Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información.
CI–2693 – Laboratorio de Algoritmos y Estructuras III.
Enero–Marzo 2026

Proyecto I: Grafo
Implementación con listas de adyacencias
(25 pts)

Introducción

Sea **G = (V, E)** un grafo dirigido, una posible forma de representarlo es usando **listas de adyacencias**. En esta implementación se mantiene para cada vértice una lista de sus sucesores. El objetivo de este proyecto es implementar una clase concreta para la interfaz `Grafo<T>` utilizando esta estructura en el lenguaje Kotlin, aprovechando sus características de nulabilidad y funciones de extensión.

Interfaz Grafo

La interfaz `Grafo<T>` debe ser genérica y definir los siguientes métodos:

agregarVertice

```
fun agregarVertice(v: T): Boolean
```

Recibe un objeto de tipo `T` y lo agrega al conjunto de vértices del grafo. Retorna `true` si el vértice fue agregado con éxito y `false` si el vértice ya existía.

conectar

```
fun conectar(desde: T, hasta: T): Boolean
```

Crea un arco dirigido desde el vértice `desde` hacia el vértice `hasta`. Retorna `true` si la conexión se realizó con éxito. Si alguno de los vértices no existe, retorna `false`.

contiene

```
fun contiene(v: T): Boolean
```

Retorna `true` si el vértice `v` pertenece al conjunto de vértices del grafo.

obtenerArcosSalida

```
fun obtenerArcosSalida(v: T): List<T>
```

Retorna una lista con todos los vértices `u` tales que existe un arco `(v, u)`. Si el vértice no existe, retorna una lista vacía.

obtenerArcosEntrada

```
fun obtenerArcosEntrada(v: T): List<T>
```

Retorna una lista con todos los vértices `u` tales que existe un arco `(u, v)`.

eliminarVertice

```
fun eliminarVertice(v: T): Boolean
```

Elimina el vértice `v` y todos los arcos asociados (tanto de entrada como de salida). Retorna `true` si la eliminación fue exitosa.

tamano

```
fun tamano(): Int
```

Retorna la cantidad de vértices en el grafo ($|V|$).

subgrafo

```
fun subgrafo(vertices: Collection<T>): Grafo<T>
```

Retorna una nueva instancia de grafo que contiene únicamente los vértices de la colección recibida que ya pertenecían al grafo original, junto con los arcos que los conectan entre sí.

Entrega

Este proyecto se realizará en equipos de 2 integrantes y debe ser gestionado mediante un repositorio en *GitHub*. Su entrega debe incluir:

- La interfaz `Grafo.kt`.
- La implementación `ListaAdyacenciaGrafo.kt`.
- Un archivo `README.md` con los nombres y carnets de los integrantes, instrucciones de ejecución, una tabla con la complejidad computacional (Big O) de cada método implementado (dando una breve justificación de cada método). Además, deben dar explicaciones sobre sus decisiones de implementación (similar a un informe).

Se recomienda el uso de las colecciones nativas de Kotlin (`MutableMap`, `MutableList`) y el uso de funciones de orden superior como `filter` o `map` para simplificar la lógica.

La fecha límite de entrega es el **viernes 13 de febrero de 2026 a las 11:59 pm**.