UNIVERSIDAD NEBRIJA

**ALUMNO: Marcos Pardo Zapico**

Asignatura: Programación de Sistemas Distribuidos

Curso: 2021/2022 Fecha: 15-02-2022
Semestre: 2º

**PRÁCTICA 1: Aplicación usando CORBA**

**Para poder realizar y ejecutar este programa necesitaremos un editor como Atom y el JDK de Java, que lo puedes descargar de http://www.java.com/es/**

**Para entregar la práctica hay que subir por un lado este doc en pdf y por otro lado 2 zip: HolaMundo.zip y Practica2.zip. En el comentario de la entrega de la práctica habrá que hacer la referencia del repositorio de Github.**

1. **Vamos a hacer un Hola Mundo en CORBA (2 punto)**

   **La aplicación contendrá un archivo IDL, un archivo servidor y uno de cliente. Todas las instrucciones de la aplicación deben estar comentadas en castellano, con nuestras palabras para argumentar que se entiende.**

   **Compilaremos primero el IDL, luego el servidor y luego el cliente usando los siguientes códigos: respectivamente:**

   **$ idlj -fall count.idl**
   **$ javac Server.java**
   **$ javac Client.java**

   **Para ejecutar el programa necesitamos tener abiertas tres ventanas del Símbolo del sistema. La primera iniciará el puerto, la segunda ejecutará el servidor y la tercera el cliente. El código para ejecutarla es, respectivamente:**

   **$ tnameserv -ORBInitialPort 2000**
   **$ java Server -ORBInitialHost localhost -ORBInitialPort 2000**
   **$ java Client -ORBInitialHost localhost -ORBInitialPort 2000**

**Hello.idl**

```
module HelloApp
{
  interface Hello
  {
    string sayHello();
    oneway void shutdown();
  };
};
```

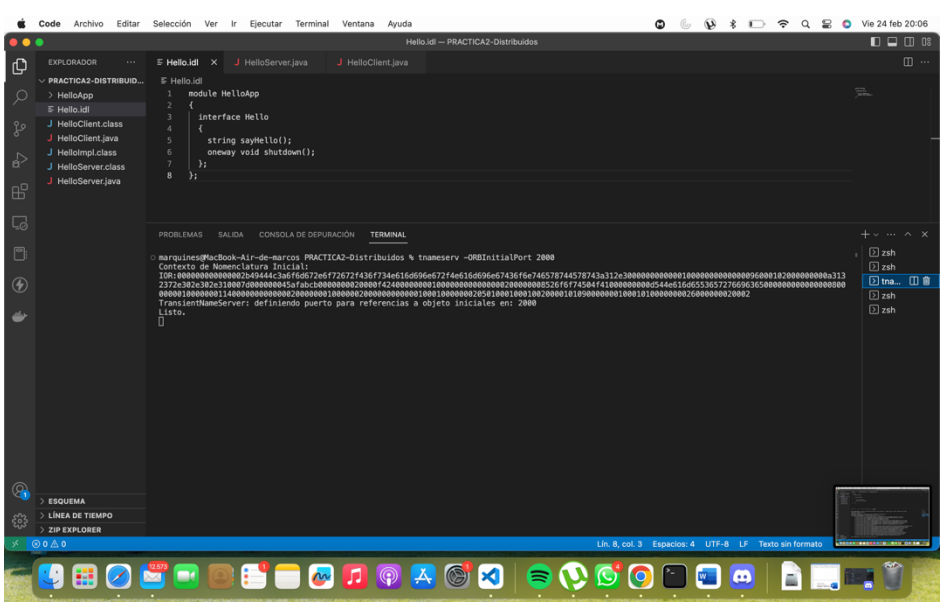
**HelloServer.java**

```
import HelloApp.*;
import org.omg.CosNaming.*;
import org.omg.CosNaming.NamingContextPackage.*;
import org.omg.CORBA.*;
import org.omg.PortableServer.*;
import org.omg.PortableServer.POA;

import java.util.Properties;

class HelloImpl extends HelloPOA {
  private ORB orb;

  public void setORB(ORB orb_val) {
    orb = orb_val;
  }

  // implement sayHello() method
  public String sayHello() {
    return "\nHello world !!\n";
  }

  // implement shutdown() method
  public void shutdown() {
    orb.shutdown(false);
  }
}


public class HelloServer {
```

