
18
19     public A(float sal) {
20        System.out.println(sal);
21    }
22
23     public static void main(String[] args) {
24        A a = new A();
25    }
26
27 }
```

<terminated> A [Java  
3456.5679  
12  
JAVA  
Default const...

QUESTION 2:

-----

```
package org.test;
```

```
public class A extends B{
    public A() {
```

```
        System.out.println("Default const...");
```

```

    }

    public static void main(String[] args) {
        A a = new A();
    }

}

package org.test;

public class B {
    public B() {
        System.out.println("Super class");
    }
}

```

The screenshot shows the Eclipse IDE interface with three open windows:

- Package Explorer:** Shows the project structure with several Java and Selenium-related projects.
- A.java:** Contains the code for class A, which extends class B and prints "Default const..." to the console.
- B.java:** Contains the code for class B, which prints "Super class" to the console.
- Console:** Displays the output of the application, showing "Super class" and "Default const...".

### QUESTION 3:

```

-----
package org.test;

public class A extends B{
    public A() {
        System.out.println("Default const...");
    }

    public static void main(String[] args) {
        A a = new A();
    }

}

package org.test;

public class B {
    public B() {
        System.out.println("Super class");
    }
}

```

```

public B(int id) {
    System.out.println(id);
}

}

Package Explorer ☰
> AbstractionMethod
> CollegelInformation5
< Constructor
> JRE System Library [jre]
> src
  > org.test
    > A.java
    > B.java
> CoreJavaPractice
> DayList
> Inheritance
> JavaProjectOne
> LanguageDetails1
> Polymorphism
> SeleniumDailyPractice
> SeleniumJan09

Java ☰
1 package org.test;
2
3
4 public class A extends B{
5     public A() {
6
7         System.out.println("Default const...");
8     }
9
10    public static void main(String[] args) {
11        A a = new A();
12    }
13
14
15 }

Console ☰
<terminated> A [Java Application] C:\Users\navee\
Super class
Default const...

B.java ☰
1 package org.test;
2
3 public class B {
4     public B() {
5         System.out.println("Super class");
6     }
7
8     public B(int id) {
9         System.out.println(id);
10    }
11
12 }


```

#### QUESTION 4:

```

-----
package org.test;

public class A extends B {
    public A() {
        super(12);
        System.out.println("Default const...");
    }

    public static void main(String[] args) {
        A a = new A();
    }
}

package org.test;

public class B {
    public B() {
        System.out.println("Super class");
    }

    public B(int id) {
        System.out.println(id);
    }
}

```

The screenshot shows the Eclipse IDE interface with the following components:

- Package Explorer:** Shows several Java projects: AbstractionMethod, Collegelnformations5, Constructor (selected), CoreJavaPractice, DayList, Inheritance, JavaProjectOne, LanguageDetails1, Polymorphism, SeleniumDailyPractice, and SeleniumJan09.
- A.java:** A code editor containing the following Java code:

```
1 package org.test;
2
3 public class A extends B {
4     public A() {
5         super(12);
6         System.out.println("Default const...");
7     }
8
9     public static void main(String[] args) {
10        A a = new A();
11    }
12
13 }
14 }
```
- B.java:** A code editor containing the following Java code:

```
1 package org.test;
2
3 public class B {
4     public B() {
5         System.out.println("Super class");
6     }
7
8     public B(int id) {
9         System.out.println(id);
10    }
11
12 }
```
- Console:** Displays the output of the application run, showing:

```
<terminated> A [Java Application] C:\Users\12
Default const...
```

#### QUESTION 5:

---

```
package org.test;

public class B {

    public B(int id) {
        System.out.println(id);
    }
}

package org.test;

public class A extends B {
    public A() {
        super(12);
        System.out.println("Default const...");
    }

    public static void main(String[] args) {
        A a = new A();
    }
}
```

The screenshot shows the Eclipse IDE interface. The Package Explorer view on the left lists several Java projects and files. The A.java file is open in the top editor, and the B.java file is open in the bottom editor. The console view on the right shows the output of the A.java program.

```
A.java
1 package org.test;
2
3
4 public class A extends B {
5     public A() {
6         super(12);
7         System.out.println("Default const...");
8     }
9
10    public static void main(String[] args) {
11        A a = new A();
12    }
13
14 }
```

```
B.java
1 package org.test;
2
3 public class B {
4
5     public B(int id) {
6         System.out.println(id);
7     }
8
9 }
```

```
Console
<terminated> A [Java]
12
Default const..
```

#### QUESTION 6:

---

```
package org.test;

public class A extends B {
    public A() {
        System.out.println("Default const...");
    }

    public static void main(String[] args) {
        A a = new A();
    }
}

package org.test;

public class B {

    public B(int id) {
        System.out.println(id);
    }
}
```

The screenshot shows the Eclipse IDE interface with three main panes: Package Explorer, Java Editor, and Console.

**Package Explorer:** Shows various Java projects and files. The `A.java` file is selected.

**Java Editor (A.java):**

```
1 package org.test;
2
3 public class A extends B {
4     public A() {
5         System.out.println("Default const...");
6     }
7
8     public static void main(String[] args) {
9         A a = new A();
10    }
11
12 }
13
14
15
```

**Java Editor (B.java):**

```
1 package org.test;
2
3 public class B {
4
5     public B(int id) {
6         System.out.println(id);
7     }
8
9 }
10
11
```

**Console:** Displays the following error message:

```
<terminated> A [Java Application] C:\Users\naveen\p2\pool\plugins\org.eclipse.jdt.core\open
Implicit super constructor B() is undefined. Must explicitly invoke it
at org.test.A.<init>(A.java:4)
at org.test.A.main(A.java:10)
```

QUESTION 1:

-----  
Description: Find the length of the below string  
String 1: GreensTechnology  
String 2: SeleniumAutomationtool  
String 3: velmurugan  
String 4: j a v a p r o g r a m  
String 5: 9095484678

The screenshot shows an IDE interface with two main panes. On the left is the code editor for a file named 'Hello.java'. The code is as follows:

```
1 package org.test;
2 import java.util.Scanner;
3 public class Hello {
4     public static void main(String args[]){
5         String s="GreensTechnology";
6         String s1="SeleniumAutomationtool";
7         String s2="velmurugan";
8         String s3="velmurugan";
9         String s4="9095484678";
10        int length =s.length();
11        System.out.println(length);
12        int length1 =s1.length();
13        System.out.println(length1);
14
15        int length2 =s.length();
16        System.out.println(length2);
17        int length3 =s3.length();
18        System.out.println(length3);
19
20        int length4 =s4.length();
21        System.out.println(length4);
22
23    }
24
25
26 }
```

On the right is the 'Console' pane, which displays the output of the program's execution. The output is:

```
<terminated>
16
22
16
10
10
```

QUESTION 2:

-----  
Description: Find the particular character index in the given string  
String 1: GreensTechnology  
Find the last index of o  
  
String 2: SeleniumAutomationtool  
Find the index of o  
  
String 3: Velmurugan  
Find the index of n  
  
String 4: j a v a p r o g r a m  
Find the last index of (emptyspace)  
  
String 5: 9095484678  
Find the index of 8

The screenshot shows a Java IDE interface. On the left, the code editor displays `Hello.java` with the following content:

```
1 package org.test;
2 import java.util.Scanner;
3 public class Hello {
4     public static void main(String args[]){
5         String s="GreensTechnology";
6         String s1="SeleniumAutomationtool";
7         String s2="velmurugan";
8         String s3="j a v a p r o g r a m";
9         String s4="9095484678";
10        int length =s.lastIndexOf('o');
11        System.out.println(length);
12
13        int length1 =s1.indexOf('o');
14        System.out.println(length1);
15
16        int length2 =s2.indexOf('n');
17        System.out.println(length2);
18
19        int length3 =s3.lastIndexOf(' ');
20        System.out.println(length3);
21
22        int length4 =s4.indexOf('8');
23        System.out.println(length4);
24
25    }
26
27
28 }
```

On the right, the `Console` tab shows the output of the program:

```
<terminated> 13
11
9
19
5
```

Question 3:

-----

Description: Find the particular character in the given string

String 1: GreensTechnology  
print the character h

String 2: SeleniumAutomationtool  
print the character o

String 3: velmurugan  
print the character u

String 4: j a v a p r o g r a m  
print the character p

String 5: 9095484678  
print the character 7

```
1 package org.test;
2 import java.util.Scanner;
3 public class Hello {
4     public static void main(String args[]){
5         String s="GreensTechnology";
6         String s1="SeleniumAutomationtool";
7         String s2="velmurugan";
8         String s3="j a v a p r o g r a m";
9         String s4="9095484678";
10        char charAt = s.charAt(9);
11        System.out.println(charAt);
12        char charAt2 = s1.charAt(11);
13        System.out.println(charAt2);
14        char charAt3 = s2.charAt(4);
15        System.out.println(charAt3);
16        char charAt4 = s3.charAt(8);
17        System.out.println(charAt4 );
18        char charAt5 = s4.charAt(8);
19        System.out.println(charAt5);
20
21    }
22
23 }
```

QUESTION 4:

-----

Description : Get two input from user and check the equality  
              : print in the output whether it is Equal or not

Example:

-----

Input :

String 1 : Java

String 2 : Java

Example:

-----

Input :

String 1 : Java

String 2 : java

Example:

-----

Input :

String 1 : Green Technology

String 2 : GreenTechnology.

Example(use equalsIgnoreCase) :

-----

Input :

String 1 : Java

String 2 : java

Example(use equalsIgnoreCase) :

-----

Input :

String 1 : Nisha

String 2 : nisha

The screenshot shows a Java IDE interface. On the left, the code editor displays a file named "Hello.java" with the following content:

```
1 package org.test;
2 import java.util.Scanner;
3 public class Hello {
4     public static void main(String args[]){
5         String s="Java";
6         String s1="Java";
7         boolean equals = s.equals(s1);
8         System.out.println(equals);
9         String s2="Java";
10        String s3="java";
11        boolean equals2 = s2.equals(s3);
12        System.out.println(equals2);
13        String s4="Green Technology";
14        String s5="GreenTechnology ";
15        boolean equals3 = s4.equals(s5);
16        System.out.println(equals3);
17        String s6="Java";
18        String s7="java";
19        boolean equalsIgnoreCase = s6.equalsIgnoreCase(s7);
20        System.out.printlnequalsIgnoreCase);
21        String s8="Java";
22        String s9="java";
23        boolean equalsIgnoreCase2 = s8.equalsIgnoreCase(s9);
24        System.out.printlnequalsIgnoreCase2);
25    }
26
27 }
```

On the right, the "Console" tab shows the output of the program:

```
<terminated>
true
false
false
true
true
```

QUESTION 5:

-----

QUESTION 5.1:

-----

Description: Get the email id from the user and verify '@' is present or not?

Example:

-----

Input = velmurugank451@gmail.com

Output = valid email id

QUESTION 5.2:

-----

Description: Get the address from the user and verify "pincode" is present or not?

Example:

-----

Input = 5-35-2a,venkatesh nivas,Aruppukottai

Output = invalid address

QUESTION 5.3:

-----

Description: Get the email from the user and verify '@' is present or not and return true or false?

Example:

-----

Input = Nishakerala24@gmail.com

Output = True/False

QUESTION 5.4:

-----  
Description: Get the phonenumber from the user and verify any character is present or not .

If character is present return invalid number

Example:

-----  
Input = 90954a6o78  
Output = False

The screenshot shows an IDE interface with two panes. The left pane is titled "HelloJava.java" and contains the following Java code:

```
1 package org.test;
2
3 public class HelloJava {
4     public static void main(String[] args) {
5         String s="velmurugank451@gmail.com";
6         boolean contains = s.contains("@");
7         if(contains == true) {
8             System.out.println("valid email id");
9         }
10        String s1="5-35-2a,venkatesh nivas,Aruppukottai";
11        boolean contains2 = s1.contains("pincode");
12        if(contains2 == false) {
13            System.out.println("invalid address");
14        }
15        String s2="Nishakerala24@gmail.com";
16        boolean contains3 = s2.contains("@");
17        System.out.println(contains3);
18        String s4="90954a6o78";
19        boolean matches = s4.matches("%[a-zA-Z]%");
20        System.out.println(matches);
21    }
22 }
23
24 }
```

The right pane is titled "Console" and shows the output of the program:

```
<terminated> HelloJ
valid email id
invalid address
true
false
```

QUESTION 6:

-----  
Description: Get the phonenumber from the user .

If phonenumber exceeds greater than 10 then return invalid number

Example:

-----  
Input = 89034256972365

Output = invalid

Example 2:

-----  
Input = 9095484678

Output = valid

```
J HelloJava.java ✘
```

```
1 package org.test;
2
3 public class HelloJava {
4     public static void main(String[] args) {
5         String s="89034256972365";
6         int length = s.length();
7         if(length>10) {
8             System.out.println("invalid");
9         }
10        String s2="9095484678";
11        int length2 = s2.length();
12        if(length2 == 10) {
13            System.out.println("valid id");
14        }
15    }
16}
17
18 }
```

```
Console ✘
<terminated>
invalid
valid id
```

```
J HelloJava.java ✘
```

```
1 package org.test;
2
3 public class HelloJava {
4     public static void main(String[] args) {
5         String s="89034256972365";
6         int length = s.length();
7         if(length>10) {
8             System.out.println("invalid");
9         }
10        String s2="9095484678";
11        int length2 = s2.length();
12        if(length2 == 10) {
13            System.out.println("valid id");
14        }
15    }
16}
17
18 }
```

```
Console ✘
<terminated>
invalid
valid id
```

QUESTION 7:

-----

QUESTION 7.1:

-----

Description: Given string as "Welcome to java class" and replace java into sql.

Example:

-----

Input = Welcome to class java  
output = Welcome to class sql

QUESTION 7.2:

-----

Description: Given string as "Greens Adayar" and replace Adayar into Omr.

Example:

-----  
Input = Greens Adayar  
Output = Greens Omr

QUESTION 7.3:

-----  
Description: Given String as "Welcome to java class" and Replace space into '#'

Example:

-----  
input:Welcome to java class  
output:Welcome#to#java#class

QUESTION 7.4:

-----  
Description: Get the email from the user and verify "gmail" is present or not.

If present replace that gmail into yahoo

Example:

-----  
Input = Nishakerala24@gmail.com  
Output = Nishakerala24@yahoo.com

QUESTION 7.5:

-----  
Description: Get the address from the user and verify "pincode" is present or not.

If present replace the pincode with empty space

Example:

-----  
Input = 5-35-2a,venkatesh nivas,Aruppukottai,pincode-626101  
Output = 5-35-2a,venkatesh nivas,Aruppukottai

The screenshot shows the Eclipse IDE interface. The left pane displays the Java code for `HelloJava.java`. The right pane shows the output from the `Console` tab.

