CODE

ClientImplementation.java

```
package Client;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;

/**
 * This Java class implements the ClientInterface and defines a method to receive and print a message.
 */
public class ClientImplementation extends UnicastRemoteObject implements ClientInterface {
   protected ClientImplementation() throws RemoteException {
   }

   public void receiveMessage(String message) throws RemoteException {
      System.out.println("Received message: " + message);
   }
}
```

ClientInterface.java

```
package Client;
import java.rmi.Remote;
import java.rmi.RemoteException;

// This code is defining a remote interface called `ClientInterface` that extends the `Remote`
// interface. It declares a single method called `receiveMessage` that takes a `String` parameter and
// throws a `RemoteException`. This interface is used to define the methods that can be called remotely
// by a server in a distributed system.
public interface ClientInterface extends Remote {
   void receiveMessage(String message) throws RemoteException;
}
```

ClientMain.java

```
package Client;

import java.rmi.Naming;
import java.util.Scanner;
import Server.ServerInterface;
/**
 * This Java class registers a client with a server and broadcasts a message to all registered clients.
 */
public class ClientMain {
   public static void main(String[] args) {
      try {
         ServerInterface server = (ServerInterface) Naming.lookup("rmi://localhost/Server");
         ClientInterface client = new ClientImplementation();
         server.registerClient(client);

         Scanner scanner = new Scanner(System.in);
         System.out.print("Enter a message to broadcast: ");String message = scanner.nextLine();

         server.broadcastMessage(message);
      } catch (Exception e) {
         System.out.println("Client exception: " + e.getMessage());
         e.printStackTrace();
      }
   }
}
```

ServerImplementation.java

```java
package Server;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;
import java.util.ArrayList;
import java.util.List;

/**
 * This Java class implements a server interface that allows clients to register and broadcast messages
 * to all registered clients.
 */
public class ServerImplementation extends UnicastRemoteObject implements ServerInterface {
    private List<Client.ClientInterface> clients;

    public ServerImplementation() throws RemoteException {
        clients = new ArrayList<>();
    }

    public void registerClient(Client.ClientInterface client) throws RemoteException {
        clients.add(client);
    }

    public void broadcastMessage(String message) throws RemoteException {
        System.out.println("Broadcasting message: " + message);
        for (Client.ClientInterface client : clients) {
            client.receiveMessage(message);
        }
    }
}
```

