

INSTITUTE VISION AND MISSION

Vision

To emerge as a destination for higher education by transforming learners into achievers by creating, encouraging and thus building a supportive academic environment.

Mission

To impart Quality Technical Education and to undertake Research and Development with a focus on application and innovation which offers an appropriate solution to the emerging societal needs by making the students globally competitive, morally valuable and socially responsible citizens.

DEPARTMENT VISION AND MISSION

Vision

To emerge as a center of excellence with global reputation with adaption of rapid advancements in the field of computer specialization.

Mission

1. To provide a strong theoretical and practical background in area of computer science with an emphasize on software development.
2. To inculcate Professional behavior, strong ethical values, leadership qualities, research capabilities and lifelong learning.
3. To educate students to become effective problem solvers, apply knowledge with social sensitivity for the betterment of the society and humanity as a whole.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

Programme educational objectives are broad statements that describe the career and professional accomplishments that the programme is preparing graduates to achieve within 3 to 5 years after graduation.

The **Programme Educational Objectives** of the B. Tech CSE programme are:

- **PEO1:** To apply the knowledge of mathematics, basic science and engineering solving the real world computing problems to succeed higher education and professional careers.
- **PEO2:** To develop the skills required to comprehend, analyze, design and create innovative computing products and solutions for real life problems.
- **PEO3:** To inculcate professional and ethical attitude, communication and teamwork skills, multi-disciplinary approach and an ability to relate computer engineering issues with social awareness.

LABORATORY OUTCOMES:

CO [1] To Understand OOP concepts and basics of Java programming.

CO[2]. To create Java programs using inheritance and polymorphism.

CO[3]; To Implement error-handling techniques using exception handling and multithreading.

CO[4] To differentiate various collections.

CO[5] To build files and establish database connection.

CO[6] To develop GUI using Swing components.

Do's

1. Come with completed observation and record
2. Wear apron and ID card before entering into the lab.
3. Know the location of the fire extinguisher and the first aid box and how to use them in case of an emergency.
4. Read and understand how to carry out an activity thoroughly before coming to the laboratory.
5. Report any broken plugs or exposed electrical wires to your lecturer/laboratory technician immediately.
6. Write in time, out time and system details in the login register.

Don'ts

1. Do not eat or drink in the laboratory.
2. Do not operate mobile phones in the lab. Keep mobile phones either in silent or switched off mode.
3. Do not change system settings.
4. Do not disturb your neighbouring students. They may be busy in completing tasks.
5. Do not remove anything from the computer laboratory without permission.
6. Do not use pen drives.
7. Do not misbehave.

INDEX

S.NO	List of Contents	Page.No
1	Use Eclipse or Net bean platform and acquaint with the various menus. Create a test project, add a test class, and run it. See how you can use auto suggestions, auto fill. Try code formatter and code refactoring like renaming variables, methods, and classes. Try debug step by step with a small program of about 10 to 15 lines which contains at least one if else condition and a for loop.	7
2	Write a java program that works as a simple calculator. Use a GridLayout to arrange Buttons for digits and for the + - * % operations. Add a text field to display the result. Handle any possible exceptions like divide by zero.	18
3	a) Write an applet program that displays a simple message b) Develop an applet that receives an integer in one text field, and computes its factorial Value and returns it in another text field, when the button named “Compute” is clicked.	22
4	Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the textfields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw NumberFormatException. If Num2 were Zero, the program would throw an ArithmeticException. Display the exception in a message dialog box.	26
5	Write a Java program that implements a multi-threaded program has three threads. First thread generates a random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd the third thread will print the value of cube of the number.	30
6	Write a Java program for the following: i) Create a doubly linked list of elements. ii) Delete a given element from the above list. iii) Display the contents of the list after deletion.	32
7	Write a java program to simulate a traffic light. The program lets the user select one of the three lights: red, yellow or green. On selecting a button, an appropriate message with ”Stop” or “Ready” or “ Go” should appear above the buttons selected color. Initially, there is no message shown.	42
8	Write a java program to create an abstract class named shape that contains two integers and an empty method named print Area(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class shape. Each one of the class contains only the method print Area() that print the area of the given shape	45
9	Suppose that table named Table.txt is stored in a text file. The first line in the file is the header, and the remaining lines correspond to rows in	48

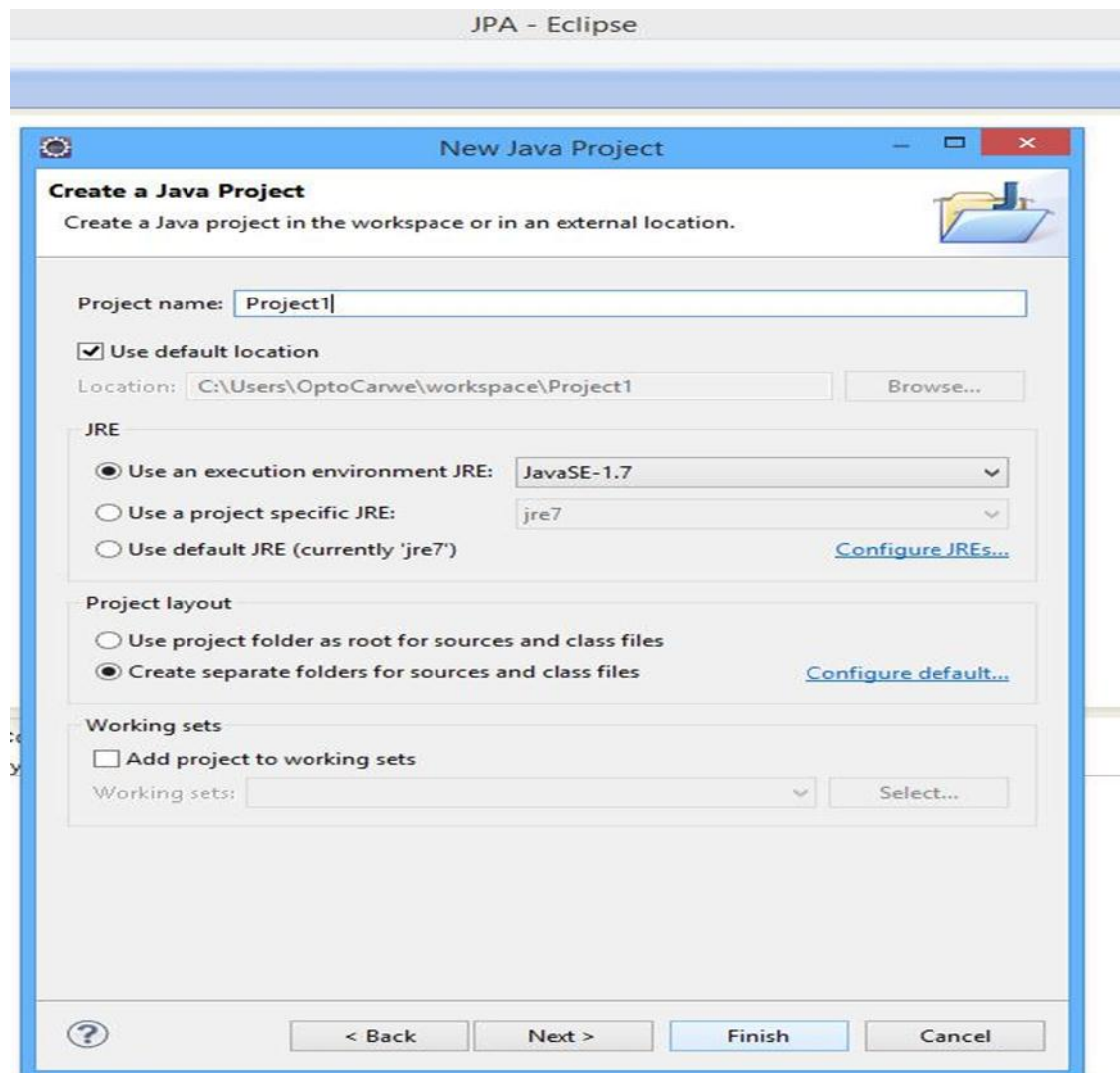
	the table. The elements are separated by commas. Write a java program to display the table using in Grid Layout.	
10	Write a java program that handles all mouse events and shows the event name at the center of the window when mouse event is fired(Use Adapter classes).	50
11	Write a java program that loads names and phone numbers from a text file where the data is organized as one line per record and each field in a record are separated by tab(\t).It takes a name or phone number as input and prints the corresponding other value from the hash table (hint : use hash tables)	52
12	a Java program that correctly implements the producer – consumer problem using the concept of inter-thread communication.	54
13	a Java program to list all the files in a directory including the files present in all its subdirectories.	57
14	Write a Java program that implements Quick sort algorithm for sorting a list of names in ascending order	59
ADDITIONAL PROGRAMS		
1	Write a java program on quick button	62
2	Write a java program on frame icon	65
3	Fibonacci Series using recursion in java	67
4	How Exception Terminates Java Program?& java exception handling example	68

PROGRAM -1

Steps To Execute Simple Java Program Using Eclipse

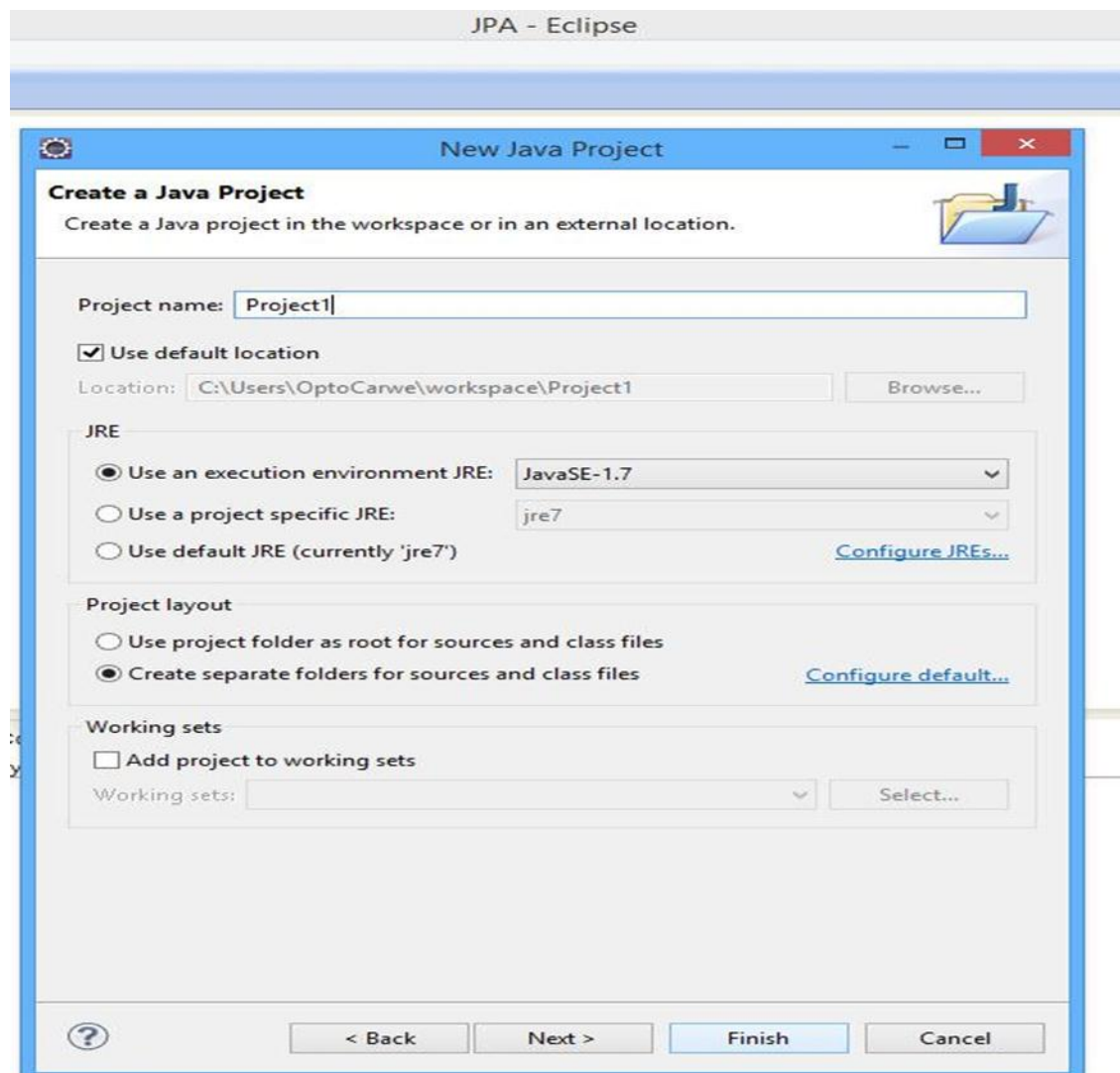
Step1: Begin by creating a new Java project.

