



**INSTITUTO POLITÉCNICO NACIONAL.**  
ESCUELA SUPERIOR DE CÓMPUTO.



## **Sistemas Distribuidos**

### **Tarea 05: Reporte de “Multiplicación de matrices utilizando objetos distribuidos”**

Alumno: Oaxaca Pérez David Arturo

Grupo:

4CV12

A cargo del profesor:

PINEDA GUERRERO CARLOS

## Contenido

Introducción .....	3
Desarrollo .....	4
Creación de la primera máquina virtual.....	4
Compilación y ejecución del programa .....	11
Conclusiones .....	14

## Introducción

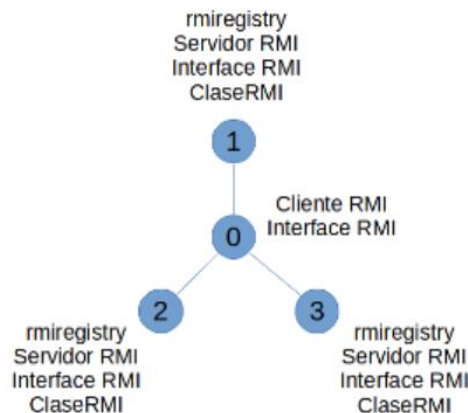
Para esta tarea se realizará la multiplicación de dos matrices cuadradas mediante el uso de objetos distribuidos usando Java RMI como fue visto en clase, con la diferencia de que las matrices a multiplicar estarán divididas en tres partes como se podrá visualizar en la siguiente imagen.



Para esto se considerará que la matriz B ha sido transpuesta y las submatrices de C quedaran de la siguiente forma antes de juntarse para formar la matriz resultado:

$$\begin{aligned} C1 &= A1 \times B1 & C2 &= A1 \times B2 & C3 &= A1 \times B3 \\ C4 &= A2 \times B1 & C5 &= A2 \times B2 & C6 &= A2 \times B3 \\ C7 &= A3 \times B1 & C8 &= A3 \times B2 & C9 &= A3 \times B3 \end{aligned}$$

Para realizar esta práctica se harán uso de 4 máquinas virtuales de Azure, donde una de ellas que será el nodo 0 actuara como cliente y otras 3 serán los nodos que actuaran como servidor a la hora de multiplicar matrices, la topología se podrá visualizar mejor de la siguiente manera:



El uso de objetos distribuidos mediante el uso de Java RMI permitirá a un objeto ejecutando en una máquina virtual para invocar métodos de un objeto en otra máquina virtual y obtener objetos existentes en dicha máquina remota, toda aplicación RMI se compone de un cliente y un servidor, el servidor crea algunos objetos remotos, crea referencias para hacerlos accesibles y espera a que el cliente los invoque, mientras que el cliente obtiene una referencia a objetos remotos en el servidor y los invoca.

## Desarrollo

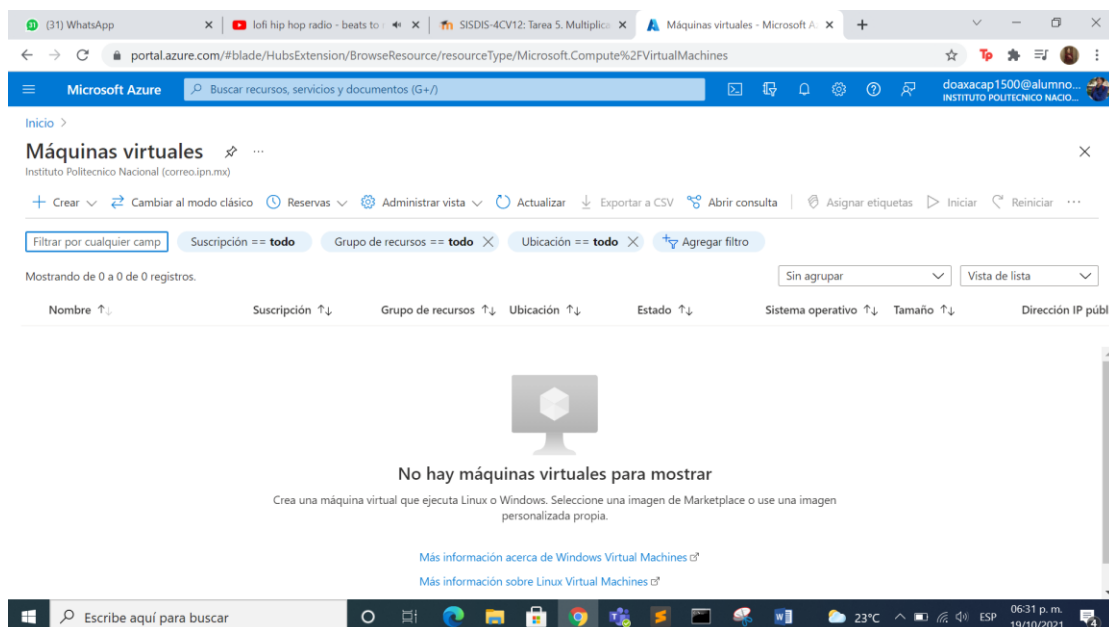
Lo primero que haremos antes del desarrollo ser tomar en cuenta las consideraciones para esta práctica. Estas son:

- 1) Las máquinas virtuales deberán llevar por nombre "JR2019630376-N" donde N es el número de nodo al que pertenecen y el numero después de JR es mi número de boleta.
- 2) La multiplicación deberá hacerse con matrices divididas en 3 partes, de manera en que al tener 3 nodos actuando como servidor, cada uno realizara los productos de 3 submatrices que conformaran a la matriz resultante de la multiplicación

Después de tener en cuenta esas consideraciones, se realizarán los cambios necesarios al programa hecho en clase para la multiplicación de matrices con RMI de manera local, se crearán las máquinas virtuales necesarias y se ejecutara el programa en ellas.

