

Name: Sayyed Sohail Rashid	Course Name: DC-LAB
Class: BE-CO	Batch: 01
Roll no: 18CO48	Experiment No: 01

Aim : To implement the Client / Server using RPC.

Code:

Client.java

```
int a, b, c;

while (true) {
    fun = receiveRead.readLine();
    if (fun != null) {
        System.out.println("Operation : " + fun);
    }
    a = Integer.parseInt(receiveRead.readLine());
    System.out.println("Parameter 1 : " + a);
    b = Integer.parseInt(receiveRead.readLine());
    if (fun.compareTo("add") == 0) {
        c = a + b;
        c = a / b;
        System.out.println("Division = " + c);
        pwrite.println("Division = " + c);
    }
    System.out.flush();
}
```

Code:

Server.java

```
import java.io.*;
import java.net.*;
class cli {
    public static void main(String[] args) throws Exception {
        Socket sock = new Socket("127.0.0.1", 3000);
        BufferedReader keyRead = new BufferedReader(new InputStreamReader(System.in));
        OutputStream ostream = sock.getOutputStream();
        PrintWriter pwrite = new PrintWriter(ostream, true);
```

```

InputStream istream = sock.getInputStream();
BufferedReader receiveRead = new BufferedReader(new InputStreamReader(istream));
System.out.println("Client ready, type and press Enter key");
String receiveMessage, sendMessage, temp;
while (true) {
    System.out.println("\nEnter operation to perform(add,sub,mul,div)....");
    temp = keyRead.readLine();
    sendMessage = temp.toLowerCase();
    pwrite.println(sendMessage);
    System.out.println("Enter first parameter :");
    sendMessage = keyRead.readLine();
    pwrite.println(sendMessage);
    System.out.println("Enter second parameter : ");
    sendMessage = keyRead.readLine();
    pwrite.println(sendMessage);
    System.out.flush();
    if ((receiveMessage = receiveRead.readLine()) != null) {
        System.out.println(receiveMessage);
    }
}
}
}
}

```

```

Selection Path Priority Status
-----
0 /usr/lib/jvm/java-1.6-openjdk-amd64/bin/java 1611 auto mode
1 /opt/jdk1.8.0_301/bin/java 180 manual mode
2 /usr/lib/jvm/java-11-openjdk-amd64/bin/java 1111 manual mode
3 /usr/lib/jvm/java-16-openjdk-amd64/bin/java 1611 auto mode

Press <Enter> to keep the current choice[*], or type selection number: 1
nikte@nikte22:~/Desktop/dc-exp-02$ sudo update-alternatives --config java
There are 3 choices for the alternative java (providing /usr/bin/java):

Selection Path Priority Status
-----
0 /usr/lib/jvm/java-1.6-openjdk-amd64/bin/javac 1611 auto mode
1 /usr/lib/jvm/java-11-openjdk-amd64/bin/javac 1111 manual mode
2 /usr/lib/jvm/java-16-openjdk-amd64/bin/javac 1611 auto mode

Press <Enter> to keep the current choice[*], or type selection number: 1
update-alternatives: using /usr/lib/jvm/java-11-openjdk-amd64/bin/javac to provide /u
nikte@nikte22:~/Desktop/dc-exp-02$ sudo update-alternatives --config java
There are 3 choices for the alternative java (providing /usr/bin/java):

Selection Path Priority Status
-----
0 /usr/lib/jvm/java-1.6-openjdk-amd64/bin/java 1611 auto mode
1 /usr/lib/jvm/java-11-openjdk-amd64/bin/java 1111 manual mode
2 /usr/lib/jvm/java-16-openjdk-amd64/bin/java 1611 auto mode

```

Conclusion:

We have successfully performed the experiment on client/server using RPC.