```

1 package org.test;
2
3 public class HelloJava {
4     public static void main(String[] args) {
5         String s="Welcome to java class";
6         String replace = s.replace("java", "sql");
7         System.out.println(replace);
8
9         String s1="Greens Adayar";
10        String replace2 = s1.replace("Adayar", "Omr");
11        System.out.println(replace2);
12
13        String s2="Welcome to java class";
14        String replace3 = s2.replace(" ", "#");
15        System.out.println(replace3);
16
17        String s3="Nishakerala24@gmail.com";
18        String replace4 = s3.replace("gmail", "yahoo");
19        System.out.println(replace4);
20        String s4="5-35-2a,venkatesh nivas,Aruppukottai,pincode-626101";
21        String replace5 = s4.replace("5-35-2a,venkatesh nivas,Aruppukottai,pincode-626101", "5-35-2a,venkatesh nivas,Aruppukottai");
22        System.out.println(replace5);
23    }
24
25 }
26

```

`<terminated> HelloJava [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\bin\javaw.exe (27-Jan-2022, 4)`

Output:

```

Welcome to sql class
Greens Omr
Welcome#to#java#class
Nishakerala24@yahoo.com
5-35-2a,venkatesh nivas,Aruppukottai

```

QUESTION 8

-----

QUESTION 8.1

-----

Description: Get the input from the user and print that word in lowercase

Example:

-----

Input = NISHANTHI  
Output = nishanthi

QUESTION 8.2

-----

Description: Get the input from the user and print that word in Uppercase

Example:

-----

Input = nishanthi  
Output = NISHANTHI

QUESTION 8.3

-----

Description: Convert all small letter and into capital letter

Example:

-----

Input = WelcomE  
Output = wELCOMe

QUESTION 8.4

-----

Description: Find the number of uppercase count and lowercase count in the given String

Example:

-----

Input = WelComeToJava

Output:

-----

UpperCase=4

LowerCase=9

The screenshot shows the Eclipse IDE interface. The top window is titled "HelloJava.java" and contains the following Java code:

```
1 package org.test;
2
3 public class HelloJava {
4     public static void main(String[] args) {
5         String s="NISHANTHI";
6         String replace = s.toLowerCase();
7         System.out.println(replace);
8
9         String s1="nishanthi";
10        String replace2 = s1.toLowerCase();
11        System.out.println(replace2);
12
13        String s2="WelcomE";
14        String replace3 = s2.substring(0,1).toLowerCase()+s2.substring(1, 6).toUpperCase()+s2.substring(6).toLowerCase();
15        System.out.println(replace3);
16
17        String s3="WelComeToJava";
18        int upper = 0,lower=0;
19        for(int i=0;i<s3.length();i++) {
20            char ch=s3.charAt(i);
21            if(ch>='A' && ch<='Z') {
22                upper++;
23            }
24            else if(ch>='a' && ch<='z') {
25                lower++;
26            }
27        }
28        System.out.println("uppercasecount "+upper);
29        System.out.println("lowercasecount:"+lower);
30    }
31
32 }
```

The bottom window is titled "Console" and shows the terminal output:

```
<terminated> HelloJava [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\bin\javaw.exe (27-)
nishanthi
nishanthi
wELCOME
uppercasecount 4
lowercasecount:9
```

QUESTION 9

-----

QUESTION 9.1

-----

Description: Given String as "Welcome to java class" and verify whether the given string startsWith welcome

Example:

-----

Input = Welcome to class java

output = True

QUESTION 9.2

-----

Description: Given String as "Hai i am nisha" and verify whether the given string startsWith welcome

Example:

```
-----  
Input = Hai i am nisha  
output = False
```

#### QUESTION 9.3

```
-----  
Description: Given String as "Welcome to java class" and verify whether the  
given string endsWith class
```

Example:

```
-----  
Input = Welcome to java class  
output = True
```

#### QUESTION 9.4

```
-----  
Description: Given String as "Welcome to java class" and verify whether the  
given string endsWith java
```

Example:

```
-----  
Input = Welcome to java class  
output = False
```

#### QUESTION 9.5

```
-----  
Description: Given String as "Welcome to java class" and verify whether the  
string is empty or not
```

Example:

```
-----  
Input = Welcome to java class  
output = False
```

#### QUESTION 9.6

```
-----  
Description: Given String as "" and verify whether the string is empty or  
not
```

Example:

```
-----  
Input = ""  
Output = False
```

The screenshot shows a Java IDE interface with two panes. The left pane displays the code for a class named HelloJava. The right pane shows the console output.

```
1 package org.test;
2
3 public class HelloJava {
4     public static void main(String[] args) {
5         String s="Welcome to class java";
6         boolean startsWith = s.startsWith("Welcome ");
7         System.out.println(startsWith);
8
9         String s1=" Hai i am nisha";
10        boolean startsWith2 = s1.startsWith("Welcome");
11        System.out.println(startsWith2);
12
13        String s2="Welcome to class java";
14        boolean endsWith = s2.endsWith("java");
15        System.out.println(endsWith);
16
17        String s3="Welcome to class java";
18        boolean endsWith2 = s3.endsWith("class");
19        System.out.println(endsWith2);
20
21        String s4="Welcome to class java";
22        boolean empty = s4.isEmpty();
23        System.out.println(empty);
24
25        String s5="";
26        boolean empty2 = s5.isEmpty();
27        System.out.println(empty2);
28
29    }
30
31 }
32 }
```

Console Output:

```
<terminated>
true
false
true
false
false
true
```

QUESTION 10

-----

Description : Get two input from user and Compare

Example

-----

```
String 1 : Nisha
String 2 : nisha
```

Example

-----

```
String 1 : Nia
String 2 : nisha
```

The screenshot shows an IDE interface with two panes. The left pane displays the Java code for `HelloJava.java`. The right pane shows the output in the `Console` tab.

```
1 package org.test;
2
3 public class HelloJava {
4     public static void main(String[] args) {
5         String s1 = "Nisha";
6         String s2= "nisha";
7         boolean equals = s1.equals(s2);
8         System.out.println(equals );
9
10        String s3 = "Nia";
11        String s4="nisha";
12    boolean equals2 = s3.equals(s4);
13    System.out.println(equals2 );
14
15    }
16
17 }
```

Console output:

```
<terminated>
false
false
```

QUESTION 11

-----

QUESTION 11.1

-----

Description : Generate the two literal string and find the identityHashCose()

Example

-----

String 1 : Nisha

String 2 : Nisha

QUESTION 11.2

-----

Description : Generate the two non literal string and find the identityHashCose()

Example

-----

String 1 : Nisha

String 2 : Nisha

QUESTION 11.3

-----

Description : Generate the three non literal string and find the identityHashCose()

Example

-----

String 1 : Nisha

String 2 : Rengan

String 3 : NishaRengan

QUESTION 11.4

-----

Description : Generate the three literal string and find the identityHashCose()

Example

-----

String 1 : Nisha

String 2 : Rengan

String 3 : NishaRengan

```
1 package org.test;
2
3 public class HellloJava {
4     public static void main(String[] args) {
5         String s1 = "Nisha";
6         String s2= "nisha";
7         int identityHashCode = System.identityHashCode(s1);
8         System.out.println(identityHashCode);
9         int identityHashCode2 = System.identityHashCode(s2);
10        System.out.println(identityHashCode2);
11
12        String s3 = "Nisha";
13        String s4="Nisha";
14        int identityHashCode3 = System.identityHashCode(s3);
15        System.out.println(identityHashCode3);
16        int identityHashCode4 = System.identityHashCode(s4);
17        System.out.println(identityHashCode4);
18
19
20    }
21
22
23 }
```

```
Console
<terminated> HellloJava [1]
1279149968
2096057945
1279149968
1279149968
```

```
1 package org.test;
2
3 public class HellloJava {
4     public static void main(String[] args) {
5         //String s1 ="Nisha";
6         //String s2= "Rengan";
7         //String s3 = "NishaRengan";
8         System.out.println("non literal string");
9         String s1=new String("Nisha");
10        String s2=new String("Rengan");
11        String s3=new String("NishaRengan");
12        int identityHashCode = System.identityHashCode(s1);
13        System.out.println(identityHashCode);
14        int identityHashCode2 = System.identityHashCode(s2);
15        System.out.println(identityHashCode2);
16        int identityHashCode3 = System.identityHashCode(s3);
17        System.out.println(identityHashCode3);
18
19
20        System.out.println("literal string");
21        String s4 = "Nisha";
22        String s5= "Rengan";
23        String s6 = "NishaRengan";
24        int identityHashCode4 = System.identityHashCode(s4);
25        System.out.println(identityHashCode4);
26        int identityHashCode5 = System.identityHashCode(s5);
27        System.out.println(identityHashCode5);
28        int identityHashCode6 = System.identityHashCode(s6);
29        System.out.println(identityHashCode6);
30    }
31
32
33
34 }
```

```
Console
<terminated> HellloJava [1]
non literal string
434176574
2096057945
1689843956
literal string
766572210
1020391880
977993101
```

QUESTION 12

-----

QUESTION 12.1

-----

Description: Given String as "Welcome to java class" and split it by space.

Example:

-----

Input :Welcome to java class

Output:

-----

Welcome

to

java

class

QUESTION 12.2

-----

Description: Given String as "Welcome to java class" and split it by l

Example:

-----

Input :Welcome to java class

Output:

-----

We

come to java c

ass

The screenshot shows a Java application window with two panes. The left pane is titled 'HelloJava.java' and contains the following Java code:

```
1 package org.test;
2
3 public class HelloJava {
4     public static void main(String[] args) {
5
6         String s1="Welcome to java class";
7         String[] splits1=s1.split(" ");
8         for (String a : splits1) {
9             System.out.println(a);
10        }
11        System.out.println("-----");
12        String[] s2=s1.split("1");
13        for (String b : s2) {
14            System.out.println(b);
15        }
16    }
}
```

The right pane is titled 'Console' and shows the output of the program:

```
<terminated> HelloJava [Java Application] C:\Users\nav
Welcome
to
java
class
-----
Welcome to java class
```

QUESTION 13

-----

QUESTION 13.1

-----

Description: Given String as "Welcome to java class" and generate a substring.

Example:

-----

Input :Welcome to java class

Output:

-----

Welcome

The screenshot shows a Java code editor with a file named "HelloJava.java". The code prints "Welcome to java class" to the console. To the right is a "Console" window showing the output "Welcome".

```
1 package org.test;
2
3 public class HelloJava {
4     public static void main(String[] args) {
5
6         String s1="Welcome to java class";
7         String substring = s1.substring(0, 8);
8         System.out.println(substring );
9     }
10 }
```

QUESTION 14

-----

Example:

-----

Description: Given String as "Welcome" and the number of consonant count and vowels count

Example:

-----

Input = Welcome

Output:

-----

vowels = 3

consonant = 4

The screenshot shows a Java code editor with a file named "HelloJava.java". The code counts vowels and consonants in the string "Welcome". The console output shows "vowels:3" and "consonants:4".

```
1 package org.test;
2
3 public class HelloJava {
4     public static void main(String[] args) {
5
6         String s1="Welcome";
7         int vowels=0,consonants=0;
8         s1 =s1.toLowerCase();
9         for (int i = 0; i < s1.length(); i++) {
10             char ch=s1.charAt(i);
11             if(ch == 'a' || ch == 'e' || ch == 'i'||ch == 'o' || ch == 'u' ) {
12                 vowels++;
13             }
14             else if ((ch >= 'a' && ch <= 'z'))
15             {
16                 ++consonants;
17             }
18         }
19         System.out.println("vowels:" +vowels);
20         System.out.println("consonants:" +consonants);
21
22     }
23 }
24 }
```

QUESTION 15:

-----

Description: Find the count of caps, small, number and special character in given string

Example:

-----

Input : Welcome To Java class @123

Output

-----

caps count :3

small count :15

number count:3

Special char:5

The screenshot shows a Java IDE interface with two tabs: "HelloJava.java" and "Console".

```

1 package org.test;
2
3 public class HelloJava {
4     public static void main(String[] args) {
5
6         String s1="Welcome To Java class @123";
7         int upper=0,lower=0,number=0,specialChar=0,space=0;
8         char ch;
9         for (int i = 0; i < s1.length(); i++) {
10            ch =s1.charAt(i);
11            if(ch>='A' && ch<='Z') {
12                upper++;
13            }
14            else if (ch>='a' && ch<='z') {
15                lower++;
16            }
17            else if(ch>='0' && ch<='9') {
18                number++;
19            }
20            else if(ch== ' ') {
21                space++;
22            }
23            else {
24                specialChar++;
25            }
26        }
27
28        System.out.println("the no of capital letters is:"+upper);
29        System.out.println("the no of small letters is:"+lower);
30        System.out.println("the no of numerical letters is:"+number);
31        System.out.println("the no of specialchar is:"+specialChar);
32    }
33}

```

The "Console" tab displays the following output:

```

<terminated> HelloJava [Java Application]
the no of capital letters is:3
the no of small letters is:15
the no of numerical letters is:3
the no of specialchar is:1

```

## QUESTION 16

Description: Replace all vowels char into '@'

Example:

Input = Welcome  
Output = W@lc@m@

The screenshot shows a Java IDE interface with two tabs: "HelloJava.java" and "Console".

```

1 package org.test;
2
3 public class HelloJava {
4     public static void main(String[] args) {
5         String s="Welcome";
6         System.out.println(s);
7         char ch[]=s.toCharArray();
8         for(int i = 0;i<ch.length;i++) {
9             if(ch[i] == 'a' || ch[i] == 'e' || ch[i] == 'i' || ch[i] == 'o' || ch[i] == 'u') {
10                 ch[i] ='@';
11             }
12         }
13         for (int i = 0; i < ch.length; i++) {
14             System.out.print(ch[i]);
15         }
16     }
17 }
18
19

```

The "Console" tab displays the following output:

```

<terminated> H
Welcome
W@lc@m@

```

DAY7:

-----

1.Arrays

2.Collections Introduction

3.List (ArrayList only)

QUESTIONS(Find the below Output)

-----

QUESTION 1:

-----

Description : Write a Java program to sum values of an array

Input a[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

Output = 55

The screenshot shows the Eclipse IDE interface. The top window is titled "CourseDetails.java" and contains the following Java code:

```
1 package org.tcs.com;
2
3 import java.util.HashMap;
4
5 public class CourseDetails {
6     public static void main(String[] args) {
7         int [] arr = new int [] {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
8         int sum = 0;
9         //Loop through the array to calculate sum of elements
10        for (int i = 0; i < arr.length; i++) {
11            sum = sum + arr[i];
12        }
13        System.out.println("Sum of all the elements of an array: " + sum);
14    }
15 }
```

The bottom window is titled "Console" and shows the output of the program:

```
<terminated> CourseDetails (4) [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.j
Sum of all the elements of an array: 55
```

QUESTION 2:

-----

Description : Write a Java program to calculate the average value of array elements.

Input a[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

Output = average = 7.0

```
CourseDetails.java
1 package org.tcs.com;
2
3 public class CourseDetails {
4     public static void main(String[] args) {
5         int[] numbers = new int[]{1,2,3,4,5,6,7,8,9,10};
6
7         double sum = 0;
8
9
10        for(int i=0; i<numbers.length ; i++)
11            sum = sum + numbers[i];
12
13
14        double average = sum / numbers.length;
15
16        System.out.println("Average value of array elements is : " + average);
17    }
18}
```

```
Console
<terminated> CourseDetails (4) [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotsp
Average value of array elements is : 5.5
```

QUESTION 3:

-----  
Description : Write a Java program to remove duplicates from array  
Input a[] = {10,10,20,50,60,80,60,50}  
Output a[] = {10,20,50,60,80}

The screenshot shows a Java IDE interface. On the left is the code editor with a file named 'Test.java'. The code is a Java program that creates an ArrayList 'a' containing integers 10, 20, 50, 60, 80, 60, and 50. It then iterates through the list, comparing each element with every subsequent element. If a match is found, the element at index i is removed from the list, and the indices are reset. The output is printed to the console.

