



Instituto Politécnico Nacional
Escuela Superior de Cómputo



González Bocio Erik Alexander

4CV11

Pineda Guerrero Carlos

Desarrollo de Sistemas Distribuidos

Tarea 2: Multiplicación de matrices distribuida
utilizando paso de mensajes

Desarrollo

Para elaborar esta práctica hicimos uso de ChatGPT, debido a los problemas que tiene el chatbot con el número de tokens como respuesta opte con ir pidiéndole paso a paso lo que debía hacer el problema, primero le pido hacer el nodo 0 con el que salieron algunos métodos usados allí, que se solicitaron después, luego con el nodo 1 se hizo lo mismo, como el nodo 2 y 3 son lo mismo que el nodo 1 solo se adapto a lo que ya se tenía. A continuación el chat con ChatGPT.

Primero comencé pidiéndole que me diera el esqueleto del programa

Erik Gonzalez

Programa en java que multiplicara matrices, trabajara con 3 matrices A, B y C de tipo double, con tamaño N, en este caso N=12, este pedira un valor entero, el programa preguntara al usuario "Con que nodo deseas trabajar", y aqui hay 4 casos, si coloca 0,1,2,3

Aun no multipliques matrices ni inicialices las matrices

CHATGPT

Aquí te dejo un código en Java que te permitirá multiplicar matrices de tamaño N = 12 y trabajar con tres matrices A, B y C de tipo double. Además, el programa preguntará al usuario con qué nodo desea trabajar y habrá cuatro casos dependiendo del número ingresado (0, 1, 2 o 3). Ten en cuenta que en este código aún no se inicializan las matrices ni se realiza la multiplicación de las mismas.

```
import java.util.Scanner;
```

```
public class MultiplicacionMatrices {
```

```
    public static void main(String[] args) {
```

```
        int n = 12;
```

```
        double[][] A = new double[n][n];
```

```
        double[][] B = new double[n][n];
```

```
        double[][] C = new double[n][n];
```

```
        Scanner sc = new Scanner(System.in);
```

```
        System.out.print("Con qué nodo deseas trabajar? ");
```

```
        int nodo = sc.nextInt();
```

```

switch (nodo) {
    case 0:
        System.out.println("Trabajando con el nodo 0");
        break;
    case 1:
        System.out.println("Trabajando con el nodo 1");
        break;
    case 2:
        System.out.println("Trabajando con el nodo 2");
        break;
    case 3:
        System.out.println("Trabajando con el nodo 3");
        break;
    default:
        System.out.println("El nodo ingresado no es válido");
        break;
}
}
}

```

Recuerda que para que el programa pueda realizar la multiplicación de las matrices, primero debes inicializarlas con los valores correspondientes.

Luego le pedí que me diera el código del case 0, aquí le dije todo lo que se tenía que hacer dentro del case omitiendo todo lo demás del código.

Erik González

Ahora dame el caso 0, aquí se inicializan las matrices, A se inicializa como $A[i][j]=2*i+j$, B como $B[i][j]=3*i-j$, se transpone B en la misma matriz B, el programa se conectará a 3 servidores, por lo que necesito que sea multithread, usando la clase Worker que extienda a Thread y tenga un método run, dichos servidores tienen ip localhost tu da los puertos, A y la transpuesta de B se dividirán en 3 partes de igual tamaño y se guardarán en matrices A1, A2, etc, de igual manera con B, al primer servidor se envía la primera parte de A y todas las partes de B, el segundo la segunda parte de A y todas las de B, y el tercer servidor la tercera parte de A y todas las partes de B, por lo

que en el worker solo se pasa una parte de A y las partes de B, se esperara recibir de cada servidor 3 matrices, del primero recibira de C1 a C3, del segundo de C4 a C6 y del tercero de C7 a C9, estas corresponden a las 9 partes de la matriz C, por lo que con este se hara el checksum, se desplegara, en el caso de que N=12 se imprimiran A,B y C pero en el caso de que N=3000 solo desplegara el checksum de la matriz C, el checksum sera de tipo double.

Procura darme solo el caso, omite las librerias y la declaracion de las matrices y N

CHATGPT

¡Claro! Aquí te dejo el código para el caso 0, ten en cuenta que debes agregar las librerías necesarias y las declaraciones de las matrices y N:

```
// Inicialización de la matriz A
```

```
for (int i = 0; i < N; i++) {  
    for (int j = 0; j < N; j++) {  
        A[i][j] = 2 * i + j;  
    }  
}
```

```
// Inicialización de la matriz B y su transposición
```

```
for (int i = 0; i < N; i++) {  
    for (int j = 0; j < N; j++) {  
        B[i][j] = 3 * i - j;  
    }  
}  
  
for (int i = 0; i < N; i++) {  
    for (int j = i + 1; j < N; j++) {  
        double temp = B[i][j];  
        B[i][j] = B[j][i];  
        B[j][i] = temp;  
    }  
}
```

```
// División de las matrices A y B en 3 partes iguales
```

```

int partSize = N / 3;

double[][] A1 = new double[partSize][N];
double[][] A2 = new double[partSize][N];
double[][] A3 = new double[partSize][N];
double[][][] BParts = new double[3][partSize][N];
for (int i = 0; i < partSize; i++) {
    A1[i] = Arrays.copyOfRange(A, i, i + 1)[0];
    A2[i] = Arrays.copyOfRange(A, i + partSize, i + partSize + 1)[0];
    A3[i] = Arrays.copyOfRange(A, i + partSize * 2, i + partSize * 2 + 1)[0];
    BParts[0][i] = Arrays.copyOfRange(B, i, i + 1)[0];
    BParts[1][i] = Arrays.copyOfRange(B, i + partSize, i + partSize + 1)[0];
    BParts[2][i] = Arrays.copyOfRange(B, i + partSize * 2, i + partSize * 2 + 1)[0];
}

```

// Creación de los Workers y conexión a los servidores

```

Worker[] workers = new Worker[3];
for (int i = 0; i < 3; i++) {
    workers[i] = new Worker("localhost", 5000 + i, AParts[i], BParts);
    workers[i].start();
}

```

// Espera de resultados de los servidores

// Espera de resultados de los servidores

```

double[][] C1 = new double[partSize][N];
double[][] C2 = new double[partSize][N];
double[][] C3 = new double[partSize][N];
double[][] C4 = new double[partSize][N];
double[][] C5 = new double[partSize][N];
double[][] C6 = new double[partSize][N];

```

```

double[][] C7 = new double[partSize][N];
double[][] C8 = new double[partSize][N];
double[][] C9 = new double[partSize][N];
for (int i = 0; i < 3; i++) {
    try {
        workers[i].join();
    }
    if (i == 0)

```

Erik Gonzalez

continua el código desde la espera de resultados de los servidores, omite lo anterior

ChatGPT

```

/ Espera de resultados de los servidores
double[][] C1 = new double[partSize][N];
double[][] C2 = new double[partSize][N];
double[][] C3 = new double[partSize][N];
double[][] C4 = new double[partSize][N];
double[][] C5 = new double[partSize][N];
double[][] C6 = new double[partSize][N];
double[][] C7 = new double[partSize][N];
double[][] C8 = new double[partSize][N];
double[][] C9 = new double[partSize][N];

```

```

for (int i = 0; i < 3; i++) {
    try {
        workers[i].join();
    }
    if (i == 0) {
        C1 = workers[i].getC1();
        C2 = workers[i].getC2();
        C3 = workers[i].getC3();
    } else if (i == 1) {

```

```

C4 = workers[i].getC();
C5 = workers[i].getC2();
C6 = workers[i].getC3();
} else {
C7 = workers[i].getC();
C8 = workers[i].getC2();
C9 = workers[i].getC3();
}
} catch (InterruptedException e) {
e.printStackTrace();
}
}

```

```

// Unión de las matrices C
double[][] C = new double[N][N];
for (int i = 0; i < partSize; i++) {
for (int j = 0; j < N; j++) {
C[i][j] = C1[i][j];
C[i + partSize][j] = C2[i][j];
C[i + partSize * 2][j] = C3[i][j];
C[i + partSize * 3][j] = C4[i][j];
C[i + partSize * 4][j] = C5[i][j];
C[i + partSize * 5][j] = C6[i][j];
C[i + partSize * 6][j] = C7[i][j];
C[i + partSize * 7][j] = C8[i][j];
C[i + partSize * 8][j] = C9[i][j];
}
}
}

```

```
// Cálculo del checksum de la matriz C
```

```
double checksum = 0;
for (int i = 0; i < N; i++) {
    for (int j = 0; j < N; j++) {
        checksum += C[i][j];
    }
}
```

```
if (N == 12) {
    // Imprimir matrices A, B y C
    System.out.println("Matrix A:");
    printMatrix(A);
    System.out.println("Matrix B:");
    printMatrix(B);
    System.out.println("Matrix C:");
    printMatrix(C);
} else if (N == 3000) {
    // Imprimir checksum de la matriz C
    System.out.println("Checksum: " + checksum);
}
```

Erik Gonzalez

continua con el código desde Imprimir checksum de la matriz C

ChatGPT

```
// Cálculo del checksum de la matriz C
double checksum = 0;
for (int i = 0; i < N; i++) {
    for (int j = 0; j < N; j++) {
        checksum += C[i][j];
    }
}
```



```

}

// Impresión de resultados
if (N == 12) {
    System.out.println("Matriz A:");
    printMatrix(A);
    System.out.println("Matriz B:");
    printMatrix(B);
    System.out.println("Matriz C:");
    printMatrix(C);
}

System.out.println("Checksum de la matriz C: " + checksum);

// Método para imprimir una matriz
public static void printMatrix(double[][] matrix) {
    for (int i = 0; i < matrix.length; i++) {
        for (int j = 0; j < matrix[0].length; j++) {
            System.out.print(matrix[i][j] + " ");
        }
        System.out.println();
    }
    System.out.println();
}

```

En este punto había partes donde el código que me dio el chat hacía uso de funciones, pero por el tema de los tokens o no me los dio completos o simplemente no me los dio, por lo que se los pedí aparte.

Erik Gonzalez

muertrame los metodos printMatrix y worker

ChatGPT

El código del método printMatrix podría ser algo como esto:

```
public static void printMatrix (double[][] matriz) {  
    for (int i = 0; i < matriz.length; i++) {  
        for (int j = 0; j < matriz[0].length; j++) {  
            System.out.print(matriz[i][j] + "\\t");  
        }  
        System.out.println();  
    }  
    System.out.println();  
}
```

Este método recibe como parámetro una matriz de números reales y la imprime por pantalla, fila por fila.

