Página principal de azure

Una captura de pantalla de una computadora

Descripción generada automáticamente

Creación de la máquina virtual

Una captura de pantalla de una computadora

Descripción generada automáticamente

Una captura de pantalla de una computadora

Descripción generada automáticamente

Una captura de pantalla de una computadora

Descripción generada automáticamente

Una captura de pantalla de una computadora

Descripción generada automáticamente

Una captura de pantalla de una computadora

Descripción generada automáticamente

Una captura de pantalla de una computadora

Descripción generada automáticamente

Una captura de pantalla de una computadora

Descripción generada automáticamente

Una captura de pantalla de una computadora

Descripción generada automáticamente

Una captura de pantalla de una computadora

Descripción generada automáticamente

Una captura de pantalla de una computadora

Descripción generada automáticamente

Actualizamos listas de paquetes

Una captura de pantalla de una computadora

Descripción generada automáticamente

Instalamos java debido a que no lo tenemos instalado

Una captura de pantalla de una computadora

Descripción generada automáticamente

Instalamos Java

Una captura de pantalla de una computadora

Descripción generada automáticamente

Una captura de pantalla de una computadora

Descripción generada automáticamente

Creación de la máquina T2-2019630034-2

Una captura de pantalla de una computadora

Descripción generada automáticamente

Selección de los discos

Una captura de pantalla de una computadora

Descripción generada automáticamente

Opciones de redes, no se mueve nada

Una captura de pantalla de una computadora

Descripción generada automáticamente

Opciones de administración

Una captura de pantalla de una computadora

Descripción generada automáticamente

Revisamos las características y creamos el recurso

Una captura de pantalla de una computadora

Descripción generada automáticamente

Instalación de java en T2-2019630034-2

Una captura de pantalla de una computadora

Descripción generada automáticamente

Creación de la máquina virtual T2-2019630034-3

Una captura de pantalla de una computadora

Descripción generada automáticamente

Discos

Una captura de pantalla de una computadora

Descripción generada automáticamente

Una captura de pantalla de una computadora

Descripción generada automáticamente

Revisar y crear

Una captura de pantalla de una computadora

Descripción generada automáticamente

Conexión con la VM mediante ssh

Una captura de pantalla de una computadora

Descripción generada automáticamente

Instalamos java

Una captura de pantalla de una computadora

Descripción generada automáticamente

Conversación con chatGPT

Para empezar con el programa, primero lo modulé en programas mas sencillos, para empezar le pedí a chatGPT que hiciera un programa en java que inicializara 2 matrices conforme a las reglas dadas en la tarea. La conversación es la siguiente:

Mario Sebastian

Hola

ChatGPT

¡Hola! ¿En qué puedo ayudarte hoy?

Mario Sebastian

Por favor crea un programa en java que inicialice 2 matrices A y B, en donde A[i][j] y B[i][j] son elementos de las matrices A y B respectivamente y A[i][j] = 2\*i+j , B[i][j] = 3\*i-j, la dimensión de la matriz es N\*N.

ChatGPT

¡Claro! Aquí te muestro cómo crear el programa en Java que inicializa dos matrices A y B, según las condiciones que indicaste:

**public class Matrices {**

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

**int N = 4; // la dimensión de la matriz N\*N**

**int[][] A = new int[N][N];**

**int[][] B = new int[N][N];**

**// Inicializar la matriz A**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**A[i][j] = 2\*i+j;**

**}**

**}**

**// Inicializar la matriz B**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**B[i][j] = 3\*i-j;**

**}**

**}**

**// Mostrar las matrices A y B en la consola**

**System.out.println("Matriz A:");**

**mostrarMatriz(A);**

**System.out.println("Matriz B:");**

**mostrarMatriz(B);**

**}**

**// Método para mostrar una matriz en la consola**

**public static void mostrarMatriz(int[][] matriz) {**

**for(int i=0; i<matriz.length; i++) {**

**for(int j=0; j<matriz[0].length; j++) {**

**System.out.print(matriz[i][j] + " ");**

**}**

**System.out.println();**

**}**

**System.out.println();**

**}**

**}**

En este programa, creamos dos matrices A y B de tamaño N\*N, luego las inicializamos con los valores correspondientes utilizando dos bucles for anidados, y finalmente las mostramos en la consola utilizando el método mostrarMatriz(). En la salida de consola, primero se mostrará la matriz A y luego la matriz B. Puedes cambiar el valor de N para ajustar la dimensión de la matriz.

Mario Sebastian

ahora modifica el programa para que adicionalmente se calcule la matriz transpuesta de B

ChatGPT

**¡Claro! Aquí te muestro cómo modificar el programa anterior para calcular la matriz transpuesta de B:**

**public class Matrices {**

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

**int N = 4; // la dimensión de la matriz N\*N**

**int[][] A = new int[N][N];**

**int[][] B = new int[N][N];**

**int[][] Bt = new int[N][N]; // matriz transpuesta de B**

**// Inicializar la matriz A**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**A[i][j] = 2\*i+j;**

**}**

**}**

**// Inicializar la matriz B**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**B[i][j] = 3\*i-j;**

**}**

**}**

**// Calcular la matriz transpuesta de B**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**Bt[i][j] = B[j][i];**

**}**

**}**

**// Mostrar las matrices A, B y Bt en la consola**

**System.out.println("Matriz A:");**

**mostrarMatriz(A);**

**System.out.println("Matriz B:");**

**mostrarMatriz(B);**

**System.out.println("Matriz transpuesta de B:");**

**mostrarMatriz(Bt);**

**}**

**// Método para mostrar una matriz en la consola**

**public static void mostrarMatriz(int[][] matriz) {**

**for(int i=0; i<matriz.length; i++) {**

**for(int j=0; j<matriz[0].length; j++) {**

**System.out.print(matriz[i][j] + " ");**

**}**

**System.out.println();**

**}**

**System.out.println();**

**}**

**}**

En este programa, hemos agregado una nueva matriz Bt para almacenar la matriz transpuesta de B. Luego, hemos utilizado dos bucles for anidados para calcular los elementos de Bt y los hemos mostrado en la consola después de la matriz B. La salida de consola mostrará la matriz A, B y la matriz transpuesta de B.