```
1 package org.test;
2
3 import java.util.ArrayList;
4
5 public class Test {//Input a[] = {10,10,20,50,60,80,60,50}
6     public static void main(String[] args) {
7         ArrayList a = new ArrayList();
8         a.add(10);
9         a.add(10);
10        a.add(20);
11        a.add(50);
12        a.add(60);
13        a.add(80);
14        a.add(60);
15        a.add(50);
16        int length=a.size();
17        for(int i=0;i<length;i++) {
18            for(int j=i+1;j<length;j++) {
19                if(a.get(i) == a.get(j)) {
20                    a.remove(i);
21                    i=0;
22                    j=0;
23                    length=length-1;
24                }
25            }
26        }
27        System.out.println(a);
28    }
29 }
30 }
```

<terminated> Test (6) [Java A]  
[10, 20, 80, 60, 50]

QUESTION 4:

-----

QUESTION 4.1:

-----

Description : Create a new ArrayListlist with values and find the length of it

Input : List = 10,20,30,90,10,10,40,50

QUESTION 4.2:

-----

Description : Create a new LinkedListlist with values and find the length of it

Input : List = 100,200,300,400,500,600,700

QUESTION 4.3:

-----

Description : Create a new vector with values and find the length of it

Input : List = 105,205,305,405,505,605,705,805

QUESTION 4.4:

-----

Description : Create a new LinkedListlist with values and find the size() of it.

Input : List = 100,200,300,400,500,600,700

Test.java

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.Vector;
5
6 public class Test { //105,205,305,405,505,605,705,805
7     public static void main(String[] args) {
8         Vector <Integer> a =new Vector<Integer>();
9         a.add(105);
10        a.add(205);
11        a.add(305);
12        a.add(405);
13        a.add(505);
14        a.add(605);
15        a.add(705);|
16        a.add(805);
17        int length=a.size();
18        System.out.println(length);
19    }
20}
21
```

Console

```
<terminated> 1
8
```

Test.java

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.LinkedList;
5 import java.util.List;
6 import java.util.Vector;
7
8 public class Test { //105,205,305,405,505,605,705,805
9     public static void main(String[] args) {
10         List <Integer> a =new LinkedList();
11         a.add(100);
12         a.add(200);
13         a.add(300);
14         a.add(400);
15         a.add(500);
16         a.add(600);
17         a.add(700);|
18         int length=a.size();
19         System.out.println(length);
20    }
21}
```

Console

```
<terminated> Test
7
```

The screenshot shows a Java development environment with two code editors and two consoles.

**Code Editors:**

- Test.java**:

```
1 package org.test;
2
3 import java.util.ArrayList;
4
5 public class Test {
6     public static void main(String[] args) {
7         ArrayList a = new ArrayList();
8         a.add(10);
9         a.add(20);
10        a.add(30);
11        a.add(90);
12        a.add(10);
13        a.add(10);
14        a.add(40);
15        a.add(50);
16        int length=a.size();
17        System.out.println(length);
18    }
19 }
```
- List1.java**:

```
1 package org.com;
2
3 import java.util.ArrayList;
4 import java.util.List;
5 import java.util.LinkedList;
6
7
8 public class List1 {
9     public static void main(String[] args) {
10
11
12     List<Integer>l = new LinkedList();
13     l.add(100);
14     l.add(200);
15     l.add(300);
16     l.add(400);
17     l.add(500);
18     l.add(600);
19     l.add(700);
20     int size = l.size();
21     System.out.println(size);
22
23 }
24 }
```

**Consoles:**

- Console 1 (Top Right)**:

```
<terminated> Test
8
```
- Console 2 (Bottom Right)**:

```
<terminated> List1
7
```

QUESTION 5:

-----

QUESTION 5.1:

-----

Description : Get the first index value of 10  
Input: List = 10,20,30,90

QUESTION 5.2:

-----

Description : Get the last index value of 10  
Input: List = 10,20,30,90,10,10,40,50

QUESTION 5.3:

-----  
Description : Get the index value of 50  
Input: List = 10,20,30,90,10,10,40,50

QUESTION 5.4:

-----  
Description : Get the index value of 90  
Input: List = 10,20,30,90,10,10,40,50

QUESTION 5.5:

-----  
Description : Get the each index value of 10 present in below list  
Input: List = 10,20,30,90,10,10,40,50,10

QUESTION 5.6:

-----  
Description : Get the each index value of 70 present in below list  
Input: List = 10,20,30,90,10,10,40,50,10

CourseDetails.java

```
1 package org.tcs.com;
2 import java.util.ArrayList;
3 import java.util.List;
4 import java.util.LinkedList;
5 public class CourseDetails {
6     public static void main(String[] args) {
7         List li = new ArrayList();           //10,20,30,90,10,10,40,
8
9         li.add(10);
10        li.add(20);
11        li.add(30);
12        li.add(90);
13        li.add(10);
14        li.add(10);
15        li.add(40);
16        li.add(50);
17        int indexOf = li.indexOf(50);
18        System.out.println(indexOf);
19    }
20 }
```

Console

```
terminated> CourseDetails (4) [Java Application] C:\Users\navee\.p2\pool\plugins\org.ecli
7
```

### CourseDetails.java

```
1 package org.tcs.com;
2 import java.util.ArrayList;
3 import java.util.List;
4 import java.util.LinkedList;
5 public class CourseDetails {
6     public static void main(String[] args) {
7         List li = new ArrayList();           //10,20,30,90,10,10,40,50
8
9         li.add(10);
10        li.add(20);
11        li.add(30);
12        li.add(90);
13        li.add(10);
14        li.add(10);
15        li.add(40);
16        li.add(50);
17        int indexOf = li.indexOf(10);
18        System.out.println(indexOf);
19    }
20 }
```

### Console

```
<terminated> CourseDetails (4) [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.ju
0
```

```
CourseDetails.java ✘
1 package org.tcs.com;
2 import java.util.ArrayList;
3 import java.util.List;
4 import java.util.LinkedList;
5 public class CourseDetails {
6     public static void main(String[] args) {
7         List li = new ArrayList();
8         li.add(10);
9         li.add(20);
10        li.add(30);
11        li.add(90);
12        int indexOf = li.indexOf(10);
13        System.out.println(indexOf);
14    }
15 }
```

Console ✘

```
<terminated> CourseDetails (4) [Java Application] C:\Users\naven\p
0 <terminated> CourseDetails (4) [Java Application] C:\User:
```

### CourseDetails.java

```
1 package org.tcs.com;
2 import java.util.ArrayList;
3 import java.util.List;
4 import java.util.LinkedList;
5 public class CourseDetails {
6     public static void main(String[] args) {
7         List li = new ArrayList();           //10,20,30,90,10,10,40,50
8
9         li.add(10);
10        li.add(20);
11        li.add(30);
12        li.add(90);
13        li.add(10);
14        li.add(10);
15        li.add(40);
16        li.add(50);
17        int indexOf = li.indexOf(90);
18        System.out.println(indexOf);
19    }
20 }
```

### Console

```
<terminated> CourseDetails (4) [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.j
3
```

QUESTION 6:

-----

QUESTION 6.1:

-----

Description : Get the value present at 2nd index

Input: List = 10,20,30,40,50,60

QUESTION 6.2:

-----

Description : Get the value present at 4th index

Input: List = 100,200,300,400,500,600,700

QUESTION 6.3:

-----

Description : Get the value present at 8th index

Input: List = 105,205,305,405,505,605,705,805

QUESTION 6.4:

-----

Description : Get the each value of list by using normal for loop

Input: List = 105,205,305,405,505,605,705,805

QUESTION 6.5:

-----  
Description : Get the each value of list by using enhanced for loop  
Input: List = 105,205,305,405,505,605,705,805

```
CourseDetails.java ✘
```

```
1 package org.tcs.com;
2 import java.util.ArrayList;
3 import java.util.List;
4 import java.util.LinkedList;
5 public class CourseDetails {
6     public static void main(String[] args) {
7         List li = new ArrayList(); //105,
8
9
10    li.add(105);
11    li.add(205);
12    li.add(305);
13    li.add(405);
14    li.add(505);
15    li.add(605);
16    li.add(705);
17    li.add(805);
18
19    for (int i = 0; i < li.size(); i++) {
20        System.out.println(li.get(i));
21    }
22}
23}
24}
```

```
Console ✘
```

```
<terminated> CourseDetails (4) [Java Application] C:\Users\navee\p
105
205
305
405
505
605
705
805
```

```
CourseDetails.java ✘
```

```
1 package org.tcs.com;
2 import java.util.ArrayList;
3 import java.util.List;
4 import java.util.LinkedList;
5 public class CourseDetails {
6     public static void main(String[] args) {
7         List li = new ArrayList(); //105,
8
9
10    li.add(105);
11    li.add(205);
12    li.add(305);
13    li.add(405);
14    li.add(505);|
15    li.add(605);
16    li.add(705);
17    li.add(805);
18    Object object = li.get(7);
19    System.out.println(object);
20 }
21 }
22
```

I

```
Console ✘
```

```
<terminated> CourseDetails (4) [Java Application] C:\Users\navee\p
805
```

CourseDetails.java

```
1 package org.tcs.com;
2 import java.util.ArrayList;
3 import java.util.List;
4 import java.util.LinkedList;
5 public class CourseDetails {
6     public static void main(String[] args) {
7         List li = new ArrayList(); //100,200,300,400,500
8
9
10    li.add(100);
11    li.add(200);
12    li.add(300);
13    li.add(400);
14    li.add(500);
15    li.add(600);
16    li.add(700);
17
18    Object object = li.get(4);
19    System.out.println(object);
20 }
21 }
22 }
```

Console

```
<terminated> CourseDetails (4) [Java Application] C:\Users\navee\.p2\pool\plugins\org.e
500
```

The screenshot shows a Java development environment with two open windows. The top window is titled 'CourseDetails.java' and contains the following Java code:

```
1 package org.tcs.com;
2 import java.util.ArrayList;
3 import java.util.List;
4 import java.util.LinkedList;
5 public class CourseDetails {
6     public static void main(String[] args) {
7         List li = new ArrayList(); //10,20,30,90,:
8
9         li.add(10);
10        li.add(20);
11        li.add(30);
12        li.add(40);
13        li.add(50);
14        li.add(60);
15
16        Object object = li.get(2);
17        System.out.println(object);
18    }
19 }
20
```

The bottom window is titled 'Console' and displays the output of the program:

```
<terminated> CourseDetails (4) [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse.jdt.core_3.11.0.v20150511-1913\src\org\tcs\com\CourseDetails.java:10
30
```

```
CourseDetails.java ✘
```

```
1 package org.tcs.com;
2 import java.util.ArrayList;
3 import java.util.List;
4 import java.util.LinkedList;
5 public class CourseDetails {
6     public static void main(String[] args) {
7         List li = new ArrayList(); //105,205,305
8
9
10        li.add(105);
11        li.add(205);
12        li.add(305);
13        li.add(405);
14        li.add(505);
15        li.add(605);
16        li.add(705);
17        li.add(805);
18
19        for (Object x : li) {
20            System.out.println(x);
21        }
22    }
23 }
```

```
Console ✘
```

```
<terminated> CourseDetails (4) [Java Application] C:\Users\navee\.p2\pool\plu
105
205
305
405
505
605
705
805
```

QUESTION 7:

-----

QUESTION 7.1:

-----

Description : Remove the value present at 2nd index  
Input: List = 10,20,30,40,50,60

QUESTION 7.2:

-----

Description : Remove the value present at 10th index  
Input: List = 10,20,30,90,10,10,40

**QUESTION 7.3:**

Description : Remove the last value of 10 present in the list  
Input: List = 10,20,30,90,10,10,40

```
1 package org.tcs.com;
2 import java.util.ArrayList;
3 import java.util.List;
4 import java.util.LinkedList;
5 public class CourseDetails {
6    public static void main(String[] args) {
7        List li = new ArrayList();          //10,20,30,90,10,10,40
8
9        li.add(10);
10       li.add(20);
11       li.add(30);
12       li.add(40);
13       li.add(50);
14       li.add(60);
15       li.remove(2);
16       System.out.println(li);
17
18
19    }
20 }
```

```
Console ✘
<terminated> CourseDetails (4) [Java Application] C:\Users\navee\p2\po
[10, 20, 40, 50, 60]
```

**QUESTION 8:**

-----

**QUESTION 8.1:**

-----

Description : Add a value 50 in the 2nd index and display the list after adding.

Input : List = 10,20,30,90,10,10,40,50

**QUESTION 8.2:**

-----

Description : Add a value 70 at the end of the list

Input : List = 10,20,30,90,10,10,40,50

**QUESTION 8.3:**

-----

Description : Add a value 80 at the 8th index of list

Input : List = 10,20,30,90,10,10,40,50

QUESTION 8.4:

-----  
Description : Add a value 100 at the last index of 10 in the list  
Input : List = 10,20,30,90,10,10,40,50

```
CourseDetails.java ✘
1 package org.tcs.com;
2 import java.util.ArrayList;
3 import java.util.List;
4 import java.util.LinkedList;
5 public class CourseDetails {
6     public static void main(String[] args) {
7         List li = new ArrayList(); // 1
8
9
10        li.add(10);
11        li.add(20);
12        li.add(30);
13        li.add(90);
14        li.add(10);
15        li.add(10);
16        li.add(40);
17        li.add(50);
18        li.add(8,70); // 2
19        System.out.println(li);
20    }
21 }
```

```
Console ✘
<terminated> CourseDetails (4) [Java Application] C:\Users\navee\
[10, 20, 30, 90, 10, 10, 40, 50, 70]
```

```
CourseDetails.java ✘
1 package org.tcs.com;
2 import java.util.ArrayList;
3 import java.util.List;
4 import java.util.LinkedList;
5 public class CourseDetails {
6     public static void main(String[] args) {
7         List li = new ArrayList(); // 10,2
8
9
10    li.add(10);
11    li.add(20);
12    li.add(30);
13    li.add(90);
14    li.add(10);
15    li.add(10);
16    li.add(40);
17    li.add(50);
18    li.add(2,50);
19    System.out.println(li);
20 }
21 }
```