There are few different ways of accomplishing this. Click the arrow next to the left-most icon on the toolbar and select “Project” from the drop-down menu. Alternately Start a new Java Project by choosing “File” then “New” followed by “Java Project”. Also use the shortcut Alt+Shift+N.



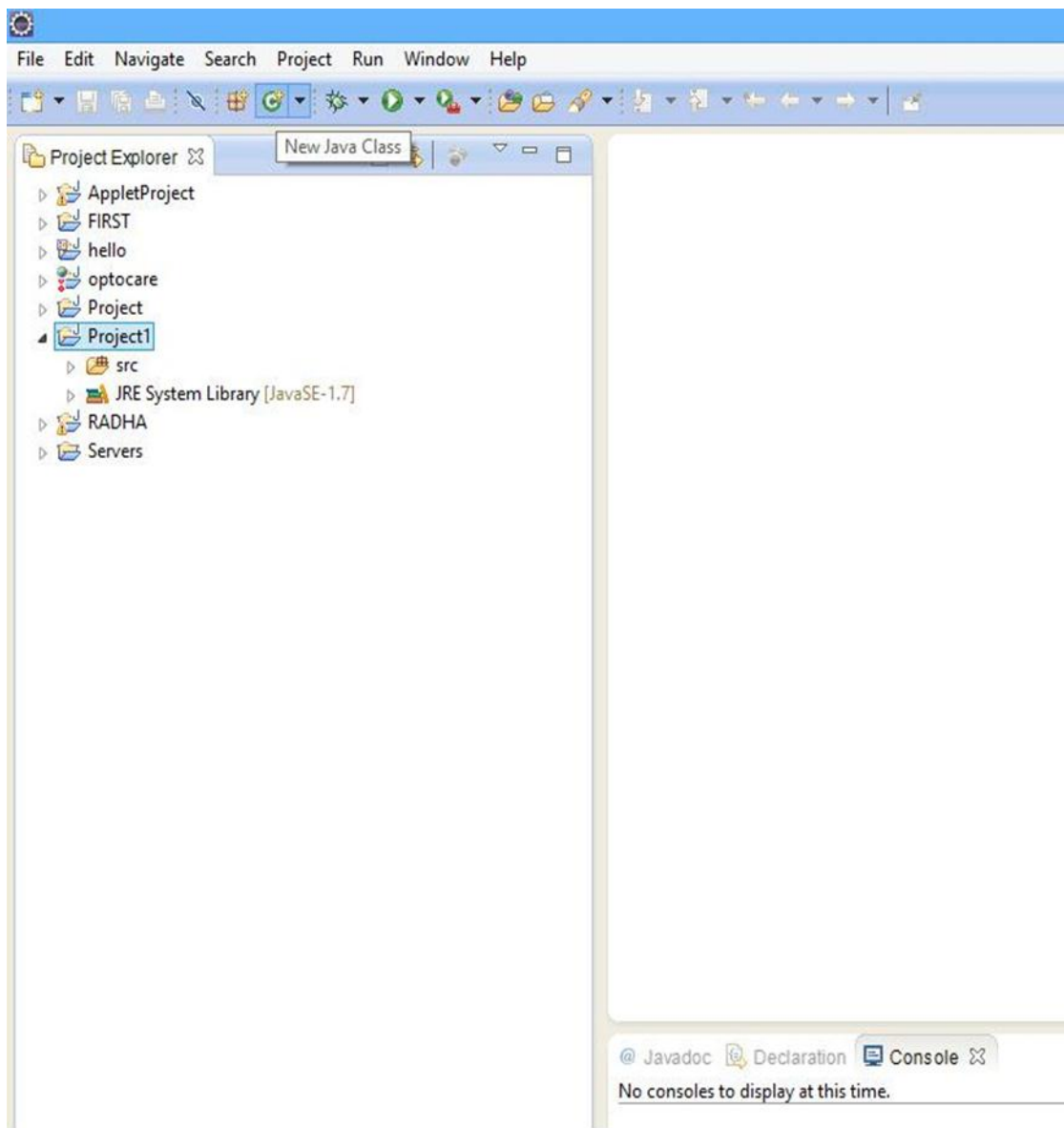
Step2: Enter a Project Name

You will see a window titled “Create a Java Project”. The buttons “Next” and “Finish” at the bottom of the window will be grayed out until a project name is entered in the first field. To proceed, give project name and enter it into this field then click “Finish”. New project will appear on the left-hand side of the screen under “Package Explorer” among existing projects. Projects are listed in alphabetical order.



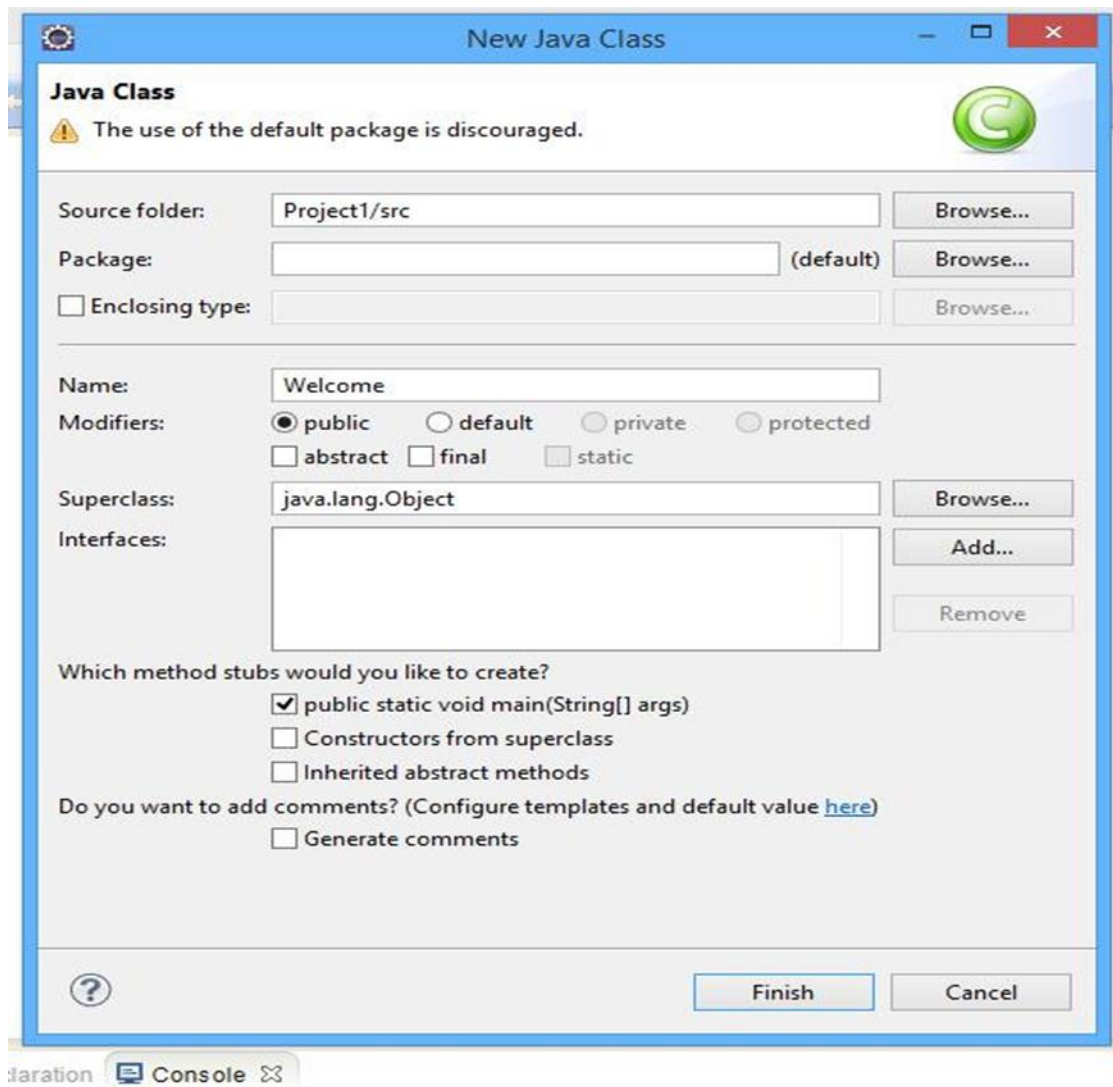
Step3: Start a new java class.

Before begin writing code, need to create a new Java class. A class is a blueprint for an object. It defines the data stored in the object as well as its actions. Create a class by **clicking the “New Java Class” icon, which looks like a green circle with the letter “C” in the center of it.**



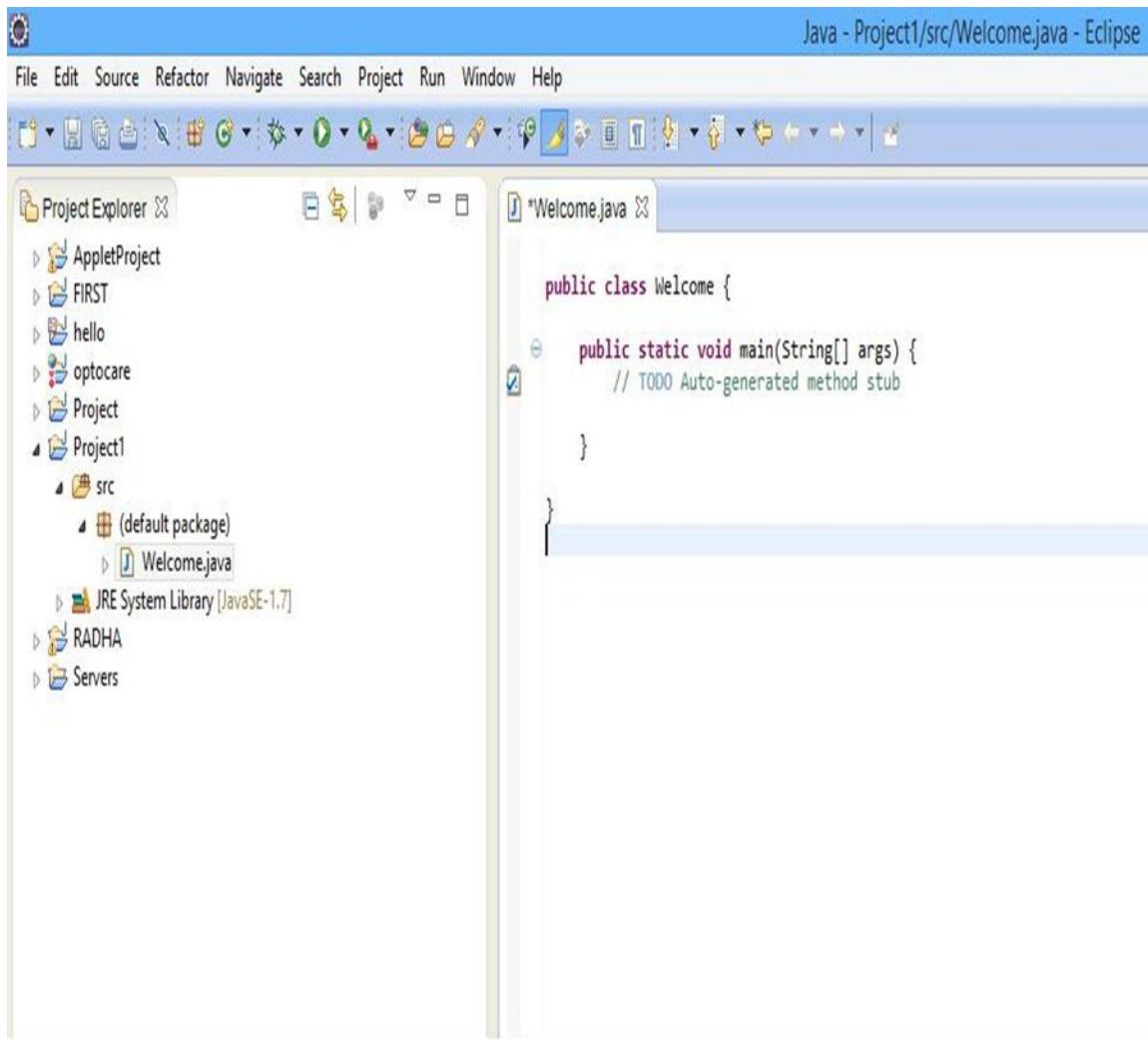
4: Enter the name of your class.