Mario Sebastian

modifica el código para que la matriz transpuesta se almacene en la matriz B

ChatGPT

¡Por supuesto! Aquí te muestro cómo modificar el programa anterior para almacenar la matriz transpuesta en la matriz B:

**public class Matrices {**

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

**int N = 4; // la dimensión de la matriz N\*N**

**int[][] A = new int[N][N];**

**int[][] B = new int[N][N];**

**// Inicializar la matriz A**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**A[i][j] = 2\*i+j;**

**}**

**}**

**// Inicializar la matriz B y calcular su transpuesta**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**B[i][j] = 3\*i-j;**

**// Calcular la matriz transpuesta de B mientras se inicializa**

**B[j][i] = 3\*i-j;**

**}**

**}**

**// Mostrar las matrices A y B en la consola**

**System.out.println("Matriz A:");**

**mostrarMatriz(A);**

**System.out.println("Matriz B:");**

**mostrarMatriz(B);**

**}**

**// Método para mostrar una matriz en la consola**

**public static void mostrarMatriz(int[][] matriz) {**

**for(int i=0; i<matriz.length; i++) {**

**for(int j=0; j<matriz[0].length; j++) {**

**System.out.print(matriz[i][j] + " ");**

**}**

**System.out.println();**

**}**

**System.out.println();**

**}**

**}**

Mario Sebastian

modifica el código para que especifiques que estas imprimiendo la matriz transpuesta y no B

ChatGPT

**¡Claro! Aquí te muestro cómo modificar el programa anterior para que se especifique que se está imprimiendo la matriz transpuesta:**

**public class Matrices {**

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

**int N = 4; // la dimensión de la matriz N\*N**

**int[][] A = new int[N][N];**

**int[][] B = new int[N][N];**

**// Inicializar la matriz A**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**A[i][j] = 2\*i+j;**

**}**

**}**

**// Inicializar la matriz B y calcular su transpuesta**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**B[i][j] = 3\*i-j;**

**// Calcular la matriz transpuesta de B mientras se inicializa**

**B[j][i] = 3\*i-j;**

**}**

**}**

**// Mostrar la matriz A en la consola**

**System.out.println("Matriz A:");**

**mostrarMatriz(A);**

**// Mostrar la matriz transpuesta de B en la consola**

**System.out.println("Matriz transpuesta de B:");**

**mostrarMatriz(B);**

**}**

**// Método para mostrar una matriz en la consola**

**public static void mostrarMatriz(int[][] matriz) {**

**for(int i=0; i<matriz.length; i++) {**

**for(int j=0; j<matriz[0].length; j++) {**

**System.out.print(matriz[i][j] + " ");**

**}**

**System.out.println();**

**}**

**System.out.println();**

**}**

**}**

En este programa, hemos agregado una nueva línea de código antes de imprimir la matriz transpuesta que especifica que se está imprimiendo la matriz transpuesta de B. Luego, hemos mostrado la matriz transpuesta de B en la consola. La salida de consola mostrará la matriz A y la matriz B transpuesta, pero ahora se especifica claramente qué matriz se está imprimiendo.

Una vez teníamos la primera parte que consiste en inicializar las matrices A, B y obtener la transpuesta de B, el siguiente paso es mandarlas por un socket TCP, por que le pedí a chatGPT en una nueva conversación que modificará el código para que las matrices se mandaran por sockets de la siguiente manera:

Mario Sebastian

el siguiente código de java inicializa 2 matrices A, B y calcula la matriz transpuesta de B, modifica el código para que pueda mandar las matrices por un socket TCP hacia un servidor: public class Matrices {

public static void main(String[] args) {

int N = 4; // la dimensión de la matriz N\*N

int[][] A = new int[N][N];

int[][] B = new int[N][N];

// Inicializar la matriz A

for(int i=0; i<N; i++) {

for(int j=0; j<N; j++) {

A[i][j] = 2\*i+j;

}

}

// Inicializar la matriz B y calcular su transpuesta

for(int i=0; i<N; i++) {

for(int j=0; j<N; j++) {

B[i][j] = 3\*i-j;

// Calcular la matriz transpuesta de B mientras se inicializa

B[j][i] = 3\*i-j;

}

}

// Mostrar la matriz A en la consola

System.out.println("Matriz A:");

mostrarMatriz(A);

// Mostrar la matriz transpuesta de B en la consola

System.out.println("Matriz transpuesta de B:");

mostrarMatriz(B);

}

// Método para mostrar una matriz en la consola

public static void mostrarMatriz(int[][] matriz) {

for(int i=0; i<matriz.length; i++) {

for(int j=0; j<matriz[0].length; j++) {

System.out.print(matriz[i][j] + " ");

}

System.out.println();

}

System.out.println();

}

}

ChatGPT

Para enviar las matrices a través de un socket TCP hacia un servidor, primero necesitarás convertirlas en un formato que se pueda enviar a través del socket. Una opción sería convertir cada matriz en una cadena de caracteres y enviar esas cadenas al servidor. Luego, el servidor podría convertir las cadenas de caracteres de nuevo en matrices.