El código del método Worker podría ser algo como esto:

```
import java.io.*;  
import java.net.*;  
  
public class Worker extends Thread {  
    private String host;  
    private int port;  
    private double[][] A;  
    private double[][][] BParts;  
    private double[][] C;  
  
    public Worker(String host, int port, double[][] A, double[][][] BParts) {  
        this.host = host;  
        this.port = port;  
        this.A = A;  
        this.BParts = BParts;  
        this.C = new double[A.length][BParts[0][0].length];  
    }  
}
```

```

    }

    public void run() {
        try (Socket socket = new Socket(host, port);
            ObjectOutputStream out = new ObjectOutputStream(socket.getOutputStream());
            ObjectInputStream in = new ObjectInputStream(socket.getInputStream())) {
            // Envío de los datos al servidor
            out.writeObject(A);
            out.writeObject(BParts);

            // Recepción del resultado del servidor
            C = (double[][] in.readObject());
        } catch (IOException | ClassNotFoundException e) {
            e.printStackTrace();
        }
    }

    public double[][] getC() {
        return C;
    }
}

```

Este código define una clase Worker que extiende de la clase Thread. Cada instancia de la clase Worker representa un hilo de ejecución que se conecta a un servidor para realizar una multiplicación de matrices. Los datos de entrada son la matriz A y un arreglo de matrices BParts. Los datos de salida son la matriz C.

El método run es el método que se ejecuta cuando se llama al método start de la instancia de la clase Worker. En este método, se establece la conexión con el servidor y se envían los datos de entrada. Después, se espera a recibir la matriz C resultante del servidor.

El método getC devuelve la matriz C calculada por el servidor.

El principal problema aquí fue que a pesar de que fui muy puntual en que las partes de B las manejara como B1, B2 y B3 el chat las manejaba como un arreglo tridimensional por lo que tuve que pedirle que lo cambiara.

Erik Gonzalez

cambia el worker, necesito que en vez de Bparts[][][] pida B1, B2 y B3 ademas que sea static class

ChatGPT

Por supuesto, aquí te muestro la nueva versión del Worker con las modificaciones que solicitaste:

```
public static class Worker extends Thread {  
    private String host;  
    private int port;  
    private double[][] A;  
    private double[][] B1;  
    private double[][] B2;  
    private double[][] B3;  
    private double[][] C;  
  
    public Worker(String host, int port, double[][] A, double[][] B1, double[][] B2, double[][] B3) {  
        this.host = host;  
        this.port = port;  
        this.A = A;  
        this.B1 = B1;  
        this.B2 = B2;  
        this.B3 = B3;  
    }  
  
    public double[][] getC() {  
        return this.C;  
    }  
}
```

```

@Override

public void run() {

    try {

        Socket socket = new Socket(host, port);

        ObjectOutputStream out = new ObjectOutputStream(socket.getOutputStream());

        ObjectInputStream in = new ObjectInputStream(socket.getInputStream());


        // Envío de A y B al servidor

        out.writeObject(A);

        out.writeObject(B1);

        out.writeObject(B2);

        out.writeObject(B3);


        // Recepción de C del servidor

        this.C = (double[][][]) in.readObject();


        // Cierre de conexiones

        out.close();

        in.close();

        socket.close();


    } catch (Exception e) {

        e.printStackTrace();

    }

}
}

```

En esta nueva versión del Worker se han agregado los arreglos B1, B2 y B3 como parámetros de entrada en el constructor, en lugar del arreglo BParts[][][] anterior. Además, se ha agregado la palabra clave "static" para indicar que se trata de una clase estática.

Una vez arreglado eso le pedí que me hiciera el uso de los workers en el nodo 0

Erik Gonzalez

ahora dame la Creación de los Workers y conexión a los servidores en el case 0 con los cambios para que funcione con esta nueva funcion worker, recuerda que en la clase worker se debe pasar la parte de A y todas las partes de B

ChatGPT

Aquí está el código actualizado para la creación de los Workers y conexión a los servidores en el case 0, usando la nueva función Worker que acepta B1, B2, y B3 como argumentos:

case 0:

```
// División de las matrices A y B en 3 partes iguales

int partSize = N / 3;

double[][] A1 = new double[partSize][N];
double[][] A2 = new double[partSize][N];
double[][] A3 = new double[partSize][N];
double[][] B1 = new double[partSize][N];
double[][] B2 = new double[partSize][N];
double[][] B3 = new double[partSize][N];

for (int i = 0; i < partSize; i++) {
    A1[i] = Arrays.copyOfRange(A, i, i + 1)[0];
    A2[i] = Arrays.copyOfRange(A, i + partSize, i + partSize + 1)[0];
    A3[i] = Arrays.copyOfRange(A, i + partSize * 2, i + partSize * 2 + 1)[0];
    B1[i] = Arrays.copyOfRange(B, i, i + 1)[0];
    B2[i] = Arrays.copyOfRange(B, i + partSize, i + partSize + 1)[0];
    B3[i] = Arrays.copyOfRange(B, i + partSize * 2, i + partSize * 2 + 1)[0];
}

// Creación de los Workers y conexión a los servidores

Worker[] workers = new Worker[3];

for (int i = 0; i < 3; i++) {
```

```

switch (i) {
    case 0:
        workers[i] = new Worker("localhost", 5000 + i, A1, B1, B2, B3);
        break;
    case 1:
        workers[i] = new Worker("localhost", 5000 + i, A2, B1, B2, B3);
        break;
    case 2:
        workers[i] = new Worker("localhost", 5000 + i, A3, B1, B2, B3);
        break;
    default:
        throw new RuntimeException("Invalid worker index");
}
workers[i].start();
}

```

// Espera de resultados de los servidores

```

double[][] C1 = new double[partSize][N];
double[][] C2 = new double[partSize][N];
double[][] C3 = new double[partSize][N];
for (int i = 0; i < 3; i++) {
    try {
        workers[i].join();
        switch (i) {
            case 0:
                C1 = workers[i].getC();
                break;
            case 1:
                C2 = workers[i].getC();

```

```

        break;

    case 2:
        C3 = workers[i].getC();

        break;

    default:
        throw new RuntimeException("Invalid worker index");
    }
} catch (InterruptedException e) {
    e.printStackTrace();
}
}

```

```

// Unión de las matrices C
double[][] C = new double[N][N];
for (int i = 0; i < partSize; i++) {
    for (int j = 0; j < N; j++) {
        C[i][j] = C1[i][j];
        C[i + partSize][j] = C2[i][j];
    }
}

```

Erik Gonzalez

continua desde Unión de las matrices C

ChatGPT

```

// Unión de las matrices C
double[][] C1 = new double[partSize][N];
double[][] C2 = new double[partSize][N];
double[][] C3 = new double[partSize][N];
double[][] C4 = new double[partSize][N];
double[][] C5 = new double[partSize][N];
double[][] C6 = new double[partSize][N];

```



```
double[][] C7 = new double[partSize][N];  
double[][] C8 = new double[partSize][N];  
double[][] C9 = new double[partSize][N];
```

```
for (int i = 0; i < partSize; i++) {  
    for (int j = 0; j < N; j++) {  
        C1[i][j] = CParts[0][i][j];  
        C2[i][j] = CParts[1][i][j];  
        C3[i][j] = CParts[2][i][j];  
        C4[i][j] = CParts[3][i][j];  
        C5[i][j] = CParts[4][i][j];  
        C6[i][j] = CParts[5][i][j];  
        C7[i][j] = CParts[6][i][j];  
        C8[i][j] = CParts[7][i][j];  
        C9[i][j] = CParts[8][i][j];  
    }  
}
```

```
// Unir las matrices C1, C2, C3, C4, C5, C6, C7, C8 y C9 en una sola matriz C  
double[][] C = new double[N][N];
```

```
for (int i = 0; i < partSize; i++) {  
    System.arraycopy(C1[i], 0, C[i], 0, N);  
    System.arraycopy(C2[i], 0, C[i + partSize], 0, N);  
    System.arraycopy(C3[i], 0, C[i + partSize * 2], 0, N);  
    System.arraycopy(C4[i], 0, C[i + partSize * 3], 0, N);  
    System.arraycopy(C5[i], 0, C[i + partSize * 4], 0, N);  
    System.arraycopy(C6[i], 0, C[i + partSize * 5], 0, N);  
    System.arraycopy(C7[i], 0, C[i + partSize * 6], 0, N);
```

```

System.arraycopy(C8[i], 0, C[i + partSize * 7], 0, N);
System.arraycopy(C9[i], 0, C[i + partSize * 8], 0, N);
}

```

```

// Cálculo del checksum de la matriz C

```

```

double checksum = 0;
for (int i = 0; i < N; i++) {
    for (int j = 0; j < N; j++) {
        checksum += C[i][j];
    }
}

System.out.println("Checksum: " + checksum);

```

Aquí me ocurre un problema similar que el que ocurrió anteriormente, el chat trabajaba con las partes de C como un arreglo de partes, en vez de C1 a C9, por lo que le pedí al chat cambiar esto.

Erik Gonzalez

en la Espera de resultados de los servidores necesito que por cada servidor reciba 3 matrices con dos parametros, en el primero c1,c2 y c3, y así sucesivamente hasta el C9, damelo de nuevo desde esa parte

ChatGPT

Por supuesto, aquí está el código con las modificaciones que pides:

```

// Espera de resultados de los servidores

double[][] C1 = new double[partSize][N];
double[][] C2 = new double[partSize][N];
double[][] C3 = new double[partSize][N];
double[][] C4 = new double[partSize][N];
double[][] C5 = new double[partSize][N];
double[][] C6 = new double[partSize][N];
double[][] C7 = new double[partSize][N];

```

```

double[][] C8 = new double[partSize][N];
double[][] C9 = new double[partSize][N];
for (int i = 0; i < 3; i++) {
    try {
        workers[i].join();
        double[][][] receivedC = workers[i].getC();
        C1 = receivedC[0];
        C2 = receivedC[1];
        C3 = receivedC[2];
        C4 = receivedC[3];
        C5 = receivedC[4];
        C6 = receivedC[5];
        C7 = receivedC[6];
        C8 = receivedC[7];
        C9 = receivedC[8];
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
}

```

```

// Unión de las matrices C
double[][] C = new double[N][N];
for (int i = 0; i < partSize; i++) {
    for (int j = 0; j < N; j++) {
        C[i][j] = C1[i][j];
        C[i + partSize][j] = C2[i][j];
        C[i + partSize * 2][j] = C3[i][j];
        C[i + partSize * 3][j] = C4[i][j];
        C[i + partSize * 4][j] = C5[i][j];
    }
}

```

```

        C[i + partSize * 5][j] = C6[i][j];
        C[i + partSize * 6][j] = C7[i][j];
        C[i + partSize * 7][j] = C8[i][j];
        C[i + partSize * 8][j] = C9[i][j];
    }
}

// Cálculo del checksum de la matriz C
double checksum = 0;
for (int i = 0; i < N; i++) {
    for (int j = 0; j < N; j++) {
        checksum += C[i][j];
    }
}

System.out.println("El checksum de la matriz C es: " + checksum);

```

Aquí le pedía al chat que hiciera cambios en el Worker ya que faltaban cosas por hacer dentro de él.