## Creación de la primera máquina virtual

En la siguiente captura podemos ver que por el momento no hay máquinas virtuales creadas, así que se creara una nueva presionando en el botón crear.



En la siguiente entramos a la creación de una nueva máquina virtual.

Microsoft Azure

Inicio > Máquinas virtuales >

### Crear una máquina virtual

**Datos básicos** | Discos | Redes | Administración | Opciones avanzadas | Etiquetas | Revisar y crear

Cree una máquina virtual que ejecuta Linux o Windows. Seleccione una imagen de Azure Marketplace o use una imagen personalizada propia. Complete la pestaña Conceptos básicos y, después, use Revisar y crear para aprovisionar una máquina virtual con parámetros predeterminados o bien revise cada una de las pestañas para personalizar la configuración. [Más información](#)

**Detalles del proyecto**

Seleccione la suscripción para administrar recursos implementados y los costes. Use los grupos de recursos como carpetas para organizar y administrar todos los recursos.

Suscripción \*

Grupo de recursos \*  [Crear nuevo](#)

**Detalles de instancia**

Nombre de máquina virtual \*

[Revisar y crear](#) < Anterior Siguiendo: Discos >

Creamos un nuevo grupo, en el que estarán todas las máquinas virtuales para esta práctica, este grupo se llamara Tarea5.

Microsoft Azure

Inicio > Máquinas virtuales >

### Crear una máquina virtual

**Datos básicos** | Discos | Redes | Administración | Opciones avanzadas | Etiquetas | Revisar y crear

Cree una máquina virtual que ejecuta Linux o Windows. Seleccione una imagen de Azure Marketplace o use una imagen personalizada propia. Complete la pestaña Conceptos básicos y, después, use Revisar y crear para aprovisionar una máquina virtual con parámetros predeterminados o bien revise cada una de las pestañas para personalizar la configuración. [Más información](#)

**Detalles del proyecto**

Seleccione la suscripción para administrar recursos implementados y los costes. Use los grupos de recursos como carpetas para organizar y administrar todos los recursos.

Suscripción \*

Grupo de recursos \*  [Crear nuevo](#)

**Detalles de instancia**

Nombre de máquina virtual \*

[Revisar y crear](#) < Anterior Siguiendo: Discos >

Nombramos a la maquina virtual JR2019630376-0, porque este sera el primer nodo, escogemos el Este de EE. UU. como región, la imagen de un servidor de Ubuntu 18 Gen 2.

Microsoft Azure

Inicio > Máquinas virtuales >

### Crear una máquina virtual

Suscripción \*

Grupo de recursos \*   
[Crear nuevo](#)

**Detalles de instancia**

Nombre de máquina virtual \*

Región \*

Opciones de disponibilidad

Imagen \*   
[Ver todas las imágenes](#) | [Configurar la generación de máquinas virtuales](#)

Instancia de Azure de acceso puntual ☐

Tamaño \*   
[Ver todos los tamaños](#)

[Revisar y crear](#) < Anterior Siguiente: Discos >

En tamaño escogemos 2 GB de RAM, perteneciente a la familia B.

Microsoft Azure

Inicio > Máquinas virtuales >

### Crear una máquina virtual

Nombre de máquina virtual \*

Región \*

Opciones de disponibilidad

Imagen \*   
[Ver todas las imágenes](#) | [Configurar la generación de máquinas virtuales](#)

Instancia de Azure de acceso puntual ☐

Tamaño \*   
[Ver todos los tamaños](#)