A continuación se muestra cómo modificar el código para enviar las matrices por un socket TCP:

**import java.io.\*;**

**import java.net.\*;**

**public class Matrices {**

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

**int N = 4; // la dimensión de la matriz N\*N**

**int[][] A = new int[N][N];**

**int[][] B = new int[N][N];**

**// Inicializar la matriz A**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**A[i][j] = 2\*i+j;**

**}**

**}**

**// Inicializar la matriz B y calcular su transpuesta**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**B[i][j] = 3\*i-j;**

**// Calcular la matriz transpuesta de B mientras se inicializa**

**B[j][i] = 3\*i-j;**

**}**

**}**

**// Mostrar la matriz A en la consola**

**System.out.println("Matriz A:");**

**mostrarMatriz(A);**

**// Mostrar la matriz transpuesta de B en la consola**

**System.out.println("Matriz transpuesta de B:");**

**mostrarMatriz(B);**

**// Enviar las matrices al servidor**

**try {**

**// Conectar al servidor**

**Socket socket = new Socket("direccion\_del\_servidor", puerto\_del\_servidor);**

**// Obtener los flujos de entrada y salida**

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

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

**// Enviar la matriz A**

**out.writeObject(A);**

**// Enviar la matriz B**

**out.writeObject(B);**

**// Recibir la matriz transpuesta de B desde el servidor**

**int[][] B\_transpuesta = (int[][])in.readObject();**

**// Mostrar la matriz transpuesta de B recibida del servidor en la consola**

**System.out.println("Matriz transpuesta de B recibida del servidor:");**

**mostrarMatriz(B\_transpuesta);**

**// Cerrar los flujos y el socket**

**out.close();**

**in.close();**

**socket.close();**

**} catch (IOException | ClassNotFoundException e) {**

**e.printStackTrace();**

**}**

**}**

**// Método para mostrar una matriz en la consola**

**public static void mostrarMatriz(int[][] matriz) {**

**for(int i=0; i<matriz.length; i++) {**

**for(int j=0; j<matriz[0].length; j++) {**

**System.out.print(matriz[i][j] + " ");**

**}**

**System.out.println();**

**}**

**System.out.println();**

**}**

**}**

En este código, se usa un ObjectOutputStream para enviar las matrices al servidor y un ObjectInputStream para recibir la matriz transpuesta de B del servidor. También se proporciona la dirección del servidor y el puerto del servidor en el que se escuchará la conexión del socket. Asegúrate de modificar estos valores para que coincidan con la configuración del servidor.

Hasta este punto ya tenemos la inicialización de matrices y el cliente que manda las matrices, ahora teníamos el problema de los servidores, en otra conversación le pedí a chatGPT que programara un servidor de la siguiente manera:

Mario Sebastian

programa en java un servidor que este en espera de una conexión por un cliente, dicho servidor recibirá 2 matrices A y B de dimensión N y mostrará en consola dichas matrices

ChatGPT

Aquí te proporciono un ejemplo de código en Java para un servidor que espera la conexión de un cliente y recibe dos matrices A y B de dimensión N. Este servidor utilizará sockets para comunicarse con el cliente.

**import java.io.\*;**

**import java.net.\*;**

**public class ServidorMatrices {**

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

**try {**

**// Creamos el socket del servidor y lo ponemos en espera de conexiones**

**ServerSocket servidor = new ServerSocket(9999);**

**System.out.println("Servidor en espera de conexión...");**

**// Aceptamos la conexión del cliente**

**Socket cliente = servidor.accept();**

**System.out.println("Cliente conectado");**

**// Obtenemos los flujos de entrada y salida**

**DataInputStream entrada = new DataInputStream(cliente.getInputStream());**

**DataOutputStream salida = new DataOutputStream(cliente.getOutputStream());**

**// Leemos la dimensión de las matrices**

**int n = entrada.readInt();**

**System.out.println("Dimensión de las matrices: " + n);**

**// Creamos las matrices A y B**

**int[][] a = new int[n][n];**

**int[][] b = new int[n][n];**

**// Leemos la matriz A**

**System.out.println("Leyendo la matriz A...");**

**for (int i = 0; i < n; i++) {**

**for (int j = 0; j < n; j++) {**

**a[i][j] = entrada.readInt();**

**}**

**}**

**// Leemos la matriz B**

**System.out.println("Leyendo la matriz B...");**

**for (int i = 0; i < n; i++) {**

**for (int j = 0; j < n; j++) {**

**b[i][j] = entrada.readInt();**

**}**

**}**

**// Mostramos las matrices A y B en consola**

**System.out.println("Matriz A:");**

**mostrarMatriz(a);**

**System.out.println("Matriz B:");**

**mostrarMatriz(b);**

**// Cerramos los flujos y el socket del cliente**

**entrada.close();**

**salida.close();**

**cliente.close();**

**} catch (IOException e) {**

**e.printStackTrace();**

**}**

**}**

**public static void mostrarMatriz(int[][] matriz) {**

**for (int i = 0; i < matriz.length; i++) {**

**for (int j = 0; j < matriz[0].length; j++) {**

**System.out.print(matriz[i][j] + " ");**

**}**

**System.out.println();**

**}**

**}**

**}**

Ten en cuenta que este código simplemente recibe las matrices A y B del cliente y las muestra en consola. No realiza ningún cálculo con las matrices. Además, es necesario que el cliente envíe las matrices en el formato correcto y que ambos, cliente y servidor, estén de acuerdo en el protocolo de comunicación a seguir.

