



Instituto Politécnico Nacional.
Escuela Superior De Cómputo.



Materia:
Desarrollo de Sistemas distribuidos.

Tema:
Multiplicación de matrices usando
objetos distribuidos.
(Tarea 06)

Profesor:
Carlos Pineda Guerrero.

Alumno:
Mario Alberto Miranda Sandoval.

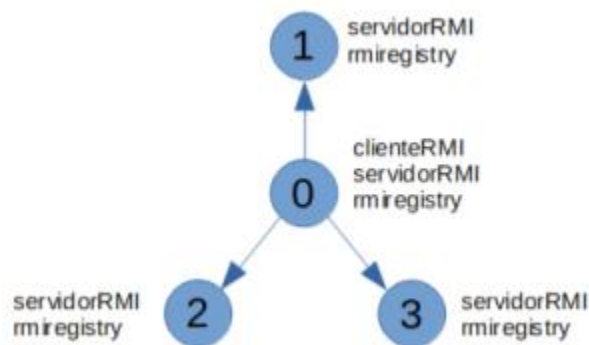
Grupo:
4CM5

Objetivo.

El alumno deberá desarrollar un sistema que calcule el producto de dos matrices cuadradas utilizando Java RMI.

Desarrollo.





Al igual que la tarea 3, se hará la multiplicación de matrices distribuidas, pero esta vez se ejecutará usando RMI, primeramente, se deberá implementar la siguiente topología de red.



Posteriormente, cada nodo es una máquina virtual de Azure, en la cual el sistema operativo que usaremos será Ubuntu.

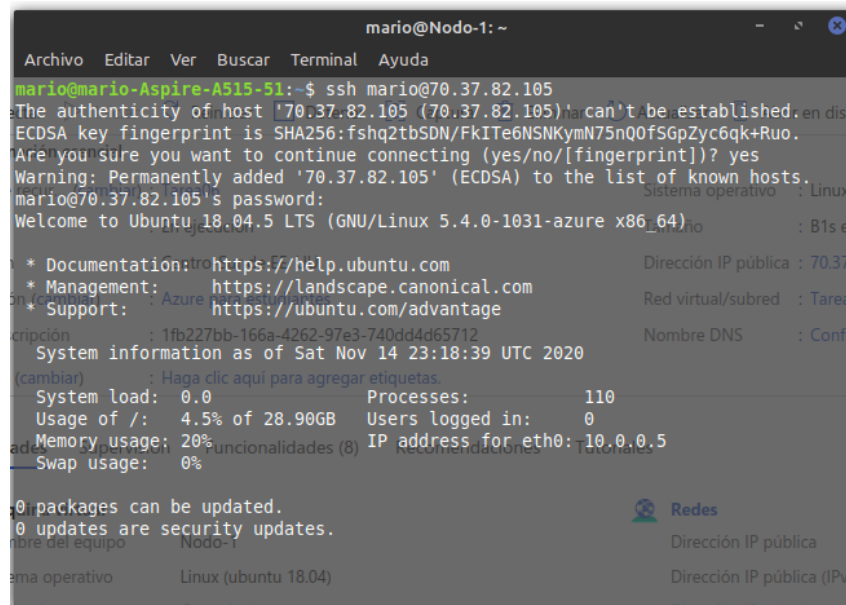
Para poder realizar la práctica es importante destacar que las máquinas virtuales deben crearse en el mismo grupo de recursos, a modo que se pueda utilizar su dirección IP privada para comunicarse entre ellas.

Suscripciones: Azure para estudiantes

Filtrar por nombre...	Todos los grupos de recursos ▾	Todos los tipos ▾	[
4 elementos			
<input type="checkbox"/> Nombre ↑↓	Tipo ↑↓	Estado	Grupo de recursos ↑↓
<input type="checkbox"/>  nodo-0	Máquina virtual	En ejecución	Tarea06
<input type="checkbox"/>  nodo-1	Máquina virtual	En ejecución	Tarea06
<input type="checkbox"/>  nodo-2	Máquina virtual	En ejecución	Tarea06
<input type="checkbox"/>  nodo-3	Máquina virtual	En ejecución	Tarea06

Se puede observar como las 4 máquinas virtuales están en el mismo grupo de recursos.

Una vez que tenemos las máquinas listas para usarse, se procede a explicarse el proceso que se hizo en cada máquina virtual con una máquina de ejemplo.