**Cuenta de administrador**

Tipo de autenticación ☒ Clave pública SSH ☐ Contraseña

[Revisar y crear](#) < Anterior Siguiente: Discos >

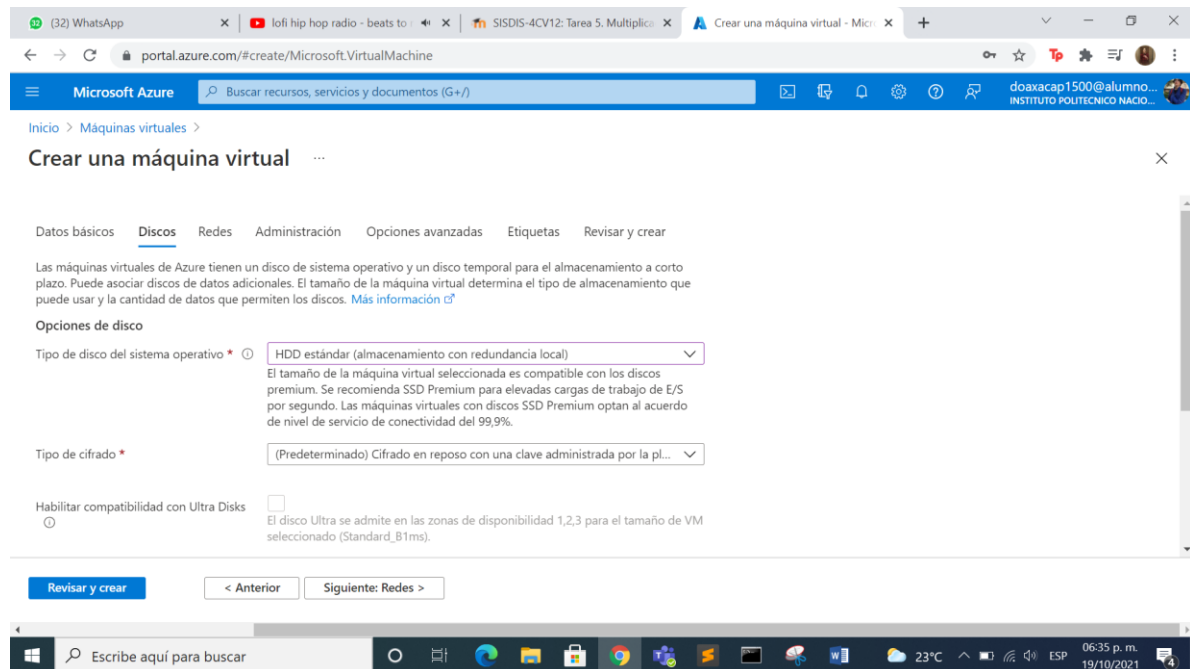
Posteriormente podemos ver que escogeremos la opción de contraseña como tipo de autenticación, usamos Ubuntu como nombre de usuario y escribimos la contraseña para este usuario.

Microsoft Azure portal showing the 'Crear una máquina virtual' (Create a virtual machine) page. The page displays the configuration for the virtual machine, including the authentication type (Contraseña), username (Ubuntu), password, and inbound port rules (SSH (22)).

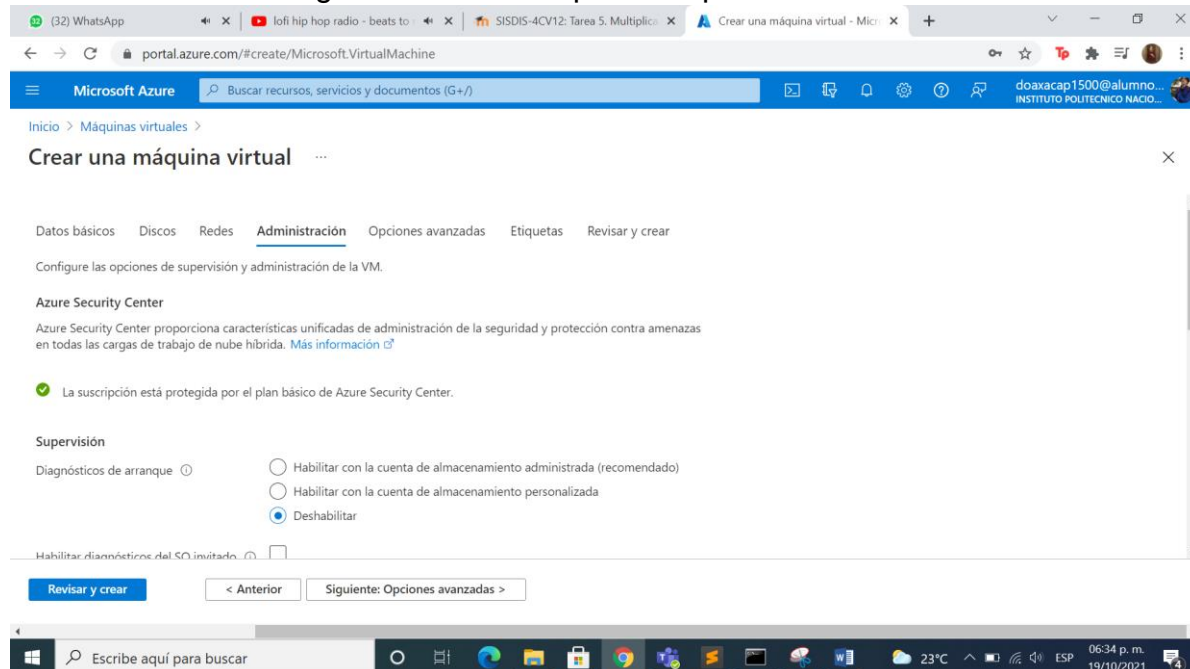
En la siguiente captura, podemos ver que se deja abierto el puerto 22 para la conexión por medio de SSH.

Microsoft Azure portal showing the 'Crear una máquina virtual' (Create a virtual machine) page. The page displays the configuration for the virtual machine, including the authentication type (Contraseña), username (Ubuntu), password, and inbound port rules (SSH (22)). A warning message indicates that allowing SSH (22) access from all IP addresses is not recommended for production environments.

En la parte de discos, escogemos como el tipo de disco del sistema operativo el HDD estándar, pues es el de menor costo y nos permitirá ahorrar.