You will see a window titled “Java Class.” To proceed, enter the name of class into the field “Name”. Since the class will be main class of the simple project, check the selection box labeled “public static void main(String[] args)” to include the method stub. Afterwards, click “Finish”.



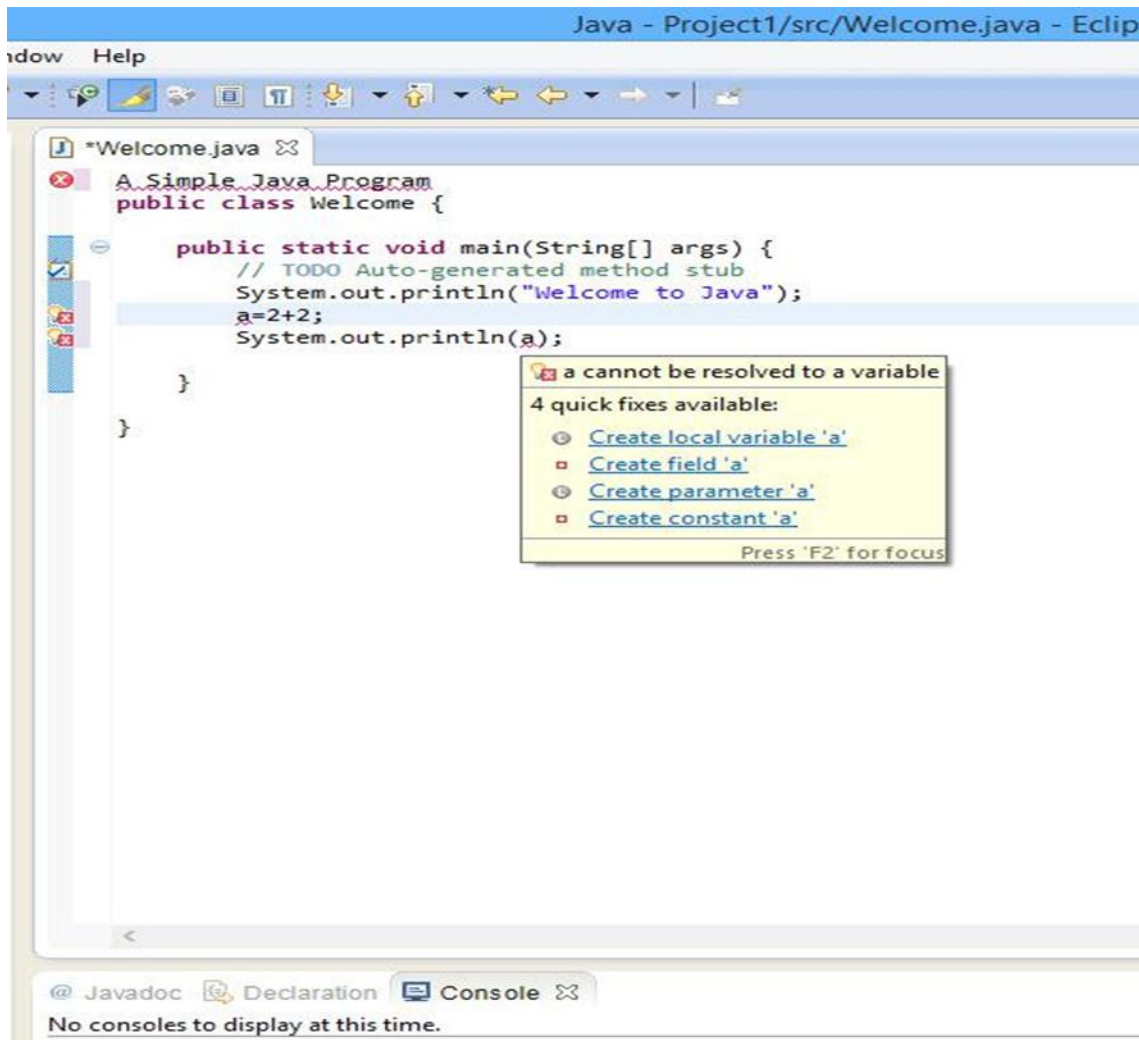
Step5: Enter Java Code.

Here new class Welcome.java is created. It appears with the method stub “**public static void main(String[] args)**” along with some automatically generated comments. A method will contain a sequence of instructions to be executed by the program. A comment is a statement that is ignored by the compiler. Comments are used by programmers to document their code. Edit this file and insert the code for Java Program.



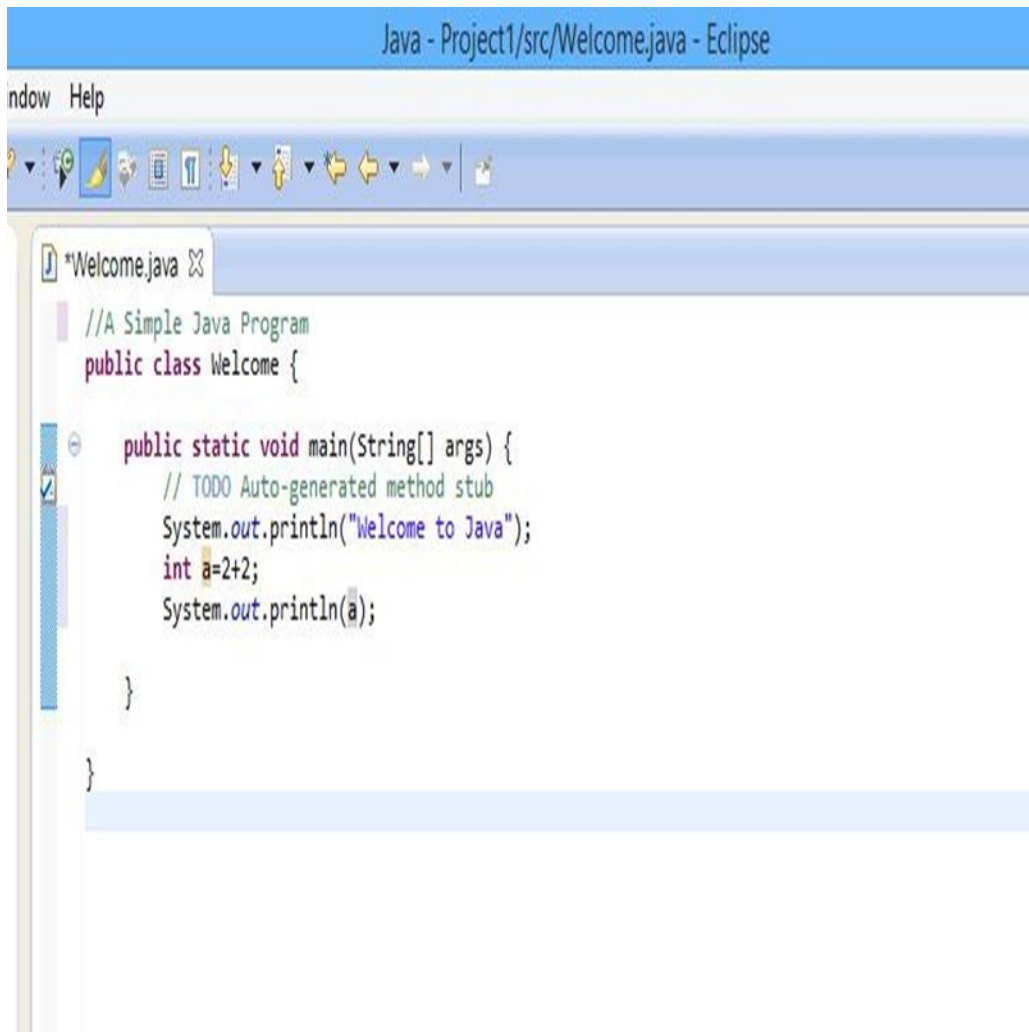
Step6: Watch out for errors in code.

Any errors will be underlined in red, and icon with an "X" will show up on the left. Fix errors. By mousing over an error icon, can see a suggestion box that lists the ways can fix the error.



Step7: Ensure that entire program is free of errors.

There are three types of errors must beware of: syntax errors, run-time errors and logic errors. The compiler will alert syntax errors. Examples of syntax errors are misspelled variable names or missing semi-colons. Until remove all syntax errors from code program will not compile. The compiler will not catch run-time errors or logic errors.

The image shows a screenshot of the Eclipse IDE. The title bar reads "Java - Project1/src/Welcome.java - Eclipse". The menu bar includes "Window" and "Help". Below the menu bar is a toolbar with various icons. The main editor window shows a Java file named "Welcome.java" with the following code:

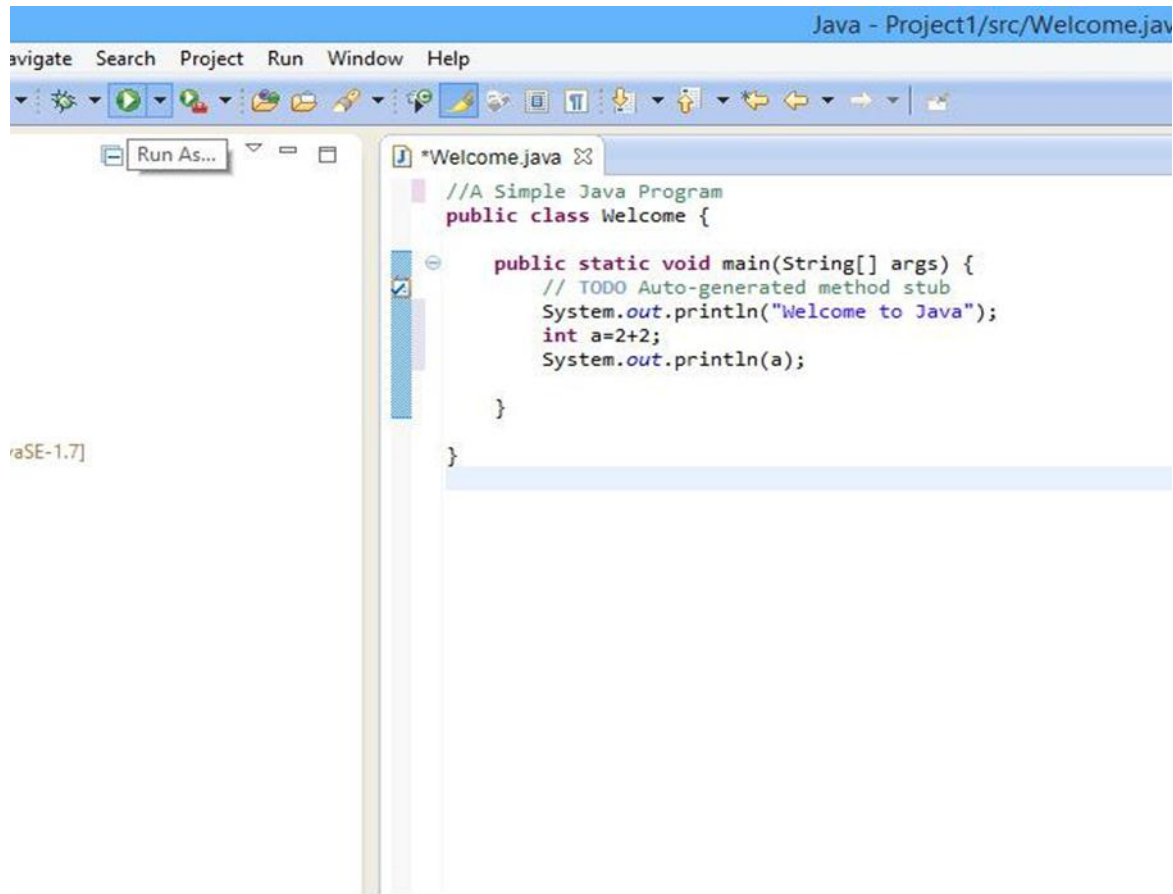
```
//A Simple Java Program
public class Welcome {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println("Welcome to Java");
        int a=2+2;
        System.out.println(a);
    }
}
```