Erik Gonzalez

de acuerdo a este código

```

public static class Worker extends Thread {
    private String host;
    private int port;
    private double[][] A;
    private double[][] B1;
    private double[][] B2;
    private double[][] B3;
    private double[][] C;

```

```
public Worker(String host, int port, double[][] A, double[][] B1, double[][] B2, double[][] B3) {  
    this.host = host;  
    this.port = port;  
    this.A = A;  
    this.B1 = B1;  
    this.B2 = B2;  
    this.B3 = B3;  
}
```

```
public double[][] getC() {  
    return this.C;  
}
```

@Override

```
public void run() {  
    try {  
        Socket socket = new Socket(host, port);  
        ObjectOutputStream out = new ObjectOutputStream(socket.getOutputStream());  
        ObjectInputStream in = new ObjectInputStream(socket.getInputStream());  
  
        // Envío de A y B al servidor  
        out.writeObject(A);  
        out.writeObject(B1);  
        out.writeObject(B2);  
        out.writeObject(B3);  
  
        // Recepción de C del servidor  
        this.C = (double[][]) in.readObject();  
    }  
}
```

```

        // Cierre de conexiones
        out.close();
        in.close();
        socket.close();

    } catch (Exception e) {
        e.printStackTrace();
    }
}
}

```

cambialo de tal forma que en la recepcion del servidor se reciban 3 matrices

ChatGPT

Para recibir 3 matrices del servidor, se puede modificar el método run() del siguiente modo:

@Override

```

public void run() {
    try {
        Socket socket = new Socket(host, port);
        ObjectOutputStream out = new ObjectOutputStream(socket.getOutputStream());
        ObjectInputStream in = new ObjectInputStream(socket.getInputStream());

        // Envío de A y B al servidor
        out.writeObject(A);
        out.writeObject(B1);
        out.writeObject(B2);
        out.writeObject(B3);

        // Recepción de las 3 matrices C del servidor
        double[][] C1 = (double[][]) in.readObject();
    }
}

```

```

double[][] C2 = (double[][]) in.readObject();
double[][] C3 = (double[][]) in.readObject();

// Union de las matrices C
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        C[i][j] = C1[i][j];
        C[i][j+n] = C2[i][j];
        C[i][j+2*n] = C3[i][j];
    }
}

// Cierre de conexiones
out.close();
in.close();
socket.close();

} catch (Exception e) {
    e.printStackTrace();
}
}

```

De esta manera, se reciben las tres matrices C del servidor, se concatenan en la matriz final C y se cierran las conexiones.

Finalmente le pedí hacer el caso 1

Erik Gonzalez

Ahora vamos con el caso 1, este servira como servidor y tendra el primer puerto que asignaste, recibira una matriz que es la primera parte de 3 de A, y las 3 partes de B, C1(primer parte de C) es igual a $A1 * B^T1$ (primera parte de la transpuesta de B), $C2 = A1 * B^T2$,

$C3=A1*BT3$, por lo que se calculara C1,C2 y C3 y las enviara a su cliente.// Crear el socket del servidor.

Solo dame ese caso por ahora y las funciones que necesita

ChatGPT

Claro, aquí están las funciones necesarias para el caso 1:

```
ServerSocket serverSocket = new ServerSocket(PORTS[0]);
```

```
// Esperar a que llegue una conexión
```

```
System.out.println("Esperando conexión en el puerto " + PORTS[0] + "...");
```

```
Socket clientSocket = serverSocket.accept();
```

```
System.out.println("Conexión aceptada");
```

```
// Recibir la matriz A1
```

```
ObjectInputStream ois = new ObjectInputStream(clientSocket.getInputStream());
```

```
double[][] A1 = (double[][]) ois.readObject();
```

```
// Recibir las tres partes de B
```

```
double[][] B1 = receiveMatrix(clientSocket);
```

```
double[][] B2 = receiveMatrix(clientSocket);
```

```
double[][] B3 = receiveMatrix(clientSocket);
```

```
// Calcular las tres partes de C
```

```
double[][] C1 = multiplyMatrix(A1, transposeMatrix(B1));
```

```
double[][] C2 = multiplyMatrix(A1, transposeMatrix(B2));
```

```
double[][] C3 = multiplyMatrix(A1, transposeMatrix(B3));
```

```
// Enviar las tres partes de C al cliente
```

```
ObjectOutputStream oos = new ObjectOutputStream(clientSocket.getOutputStream());
```

```
oos.writeObject(C1);
```



```

oos.writeObject(C2);
oos.writeObject(C3);

// Cerrar la conexión
oos.close();
ois.close();
clientSocket.close();
serverSocket.close();
} catch (IOException e) {
    e.printStackTrace();
} catch (ClassNotFoundException e) {
    e.printStackTrace();
}

```

Erik Gonzalez

Ahora muéstrame las funciones que ocupa el caso 1

ChatGPT

Aquí te muestro las funciones necesarias para el caso 1

```

private static double[][] receiveMatrix(Socket socket) throws IOException,
ClassNotFoundException {

    // Recibir una matriz del socket

    ObjectInputStream ois = new ObjectInputStream(socket.getInputStream());

    return (double[][]) ois.readObject();

}

private static double[][] multiplyMatrix(double[][] A, double[][] B) {

    // Multiplicar dos matrices

    int n = A.length;

    int m = A[0].length;

    int p = B[0].length;

    double[][] result = new double[n][p];

```

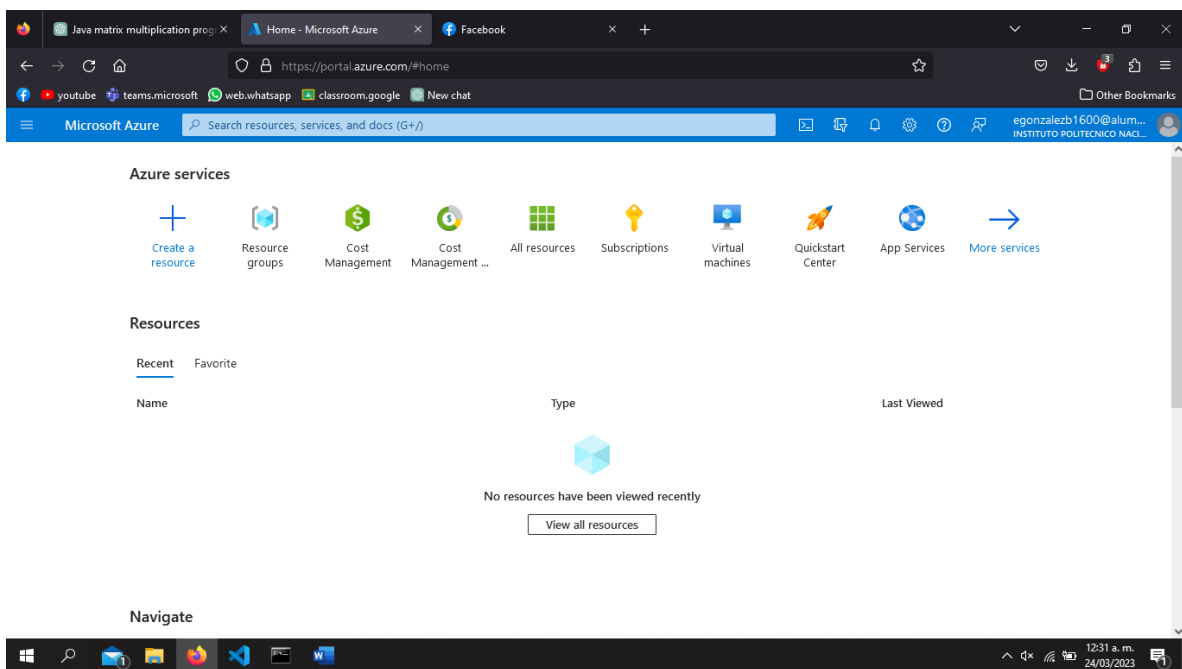
```
for (int i = 0; i < n; i++) {  
    for (int j = 0; j < p; j++) {  
        for (int k = 0; k < m; k++) {  
            result[i][j] += A[i][k] * B[k][j];  
        }  
    }  
}  
return result;  
}
```

Como los demás casos son casi lo mismo que el caso 1 no pensé que fuera necesario pedirle más, por lo que usé el código de este caso para hacer los demás, solo cambiando el tema de las partes de C correspondientes, la IP y el puerto.

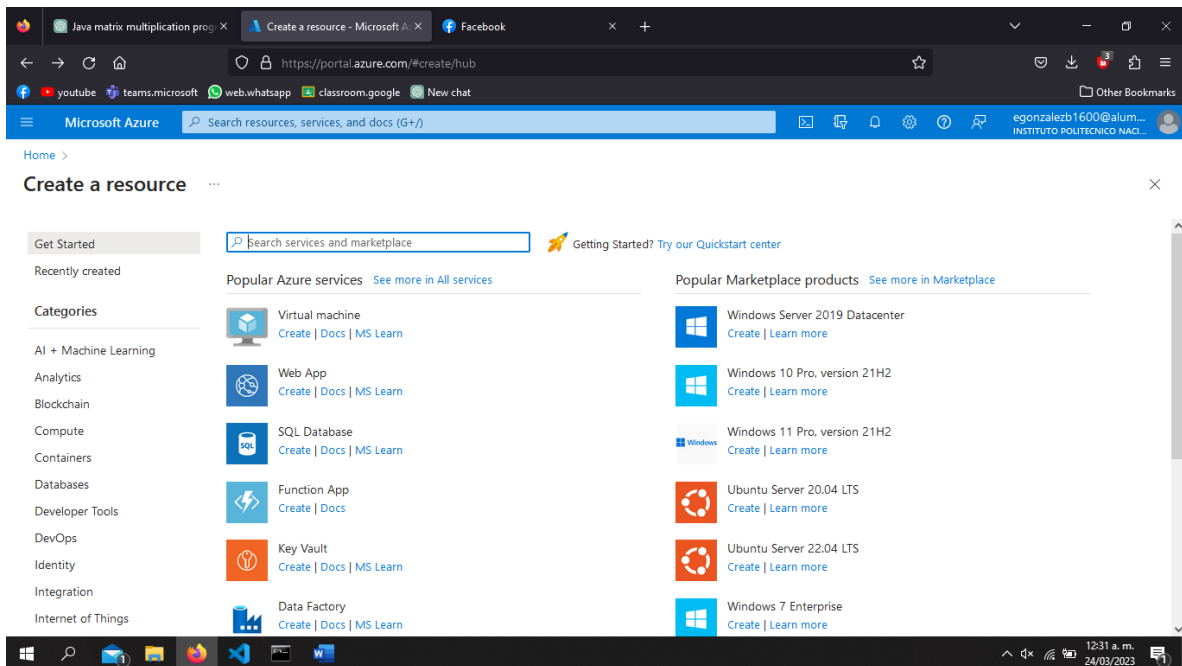
Al final tuve que hacer algunos cambios, ya que, aunque el chat mostraba la mayor parte del código es verdad que en algunos casos sentí que había mejores formas de implementar el código, como es el caso del uso del Worker, donde este creaba los Worker con arrays y yo hacia uno por uno ya que sentía era más fácil. Algo similar ocurrió en el método run del Worker donde para crear el socket no usaba un for (;;) como ya se había visto en clase ni usaba try-catch.