Mario Sebastian

modifica el programa para que la dimensión de la matrices sea N/3

ChatGPT

Aquí te proporciono la modificación del código anterior para que la dimensión de las matrices sea N/3:

**import java.io.\*;**

**import java.net.\*;**

**public class ServidorMatrices {**

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

**try {**

**// Creamos el socket del servidor y lo ponemos en espera de conexiones**

**ServerSocket servidor = new ServerSocket(9999);**

**System.out.println("Servidor en espera de conexión...");**

**// Aceptamos la conexión del cliente**

**Socket cliente = servidor.accept();**

**System.out.println("Cliente conectado");**

**// Obtenemos los flujos de entrada y salida**

**DataInputStream entrada = new DataInputStream(cliente.getInputStream());**

**DataOutputStream salida = new DataOutputStream(cliente.getOutputStream());**

**// Leemos la dimensión de las matrices**

**int n = entrada.readInt();**

**System.out.println("Dimensión de las matrices: " + n);**

**// Calculamos la dimensión de las submatrices**

**int m = n / 3;**

**// Creamos las matrices A y B**

**int[][] a = new int[m][m];**

**int[][] b = new int[m][m];**

**// Leemos la matriz A**

**System.out.println("Leyendo la matriz A...");**

**for (int i = 0; i < m; i++) {**

**for (int j = 0; j < m; j++) {**

**a[i][j] = entrada.readInt();**

**}**

**}**

**// Leemos la matriz B**

**System.out.println("Leyendo la matriz B...");**

**for (int i = 0; i < m; i++) {**

**for (int j = 0; j < m; j++) {**

**b[i][j] = entrada.readInt();**

**}**

**}**

**// Mostramos las matrices A y B en consola**

**System.out.println("Matriz A:");**

**mostrarMatriz(a);**

**System.out.println("Matriz B:");**

**mostrarMatriz(b);**

**// Cerramos los flujos y el socket del cliente**

**entrada.close();**

**salida.close();**

**cliente.close();**

**} catch (IOException e) {**

**e.printStackTrace();**

**}**

**}**

**public static void mostrarMatriz(int[][] matriz) {**

**for (int i = 0; i < matriz.length; i++) {**

**for (int j = 0; j < matriz[0].length; j++) {**

**System.out.print(matriz[i][j] + " ");**

**}**

**System.out.println();**

**}**

**}**

**}**

Sin embargo, al correr el programa que escribió lanzaba un error, por lo tanto le mandé el error para que lo corrigiera

Mario Sebastian

tengo este error al correr el servidor: error: class, interface, enum, or record expected

int n = entrada.readInt();

^

ChatGPT

Este error puede ocurrir cuando una instrucción está fuera de lugar en el código, por lo que Java espera encontrar una clase, interfaz, enum o registro en ese punto en su lugar.

En el código que proporcioné anteriormente, puede ocurrir este error si la instrucción int n = entrada.readInt(); se encuentra fuera de un método o bloque de código. Asegúrate de que esta instrucción esté dentro del método main:

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

**try {**

**// Creamos el socket del servidor y lo ponemos en espera de conexiones**

**ServerSocket servidor = new ServerSocket(9999);**

**System.out.println("Servidor en espera de conexión...");**

**// Aceptamos la conexión del cliente**

**Socket cliente = servidor.accept();**

**System.out.println("Cliente conectado");**

**// Obtenemos los flujos de entrada y salida**

**DataInputStream entrada = new DataInputStream(cliente.getInputStream());**

**DataOutputStream salida = new DataOutputStream(cliente.getOutputStream());**

**// Leemos la dimensión de las matrices**

**int n = entrada.readInt(); // Esta instrucción debe estar dentro del método main**

**System.out.println("Dimensión de las matrices: " + n);**

**// Calculamos la dimensión de las submatrices**

**int m = n / 3;**

**// Creamos las matrices A y B**

**int[][] a = new int[m][m];**

**int[][] b = new int[m][m];**

**// Leemos la matriz A**

**System.out.println("Leyendo la matriz A...");**

**for (int i = 0; i < m; i++) {**

**for (int j = 0; j < m; j++) {**

**a[i][j] = entrada.readInt();**

**}**

**}**

**// Leemos la matriz B**

**System.out.println("Leyendo la matriz B...");**

**for (int i = 0; i < m; i++) {**

**for (int j = 0; j < m; j++) {**

**b[i][j] = entrada.readInt();**

**}**

**}**

**// Mostramos las matrices A y B en consola**

**System.out.println("Matriz A:");**

**mostrarMatriz(a);**

**System.out.println("Matriz B:");**

**mostrarMatriz(b);**

**// Cerramos los flujos y el socket del cliente**

**entrada.close();**

**salida.close();**

**cliente.close();**

**} catch (IOException e) {**

**e.printStackTrace();**

**}**

**}**

Si este no es el problema, revisa cuidadosamente el código y verifica que todas las llaves y paréntesis estén correctamente cerrados y abiertos en el lugar correcto.

Al final hacia falta una {

Luego al revisar los detalles de la práctica olvidé que las matrices no son de tipo int si no de tipo double, entonces le pedí que lo hiciera

Mario Sebastian

el siguiente código en java inicializa 2 matrices A, B, calcula la transpuesta de la matriz B y manda dichas matrices por un socket tcp, modifica el programa para que las matrices sean de tipo double en vez de int: **import java.io.\*;**

**import java.net.\*;**

**public class Nodo0 {**

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

**int N = 4; // la dimensión de la matriz N\*N**

**int[][] A = new int[N][N];**

**int[][] B = new int[N][N];**

**// Inicializar la matriz A**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**A[i][j] = 2\*i+j;**

**}**

**}**

**// Inicializar la matriz B y calcular su transpuesta**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**B[i][j] = 3\*i-j;**