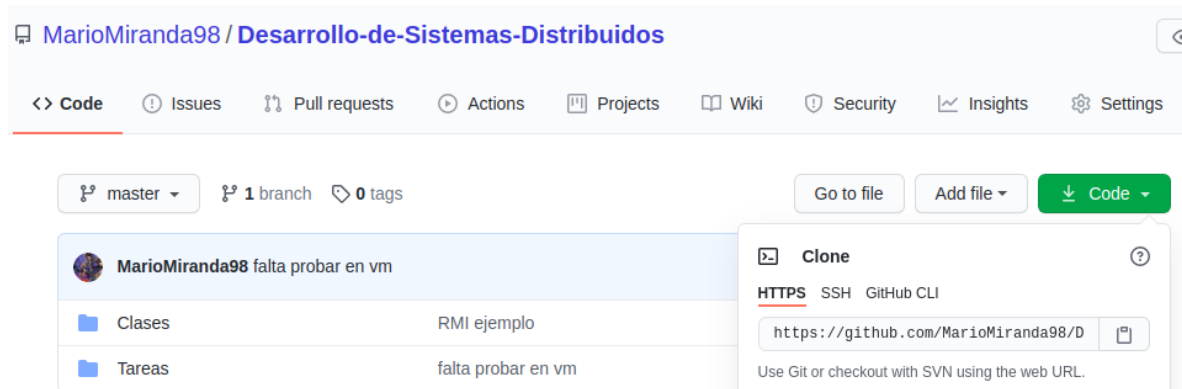
```
mario@Nodo-1: ~  
Archivo Editar Ver Buscar Terminal Ayuda  
mario@mario-Aspire-A515-51:~$ ssh mario@70.37.82.105  
The authenticity of host '70.37.82.105 (70.37.82.105)' can't be established.  
ECDSA key fingerprint is SHA256:fshq2tbSDN/FkITe6NSNKymN75nQ0fSGpZyc6qk+Ruo.  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  
Warning: Permanently added '70.37.82.105' (ECDSA) to the list of known hosts.  
mario@70.37.82.105's password:  
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 5.4.0-1031-azure x86_64)  
  
 * Documentation: https://help.ubuntu.com  
 * Management: https://landscape.canonical.com  
 * Support: https://ubuntu.com/advantage  
  
System information as of Sat Nov 14 23:18:39 UTC 2020  
(cambiar) : Haga clic aquí para agregar etiquetas  
System load: 0.0 Processes: 110  
Usage of /: 4.5% of 28.90GB Users logged in: 0  
Memory usage: 20% IP address for eth0: 10.0.0.5  
Swap usage: 0%  
  
0 packages can be updated.  
0 updates are security updates.  
  
Nombre del equipo: Nodo-1  
Sistema operativo: Linux (ubuntu 18.04)  
Dirección IP pública: 70.37.82.105  
Red virtual/subred: 10.0.0.0/24  
Nombre DNS: Conf...
```

Como primer paso abrimos una terminal en Linux, donde nos conectaremos de manera remota a la máquina virtual usando ssh, con el siguiente comando:

ssh <nombre usuario>@<dirección IP pública>

Nos pedirá la contraseña con la cual creamos la máquina virtual, una vez introducida nos dejará tener acceso a la máquina virtual desde la terminal.

Ahora para obtener el código, se uso el wget donde se hará la petición al repositorio de GitHub, para esto primero se necesita copiar el enlace del repositorio, este puede ser encontrado al entrar al repositorio como se ve en la siguiente imagen.



Una vez copiado el enlace, procedemos a la máquina virtual para usar el wget.

```
mario@Nodo-1: ~
Archivo Editar Ver Buscar Terminal Ayuda
System information as of Sat Nov 14 23:18:39 UTC 2020
System load: 0.0 Processes: 110
Usage of /: 4.5% of 28.90GB Users logged in: 0
Memory usage: 20% IP address for eth0: 10.0.0.5
Swap usage: 0%
0 packages can be updated.
0 updates are security updates.
n (cambiar) : Azure para estudiantes
cripción : 1fb227bb-166a-4262-97e3-740dd4d65712
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo root" for details.
mario@Nodo-1:~$ wget https://github.com/MarioMiranda98/Desarrollo-de-Sistemas-Di
stribuidos/archive/master.tar.gz
```

En la imagen se puede observar el comando, que tiene la siguiente estructura.