Ahora vamos con los pasos para crear la máquina virtual en Azure, en este caso se usaron 3.

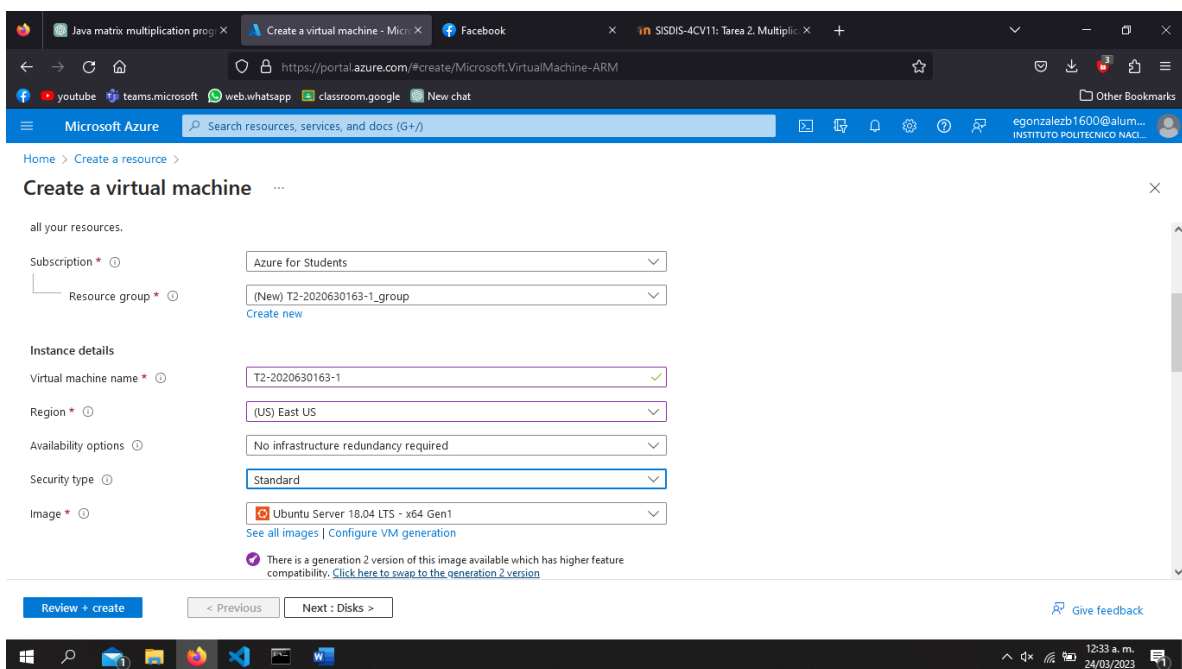
Primero nos dirigimos al link <https://azure.microsoft.com/es-mx/get-started/azure-portal/> donde tendremos que iniciar sesión y nos mostrara lo siguiente:



Entonces le damos a Create a resource y nos muestra la siguiente ventana:



Por lo que elegimos la opción de Virtual Machine y nos muestra la siguiente ventana:



Aquí tenemos que colocar lo importante de la máquina virtual, como lo es el nombre, la región en donde va a estar, el tipo de seguridad, que Sistema Operativo tendrá, la cantidad de RAM, etc., en este caso usaremos Ubuntu Server 18.0.4 LTS con 1GB de memoria RAM y 1 VCPU, que corresponde al Standard B1s, finalmente le ponemos una autenticación.



Java matrix multiplication prog x Create a virtual machine - Micr x Facebook x SSDIS-4CV11: Tarea 2. Multiplic x +


https://portal.azure.com/#create/Microsoft.VirtualMachine-ARM


Microsoft Azure Search resources, services, and docs (G+)

Home > Create a resource >


Create a virtual machine


VM architecture  Arm64
☐ x64
 Arm64 is not supported with the selected image.


Run with Azure Spot discount  ☐

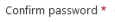
Size *  Standard_B1s - 1 vcpu, 1 GiB memory (\$7.59/month)
[See all sizes](#)

Administrator account

Authentication type  ☐ SSH public key
☒ Password

Username *  Erikgb ✓

Password *  ✓

Confirm password *  ✓

[Review + create](#) < Previous Next : Disks > [Give feedback](#)

Windows taskbar: 12:34 a. m. 24/03/2023

Y seleccionamos el puerto para SSH


Java matrix multiplication prog x Create a virtual machine - Micr x Facebook x SSDIS-4CV11: Tarea 2. Multiplic x +

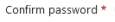
https://portal.azure.com/#create/Microsoft.VirtualMachine-ARM

Microsoft Azure Search resources, services, and docs (G+)

Home > Create a resource >


Create a virtual machine

Password *  ✓


Confirm password *  ✓

Inbound port rules

Select which virtual machine network ports are accessible from the public internet. You can specify more limited or granular network access on the Networking tab.

Public inbound ports *  ☐ None
☒ Allow selected ports

Select inbound ports * SSH (22)

 This will allow all IP addresses to access your virtual machine. This is only recommended for testing. Use the Advanced controls in the Networking tab to create rules to limit inbound traffic to known IP addresses.

[Review + create](#) < Previous Next : Disks > [Give feedback](#)

Windows taskbar: 12:34 a. m. 24/03/2023

Le damos a continuar y nos envía a la parte de Disks, aquí colocamos como Standard HDD de 30GB

The screenshot shows the 'Create a virtual machine' page in the Microsoft Azure portal, specifically the 'Disks' tab. The page is titled 'Create a virtual machine' and has a navigation bar with 'Home > Create a resource >'. The 'Disks' tab is selected, and the 'OS disk' section is visible. The 'OS disk size' is set to 'Default size (30 GiB)'. The 'OS disk type' is set to 'Standard HDD (locally-redundant storage)'. A message states: 'The selected VM size supports premium disks. We recommend Premium SSD for high IOPS workloads. Virtual machines with Premium SSD disks qualify for the 99.9% connectivity SLA.' The 'Delete with VM' checkbox is checked. The 'Key management' dropdown is set to 'Platform-managed key'. The 'Enable Ultra Disk compatibility' checkbox is unchecked, with a note: 'Ultra disk is supported in Availability Zone(s) 1,2,3 for the selected VM size Standard_B1s.' At the bottom, there are buttons for 'Review + create', '< Previous', and 'Next : Networking >'. A 'Give feedback' link is also present.

Encryption at host ☐

OS disk

OS disk size *

OS disk type *

Delete with VM ☒

Key management

Enable Ultra Disk compatibility ☐

Ultra disk is supported in Availability Zone(s) 1,2,3 for the selected VM size Standard_B1s.

[Review + create](#) [< Previous](#) [Next : Networking >](#) [Give feedback](#)

Continuamos y dejamos la configuración que tiene de networking, por lo que le damos a Next

The screenshot shows the 'Create a virtual machine' page in the Microsoft Azure portal, specifically the 'Networking' tab. The page is titled 'Create a virtual machine' and has a navigation bar with 'Home > Create a resource >'. The 'Networking' tab is selected, and the 'Network interface' section is visible. The 'Virtual network' is set to '(new) T2-2020630163-1-vnet'. The 'Subnet' is set to '(new) default (10.0.0.0/24)'. The 'Public IP' is set to '(new) T2-2020630163-1-ip'. The 'NIC network security group' is set to 'Basic'. At the bottom, there are buttons for 'Review + create', '< Previous', and 'Next : Management >'. A 'Give feedback' link is also present.

Basics **Disks** Networking Management Monitoring Advanced Tags Review + create

Define network connectivity for your virtual machine by configuring network interface card (NIC) settings. You can control ports, inbound and outbound connectivity with security group rules, or place behind an existing load balancing solution. [Learn more](#)

Network interface

When creating a virtual machine, a network interface will be created for you.