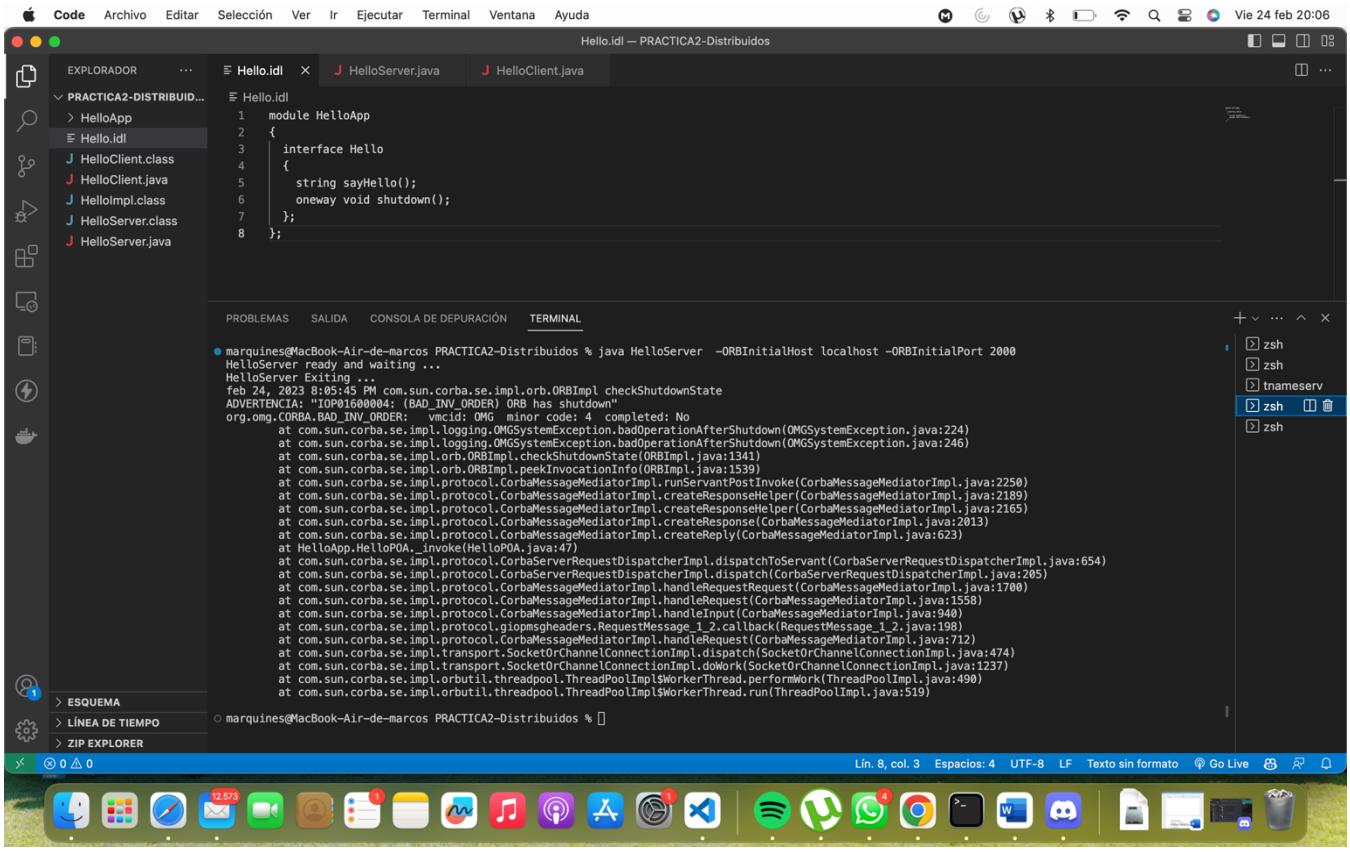
```
  public static void main(String args[]) {
    try{
      // create and initialize the ORB
      ORB orb = ORB.init(args, null);

      // get reference to rootpoa and activate the POAManager
      POA rootpoa = POAHelper.narrow(orb.resolve_initial_references("RootPOA"));
      rootpoa.the_POAManager().activate();

      // create servant and register it with the ORB
      HelloImpl helloImpl = new HelloImpl();
      helloImpl.setORB(orb);

      // get object reference from the servant
      org.omg.CORBA.Object ref = rootpoa.servant_to_reference(helloImpl);
      Hello href = HelloHelper.narrow(ref);

      // get the root naming context
      org.omg.CORBA.Object objRef =
          orb.resolve_initial_references("NameService");
      // Use NamingContextExt which is part of the Interoperable
      // Naming Service (INS) specification.
      NamingContextExt ncRef = NamingContextExtHelper.narrow(objRef);

      // bind the Object Reference in Naming
      String name = "Hello";
      NameComponent path[] = ncRef.to_name( name );
      ncRef.rebind(path, href);

      System.out.println("HelloServer ready and waiting ...");

      // wait for invocations from clients
      orb.run();
    }

      catch (Exception e) {
        System.err.println("ERROR: " + e);
        e.printStackTrace(System.out);
      }

      System.out.println("HelloServer Exiting ...");

  }
}
```

