
UNIVERSIDAD MARIANO GALVEZ DE GUATEMALA

TAREA

# QUEUE DE CANCIONES

**MILDRED HANANI PINEDA PINEDA**
**0905-22-5811**

PROGRAMACION 3

FEBRERO 2026

# Parte A — Librería de Cola Propia

**PASO 1**

Verificar que versión de **Maven** y que versión de **Java** esta instalado con los siguientes comandos

**java -version**

**mvn -version**

```
PS C:\Users\COMPUFIRE> java -version
openjdk version "11.0.30" 2026-01-20
OpenJDK Runtime Environment Temurin-11.0.30+7 (build 11.0.30+7)
OpenJDK 64-Bit Server VM Temurin-11.0.30+7 (build 11.0.30+7, mixed mode)
PS C:\Users\COMPUFIRE> mvn -version
Apache Maven 3.9.12 (848fbb4bf2d427b72bdb2471c22fced7ebd9a7a1)
Maven home: C:\Maven\apache-maven-3.9.12
Java version: 11.0.30, vendor: Eclipse Adoptium, runtime: C:\Program Files\Eclipse Adoptium\jdk-11.0.30.7-hotspot
Default locale: es_ES, platform encoding: Cp1252
OS name: "windows 11", version: "10.0", arch: "amd64", family: "windows"
PS C:\Users\COMPUFIRE>
```

**PASO 2**

**Node.java** no es genérico

queue/pom.xml | QueueLinked.java | Node.java | Main.java

```java
package umg.edu.gt.data_structure.queue.manual;


public class Node {
    int value;
    Node next;

    Node(int value) {
        this.value = value;
    }
}
```

**Node.java** genérico

queue/pom.xml | QueueLinked.java | *Node.java | Main.java

```java
package umg.edu.gt.data_structure.queue.manual;

public class Node<T> {
    T value;
    Node<T> next;

    public Node(T value) {
        this.value = value;
        this.next = null;
    }
}
```

**PASO 3**

**QueueLiken.java** no genérico

queue/pom.xml | QueueLinked.java | *Node.java | Main.java

```java
package umg.edu.gt.data_structure.queue.manual;

public class QueueLinked {

    private Node head; // primero en salir
    private Node tail; // ultimo en entrar
    private int size;

    public boolean isEmpty() {
        return size == 0;
    }

    public int size() {
        return size;
    }

    // add
    public void enqueue(int value) {
        Node node = new Node(value);
        if (isEmpty()) {
            head = tail = node;
        } else {
            tail.next = node;
            tail = node;
        }
        size++;
    }

    // delete
    public int dequeue() {
        if (isEmpty())
            throw new IllegalStateException("Queue underflow");

        int value = head.value;
        head = head.next;
        if (head == null) {
            tail = null;
        }
        size--;
        return value;
    }

    public int peek() {
        if (isEmpty()) {
            throw new IllegalStateException("Queue underflow");
        }
        return head.value;
    }

}
```

**QueueLikend.java** genérico

queue/pom.xml | *QueueLinked.java | *Node.java | Main.java

```java
package umg.edu.gt.data_structure.queue.manual;

public class QueueLinked<T> {

    private Node<T> head; // primero en salir
    private Node<T> tail; // ultimo en entrar
    private int size;

    public boolean isEmpty() {
        return size == 0;
    }

    public int size() {
        return size;
    }

    // Agregar elemento - O(1)
    public void enqueue(T item) {
        Node<T> node = new Node<>(item);
        if (isEmpty()) {
            head = tail = node;
        } else {
            tail.next = node;
            tail = node;
        }
        size++;
    }

    // Eliminar y retornar el primero - O(1)
    public T dequeue() {
        if (isEmpty()) {
            throw new IllegalStateException("Queue underflow: cannot dequeue from empty queue");
        }

        T value = head.value;
        head = head.next;
        if (head == null) {
            tail = null;
        }
        size--;
        return value;
    }

    // Ver el primero sin eliminar
    public T peek() {
        if (isEmpty()) {
            throw new IllegalStateException("Queue underflow: cannot peek from empty queue");
        }
        return head.value;
    }
}
```