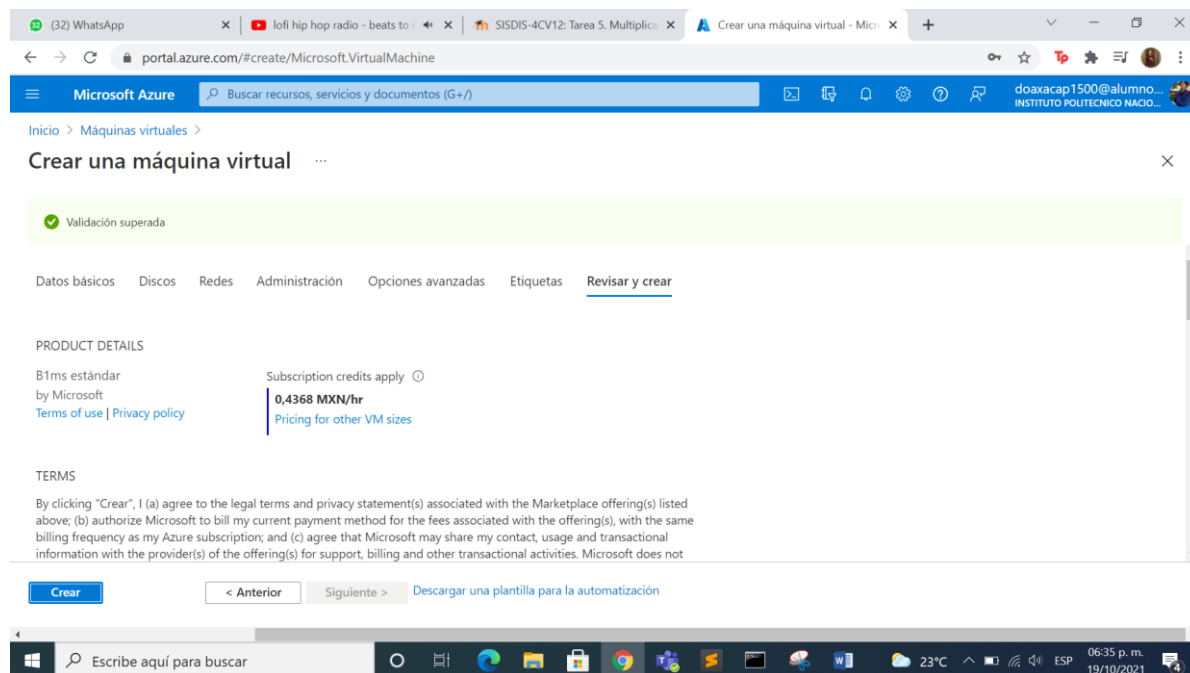


Deshabilitamos el diagnostico de arranque en la parte de administración.

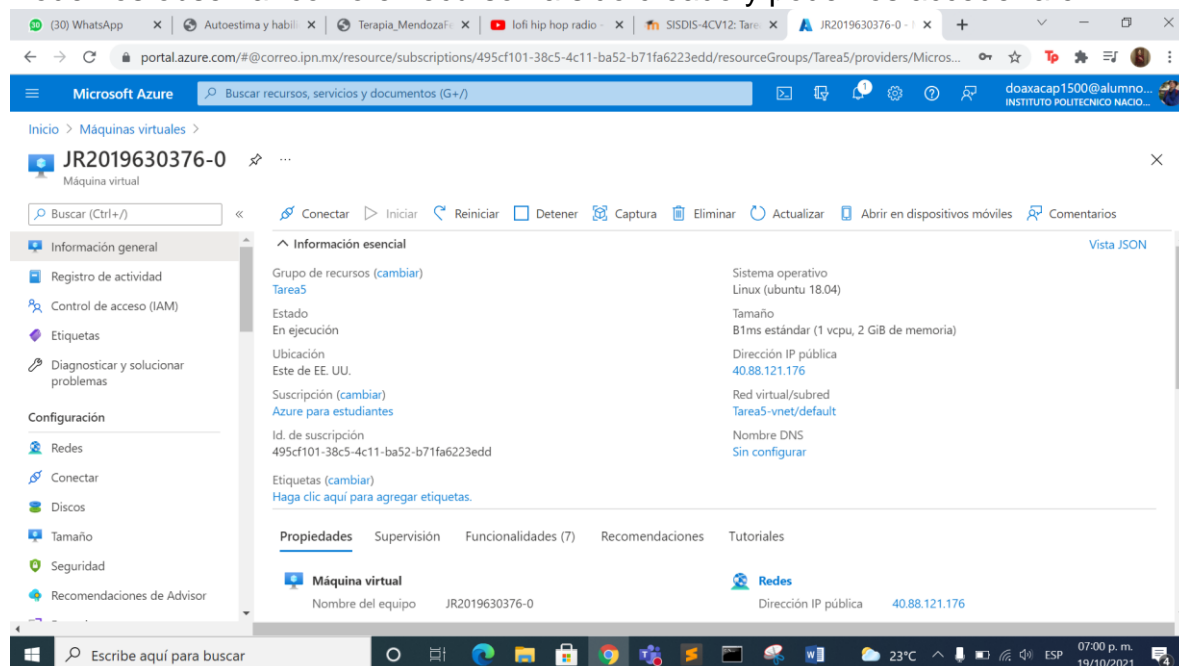




Le damos en revisar y crear y podemos observar en la siguiente captura como aparece que la validación fue superada.



Podemos observar como el recurso ha sido creado y podemos acceder a él.



Tras la creación del resto de las máquinas virtuales, 4 en total, una para cada nodo y todas ellas en el mismo grupo de recursos, con su respectivo nombre que indica su nodo.