**HelloClient.java**

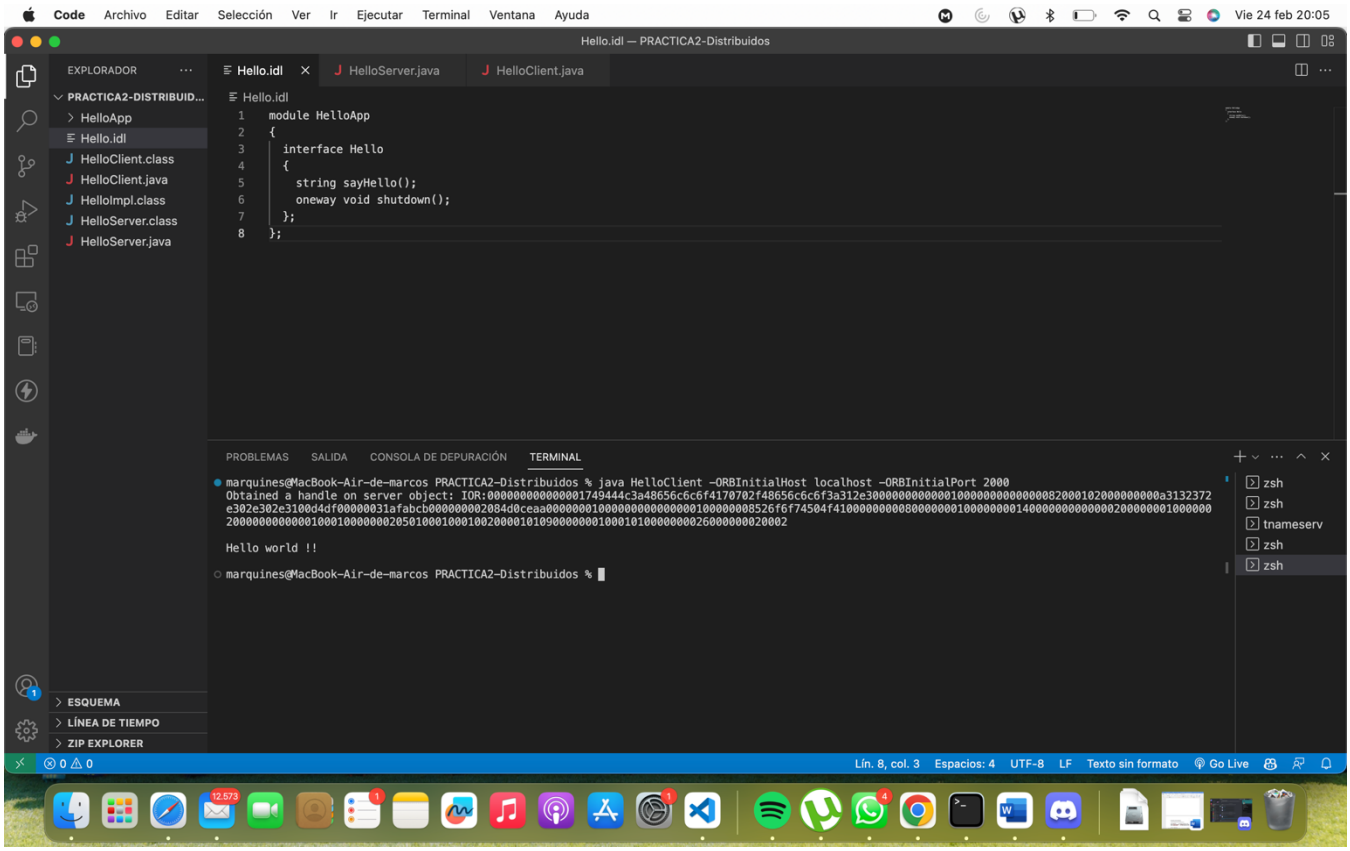
```
import HelloApp.*;
import org.omg.CosNaming.*;
import org.omg.CosNaming.NamingContextPackage.*;
import org.omg.CORBA.*;

public class HelloClient
{
  static Hello helloImpl;

  public static void main(String args[])
    {
      try{
        // create and initialize the ORB
        ORB orb = ORB.init(args, null);

        // get the root naming context
        org.omg.CORBA.Object objRef =
            orb.resolve_initial_references("NameService");
        // Use NamingContextExt instead of NamingContext. This is
        // part of the Interoperable naming Service.
        NamingContextExt ncRef = NamingContextExtHelper.narrow(objRef);

        // resolve the Object Reference in Naming
        String name = "Hello";
        helloImpl = HelloHelper.narrow(ncRef.resolve_str(name));

        System.out.println("Obtained a handle on server object: " + helloImpl);
        System.out.println(helloImpl.sayHello());
        helloImpl.shutdown();

        } catch (Exception e) {
          System.out.println("ERROR : " + e) ;
          e.printStackTrace(System.out);
          }
    }
}
```