ServerInterface.java

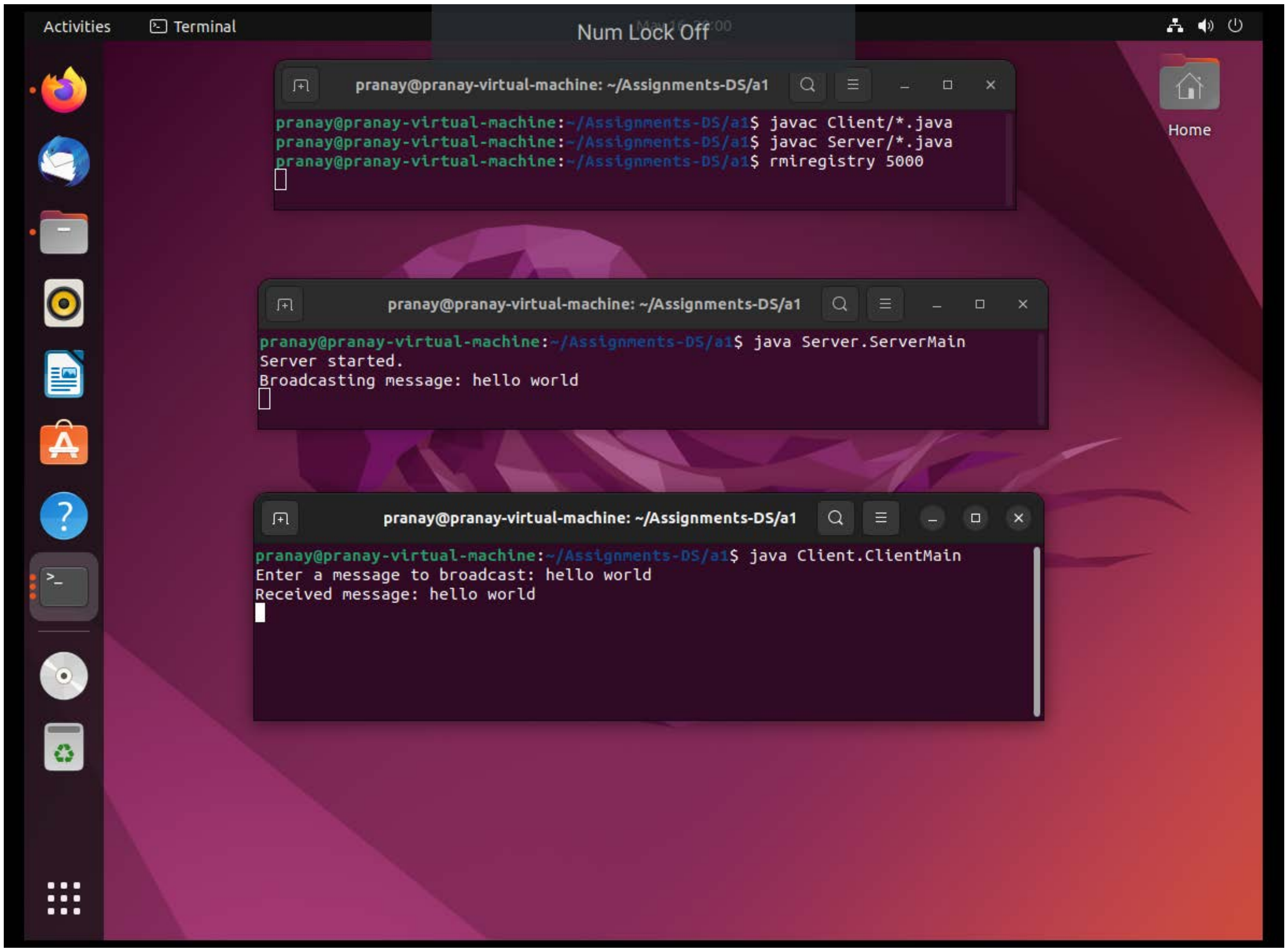
```java
package Server;
import java.rmi.Remote;
import java.rmi.RemoteException;

import Client.ClientInterface;

// This is a Java interface for a remote server that extends the `Remote` interface, indicating that it
// can be accessed remotely. It declares two methods that can be called by clients: `registerClient`
// and `broadcastMessage`.
public interface ServerInterface extends Remote {
    void registerClient(ClientInterface client) throws RemoteException;
    void broadcastMessage(String message) throws RemoteException;
}
```

ServerMain.java

```java
package Server;
import java.rmi.Naming;
import java.rmi.registry.LocateRegistry;
/**
 * This Java class starts a server using RMI (Remote Method Invocation) technology.
 */
public class ServerMain {
    public static void main(String[] args) {
```

```
try {
        ServerInterface server = new ServerImplementation();
        LocateRegistry.createRegistry(1099);
        Naming.rebind("rmi://localhost/Server", server);
        System.out.println("Server started.");
    } catch (Exception e) {
        System.out.println("Server exception: " + e.getMessage());
        e.printStackTrace();
    }
  }
}
```

OUTPUT



```
pranay@pranay-virtual-machine:~/Assignments-DS/a1$ javac Client/*.java
pranay@pranay-virtual-machine:~/Assignments-DS/a1$ javac Server/*.java
pranay@pranay-virtual-machine:~/Assignments-DS/a1$ rmiregistry 5000
```

```
pranay@pranay-virtual-machine:~/Assignments-DS/a1$ java Server.ServerMain
Server started.
Broadcasting message: hello world
```

```
pranay@pranay-virtual-machine:~/Assignments-DS/a1$ java Client.ClientMain
Enter a message to broadcast: hello world
Received message: hello world
```