```
Console ✘
<terminated> CourseDetails (4) [Java Application] C:\Users\navee\p2\
[10, 20, 50, 30, 90, 10, 10, 40, 50]
```

The screenshot shows an IDE interface with two tabs: 'CourseDetails.java' and 'Console'. The 'CourseDetails.java' tab displays the following Java code:

```
1 package org.tcs.com;
2 import java.util.ArrayList;
3 import java.util.List;
4 import java.util.LinkedList;
5 public class CourseDetails {
6     public static void main(String[] args) {
7         List li = new ArrayList(); // 10,20,30,
8
9
10        li.add(10);
11        li.add(20);
12        li.add(30);
13        li.add(90);
14        li.add(10);
15        li.add(10);
16        li.add(40);
17        li.add(50);
18        li.add(8,80);|
19        System.out.println(li);
20    }
21 }
```

The 'Console' tab shows the output of the program:

```
<terminated> CourseDetails (4) [Java Application] C:\Users\navee\p2\pool\
[10, 20, 30, 90, 10, 10, 40, 50, 80]
```

QUESTION 9:

-----

QUESTION 9.1:

-----

Description : Replace the value 300 into 350 in the list  
Input : List = 100,200,300,400,500,600,700

QUESTION 9.2:

-----

Description : Replace the value present in 7th index as 90  
Input: List = 10,20,30,90,10,10,40,50,10

QUESTION 9.3:

-----

Description : Replace the 10 into 100 in List  
Input: List = 10,20,30,90,10,10,40,50,30  
Output: List = 100,20,30,90,100,100,40,50,30

The screenshot shows a Java IDE interface with two windows. The top window displays the code for `Test.java`. The code creates an `ArrayList` named `a` and adds elements at indices 7 through 16. It then removes the element at index 7 and adds a new element at index 7 with the value 90. Finally, it prints the list. The bottom window shows the output in the console: `[10, 20, 30, 90, 10, 10, 40, 90, 10]`.

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.LinkedList;
5 import java.util.List;
6 import java.util.Vector;
7
8 public class Test { //10,20,30,90,10,10,40,50,10
9     public static void main(String[] args) {
10         List <Integer> a =new ArrayList();
11         a.add(10);
12         a.add(20);
13         a.add(30);
14         a.add(90);
15         a.add(10);
16         a.add(10); //10,20,30,90,10,10,40,50,10
17         a.add(40);
18         a.add(50);
19         a.add(10);
20
21         a.remove(7);
22         a.add(7, 90);
23         System.out.println(a);
24     }
25 }
```

The screenshot shows a Java IDE interface with two windows. The top window displays the code for `Test.java`. The code creates an `ArrayList` named `a` and adds elements from 100 to 700. It then removes the element at index 2 and sets the element at index 2 to 350. Finally, it prints the list. The bottom window shows the output in the console: `[100, 200, 350, 500, 600, 700]`.

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.LinkedList;
5 import java.util.List;
6 import java.util.Vector;
7
8 public class Test {
9     public static void main(String[] args) {
10         List <Integer> a =new ArrayList();
11         a.add(100);
12         a.add(200);
13         a.add(300);
14         a.add(400);
15         a.add(500);
16         a.add(600);
17         a.add(700);
18         //int length=a.size();
19         //System.out.println(length);
20         a.remove(2);
21         a.set(2, 350);
22         System.out.println(a);
23     }
24 }
```

The screenshot shows an IDE interface with two windows. On the left is the code editor window titled "Test.java" containing the following Java code:

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.Collections;
5 import java.util.LinkedList;
6 import java.util.List;
7 import java.util.Vector;
8
9 public class Test {
10    public static void main(String[] args) {
11        List <Integer> a =new ArrayList();
12        a.add(10);
13        a.add(20);
14        a.add(30);
15        a.add(90);
16        a.add(10);
17        a.add(10);
18        a.add(40);
19        a.add(50);
20        a.add(30);
21
22        //int length=a.size();
23        //System.out.println(length);
24
25        boolean replaceAll = Collections.replaceAll(a, 10, 100);
26        //System.out.println(replaceAll);
27        System.out.println(a);
28    }
29}
```

On the right is the "Console" window showing the output of the program:

```
<terminated> Test (6) [Java Application] C:\Users\naveen
[100, 20, 30, 90, 100, 100, 40, 50, 30]
```

QUESTION 10:

-----

QUESTION 10.1:

-----

Description : Create a new ArrayList lists with values and return the common values

Input : List = 10,20,30,90,10,10,40,50  
Input : List = 30,40,50,60,80

QUESTION 10.2:

-----

Description : Create a new ArrayList lists with values and return the common values

Input : List = 10,20,30,90,10,10,40,50  
Input : List = 10,20,60,50,40,70,80,90

QUESTION 10.3:

-----

Description : Create a new ArrayList lists with values and return the common values

Input : List = 10,20,30,40,50,60,70,80  
Input : List = 100,200,300,400,500,600,700,8000

List1.java

```
1 package org.com;
2
3import java.util.ArrayList;
4import java.util.List;
5import java.util.LinkedList;
6
7
8 public class List1 {
9@public static void main(String[] args) {
10    List li =new ArrayList();
11    li.add(10);
12    li.add(20);
13    li.add(30);
14    li.add(90);
15    li.add(10);
16    li.add(10);
17    li.add(40);
18    li.add(80);
19
20    List<Integer>l = new LinkedList();
21    l.add(10);
22    l.add(20);
23    l.add(60);
24    l.add(50);
25    l.add(40);
26    l.add(70);
27    l.add(80);
28    l.add(90);
29    li.addAll(l);
30    System.out.println(li);
31
32 }
33 }
```

Console

```
<terminated> List1 [Java Application]
[10, 20, 90, 10, 10, 40, 80]
```

\*List1.java

```
1 package org.com;
2
3import java.util.ArrayList;
4import java.util.List;
5import java.util.LinkedList;
6
7
8 public class List1 {
9@public static void main(String[] args) {
10    List li =new ArrayList();
11    li.add(10);
12    li.add(20);
13    li.add(30);
14    li.add(90);
15    li.add(10);
16    li.add(10);
17    li.add(40);
18    li.add(50);
19
20    //System.out.println(li);
21    //System.out.println("-----");
22    List<Integer>l = new LinkedList();
23    l.add(30);
24    l.add(40);
25    l.add(50);
26    l.add(60);
27    l.add(80);
28    li.addAll(l);
29    System.out.println(li);
30
31 }
32 }
```

Console

```
<terminated> List
[30, 40, 50]
```

The screenshot shows two Java code editors and their respective consoles.

**Top Editor:**

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.Collections;
5 import java.util.LinkedList;
6 import java.util.List;
7 import java.util.Vector;
8
9 public class Test {
10     public static void main(String[] args) {
11         List <Integer> a =new ArrayList();
12         a.add(10);
13         a.add(20);
14         a.add(30);
15         a.add(90);
16         a.add(10);
17         a.add(10);
18         a.add(40);
19         a.add(50);
20         a.add(30);
21
22         //int length=a.size();
23         //System.out.println(length);
24
25         boolean replaceAll = Collections.replaceAll(a, 10, 100);
26         //System.out.println(replaceAll);
27         System.out.println(a);
28     }
29 }
```

**Top Console:**

```
<terminated> Test (6) [Java Application] C:\Users\naveen
[100, 20, 30, 90, 100, 100, 40, 50, 30]
```

**Bottom Editor:**

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.LinkedList;
5 import java.util.List;
6 import java.util.Vector;
7
8 public class Test { //10,20,30,90,10,10,40,50,10
9     public static void main(String[] args) {
10         List <Integer> a =new ArrayList();
11         a.add(10);
12         a.add(20);
13         a.add(30);
14         a.add(90);
15         a.add(10);
16         a.add(10);|
17         a.add(40);
18         a.add(50);
19         a.add(10);
20
21         a.remove(7);
22         a.add(7, 90);
23         System.out.println(a);
24     }
25 }
```

**Bottom Console:**

```
<terminated> Test (6) [Java Application] C:\Users\naveen
[10, 20, 30, 90, 10, 10, 40, 90, 10]
```

The screenshot shows an IDE interface with two code editors and a central console window.

**Test.java**

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.LinkedList;
5 import java.util.List;
6 import java.util.Vector;
7
8 public class Test {
9     public static void main(String[] args) {
10         List <Integer> a =new ArrayList();
11         a.add(100);
12         a.add(200);
13         a.add(300);
14         a.add(400);
15         a.add(500);
16         a.add(600);
17         a.add(700);
18         //int length=a.size();
19         //System.out.println(length);
20         a.remove(2);
21         a.set(2, 350);
22         System.out.println(a);
23     }
24 }
```

**List1.java**

```
1 package org.com;
2
3 import java.util.ArrayList;
4 import java.util.List;
5 import java.util.LinkedList;
6
7
8 public class List1 {
9     public static void main(String[] args) {
10         List li =new ArrayList();
11         li.add(10);
12         li.add(20);
13         li.add(30);
14         li.add(90);
15         li.add(10);           //| 100,200,300,400,500,600,700,8000
16
17         li.add(10);
18         li.add(40);
19         li.add(80);
20
21         List<Integer>l = new LinkedList();
22         l.add(100);
23         l.add(200);
24         l.add(300);
25         l.add(400);
26         l.add(500);
27         l.add(600);
28         l.add(700);
29         l.add(800);
30         li.retainAll(l);
31         System.out.println(li);
32
33     }
34 }
```

**Console**

```
<terminated> Test (6) [Java Application] C:\[100, 200, 350, 500, 600, 700]
```

The console output shows the result of running the `Test` class, which prints a list containing `[100, 200, 350, 500, 600, 700]`. The `List1` class is currently being run, and its console output is partially visible, showing the addition of elements to a list and the creation of another list `l`.

QUESTION 11:

-----

QUESTION 11.1:

-----

Description : Create a new ArrayListlists with values and perform  
removeAll() function

Input : List = 10,20,30,90,10,10,40,50

Input : List = 30,40,50,60,80

QUESTION 11.2:

-----

Description : Create a new ArrayListlists with values and perform  
removeAll() function

Input : List = 10,20,30,90,10,10,40,50

Input : List = 10,20,60,50,40,70,80,90

QUESTION 11.3:

-----

Description : Create a new ArrayListlists with values and perform  
removeAll() function

Input : List = 10,20,30,40,50,60,70,80

Input : List = 100,200,300,400,500,600,700,8000

The screenshot shows a Java application window with two panes. The left pane is a code editor containing the following Java code:

```
1 package org.com;
2
3 import java.util.ArrayList;
4 import java.util.List;
5 import java.util.LinkedList;
6
7
8 public class List1 {
9     public static void main(String[] args) {
10         List li = new ArrayList();
11         li.add(10);
12         li.add(20);
13         li.add(30);
14         li.add(90);
15         li.add(10);
16         li.add(10);
17         li.add(40);
18         li.add(50);
19
20         //System.out.println(li);
21         //System.out.println("-----");
22         List<Integer>l = new LinkedList();
23         l.add(30);
24         l.add(40);
25         l.add(50);
26         l.add(60);
27         l.add(8);
28         li.removeAll(l);
29         System.out.println(li);
30
31     }
32 }
33
```

The right pane is a console window titled "Console" with the output:

```
<terminated> List1 [Java Application]
[10, 20, 90, 10, 10]
```

The screenshot shows a Java application window with two panes. The left pane is a code editor containing the following Java code:

```
1 package org.com;
2
3 import java.util.ArrayList;
4 import java.util.List;
5 import java.util.LinkedList;
6
7
8 public class List1 {
9     public static void main(String[] args) {
10         List li = new ArrayList();
11         li.add(10);
12         li.add(20);
13         li.add(30);
14         li.add(90);
15         li.add(10); // 100,200,300,400,500,600,700,8000
16
17         li.add(10);
18         li.add(40);
19         li.add(80);
20
21         List<Integer>l = new LinkedList();
22         l.add(100);
23         l.add(200);
24         l.add(300);
25         l.add(400);
26         l.add(500);
27         l.add(600);
28         l.add(700);
29         l.add(800);
30         li.removeAll(l);
31         System.out.println(li);
32
33     }
34 }
```

The right pane is a console window titled "Console" with the output:

```
<terminated> List1 [Java Application] C:\User
[10, 20, 30, 90, 10, 10, 40, 80]
```

```
CourseDetails.java ✘
1 package org.tcs.com;
2 import java.util.ArrayList;
3 import java.util.List;
4 import java.util.LinkedList;
5 public class CourseDetails {
6     public static void main(String[] args) {
7         List li = new ArrayList();           // 10,20,30,90,10,10,40,50
8         li.add(10);
9         li.add(20);
10        li.add(30);
11        li.add(90);
12        li.add(10);
13        li.add(10);
14        li.add(40);
15        li.add(50);
16        List <Integer> l = new LinkedList();
17        l.add(30);
18        l.add(40);
19        l.add(50);
20        l.add(60);|
21        l.add(80);
22        li.removeAll(l);
23        System.out.println(li);
24    }
25 }
```

```
Console ✘
<terminated> CourseDetails (4) [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse.ju
[10, 20, 90, 10, 10]
```

DAY8:

-----

- 1.Userdefine List
- 2.LinkedList vs Vector
- 3.Set and types

QUESTIONS:

-----

QUESTION 1.1:

-----

Description : Create a HashSet for the below values

Input : List = 10,20,30,40,50,60,70,80,90,10,20

QUESTION 1.2:

-----

Description : Create a LinkedHashSet for the below values

Input : List = 10,20,30,40,50,60,70,80,90,10,20

QUESTION 1.3:

-----

Description : Create a TreeSet for the below values

Input : List = 10,20,30,40,50,60,70,80,90,10,20

The image shows a Java development environment with two code editors and two consoles.

**Code Editor 1 (Top Left):**

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.Collections;
5 import java.util.HashSet;
6 import java.util.LinkedHashSet;
7 import java.util.LinkedList;
8 import java.util.List;
9 import java.util.Set;
10 import java.util.TreeSet;
11 import java.util.Vector;
12
13 public class Test {
14     public static void main(String[] args) {
15         Set <Integer> a =new LinkedHashSet();
16         a.add(10);
17         a.add(20);
18         a.add(30); // Cursor is here
19         a.add(40);
20         a.add(50);
21         a.add(60);
22         a.add(70);
23         a.add(80);
24         a.add(90);
25         a.add(10);
26         a.add(20);
27
28         System.out.println(a);
29
30
31     }
32 }
33 }
```

**Console 1 (Top Right):**

```
<terminated> Test (6) [Java Application] C:\Users\nav
[10, 20, 30, 40, 50, 60, 70, 80, 90]
```

**Code Editor 2 (Bottom Left):**

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.Collections;
5 import java.util.HashSet;
6 import java.util.LinkedList;
7 import java.util.List;
8 import java.util.Set;
9 import java.util.TreeSet;
10 import java.util.Vector;
11
12 public class Test {
13     public static void main(String[] args) {
14         Set <Integer> a =new HashSet();
15         a.add(10);
16         a.add(20);
17         a.add(30);
18         a.add(40);
19         a.add(50);
20         a.add(60);
21         a.add(70);
22         a.add(80);
23         a.add(90);
24         a.add(10);
25         a.add(20);
26
27
28         System.out.println(a);
29
30
31     }
32 }
```

**Console 2 (Bottom Right):**

```
<terminated> Test (6) [Java Application] C:\Users\nav
[80, 50, 20, 70, 40, 10, 90, 60, 30]
```

The screenshot shows a Java IDE interface. On the left is the code editor window titled "Test.java" containing the following Java code:

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.Collections;
5 import java.util.HashSet;
6 import java.util.LinkedList;
7 import java.util.List;
8 import java.util.Set;
9 import java.util.TreeSet;
10 import java.util.Vector;
11
12 public class Test {
13     public static void main(String[] args) {
14         Set <Integer> a =new TreeSet();
15         a.add(10);
16         a.add(20);
17         a.add(30);
18         a.add(40);
19         a.add(50);
20         a.add(60);
21         a.add(70);
22         a.add(80);
23         a.add(90);
24         a.add(10);
25         a.add(20);
26
27         System.out.println(a);
28     }
29 }
30
31 }
```

On the right is the "Console" window showing the output of the program:

```
<terminated> Test (6) [Java Application] C:\Users\ [10, 20, 30, 40, 50, 60, 70, 80, 90]
```

QUESTION 2:

-----

QUESTION 2.1:

-----

Description : Convert the below list in to set(use addAll()) method  
Input : List = 10,20,30,90,10,10,40,50

QUESTION 2.2:

-----

Description : Convert the below list in to set(use addAll()) method  
Input : List = 105,205,305,405,505,605,705,805,505,605

QUESTION 2.3:

-----

Description : Convert the below list in to set(use addAll()) method  
Input : List = 100,200,300,400,500,600,700,100,300,500

The image shows a Java development environment with two code editors and two consoles.