**Máquinas virtuales**

Instituto Politécnico Nacional (correo.ipn.mx)

+ Crear ▾ Cambiar al modo clásico Reservar ▾ Administrar vista ▾ Actualizar Exportar a CSV Abrir consulta Asignar etiquetas Iniciar Reiniciar ⋮

Filtrar por cualquier campo Suscripción == todo Grupo de recursos == todo Ubicación == todo Agregar filtro

Mostrando de 1 a 4 de 4 registros. Sin agrupar Vista de lista

<input type="checkbox"/> Nombre ↑↓	Suscripción ↑↓	Grupo de recursos ↑↓	Ubicación ↑↓	Estado ↑↓	Sistema operativo ↑↓	Tamaño ↑↓	Dirección IP públ..
<input type="checkbox"/> JR2019630376-0	Azure para estudiantes	Tarea5	Este de EE. UU.	En ejecución	Linux	Standard_B1ms	40.88.121.176
<input type="checkbox"/> JR2019630376-1	Azure para estudiantes	Tarea5	Este de EE. UU.	En ejecución	Linux	Standard_B1ms	52.188.125.96
<input type="checkbox"/> JR2019630376-2	Azure para estudiantes	Tarea5	Este de EE. UU.	En ejecución	Linux	Standard_B1ms	52.249.251.56
<input type="checkbox"/> JR2019630376-3	Azure para estudiantes	Tarea5	Este de EE. UU.	En ejecución	Linux	Standard_B1ms	20.185.194.19

< Anterior Página 1 de 1 Siguiente >

Después de la creación de las máquinas virtuales entramos a ellas usando SSH.

```

C:\Users\tdwda\OneDrive\Escritorio\RM\I>ssh Ubuntu@40.88.121.176
The authenticity of host '40.88.121.176 (40.88.121.176)' can't be established.
ECDSA key fingerprint is SHA256:9EPDOHT/k1010o2ICiknZ7IJ2+Ik3pFXS+wQyLe3jqI.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '40.88.121.176' (ECDSA) to the list of known hosts.
Ubuntu@40.88.121.176's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1061-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Wed Oct 20 00:26:53 UTC 2021

System load:  0.01          Processes:    97
Usage of /:   4.6% of 28.9GB Users logged in:  0
Memory usage: 9%           IP address for eth0: 10.0.0.4
Swap usage:   0%

0 updates can be applied immediately.

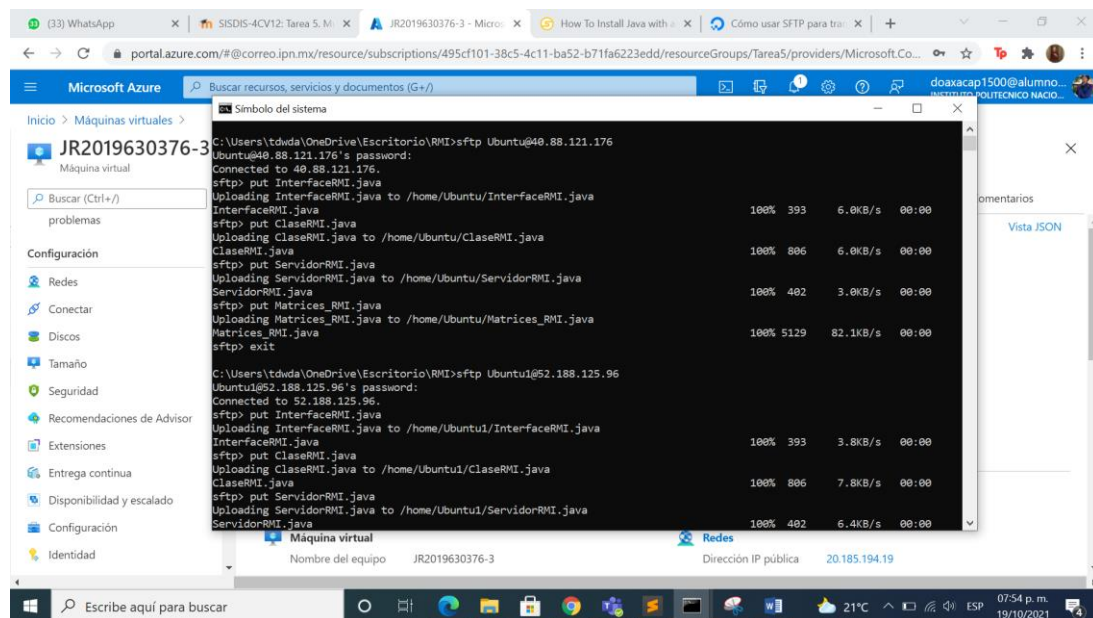
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

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.

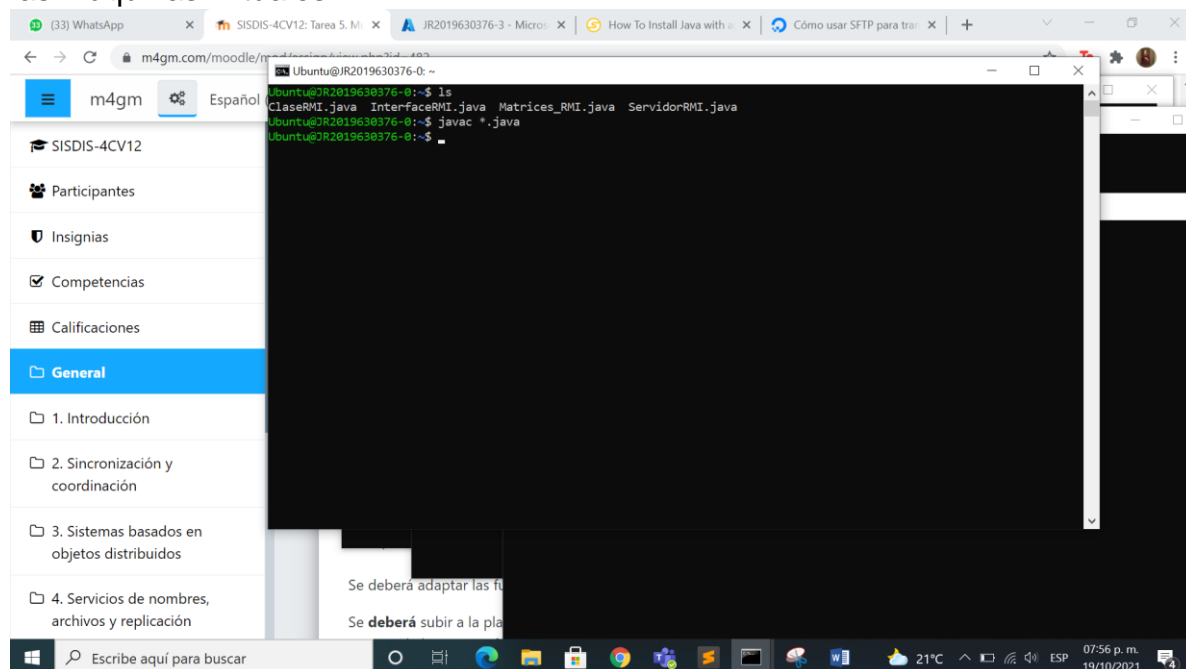
Ubuntu@JR2019630376-0:~$
  