**PASO 4**

Actualizar **Main.java**

queue/pom.xml | Queue.java | Node.java | *Main.java

```java
package umg.edu.gt.data_structure.queue;
import java.util.ArrayDeque;
import java.util.Queue;
import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;
import umg.edu.gt.data_structure.queue.manual.Queue;

public class Main {

    private static final Logger logger = LogManager.getLogger(Main.class);
    public static void main(String[] args) {

        Queue<String> q = new ArrayDeque<>();
        q.offer("A");
        q.offer("B");
        q.offer("C");

        System.out.println(q.peek()); // A
        System.out.println(q.poll()); // A
        System.out.println(q.poll()); // B
        System.out.println(q.isEmpty()); // false
        logger.info("este es mi mensaje");
        logger.error("este es mi mensaje");

        Queue queue = new Queue();
        queue.enqueue(10);
        queue.enqueue(20); //queue.enqueuePriority(50);
        queue.enqueue(30);

        System.out.println(queue.peek());   // 10
        System.out.println(queue.dequeue()); // 10
        System.out.println(queue.dequeue()); // 20
        System.out.println(queue.size());    // 1
    }
}
```

Actualizado **Main.java**

queue/pom.xml | Queue.java | Node.java | *Main.java

```java
package umg.edu.gt.data_structure.queue;
import umg.edu.gt.data_structure.queue.manual.Queue;
public class Main {

    public static void main(String[] args) {

        // Prueba con Strings
        Queue<String> queueStrings = new Queue<>();
        queueStrings.enqueue("A");
        queueStrings.enqueue("B");
        queueStrings.enqueue("C");

        System.out.println("=== Prueba con Strings ===");
        System.out.println("Peek: " + queueStrings.peek()); // A
        System.out.println("Dequeue: " + queueStrings.dequeue()); // A
        System.out.println("Dequeue: " + queueStrings.dequeue()); // B
        System.out.println("Size: " + queueStrings.size()); // 1
        System.out.println("Dequeue: " + queueStrings.dequeue()); // C
        System.out.println("IsEmpty: " + queueStrings.isEmpty()); // true

        // Prueba con Integers
        System.out.println("\n=== Prueba con Integers ===");
        Queue<Integer> queueIntegers = new Queue<>();
        queueIntegers.enqueue(10);
        queueIntegers.enqueue(20);
        queueIntegers.enqueue(30);

        System.out.println("Peek: " + queueIntegers.peek()); // 10
        System.out.println("Dequeue: " + queueIntegers.dequeue()); // 10
        System.out.println("Size: " + queueIntegers.size()); // 2

        System.out.println("\n=== Todas las pruebas completadas ===");
    }
}
```

**PASO 5**

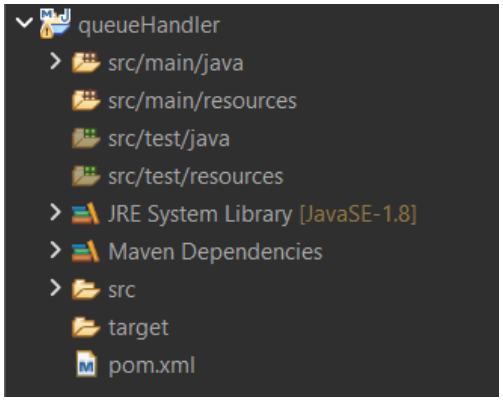
Instalar la librería localmente en **Maven**

Con el siguiente comando en la terminal **mvn clean install**

```
[INFO]
[INFO] --- surefire:3.2.5:test (default-test) @ queue ---
[INFO]
[INFO] --- jar:3.4.1:jar (default-jar) @ queue ---
[INFO] Building jar: C:\Users\COMPUFIRE\Downloads\queue\queue\target\queue-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- install:3.1.2:install (default-install) @ queue ---
[INFO] Installing C:\Users\COMPUFIRE\Downloads\queue\queue\pom.xml to C:\Users\COMPUFIRE\.m2\repository\umg\edu\gt\data-
structure\queue\queue\0.0.1-SNAPSHOT\queue-0.0.1-SNAPSHOT.pom
[INFO] Installing C:\Users\COMPUFIRE\Downloads\queue\queue\target\queue-0.0.1-SNAPSHOT.jar to C:\Users\COMPUFIRE\.m2\rep
ository\umg\edu\gt\data-structure\queue\queue\0.0.1-SNAPSHOT\queue-0.0.1-SNAPSHOT.jar
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  3.047 s
[INFO] Finished at: 2026-02-24T11:56:16-06:00
[INFO] ------------------------------------------------------------------------
PS C:\Users\COMPUFIRE\Downloads\queue\queue>
```