Virtual network *

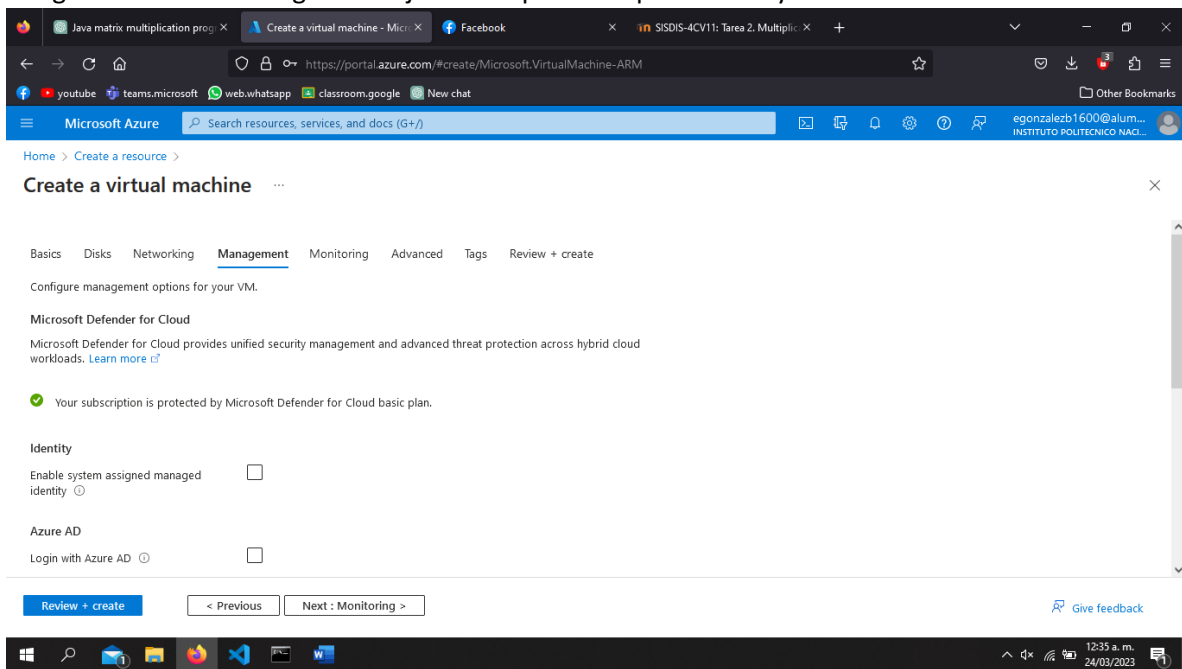
Subnet *

Public IP

NIC network security group ☐ None ☒ Basic ☐ Advanced

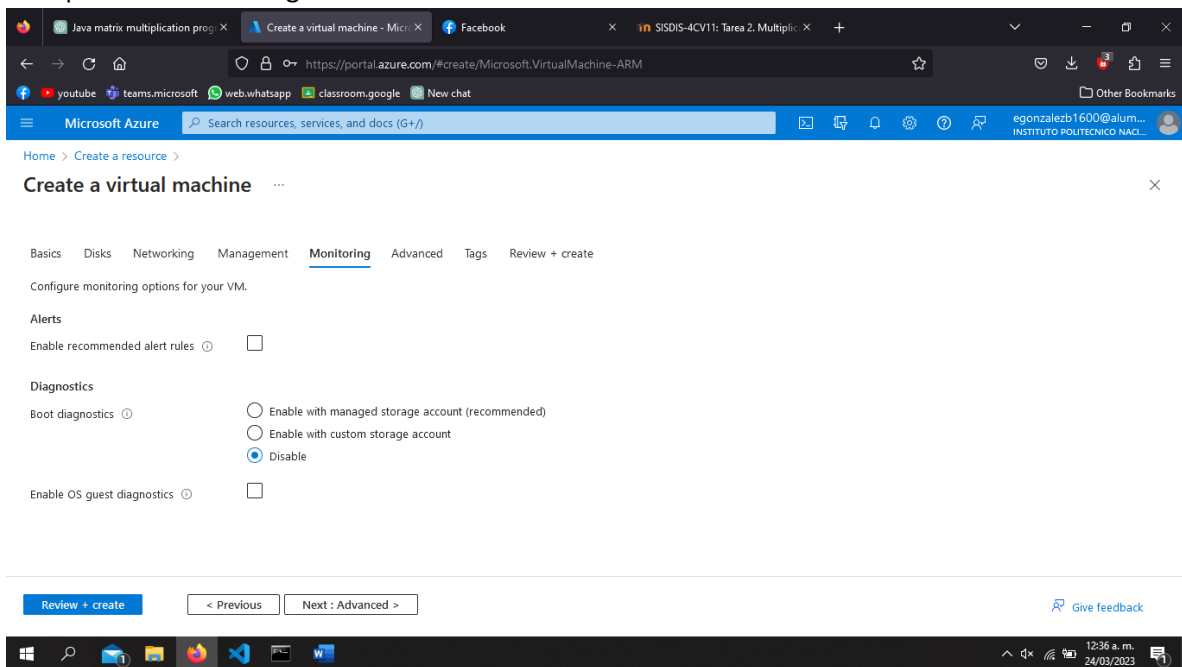
[Review + create](#) [< Previous](#) [Next : Management >](#) [Give feedback](#)

De igual forma en Management dejamos lo que viene por default y le damos a Next



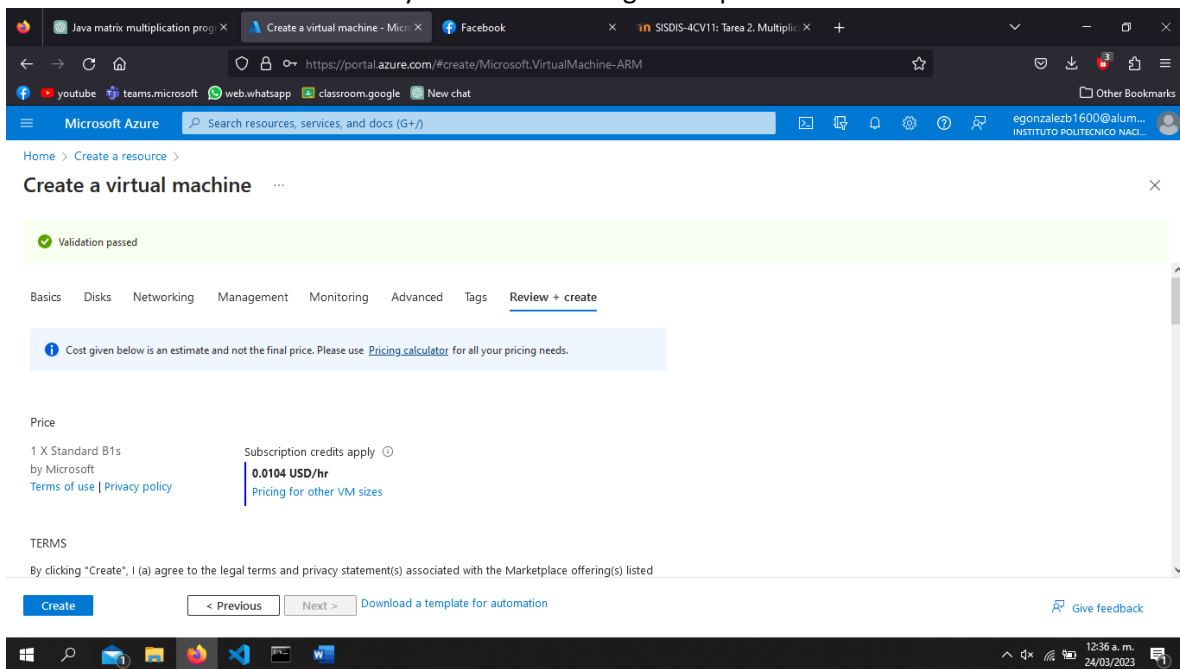
The screenshot shows the 'Create a virtual machine' page in the Microsoft Azure portal, specifically the 'Management' tab. The page title is 'Create a virtual machine'. Below the title, there are tabs for 'Basics', 'Disks', 'Networking', 'Management' (selected), 'Monitoring', 'Advanced', 'Tags', and 'Review + create'. The 'Management' section is titled 'Configure management options for your VM.' and includes a section for 'Microsoft Defender for Cloud' with a green checkmark indicating the subscription is protected. Below this, there are two sections: 'Identity' with a checkbox for 'Enable system assigned managed identity' and 'Azure AD' with a checkbox for 'Login with Azure AD'. At the bottom, there are buttons for 'Review + create', '< Previous', and 'Next : Monitoring >'. The Windows taskbar at the bottom shows the time as 12:35 a.m. on 24/03/2023.

Ahora vamos con la parte de Monitoring, aquí es importante que en la parte de Diagnostics coloquemos el Boot diagnostics como Disable

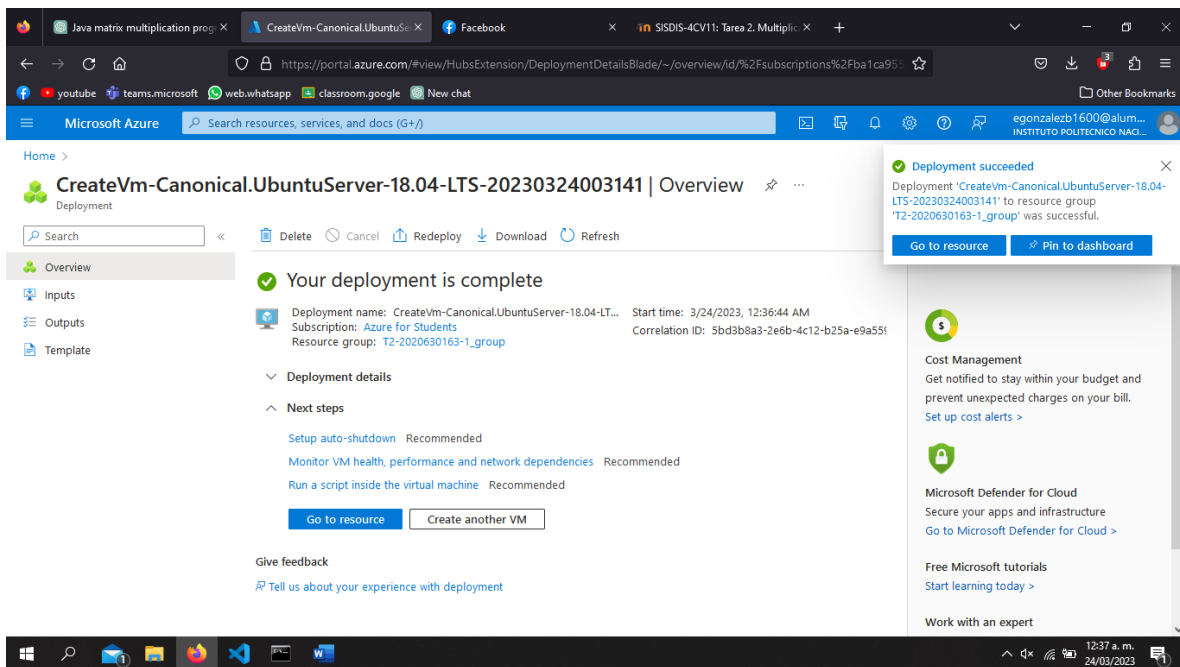


The screenshot shows the 'Create a virtual machine' page in the Microsoft Azure portal, specifically the 'Monitoring' tab. The page title is 'Create a virtual machine'. Below the title, there are tabs for 'Basics', 'Disks', 'Networking', 'Management', 'Monitoring' (selected), 'Advanced', 'Tags', and 'Review + create'. The 'Monitoring' section is titled 'Configure monitoring options for your VM.' and includes a section for 'Alerts' with a checkbox for 'Enable recommended alert rules'. Below this, there is a section for 'Diagnostics' with a checkbox for 'Boot diagnostics'. Under 'Boot diagnostics', there are three radio button options: 'Enable with managed storage account (recommended)', 'Enable with custom storage account', and 'Disable' (selected). At the bottom, there are buttons for 'Review + create', '< Previous', and 'Next : Advanced >'. The Windows taskbar at the bottom shows the time as 12:36 a.m. on 24/03/2023.

Ahora le damos a Review+create y nos muestra la siguiente pantalla.



Aquí nos dice cuanto nos va a cobrar y también un breve resumen de lo que va a tener nuestra máquina, así como su configuración. Damos click a Create.



Se nos muestra la siguiente pantalla por lo que le damos a Go to resource para ir a nuestra máquina.