The code is syntactically correct. However, the image is likely intended to illustrate a common mistake, such as a missing semicolon or a misspelled keyword, which would cause a compilation error. In this specific screenshot, the code is valid, but the context of the text above suggests it's part of a lesson on error types.

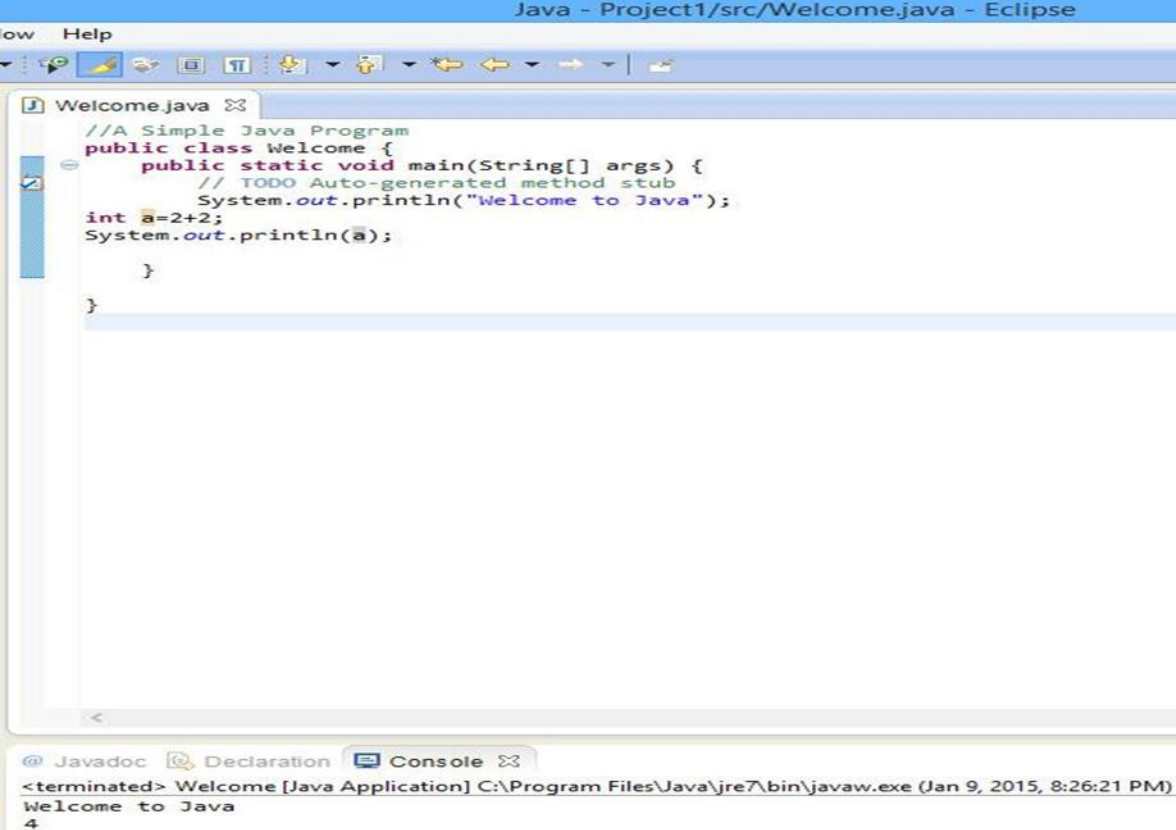
Step8: Compile Java Program.

Now the program is free for errors, click the triangular icon to run program. Another way to run **program** is to select **“Run”** from the main menu and then select **“Run”** again from the drop-down menu. The shortcut is Ctrl+F11.



Step9: Verify the output is what you expected.

When program runs, the output will be displayed on console at the bottom of the screen.



The screenshot shows the Eclipse IDE interface. The main editor window displays the following Java code:

```
//A Simple Java Program
public class Welcome {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println("Welcome to Java");
        int a=2+2;
        System.out.println(a);
    }
}
```

At the bottom of the IDE, the Console window is open, showing the output of the program:

```
<terminated> Welcome [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (Jan 9, 2015, 8:26:21 PM)
Welcome to Java
4
```

Step10: Fix any run-time or logic errors.

If the output is different from what you expected, then there might have been an error even though the program compiled. For example, if the output was zero instead of **four**, then there was a mistake in the program's calculation.

PROGRAM – 2

Create a test project, add a test class, and run it

Aim: Use Eclipse or Net bean platform and acquaint with the various menus. Create a testproject, add a test class, and run it. See how you can use auto suggestions, auto fill. Try code formatter and code refactoring like renaming variables, methods, and classes. Try debug step by step with a small program of about 10 to 15 lines which contains at least one if else condition and a for loop.

Program:

```
class PrimeNumbers
{

public static void main(String[] args)
{
try
{
System.out.println("***** PRIME NUMBERS *****");

Scanner objScanner = new Scanner(System.in);

System.out.print("\n Enter n Value:");
long n = objScanner.nextInt();

for (long i = 2; i <= n; i++)
{
boolean isprime = isNumPrime(i);
if (isprime)
{
System.out.print(i + " ");
}
}
catch (Exception e)
{
e.printStackTrace();
}

public static boolean isNumPrime(long number)
{
boolean result = true;

for (long i = 2; i <= number / 2; i++)
{
if ((number % i) != 0)
```



```
{
result = true;
}
else
{
result = false;
break;
}
}

return result;
}
}
```

VIVA QUESTIONS:

1. What do you mean by Platform Independence?

Platform Independence you can run and compile program in one platform and can execute in any other platform.

2. What is JIT Compiler?

Just-In-Time(JIT) compiler is used to improve the performance. JIT compiles parts of the byte code that have similar functionality at the same time

3. What all memory areas are allocated by JVM?

Heap, Stack, Program Counter Register and Native Method Stack

4. What is the base class of all classes?

java.lang.Object

5. What are two different ways to call garbage collector?

System.gc() OR Runtime.getRuntime().gc().

6. Use of finalize() method in java?

finalize() method is used to free the allocated resource.

7. List two java IDEs?

1.Eclipse, 2.Net beans and 3.IntelliJ

8. What are java buzzwords?

Java buzzwords explain the important features of java. They are Simple,Secured, Portable, architecture neutral, high performance, dynamic, robust,interpreted etc.

9. Is byte code is similar to .obj file in C?

Yes, both are machine understandable codes No, .obj file directly understood by machine, byte code requires JVM.

10.What are length and length() in Java?

Both gives number of char/elements, length is variable defined in Array class,length() is method defined in String class.

PROGRAM -2

Design of Simple Calculator

Aim: Write a java program that works as a simple calculator. Use a GridLayout to arrange Buttons for digits and for the + - * % operations. Add a text field to display the result. Handle any possible exceptions like divide by zero.

THEORY: GridLayout is one of the Layout managers. A layout manager automatically arranges your controls within a window by using some type of algorithm. Grid Layout lays out components in a two-dimensional grid. When you instantiate a GridLayout, you define the number of rows and columns.

Program:

```
import java.awt.*; import
java.awt.event.*; import
java.applet.*;
```

```
/*<applet code="Cal" width=300 height=300></applet>*/
```

```
public class Cal extends Applet implements ActionListener
{
String msg=" "; int v1,v2,result;
TextField t1;
Button b[]=new Button[10];
Button add,sub,mul,div,clear,mod,EQ; char OP;
public void init()
{
Color k=new Color(120,89,90); setBackground(k);
t1=new TextField(10);
GridLayout gl=new GridLayout(4,5); setLayout(gl);
for(int i=0;i<10;i++)
{
b[i]=new Button(""+i);
}
add=new Button("add"); sub=new Button("sub"); mul=new Button("mul"); div=new Button("div");
mod=new Button("mod"); clear=new Button("clear"); EQ=new Button("EQ"); t1.addActionListener(this);

add(t1);

for(int i=0;i<10;i++)
{
add(b[i]);}
}
```

```
add(add);
add(sub);
add(mul);
add(div);
add(mod);
add(clear);
add(EQ);
for(int i=0;i<10;i++)
{
b[i].addActionListener(this);
}
add.addActionListener(this);
sub.addActionListener(this);
mul.addActionListener(this);
div.addActionListener(this);
mod.addActionListener(this);
clear.addActionListener(this);
EQ.addActionListener(this);
}
public void actionPerformed(ActionEvent ae)
{
String str=ae.getActionCommand(); char ch=str.charAt(0);
if ( Character.isDigit(ch)) t1.setText(t1.getText()+str);
else
if(str.equals("add"))
{
v1=Integer.parseInt(t1.getText());
OP='+';
t1.setText("");
}
else if(str.equals("sub"))
{
v1=Integer.parseInt(t1.getText()); OP='-';
t1.setText("");
} else if(str.equals("mul"))
{
v1=Integer.parseInt(t1.getText());
OP='*';
t1.setText("");
}
}
else if(str.equals("div"))
{
```

```
v1=Integer.parseInt(t1.getText());
OP='/';
t1.setText("");
}
else if(str.equals("mod"))
{
v1=Integer.parseInt(t1.getText());
OP='%';
t1.setText("");
}
if(str.equals("EQ"))
{
v2=Integer.parseInt(t1.getText());
if(OP=='+')
result=v1+v2; else if(OP=='-')
result=v1-v2; else if(OP=='*')
result=v1*v2; else if(OP=='/')
result=v1/v2; else if(OP=='%')
result=v1%v2;
t1.setText(""+result);
}
if(str.equals("clear"))
{
t1.setText("");
}
}
}
```

VIVA QUESTIONS:

1. What is object cloning?

The object cloning is used to create the exact copy of an object.

2. When parseInt() method can be used?

This method is used to get the primitive data type of a certain String.

3. java.util.regex consists of which classes?

java.util.regex consists of three classes – Pattern class, Matcher class and PatternSyntaxException class.

4. Which package is used for pattern matching with regular expressions?

java.util.regex package is used for this purpose.

5. Define immutable object?

An immutable object can't be changed once it is created.

6. Explain Set Interface?

It is a collection of element which cannot contain duplicate elements. The Set interface contains only methods inherited from Collection and adds the restriction that duplicate elements are prohibited.