**Top Editor:**

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.Collections;
5 import java.util.HashSet;
6 import java.util.LinkedList;
7 import java.util.List;
8 import java.util.Set;
9 import java.util.TreeSet;
10 import java.util.Vector;
11
12 public class Test {
13     public static void main(String[] args) {
14
15         List<Integer> a =new ArrayList();
16         a.add(105);
17         a.add(205);
18         a.add(305);
19         a.add(405);
20         a.add(505);
21         a.add(605);
22         a.add(705);
23         a.add(805);
24         a.add(505);
25         a.add(605);
26
27         Set <Integer> z =new TreeSet();
28         z.addAll(a);
29         System.out.println(z);
30     }
31 }
32 }
```

**Top Console:**

```
<terminated> Test (6) [Java Application] C:\Users\naveen
[105, 205, 305, 405, 505, 605, 705, 805]
```

**Bottom Editor:**

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.Collections;
5 import java.util.HashSet;
6 import java.util.LinkedList;
7 import java.util.List;
8 import java.util.Set;
9 import java.util.TreeSet;
10 import java.util.Vector;
11
12 public class Test {
13     public static void main(String[] args) {
14
15         List<Integer> a =new ArrayList();
16         a.add(10);
17         a.add(20);
18         a.add(30);
19         a.add(90);
20         a.add(10);
21         a.add(10);
22         a.add(40);
23         a.add(50);
24
25         Set <Integer> z =new TreeSet();
26         z.addAll(a);
27         System.out.println(z);
28     }
29 }
30 }
```

**Bottom Console:**

```
<terminated> Test (6) [Java Application]
[10, 20, 30, 40, 50, 90]
```

The screenshot shows an IDE interface with two windows. The left window, titled 'Test.java', contains the following Java code:

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.Collections;
5 import java.util.HashSet;
6 import java.util.LinkedList;
7 import java.util.List;
8 import java.util.Set;
9 import java.util.TreeSet;
10 import java.util.Vector;
11
12 public class Test {
13     public static void main(String[] args) {
14
15         List<Integer> a = new ArrayList();
16         a.add(100);
17         a.add(200);
18         a.add(300);
19         a.add(400);
20         a.add(500);
21         a.add(600);
22         a.add(700);
23         a.add(100);
24         a.add(300);
25         a.add(500);
26
27         Set <Integer> z = new TreeSet();
28         z.addAll(a);
29         System.out.println(z);
30     }
31 }
32
```

The right window, titled 'Console', shows the output of the program:

```
<terminated> Test (6) [Java Application] C:\Users\naveen
[100, 200, 300, 400, 500, 600, 700]
```

QUESTION 3:

-----

QUESTION 3.1:

-----

Description : Get the each value of set by using enhanced for loop

Input: List = 105,205,305,405,505,605,705,805

QUESTION 3.2:

-----

Description : Create a TreeSet and iterate each value in the set by using enhanced for loop

Input : List = 10,20,30,40,50,60,70,80,90,10,20

QUESTION 3.3:

-----

Description : Create a HashSet and iterate each value in the set by using enhanced for loop

Input : List = 10,20,30,40,50,60,70,80,90,10,20

The screenshot shows a Java development environment with a code editor and a terminal window.

**Code Editor (Test.java):**

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.Collections;
5 import java.util.HashSet;
6 import java.util.LinkedHashSet;
7 import java.util.LinkedList;
8 import java.util.List;
9 import java.util.Set;
10 import java.util.TreeSet;
11 import java.util.Vector;
12
13 public class Test {
14     public static void main(String[] args) {
15         Set<Integer> a = new LinkedHashSet();
16         a.add(105);
17         a.add(205);
18         a.add(305);
19         a.add(405);
20         a.add(505);
21         a.add(605);
22         a.add(705);
23         a.add(805);
24         a.add(505);
25         a.add(605);
26         for (Integer i: a) {
27             System.out.println(i);
28         }
29     }
30 }
31
32 ``
```

**Console:**

```
<terminated> Test (t)
105
205
305
405
505
605
705
805
```

The screenshot shows a Java IDE interface. On the left is the code editor window titled "Test.java" containing the following Java code:

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.Collections;
5 import java.util.HashSet;
6 import java.util.LinkedHashSet;
7 import java.util.LinkedList;
8 import java.util.List;
9 import java.util.Set;
10 import java.util.TreeSet;
11 import java.util.Vector;
12
13 public class Test {
14     public static void main(String[] args) {
15         Set<Integer> a = new HashSet();
16         a.add(10);
17         a.add(20);
18         a.add(30);
19         a.add(40);
20         a.add(50);
21         a.add(60);
22         a.add(70);
23         a.add(80);
24         a.add(90);
25         a.add(10);
26         a.add(20);
27
28
29         for (Integer i: a) {
30             System.out.println(i);
31         }
32     }
33 }
34
35 }
36
37 }
```

On the right is the "Console" window titled "<terminated> Test" which displays the output of the program:

```
80
50
20
70
40
10
90
60
30
```

QUESTION 4:

-----

QUESTION 4.1:

-----

Description : Create a new HashSet with values and return the common values

Input : List = 10,20,30,90,10,10,40,50

Input : List = 30,40,50,60,80

QUESTION 4.2:

-----

Description : Create a new LinkedHashSet with values and return the common values

Input : List = 10,20,30,90,10,10,40,50

Input : List = 10,20,60,50,40,70,80,90

QUESTION 4.3:

-----

Description : Create a new TreeSet with values and return the common values

Input : List = 10,20,30,40,50,60,70,80

Input : List = 100,200,300,400,500,600,700,8000

```
Test.java ✘ | Console ✘
6 import java.util.LinkedHashSet;
7 import java.util.LinkedList;
8 import java.util.List;
9 import java.util.Set;
10 import java.util.TreeSet;
11 import java.util.Vector;
12
13 public class Test {
14     public static void main(String[] args) {
15         Set<Integer> a =new LinkedHashSet();
16         a.add(10);
17         a.add(20);
18         a.add(30);
19         a.add(90);
20         a.add(10);
21         a.add(10);
22         a.add(40);
23         a.add(50);
24
25 Set<Integer>a1 =new LinkedHashSet();
26 a1.add(10);
27 a1.add(20);
28 a1.add(60);
29 a1.add(50);
30 a1.add(40);|
31 a1.add(70);
32 a1.add(80);
33 a1.add(90);
34 a.retainAll(a1);
35 System.out.println(a);
36 }
37
38 }
```

The screenshot shows a Java development environment with two tabs: "Test.java" and "Console".

The code in "Test.java" is as follows:

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.Collections;
5 import java.util.HashSet;
6 import java.util.LinkedHashSet;
7 import java.util.LinkedList;
8 import java.util.List;
9 import java.util.Set;
10 import java.util.TreeSet;
11 import java.util.Vector;
12
13 public class Test {
14     public static void main(String[] args) {
15
16         Set<Integer> a =new HashSet();
17         a.add(10);
18         a.add(20);
19         a.add(30);
20         a.add(90);
21         a.add(10);
22         a.add(10);
23         a.add(40);
24         a.add(50);
25
26     Set<Integer>a1 =new HashSet();
27     a1.add(30);
28     a1.add(40);
29     a1.add(50);
30     a1.add(60);
31     a1.add(80);
32
33     a.retainAll(a1);
34     System.out.println(a);
35 }
36
37 }
```

The "Console" tab shows the output of the program:

```
<terminated> Test [50, 40, 30]
```

The screenshot shows a Java IDE interface. On the left is the code editor with a file named 'Test.java'. The code creates a TreeSet 'a' containing integers from 10 to 80, then creates another TreeSet 'a1' containing integers from 100 to 800, retains all elements of 'a1' in 'a', and prints the resulting set. On the right is a terminal window titled 'Console' showing the output: '<terminated: []>'.

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.Collections;
5 import java.util.HashSet;
6 import java.util.LinkedHashSet;
7 import java.util.LinkedList;
8 import java.util.List;
9 import java.util.Set;
10 import java.util.TreeSet;
11 import java.util.Vector;
12
13 public class Test {
14     public static void main(String[] args) {
15         Set<Integer> a =new TreeSet();
16         a.add(10);
17         a.add(20);
18         a.add(30);
19         a.add(40);
20         a.add(50);
21         a.add(60);
22         a.add(70);
23         a.add(80);
24
25     Set<Integer>a1 =new TreeSet();
26     a1.add(100);
27     a1.add(200);
28     a1.add(300);
29     a1.add(400);
30     a1.add(500);
31     a1.add(600);
32     a1.add(700);
33     a1.add(800);
34     a.retainAll(a1);
35     System.out.println(a);
36 }
37
38 }
```

QUESTION 5:

-----

QUESTION 5.1:

-----

Description : Create a new HashSet with values and perform removeAll() function

Input : List = 10,20,30,90,10,10,40,50  
Input : List = 30,40,50,60,80

QUESTION 5.2:

-----

Description : Create a new LinkedHashSet with values and perform removeAll() function

Input : List = 10,20,30,90,10,10,40,50  
Input : List = 10,20,60,50,40,70,80,90

QUESTION 5.3:

-----

Description : Create a new TreeSet with values and perform removeAll() function

Input : List = 10,20,30,40,50,60,70,80

```
Input : List = 100,200,300,400,500,600,700,8000
```

The screenshot shows an IDE interface with two main panes. The left pane displays the code for a Java class named 'Test'. The right pane shows the output of the program's execution.

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.Collections;
5 import java.util.HashSet;
6 import java.util.LinkedHashSet;
7 import java.util.LinkedList;
8 import java.util.List;
9 import java.util.Set;
10 import java.util.TreeSet;
11 import java.util.Vector;
12
13 public class Test {
14     public static void main(String[] args) {
15         Set<Integer> a =new LinkedHashSet();
16         a.add(10);
17         a.add(20);
18         a.add(30);
19         a.add(90);
20         a.add(10);
21         a.add(10);
22         a.add(40);
23         a.add(50);
24
25     Set<Integer>a1 =new LinkedHashSet();
26     a1.add(10);
27     a1.add(20);
28     a1.add(60);
29     a1.add(50);
30     a1.add(40);
31     a1.add(70);
32     a1.add(80);
33     a1.add(90);
34     a.removeAll(a1);
35     System.out.println(a);
36 }
37
38 }
```

The 'Console' tab on the right shows the output of the program:

```
<terminated> T
[30]
```

The screenshot shows a Java application window with two tabs: "Test.java" and "Console".

**Test.java:**

```
1 package org.test;
2
3 import java.util.ArrayList;
4 import java.util.Collections;
5 import java.util.HashSet;
6 import java.util.LinkedHashSet;
7 import java.util.LinkedList;
8 import java.util.List;
9 import java.util.Set;
10 import java.util.TreeSet;
11 import java.util.Vector;
12
13 public class Test {
14     public static void main(String[] args) {
15         Set<Integer> a = new HashSet();
16         a.add(10);
17         a.add(20);
18         a.add(30);
19         a.add(90);
20         a.add(10);
21         a.add(10);
22         a.add(40);
23         a.add(50);
24
25     Set<Integer> a1 = new HashSet();
26     a1.add(30);
27     a1.add(40);
28     a1.add(50);
29     a1.add(60);
30     a1.add(80);
31     a.removeAll(a1);
32     System.out.println(a);
33 }
34
35 }
36
37
```

**Console:**

```
<terminated> Test (6)
[20, 10, 90]
```

The screenshot shows a Java IDE interface. On the left, there is a code editor window titled "test (b)" containing the following Java code:

```
4 import java.util.Collections;
5 import java.util.HashSet;
6 import java.util.LinkedHashSet;
7 import java.util.LinkedList;
8 import java.util.List;
9 import java.util.Set;
10 import java.util.TreeSet;
11 import java.util.Vector;
12
13 public class Test {
14     public static void main(String[] args) {
15         Set<Integer> a =new TreeSet();
16         a.add(10);
17         a.add(20);
18         a.add(30);
19         a.add(40);
20         a.add(50);
21         a.add(60);
22         a.add(70);
23         a.add(80);
24
25         Set<Integer>a1 =new TreeSet();
26         a1.add(100);
27         a1.add(200);
28         a1.add(300);
29         a1.add(400);
30         a1.add(500);
31         a1.add(600);
32         a1.add(700);
33         a1.add(800);
34         a.removeAll(a1);
35         System.out.println(a);
36     }
37
38 }
39
```

On the right, there is a "Console" window showing the output of the program:

```
<terminated> Test (6) [Java Application] C:\Us
[10, 20, 30, 40, 50, 60, 70, 80]
```

**QUESTION 6:**

Description: Create a userdefine Set and insert the 10 employee details  
Input : empId,name,phone,address,dob,doj,eMail,gender,Sal  
Output: Using scanner class insert 10 employee details

**QUESTION 7:**

Description: Create a userdefine Set and insert the 10 Student details  
Key : stdId,stdName,stdPhone,address,dOB,eMail,gender  
Give the related values for key for each Student

DAY9:

- 
- 1.Map and types
  - 2.User define Map

QUESTIONS(practical)

-----

QUESTION 1.1:

Description : Create a HashMap with the below key and values

key : 10,20,30,40,50,60,10,50,40

values : java,sql,oops,Sql,oracle,DB,selenium,pgsql,Hadoop.