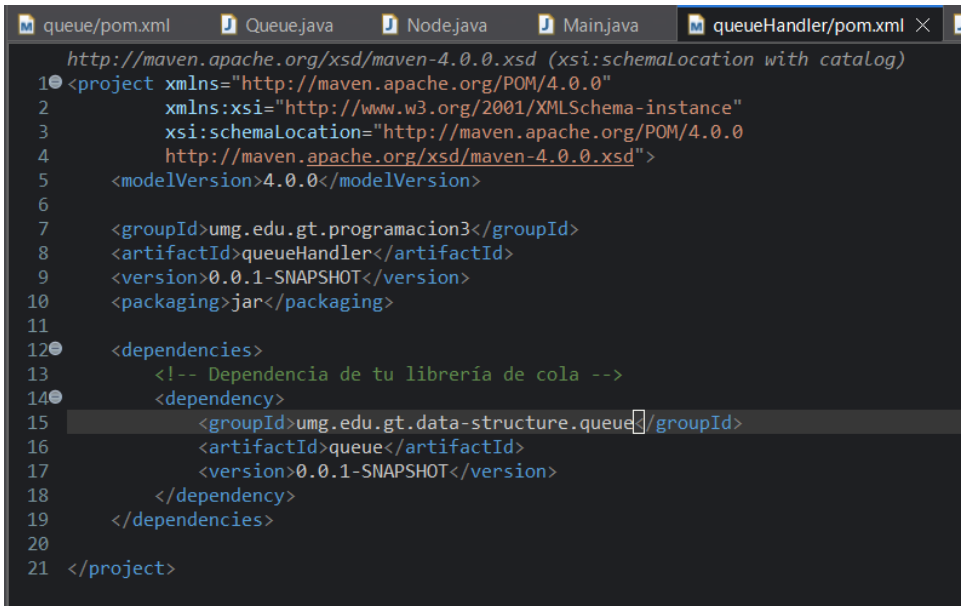
**PASO 6**

Crear un **New Project** llamado **queueHandler**




**PASO 7**

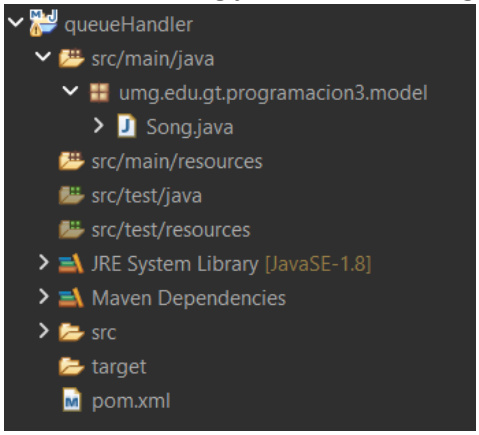
Agregar este código a **pom.xml**




**PASO 8**

En el Package Explorer, haz clic derecho sobre **src/main/java** (dentro del proyecto queueHandler) Selecciona New → Package En "Name", escribe: **umg.edu.gt.programacion3.model**

Crear la clase **Song.java** dentro de **umg.edu.gt.programacion3.model**



**PASO 9**

Colocar este código en la clase **Song.java**

queue/pom.xml | Queue.java | Node.java | Main.java | queueHandler/pom.xml | Song.java

