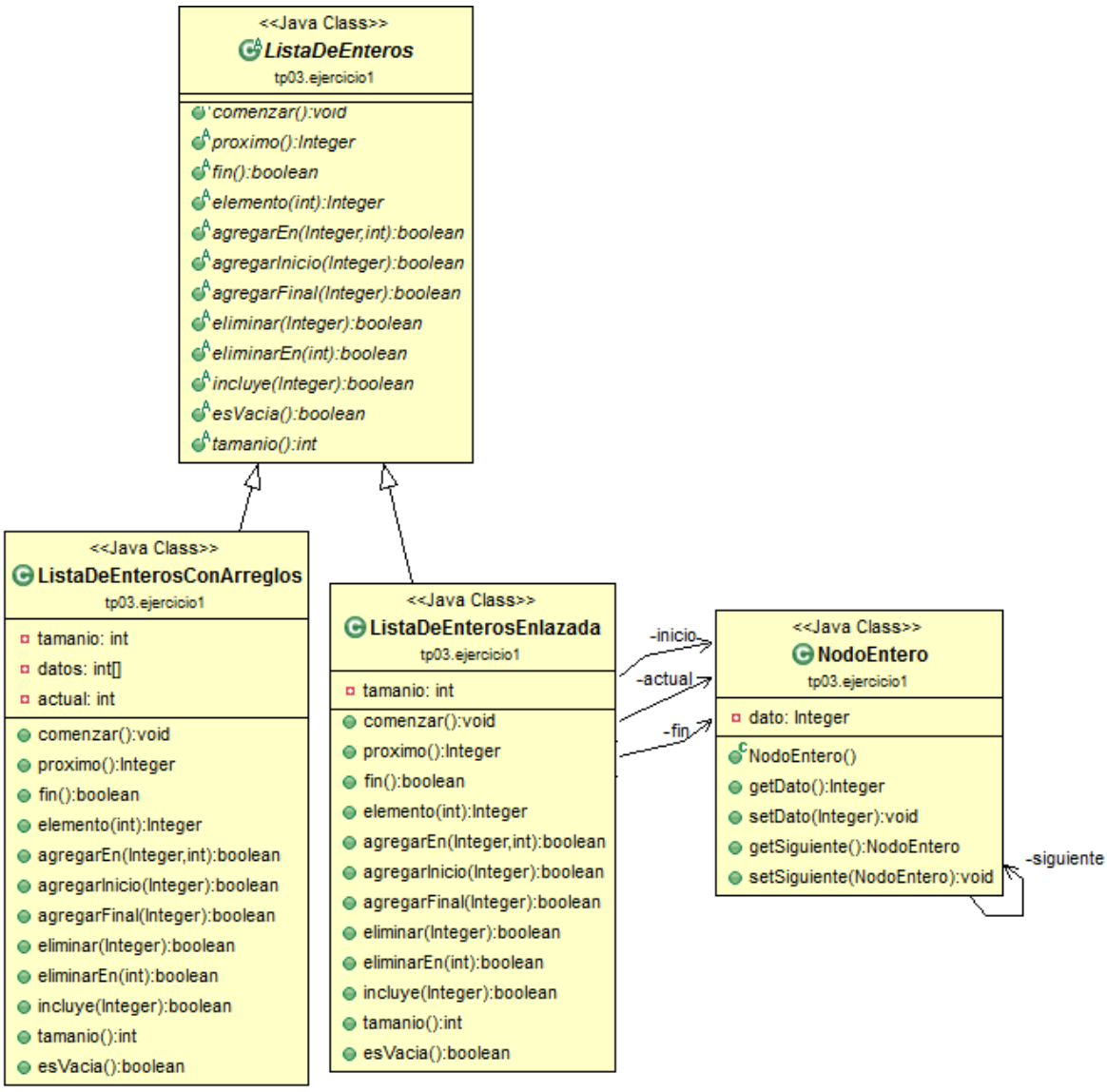