UNIVERSIDAD NEBRIJA



UNIVERSIDAD
NEBRIJA



UNIVERSIDAD NEBRIJA

**2.Preguntas sobre Hola Mundo en CORBA (puedes añadir capturas) (2 puntos):**

**a. ¿Qué sucede si lanzo antes el cliente que el servidor?**

```
marquines@MacBook-Air-de-marcos PRACTICA2-Distribuidos % java HelloClient -ORBInitialHost localhost -ORBInitialPort 2000
Obtained a handle on server object: IOR:000000000000001749444c3a48656c6c6f4170702f48656c6c6f3a312e3000000000000100000000000000820001020000000000a3132372
e302e302e3100d4fc00000031afabcb000000002084d36f5d0000000100000000000000010000000526f6f74504f4100000000800000001000000014000000000000020000001000000
2000000000000100010000000205010001000100200001010900000001000101000000002600000020002
feb 24, 2023 8:13:11 PM com.sun.corba.se.impl.transport.SocketOrChannelConnectionImpl <init>
ADVERTENCIA: "IOP00410201: (COMM_FAILURE) Connection failure: socketType: IIOP_CLEAR_TEXT; hostname: 127.0.0.1; port: 54524"
org.omg.CORBA.COMM_FAILURE:   vmcid: SUN  minor code: 201  completed: No
        at com.sun.corba.se.impl.logging.ORBUtilSystemException.connectFailure(ORBUtilSystemException.java:2200)
        at com.sun.corba.se.impl.logging.ORBUtilSystemException.connectFailure(ORBUtilSystemException.java:2221)
        at com.sun.corba.se.impl.transport.SocketOrChannelConnectionImpl.<init>(SocketOrChannelConnectionImpl.java:223)
        at com.sun.corba.se.impl.transport.SocketOrChannelConnectionImpl.<init>(SocketOrChannelConnectionImpl.java:236)
        at com.sun.corba.se.impl.transport.SocketOrChannelContactInfoImpl.createConnection(SocketOrChannelContactInfoImpl.java:119)
        at com.sun.corba.se.impl.protocol.CorbaClientRequestDispatcherImpl.beginRequest(CorbaClientRequestDispatcherImpl.java:187)
        at com.sun.corba.se.impl.protocol.CorbaClientDelegateImpl.request(CorbaClientDelegateImpl.java:137)
        at org.omg.CORBA.portable.ObjectImpl._request(ObjectImpl.java:449)
        at HelloApp._HelloStub.sayHello(_HelloStub.java:18)
        at HelloClient.main(HelloClient.java:28)
Caused by: java.net.ConnectException: Connection refused
        at sun.nio.ch.Net.connect0(Native Method)
        at sun.nio.ch.Net.connect(Net.java:459)
        at sun.nio.ch.Net.connect(Net.java:451)
        at sun.nio.ch.SocketChannelImpl.connect(SocketChannelImpl.java:646)
        at java.nio.channels.SocketChannel.open(SocketChannel.java:189)
        at com.sun.corba.se.impl.transport.DefaultSocketFactoryImpl.createSocket(DefaultSocketFactoryImpl.java:95)
        at com.sun.corba.se.impl.transport.SocketOrChannelConnectionImpl.<init>(SocketOrChannelConnectionImpl.java:207)
        ... 7 more

ERROR : org.omg.CORBA.COMM_FAILURE:   vmcid: SUN  minor code: 201  completed: No
org.omg.CORBA.COMM_FAILURE:   vmcid: SUN  minor code: 201  completed: No
        at com.sun.corba.se.impl.logging.ORBUtilSystemException.connectFailure(ORBUtilSystemException.java:2200)
        at com.sun.corba.se.impl.logging.ORBUtilSystemException.connectFailure(ORBUtilSystemException.java:2221)
        at com.sun.corba.se.impl.transport.SocketOrChannelConnectionImpl.<init>(SocketOrChannelConnectionImpl.java:223)
        at com.sun.corba.se.impl.transport.SocketOrChannelConnectionImpl.<init>(SocketOrChannelConnectionImpl.java:236)
        at com.sun.corba.se.impl.transport.SocketOrChannelContactInfoImpl.createConnection(SocketOrChannelContactInfoImpl.java:119)
        at com.sun.corba.se.impl.protocol.CorbaClientRequestDispatcherImpl.beginRequest(CorbaClientRequestDispatcherImpl.java:187)
        at com.sun.corba.se.impl.protocol.CorbaClientDelegateImpl.request(CorbaClientDelegateImpl.java:137)
        at org.omg.CORBA.portable.ObjectImpl._request(ObjectImpl.java:449)
        at HelloApp._HelloStub.sayHello(_HelloStub.java:18)
        at HelloClient.main(HelloClient.java:28)
Caused by: java.net.ConnectException: Connection refused
        at sun.nio.ch.Net.connect0(Native Method)
        at sun.nio.ch.Net.connect(Net.java:459)
        at sun.nio.ch.Net.connect(Net.java:451)
        at sun.nio.ch.SocketChannelImpl.connect(SocketChannelImpl.java:646)
        at java.nio.channels.SocketChannel.open(SocketChannel.java:189)
        at com.sun.corba.se.impl.transport.DefaultSocketFactoryImpl.createSocket(DefaultSocketFactoryImpl.java:95)
        at com.sun.corba.se.impl.transport.SocketOrChannelConnectionImpl.<init>(SocketOrChannelConnectionImpl.java:207)
        ... 7 more
```

