**Experiment : 2**

**Aim:**  Develop any distributed application using CORBA to demonstrate object brokering.

(Calculator or String operations).

1. Create a new folder C:\Calc in C:\ directory.

2. Create Calc.idl file using Notepad. Save it with in C:\Calc folder with idl as extension. Paste the following into the idl file:

module WssCalculator

{

interface Calc

{

//Performs the Calculations:ADD/SUB/MUL/DIV

long calculate(in long operator,in long num1,in long num2);

//The Server EXITS when the Client prompts it to do so

oneway void shutdown();

};

};

3. This file will define language neutral definition for Remote Interface called Calc with specified Methods/Functions (Note: It can be Java or C++ or any OOP based object).

4. Next, compile the .idl file, open the command prompt and change the directory to C:\Calc. Now enter the following command:

idlj -fall Calc.idl

The -fall specifies create binding for both client as well as server.  
This will create the folder C:\Calc\WssCalculator and the following java files within it:  
\_CalcStub.java  
Calc.java  
CalcHelper.java  
CalcHolder.java  
CalcOperations.java  
CalcPOA.java

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

5. Now a java file named CalcServer is created in C:\Calc with the following code:

//Importing all the packages and classes

//Import the package which contains the Server Skeleton

import WssCalculator.\*;

//Import the below two packages to use the Naming Service

import org.omg.CosNaming.\*;

import org.omg.CosNaming.NamingContextPackage.\*;

//Import this package to run the CORBA Application

import org.omg.CORBA.\*;//

//Import the below to Classes for inheriting Portable Server

import org.omg.PortableServer.\*;

import org.omg.PortableServer.POA;

//Initiate the ORB using the class Properties

import java.util.Properties;

//Perform the Input-Output functionalities

import java.io.\*;

import java.util.\*;

//Write the Servant class

//It inherits the general CORBA utilities generated by the Compiler

class Calcserverimpl extends CalcPOA

{

//orb variable is used to invoke the shutdown()

private ORB orb;

public void setORB(ORB orb\_val)

{

orb = orb\_val;

}

//Declaring and Implementing the required method

public int calculate(int a,int b,int c)

{

//ADDITION

if(a==43)

{

return (b+c);

}

//SUBTRACTION

else if(a==45)

{

return (b-c);

}

//MULTIPLICATION

else if(a==42)

{

return (b\*c);

}

//DIVISION

else if(a==47)

{

return (b/c);

}

//DEFAULT

else

{

return 0;

}

}

//Closing the server

public void shutdown()

{

orb.shutdown(false);

}

}//end of the servant class

public class CalcServer

{

public static void main(String args[])

{

try

{

//Create and Initialize the ORB object

//init() allows to set the properties at run time

ORB orb=ORB.init(args,null);

//Obtain the initial Naming Context

//Obtain an initial object reference to the name server

//orb retrieves the reference to the Root POA

//Activate the POA Manager

//activate() causes the POAs to process the client requests

POA rootpoa=POAHelper.narrow(orb.resolve\_initial\_references("RootPOA"));

rootpoa.the\_POAManager().activate();

//The server instantiates the servant objects

//The servant performs the operations defined in the idlj interface

Calcserverimpl simpl=new Calcserverimpl();

simpl.setORB(orb);

//Get the object reference associated with the servant

//narrow() is used to cast CORBA obj ref to its proper type

org.omg.CORBA.Object ref = rootpoa.servant\_to\_reference(simpl);

Calc href=CalcHelper.narrow(ref);

//Obtain the initial Naming Context

//Obtain an object reference to the Name Server

org.omg.CORBA.Object objRef=orb.resolve\_initial\_references("NameService");

//Narrow the objref to its proper type

NamingContextExt ncRef=NamingContextExtHelper.narrow(objRef);

//Register the Servant with the Name Server

String name = "Calc";

//NameComponent array contains the path to Calc

NameComponent path[]=ncRef.to\_name(name);

//Pass the path and the servant object to the Naming Service

//Bind the servant object to Calc

ncRef.rebind(path,href);

System.out.println("The SERVER is READY");

System.out.println("The SERVER is WAITING to receive the C //run() enables the ORB to perform work using the main thread

//the server waits until an invocation comes from the ORB

orb.run();

}

catch (Exception e)

{

System.err.println("ERROR: " + e);

e.printStackTrace(System.out);

}

//This statement is executed when the Client wishes to discontinue

System.out.println("The Server Exits");

}//end of main()

}//end of CalcServer()

6. Create another file C:\Calc\CalcClient.java with the following code in it:LIENT requests");

/ //run() is called by the main thread

//Import all the important packages

//Import the package which contains the Client Stub

import WssCalculator.\*;

//Import the below two packages to use the Naming Service

import org.omg.CosNaming.\*;

import org.omg.CosNaming.NamingContextPackage.\*;

//Import this package to run the CORBA Applicaion

import org.omg.CORBA.\*;

//Import to perform Input-Output functionalities

import java.io.\*;

import java.util.\*;

public class CalcClient

{

static Calc cimpl;

public static void main(String args[])