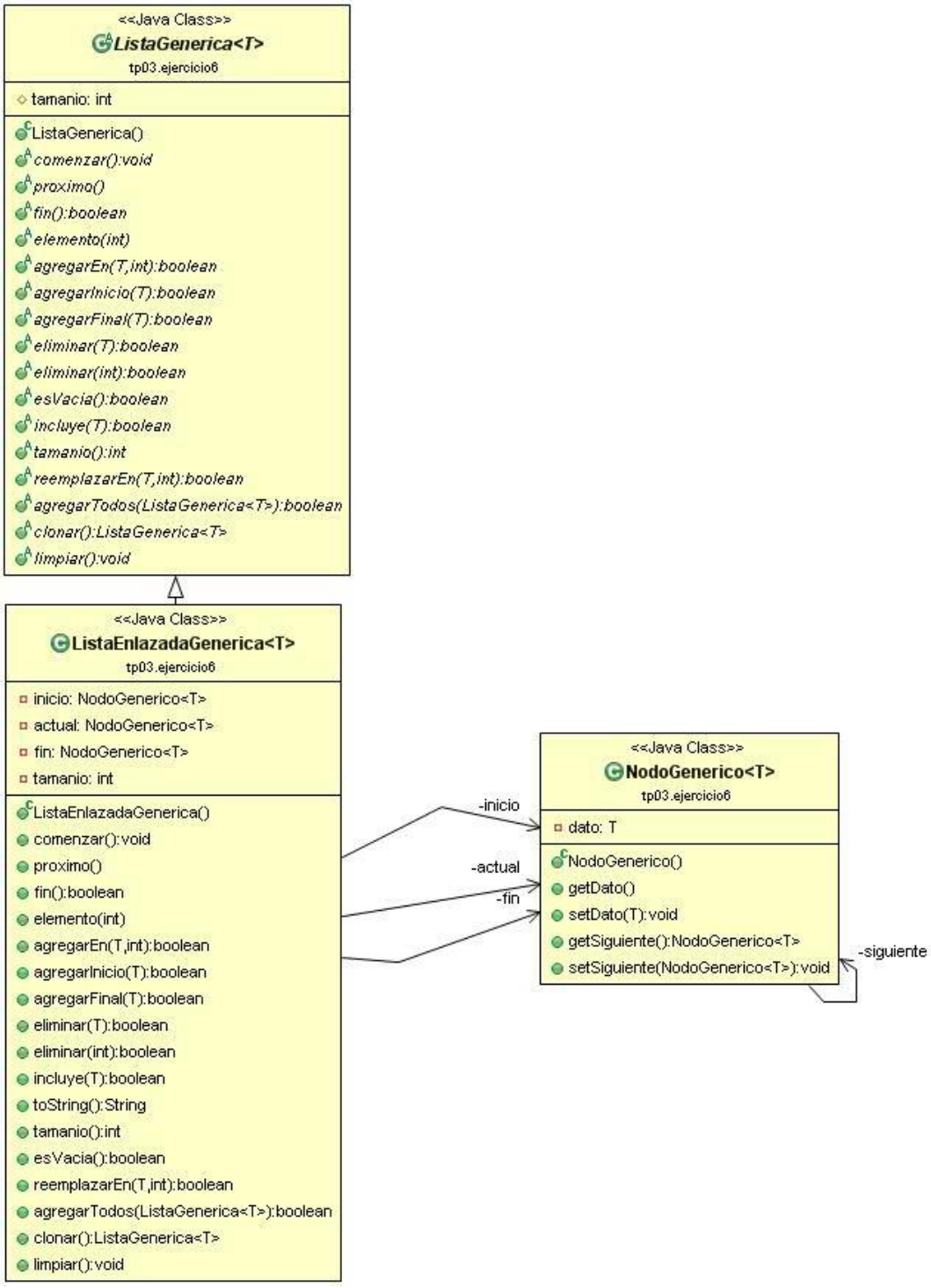
Sinopsis de las estructuras de datos