wget <enlace al repositorio (quitar el .git)>/archive/<nombre y formato en el que se va a descargar>

```
mario@Nodo-1: ~
Archivo Editar Ver Buscar Terminal Ayuda
mario@Nodo-1:~$ wget https://github.com/MarioMiranda98/Desarrollo-de-Sistemas-Di
stribuidos/archive/master.tar.gz
--2020-11-14 23:19:54-- https://github.com/MarioMiranda98/Desarrollo-de-Sistema
s-Distribuidos/archive/master.tar.gz
Resolving github.com (github.com)... 140.82.114.4
Connecting to github.com (github.com)|140.82.114.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://codeload.github.com/MarioMiranda98/Desarrollo-de-Sistemas-Di
stribuidos/tar.gz/master [following]
--2020-11-14 23:19:55-- https://codeload.github.com/MarioMiranda98/Desarrollo-d
e-Sistemas-Distribuidos/tar.gz/master
Resolving codeload.github.com (codeload.github.com)... 140.82.114.10
Connecting to codeload.github.com (codeload.github.com)|140.82.114.10|:443... co
nected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/x-gzip]
Saving to: 'master.tar.gz'

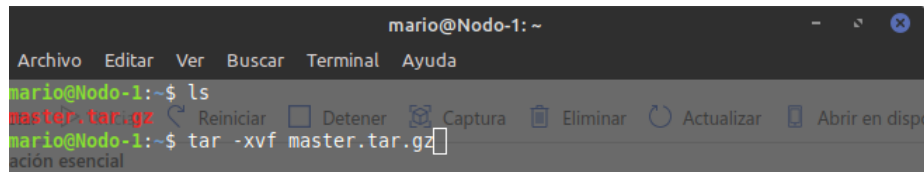
master.tar.gz      [ <=> ] 10.52M  21.5MB/s  saved in 0.5s
2020-11-14 23:19:55 (21.5 MB/s) - 'master.tar.gz' saved [11033856]
```

```
mario@Nodo-1: ~
Archivo Editar Ver Buscar Terminal Ayuda
mario@Nodo-1:~$ ls
master.tar.gz
mario@Nodo-1:~$
```

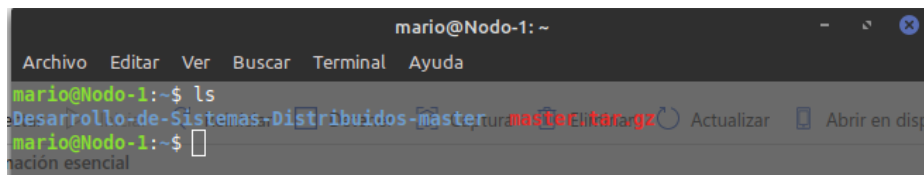
Se puede ver la descarga del código del repositorio, si usamos el comando `ls`, podremos observar que tenemos el archivo `tar.gz` que se especificó.

Ahora para extraer los archivos usamos el siguiente comando:

```
tar -xvf <nombre_archivo.tar.gz>
```



Una vez ejecutado el comando tar, nos quedaran los archivos que descargamos del repositorio en la máquina virtual, así como se ve en la siguiente imagen.



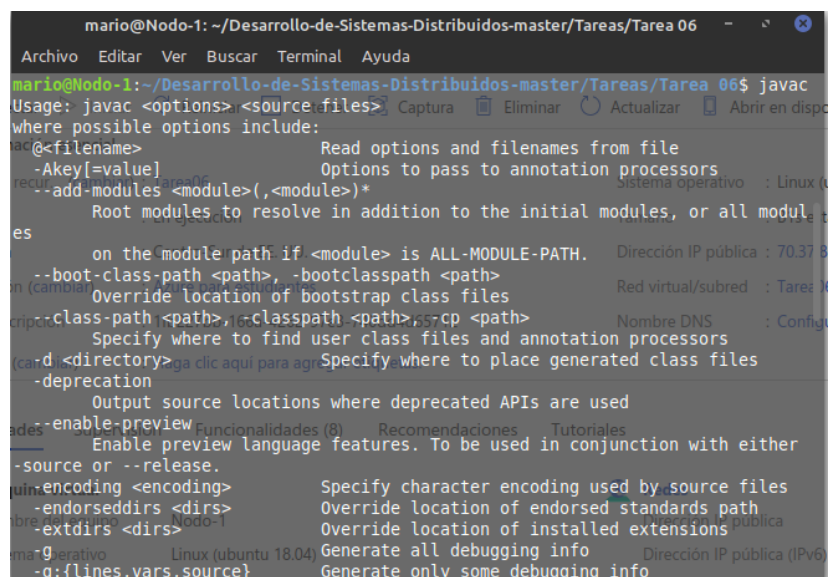
Ahora al ser máquinas virtuales recién creadas estas no poseen java instalado, por lo que correremos los dos siguientes comandos en el orden que son mostrados.

```
sudo apt update
```

```
sudo apt-get install openjdk-11-jdk-headless
```