**b. ¿Qué sucedería si lanzase varios servidores a la vez y un solo cliente?**

**Si varios servidores se lanzaran al mismo tiempo por el mismo puerto y un solo cliente, el cliente solo podría establecer conexión con uno de los servidores. Esto significa que el resto de los servidores se quedarían sin recibir peticiones. Para evitar esto, los servidores deben escuchar en diferentes puertos para poder establecer conexión con el cliente.**

**PRIMER SERVIDOR**

```
marquines@MacBook-Air-de-marcos PRACTICA2-Distribuidos % java HelloServer  -ORBInitialHost localhost -ORBInitialPort 2000
HelloServer ready and waiting ...
```

**SEGUNDO SERVIDOR**

```
HelloServer ready and waiting ...
HelloServer Exiting ...
feb 24, 2023 8:32:04 PM com.sun.corba.se.impl.orb.ORBImpl checkShutdownState
ADVERTENCIA: "IOP01600004: (BAD_INV_ORDER) ORB has shutdown"
org.omg.CORBA.BAD_INV_ORDER:   vmcid: OMG  minor code: 4  completed: No
        at com.sun.corba.se.impl.logging.OMGSystemException.badOperationAfterShutdown(OMGSystemException.java:224)
        at com.sun.corba.se.impl.logging.OMGSystemException.badOperationAfterShutdown(OMGSystemException.java:246)
        at com.sun.corba.se.impl.orb.ORBImpl.checkShutdownState(ORBImpl.java:1341)
        at com.sun.corba.se.impl.orb.ORBImpl.peekInvocationInfo(ORBImpl.java:1539)
        at com.sun.corba.se.impl.protocol.CorbaMessageMediatorImpl.runServantPostInvoke(CorbaMessageMediatorImpl.java:2250)
        at com.sun.corba.se.impl.protocol.CorbaMessageMediatorImpl.createResponseHelper(CorbaMessageMediatorImpl.java:2189)
        at com.sun.corba.se.impl.protocol.CorbaMessageMediatorImpl.createResponseHelper(CorbaMessageMediatorImpl.java:2165)
        at com.sun.corba.se.impl.protocol.CorbaMessageMediatorImpl.createResponse(CorbaMessageMediatorImpl.java:2013)
        at com.sun.corba.se.impl.protocol.CorbaMessageMediatorImpl.createReply(CorbaMessageMediatorImpl.java:623)
        at HelloApp.HelloPOA._invoke(HelloPOA.java:47)
        at com.sun.corba.se.impl.protocol.CorbaServerRequestDispatcherImpl.dispatchToServant(CorbaServerRequestDispatcherImpl.java:654)
        at com.sun.corba.se.impl.protocol.CorbaServerRequestDispatcherImpl.dispatch(CorbaServerRequestDispatcherImpl.java:205)
        at com.sun.corba.se.impl.protocol.CorbaMessageMediatorImpl.handleRequestRequest(CorbaMessageMediatorImpl.java:1700)
        at com.sun.corba.se.impl.protocol.CorbaMessageMediatorImpl.handleRequest(CorbaMessageMediatorImpl.java:1558)
        at com.sun.corba.se.impl.protocol.CorbaMessageMediatorImpl.handleInput(CorbaMessageMediatorImpl.java:940)
        at com.sun.corba.se.impl.protocol.giopmsgheaders.RequestMessage_1_2.callback(RequestMessage_1_2.java:198)