7. What is function overloading?

If a class has multiple functions by same name but different parameters, it is known as Method Overloading.

8. What is function overriding?

If a subclass provides a specific implementation of a method that is already provided by its parent class, it is known as Method Overriding.

9. What is the difference between yielding and sleeping?

When a task invokes its yield() method, it returns to the ready state. When a task invokes its sleep() method, it returns to the waiting state.

10. What are Wrapper classes?

These are classes that allow primitive types to be accessed as objects. Example: Integer, Character, Double, Boolean etc.

PROGRAM -3(a)

Simple Applet Creation

Aim: Write an applet program that displays a simple message

THEORY:

Applets are designed to bring the web alive. they function to add animation sound and eventually complete multi media into HTML documents. java is also part of the future of interfacing with virtual-reality environments implemented via VRML. At present ,java is limited only by the capabilities of the internet itself. applets are java programs that are specialized for use over the Web.

Program:

Applet1.java:

```
// Import the packages to access the classes and methods in awt and applet classes.
```

```
import java.awt.*; import
java.applet.*;
```

```
public class Applet1 extends Applet
{
// Paint method to display the message.
```

```
public void paint(Graphics g)
{
g.drawString("HELLO WORLD",20,20);
}}
```

Applet1.html:

```
/* <applet code="Applet1" width=200 height=300></applet>*/
```

PROGRAM -3(b)

Factorial using applet

Aim: Write a java program that Develop an applet that receives an integer in one text field, and computes its factorial Value and returns it in another text field, when the button **named “Compute” is clicked.**

THEORY: Applets are designed to bring the web alive.they function to add animation sound and eventually complete multi media into HTML documents.java is also part of the future of interfacing with virtual-reality environments implemented via VRML.At present ,java is limited only by the capabilities of the internet itself.applets are java programs that are specialized for use over the Web. The Applet life cycle The init()Method: The init()method is where your applet does much of its setup,such as defined its layout,parsingparameters,or setting the background colors. The starts() Method: The start()method is used mainly when implementing threads in java. The stop() Mehtod: The stop() method is used to do what its name suggests: stop what is going on. The destroy() method: when it is called,the applet is told to free up system resources.

Program:

```
import java.awt.*; import
java.awt.event.*;
import java.applet.Applet;
public class Fact extends Applet implements ActionListener
{
Label l1,l2; TextField
t1,t2; Button b1;
public void init(){
l1=new Label("enter the value"); add(l1);
t1=new TextField(10); add(t1);
b1=new Button("Factorial"); add(b1);
b1.addActionListener(this);
l2=new Label("Factorial of given no is"); add(l2);
t2=new TextField(10); add(t2);
}
public void actionPerformed(ActionEvent e)
{
if(e.getSource()==b1)
{
int fact=fact(Integer.parseInt(t1.getText()));
t2.setText(String.valueOf(fact));
}
}
```

```
int fact(int f)
{
int s=0; if(f==0)
return 1; else
return f*fact(f-1);

}
}
/*<applet code="Fact.class" height=300 width=300></applet>*/
```

VIVA QUESTIONS:

1. **What is an applet? How does applet differ from applications?**

What is an applet? How does applet differ from applications? - A program that a Java enabled browser can download and run is an Applet.

2. What are the attributes of Applet tags? Explain the purposes?

What are the attributes of Applet tags? - height: Defines height of applet, width: Defines width of applet.

3. How will you initialize an Applet?

Write my initialization code in the applets init method or applet constructor.

4. What is the sequence for calling the methods by AWT for applets?

When an applet begins, the AWT calls the following methods, in this sequence:

- ▶ init()
- ▶ start()
- ▶ paint()

When an applet is terminated, the following sequence of method calls takes place :

- ▶ stop()
- ▶ destroy()

5. What is AppletStub Interface?

The applet stub interface provides the means by which an applet and the browser communicate. Your code will not typically implement this interface.

6. What is the base class for all swing components?

JComponent (except top-level containers)

7. How will you communicate between two Applets?

All the applets on a given page share the same AppletContext. We obtain this applet context as follows:

```
AppletContext ac = getAppletContext();
```

AppletContext provides applets with methods such as getApplet(name), getApplets(), getAudioClip(url), getImage(url), showDocument(url) and showStatus(status).

8. What is exception propagation ?

Forwarding the exception object to the invoking method is known as exception propagation.

9. What is the meaning of immutable in terms of String?

The simple meaning of immutable is unmodifiable or unchangeable. Once string object has been created, its value can't be changed.

PROGRAM -4

Creates a User Interface to perform Integer Divisions

Aim: Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw Number Format Exception. If Num2 were Zero, the program would throw an Arithmetic Exception Display the exception in a message dialog box.

THEORY: The AWT supports a rich assortment of graphics methods. All graphics are drawn relative to a window. This can be the main window of an applet, a child window of an applet, or a stand alone application window. The origin of each window is at the top-left corner and is 0,0 coordinates are specified in pixels. All output to a window takes place through a graphics context.

Program:

```
import java.awt.*; import
java.awt.event.*; import
java.applet.*;
/*<applet code="Div"width=230 height=250></applet>*/ public class
Div extends Applet implements ActionListener
{
String msg;
TextField num1,num2,res;Label l1,l2,l3; Button div;
public void init()
{
l1=new Label("Number 1"); l2=new
Label("Number 2"); l3=new
Label("result"); num1=new
TextField(10); num2=new TextField(10);
res=new TextField(10); div=new
Button("DIV");
div.addActionListener(this); add(l1);
add(num1);
add(l2);
add(num2);
add(l3);
add(res);
add(div);
}

public void actionPerformed(ActionEvent ae)
{
String arg=ae.getActionCommand();
if(arg.equals("DIV"))
```

```
{
String s1=num1.getText();
String s2=num2.getText();
int num1=Integer.parseInt(s1); int
num2=Integer.parseInt(s2); if(num2==0)
{
try
{
System.out.println(" ");
}
catch(Exception e)
{
System.out.println("ArithmeticException"+e);
}
msg="Arithmetic";
repaint();
}
else if((num1<0)||(num2<0))
{
try
{
System.out.println("");
}
catch(Exception e)
{
System.out.println("NumberFormatException"+e);
}
msg="NumberFormatException";
repaint();
}
else
{
int num3=num1/num2;
res.setText(String.valueOf(num3));
}
}
}
```

```
public void paint(Graphics g)
{
g.drawString(msg,30,70);
}
```

```
}
```

VIVA QUESTIONS:

1. What is Exception Handling?

Exception Handling is a mechanism to handle runtime errors. It is mainly used to handle checked exceptions.

2. What is difference between Checked Exception and Unchecked Exception?

i). Checked Exception

The classes that extend Throwable class except RuntimeException and Error are known as checked exceptions e.g. IOException, SQLException etc. Checked exceptions are checked at compile-time.

ii). Unchecked Exception

The classes that extend RuntimeException are known as unchecked exceptions e.g. ArithmeticException, NullPointerException etc. Unchecked exceptions are not checked at compile-time.

3. What is the base class for Error and Exception?

Throwable.

4. What is finally block?

finally block is a block that is always executed

5. Can finally block be used without catch?

Yes, by try block. finally must be followed by either try or catch.

6. Is there any case when finally will not be executed?

finally block will not be executed if program exits (either by calling System.exit() or by causing a fatal error that causes the process to abort)

7. What is exception propagation ?

Forwarding the exception object to the invoking method is known as exception propagation.

8. What is nested class?

A class which is declared inside another class is known as nested class. There are 4 types of nested class member inner class, local inner class, anonymous inner class and static nested class.

9.What is nested interface ?

Any interface i.e. declared inside the interface or class, is known as nested interface. It is static by default.

10.Can an Interface have a class?

Yes, they are static implicitly.

PROGRAM -5: Multithreaded Program

Aim: Write a Java program that implements a multithreaded program has three threads. First thread generates a random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd the third thread will print the value of cube of the number.

THEORY: The java run-time system depends on the threads for many things, and all the class libraries are designed with multithreading in mind. In fact ,java uses threads to enable the entire environment to be asynchronous. This helps reduce inefficiency by preventing the waste of CPU cycles. The benefits of java's multithreading is that the main loop/polling mechanism is eliminated. one thread can pause without stopping other parts of your program. when a thread blocks in a java program, only the single thread that is blocked pauses. All other threads continue to run.

Program:

```
import java.io.*; import
java.util.*;

class First extends Thread
{
public void run()
{
for(;;)
{
int r;
Random d = new Random(); roll =
d.nextInt(200) + 1; System.out.println(r);
Thread t2=new Second(r); Thread
t3=new Third(r); try
{
Thread.sleep(1000);
if(roll%2==0)
t2.start(); else t3.start();
}

catch(InterruptedException e){ }
}
}
}

class Second extends Thread
```

```
{
int r1; Second(int r)
{

r1=r;
}
public void run()

{
System.out.println("The square of number"+r1+"is:"+r1*r1);
}
}

class Third extends Thread
{
int r1; Third(int r)
{
r1=r;
}
public void run()

{
System.out.println("The Cube of the Number"+r1+"is: "+r1*r1*r1);
}
}

class Mthread
{
public static void main(String[] args)
{
Thread t1=new First(); System.out.println("press Ctrl+c to
stop....."); t1.start();
}
}
```

VIVA QUESTIONS**1) What is multithreading?**

Multithreading is a process of executing multiple threads simultaneously. Its main advantage is:

- Threads share the same address space.
- Thread is lightweight.
- Cost of communication between process is low.

2) What is thread?

A thread is a lightweight subprocess. It is a separate path of execution. It is called separate path of execution because each thread runs in a separate stack frame.

3) What is the difference between preemptive scheduling and time slicing?

Under preemptive scheduling, the highest priority task executes until it enters the waiting or dead states or a higher priority task comes into existence. Under time slicing, a task executes for a predefined slice of time and then reenters the pool of ready tasks. The scheduler then determines which task should execute next, based on priority and other factors.

4) What does join() method?

The join() method waits for a thread to die. In other words, it causes the currently running threads to stop executing until the thread it joins with completes its task.

5) Is it possible to start a thread twice?

No, there is no possibility to start a thread twice. If we do, it throws an exception.

6) Can we call the run() method instead of start()?

Yes, but it will not work as a thread rather it will work as a normal object so there will not be context-switching between the threads.

7) What about the daemon threads?

The daemon threads are basically the low priority threads that provide the background support to the user threads. It provides services to the user threads.

8) Can we make the user thread as a daemon thread if the thread is started?

No, if you do so, it will throw `IllegalThreadStateException`

9) What is shutdown hook?

The shutdown hook is basically a thread i.e. invoked implicitly before JVM shuts down. So we can use it to perform clean up resources.

10) When should we interrupt a thread?

We should interrupt a thread if we want to break out the sleep or wait state of a thread.

PROGRAM -6 :

Aim: Write a Java program for the following:

- i) Create a doubly linked list of elements.
- ii) Delete a given element from the above list.
- iii) Display the contents of the list after deletion

THEORY: A doubly-linked list is a linked data structure that consists of a set of sequentially linked records called nodes. Each node contains two fields, called links, that are references to the previous and to the next node in the sequence of nodes. The beginning and ending nodes previous and next links, respectively, point to some kind of terminator, typically a sentinel node or null, to facilitate traversal of the list. If there is only one sentinel node, then the list is circularly linked via the sentinel node. It can be conceptualized as two singly linked lists formed from the same data items, but in opposite sequential orders.