{

try

{

//Declaring and initializing the variables

int dec=1;

int i=0;

int j=0;

int k=0;

int result=0;

int x=1;

char c='x';

char d='y'; ORB orb=ORB.init(args,null);

//ORB helps the Client to locate the actual services which it needs

//COS Naming Service helps the client to do so

//Obtain the initial Naming Context

//Obtain an object reference to the name server

org.omg.CORBA.Object objRef=orb.resolve\_initial\_references("NameService");

//Narrow the objref to its proper type

NamingContextExt ncRef=NamingContextExtHelper.narrow(objRef);

//Identify a String to refer the Naming Service to Calc object

String name="Calc";

//Get a reference to the CalcServer and Narrow it to Calc object

cimpl=CalcHelper.narrow(ncRef.resolve\_str(name));

System.out.println("Obtained a handle on the server object");

char f='z';

String abc="vas";

//Create and Initialize the ORB object

//init() allows to set properties at run time BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

while(x==1)

{

System.out.println("Enter the string:");

abc=br.readLine();

//Separate the input string into separate characters

c=abc.charAt(0);

d=abc.charAt(1);

f=abc.charAt(2);

//Get the ASCII value of the Operator

i=(int)c;

//Get the Integer values of the other two characters

j=Character.getNumericValue(d);

k=Character.getNumericValue(f);

result=cimpl.calculate(i,j,k);

System.out.println("The result of the operation is "+result);

System.out.println("Enter 1 to continue and 0 to exit ");

x=Integer.parseInt(br.readLine());

}

//If the Client wants to discontinue

cimpl.shutdown();

}

catch(Exception e)

{

System.out.println("ERROR : " + e) ;

e.printStackTrace(System.out);

}

}//end of main()