-----

QUESTION 1.2:

Description : Create a LinkedHashMap with the below key and values

key : 10,20,30,40,50,60,10,50,40

Values : 10,20,30,40,50,60,10,50,40

-----

QUESTION 1.3:

Description : Create a TreeMap with the below key and values

key : !,@,#,\$,%,&,\*,(,

Values : 10,20,30,40,50,60,10,50,40

-----

QUESTION 1.4:

Description : Create a HashTable with the below key and values

Key : vel,Ganesh,Dinesh,Vengat,subash

Values : Selenium,framework,oracle,corejava,jira

The screenshot shows the Eclipse IDE interface. The top bar has tabs for 'Test3.java' and 'Test2.java'. The main editor window displays the following Java code:

```
1 package org.test;
2
3 import java.util.Map;
4 import java.util.TreeMap;
5
6 public class Test3 {
7     public static void main(String[] args) {
8
9
10    Map<String, Integer> a = new TreeMap();
11    a.put("!", 10);
12    a.put("@", 20);
13    a.put("#", 30);
14    a.put("$", 40);
15    a.put("%", 50);
16    a.put("^", 60);
17    a.put("&", 10);
18    a.put("*", 50);|
19    a.put("(", 40);
20    System.out.println(a);
21 }
22 }//key      : !,@,#,$,%,&,*,(
23 //          Values : 10,20,30,40,50,60,10,50,40
24
```

The cursor is at line 18, character 11, where the character '\*' is being typed. The console window below shows the output of the program:

```
<terminated> Test3 [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hots
{!=10, #=30, $=40, %=50, &=10, (=40, *=50, @=20, ^=60}
```

```
Test2.java
1 package org.test;
2
3 import java.util.LinkedHashMap;
4 import java.util.Map;
5
6 public class Test2 {
7     public static void main(String[] args) {
8         Map<Integer, Integer> q = new LinkedHashMap();
9         q.put(10, 20);
10        q.put(20, 20);
11        q.put(30, 30);
12        q.put(40, 40);
13        q.put(50, 50);
14        q.put(60, 60);
15        q.put(10, 50);
16        q.put(50, 40);
17        q.put(40, 40);
18        System.out.println(q);
19    }
20}
21
22}
```

```
Console
<terminated> Test2 [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.jdt.core\src\org\test\Test2.java:14: error: incompatible types: Object cannot be converted to Map<Integer, Integer>
        System.out.println(q);
                           ^
1 error
```

Test.java

```
1 package org.test;
2
3 import java.util.ArrayList;
4
5 public class Test {
6     public static void main(String[] args) {
7
8         Map<Integer, String> q = new HashMap();
9         q.put(10, "java");
10        q.put(20, "sql");
11        q.put(30, "oops");
12        q.put(40, "sql");
13        q.put(50, "oracle");
14        q.put(60, "DB");
15        q.put(10, "selenium");
16        q.put(50, "pgsql");
17        q.put(40, "Hadoop");
18        System.out.println(q);
19    }
20}
21
22
23
24
25
26
27
28
29
30
31
```

Console

```
<terminated> Test (6) [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.v
{50=pgsql, 20=sql, 40=Hadoop, 10=selenium, 60=DB, 30=oops}
```

Test4.java

```
1 package org.test;
2
3 import java.util.Hashtable;
4 import java.util.Map;
5
6 public class Test4 {
7     public static void main(String[] args) {
8         Map<String, String> u= new Hashtable();
9         u.put("vel", "Selenium");
10        u.put("Ganesh", "framework");
11        u.put("Dinesh", "oracle");
12        u.put("Venkat", "corejava");
13        u.put("subash", "jira");
14        System.out.println(u);
15
16    }
17
18
19}
```

Console

```
<terminated> Test4 [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jr
{Ganesh=framework, Venkat=corejava, vel=Selenium, Dinesh=oracle, subash=jira}
```

QUESTION 2:

-----  
QUESTION 2.1:  
-----

Description : Create a HashMap with the below key and values and get(print) the key only in the map.

key : 10,20,30,40,50,60,10,50,40

values : java,sql,oops,Sql,oracle,DB,selenium,pgsql,Hadoop.

QUESTION 2.2:  
-----

Description : Create a LinkedHashMap with the below key and values and get(print) the key only in the map.

key : 10,20,30,40,50,60,10,50,40

Values : 10,20,30,40,50,60,10,50,40

QUESTION 2.3:  
-----

Description : Create a TreeMap with the below key and values and get(print) the key only in the map.

key : !,@,#,\$,%,&,\*,(),

Values : 10,20,30,40,50,60,10,50,409

QUESTION 2.4:  
-----

Description : Create a HashTable with the below key and values and get(print) the key only in the map.

Key : vel,Ganesh,Dinesh,Vengat,subash

Values : Selenium,framework,oracle,corejava,jira

The screenshot shows the Eclipse IDE interface. The top bar has tabs for 'Test2.java' and '\*Test3.java'. The code editor window displays 'Test3.java' with the following content:

```
1 package org.test;
2 import java.util.Map;
3 import java.util.Set;
4 import java.util.TreeMap;
5 import java.util.Map.Entry;
6 public class Test3 {
7     public static void main(String[] args) {
8         Map<String, Integer> a = new TreeMap();
9         a.put("!", 10);
10        a.put("@", 20);
11        a.put("#", 30);
12        a.put("$", 40);
13        a.put("%", 50);
14        a.put("^", 60);
15        a.put("&", 10);
16        a.put("*", 50);
17        a.put("(", 40);
18        Set<Entry<String, Integer>> entrySet = a.entrySet();
19        for (Entry<String, Integer> entry : entrySet) {
20            System.out.println(entry.getKey());
21        }
22    }
23 }
```

The 'Console' tab at the bottom shows the output of the program:

```
<terminated> Test2 [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.ope
10
20
30
40
50
60
```

```
Test2.java ✘
1 package org.test;
2
3 import java.util.LinkedHashMap;
4
5 public class Test2 {
6     public static void main(String[] args) {
7         Map<Integer, Integer> q = new LinkedHashMap();
8         q.put(10, 20);
9         q.put(20, 20);
10        q.put(30, 30);
11        q.put(40, 40);
12        q.put(50, 50);
13        q.put(60, 60);
14        q.put(10, 50);
15        q.put(50, 40);
16        q.put(40, 40);
17        Set<Entry<Integer, Integer>> entrySet = q.entrySet();
18        for (Entry<Integer, Integer> entry : entrySet) {
19            System.out.println(entry.getKey());
20        }
21    }
22 }
23
24
25
26
27 }
28 //key      10 20 30 40 50 60 10 50 40
```

```
Console ✘
<terminated> Test2 [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse.justj.openjdk.hc
10
20
30
40
50
60
```

```
Test.java ✘
1 package org.test;
2
3 import java.util.ArrayList;
4
5 public class Test {
6     public static void main(String[] args) {
7
8         Map<Integer, String> q = new HashMap();
9         q.put(10, "java");
10        q.put(20, "sql");
11        q.put(30, "oops");
12        q.put(40, "sql");
13        q.put(50, "oracle");
14        q.put(60, "DB");
15        q.put(10, "selenium");
16        q.put(50, "pgsql");
17        q.put(40, "Hadoop");
18        Set<Entry<Integer, String>> entrySet = q.entrySet();
19        for (Entry<Integer, String> entry : entrySet) {
20            System.out.println(entry.getKey());
21        }
22    }
23 }
24
25
26
27
28
29
30
31
32
33
34
35
36
```

```
Console ✘
<terminated> Test (6) [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse.jus
50
20
40
10
60
30
```

```
Test4.java ✘
1 package org.test;
2
3+ import java.util.Hashtable;
4
5
6 public class Test4 {
7     public static void main(String[] args) {
8         Map<String, String> u= new Hashtable();
9         u.put("vel", "Selenium");
10        u.put("Ganesh", "framework");
11        u.put("Dinesh", "oracle");
12        u.put("Venkat", "corejava");
13        u.put("subash", "jira");
14        Set<Entry<String, String>> entrySet = u.entrySet();
15        for (Entry<String, String> entry : entrySet) {
16            System.out.println(entry.getKey());
17        }
18    }
19
20}
21
22
23}
24
```

```
Console ✘
<terminated> Test4 [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hc
Ganesh
Venkat
vel
Dinesh
subash
```

QUESTION 3:

-----  
QUESTION 3.1:

-----  
Description : Create a HashMap with the below key and values and get(print) the values only in the map.

key : 10,20,30,40,50,60,10,50,40  
values : java,sql,oops,Sql,oracle,DB,selenium,pgsql,Hadoop.

-----  
QUESTION 3.2:

-----  
Description : Create a LinkedHashMap with the below key and values and get(print) the values only in the map.

key : 10,20,30,40,50,60,10,50,40  
Values : 10,20,30,40,50,60,10,50,40

-----  
QUESTION 3.3:

-----  
Description : Create a TreeMap with the below key and values and get(print) the values only in the map.

key : !,@,#,\$,%,&,\*,(,

Values : 10,20,30,40,50,60,10,50,409

QUESTION 3.4:

-----  
Description : Create a HashTable with the below key and values and  
get(print) the key only in the map.

Key : vel,Ganesh,Dinesh,Vengat,subash

Values : Selenium,framework,oracle,corejava,jira

```
Test3.java ✘
1 package org.test;
2 import java.util.Map;
3
4 public class Test3 {
5     public static void main(String[] args) {
6         Map<String, Integer> a = new TreeMap();
7         a.put("!", 10);
8         a.put("@", 20);
9         a.put("#", 30);
10        a.put("$", 40);
11        a.put("%", 50);
12        a.put("^", 60);
13        a.put("&", 10);
14        a.put("*", 50);
15        a.put("(", 40);
16        Set<Entry<String, Integer>> entrySet = a.entrySet();
17        for (Entry<String, Integer> entry : entrySet) {
18            System.out.println(entry.getValue());
19        }
20    }
21 }
22 }
23 }
```

```
Console ✘
<terminated> Test3 [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse.justj.openjdk.hots
10
30
40
50
10
40
50
20
60
```

The screenshot shows the Eclipse IDE interface with two open views: 'Test2.java' and 'Console'.

**Test2.java** content:

```
1 package org.test;
2
3 import java.util.LinkedHashMap;
4
5 public class Test2 {
6     public static void main(String[] args) {
7         Map<Integer, Integer> q = new LinkedHashMap();
8         q.put(10, 20);
9         q.put(20, 20);
10        q.put(30, 30);
11        q.put(40, 40);
12        q.put(50, 50);
13        q.put(60, 60);
14        q.put(10, 50);
15        q.put(50, 40);
16        q.put(40, 40);
17        Set<Entry<Integer, Integer>> entrySet = q.entrySet();
18        for (Entry<Integer, Integer> entry : entrySet) {
19            System.out.println(entry.getValue());
20        }
21    }
22 }
23 }
```

**Console** output:

```
<terminated> Test2 [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspo
50
20
30
40
40
60
```

```
Test.java ✘
1 package org.test;
2
3 import java.util.ArrayList;
4
5 public class Test {
6     public static void main(String[] args) {
7
8         Map<Integer, String> q = new HashMap();
9         q.put(10, "java");
10        q.put(20, "sql");
11        q.put(30, "oops");
12        q.put(40, "sql");
13        q.put(50, "oracle");
14        q.put(60, "DB");
15        q.put(10, "selenium");
16        q.put(50, "psql");
17        q.put(40, "Hadoop");
18        Set<Entry<Integer, String>> entrySet = q.entrySet();
19        for (Entry<Integer, String> entry : entrySet) {
20            System.out.println(entry.getValue());
21        }
22    }
23 }
24 }
```

```
Console ✘
<terminated> Test (6) [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse.justj.o
psql
sql
Hadoop
selenium
DB
oops
```

```
Test4.java ✘
1 package org.test;
2
3④ import java.util.Hashtable;□
4
5
6 public class Test4 {
7     public static void main(String[] args) {
8         Map<String, String> u= new Hashtable();
9         u.put("vel", "Selenium");
10        u.put("Ganesh", "framework");
11        u.put("Dinesh", "oracle");
12        u.put("Venkat", "corejava");
13        u.put("subash", "jira");
14        Set<Entry<String, String>> entrySet = u.entrySet();
15        for (Entry<String, String> entry : entrySet) {
16            System.out.println(entry.getValue());
17        }
18    }
19
20}
21
22
23 }
24
```

```
Console ✘
<terminated> Test4 [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse.justj.open
framework
corejava
Selenium
oracle
jira
```

QUESTION 4:

QUESTION 4.1:

Description : Create a HashMap with the below key and values and iterate it using enhanced for loop.

key	: 10,20,30,40,50,60,10,50,40
values	: java,sql,oops,Sql,oracle,DB,selenium,pgsql,Hadoop.

QUESTION 4.2:

Description : Create a LinkedHashMap with the below key and values and iterate it using enhanced for loop.

key	: 10,20,30,40,50,60,10,50,40
Values	: 10,20,30,40,50,60,10,50,40

QUESTION 4.3:

Description : Create a TreeMap with the below key and values and iterate it using enhanced for loop.

key	: !,@,#,\$,%,&,*,(,
-----	---------------------

Values : 10,20,30,40,50,60,10,50,409

QUESTION 4.4:

-----  
Description : Create a HashTable with the below key and values and iterate it using enhanced for loop.