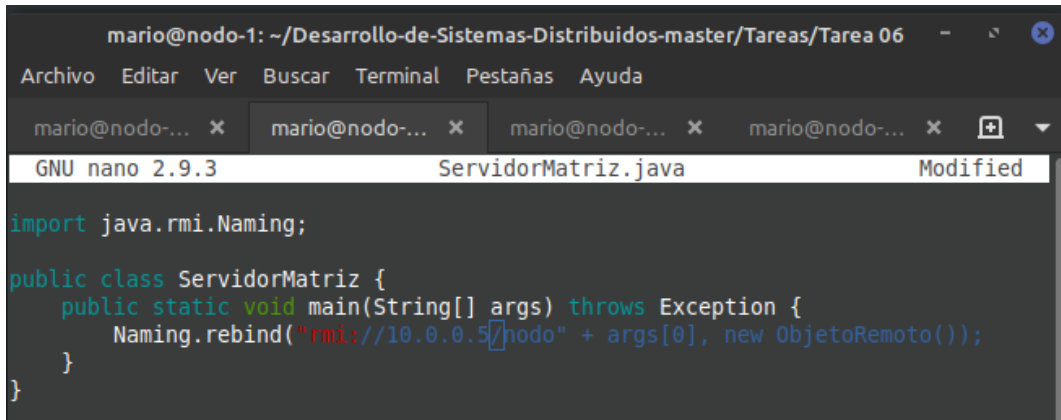
El primer comando es para actualizar los paquetes que tiene Ubuntu, mientras que el segundo comando es usado para instalar el jdk con la configuración del compilador.

En la siguiente imagen se aprecia como la máquina virtual cuenta ya con Java y su compilador.



Ese proceso es el que se repetirá para cada máquina virtual que tengamos.

Ahora, en las cuatro máquinas virtuales que hemos creado y configurado previamente, en el archivo **ServidorRMI.java**, se colocará la dirección IP privada correspondiente a cada máquina virtual.

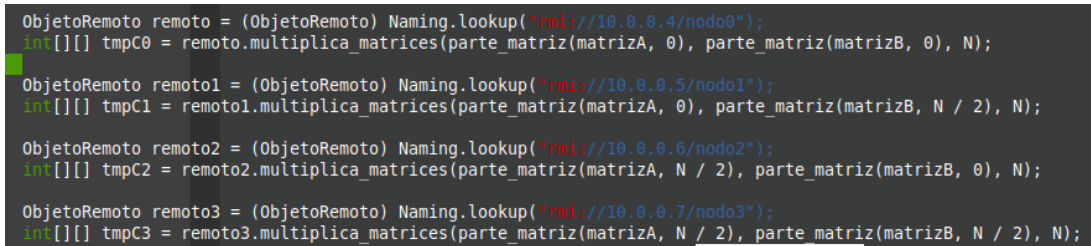


```
mario@nodo-1: ~/Desarrollo-de-Sistemas-Distribuidos-master/Tareas/Tarea 06
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
mario@nodo-... x mario@nodo-... x mario@nodo-... x mario@nodo-... x
GNU nano 2.9.3 ServidorMatriz.java Modified

import java.rmi.Naming;

public class ServidorMatriz {
    public static void main(String[] args) throws Exception {
        Naming.rebind("rmi://10.0.0.5/nodo" + args[0], new ObjetoRemoto());
    }
}
```

Y en el archivo **ClienteRMI.java**, en el método **Naming.lookup** se colocará la misma cadena que se colocó a cada servidor.