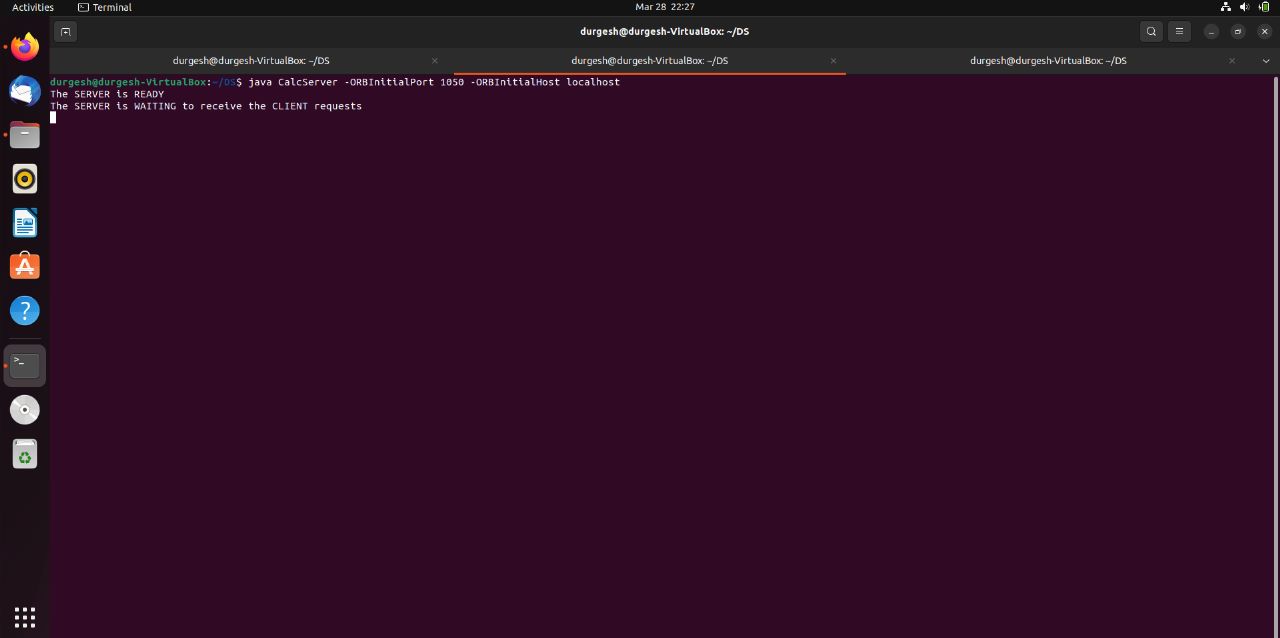
}//end of the class

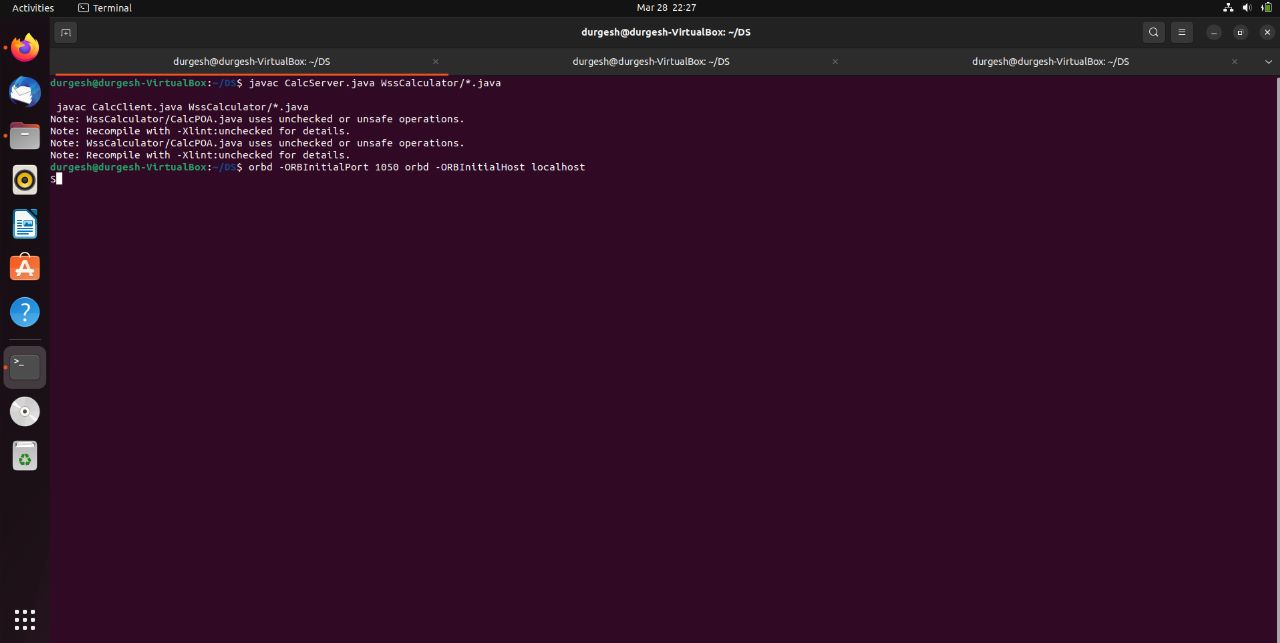
**OUTPUT::**

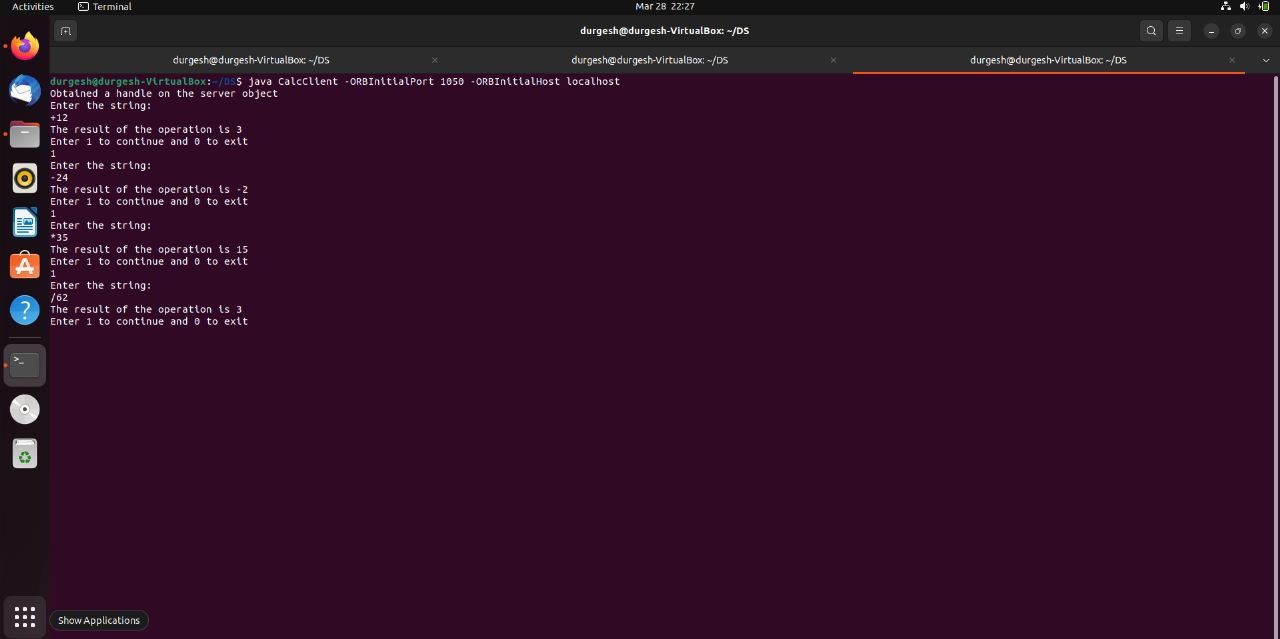
**preeti@ubuntu:~$ idlj -fall Calc.idl**

**preeti@ubuntu:~$ javac CalcServer.java WssCalculator/\*.java**

**preeti@ubuntu:~$ javac CalcClient.java WssCalculator/\*.java**

****

****

****