```

Pasamos las clases a las máquinas virtuales mediante sftp, en este caso no necesitaremos todas ellas, pero las pasaremos por si acaso y ya en la ejecución usaremos las que ocupamos.

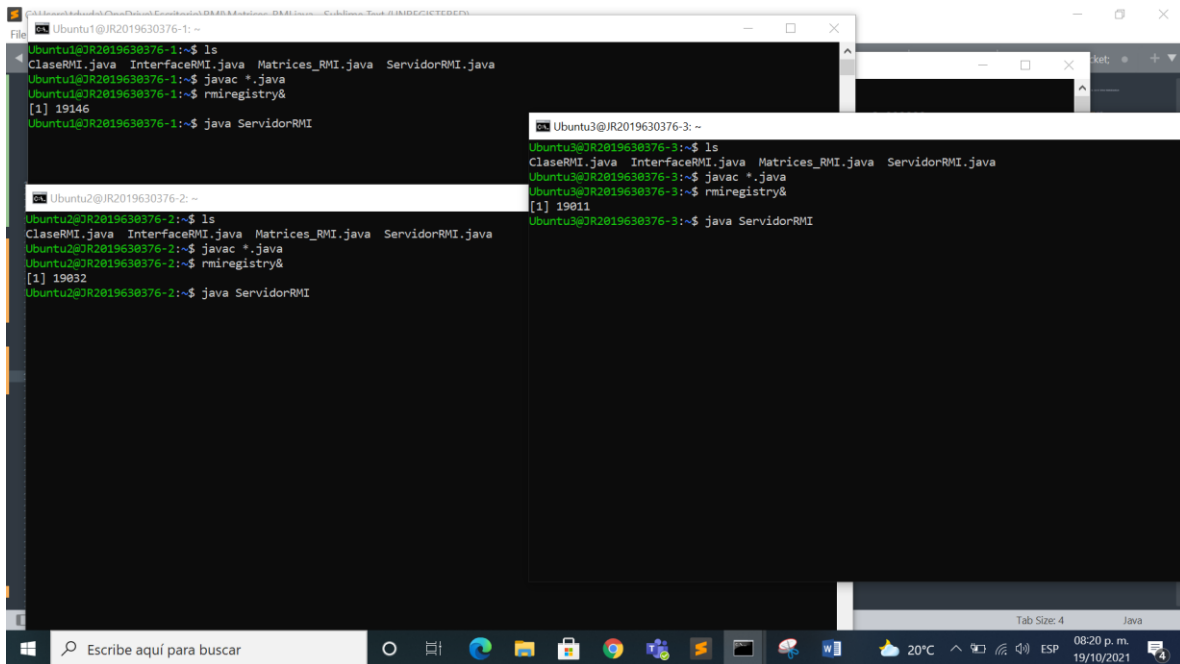


## Compilación y ejecución del programa

Instalamos el JDK y el JRE en la máquina virtual usando los comandos `sudo apt update`, `sudo apt install default-jre` y `sudo apt install default-jdk`, posteriormente compilamos todos los archivos Java que pasamos. Hacemos esto en cada una de las máquinas virtuales.



Posteriormente empezamos con lo que será la ejecución del programa, para esto tendremos que en las 3 máquinas virtuales en las que se encontraran los nodos usaremos el `rmiregistry` y después ejecutaremos el `ServidorRMI`, ese se quedara en ejecución esperando al cliente.



```
Ubuntu1@JR2019630376-1:~$ ls
ClaseRMI.java  InterfaceRMI.java  Matrices_RMI.java  ServidorRMI.java
Ubuntu1@JR2019630376-1:~$ javac *.java
Ubuntu1@JR2019630376-1:~$ rmiregistry&
[1] 19146
Ubuntu1@JR2019630376-1:~$ java ServidorRMI

Ubuntu2@JR2019630376-2:~$ ls
ClaseRMI.java  InterfaceRMI.java  Matrices_RMI.java  ServidorRMI.java
Ubuntu2@JR2019630376-2:~$ javac *.java
Ubuntu2@JR2019630376-2:~$ rmiregistry&
[1] 19832
Ubuntu2@JR2019630376-2:~$ java ServidorRMI

Ubuntu3@JR2019630376-3:~$ ls
ClaseRMI.java  InterfaceRMI.java  Matrices_RMI.java  ServidorRMI.java
Ubuntu3@JR2019630376-3:~$ javac *.java
Ubuntu3@JR2019630376-3:~$ rmiregistry&
[1] 19011
Ubuntu3@JR2019630376-3:~$ java ServidorRMI
```