Key : vel,Ganesh,Dinesh,Vengat,subash

Values : Selenium,framework,oracle,corejava,jira

```
Test3.java ✘
1 package org.test;
2 import java.util.Map;
3 public class Test3 {
4     public static void main(String[] args) {
5         Map<String, Integer> a = new TreeMap();
6         a.put("!", 10);
7         a.put("@", 20);
8         a.put("#", 30);
9         a.put("$", 40);
10        a.put("%", 50);
11        a.put("^", 60);
12        a.put("&", 10);
13        a.put("*", 50);
14        a.put("(", 40);
15        Set<Entry<String, Integer>> entrySet = a.entrySet();
16        for (Entry<String, Integer> entry : entrySet) {
17            System.out.println(entry);
18        }
19    }
20 }
21 }
```

```
Console ✘
<terminated> Test3 [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse
!=10
#=30
$=40
%=50
&=10
(=40
*=50
@=20
^=60
```

The screenshot shows the Eclipse IDE interface with two open windows:

- Test2.java**: The code is a Java application named `Test2`. It imports `java.util.LinkedHashMap` and defines a class `Test2` with a `main` method. The `main` method creates a `LinkedHashMap` and populates it with integer pairs. It then iterates over the entry set and prints each value.
- Console**: The output window shows the terminal session for the application. The output consists of six lines of text, each showing a key-value pair from the map: `10=50`, `20=20`, `30=30`, `40=40`, `50=40`, and `60=60`.

```
1 package org.test;
2
3 import java.util.LinkedHashMap;
4
5
6 public class Test2 {
7     public static void main(String[] args) {
8         Map<Integer, Integer> q = new LinkedHashMap();
9         q.put(10, 20);
10        q.put(20, 20);
11        q.put(30, 30);
12        q.put(40, 40);
13        q.put(50, 50);
14        q.put(60, 60);
15        q.put(10, 50);
16        q.put(50, 40);
17        q.put(40, 40);
18        Set<Entry<Integer, Integer>> entrySet = q.entrySet();
19        for (Entry<Integer, Integer> entry : entrySet) {
20            System.out.println(entry.getValue());
21        }
22    }
23
24
25 }
26
```

```
<terminated> Test2 [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64\bin>
10=50
20=20
30=30
40=40
50=40
60=60
```

Test2.java

```
1 package org.test;
2
3 import java.util.LinkedHashMap;
4
5 public class Test2 {
6     public static void main(String[] args) {
7         Map<Integer, Integer> q = new LinkedHashMap();
8         q.put(10, 20);
9         q.put(20, 20);
10        q.put(30, 30);
11        q.put(40, 40);
12        q.put(50, 50);
13        q.put(60, 60);
14        q.put(10, 50);
15        q.put(50, 40);
16        q.put(40, 40);
17        Set<Entry<Integer, Integer>> entrySet = q.entrySet();
18        for (Entry<Integer, Integer> entry : entrySet) {
19            System.out.println(entry);
20        }
21    }
22 }
23
24 }
```

Console

```
<terminated> Test2 [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.wi
10=50
20=20
30=30
40=40
50=40
60=60
```

```
Test.java ✘
1 package org.test;
2
3④ import java.util.ArrayList;⑤
4
5 public class Test {
6     public static void main(String[] args) {
7
8         Map<Integer, String> q =new HashMap();
9         q.put(10, "java");
10        q.put(20, "sql");
11        q.put(30, "oops");
12        q.put(40, "sql");
13        q.put(50, "oracle");
14        q.put(60, "DB");
15        q.put(10, "selenium");
16        q.put(50, "pgsql");
17        q.put(40, "Hadoop");
18        Set<Entry<Integer, String>> entrySet = q.entrySet();
19        for (Entry<Integer, String> entry : entrySet) {
20            System.out.println(entry);
21        }
22    }
23}
24
25}
26
27}
28
29}
30
31}
32
33}
34}
35}
36}
```

### Console ✘

```
<terminated> Test (6) [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot
50=pgsql
20=sql
40=Hadoop
10=selenium
60=DB
30=oops
```

```
Test4.java
1 package org.test;
2
3+ import java.util.Hashtable;
4
5
6 public class Test4 {
7     public static void main(String[] args) {
8         Map<String, String> u= new Hashtable();
9         u.put("vel", "Selenium");
10        u.put("Ganesh", "framework");
11        u.put("Dinesh", "oracle");
12        u.put("Venkat", "corejava");
13        u.put("subash", "jira");
14        Set<Entry<String, String>> entrySet = u.entrySet();
15        for (Entry<String, String> entry : entrySet) {
16            System.out.println(entry);
17        }
18    }
19
20}
21
22}
23}
24
```

```
Console
<terminated> Test4 [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse.justj.openjdk.java_11.0.1_v20200922-1200\lib\jre\bin\java.exe
Ganesh=framework
Venkat=corejava
vel=Selenium
Dinesh=oracle
subash=jira
```

QUESTION 5:

QUESTION 5.1:

Description : Create a HashMap with the below key and values and iterate it using enhanced for loop and get the key and values combination.

key	:	10,20,30,40,50,60,10,50,40
values	:	java,sql,oops,Sql,oracle,DB,selenium,pgsql,Hadoop,

Sample Output:

-----

10

java

20

sql

30

oops

40

sql

Description : like this you have to iterate the map

QUESTION 5.2:

-----

Description : Create a LinkedHashMap with the below key and values and iterate it using enhanced for loop and get the key and values combination..

key : 10,20,30,40,50,60,10,50,40
Values : 10,20,30,40,50,60,10,50,40

QUESTION 5.3:

Description : Create a TreeHashMap with the below key and values and iterate it using enhanced for loop and get the key and values combination.

key : !,@,#,\$,%,&,*,(,
Values : 10,20,30,40,50,60,10,50,409

QUESTION 5.4:

Description : Create a HashTable with the below key and values and iterate it using enhanced for loop and get the key and values combination.

Key : vel,Ganesh,Dinesh,Vengat,subash
Values : Selenium,framework,oracle,corejava,jira

The screenshot shows an IDE interface with two main panes: a code editor and a console window.

**Code Editor (Test3.java):**

```
1 package org.test;
2 import java.util.Map;
3 public class Test3 {
4     public static void main(String[] args) {
5         Map<String, Integer> a = new TreeMap();
6         a.put("!", 10);
7         a.put("@", 20);
8         a.put("#", 30);
9         a.put("$", 40);
10        a.put("%", 50);
11        a.put("^", 60);
12        a.put("&", 10);
13        a.put("*", 50);
14        a.put("//", 40);
15        Set<Entry<String, Integer>> entrySet = a.entrySet();
16        for (Entry<String, Integer> entry : entrySet) {
17            System.out.println(entry.getKey());
18            System.out.println(entry.getValue());
19        }
20    }
21 }
22 }
```

**Console Window:**

```
<terminated> Test3
!
10
#
30
$
40
%
50
&
10
(
40
*
50
@
20
^
60
```

The screenshot shows the Eclipse IDE interface with two open windows: 'Test2.java' and 'Console'.

**Test2.java:**

```
1 package org.test;
2
3 import java.util.LinkedHashMap;
4
5 public class Test2 {
6     public static void main(String[] args) {
7         Map<Integer, Integer> q = new LinkedHashMap();
8         q.put(10, 20);
9         q.put(20, 20);
10        q.put(30, 30);
11        q.put(40, 40);
12        q.put(50, 50);
13        q.put(60, 60);
14        q.put(10, 50);
15        q.put(50, 40);
16        q.put(40, 40);
17        Set<Entry<Integer, Integer>> entrySet = q.entrySet();
18        for (Entry<Integer, Integer> entry : entrySet) {
19            System.out.println(entry.getKey());
20            System.out.println(entry.getValue());
21        }
22    }
23 }
24 }
```

**Console:**

```
<terminated> Test2 [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32
10
50
20
20
30
30
40
40
50
40
60
60
```

The screenshot shows the Eclipse IDE interface. The top part is the Java code editor for a file named 'Test.java'. The code defines a class 'Test' with a main method that prints key-value pairs from a HashMap. The bottom part is the 'Console' tab, which displays the output of the program's execution.

```
1 package org.test;
2
3+ import java.util.ArrayList;[]
16
17 public class Test {
18     public static void main(String[] args) {
19
20         Map<Integer, String> q =new HashMap();
21         q.put(10, "java");
22         q.put(20, "sql");
23         q.put(30, "oops");
24         q.put(40, "sql");
25         q.put(50, "oracle");
26         q.put(60, "DB");
27         q.put(10, "selenium");
28         q.put(50, "psql");
29         q.put(40, "Hadoop");
30         Set<Entry<Integer, String>> entrySet = q.entrySet();
31         for (Entry<Integer, String> entry : entrySet) {
32             System.out.println(entry.getKey());
33             System.out.println(entry.getValue());
34         }
35     }
36 }
```

```
<terminated> Test (6) [Java Application] C:\Users\navee\.p2\pool\plugins\org.eclipse.justj.ope
50
psql
20
sql
40
Hadoop
10
selenium
60
DB
30
oops
```

The screenshot shows an IDE interface with two main panes. On the left is the code editor for `Test4.java`, and on the right is the console window.

**Code Editor (Test4.java):**

```
1 package org.test;
2
3 import java.util.Hashtable;
4
5 public class Test4 {
6     public static void main(String[] args) {
7         Map<String, String> u = new Hashtable();
8         u.put("vel", "Selenium");
9         u.put("Ganesh", "framework");
10        u.put("Dinesh", "oracle");
11        u.put("Venkat", "corejava");
12        u.put("subash", "jira");
13        Set<Entry<String, String>> entrySet = u.entrySet();
14        for (Entry<String, String> entry : entrySet) {
15            System.out.println(entry.getKey());
16            System.out.println(entry.getValue());
17        }
18    }
19 }
20 }
```

**Console Output:**

```
<terminated> Test4
Ganesh
framework
Venkat
corejava
vel
Selenium
Dinesh
oracle
subash
jira
```

**QUESTION 6:**

Description: Create a userdefine Map and insert the 10 employee details  
Key : empId, name, phone, address, dob, doj, eMail, gender, Sal  
Give the related values for key for each employee

**QUESTION 7:**

Description: Create a userdefine Map and insert the 10 Student details  
Key : stdId, stdName, stdPhone, address, dOB, eMail, gender  
Give the related values for key for each Student

DAY10:  
-----  
1.Exception  
2.throw and throws  
3.User define exception

QUESTIONS (Programs)

QUESTION 1:

Description : Find the output for the program:

```
public class Test
{
    public static void main(String[] args)
    {
        try
        {
            System.out.printf("1");
            int sum = 9 / 0;
            System.out.printf("2");
        }
        catch(ArithmaticException e)
        {
            System.out.printf("3");
        }
        catch(Exception e)
        {
            System.out.printf("4");
        }
        finally
        {
            System.out.printf("5");
        }
    }
}
```

QUESTION 2:

Description : Find the output for the program:

```
public class Test
{
    private void m1()
    {
        m2();
        System.out.printf("1");
    }
    private void m2()
    {
        m3();
        System.out.printf("2");
    }
    private void m3()
    {
        System.out.printf("3");
    }
    try
    {
        int sum = 4/0;
    }
```

```

        System.out.printf("4");
    }
catch(ArithmetricException e)
{
    System.out.printf("5");
}

System.out.printf("7");
}
public static void main(String[] args)
{
    Test obj = new Test();
    obj.m1();
}
}

```

QUESTION 3:

Description : Find the output for the program:

```

public class Test
{
public static void main(String[] args)
{
    try
    {
        System.out.printf("1");
        int data = 5 / 0;
    }
catch(ArithmetricException e)
{
    System.out.printf("2");
    System.exit(0);
}
finally
{
    System.out.printf("3");
}
System.out.printf("4");
}
}

```

QUESTION 4:

Description : Find the output for the program:

```

public class Test
{
public static void main(String[] args)
{
    try
    {
        System.out.printf("1");
        int data = 5 / 0;
    }
catch(ArithmetricException e)
{
    Throwble obj = new Throwble("Sample");
    try
    {
        throw obj;
    }
}
}

```

```

        }
    catch (Throwable e1)
    {
        System.out.printf("8");
    }
}
finally
{
    System.out.printf("3");
}
System.out.printf("4");
}
}
}

```

QUESTION 5:

-----  
Description : Find the output for the program:

```

import java.io.EOFException;
import java.io.IOException;

public class Test
{
    public static void main(String[] args)
    {
        try
        {
            System.out.printf("1");
            int value = 10 / 0;
            throw new IOException();
        }
        catch(EOFException e)
        {
            System.out.printf("2");
        }
        catch(ArithmaticException e)
        {
            System.out.printf("3");
        }
        catch(NullPointerException e)
        {
            System.out.printf("4");
        }
        catch(IOException e)
        {
            System.out.printf("5");
        }
        catch(Exception e)
        {
            System.out.printf("6");
        }
    }
}

```

Test.java

```
1 package org.test;
2
3 public class Test {
4     public static void main(String[] args)
5     {
6         try
7         {
8             System.out.printf("1");
9             int data = 5 / 0;
10        }
11    catch(ArithmeticException e)
12    {
13        Throwable obj = new Throwable("Sample");
14        try
15        {
16            throw obj;
17        }
18        catch (Throwable e1)
19        {
20            System.out.printf("8");
21        }
22    }
23    finally
24    {
25        System.out.printf("3");
26    }
27    System.out.printf("4");
28 }
29 }
```

Test.java

```
1 package org.test;
2
3 public class Test {
4     public static void main(String[] args)
5     {
6         try
7         {
8             System.out.printf("1");
9             int data = 5 / 0;
10        }
11    catch(ArithmeticException e)
12    {
13        System.out.printf("2");
14        System.exit(0);
15    }
16    finally
17    {
18        System.out.printf("3");
19    }
20    System.out.printf("4");
21 }
22 }
```

The screenshot shows an IDE interface with two main panes. The left pane displays the code for `Test.java`, and the right pane shows the `Console` output.

**Test.java:**

```
1 package org.test;
2
3 public class Test {
4     private void m1()
5     {
6         m2();
7         System.out.printf("1");
8     }
9     private void m2()
10    {
11        m3();
12        System.out.printf("2");
13    }
14    private void m3()
15    {
16        System.out.printf("3");
17    }
18
19    int sum = 4/0;
20    System.out.printf("4");
21}
22 catch(ArithmException e)
23{
24    System.out.printf("5");
25}
26
27 System.out.printf("7");
28}
29 public static void main(String[] args)
30{
31    Test obj = new Test();
32    obj.m1();
33}
34}
35}
```

**Console Output:**

```
<terminated> Test
35721
```

```
Test.java ✘
1 package org.test;
2
3 public class Test {
4     public static void main(String[] args)
5     {
6         try
7     {
8             System.out.printf("1");
9             int sum = 9 / 0;
10            System.out.printf("2");
11        }
12        catch(ArithmeticException e)
13        {
14            System.out.printf("3");
15        }
16        catch(Exception e)
17        {
18            System.out.printf("4");
19        }
20        finally
21        {
22            System.out.printf("5");
23        }
24    }
25}
26
27
```

```
Console ✘
<terminated> Test (6) [Java Application] C:\Users\navee\.p2\pool\p
|135
```

The screenshot shows an IDE interface with two main panes. The left pane displays the Java code for a class named 'Test'. The right pane shows the console output.

```
1 package org.test;
2
3 import java.io.EOFException;
4 import java.io.IOException;
5
6 public class Test {
7     public static void main(String[] args)
8     {
9         try
10     {
11         System.out.printf("1");
12         int value = 10 / 0;
13         throw new IOException();
14     }
15     catch(EOFException e)
16     {
17         System.out.printf("2");
18     }
19     catch(ArithmetricException e)
20     {
21         System.out.printf("3");
22     }
23     catch(NullPointerException e)
24     {
25         System.out.printf("4");
26     }
27     catch(IOException e)
28     {
29         System.out.printf("5");
30     }
31     catch(Exception e)
32     {
33         System.out.printf("6");
34     }
35     }
36 }
```