Home > CreateVm-Canonical.UbuntuServer-18.04-LTS-20230324003141 | Overview >

T2-2020630163-1

Virtual machine

Search

Connect Start Restart Stop Capture Delete Refresh Open in mobile Feedback CLI / PS

Overview

- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems

Settings

- Networking
- Connect
- Disks
- Size
- Microsoft Defender for Cloud
- Advisor recommendations
- Extensions + applications
- Continuous delivery

Essentials

Resource group (move) : [T2-2020630163-1_group](#)

Status : Running

Location : East US

Subscription (move) : [Azure for Students](#)

Subscription ID : ba1ca955-c0b1-455b-a32c-127344a78844

Tags (edit) : [Click here to add tags](#)

Operating system : Linux (ubuntu 18.04)

Size : Standard B1s (1 vcpu, 1 GiB memory)

Public IP address : [20.25.113.127](#)

Virtual network/subnet : [T2-2020630163-1-vnet/default](#)

DNS name : [Not configured](#)

Properties

Monitoring Capabilities (7) Recommendations Tutorials

Virtual machine

Computer name	T2-2020630163-1
Health state	-
Operating system	Linux (ubuntu 18.04)
Publisher	Canonical

Networking

Public IP address	20.25.113.127 (Network interface t2-2020630163-1955)
Public IP address (IPv6)	-
Private IP address	10.0.0.4
Private IP address (IPv6)	-

Ya en la maquina es importante habilitar los puertos con los que vamos a trabajar, por lo que vamos a Networking y le damos a Add inbound port rule

Home > CreateVm-Canonical.UbuntuServer-18.04-LTS-20230324003141 | Overview > T2-2020630163-1

T2-2020630163-1 | Networking

Virtual machine

Search

Feedback Attach network interface Detach network interface

t2-2020630163-1955

IP configuration

Network Interface: **t2-2020630163-1955** Effective security rules Troubleshoot VM connection issues Topology

Virtual network/subnet: T2-2020630163-1-vnet/default NIC Public IP: **20.25.113.127** NIC Private IP: **10.0.0.4** Accelerated networking: **Disabled**

Inbound port rules Outbound port rules Application security groups Load balancing

Network security group **T2-2020630163-1-nsg** (attached to network interface: t2-2020630163-1955) [Add inbound port rule](#)

Impacts 0 subnets, 1 network interfaces

Priority	Name	Port	Protocol	Source	Destination	Action
300	SSH	22	TCP	Any	Any	Allow
65000	AllowVnetInBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowAzureLoadBalancerInBound	Any	Any	AzureLoadBalancer	Any	Allow
65500	DenyAllInBound	Any	Any	Any	Any	Deny

Se nos abre un cuadro de configuración, aquí en la parte de Destination port ranges colocamos el puerto, en este caso se trabaja con el puerto 60001, además de que lo colocamos como TCP.

The screenshot shows the Microsoft Azure portal interface. On the left, the 'Networking' section is expanded, showing the 'T2-2020630163-1' virtual machine. The 'Network Interface: t2-2020630163-1955' is selected, and the 'Effective security rules' tab is active. A table lists the inbound port rules:

Priority	Name	Port	Protocol
300	SSH	22	TCP
65000	AllowVnetInBound	Any	Any
65001	AllowAzureLoadBalanc...	Any	Any
65500	DenyAllInBound	Any	Any

On the right, the 'Add inbound security rule' dialog is open. The 'Source' is set to 'Any', 'Source port ranges' is '*', 'Destination' is 'Any', 'Service' is 'Custom', and 'Destination port ranges' is '60001'. The 'Protocol' is set to 'TCP'. The 'Add' button is highlighted.

Ahora ya con todo configurado nos vamos a la parte donde interactuamos con nuestra máquina virtual, por lo que en una terminal de Windows colocamos el comando `ssh user@ip` y nos pedirá posteriormente nuestra contraseña

The screenshot shows the Microsoft Azure portal interface with a terminal window open. The terminal displays the following output:

```
C:\Users\erikg>ssh erikg@20.25.113.127
erikg@20.25.113.127's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1104-azure x86_64)

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

System information as of Fri Mar 24 06:47:06 UTC 2023

System load:  0.0          Processes:    105
Usage of /:   4.5% of 28.89GB Users logged in: 0
Memory usage: 21%         IP address for eth0: 10.0.0.4
Swap usage:  0%

Expanded Security Maintenance for Applications is not enabled.

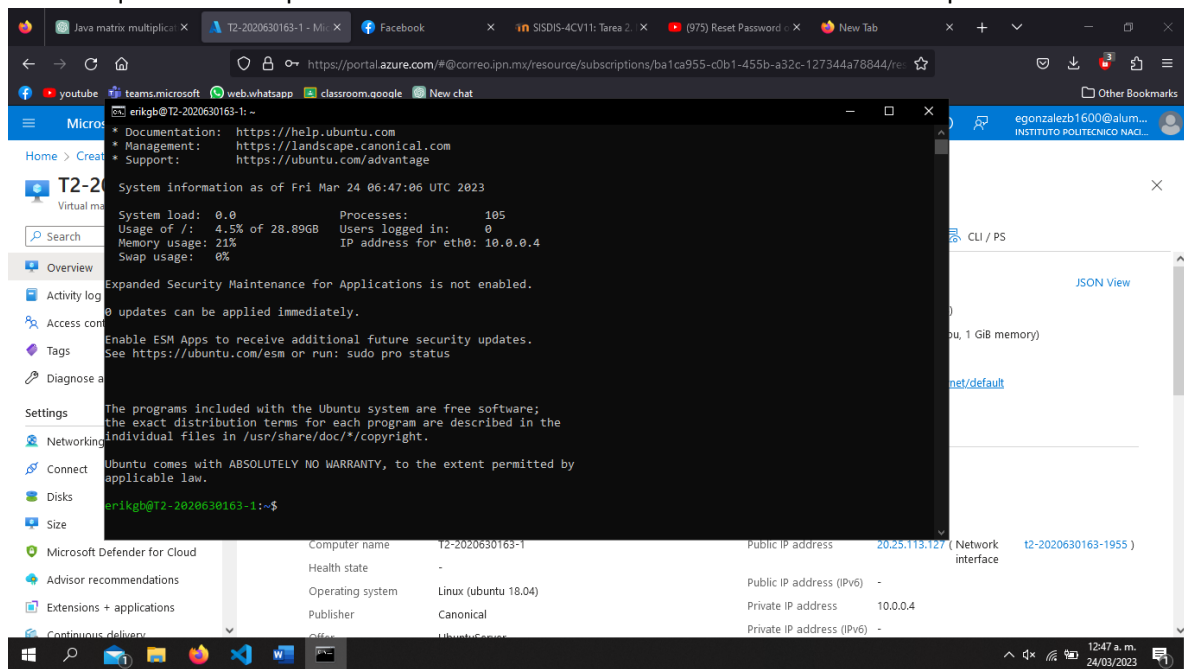
0 updates can be applied immediately.
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

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.
```

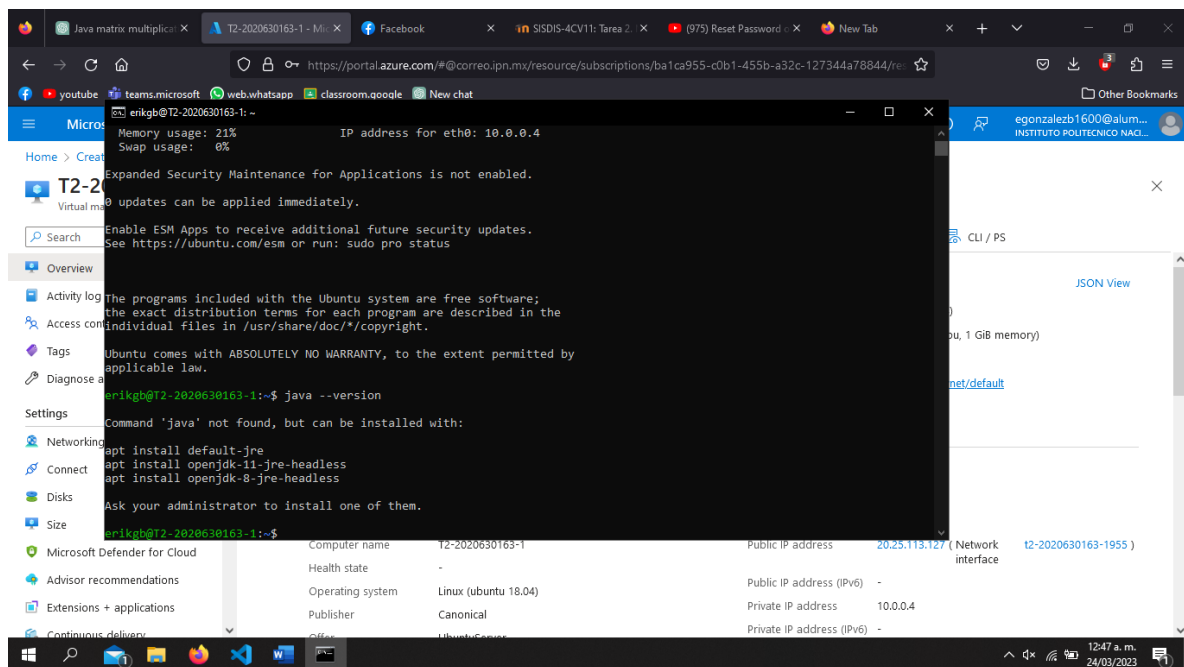
On the right, the 'T2-2020630163-1' virtual machine is selected, and the 'Networking' section is expanded. The 'Network Interface: t2-2020630163-1955' is selected, and the 'Effective security rules' tab is active. A table lists the inbound port rules:

Priority	Name	Port	Protocol
300	SSH	22	TCP
65000	AllowVnetInBound	Any	Any
65001	AllowAzureLoadBalanc...	Any	Any
65500	DenyAllInBound	Any	Any

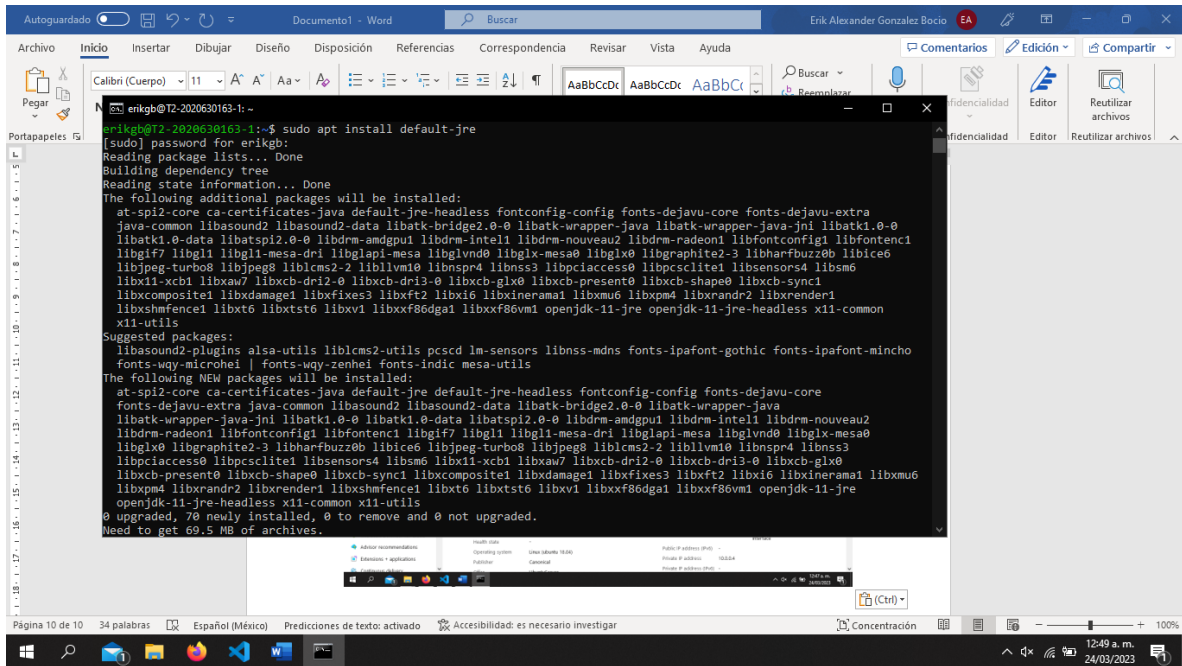
Una vez que se la diste puedes usar tu terminal de Windows como la de tu máquina virtual