**// Calcular la matriz transpuesta de B mientras se inicializa**

**B[j][i] = 3\*i-j;**

**}**

**}**

**// Mostrar la matriz A en la consola**

**System.out.println("Matriz A:");**

**mostrarMatriz(A);**

**// Mostrar la matriz transpuesta de B en la consola**

**System.out.println("Matriz transpuesta de B:");**

**mostrarMatriz(B);**

**// Enviar las matrices al servidor**

**try {**

**// Conectar al servidor**

**Socket socket = new Socket("localhost", 5000);**

**// Obtener los flujos de entrada y salida**

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

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

**//Enviar dimension**

**out.writeObject(N);**

**// Enviar la matriz A**

**out.writeObject(A);**

**// Enviar la matriz B**

**out.writeObject(B);**

**// Recibir la matriz transpuesta de B desde el servidor**

**int[][] B\_transpuesta = (int[][])in.readObject();**

**// Mostrar la matriz transpuesta de B recibida del servidor en la consola**

**System.out.println("Matriz transpuesta de B recibida del servidor:");**

**mostrarMatriz(B\_transpuesta);**

**// Cerrar los flujos y el socket**

**out.close();**

**in.close();**

**socket.close();**

**} catch (IOException | ClassNotFoundException e) {**

**e.printStackTrace();**

**}**

**}**

**// Método para mostrar una matriz en la consola**

**public static void mostrarMatriz(int[][] matriz) {**

**for(int i=0; i<matriz.length; i++) {**

**for(int j=0; j<matriz[0].length; j++) {**

**System.out.print(matriz[i][j] + " ");**

**}**

**System.out.println();**

**}**

**System.out.println();**

**}**

**}**

ChatGPT

Para modificar el programa y hacer que las matrices sean de tipo double en lugar de int, debemos realizar los siguientes cambios:

1. Cambiar la definición de las matrices A y B de int[][] a double[][].
2. Actualizar los valores de las matrices A y B para que sean de tipo double en lugar de int.
3. Actualizar el cálculo de la matriz transpuesta de B para que se realice con valores de tipo double.
4. Actualizar el tipo de datos en el método mostrarMatriz de int[][] a double[][].
5. Actualizar el tipo de datos en la línea donde se recibe la matriz transpuesta de B desde el servidor de int[][] a double[][].

Aquí está el código actualizado con estos cambios:

**import java.io.\*;**

**import java.net.\*;**

**public class Nodo0 {**

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

**int N = 4; // la dimensión de la matriz N\*N**

**double[][] A = new double[N][N];**

**double[][] B = new double[N][N];**

**// Inicializar la matriz A**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**A[i][j] = 2\*i+j;**

**}**

**}**

**// Inicializar la matriz B y calcular su transpuesta**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**B[i][j] = 3\*i-j;**

**// Calcular la matriz transpuesta de B mientras se inicializa**

**B[j][i] = 3\*i-j;**

**}**

**}**

**// Mostrar la matriz A en la consola**

**System.out.println("Matriz A:");**

**mostrarMatriz(A);**

**// Mostrar la matriz transpuesta de B en la consola**

**System.out.println("Matriz transpuesta de B:");**

**mostrarMatriz(B);**

**// Enviar las matrices al servidor**

**try {**

**// Conectar al servidor**

**Socket socket = new Socket("localhost", 5000);**

**// Obtener los flujos de entrada y salida**

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

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

**//Enviar dimension**

**out.writeObject(N);**

**// Enviar la matriz A**

**out.writeObject(A);**

**// Enviar la matriz B**

**out.writeObject(B);**

**// Recibir la matriz transpuesta de B desde el servidor**

**double[][] B\_transpuesta = (double[][])in.readObject();**

**// Mostrar la matriz transpuesta de B recibida del servidor en la consola**

**System.out.println("Matriz transpuesta de B recibida del servidor:");**

**mostrarMatriz(B\_transpuesta);**

**// Cerrar los flujos y el socket**

**out.close();**

**in.close();**

**socket.close();**

**} catch (IOException | ClassNotFoundException e) {**

**e.printStackTrace();**

**}**

**}**

**// Método para mostrar una matriz en la consola**

**public static void mostrarMatriz(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] + " ");**

**}**

**System.out.println();**

**}**

**System.out.println();**

**}**

**}**

Hasta este punto ya tenemos la inicialización de matrices, el cliente que manda las matrices y el servidor que las recibe, sin embargo, al juntar las partes, en los sockets por donde se mandaban los datos, no se recibían correctamente o muchas veces se recibía basura. Esto se debía a que el cliente mandaba un tipo de dato y el servidor recibía otro tipo de dato. Asi que le pregunte a ChatGPT para ver si lo podía resolver

Mario Sebastian

por que en el cliente mando un entero por medio de sockets TCP en java y el servidor recibe un número negativo?

ChatGPT

Hay varias posibles razones por las cuales esto podría estar sucediendo, pero sin ver el código específico es difícil determinar la causa exacta. Algunas de las posibles causas son:

Problemas de conversión de tipos: Puede ser que en el cliente se esté enviando un valor de tipo entero sin signo (es decir, un valor positivo), pero en el servidor se está recibiendo como un entero con signo (que puede ser negativo). En este caso, es posible que deba realizarse una conversión explícita de tipos para asegurarse de que se transmita correctamente el valor.

Problemas con la representación binaria: Los datos se envían a través de la red en formato binario, y es posible que haya un problema con la forma en que se está codificando o decodificando el entero en cada extremo. Por ejemplo, el cliente podría estar usando un byte-ordering diferente al del servidor, lo que resultaría en un número negativo cuando se recibe en el servidor.

Problemas en el protocolo: Dependiendo del protocolo que se esté utilizando, puede haber ciertos campos reservados para indicar el tipo de datos que se están enviando. Si estos campos no se están configurando correctamente en el cliente, es posible que el servidor interprete los datos de manera incorrecta.