Para el nodo 0, donde se encontrará el cliente, simplemente ejecutaremos la clase `Matrices_RMI`, que será la clase cliente donde se encuentra el lookup y pasaremos como parámetro `N`, es decir, el número de elementos que tendrá la matriz tanto en filas como en columnas (Es una matriz cuadrada). Si `N = 9`, al terminar la multiplicación de la matriz por medio de objetos distribuidos se imprimirán las matrices y el checksum, en caso contrario solo se imprimirá el checksum.

En esta captura de pantalla podemos ver la impresión de las matrices A y B cuando N es igual a 9.

```

Ubuntu1@JR2019630376-1:~$ ls
ClaseRMI.java  InterfaceRMI.java
Ubuntu1@JR2019630376-1:~$ javac *.java
[1] 19146
Ubuntu1@JR2019630376-1:~$ rmiregistry
ClaseRMI.class  InterfaceRMI.class  Matrices_RMI.class  ServidorRMI.class
Ubuntu1@JR2019630376-0:~$ java Matrices_RMI 9
Matriz A:
0.000000 1.000000 2.000000 3.000000 4.000000 5.000000 6.000000 7.000000 8.000000
4.000000 5.000000 6.000000 7.000000 8.000000 9.000000 10.000000 11.000000 12.000000
8.000000 9.000000 10.000000 11.000000 12.000000 13.000000 14.000000 15.000000 16.000000
12.000000 13.000000 14.000000 15.000000 16.000000 17.000000 18.000000 19.000000 20.000000
16.000000 17.000000 18.000000 19.000000 20.000000 21.000000 22.000000 23.000000 24.000000
20.000000 21.000000 22.000000 23.000000 24.000000 25.000000 26.000000 27.000000 28.000000
24.000000 25.000000 26.000000 27.000000 28.000000 29.000000 30.000000 31.000000 32.000000
28.000000 29.000000 30.000000 31.000000 32.000000 33.000000 34.000000 35.000000 36.000000
32.000000 33.000000 34.000000 35.000000 36.000000 37.000000 38.000000 39.000000 40.000000

Matriz B:
0.000000 -1.000000 -2.000000 -3.000000 -4.000000 -5.000000 -6.000000 -7.000000 -8.000000
4.000000 3.000000 2.000000 1.000000 0.000000 -1.000000 -2.000000 -3.000000 -4.000000
8.000000 7.000000 6.000000 5.000000 4.000000 3.000000 2.000000 1.000000 0.000000
12.000000 11.000000 10.000000 9.000000 8.000000 7.000000 6.000000 5.000000 4.000000
16.000000 15.000000 14.000000 13.000000 12.000000 11.000000 10.000000 9.000000 8.000000
20.000000 19.000000 18.000000 17.000000 16.000000 15.000000 14.000000 13.000000 12.000000
24.000000 23.000000 22.000000 21.000000 20.000000 19.000000 18.000000 17.000000 16.000000
28.000000 27.000000 26.000000 25.000000 24.000000 23.000000 22.000000 21.000000 20.000000
32.000000 31.000000 30.000000 29.000000 28.000000 27.000000 26.000000 25.000000 24.000000
  
```

En esta captura de pantalla podemos ver la matriz C, resultante de la multiplicación de matrices A y B, así como el checksum de esta matriz.