El siguiente paso es ver si tenemos instalado java, por lo que escribimos el comando de la siguiente imagen, donde nos dirá que versión tenemos, en este caso no tenemos instalado el jdk ni el jre, así que los instalamos

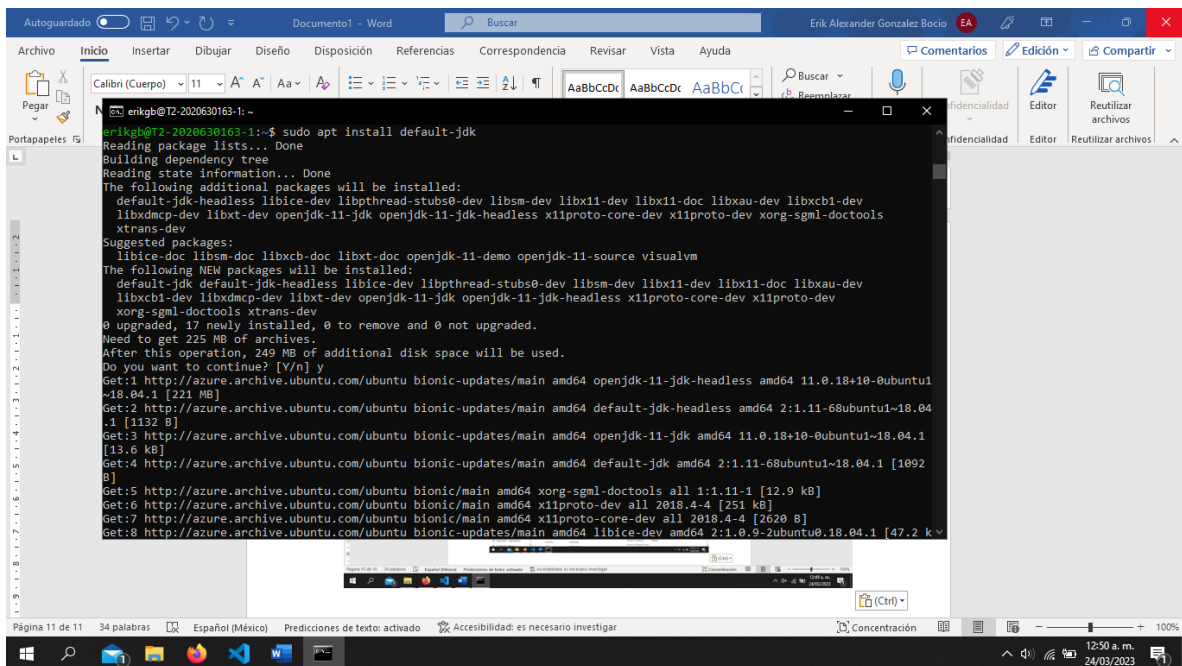


Primero instalamos el jre de la siguiente forma



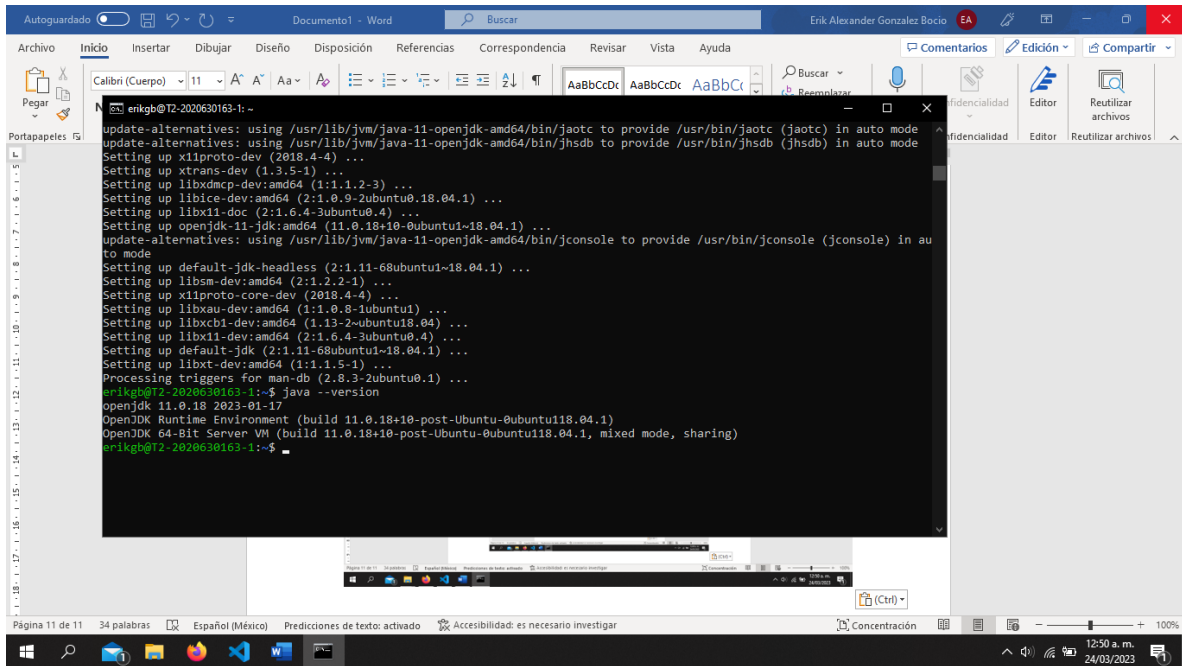
The screenshot shows a Windows Word document with a terminal window open. The terminal output shows the command `sudo apt install default-jre` being executed. The output lists various additional packages to be installed, including `at-spi2-core`, `ca-certificates`, `java`, `default-jre-headless`, `fontconfig-config`, `fonts-dejavu-core`, `fonts-dejavu-extra`, `java-common`, `libasound2`, `libasound2-data`, `libatk-bridge2.0-0`, `libatk-wrapper-java`, `libatk-wrapper-java-jni`, `libatk1.0-0`, `libatspi2.0-0`, `libdrm-amdgpu1`, `libdrm-intel1`, `libdrm-nouveau2`, `libdrm-radeon1`, `libfontconfig1`, `libfontenc1`, `libgif7`, `libgl1`, `libgl1-mesa-dri`, `libglapi-mesa`, `libglvnd0`, `libglx-mesa0`, `libgraphite2-3`, `libharfbuzz0b`, `libice6`, `libjpeg-turbo8`, `libjpeg8`, `liblcms2-2`, `liblvm20`, `libnspr4`, `libnss3`, `libpciaccess0`, `libpcsclite1`, `libensors4`, `libsm6`, `libx11-xcb1`, `libxaw7`, `libxcb-dri2-0`, `libxcb-dri3-0`, `libxcb-glx0`, `libxcb-present0`, `libxcb-shape0`, `libxcb-sync1`, `libxcomposite1`, `libxdamage1`, `libxfixes3`, `libxft2`, `libxi6`, `libxinerama1`, `libxmu6`, `libxpm4`, `libxrandr2`, `libxrender1`, `libxshmfence1`, `libxt6`, `libxtst6`, `libxv1`, `libxxf86dga1`, `libxxf86vm1`, `openjdk-11-jre`, `openjdk-11-jre-headless`, `x11-common`, and `x11-utils`. The output also shows the suggested packages: `libasound2-plugins`, `alsa-utils`, `liblcms2-utils`, `pcscd`, `lm-sensors`, `libnss-mdns`, `fonts-ipafont-gothic`, `fonts-ipafont-mincho`, `fonts-wqy-microhei`, `fonts-wqy-zenhei`, `fonts-indic`, and `mesa-utils`. The output concludes with the message: `Need to get 69.5 MB of archives.`

Y después el jdk

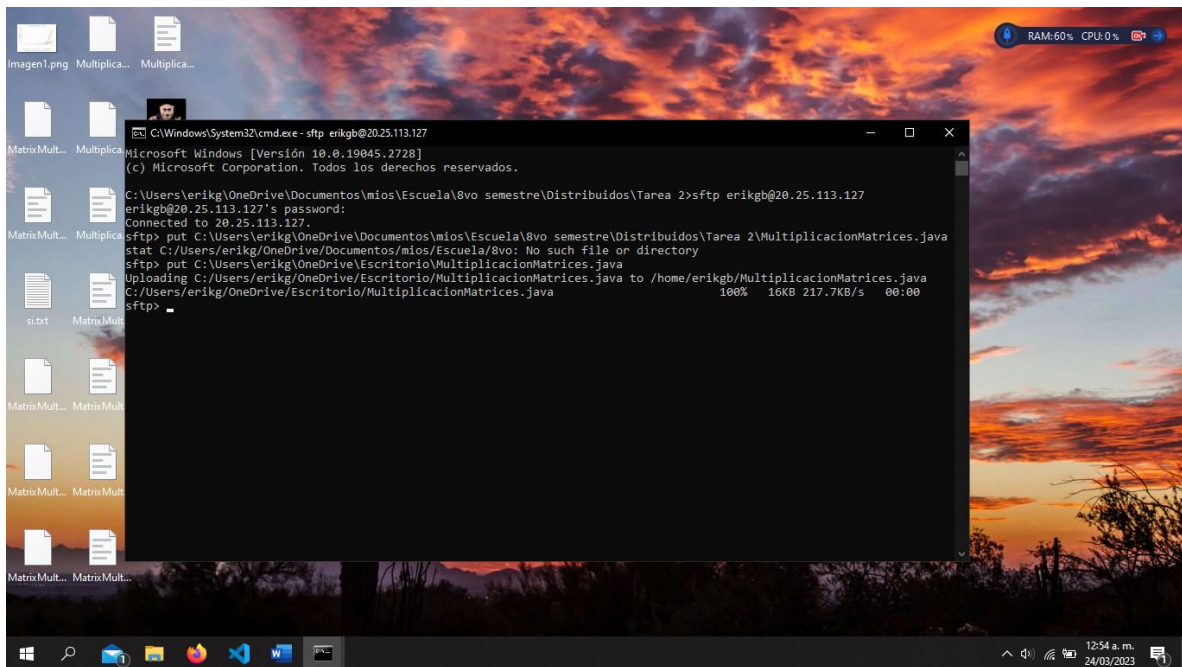


The screenshot shows a Windows Word document with a terminal window open. The terminal output shows the command `sudo apt install default-jdk` being executed. The output lists various additional packages to be installed, including `default-jdk-headless`, `libice-dev`, `libpthread-stubs0-dev`, `libsm-dev`, `libx11-dev`, `libx11-doc`, `libxau-dev`, `libxcb1-dev`, `libxdmcp-dev`, `libxtd-dev`, `openjdk-11-jdk-headless`, `x11proto-core-dev`, `x11proto-dev`, `xorg-sgml-doctools`, and `xtrans-dev`. The output also shows the suggested packages: `libice-dev`, `libsm-dev`, `libx11-dev`, `libx11-doc`, `libxau-dev`, `libxcb1-dev`, `libxdmcp-dev`, `libxtd-dev`, `openjdk-11-jdk`, `openjdk-11-jdk-headless`, `x11proto-core-dev`, `x11proto-dev`, `xorg-sgml-doctools`, and `xtrans-dev`. The output concludes with the message: `Need to get 225 MB of archives.`