Para solucionar este problema, se recomienda revisar cuidadosamente el código del cliente y el servidor para verificar si hay errores de conversión o problemas de protocolo. También puede ser útil utilizar herramientas de depuración y monitoreo para identificar el problema con mayor precisión

Mario Sebastian

este es el código del cliente: **import java.io.\*;**

**import java.net.\*;**

**public class Nodo0 {**

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

**int N = 4; // la dimensión de la matriz N\*N**

**double[][] A = new double[N][N];**

**double[][] B = new double[N][N];**

**// Inicializar la matriz A**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**A[i][j] = 2\*i+j;**

**}**

**}**

**// Inicializar la matriz B y calcular su transpuesta**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**B[i][j] = 3\*i-j;**

**// Calcular la matriz transpuesta de B mientras se inicializa**

**B[j][i] = 3\*i-j;**

**}**

**}**

**// Mostrar la matriz A en la consola**

**System.out.println("Matriz A:");**

**mostrarMatriz(A);**

**// Mostrar la matriz transpuesta de B en la consola**

**System.out.println("Matriz transpuesta de B:");**

**mostrarMatriz(B);**

**// Enviar las matrices al servidor**

**try {**

**// Conectar al servidor**

**Socket socket = new Socket("localhost", 5000);**

**// Obtener los flujos de entrada y salida**

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

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

**//Enviar dimension**

**out.writeInt(N);**

**// Enviar la matriz A**

**out.writeObject(A);**

**// Enviar la matriz B**

**out.writeObject(B);**

**// Recibir la matriz transpuesta de B desde el servidor**

**double[][] B\_transpuesta = (double[][])in.readObject();**

**// Mostrar la matriz transpuesta de B recibida del servidor en la consola**

**System.out.println("Matriz transpuesta de B recibida del servidor:");**

**mostrarMatriz(B\_transpuesta);**

**// Cerrar los flujos y el socket**

**out.close();**

**in.close();**

**socket.close();**

**} catch (IOException | ClassNotFoundException e) {**

**e.printStackTrace();**

**}**

**}**

**// Método para mostrar una matriz en la consola**

**public static void mostrarMatriz(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] + " ");**

**}**

**System.out.println();**

**}**

**System.out.println();**

**}**

**}**

**este es el código del servidor: import java.io.\*;**

**import java.net.\*;**

**public class Servidor{**

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

**try {**

**// Creamos el socket del servidor y lo ponemos en espera de conexiones**

**ServerSocket servidor = new ServerSocket(5000);**

**System.out.println("Servidor en espera de conexión...");**

**// Aceptamos la conexión del cliente**

**Socket cliente = servidor.accept();**

**System.out.println("Cliente conectado");**

**// Obtenemos los flujos de entrada y salida**

**DataInputStream entrada = new DataInputStream(cliente.getInputStream());**

**DataOutputStream salida = new DataOutputStream(cliente.getOutputStream());**

**// Leemos la dimensión de las matrices**

**int n = entrada.readInt();**

**System.out.println("Dimensión de las matrices: " + n);**

**// Calculamos la dimensión de las submatrices**

**int m = n / 3;**

**// Creamos las matrices A y B**

**double[][] a = new double[m][m];**

**double[][] b = new double[m][m];**

**// Leemos la matriz A**

**System.out.println("Leyendo la matriz A...");**

**for (int i = 0; i < m; i++) {**

**for (int j = 0; j < m; j++) {**

**a[i][j] = entrada.readInt();**

**}**

**}**

**// Leemos la matriz B**

**System.out.println("Leyendo la matriz B...");**

**for (int i = 0; i < m; i++) {**

**for (int j = 0; j < m; j++) {**

**b[i][j] = entrada.readInt();**

**}**

**}**

**// Mostramos las matrices A y B en consola**

**System.out.println("Matriz A:");**

**mostrarMatriz(a);**

**System.out.println("Matriz B:");**

**mostrarMatriz(b);**

**// Cerramos los flujos y el socket del cliente**

**entrada.close();**

**salida.close();**

**cliente.close();**

**} catch (IOException e) {**

**e.printStackTrace();**

**}**

**}**

**public static void mostrarMatriz(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] + " ");**

**}**

**System.out.println();**

**}**

**}**

**}**

ChatGPT

El problema en este caso es que en el servidor se está leyendo los datos de las matrices como enteros utilizando el método readInt(), pero en el cliente se están enviando las matrices como objetos utilizando el método writeObject(). Esto causa una discrepancia en el tipo de datos esperado y leído en el servidor, lo que puede llevar a que los valores se interpreten de forma incorrecta.

Para solucionar este problema, se debe utilizar el método readObject() en el servidor para leer las matrices como objetos, y luego realizar la conversión correspondiente a un arreglo de dobles. En el cliente, se deben enviar las matrices como objetos utilizando el método writeObject().

Mario Sebastian

cómo se vería en código?

ChatGPT

En el código del servidor, el cliente está enviando un objeto de tipo double[][] para la matriz A y la matriz B a través de un ObjectOutputStream. Sin embargo, en el servidor, se están leyendo las matrices como una secuencia de enteros a través de un DataInputStream, lo cual no es compatible y es la causa del problema que estás experimentando.

Para solucionar este problema, el servidor también debe leer las matrices como objetos de tipo double[][]. Aquí te dejo una posible solución:

**import java.io.\*;**

**import java.net.\*;**

**public class Servidor{**

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

**try {**

**// Creamos el socket del servidor y lo ponemos en espera de conexiones**

**ServerSocket servidor = new ServerSocket(5000);**

**System.out.println("Servidor en espera de conexión...");**

**// Aceptamos la conexión del cliente**

**Socket cliente = servidor.accept();**

**System.out.println("Cliente conectado");**

**// Obtenemos los flujos de entrada y salida**

**ObjectOutputStream salida = new ObjectOutputStream(cliente.getOutputStream());**

**ObjectInputStream entrada = new ObjectInputStream(cliente.getInputStream());**

**// Leemos la dimensión de las matrices**

**int n = entrada.readInt();**

**System.out.println("Dimensión de las matrices: " + n);**

**// Calculamos la dimensión de las submatrices**

**int m = n / 3;**

**// Creamos las matrices A y B**

**double[][] a = (double[][])entrada.readObject();**

**double[][] b = (double[][])entrada.readObject();**

**// Mostramos las matrices A y B en consola**

**System.out.println("Matriz A:");**

**mostrarMatriz(a);**

**System.out.println("Matriz B:");**

**mostrarMatriz(b);**

**// Cerramos los flujos y el socket del cliente**

**entrada.close();**

**salida.close();**

**cliente.close();**

**} catch (IOException | ClassNotFoundException e) {**

**e.printStackTrace();**

**}**

**}**

**public static void mostrarMatriz(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] + " ");**

**}**

**System.out.println();**

**}**

**}**

**}**

En el código del servidor, se cambió la creación de flujos de entrada y salida a ObjectOutputStream y ObjectInputStream, respectivamente. También se cambió la forma de leer las matrices A y B, ahora se están leyendo como objetos de tipo double[][] a través del método readObject() del ObjectInputStream.