The console output is as follows:

```
<terminate
13
```

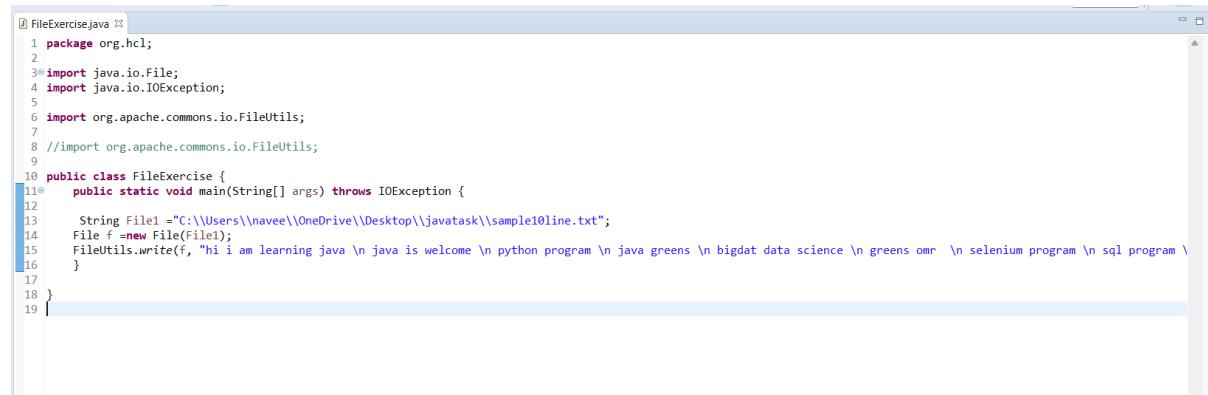
## DAY12-JAVA TASK

File Operation:

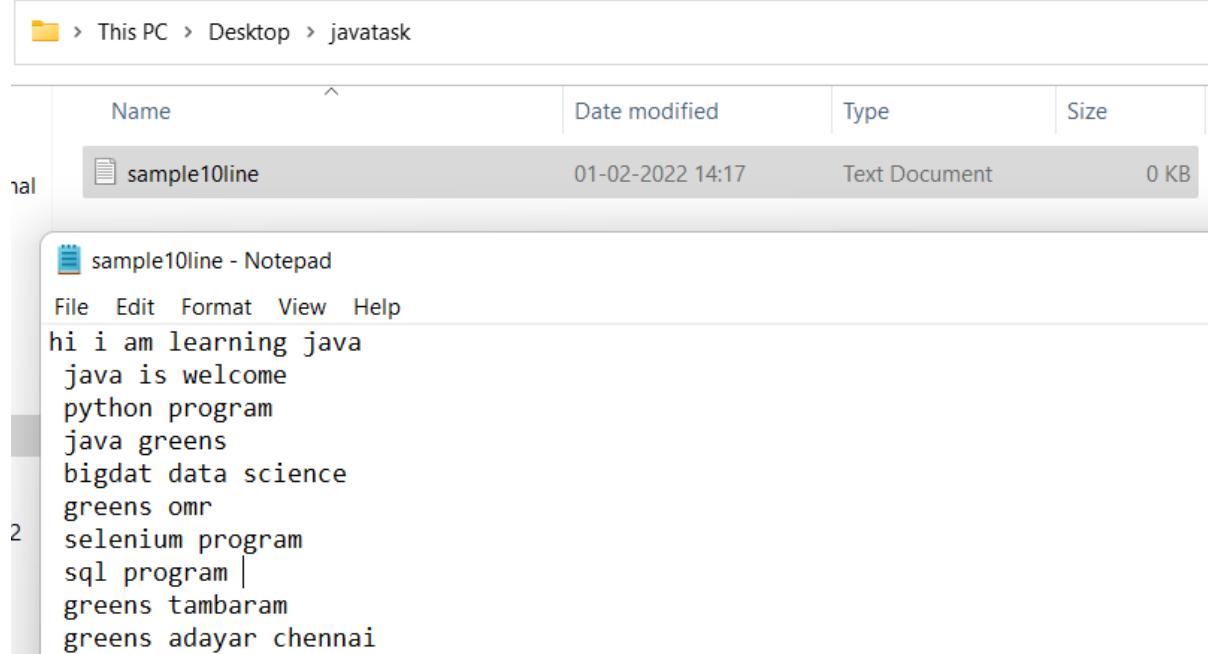
QUESTIONS (Practical)

-----  
QUESTION 1  
-----

NOTE: Create a new file  
write the file with some 10 lines of text.



```
FileExercise.java
1 package org.hcl;
2
3 import java.io.File;
4 import java.io.IOException;
5
6 import org.apache.commons.io FileUtils;
7
8 //import org.apache.commons.io FileUtils;
9
10 public class FileExercise {
11     public static void main(String[] args) throws IOException {
12
13         String File1 ="C:\\\\Users\\\\navee\\\\OneDrive\\\\Desktop\\\\javatask\\\\sample10line.txt";
14         File f =new File(File1);
15         FileUtils.write(f, "hi i am learning java \n java is welcome \n python program \n java greens \n bigdat data science \n greens omr \n selenium program \n sql program \n");
16     }
17
18 }
```



This PC > Desktop > javatask

Name	Date modified	Type	Size
sample10line	01-02-2022 14:17	Text Document	0 KB

sample10line - Notepad

```
File Edit Format View Help
hi i am learning java
java is welcome
python program
java greens
bigdat data science
greens omr
selenium program
sql program |
greens tambaram
greens adayar chennai
```

QUESTION 2

-----

NOTE: Retrieve the text from the file  
Check "java" word contains or not?

QUESTION 3

-----

NOTE: Find the row count from the file.

The screenshot shows the Eclipse IDE interface. The top window is titled "FileExercise.java" and displays the following Java code:

```
1 package org.hcl;
2
3 import java.io.BufferedReader;
4 import java.io.File;
5 import java.io.FileInputStream;
6 import java.io.FileReader;
7 import java.io.IOException;
8 import java.nio.file.Files;
9 import java.nio.file.Path;
10 import java.nio.file.Paths;
11 import java.util.Scanner;
12
13 import org.apache.commons.io.FileUtils;
14
15 //import org.apache.commons.io.FileUtils;
16
17 public class FileExercise {
18     public static void main(String[] args) throws IOException {
19         File file = new File("D://SampleFile.txt");
20         FileInputStream fis = new FileInputStream(file);
21         byte[] byteArray = new byte[(int)file.length()];
22         fis.read(byteArray);
23         String data = new String(byteArray);
24         String[] stringArray = data.split("\r\n");
25         System.out.println("Number of lines in the file are ::"+stringArray.length);
26     }
27 }
```

The code reads a file named "SampleFile.txt" located at "D://SampleFile.txt". It uses a FileInputStream to read the file into a byte array, then converts it to a String. Finally, it splits the string into an array of lines using "\r\n" as the delimiter and prints the number of lines.

The bottom window is titled "Console" and shows the output of the application:

```
<terminated> FileExercise [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64\lib\javase\bin\java.exe -jar C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64\lib\javase\bin\java.jar
Number of lines in the file are ::7
```

QUESTION 4

-----

NOTE: Print the last 5 lines from the file.

The screenshot shows a Java IDE interface. On the left is a code editor titled "ExeDemo.java" containing Java code. On the right is a "Console" window showing the output of the program. The code in the editor is as follows:

```
1 package org.hcl;
2
3 import java.io.File;
4 import java.io.IOException;
5 import java.io.LineNumberReader;
6 import java.nio.file.Files;
7 import java.nio.file.Path;
8 import java.nio.file.Paths;
9
10 import org.apache.commons.io.input.ReversedLinesFileReader;
11
12 public class ExeDemo {
13
14     public static void main(String[] args) {
15         File file = new File("D:\\SampleFile.txt");
16         int n_lines = 5;
17         int counter = 0;
18         ReversedLinesFileReader object = null;
19         try {
20             object = new ReversedLinesFileReader(file);
21         } catch (IOException e) {
22             // TODO Auto-generated catch block
23             e.printStackTrace();
24         }
25         while(counter < n_lines) {
26             try {
27                 System.out.println(object.readLine());
28             } catch (IOException e) {
29                 // TODO Auto-generated catch block
30                 e.printStackTrace();
31             }
32             counter++;
33         }
34     }
35 }
```

The console output shows the following lines:

```
hi i am learning python10
learning java9
hi i am java8
hi i am learning java7
hi i am learning java6
```

#### QUESTION 5

---

NOTE: Print the odd lines from the file.

```
1 package org.hcl;
2
3 import java.io.BufferedReader;
4 import java.io.File;
5 import java.io.FileReader;
6 import java.io.IOException;
7
8 public class FileOperationPractice {
9     //public static void main(String[] args) {
10     public static void main(String[] args) throws IOException {
11         File fileIn1 = new File("D:\\\\SampleFile.txt");
12
13         try (BufferedReader bf = new BufferedReader(new FileReader(fileIn1))) { //SURROUND WITH try with resources FOR THE EXCEPTIONS
14
15             String readLine;
16             int line = 0;
17
18             while ((readLine = bf.readLine()) != null) {
19                 if (line % 2 != 0) {//CHECKING WHETHER THE ROW IS ODD
20                     System.out.println(readLine);
21                 }
22                 line++;
23             }
24         }
25     }
26 }
```

#### QUESTION 6

---

NOTE: Count the number of duplicate words available in the file.

\*FileOperationPractice.java

```
1 package org.hcl;
2
3 import java.io.File;
4
5 public class FileOperationPractice {
6     public static void main(String[] args) {
7         File file = new File("D:\\Java");
8
9         boolean mkdir = file.mkdir();
10        System.out.println();
11    }
12 }
13
```

QUESTION 7

-----

NOTE: Check directory D:\\Java is available or not.

FileOperationPractice.java

```
1 package org.hcl;
2
3 import java.io.File;
4
5 public class FileOperationPractice {
6     public static void main(String[] args) {
7
8
9         File f =new File ("D:\\Java");
10        boolean exists = f.exists();
11        System.out.println(exists);
12    }
13 }
```

Console

```
<terminated> FileOperationPractice [Java Application] C:\Users\navee\p
true
```

FileOperationPractice.java

```
1 package org.hcl;
2
3 import java.io.File;
4
5 public class FileOperationPractice {
6     public static void main(String[] args) {
7
8
9         try {
10             File file = new File("D:\\Java");
11             file.createNewFile();
12             System.out.println(file.exists());
13         } catch(Exception e) {
14             e.printStackTrace();
15         }
16     }
17
18 }
19
20 }
```

Console

```
<terminated> FileOperationPractice [Java Application] C:\Users\navee\p2\pool\plugins\org.eclip
true
```

QUESTION 8

-----

NOTE: Check directory D:\\Java is available or not.

If not create new directory.

The screenshot shows the Eclipse IDE interface. The top window is titled "FileOperationPractice.java" and contains the following Java code:

```
1 package org.hcl;
2
3 import java.io.File;
4
5 public class FileOperationPractice {
6     public static void main(String[] args) {
7
8
9         File f = new File("D:\\Java\\Selenium\\Material");
10        boolean mkdirs = f.mkdirs();
11        System.out.println(mkdirs);
12    }
13 }
14
```

The bottom window is titled "Console" and shows the output of the application:

```
<terminated> FileOperationPractice [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_6
true
```

QUESTION 9

-----

NOTE: Create sub directory D:\\Java\\Selenium\\Material.

QUESTION 10

-----

NOTE: Print all the available files from an existing folder.

The screenshot shows the Eclipse IDE interface. The left window is titled "ExeDemo.java" and contains the following Java code:

```
1 package org.hcl;
2
3 import java.io.BufferedReader;
4 import java.io.File;
5 import java.io.IOException;
6 import java.io.LineNumberReader;
7 import java.nio.file.Files;
8 import java.nio.file.Path;
9 import java.nio.file.Paths;
10 import java.util.ArrayList;
11 import java.util.Iterator;
12
13 import org.apache.commons.io.input.ReversedLinesFileReader;
14
15 public class ExeDemo {
16
17     public static void main(String[] args) {
18
19         // creates a file object
20         File file = new File("C:\\\\Users\\\\navee\\\\OneDrive\\\\Desktop");
21
22         // returns an array of all files
23         String[] fileList = file.list();
24
25         for(String str : fileList) {
26             System.out.println(str);
27         }
28     }
29 }
30
```

The right window is titled "Console" and shows the output of the application, listing various files in the specified directory:

```
<terminated> ExeDemo [Java Application] C:\Users\navee\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_6
desktop.ini
eclipse - Shortcut.lnk
File operation
Folderfile
home.html
javatask
Microsoft Edge.lnk
navee.html
NAVEEN123
New Text Document.txt
selenium driver path.txt
SELENIUM2022
sqldeveloper - Shortcut.lnk
Subash Balachandran is inviting you to a scheduled Zoom meeting
syntax for java.txt
Zoom.lnk
```

4

```
package org.hcl;

import java.io.File;
import java.io.IOException;
import java.io.LineNumberReader;
import java.nio.file.Files;
import java.nio.file.Path;
```

```

import java.nio.file.Paths;

import org.apache.commons.io.input.ReversedLinesFileReader;

public class ExeDemo {

    public static void main(String[] args) {
        File file = new File("D:\\SampleFile.txt");
        int n_lines = 5;
        int counter = 0;
        ReversedLinesFileReader object = null;
        try {
            object = new ReversedLinesFileReader(file);
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        while(counter < n_lines) {
            try {
                System.out.println(object.readLine());
            } catch (IOException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
            counter++;
        }
    }
}

```

10

```

package org.hcl;

import java.io.BufferedReader;
import java.io.File;
import java.io.IOException;
import java.io.LineNumberReader;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.Iterator;

import org.apache.commons.io.input.ReversedLinesFileReader;

public class ExeDemo {

    public static void main(String[] args) {

        // creates a file object
        File file = new
File("C:\\\\Users\\\\navee\\\\OneDrive\\\\Desktop");

        // returns an array of all files
String[] fileList = file.list();

        for(String str : fileList) {

```

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

9.

```
package org.hcl;

import java.io.File;

public class FileOperationPractice {
public static void main(String[] args) {

    File f =new File("D:\\Java\\Selenium\\Material");
    boolean mkdirs = f.mkdirs();
    System.out.println(mkdirs);
}
}
```

7.

```
package org.hcl;

import java.io.File;

public class FileOperationPractice {
public static void main(String[] args) {

    try {
        File file = new File("D:\\Java");
        file.createNewFile();
        System.out.println(file.exists());
    } catch(Exception e) {
        e.printStackTrace();
    }
}
}
```

8.

```
package org.hcl;

import java.io.File;

public class FileOperationPractice {
public static void main(String[] args) {
    File file = new File("D:\\Java");

    boolean mkdir = file.mkdir();
    System.out.println();
}
}
```

7.

```
package org.hcl;

import java.io.File;

public class FileOperationPractice {
    public static void main(String[] args) {

        File f =new File ("D:\\Java1");
        boolean exists = f.exists();
        System.out.println(exists);
    }
}
```

5.

```
package org.hcl;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;

public class FileOperationPractice {
    //public static void main(String[] args) {
    public static void main(String[] args) throws IOException {
        File fileIn1 = new File("D:\\SampleFile.txt");

        try (BufferedReader bf = new BufferedReader(new
FileReader(fileIn1))) {//SURROUND WITH try with resources FOR THE EXCEPTIONS

            String readLine;
            int line = 0;

            while ((readLine = bf.readLine()) != null) {
                if (line % 2 != 0) {//CHECKING WHETHER THE ROW IS ODD
                    System.out.println(readLine);
                }
                line++;
            }
        }
    }
}
```