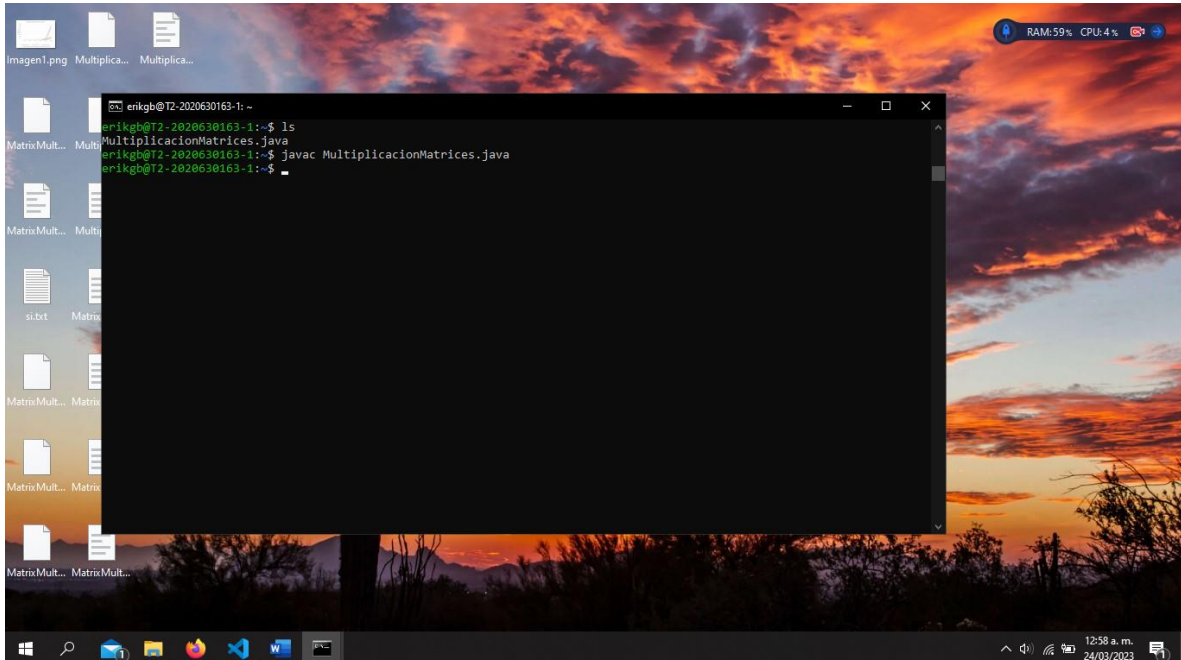
Ahora si colocamos de nuevo el comando para la versión ya nos muestra la versión.



Ahora mandamos el archivo java a nuestra máquina virtual, para ellos usamos el primer comando `sftp` que se ve en la imagen

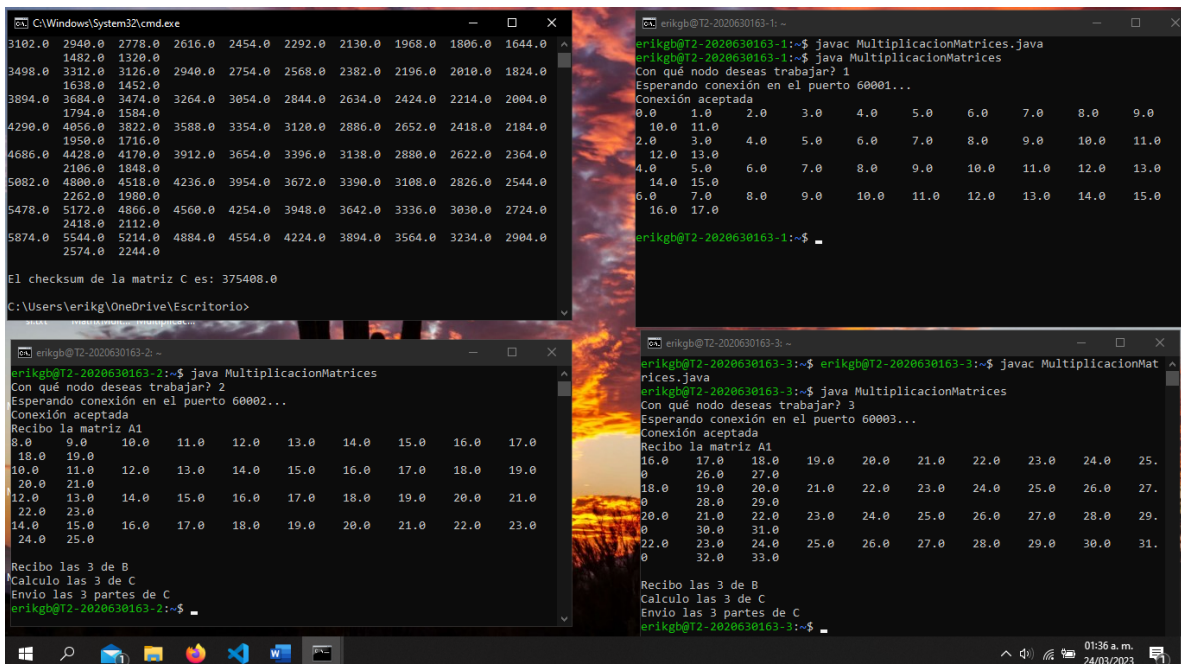


Nos pide nuestra contraseña y luego le damos el comando put donde colocamos el directorio donde tenemos el archivo, por lo que se envía.



Ahora como vimos en la captura anterior si en la máquina virtual vemos que tenemos, podemos observar que se recibió el archivo java, y usamos javac para crear el class y compilarlo.

Ahora, luego de crear las 3 máquinas virtuales, configurarlas y mandarles el programa, probamos con N=12 y obtenemos lo siguiente



Para el caso de N=12 obtenemos el checksum 375408.0, además como podemos observar en las siguientes imágenes se imprimen las 3 matrices y el checksum de C.

```
C:\Windows\System32\cmd.exe
C:\Users\erikg\OneDrive\Escritorio>java MultiplicacionMatrices.java
Con qu  nodo deseas trabajar?  

Matriz A:
0.0 1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0
1.0 10.0 11.0
2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0 10.0 11.0
4.0 5.0 6.0 7.0 8.0 9.0 10.0 11.0 12.0 13.0
6.0 7.0 8.0 9.0 10.0 11.0 12.0 13.0 14.0 15.0
8.0 9.0 10.0 11.0 12.0 13.0 14.0 15.0 16.0 17.0
10.0 11.0 12.0 13.0 14.0 15.0 16.0 17.0 18.0 19.0
12.0 13.0 14.0 15.0 16.0 17.0 18.0 19.0 20.0 21.0
14.0 15.0 16.0 17.0 18.0 19.0 20.0 21.0 22.0 23.0
16.0 17.0 18.0 19.0 20.0 21.0 22.0 23.0 24.0 25.0
18.0 19.0 20.0 21.0 22.0 23.0 24.0 25.0 26.0 27.0
20.0 21.0 22.0 23.0 24.0 25.0 26.0 27.0 28.0 29.0
22.0 23.0 24.0 25.0 26.0 27.0 28.0 29.0 30.0 31.0
32.0

Matriz B:
0.0 3.0 6.0 9.0 12.0 15.0 18.0 21.0 24.0 27.0
-1.0 2.0 5.0 8.0 11.0 14.0 17.0 20.0 23.0 26.0
-2.0 1.0 4.0 7.0 10.0 13.0 16.0 19.0 22.0 25.0
-3.0 0.0 3.0 6.0 9.0 12.0 15.0 18.0 21.0 24.0
-4.0 -1.0 2.0 5.0 8.0 11.0 14.0 17.0 20.0 23.0
-5.0 -2.0 1.0 4.0 7.0 10.0 13.0 16.0 19.0 22.0
-6.0 -3.0 0.0 3.0 6.0 9.0 12.0 15.0 18.0 21.0
-7.0 -4.0 -1.0 2.0 5.0 8.0 11.0 14.0 17.0 20.0

Matriz C:
1518.0 1452.0 1386.0 1320.0 1254.0 1188.0 1122.0 1056.0 990.0 924.0
858.0 792.0
1914.0 1824.0 1734.0 1644.0 1554.0 1464.0 1374.0 1284.0 1194.0 1104.0
1814.0 924.0
2310.0 2196.0 2082.0 1968.0 1854.0 1740.0 1626.0 1512.0 1398.0 1284.0
1170.0 1056.0
2706.0 2568.0 2430.0 2292.0 2154.0 2016.0 1878.0 1740.0 1602.0 1464.0
1326.0 1188.0
3102.0 2940.0 2778.0 2616.0 2454.0 2292.0 2130.0 1968.0 1806.0 1644.0
1482.0 1320.0
3498.0 3312.0 3126.0 2940.0 2754.0 2568.0 2382.0 2196.0 2010.0 1824.0
1638.0 1452.0
3894.0 3684.0 3474.0 3264.0 3054.0 2844.0 2634.0 2424.0 2214.0 2004.0
1794.0 1584.0
4290.0 4056.0 3822.0 3588.0 3354.0 3120.0 2886.0 2652.0 2418.0 2184.0
1950.0 1716.0
4686.0 4428.0 4170.0 3912.0 3654.0 3396.0 3138.0 2880.0 2622.0 2364.0
2106.0 1848.0
5082.0 4800.0 4518.0 4236.0 3954.0 3672.0 3390.0 3108.0 2826.0 2544.0
2262.0 1980.0
5478.0 5172.0 4866.0 4560.0 4254.0 3948.0 3642.0 3336.0 3030.0 2724.0
2418.0 2112.0
5874.0 5544.0 5214.0 4884.0 4554.0 4224.0 3894.0 3564.0 3234.0 2904.0
2574.0

El checksum de la matriz C es: 375408.0
C:\Users\erikg\OneDrive\Escritorio>
```

Ahora probamos con N=3000