Teniendo es parte solucionada, pasamos a la parte de la división de matrices en 3 partes. En otra conversación le pedí lo siguiente:

Mario Sebastian

el siguiente programa en java inicializa 2 matrices A, B, obtiene la transpuesta de B y manda dichas matrices por un socket TCP, modifica el código para obtener 3 submatrices a partir de la matriz A y B transpuesta, es decir vas a dividir la matriz A y B en 3 filas, cada división será una nueva submatriz A1,A2,A3. Lo mismo para B, cada 3 filas será una nueva submatriz Bt1,Bt2,Bt3. El código es el siguiente: **import java.io.\*;**

**import java.net.\*;**

**public class Nodo0 {**

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

**int N = 4; // la dimensión de la matriz N\*N**

**double[][] A = new double[N][N];**

**double[][] B = new double[N][N];**

**// Inicializar la matriz A**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**A[i][j] = 2\*i+j;**

**}**

**}**

**// Inicializar la matriz B y calcular su transpuesta**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**B[i][j] = 3\*i-j;**

**// Calcular la matriz transpuesta de B mientras se inicializa**

**B[j][i] = 3\*i-j;**

**}**

**}**

**// Mostrar la matriz A en la consola**

**System.out.println("Matriz A:");**

**mostrarMatriz(A);**

**// Mostrar la matriz transpuesta de B en la consola**

**System.out.println("Matriz transpuesta de B:");**

**mostrarMatriz(B);**

**// Enviar las matrices al servidor**

**try {**

**// Conectar al servidor**

**Socket socket = new Socket("localhost", 5000);**

**// Obtener los flujos de entrada y salida**

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

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

**//Enviar dimension**

**out.writeInt(N);**

**// Enviar la matriz A**

**out.writeObject(A);**

**// Enviar la matriz B**

**out.writeObject(B);**

**// Recibir la matriz transpuesta de B desde el servidor**

**double[][] B\_transpuesta = (double[][])in.readObject();**

**// Mostrar la matriz transpuesta de B recibida del servidor en la consola**

**System.out.println("Matriz transpuesta de B recibida del servidor:");**

**mostrarMatriz(B\_transpuesta);**

**// Cerrar los flujos y el socket**

**out.close();**

**in.close();**

**socket.close();**

**} catch (IOException | ClassNotFoundException e) {**

**e.printStackTrace();**

**}**

**}**

**// Método para mostrar una matriz en la consola**

**public static void mostrarMatriz(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] + " ");**

**}**

**System.out.println();**

**}**

**System.out.println();**

**}**

**}**

Para dividir la matriz A y B transpuesta en tres submatrices cada una, podemos modificar el código de la siguiente manera:

**import java.io.\*;**

**import java.net.\*;**

**public class Nodo0 {**

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

**int N = 4; // la dimensión de la matriz N\*N**

**double[][] A = new double[N][N];**

**double[][] B = new double[N][N];**

**// Inicializar la matriz A**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**A[i][j] = 2\*i+j;**

**}**

**}**

**// Inicializar la matriz B y calcular su transpuesta**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**B[i][j] = 3\*i-j;**

**// Calcular la matriz transpuesta de B mientras se inicializa**

**B[j][i] = 3\*i-j;**

**}**

**}**

**// Mostrar la matriz A en la consola**

**System.out.println("Matriz A:");**

**mostrarMatriz(A);**

**// Mostrar la matriz transpuesta de B en la consola**

**System.out.println("Matriz transpuesta de B:");**

**mostrarMatriz(B);**

**// Dividir la matriz A en 3 submatrices**

**double[][] A1 = new double[N/3][N];**

**double[][] A2 = new double[N/3][N];**

**double[][] A3 = new double[N/3][N];**

**for(int i=0; i<N; i++) {**

**if(i < N/3) {**

**A1[i] = A[i];**

**} else if(i < 2\*N/3) {**

**A2[i-N/3] = A[i];**

**} else {**

**A3[i-2\*N/3] = A[i];**

**}**

**}**

**// Dividir la matriz transpuesta de B en 3 submatrices**

**double[][] Bt = new double[N][N];**

**for(int i=0; i<N; i++) {**

**for(int j=0; j<N; j++) {**

**Bt[i][j] = B[j][i];**

**}**

**}**

**double[][] Bt1 = new double[N/3][N];**

**double[][] Bt2 = new double[N/3][N];**

**double[][] Bt3 = new double[N/3][N];**

**for(int i=0; i<N; i++) {**

**if(i < N/3) {**

**Bt1[i] = Bt[i];**

**} else if(i < 2\*N/3) {**

**Bt2[i-N/3] = Bt[i];**

**} else {**

**Bt3[i-2\*N/3] = Bt[i];**

**}**

**}**

**// Enviar las matrices al servidor**

**try {**

**// Conectar al servidor**

**Socket socket = new Socket("localhost", 5000);**

**// Obtener los flujos de entrada y salida**

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

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

**//Enviar dimension**

**out.writeInt(N);**

**// Enviar la submatriz A1**

**out.writeObject(A1);**

**// Enviar la submatriz A2**

**out.writeObject(A2);**

**// Enviar la submatriz A3**

**out.writeObject(A3**

En esta parte se trabo porque el código fue algo largo y no le alcanzaron los tokens, sin embargo, la parte que nos interesaba esta al principio y si la alcanzó a escribir, por lo tanto, solamente copie los fragmentos clave del código y lo agregue al código base que ya se tenía.