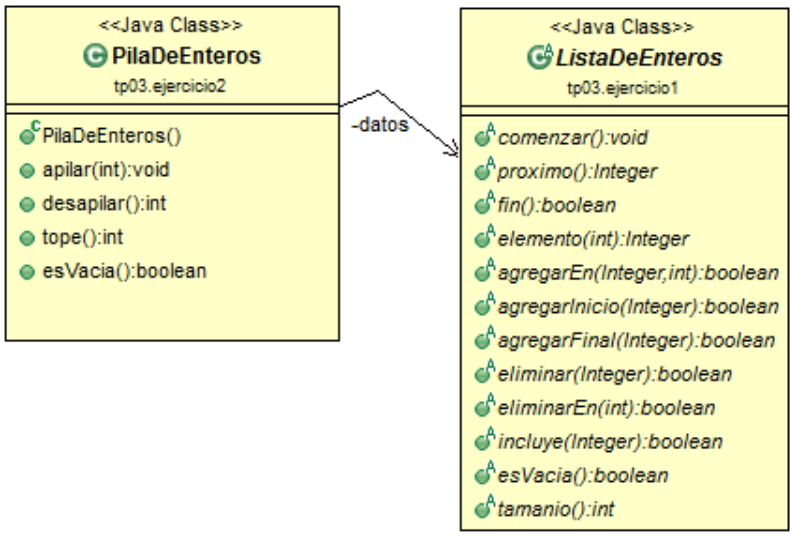
LISTA DE ENTEROS



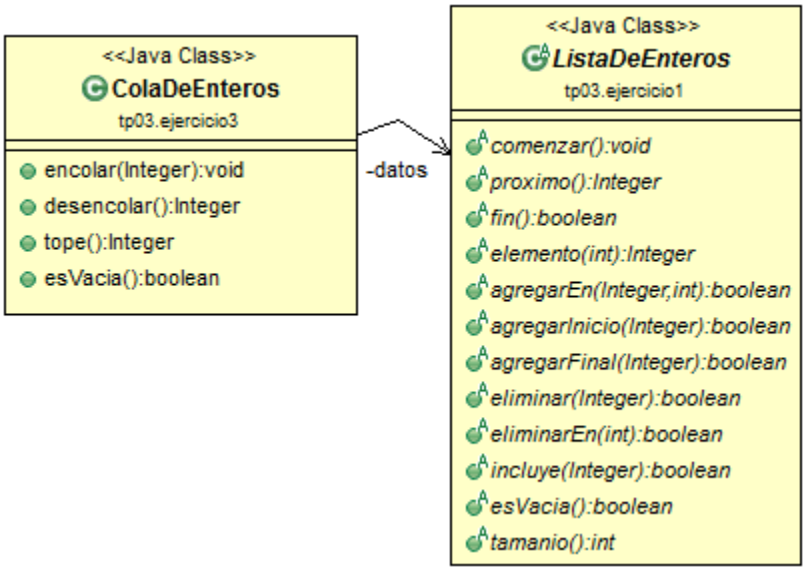
LISTA GENÉRICA



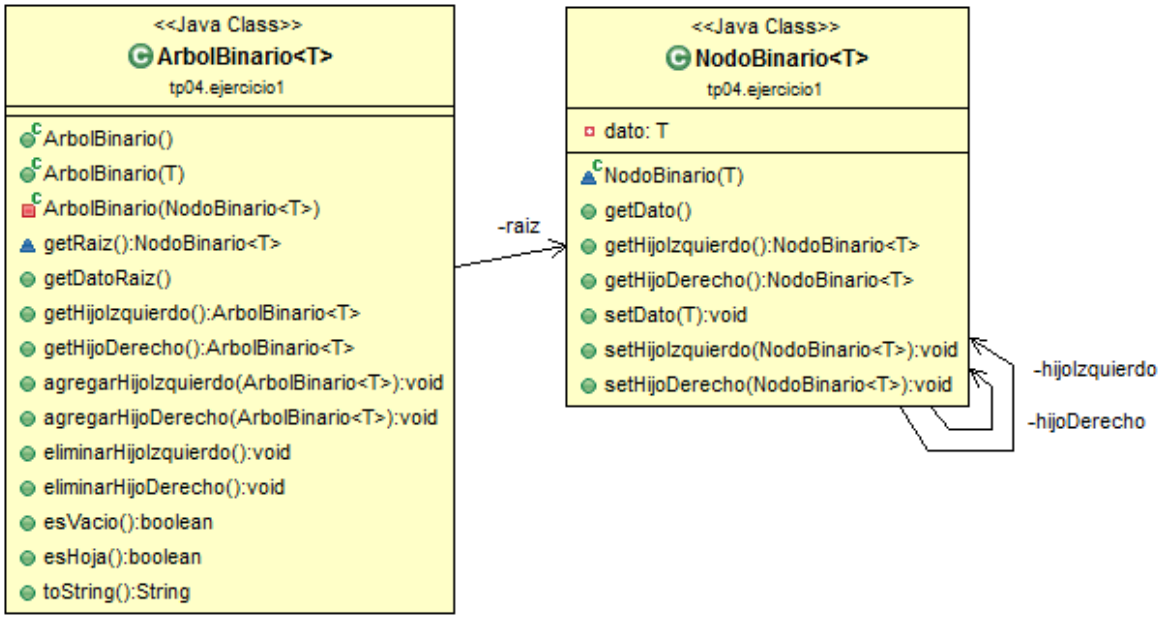
PILA DE ENTEROS



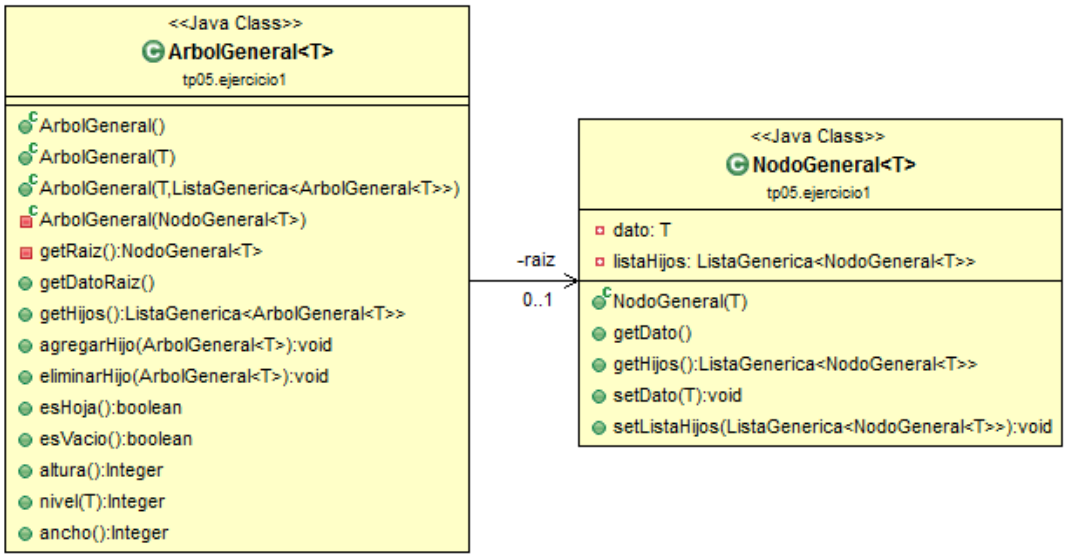
COLA DE ENTEROS



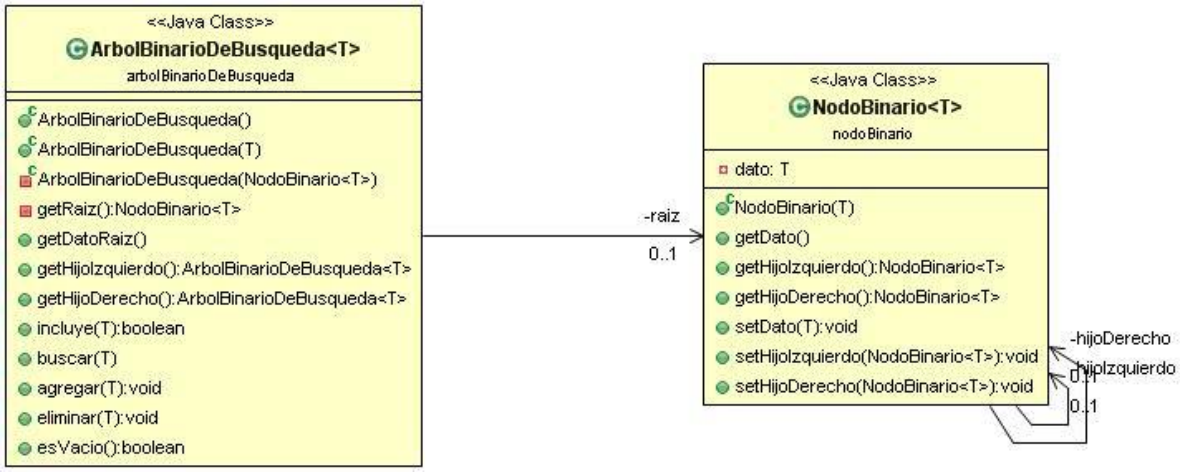
ÁRBOL BINARIO



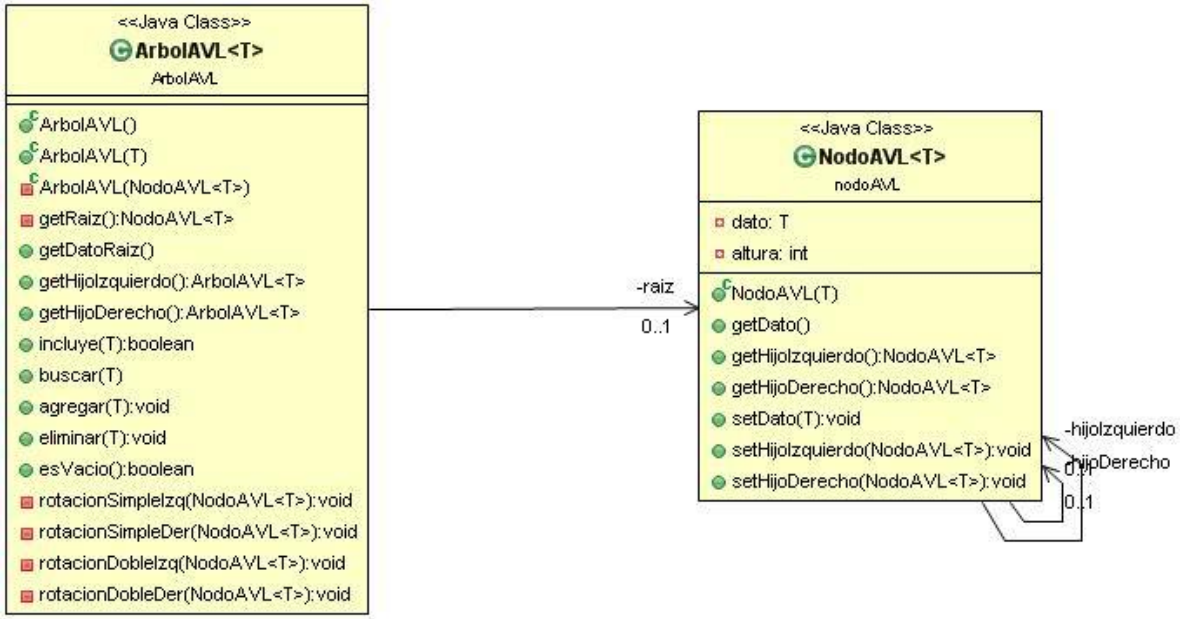