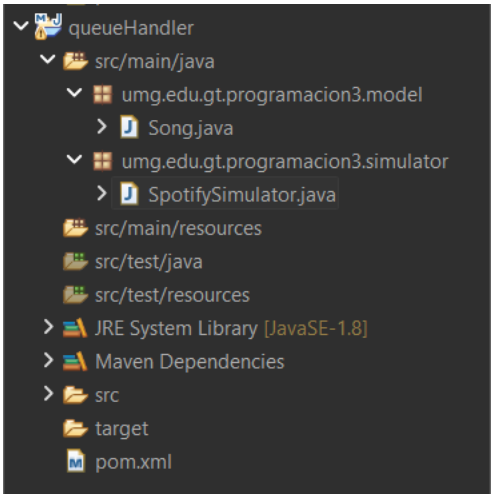
```java
package umg.edu.gt.programacion3.model;

public class Song {
    private String title;
    private String artist;
    private int duration; // en segundos (5-30)
    private int priority; // 1 = alta, 2 = normal

    public Song(String title, String artist, int duration, int priority) {
        this.title = title;
        this.artist = artist;
        this.duration = duration;
        this.priority = priority;
    }

    public String getTitle() {
        return title;
    }

    public String getArtist() {
        return artist;
    }

    public int getDuration() {
        return duration;
    }
```

```java
    public int getPriority() {
        return priority;
    }

    @Override
    public String toString() {
        return title + " - " + artist + " (" + duration + "s)";
    }
}
```

**PASO 10**

En el Package Explorer, haz clic derecho sobre **src/main/java** (dentro del proyecto queueHandler) Selecciona New → Package En "Name", escribe:
**umg.edu.gt.programacion3.simulator**

Crear la clase **SpotifySimulator.java** dentro de **umg.edu.gt.programacion3.simulator**



**PASO 11**

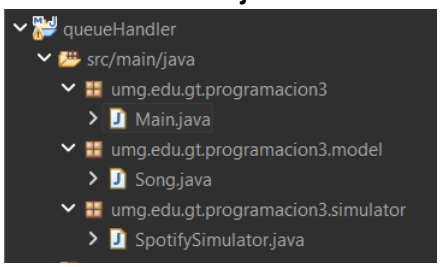
Colocar este código en la clase **SpotifySimulator.java**

queue/pom.xml | Queue.java | Node.java | Main.java | queueHandler/pom.xml | Song.java | SpotifySimulator.java

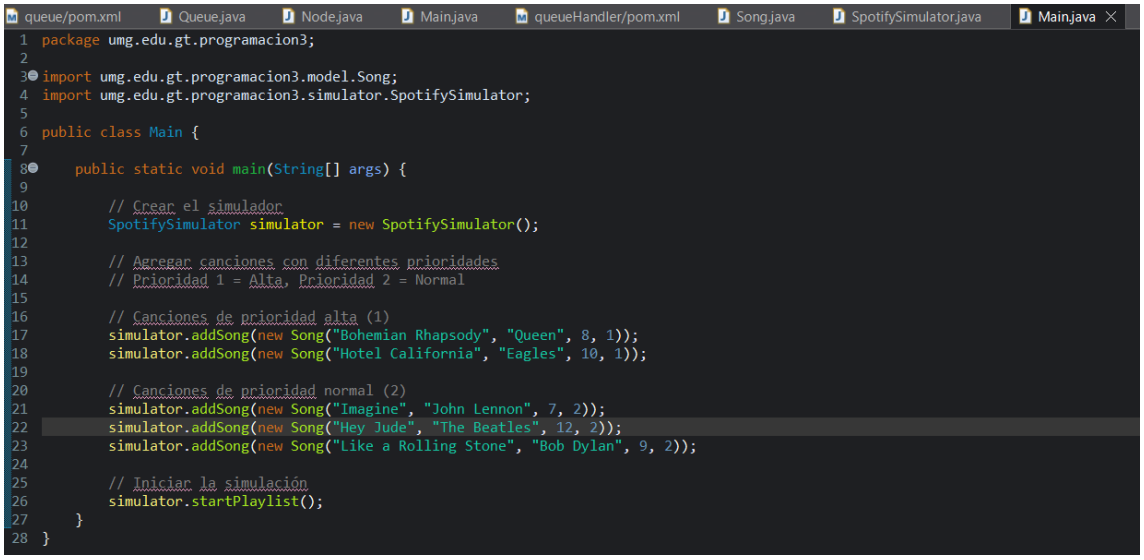
```java
package umg.edu.gt.programacion3.simulator;

import umg.edu.gt.data_structure.queue.manual.Queue;
import umg.edu.gt.programacion3.model.Song;

public class SpotifySimulator {

    private Queue<Song> highPriorityQueue;  // Prioridad 1
    private Queue<Song> normalPriorityQueue; // Prioridad 2

    public SpotifySimulator() {
        highPriorityQueue = new Queue<>();
        normalPriorityQueue = new Queue<>();
    }

    // Agregar canción según su prioridad
    public void addSong(Song song) {
        if (song.getPriority() == 1) {
            highPriorityQueue.enqueue(song);
        } else {
            normalPriorityQueue.enqueue(song);
        }
    }
    // Obtener siguiente canción (prioridad primero)
    private Song getNextSong() {
        if (!highPriorityQueue.isEmpty()) {
            return highPriorityQueue.dequeue();
        } else if (!normalPriorityQueue.isEmpty()) {
            return normalPriorityQueue.dequeue();
        }
        return null;
    }

    // Verificar si hay canciones
    public boolean hasSongs() {
        return !highPriorityQueue.isEmpty() || !normalPriorityQueue.isEmpty();
    }

    // Reproducir canción con simulación segundo a segundo
    public void playSong(Song song) {
        System.out.println("[LOG] Now playing: " + song.getTitle() + " - " +
                           song.getArtist() + " (" + song.getDuration() + "s)");

        // Simular reproducción segundo a segundo
        for (int i = 1; i <= song.getDuration(); i++) {
            System.out.println("[LOG] Playing: " + song.getTitle() + " | " + i + "s / " +
                               song.getDuration() + "s");
            try {
                Thread.sleep(1000); // 1 segundo real = 1 segundo de simulación
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }

        System.out.println("[LOG] Finished: " + song.getTitle());
    }

    // Iniciar playlist
    public void startPlaylist() {
        System.out.println("[LOG] Starting playlist...");

        while (hasSongs()) {
            Song song = getNextSong();
            if (song != null) {
                playSong(song);
            }
        }

        System.out.println("[LOG] Playlist finished.");
    }
}
    public int getPriority() {
        return priority;
    }

    @Override
    public String toString() {
        return title + " - " + artist + " (" + duration + "s)";
    }
}
```