Ahora nos faltaba la parte de enviar las matrices que tenemos divididas a 3 servidores, en otra conversación le pedí lo siguiente:

Mario Sebastian

tengo un cliente en java que usa sockets TCP, quiero enviar datos a 3 servidores al mismo tiempo, que debo hacer?

ChatGPT

Para enviar datos a tres servidores al mismo tiempo en Java utilizando sockets TCP, puedes crear tres instancias de la clase Socket y conectarlas a los servidores correspondientes. Luego, puedes enviar los datos a cada uno de los servidores utilizando los métodos getOutputStream() y write() de cada una de las instancias de Socket.

Aquí te dejo un ejemplo básico que ilustra cómo podrías enviar datos a tres servidores utilizando threads para hacerlo simultáneamente:

**import java.io.\*;**

**import java.net.\*;**

**public class ClienteTCP {**

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

**String servidor1 = "localhost";**

**int puerto1 = 8080;**

**String servidor2 = "localhost";**

**int puerto2 = 8081;**

**String servidor3 = "localhost";**

**int puerto3 = 8082;**

**try {**

**// Conectarse a los servidores**

**Socket socket1 = new Socket(servidor1, puerto1);**

**Socket socket2 = new Socket(servidor2, puerto2);**

**Socket socket3 = new Socket(servidor3, puerto3);**

**// Crear threads para enviar datos simultáneamente**

**Thread t1 = new Thread(new Runnable() {**

**public void run() {**

**try {**

**OutputStream outputStream = socket1.getOutputStream();**

**outputStream.write("Datos para el servidor 1".getBytes());**

**outputStream.flush();**

**socket1.close();**

**} catch (IOException e) {**

**e.printStackTrace();**

**}**

**}**

**});**

**Thread t2 = new Thread(new Runnable() {**

**public void run() {**

**try {**

**OutputStream outputStream = socket2.getOutputStream();**

**outputStream.write("Datos para el servidor 2".getBytes());**

**outputStream.flush();**

**socket2.close();**

**} catch (IOException e) {**

**e.printStackTrace();**

**}**

**}**

**});**

**Thread t3 = new Thread(new Runnable() {**

**public void run() {**

**try {**

**OutputStream outputStream = socket3.getOutputStream();**

**outputStream.write("Datos para el servidor 3".getBytes());**

**outputStream.flush();**

**socket3.close();**

**} catch (IOException e) {**

**e.printStackTrace();**

**}**

**}**

**});**

**// Iniciar los threads**

**t1.start();**

**t2.start();**

**t3.start();**

**} catch (IOException e) {**

**e.printStackTrace();**

**}**

**}**

**}**

Este ejemplo crea tres instancias de Socket para conectarse a los servidores en los puertos 8080, 8081 y 8082 respectivamente. Luego, se crean tres threads que envían datos a cada uno de los servidores utilizando sus respectivas instancias de Socket. Los threads se inician utilizando el método start() y corren simultáneamente.

Es importante destacar que en este ejemplo se están cerrando las conexiones a los servidores después de enviar los datos. Si necesitas mantener la conexión abierta para recibir una respuesta del servidor, deberás ajustar el código en consecuencia.

Posteriormente pasamos al algoritmo de la multiplicación, para esto se lo pedí de la siguiente manera tratando de no utilizar la palabra matriz debido a que no tuviera la idea de que estuviéramos trabajando con matrices, en su lugar utilice la palabra arreglo bidimensional.

Mario Sebastian

por favor haz un programa en java que multiplique los elementos de 2 arreglos bidimensionales de tamaño N/3\*N , sea A[i][j] y B[i][j] elementos de los arreglos, la multiplicación será A[i][j] \* B[i][j]

ChatGPT

Por supuesto, aquí tienes un programa en Java que multiplica los elementos de dos arreglos bidimensionales de tamaño N/3 \* N y almacena el resultado en un tercer arreglo:

**import java.util.Random;**

**public class MultiplicacionArreglos {**

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

**// Definir el tamaño del arreglo**

**int n = 9; // Debe ser divisible por 3**

**// Inicializar los arreglos A y B con valores aleatorios**

**double[][] A = new double[n/3][n];**

**double[][] B = new double[n/3][n];**

**Random rand = new Random();**

**for (int i = 0; i < n/3; i++) {**

**for (int j = 0; j < n; j++) {**

**A[i][j] = rand.nextDouble() \* 10.0;**

**B[i][j] = rand.nextDouble() \* 10.0;**

**}**

**}**

**// Imprimir los arreglos A y B**

**System.out.println("Arreglo A:");**

**imprimirArreglo(A);**

**System.out.println("Arreglo B:");**

**imprimirArreglo(B);**

**// Multiplicar los elementos de A y B y almacenar el resultado en C**

**double[][] C = new double[n/3][n];**

**for (int i = 0; i < n/3; i++) {**

**for (int j = 0; j < n; j++) {**

**C[i][j] = A[i][j] \* B[i][j];**

**}**

**}**

**// Imprimir el arreglo resultante C**

**System.out.println("Arreglo C:");**

**imprimirArreglo(C);**

**}**

**// Método para imprimir un arreglo bidimensional**

**public static void imprimirArreglo(double[][] arr) {**

**for (int i = 0; i < arr.length; i++) {**

**for (int j = 0; j < arr[i].length; j++) {**

**System.out.print(arr[i][j] + " ");**

**}**

**System.out.println();**

**}**

**System.out.println();**

**}**

**}**

Sin embargo, estaba mal porque infirió que estábamos multiplicando matrices y en vez de hacerlo renglón por renglón lo hacia columna por columna, esa parte la modifique para que fuera renglón por renglón.

Al final hice algunas modificaciones extras al código que ya se tenía.