        at com.sun.corba.se.impl.protocol.CorbaMessageMediatorImpl.handleRequest(CorbaMessageMediatorImpl.java:712)
        at com.sun.corba.se.impl.transport.SocketOrChannelConnectionImpl.dispatch(SocketOrChannelConnectionImpl.java:474)
        at com.sun.corba.se.impl.transport.SocketOrChannelConnectionImpl.doWork(SocketOrChannelConnectionImpl.java:1237)
        at com.sun.corba.se.impl.orbutil.threadpool.ThreadPoolImpl$WorkerThread.performWork(ThreadPoolImpl.java:490)
        at com.sun.corba.se.impl.orbutil.threadpool.ThreadPoolImpl$WorkerThread.run(ThreadPoolImpl.java:519)
```

**CLIENTE**

```
marquines@MacBook-Air-de-marcos PRACTICA2-Distribuidos % java HelloClient -ORBInitialHost localhost -ORBInitialPort 2000
Obtained a handle on server object: IOR:000000000000001749444c3a48656c6c6f4170702f48656c6c6f3a312e3000000000000100000000000000820001020000000000a3132372
e302e302e3100d4fc00000031afabcb000000002084d36f5d0000000100000000000000010000008526f6f74504f4100000000800000001000000014000000000000020000000100000
2000000000000100010000000205010001000100200001010900000001000101000000026000000020002

Hello world !!
```

**c. ¿Puedes conectarte al servidor de un compañero? ¿Cómo lo harías?**

SI, Para poder conectar dos equipos independientes, una máquina debe actuar como servidor y la otra como cliente. En la máquina servidor, se debe ejecutar el comando java HelloServer, especificando un puerto abierto en el equipo de la máquina servidor y su propia dirección IP en los parámetros -ORBInitialPort y -ORBInitialHost, respectivamente. Posteriormente, en la máquina cliente, se debe ejecutar el comando java HelloClient, utilizando los mismos parámetros que en el servidor. Esto permitirá la conexión entre los dos equipos.

UNIVERSIDAD NEBRIJA

2. **Actualiza un repositorio de Github con una aplicación Java CORBA (7 puntos)**
   **Aquí debéis hacer un fork de una aplicación en Github y realizar modificaciones en ella. Por ejemplo, imaginemos que tenemos una calculadora que funciona con CORBA y únicamente tiene las funciones de suma, resta, multiplicar y dividir. Podéis añadir por ejemplo: operar con raíces cuadradas o añadir que utilice decimales. Pautas:**
   - **El código que se añada debe ser por un lado pegado en este documento y por otro lado, se deben realizar los commits en el repositorio.**
   - **El código debe contener comentarios propios respecto a como funciona la aplicación.**
   - **Toda la información que tenga el README.md nunca está demás.**
   - **Intenta que sea una aplicación/funcionalidad diferente la que modificas (3 puntos) No todos los compañeros vamos a tener Calculadoras, busca otras aplicaciones y diferénciate.**