**PASO 12**

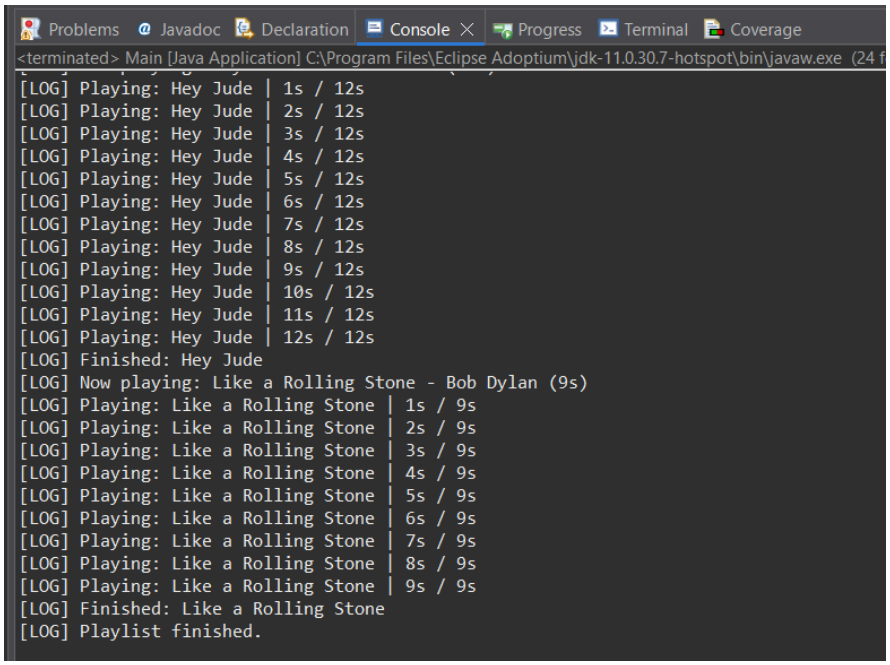
En el Package Explorer, haz clic derecho sobre **src/main/java** (dentro del proyecto queueHandler) Selecciona New → Package En "Name", escribe: **umg.edu.gt.programacion3**
Crear la clase **Main.java** dentro de **umg.edu.gt.programacion3**




**PASO 11**
Colocar este código en la clase **Main.java**




**PASO 12**
Simulacion de Playlist en Consola



evidencia_extensiones_parteD