- i) Create a doubly linked list of elements

Program:

```
import java.util.NoSuchElementException;

public class DoublyLinkedListImpl<E> {

    private Node head;
    private Node tail;
    private int size;

    public DoublyLinkedListImpl() {
        size = 0;
    }
    /**
     * this class keeps track of each element information
     * @author java2novice
     */
    private class Node {
        E element;
        Node next;
        Node prev;

        public Node(E element, Node next, Node prev) {
            this.element = element;
            this.next = next;
            this.prev = prev;
        }
    }
    /**
     * returns the size of the linked list
     * @return
     */
}
```

```
public int size() { return size; }

/**
 * return whether the list is empty or not
 * @return
 */
public boolean isEmpty() { return size == 0; }

/**
 * adds element at the starting of the linked list
 * @param element
 */
public void addFirst(E element) {
    Node tmp = new Node(element, head, null);
    if(head != null ) {head.prev = tmp;}
    head = tmp;
    if(tail == null) { tail = tmp;}
    size++;
    System.out.println("adding: "+element);
}

/**
 * adds element at the end of the linked list
 * @param element
 */
public void addLast(E element) {

    Node tmp = new Node(element, null, tail);
    if(tail != null) { tail.next = tmp;}
    tail = tmp;
    if(head == null) { head = tmp;}
    size++;
    System.out.println("adding: "+element);
}

/**
 * this method walks forward through the linked list
 */
public void iterateForward(){

    System.out.println("iterating forward..");
    Node tmp = head;
    while(tmp != null){
        System.out.println(tmp.element);
        tmp = tmp.next;
    }
}

/**
 * this method walks backward through the linked list
 */
public void iterateBackward(){
```

```
System.out.println("iterating backward..");
Node tmp = tail;
while(tmp != null){
    System.out.println(tmp.element);
    tmp = tmp.prev;
}
}

/**
 * this method removes element from the start of the linked list
 * @return
 */
public E removeFirst() {
    if (size == 0) throw new NoSuchElementException();
    Node tmp = head;
    head = head.next;
    head.prev = null;
    size--;
    System.out.println("deleted: "+tmp.element);
    return tmp.element;
}

/**
 * this method removes element from the end of the linked list
 * @return
 */
public E removeLast() {
    if (size == 0) throw new NoSuchElementException();
    Node tmp = tail;
    tail = tail.prev;
    tail.next = null;
    size--;
    System.out.println("deleted: "+tmp.element);
    return tmp.element;
}

public static void main(String a[]){

    DoublyLinkedListImpl<Integer> dll = new DoublyLinkedListImpl<Integer>();
    dll.addFirst(10);
    dll.addFirst(34);
    dll.addLast(56);
    dll.addLast(364);
    dll.iterateForward();
    dll.removeFirst();
    dll.removeLast();
    dll.iterateBackward();
}
}
```

ii) Delete a given element from the above list.

Program:

```
import java.util.Scanner;
public class Delete
{
    public static void main(String[] args)
    {
        int n, x, flag = 1, loc = 0;
        Scanner s = new Scanner(System.in);
        System.out.print("Enter no. of elements you want in array:");
        n = s.nextInt();
        int a[] = new int[n];
        System.out.println("Enter all the elements:");
        for (int i = 0; i < n; i++)
        {
            a[i] = s.nextInt();
        }
        System.out.print("Enter the element you want to delete:");
        x = s.nextInt();
        for (int i = 0; i < n; i++)
        {
            if(a[i] == x)
            {
                flag = 1;
                loc = i;
                break;
            }
            else
            {
                flag = 0;
            }
        }
        if(flag == 1)
```

```
{
    for(int i = loc+1; i < n; i++)
    {
        a[i-1] = a[i];
    }
    System.out.print("After Deleting:");
    for (int i = 0; i < n-2; i++)
    {
        System.out.print(a[i]+",");
    }
    System.out.print(a[n-2]);
}
else
{
    System.out.println("Element not found");
}
}
```

iii) Display the contents of the list after deletion.

Program:

```
import java.util.Scanner;

public class JavaProgram
{
    public static void main(String args[])
    {
        int size, i, del, count=0;
        int arr[] = new int[50];
        Scanner scan = new Scanner(System.in);

        System.out.print("Enter Array Size : ");
        size = scan.nextInt();

        System.out.print("Enter Array Elements : ");
        for(i=0; i<size; i++)
        {
            arr[i] = scan.nextInt();
        }

        System.out.print("Enter Element to be Delete : ");
        del = scan.nextInt();
        for(i=0; i<size; i++)
        {
            if(arr[i] == del)
            {
                for(int j=i; j<(size-1); j++)
                {
                    arr[j] = arr[j+1];
                }
                count++;
                break;
            }
        }
        if(count==0)
        {
            System.out.print("Element Not Found..!!");
        }
        else
        {
            System.out.print("Element Deleted Successfully..!!");
            System.out.print("\nNow the New Array is :\n");
        }
    }
}
```

```
for(i=0; i<(size-1); i++)  
{  
    System.out.print(arr[i]+ " ");  
}  
}  
}
```

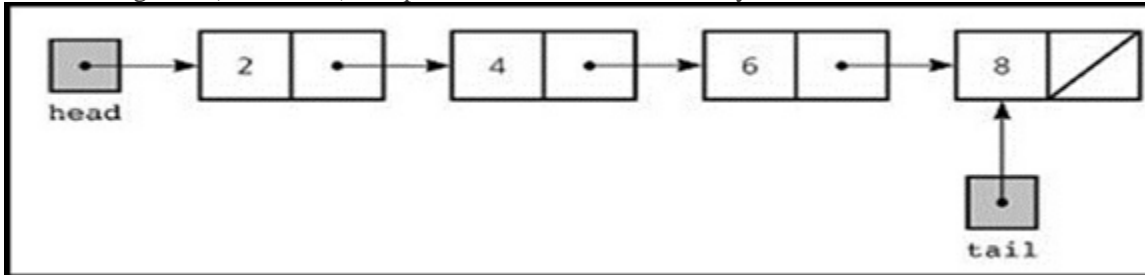
VIVA QUESTIONS:

1. What is a Linked list?

Linked list is an ordered set of data elements, each containing a link to its successor (and typically its predecessor).

2. Can you represent a Linked list graphically?

The fundamental data structure for the linked record involves 3 segments: the data itself and also the link to another element. Together (data + link) this particular structure is normally called the Node.



3. How many pointers are required to implement a simple Linked list?

You can find generally 3 pointers engaged:

- A head pointer, pointing to the start of the record.
- A tail pointer, pointing on the last node of the list. The key property in the last node is that its subsequent pointer points to nothing at all (NULL).
- A pointer in every node, pointing to the next node element.

4. How many types of Linked lists are there?

Singly linked list, [doubly linked list](#), multiply linked list, Circular Linked list.

5. How to delete a node from linked list?

- The following are the steps to delete node from the list at the specified position.
- Set the head to point to the node that head is pointing to.
- Traverse to the desired position or till the list ends; whichever comes first
 - You have to point the previous node to the next node.

6. How to [reverse a singly linked list](#)?

- First, set a pointer (*current) to point to the first node i.e. current=head.
- Move ahead until current!=null (till the end)
- set another pointer (*next) to point to the next node i.e. next=current->next
- store reference of *next in a temporary variable (*result) i.e. current->next=result
- swap the result value with current i.e. result=current
- And now swap the current value with next. i.e. current=next
- return result and repeat from step 2
- A linked list can also be reversed using recursion which eliminates the use of a temporary variable.

7. Compare Linked lists and Dynamic Arrays

A dynamic array is a data structure that allocates all elements contiguously in memory, and keeps a count of the present number of elements. If the area reserved for the dynamic array is exceeded, it's reallocated and traced, a costly operation.

Linked lists have many benefits over dynamic arrays. Insertion or deletion of an element at a specific point of a list, is a constant-time operation, whereas insertion in a dynamic array at random locations would require moving half the elements on the average, and all the elements in the worst case.

Whereas one can delete an element from an array in constant time by somehow marking its slot as vacant, this causes fragmentation that impedes the performance of iteration.

8. What is a Circular Linked list?

In the last node of a singly linear list, the link field often contains a null reference. A less common convention is to make the last node to point to the first node of the list; in this case the list is said to be 'circular' or 'circularly linked'.

PROGRAM -7 :**Traffic Light Simulation**

Aim: Write a java program to simulate a traffic light. The program lets the user select one of the three lights: red, yellow or green. On selecting a button, an appropriate message **with "Stop" or "Ready" or "Go" should appear above the buttons selected color.**

THEORY: The AWT supports a rich assortment of graphics methods. All graphics are drawn relative to a window. This can be the main window of an applet, a child window of an applet, or a stand alone application window. The origin of each window is at the top-left corner and its 0,0 coordinates are specified in pixels. All output to a window takes place through a graphics context.

Program:

```
import java.awt.event.*; import
java.applet.*; import java.awt.*;
```

```
/*
```

```
<applet code="TrafficLight" width=250 height=200></applet> */
```

```
public class TrafficLight extends Applet implements ItemListener { String msg
= "";
```

```
Checkbox red,green,yellow;
CheckboxGroup cbg;
```

```
public void init() {
cbg = new CheckboxGroup();
red = new Checkbox("Red", cbg, false); green = new
Checkbox("Green", cbg, false);
yellow = new Checkbox("Yellow", cbg, false); add(red);
add(yellow);
add(green);
red.addItemListener(this);
yellow.addItemListener(this);
green.addItemListener(this);
```

```
}
public void itemStateChanged(ItemEvent ie) { repaint();
}
```

```
// Display current state of the check boxes. public void
paint(Graphics g) {
Color color;
```

```
color=Color.BLACK;
g.setColor(color); g.drawOval(50, 50, 52,
```

```
52); g.drawOval(50, 103, 52, 52);

g.drawOval(50, 156, 52, 52);

String col = cbg.getSelectedCheckbox().getLabel();
System.out.println(col);

if(col.equalsIgnoreCase("Green"))
{
color= Color.GREEN; g.setColor(color);
g.fillOval(50, 156, 52, 52);
g.drawString("GO",110,190);
}if(col.equalsIgnoreCase("Red"))
{
color=Color.RED;
g.setColor(color); g.fillOval(51, 51, 51, 51);
g.drawString("STOP",110,80);
}
if(col.equalsIgnoreCase("Yellow"))
{
color=Color.YELLOW;
g.setColor(color);

g.fillOval(50, 103, 51, 51);
color= Color.BLACK; g.setColor(color);
g.drawString("READY",110,140);
}
}
}
```

VIVA QUESTIONS:

1.What is GUI?

GUI stands for Graphical User Interface.

- GUI allows users to click, drag, select graphical objects such as icons, images, buttons etc.
- GUI suppresses entering text using a command line.
- Examples of GUI operating systems are Windows, Mac, Linux.
- GUI is user friendly and increases the speed of work by the end users.
- A novice can understand the functionalities of certain application through GUI.

2) What is the difference between HashSet and TreeSet?

HashSet maintains **no order** whereas TreeSet maintains **ascending order**.

3) What is the difference between Set and Map?

Set contains values only whereas Map contains key and values both.

4) What is the difference between HashSet and HashMap?

HashSet contains only values whereas HashMap contains entry(key,value). HashSet can be iterated but HashMap need to convert into Set to be iterated.

5) What is the difference between HashMap and TreeMap?

HashMap maintains **no order** but TreeMap maintains **ascending order**.

6) What is the difference between Collection and Collections?

Collection is an interface whereas Collections is a class. Collection interface provides normal functionality of data structure to List, Set and Queue. But, Collections class is to sort and synchronize collection elements.

7) What is the advantage of generic collection?

If we use generic class, we don't need typecasting. It is typesafe and checked at compile time.

8) What is hash-collision in Hashtable and how it is handled in Java?

Two different keys with the same hash value is known as hash-collision. Two different entries will be kept in a single hash bucket to avoid the collision.

9) What is the Dictionary class?

The Dictionary class provides the capability to store key-value pairs.

10) What is the default size of load factor in hashing based collection?

The default size of load factor is **0.75**. The default capacity is computed as initial capacity * load factor. For example, $16 * 0.75 = 12$. So, 12 is the default capacity of Map.