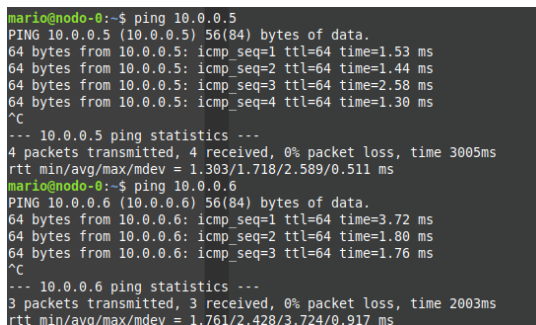
```
ObjetoRemoto remoto = (ObjetoRemoto) Naming.lookup("rmi://10.0.0.4/nodo0");
int[][] tmpC0 = remoto.multiplica_matrices(parte_matriz(matrizA, 0), parte_matriz(matrizB, 0), N);

ObjetoRemoto remoto1 = (ObjetoRemoto) Naming.lookup("rmi://10.0.0.5/nodo1");
int[][] tmpC1 = remoto1.multiplica_matrices(parte_matriz(matrizA, 0), parte_matriz(matrizB, N / 2), N);

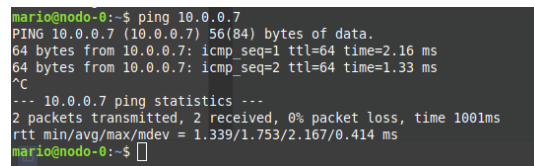
ObjetoRemoto remoto2 = (ObjetoRemoto) Naming.lookup("rmi://10.0.0.6/nodo2");
int[][] tmpC2 = remoto2.multiplica_matrices(parte_matriz(matrizA, N / 2), parte_matriz(matrizB, 0), N);

ObjetoRemoto remoto3 = (ObjetoRemoto) Naming.lookup("rmi://10.0.0.7/nodo3");
int[][] tmpC3 = remoto3.multiplica_matrices(parte_matriz(matrizA, N / 2), parte_matriz(matrizB, N / 2), N);
```

Nota: En la imagen hay un error, debido a que donde se hace el cast debe ser con la interfaz no con el objeto remoto, este error fue corregido para el funcionamiento de la práctica.



```
mario@nodo-0:~$ ping 10.0.0.5
PING 10.0.0.5 (10.0.0.5) 56(84) bytes of data.
64 bytes from 10.0.0.5: icmp_seq=1 ttl=64 time=1.53 ms
64 bytes from 10.0.0.5: icmp_seq=2 ttl=64 time=1.44 ms
64 bytes from 10.0.0.5: icmp_seq=3 ttl=64 time=2.58 ms
64 bytes from 10.0.0.5: icmp_seq=4 ttl=64 time=1.30 ms
^C
--- 10.0.0.5 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 1.303/1.718/2.589/0.511 ms
mario@nodo-0:~$ ping 10.0.0.6
PING 10.0.0.6 (10.0.0.6) 56(84) bytes of data.
64 bytes from 10.0.0.6: icmp_seq=1 ttl=64 time=3.72 ms
64 bytes from 10.0.0.6: icmp_seq=2 ttl=64 time=1.80 ms
64 bytes from 10.0.0.6: icmp_seq=3 ttl=64 time=1.76 ms
^C
--- 10.0.0.6 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 1.761/2.428/3.724/0.917 ms
```



```
mario@nodo-0:~$ ping 10.0.0.7
PING 10.0.0.7 (10.0.0.7) 56(84) bytes of data.
64 bytes from 10.0.0.7: icmp_seq=1 ttl=64 time=2.16 ms
64 bytes from 10.0.0.7: icmp_seq=2 ttl=64 time=1.33 ms
^C
--- 10.0.0.7 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 1.339/1.753/2.167/0.414 ms
mario@nodo-0:~$
```

Antes de pasar a la compilación y ejecución, mediante un ping se comprueba la conectividad de las máquinas virtuales.

Ahora, procedemos a levantar el servicio de `rmiregistry` en cada máquina con el comando `rmiregistry &`, además, compilamos y ponemos en ejecución los programas como se ve en las siguientes imágenes.

The image contains four terminal windows showing the execution of commands on different nodes:

- mario@nodo-0:** Executes `rmiregistry &` (PID 5387) and `javac *.java`.
- mario@nodo-1:** Executes `rmiregistry &` (PID 6580) and `java ServidorMatriz 1`.
- mario@nodo-2:** Executes `rmiregistry &` (PID 6044) and `java ServidorMatriz 2`.
- mario@nodo-3:** Executes `rmiregistry &` (PID 6375) and `java ServidorMatriz 3`.

Los servidores deben correrse primero, mientras que el cliente se corra después.

