UNIVERSIDAD
NACIONAL
DE COLOMBIA

PROYECTO **CULTURAL, CIENTÍFICO Y COLECTIVO** DE NACIÓN

Uso de Programación Genérica en Java: Ejemplo Práctico.

Juan felipe prado gordillo

Facultad de ingeniería –Departamento de sistemas- Sede Bogota

Universidad Nacional de Colombia

PROYECTO **CULTURAL, CIENTÍFICO Y COLECTIVO** DE NACIÓN

Clase genérica.

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

        Conexion conexion = new Conexion();

        List<Pair<String, Integer>> persona = new ArrayList<>();
        List<Pair<String, String>> datosSalud = new ArrayList<>();

        persona.add(new Pair<>("Camila", 30));
        persona.add(new Pair<>("Ana", 25));

        datosSalud.add(new Pair<>("EPS Salud", "1990-01-01"));
        datosSalud.add(new Pair<>("EPS Vida", "1995-05-15"));

        for (int i = 0; i < persona.size(); i++) {
            conexion.insertarUsuario(persona.get(i), datosSalud.get(i));
        }
        Pair<String, Integer> person = persona.get(1);
        Pair<String, String> datos = datosSalud.get(1);

        System.out.println("getFirst: " + person.getFirst());
        System.out.println("getSecond: " + datos.getSecond());

        System.out.println("Lista de Personas: " + persona.toString());
        System.out.println("Lista de Datos de Salud: " + datosSalud.toString());
        conexion.cerrarConexion(); // Llama al método para cerrar la conexión
    }
}
```

1) Clase “Clase_Generica”

- **Función Principal:** Esta clase contiene el método “main”, que es el punto de entrada del programa.
- **Conexión a la Base de Datos:** Se crea una instancia de “Conexión” para gestionar la conexión a una base de datos MySQL.

Listas Genéricas:

- **List<Pair<String, Integer>> persona**: Almacena pares de nombres y edades.
- **List<Pair<String, String>> datosSalud:** Almacena pares de EPS y fechas de nacimiento.

Añadir Datos:

- Se añaden pares a las listas y se inserta cada usuario en la base de datos utilizando un bucle.

Acceso a Elementos:

- Se accede a elementos específicos de las listas para imprimir su contenido, mostrando cómo se utilizan los métodos de la clase “Pair”

Cierre de Conexión:

- Se cierra la conexión a la base de datos al final del proceso.

Clase Pair.

```java
package clase_generica;


public class Pair<T, U> {

    private T first;
    private U second;

    public Pair(T first, U second) {
        this.first = first;
        this.second = second;
    }

    public T getFirst() {
        return first;
    }

    public U getSecond() {
        return second;
    }

    @Override
    public String toString() {
        return "(" + first + ", " + second + ")";
    }
}
```

2) Clase “Pair<T, U>”

Definición Genérica:

- Esta clase permite almacenar un par de objetos de tipos diferentes.

Atributos:

- Tiene dos atributos: “first” y “second”.

Constructor y Métodos:

- Constructor para inicializar los atributos.
- Métodos “getFirst()” y “getSecond()” para acceder a los elementos del par.
- Método “toString()” para representar el par en forma de cadena.

Clase Conexion.

```java
public class Conexion {
    private Connection con;

    public Conexion() {
        try {

            Class.forName("com.mysql.jdbc.Driver");

            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/Usuarios", "root", "");
            System.out.println("Conexión establecida");
        } catch (Exception e) {

            System.out.println("Conexión no establecida: " + e);
        }
    }

    public Connection obtenerConexion() {
        return con;
    }

    public void insertarUsuario(Pair<String, Integer> persona, Pair<String, String> datosSalud) {

        String sql = "INSERT INTO Usuarios (nombre, edad, eps, fecha_nacimiento) VALUES (?, ?, ?, ?)";
        try (PreparedStatement pstmt = obtenerConexion( ).prepareStatement(sql)) {
            pstmt.setString(1, persona.getFirst()); // Nombre
            pstmt.setInt(2, persona.getSecond());     // Edad
            pstmt.setString(3, datosSalud.getFirst()); // EPS
            pstmt.setString(4, datosSalud.getSecond()); // Fecha de nacimiento
            pstmt.executeUpdate(); // Ejecuta la actualización
            System.out.println("Usuario insertado correctamente"); // Mensaje
        } catch (SQLException e) {

            e.printStackTrace();
        }
    }

    public void cerrarConexion() {
        if (con != null) {
            try {
                con.close();
                System.out.println("Conexión cerrada");
            } catch (SQLException e) {

                System.out.println("Error al cerrar la conexión: " + e);
            }
        }
    }
}
```

Gestión de la Base de Datos:

- Establece una conexión a la base de datos en el constructor, manejando excepciones.
- Método “insertarUsuario()” que utiliza “PreparedStatement” para insertar datos de manera segura.

Cierre de Conexión:

- Método para cerrar la conexión cuando ya no es necesaria.

Flujo de ejecución.

1. **Inicio:**

- Se inicia el programa y se establece la conexión a la base de datos.

2. **Creación de Listas:**

- Se crean listas para almacenar información de personas y datos de salud.

3. **Inserción de Datos:**

- Se añaden datos a las listas y se insertan en la base de datos.

4. **Acceso y Presentación:**

- Se accede a elementos específicos y se imprimen las listas completas.

5. **Cierre de Conexión:**

- Se cierra la conexión a la base de datos.

4. Conclusiones

- La programación genérica proporciona una forma eficiente de manejar diferentes tipos de datos en Java, mejorando la reutilización y la seguridad del código.

Gracias