ÁRBOL GENERAL



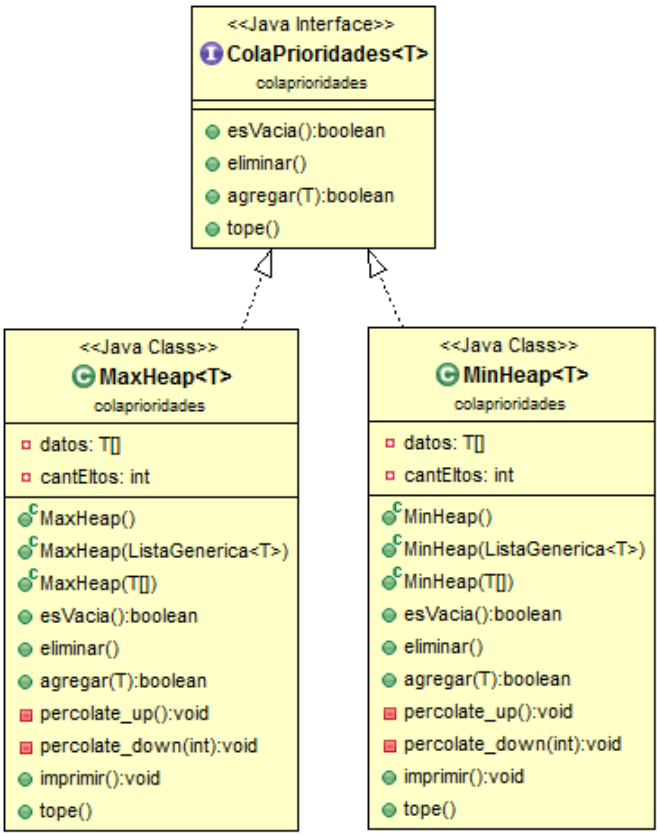
ÁRBOL BINARIO DE BÚSQUEDA



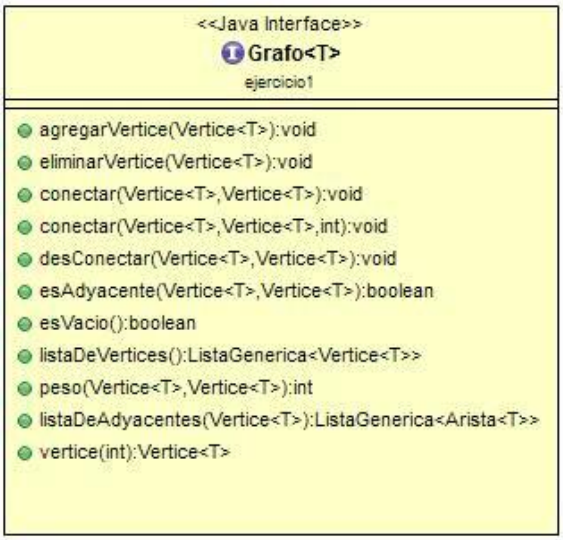
ÁRBOL AVL

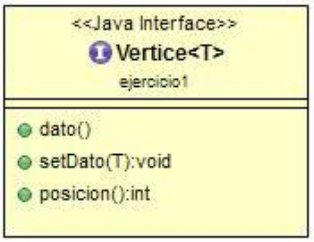


HEAP

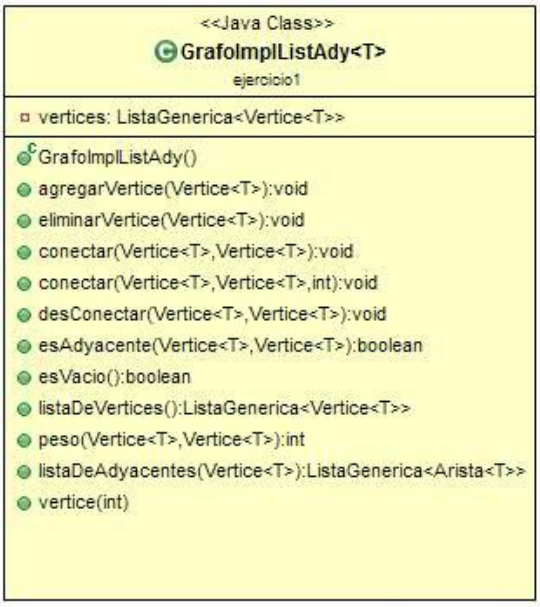

<<Java Interface>>
ColaPrioridades<T>
colaprioridades
esVacia():boolean
eliminar()