PROGRAM -8 :**Abstract Class**

Aim: Write a java program to create an abstract class named shape that contains two integers and an empty method named printArea() Provide three classes named Rectangle,, Triangle and Circle such that each one of the classes extends the class shape. Each one of the class contains only the method printArea() that print the area of the given shape.

THEORY: To create an abstract class that shows the hiding of elements in a class. At the same time inheritance property is used to extend the class shape into different geometrical figures. This represents the reusability of code for a programmer.

Program:

```
// Abstract class that contains abstract method.
```

```
abstract class Shape
{
int h=10,w=5;
abstract void printArea();
}
```

```
// Classes that illustrates the abstract method.
```

```
class Rectangle extends Shape
{
void printArea()
{
float area=0.5*h*w;
System.out.println("The Area of rectangle is"+area);
}
}
```

```
class Triangle extends Shape
{
void printArea()
{
float area=h*w;
System.out.println("The Area of Triangle is"+area);
}
}
```

```
class Circle extends Shape
{
void printArea()
{
System.out.println("Circle area is not possible ");
}
}

// Class that create objects and call the method.

class ShapeDemo
{
public static void main(String args[])
{
Rectangle obj1 = new Rectangle (); Triangle obj2 = new
Triangle (); Circle obj3 = new Circle ();
obj1. printArea (); obj2. printArea (); obj3.
printArea ();
}
}
```

VIVA QUESTIONS:

1. What is abstraction?

Abstraction is a process of hiding the implementation details and showing only functionality to the user.

Abstraction lets you focus on what the object does instead of how it does it.

2. What is the difference between abstraction and encapsulation?

Abstraction hides the implementation details whereas encapsulation wraps code and data into a single unit.

3. What is abstract class?

A class that is declared as abstract is known as abstract class. It needs to be extended and its method implemented. It cannot be instantiated.

4. Can there be any abstract method without abstract class?

No, if there is any abstract method in a class, that class must be abstract.

5. Can you use abstract and final both with a method?

No, because abstract method needs to be overridden whereas you can't override final method.

6. Is it possible to instantiate the abstract class?

No, abstract class can never be instantiated.

7. What is interface?

Interface is a blueprint of a class that have static constants and abstract methods. It can be used to achieve fully abstraction and multiple inheritance.

8. Can you declare an interface method static?

No, because methods of an interface is abstract by default, and static and abstract keywords can't be used together.

9. Can an Interface be final?

No, because its implementation is provided by another class.

10. What is marker interface?

An interface that have no data member and method is known as a marker interface. For example Serializable, Cloneable etc.

PROGRAM -9 :**Display Table using Grid Layout**

Aim: Suppose that table named Table.txt is stored in a text file. The first line in the file is the header, and the remaining lines correspond to rows in the table. The elements are separated by commas. Write a java program to display the table using in Grid Layout.

THEORY: To create an table and display it using JTable components

Program:

```
import java.applet.Applet; import
java.awt.*;
import java.io.*; import
java.util.*;
public class Grid extends Applet
{
public void init()
{
String name,num,s; try
{
int i=lineCount();
setLayout (new GridLayout(i,2)); File
fr=new File("Table.txt");
BufferedReader frader=new BufferedReader(new FileReader(fr));
while((s=frader.readLine())!=null)
{
String[] st=s.split(",");
name=st[0]; num=st[1];
add (new TextField(" " + name)); add (new
TextField(" " + num));
}
}
catch(Exception e)
{
System.out.println("error from file");
}

}

int lineCount() throws Exception
{
int count = 0;
File f = new File("Table.txt");
```



```
Scanner input = new Scanner(f); while
(input.hasNextLine()) {
String line = input.nextLine(); count++;
}
return count;
}
}
/*<applet code="Grid.java" width=300 height=300></applet> */
```

VIVA QUESTIONS:

1. polymorphism is a feature that allows _____
2. polymorphism is expressed by the phrases one interface _____ methods.
3. Method override is the basis _____.
4. Java implements _____ using dynamic method dispatch.
5. A super class reference variable can refer to a _____ object.
6. The type of object being referred to determines which version of an _____ method.
7. Override methods allows java to support _____ 63
8. Method override occurs only when the names and the type signature of the two methods are _____
9. When a method in the subclass has the same name and type as the method in the super class then the method in the subclass is said to _____ the method in the super class.
10. Difference between overloading and overriding is _____

PROGRAM -10 :

Mouse Events

Aim: Write a java program that handles all mouse events and shows the event name at the center of the window when mouse event is fired (Use Adapter classes).

THEORY: To handle mouse events you must implement the `MouseListener` and the `MouseMotionListener` interfaces. These two interfaces contain methods that receive and process the various types of mouse events.

Program:

```
import java.awt.*; import
java.awt.event.*; import java.applet.*;

/* <applet code="AdapterDemo1" width=300 height=100></applet>*/

public class AdapterDemo1 extends Applet
{
String msg="welcome";
int mouseX=30, mouseY=30; public void
init()
{
addMouseListener(new MyMouseAdapter1(this));
addMouseMotionListener(new MyMouseMotionAdapter1(this));
}
public void paint(Graphics g)
{
g.drawString(msg, mouseX, mouseY);
}
}
class MyMouseAdapter1 extends MouseAdapter
{
AdapterDemo1 adapterDemo;
public MyMouseAdapter1(AdapterDemo1 adapterDemo)
{
this.adapterDemo = adapterDemo;
}
// Handle mouse clicked.
public void mouseClicked(MouseEvent me)
{
adapterDemo.msg="Mouse
clicked."; adapterDemo.repaint();
}
}
```

```
public void mouseEntered(MouseEvent me)
{
adapterDemo.msg = "Mouse
entered.";adapterDemo.repaint();

}
}

class MyMouseMotionAdapter1 extends MouseMotionAdapter
{
AdapterDemo1 adapterDemo;
intmouseX,mouseY;
public MyMouseMotionAdapter1(AdapterDemo1 adapterDemo)
{
this.adapterDemo = adapterDemo;
}
// Handle mouse dragged.
public void mouseDragged(MouseEvent me) { // save
coordinates
mouseX = me.getX(); mouseY =
me.getY();
adapterDemo.msg = "Mouse Dragged"; adapterDemo.showStatus("Dragging mouse at "
+ mouseX + " " + mouseY); adapterDemo.repaint();
}
}
```

VIVA QUESTIONS

1. MOUSE_CLICKED events occurs when_____
2. MOUSE_DRAGGED events occurs when _____
3. _____ events occur when mouse enters a component.
4. _____ events occur when the mouse exists a component
5. MOUSE_MOVED event occurs when _____
6. MOUSE_PRESSED even occurs when _____
7. The _____ events occur when mouse was released.
8. The _____ event occurs when mouse wheel is moved.
9. An event source is _____
10. A _____ is an object that describes the state change in a source

PROGRAM -11 :**Using File and Hash Table**

Aim: Write a java program that loads names and phone numbers from a text file where the data is organized as one line per record and each field in a record are separated by tab(\t). It takes a name or phone number as input and prints the corresponding other value from the hash table (use hash tables).

THEORY: Text file will contain names and phone numbers which are separated by a tab. This information has to be recorded in to hash table.

Program:

```
import java.lang.*; import
java.util.*; import java.io.*;

public class HashTablepro1
{
public static void main(String ar[])
{
String s,str,name,num; Hashtable ht = new
Hashtable();

ht.put("Sita","9999999991");

ht.put("Rama","9999999992");

ht.put("Rani","9999999993" );

ht.put("Raju","9999999994");
ht.put("Usha","9999999997"); try
{
File fr=new File("file.txt");
BufferedReader freader=new BufferedReader(new FileReader(fr));

while((s=freader.readLine())!=null)
{
String[] st=s.split(" ");
name=st[0];num=st[1];
searchHash(ht,name,num);
}
}
catch(Exception e)
```

```
{
System.out.println("error from file");

}
}

static void searchHash(Hashtable ht,String name,String num)
{
String str;
Enumeration names = ht.keys();

int f=0;
System.out.println("The searched Name is: "+name);
while(names.hasMoreElements())
{
//next element retrieves the next element in the dictionary

str = (String) names.nextElement();

if(name.equals(str))
{
//.get(key) returns the value of the key stored in the hashtable

System.out.println(str + ": " + ht.get(str) + "\n");
}

}

}
}
```

VIVA QUESTIONS

1. _____ is a package in which Hashtable class is available
2. split is a method used for _____
3. capacity of hashtable can be determined by _____
4. _____ method is used to insert record in to hash table
5. _____ method is used to know the number of entries in hash table
6. alternative to hashtable is _____
7. _____ is used to remove all entries of hash table
8. Hash table can be enumerated using _____
9. Scanner class is present in _____ package
10. Buffered Reader is available in _____ package

PROGRAM -12 :**Producer-Consumer Problem Using Inter-thread Communication**

Aim: Write a Java program that correctly implements the producer – consumer problem using the concept of interthread communication.

THEORY: **Inter-thread communication** or **Co-operation** is all about allowing synchronized threads to communicate with each other.

Program:

```
class Q
{
    int n;
    boolean valueSet=false;
    synchronized int get()
    {
        if(!valueSet)
        try
        {
            wait();
        }
        catch(InterruptedException e)
        {
            System.out.println("Interrupted Exception caught");
        }
        System.out.println("Got:"+n);
        valueSet=false;
        notify();
        return n;
    }
    synchronized void put(int n)
    {
        if(valueSet)
        try
        {
            wait();
        }
        catch(InterruptedException e)
        {
```

```
        System.out.println("Interrupted Exception caught");
    }
    this.n=n;
    valueSet=true;
    System.out.println("Put:"+n);
    notify();
}
}
class Producer implements Runnable
{
    Q q;
    Producer(Q q)
    {
        this.q=q;
        new Thread(this,"Producer").start();
    }
    public void run()
    {
        int i=0;
        while(true)
        {
            q.put(i++);
        }
    }
}
class Consumer implements Runnable
{
    Q q;
    Consumer(Q q)
    {
        this.q=q;
        new Thread(this,"Consumer").start();
    }
    public void run()
    {
        while(true)
        {
            q.get();
        }
    }
}
class ProdCons
{
    public static void main(String[] args)
```

```
    {
        Q q=new Q();
        new Producer(q);
        new Consumer(q);
        System.out.println("Press Control-c to stop");
    }
}
```

VIVA QUESTIONS

1) What is Thread in Java?

The thread is an independent path of execution. It's way to take advantage of multiple CPU available in a machine. By employing multiple threads you can speed up CPU bound task. For example, if one thread takes 100 milliseconds to do a job, you can use 10 thread to reduce that task into 10 milliseconds. Java provides excellent support for multithreading at the language level, and it's also one of the strong selling points.

2) What is the difference between Thread and Process in Java?

The thread is a subset of Process, in other words, one process can contain multiple threads. Two process runs on different memory space, but all threads share same memory space. Don't confuse this with stack memory, which is different for the different thread and used to store local data to that thread. For more detail see the answer.

3) How do you implement Thread in Java?

At the language level, there are two ways to implement Thread in Java. An instance of `java.lang.Thread` represent a thread but it needs a task to execute, which is an instance of interface `java.lang.Runnable`. Since `Thread` class itself implement `Runnable`, you can override `run()` method either by extending `Thread` class or just implementing `Runnable` interface. For detailed answer and discussion see this article.

4) When to use Runnable vs Thread in Java?

2) This is a follow-up of previous multi-threading interview question. As we know we can implement thread either by extending `Thread` class or implementing `Runnable` interface, the question arise, which one is better and when to use one? This question will be easy to answer if you know that Java programming language doesn't support multiple inheritances of class, but it allows you to implement multiple interfaces. Which means, it's better to implement `Runnable` then extends `Thread` if you also want to extend another class e.g. `Canvas` or `CommandListener`. For more points and discussion you can also refer this post.

6) What is the difference between start() and run() method of Thread class?