El **checksum = 194400.0** cuando N = 9.

```

Ubuntu1@JR2019630376-1:~$ ls
ClaseRMI.java  InterfaceRMI.java
Ubuntu1@JR2019630376-1:~$ javac *.java
[1] 19146
Ubuntu1@JR2019630376-1:~$ rmiregistry
ClaseRMI.class  InterfaceRMI.class  Matrices_RMI.class  ServidorRMI.class
Ubuntu1@JR2019630376-0:~$ java Matrices_RMI 9
Matriz A:
0.000000 1.000000 2.000000 3.000000 4.000000 5.000000 6.000000 7.000000 8.000000
4.000000 5.000000 6.000000 7.000000 8.000000 9.000000 10.000000 11.000000 12.000000
8.000000 9.000000 10.000000 11.000000 12.000000 13.000000 14.000000 15.000000 16.000000
12.000000 13.000000 14.000000 15.000000 16.000000 17.000000 18.000000 19.000000 20.000000
16.000000 17.000000 18.000000 19.000000 20.000000 21.000000 22.000000 23.000000 24.000000
20.000000 21.000000 22.000000 23.000000 24.000000 25.000000 26.000000 27.000000 28.000000
24.000000 25.000000 26.000000 27.000000 28.000000 29.000000 30.000000 31.000000 32.000000
28.000000 29.000000 30.000000 31.000000 32.000000 33.000000 34.000000 35.000000 36.000000
32.000000 33.000000 34.000000 35.000000 36.000000 37.000000 38.000000 39.000000 40.000000

Matriz B:
0.000000 -1.000000 -2.000000 -3.000000 -4.000000 -5.000000 -6.000000 -7.000000 -8.000000
4.000000 3.000000 2.000000 1.000000 0.000000 -1.000000 -2.000000 -3.000000 -4.000000
8.000000 7.000000 6.000000 5.000000 4.000000 3.000000 2.000000 1.000000 0.000000
12.000000 11.000000 10.000000 9.000000 8.000000 7.000000 6.000000 5.000000 4.000000
16.000000 15.000000 14.000000 13.000000 12.000000 11.000000 10.000000 9.000000 8.000000
20.000000 19.000000 18.000000 17.000000 16.000000 15.000000 14.000000 13.000000 12.000000
24.000000 23.000000 22.000000 21.000000 20.000000 19.000000 18.000000 17.000000 16.000000
28.000000 27.000000 26.000000 25.000000 24.000000 23.000000 22.000000 21.000000 20.000000
32.000000 31.000000 30.000000 29.000000 28.000000 27.000000 26.000000 25.000000 24.000000

Matriz C:
816.000000 780.000000 744.000000 708.000000 672.000000 636.000000 600.000000 564.000000 528.000000
1392.000000 1320.000000 1248.000000 1176.000000 1104.000000 1032.000000 960.000000 888.000000 816.000000
1968.000000 1860.000000 1752.000000 1644.000000 1536.000000 1428.000000 1320.000000 1212.000000 1104.000000
2544.000000 2400.000000 2256.000000 2112.000000 1968.000000 1824.000000 1680.000000 1536.000000 1392.000000
3120.000000 2940.000000 2760.000000 2580.000000 2400.000000 2220.000000 2040.000000 1860.000000 1680.000000
3696.000000 3480.000000 3264.000000 3048.000000 2832.000000 2616.000000 2400.000000 2184.000000 1968.000000
4272.000000 4020.000000 3768.000000 3516.000000 3264.000000 3012.000000 2760.000000 2508.000000 2256.000000
4848.000000 4560.000000 4272.000000 3984.000000 3696.000000 3408.000000 3120.000000 2832.000000 2544.000000
5424.000000 5100.000000 4776.000000 4452.000000 4128.000000 3804.000000 3480.000000 3156.000000 2832.000000

Checksum: 194400.0
Ubuntu1@JR2019630376-0:~$
  
```

En esta siguiente captura de pantalla tenemos el resultado del checksum de la matriz cuando  $N = 3000$ , este proceso fue bastante más tardado de realizar, aun siendo que se estaba implementando un programa distribuido para la multiplicación.

El **checksum = 9.916425922659735E17** cuando  $N = 3000$ .

```

Ubuntu1@JR2019630376-1:~$ ls
ClaseRMI.java InterfaceRMI.java
Ubuntu1@JR2019630376-1:~$ javac *.java
Ubuntu1@JR2019630376-1:~$ rmiregist
[1] 19146
Ubuntu1@JR2019630376-1:~$ java Serv
20.000000 19.000000 18.000000 17.000000 16.000000 15.000000 14.000000 13.000000 12.000000
24.000000 23.000000 22.000000 21.000000 20.000000 19.000000 18.000000 17.000000 16.000000
28.000000 27.000000 26.000000 25.000000 24.000000 23.000000 22.000000 21.000000 20.000000
32.000000 31.000000 30.000000 29.000000 28.000000 27.000000 26.000000 25.000000 24.000000

Matriz C:
816.000000 780.000000 744.000000 708.000000 672.000000 636.000000 600.000000 564.000000 528.000000
1392.000000 1320.000000 1248.000000 1176.000000 1104.000000 1032.000000 960.000000 888.000000 816.000000
1968.000000 1860.000000 1752.000000 1644.000000 1536.000000 1428.000000 1320.000000 1212.000000 1104.000000
2544.000000 2400.000000 2256.000000 2112.000000 1968.000000 1824.000000 1680.000000 1536.000000 1392.000000
3120.000000 2840.000000 2760.000000 2580.000000 2400.000000 2220.000000 2040.000000 1860.000000 1680.000000
3696.000000 3480.000000 3264.000000 3048.000000 2832.000000 2616.000000 2400.000000 2184.000000 1968.000000
4272.000000 4020.000000 3768.000000 3516.000000 3264.000000 3012.000000 2760.000000 2508.000000 2256.000000
4848.000000 4560.000000 4272.000000 3984.000000 3696.000000 3408.000000 3120.000000 2832.000000 2544.000000
5424.000000 5100.000000 4776.000000 4452.000000 4128.000000 3804.000000 3480.000000 3156.000000 2832.000000

Checksum: 194400.0
Ubuntu@JR2019630376-0:~$ java Matrices_RMI 3000

Checksum: 9.916425922659735E17
Ubuntu@JR2019630376-0:~$
  
```

## Conclusiones

Esta práctica fue de ayuda para poner a prueba los conocimientos adquiridos en clase respecto al uso de objetos distribuidos mediante Java RMI, si bien se usó un problema familiar con algo parecido en clase y que incluso se había resuelto de otra manera en la tarea 3, esto ayudo a que el enfoque principal para esta tarea fuera el uso de RMI en máquinas virtuales, pues las modificaciones para la forma de resolución en la multiplicación de matrices no variaban tanto. Considero que fue una práctica entretenida que también se sintió un poco más rápida que las anteriores debido a la mayor familiaridad con la creación de máquinas virtuales en Azure que se ha ido adquiriendo en clases, además de que clarificaba de una manera practica el uso de los objetos distribuidos.