agregar(T):boolean
tope()
<<Java Class>>
MaxHeap<T>
colaprioridades
datos: T[]
cantElTos: int
MaxHeap()
MaxHeap(ListaGenerica<T>)
MaxHeap(T[])
esVacia():boolean
eliminar()
agregar(T):boolean
percolate_up():void
percolate_down(int):void
imprimir():void
tope()
<<Java Class>>
MinHeap<T>
colaprioridades
datos: T[]
cantElTos: int
MinHeap()
MinHeap(ListaGenerica<T>)
MinHeap(T[])
esVacia():boolean
eliminar()
agregar(T):boolean
percolate_up():void
percolate_down(int):void
imprimir():void
tope()

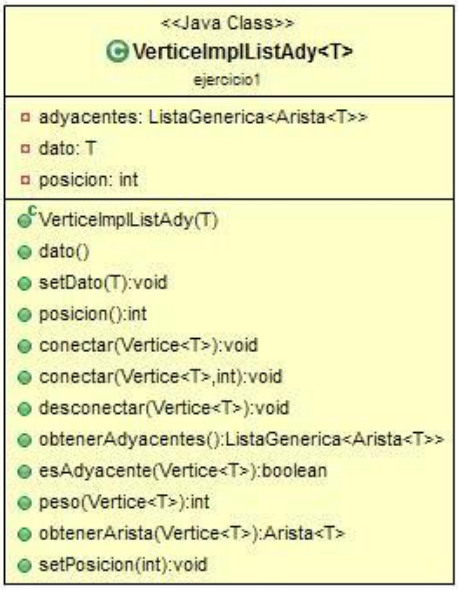
GRAFOS


<<Java Interface>>
Grafo<T>
ejercicio1
agregarVertice(Vertice<T>):void
eliminarVertice(Vertice<T>):void
conectar(Vertice<T>,Vertice<T>):void
conectar(Vertice<T>,Vertice<T>,int):void
desConectar(Vertice<T>,Vertice<T>):void
esAdyacente(Vertice<T>,Vertice<T>):boolean
esVacio():boolean
listaDeVertices():ListaGenerica<Vertice<T>>
peso(Vertice<T>,Vertice<T>):int
listaDeAdyacentes(Vertice<T>):ListaGenerica<Arista<T>>
vertice(int):Vertice<T>