**Calculadora.idl**

```
/*      Calculadora.idl     */
module MCalculadora {
    struct Operadores {
        long a;
        long b;
    };
    interface ICalculadora
    {
        long sumar(in Operadores Ops);
        long restar(in Operadores Ops);
        long multiplicar(in Operadores Ops);
        long dividir(in Operadores Ops);
        double exponencial(in Operadores Ops);
        double raiz(in Operadores Ops);
        double modulo(in Operadores Ops);
        double factorial(in Operadores Ops);
        double potencia(in Operadores Ops);
        double logaritmo(in Operadores Ops);
        double logaritmoBase10(in Operadores Ops);
        double seno(in Operadores Ops);
        double coseno(in Operadores Ops);
        double tangente(in Operadores Ops);
        double secante(in Operadores Ops);
        double cosecante(in Operadores Ops);
        double cotangente(in Operadores Ops);

    };
};
```

**ImplementacionFunciones.java**

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

public class ImplementacionFunciones extends ICalculadoraPOA {
    public ImplementacionFunciones() {
        super();
    }

    public int sumar(Operadores ops) {
        return ops.a + ops.b;
    }

    public int restar(Operadores ops) {
        return ops.a - ops.b;
    }

    public int multiplicar(Operadores ops) {
        return ops.a * ops.b;
    }

    public int dividir(Operadores ops) {
        return ops.a / ops.b;
    }

    public double exponencial(Operadores ops) {
        return (double) Math.exp(ops.a);
    }

    public double raiz(Operadores ops) {
        return (int) Math.sqrt(ops.a);
    }

    public double modulo(Operadores ops) {
        return ops.a % ops.b;
    }

    public double factorial(Operadores ops) {
        int fact = 1;
        for (int i = 1; i <= ops.a; i++) {
            fact = fact * i;
        }
        return fact;
    }

    public double potencia(Operadores ops) {
        return (double) Math.pow(ops.a, ops.b);
```

```
    }

    public double logaritmo(Operadores ops) {
        return (double) Math.log(ops.a);
    }

    public double logaritmoBase10(Operadores ops) {
        return (double) Math.log10(ops.a);
    }

    public double seno(Operadores ops) {
        return (double) Math.sin(ops.a);
    }

    public double coseno(Operadores ops) {
        return (double) Math.cos(ops.a);
    }

    public double tangente(Operadores ops) {
        return (double) Math.tan(ops.a);
    }

    public double cosecante(Operadores ops) {
        return (double) 1 / Math.sin(ops.a);
    }

    public double secante(Operadores ops) {
        return (double) 1 / Math.cos(ops.a);
    }

    public double cotangente(Operadores ops) {
        return (double) 1 / Math.tan(ops.a);
    }
}
```

**CalcServer.java**

```java
import MCalculadora.*;
import org.omg.CosNaming.*;
import org.omg.CosNaming.NamingContextPackage.*;
import org.omg.CORBA.*;
import org.omg.PortableServer.*;
import org.omg.PortableServer.POA;
import java.util.Properties;

public class CalcServer {
    public CalcServer() {

    }
    public static void main(String[] args) {
        try {
            // Creación e Inicialización del ORB
            ORB orb = ORB.init(args, null);

            // Obtenemos las referencias a rootpoa y se activa el administrador POA
            // POAManager
            POA rootpoa = POAHelper.narrow(orb.resolve_initial_references("RootPOA"));
            rootpoa.the_POAManager().activate();

            // Se realiza la instancia para implementar las Funciones Remotas
            ImplementacionFunciones calcimp = new ImplementacionFunciones();

            // Se obtiene la referencia del servidor (servant)
            org.omg.CORBA.Object ref = rootpoa.servant_to_reference(calcimp);
            ICalculadora href = ICalculadoraHelper.narrow(ref);

            // Se obtiene la raiz del contexto de nombres de CORBA
            org.omg.CORBA.Object objRef = orb.resolve_initial_references("NameService");

            // NamingContextExt es parte de la especificación INS para interoperabilidad
            NamingContextExt ncRef = NamingContextExtHelper.narrow(objRef);

            // Se asigna el nombre del servicio para poder localizarlo en el servicio de
            // nombres (binding)
            String nombreServicio = "Calculadora";
            NameComponent path[] = ncRef.to_name(nombreServicio);
            ncRef.rebind(path, href);