The terminal window on **mario@nodo-0:** shows the execution of `ls` after compilation, displaying the following files:

```

ClienteMatriz.class      MatrizMethodsInterface.java  ServidorMatriz.class
ClienteMatriz.java       ObjetoRemoto.class           ServidorMatriz.java
MatrizMethodsInterface.class  ObjetoRemoto.java

```

Como el cliente se corre en el nodo 0, es necesario crear una nueva terminal para conectarse al nodo 0, pero al ser la misma máquina, se puede omitir la compilación de los archivos.

The terminal window on **mario@nodo-0:** shows the execution of `java ClienteMatriz`, which displays three matrices and a checksum:

```

Matriz A
0 -1 -2 -3
2 1 0 -1
4 3 2 1
6 5 4 3

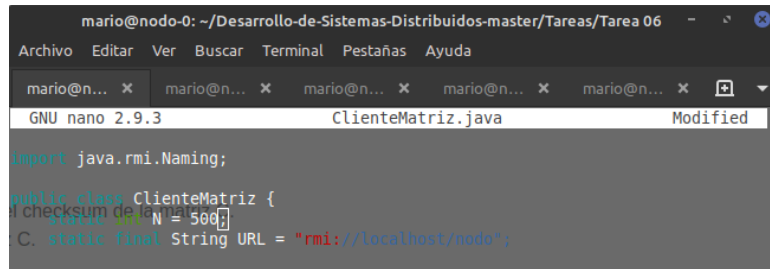
Matriz B
0 2 4 6
1 3 5 7
2 4 6 8
3 5 7 9

Matriz C
-28 -34 -40 -46
-4 -2 0 2
20 30 40 50
44 62 80 98

Checksum: 272.0

```

Ejecución del cliente, con $N = 4$.

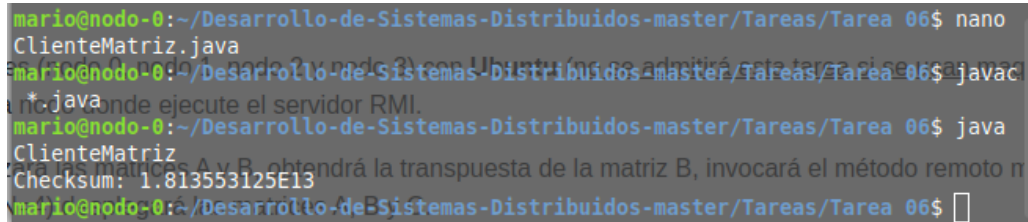


```
mario@nodo-0: ~/Desarrollo-de-Sistemas-Distribuidos-master/Tareas/Tarea 06
Archivo Editar Ver Buscar Terminal Pestañas Ayuda
mario@n... x mario@n... x mario@n... x mario@n... x mario@n... x
GNU nano 2.9.3 ClienteMatriz.java Modified

import java.rmi.Naming;

public class ClienteMatriz {
    static final int N = 500;
    static final String URL = "rmi://localhost/nodo0";
}
```

Ajustamos el cambio a probar con $N = 500$.



```
mario@nodo-0:~/Desarrollo-de-Sistemas-Distribuidos-master/Tareas/Tarea 06$ nano
ClienteMatriz.java
mario@nodo-0:~/Desarrollo-de-Sistemas-Distribuidos-master/Tareas/Tarea 06$ javac
*.java
mario@nodo-0:~/Desarrollo-de-Sistemas-Distribuidos-master/Tareas/Tarea 06$ java
ClienteMatriz
Checksum: 1.813553125E13
mario@nodo-0:~/Desarrollo-de-Sistemas-Distribuidos-master/Tareas/Tarea 06$
```

Recompilamos y ejecutamos para el programa con $N = 500$, se puede ver el comando nano, que nos sirve para abrir y modificar archivos desde terminal.

Conclusiones.

En esta práctica se puede notar a la perfección la diferencia de usar objetos remotos, ya que cuando esta misma actividad se hizo en la tarea 3, usar sockets hizo que fuera una tarea más complicada ya que él envió de los datos lo hacían una tarea complicada.

El usar objetos remotos nos garantiza una forma más sencilla de realizar la tarea además de que la integridad de los datos se garantiza de una mejor manera en contraste a enviar datos por sockets, incluso esto se ve reflejado en la cantidad de líneas de código que se usaron para el programa.

Aunque se documento en el reporte del error en su respectiva imagen, decidí incluirlo ya que cuando me salto ese error y buscar como solucionarlo, vi que era un error algo común por eso tome la decisión de dejar la imagen con el error y hacer su respectiva observación.