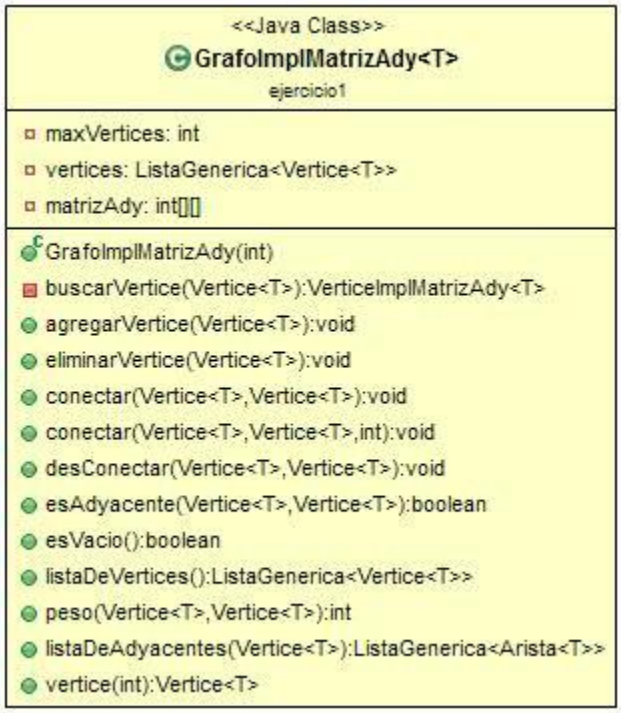

<<Java Interface>>
Vertice<T>
ejercicio1
dato()
setDato(T):void
posicion():int

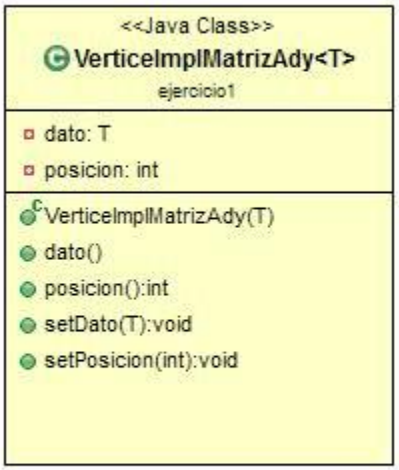

<<Java Interface>>
Arista<T>
ejercicio1
verticeDestino():Vertice<T>
peso():int


<<Java Class>>
GrafoImplListAdy<T>
ejercicio1
vertices: ListaGenerica<Vertice<T>>
GrafoImplListAdy()
agregarVertice(Vertice<T>):void
eliminarVertice(Vertice<T>):void
conectar(Vertice<T>,Vertice<T>):void
conectar(Vertice<T>,Vertice<T>,int):void
desConectar(Vertice<T>,Vertice<T>):void
esAdyacente(Vertice<T>,Vertice<T>):boolean
esVacio():boolean
listaDeVertices():ListaGenerica<Vertice<T>>
peso(Vertice<T>,Vertice<T>):int
listaDeAdyacentes(Vertice<T>):ListaGenerica<Arista<T>>
vertice(int)


<<Java Class>>
VerticeImplListAdy<T>
ejercicio1
adyacentes: ListaGenerica<Arista<T>>
dato: T
posicion: int
VerticeImplListAdy(T)
dato()
setDato(T):void
posicion():int
conectar(Vertice<T>):void
conectar(Vertice<T>,int):void
desconectar(Vertice<T>):void
obtenerAdyacentes():ListaGenerica<Arista<T>>
esAdyacente(Vertice<T>):boolean
peso(Vertice<T>):int
obtenerArista(Vertice<T>):Arista<T>
setPosicion(int):void


<<Java Class>>
AristaImpl<T>
ejercicio1
destino: Vertice<T>
peso: int
AristaImpl(Vertice<T>,int)
verticeDestino():Vertice<T>
peso():int


<<Java Class>>
GrafoImplMatrizAdy<T>
ejercicio1
maxVertices: int
vertices: ListaGenerica<Vertice<T>>
matrizAdy: int[][]
GrafoImplMatrizAdy(int)
buscarVertice(Vertice<T>):VerticeImplMatrizAdy<T>
agregarVertice(Vertice<T>):void
eliminarVertice(Vertice<T>):void
conectar(Vertice<T>,Vertice<T>):void
conectar(Vertice<T>,Vertice<T>,int):void
desConectar(Vertice<T>,Vertice<T>):void
esAdyacente(Vertice<T>,Vertice<T>):boolean
esVacio():boolean
listaDeVertices():ListaGenerica<Vertice<T>>
peso(Vertice<T>,Vertice<T>):int
listaDeAdyacentes(Vertice<T>):ListaGenerica<Arista<T>>
vertice(int):Vertice<T>


<<Java Class>>
VerticeImplMatrizAdy<T>
ejercicio1
dato: T
posicion: int
VerticeImplMatrizAdy(T)
dato()
posicion():int
setDato(T):void
setPosicion(int):void