            System.out.println("Servidor Calculadora Remota Lista!! ...");
            orb.run();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

**CalcClient.java**

```java
import MCalculadora.*;

import org.omg.CORBA.*;
import org.omg.CosNaming.*;
import org.omg.CosNaming.NamingContextPackage.*;
import java.io.*;
import java.util.*;

public class CalcClient {
    static ICalculadora c = null;

    public static void main(String[] args) {
        try {
            // Se crea e inicializa el ORB
            ORB orb = ORB.init(args, null);
            // Se obtienen la raíz del contexto de nombres de CORBA
            org.omg.CORBA.Object objRef = orb.resolve_initial_references("NameService");
            // Se recomienda el uso de NamingCOntextExt en lugar de NamingContext, ya que
            // es parte de la interoperabilidad en el servicio de nombres
            NamingContextExt ncRef = NamingContextExtHelper.narrow(objRef);

            // Asignamos el nombre del servicio que estamos buscando para que sea
            // resuelto por el servicio de nombres
            String nombreServicio = "Calculadora";
            c = ICalculadoraHelper.narrow(ncRef.resolve_str(nombreServicio));

            // Llamada a los procedimientos remotos
            Operadores ops = new Operadores();
            ops.a = 10;
            ops.b = 10;
            System.out.println("Suma: " + c.sumar(ops));
            System.out.println("Resta: " + c.restar(ops));
            System.out.println("Multiplicación: " + c.multiplicar(ops));
            System.out.println("División: " + c.dividir(ops));
            System.out.println("Exponencial: " + c.exponencial(ops));
            System.out.println("Raíz: " + c.raiz(ops));
            System.out.println("Módulo: " + c.modulo(ops));
            System.out.println("Factorial: " + c.factorial(ops));
            System.out.println("Potencia: " + c.potencia(ops));
            System.out.println("Logaritmo: " + c.logaritmo(ops));
            System.out.println("Logaritmo Base 10: " + c.logaritmoBase10(ops));
            System.out.println("Seno: " + c.seno(ops));
            System.out.println("Coseno: " + c.coseno(ops));
            System.out.println("Tangente: " + c.tangente(ops));
            System.out.println("Cosecante: " + c.cosecante(ops));
            System.out.println("Secante: " + c.secante(ops));
            System.out.println("Cotangente: " + c.cotangente(ops));
```

```
        } catch (Exception e) {
            e.printStackTrace();
            System.exit(-1);
        }
    }
}
```

**Para ejecutar el programa necesitamos tener abiertas tres ventanas del Símbolo del sistema. La primera iniciará el puerto, la segunda ejecutará el servidor y la tercera el cliente. Metiendo el código del Hola Mundo tenemos estos resultados:**

```
marquines@macbook-air-de-marcos practica2 % tnameserv -ORBInitialPort 2000
Contexto de Nomenclatura Inicial:
IOR:000000000000002b49444c3a6f6d672e6f72672f436f734e616d696e672f4e616d696e67436f6e746578744578743a312e3000000000000100000000000000090a000102000000000d3139322e3136382
e312e333200000 7d000000045afabcb0000000020000f424000000001000000000000000200000008526f6f74504f41000000000d544e616d65536572766963650000000000000000800000001000000114
0000000000000200000001000000200000000000001000100000002050100010001002000010109000000010001010000000026000000020002
TransientNameServer: definiendo puerto para referencias a objeto iniciales en: 2000
Listo.
```

```
marquines@macbook-air-de-marcos practica2 % java CalcServer  -ORBInitialHost localhost -ORBInitialPort 2000
Servidor Calculadora Remota Lista!! ...
```

```
marquines@macbook-air-de-marcos practica2 % java CalcClient  -ORBInitialHost localhost -ORBInitialPort 2000
Suma: 20
Resta: 0
Multiplicación: 100
División: 1
Exponencial: 22026.465794806718
Raíz: 3.0
Módulo: 0.0
Factorial: 3628800.0
Potencia: 1.0E10
Logaritmo: 2.302585092994046
Logaritmo Base 10: 1.0
Seno: -0.5440211108893698
Coseno: -0.8390715290764524
Tangente: 0.6483608274590866
Cosecante: -1.8381639608896658
Secante: -1.1917935066878957
Cotangente: 1.5423510453569202
marquines@macbook-air-de-marcos practica2 %
```