One of trick Java question from early days, but still good enough to differentiate between shallow understanding of Java threading model `start()` method is used to start newly created thread, while `start()` internally calls `run()` method, there is difference calling `run()` method directly. When you

invoke `run()` as normal method, its called in the same thread, no new thread is started, which is the case when you call `start()` method. Read this answer for much more detailed discussion.

PROGRAM -13 :

Using File and Directories

Aim: . Write a Java program to list all the files in a directory including the files present in all its subdirectories.

THEORY: The listFiles(File directory, IOFileFilter fileFilter, IOFileFilter dirFilter) method of the FileUtils class of the Apache Commons IO library returns a Collection of files in a specified directory passed in as its first parameter. If the third parameter (dirFilter) is null, only the files in the specified directory are returned. If TrueFileFilter.INSTANCE is passed in, all of the files within the specified directory are returned, including all subdirectories.

Program:

```
package test;

import java.io.File;
import java.io.IOException;
import java.util.List;

import org.apache.commons.io.FileUtils;
import org.apache.commons.io.filefilter.TrueFileFilter;

public class GetAllFilesInDirectory {

    public static void main(String[] args) throws IOException {

        File dir = new File("dir");

        System.out.println("Getting all files in " + dir.getCanonicalPath() + " including those in
subdirectories");

        List<File> files = (List<File>) FileUtils.listFiles(dir, TrueFileFilter.INSTANCE,
TrueFileFilter.INSTANCE);

        for (File file : files) {
            System.out.println("file: " + file.getCanonicalPath());
        }

    }

}
```

VIVA QUESTIONS

1. **What is a IO stream?**

It is a stream of data that flows from source to destination. Good example is file copying. Two streams are involved – input stream and output stream. An input stream reads from the file and stores the data in the process (generally in a temporary variable). The output stream reads from the process and writes to the destination file.

2. **What is the necessity of two types of streams – byte streams and character streams?**

Byte streams were introduced with JDK 1.0 and operate on the files containing ASCII characters. We know Java supports other language characters also known as Unicode characters. To read the files containing Unicode characters, the designers introduced character streams with JDK 1.1. As ASCII is a subset of Unicode, for the files of English characters, we can go with either byte streams or character streams.

3. **What are the super most classes of all streams?**

All the byte stream classes can be divided into two categories (input stream classes and output stream classes) and all character streams classes into two (reader classes and writer classes). There are four abstract classes from which all these streams are derived. The super most class of all byte stream classes is **java.io.InputStream** and for all output stream classes, **java.io.OutputStream**. Similarly for all reader classes is **java.io.Reader** and for all writer classes is **java.io.Writer**.

4. **What are FileInputStream and FileOutputStream?**

These two are general purpose classes used by the programmer very often to copy file to file. These classes work well with files containing less data of a few thousand bytes as by performance these are very poor. For larger data, it is preferred to use BufferedInputStream (or BufferedReader) and BufferedOutputStream (or BufferedWriter).

5. **Which you feel better to use – byte streams or character streams?**

I feel personally to go with character streams as they are the latest. Many features exist in character streams that do not in byte streams like a) using BufferedReader in place of BufferedInputStream and DataInputStream (one stream for two) and b) using newLine() method to go for next line and for this effect we must go for extra coding in byte streams etc.

6. **What System.out.println()?**

"println()" is a method of PrintStream class. "out" is a static object of PrintStream class defined in "System" class. System is a class from **java.lang** package used to interact with the underlying operating system by the programmer.

7. **What are filter streams?**

Filter streams are a category of IO streams whose responsibility is to add extra functionality (advantage) to the existing streams like giving line numbers in the destination file that do not exist in the source file or increasing performance of copying etc.

8. **Name the filter streams available?**

There are four filter streams in **java.io** package – two in byte streams side and two in character streams side. They are **FilterInputStream**, **FilterOutputStream**, **FilterReader** and **FilterWriter**. These classes are abstract classes and you cannot create objects of these classes.

9. **Name the filter stream classes on reading side of byte stream?**

There are four classes – LineNumberInputStream (the extra functionality is it adds line numbers in the destination file), DataInputStream (contains special methods like readInt(), readDouble() and readLine() etc that can read an int, a double and a string at a time), BufferedInputStream (gives buffering effect that increases the performance to the peak) and PushbackInputStream (pushes the required character back to the system).

10. **What is the functionality of SequenceInputStream?**

It is very useful to copy multiple source files into one destination file with very less code.

PROGRAM -14 :

Quick Sort Algorithm

Aim: Write a Java program that implements Quick sort algorithm for sorting a list of names in ascending order

THEORY: Sort algorithms order the elements of an array according to a predefined order. Quicksort is a divide and conquer algorithm. In a divide and conquer sorting algorithm the original data is separated into two parts "divide" which are individually sorted and "conquered" and then combined.

Program:

```
package com.java2novice.sorting;

public class MyQuickSort {

    private int array[];
    private int length;

    public void sort(int[] inputArr) {

        if (inputArr == null || inputArr.length == 0) {
            return;
        }
        this.array = inputArr;
        length = inputArr.length;
        quickSort(0, length - 1);
    }

    private void quickSort(int lowerIndex, int higherIndex) {

        int i = lowerIndex;
        int j = higherIndex;
        // calculate pivot number, I am taking pivot as middle index number
        int pivot = array[lowerIndex+(higherIndex-lowerIndex)/2];
        // Divide into two arrays
        while (i <= j) {
            /**
             * In each iteration, we will identify a number from left side which
             * is greater then the pivot value, and also we will identify a number
             * from right side which is less then the pivot value. Once the search
```

```
    * is done, then we exchange both numbers.
    */
    while (array[i] < pivot) {
        i++;
    }
    while (array[j] > pivot) {
        j--;
    }
    if (i <= j) {
        exchangeNumbers(i, j);
        //move index to next position on both sides
        i++;
        j--;
    }
}
// call quickSort() method recursively
if (lowerIndex < j)
    quickSort(lowerIndex, j);
if (i < higherIndex)
    quickSort(i, higherIndex);
}

private void exchangeNumbers(int i, int j) {
    int temp = array[i];
    array[i] = array[j];
    array[j] = temp;
}

public static void main(String a[]){

    MyQuickSort sorter = new MyQuickSort();
    int[] input = {24,2,45,20,56,75,2,56,99,53,12};
    sorter.sort(input);
    for(int i:input){
        System.out.print(i);
        System.out.print(" ");
    }
}
}
```

VIVA QUESTIONS

1) Explain what is an algorithm in computing?

An algorithm is a well-defined computational procedure that take some value as input and generate some value as output. In simple words, it's a sequence of computational steps that converts input into the output.

2) Explain what is Quick Sort algorithm?

Quick Sort algorithm has the ability to sort list or queries quickly. It is based on the principle of partition exchange sort or Divide and conquer. This type of algorithm occupies less space, and it segregates the list into three main parts

- Elements less than the Pivot element
- Pivot element
- Elements greater than the Pivot element

3) Explain what is time complexity of Algorithm?

Time complexity of an algorithm indicates the total time needed by the program to run to completion. It is usually expressed by using the **big O notation**.

4) Mention what are the types of Notation used for Time Complexity?

The types of Notations used for Time Complexity includes

- **Big Oh:** It indicates “fewer than or the same as” <expression>iterations
- **Big Omega:** It indicates “more than or same as” <expression>iterations
- **Big Theta:** It indicates “the same as” <expression>iterations
- **Little Oh:** It indicates “fewer than” <expression>iterations
- **Little Omega:** It indicates “more than” <expression>iterations

5) Explain how binary search works?

In binary search, we compare the key with the item in the middle position of the array. If the key is less than the item searched then it must lie in the lower half of the array, if the key is greater than the item searched than it should be in upper half of the array.

Additional programs

1. Write a java program on QuitButton

```
package com.zetcode;

import java.awt.Container;
import java.awt.EventQueue;
import java.awt.event.ActionEvent;
import javax.swing.GroupLayout;
import javax.swing.JButton;
import javax.swing.JComponent;
import javax.swing.JFrame;

public class QuitButtonEx extends JFrame {

    public QuitButtonEx() {

        initUI();
    }

    private void initUI() {

        JButton quitButton = new JButton("Quit");
```

```
quitButton.addActionListener((ActionEvent event) -> {
    System.exit(0);
});

createLayout(quitButton);

setTitle("Quit button");
setSize(300, 200);
setLocationRelativeTo(null);
setDefaultCloseOperation(EXIT_ON_CLOSE);
}

private void createLayout(JComponent... arg) {

    Container pane = getContentPane();
    GroupLayout gl = new GroupLayout(pane);
    pane.setLayout(gl);

    gl.setAutoCreateContainerGaps(true);

    gl.setHorizontalGroup(gl.createSequentialGroup()
        .addComponent(arg[0])
    );
```



```
gl.setVerticalGroup(gl.createSequentialGroup()  
    .addComponent(arg[0])  
);  
}
```

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

```
    EventQueue.invokeLater(() -> {
```

```
        QuitButtonEx ex = new QuitButtonEx();
```

```
        ex.setVisible(true);
```

```
    });
```

```
}
```

```
}
```

2. Write a java program on FrameIcon

```
package com.zetcode;
```

```
import java.awt.EventQueue;
```

```
import javax.swing.ImageIcon;
```

```
import javax.swing.JFrame;
```

```
import static javax.swing.JFrame.EXIT_ON_CLOSE;
```

```
public class FrameIconEx extends JFrame {
```

```
    public FrameIconEx() {
```

```
        initUI();
```

```
    }
```

```
    private void initUI() {
```

```
        ImageIcon webIcon = new ImageIcon("web.png");
```

```
        setIconImage(webIcon.getImage());
```

```
        setTitle("Icon");
```

```
setSize(300, 200);
setLocationRelativeTo(null);
setDefaultCloseOperation(EXIT_ON_CLOSE);
}

public static void main(String[] args) {

    EventQueue.invokeLater(() -> {
        JFrameIconEx ex = new JFrameIconEx();
        ex.setVisible(true);
    });
}
}
```

3. Fibonacci Series using recursion in java

Let's see the fibonacci series program in java using recursion.

```
1. class FibonacciExample2{
2.   static int n1=0,n2=1,n3=0;
3.   static void printFibonacci(int count){
4.     if(count>0){
5.       n3 = n1 + n2;
6.       n1 = n2;
7.       n2 = n3;
8.       System.out.print(" "+n3);
9.       printFibonacci(count-1);
10.    }
11. }
12. public static void main(String args[]){
13.   int count=10;
14.   System.out.print(n1+" "+n2);//printing 0 and 1
15.   printFibonacci(count-2);//n-2 because 2 numbers are already printed
16. }
17. }
```

4. How Exception Terminates Java Program?

Whenever exception arises, it terminates the program execution, means it stops the execution of the current java program. Below example shows what happens when an exception occurs.

Sample Code

Code:

[?](#)

```
1 package com.myjava.exceptions;
2
3 public class MyException {
4     public static void main(String a[]){
5         for(int i=5;i>=0;i--){
6             System.out.println(10/i);
7         }
8         System.out.println("After for loop...");
9     }
10 }
```

Java Exception Handling Examples

Exceptions can be handled by using '**try-catch**' block. Try block contains the code which is under observation for exceptions. The catch block contains the remedy for the exception. If any exception occurs in the try block then the control jumps to catch block.

Java Exception Handling Sample Code

Code:

[?](#)

```
1 package com.myjava.exceptions;
2
3 public class MyExceptionHandle {
4     public static void main(String a[]){
5         try{
6             for(int i=5;i>=0;i--){
7                 System.out.println(10/i);
8             }
9         } catch(Exception ex){
10            System.out.println("Exception Message: "+ex.getMessage());
11            ex.printStackTrace();
12        }
13        System.out.println("After for loop...");
```

```
14    }
```

```
15    }
```