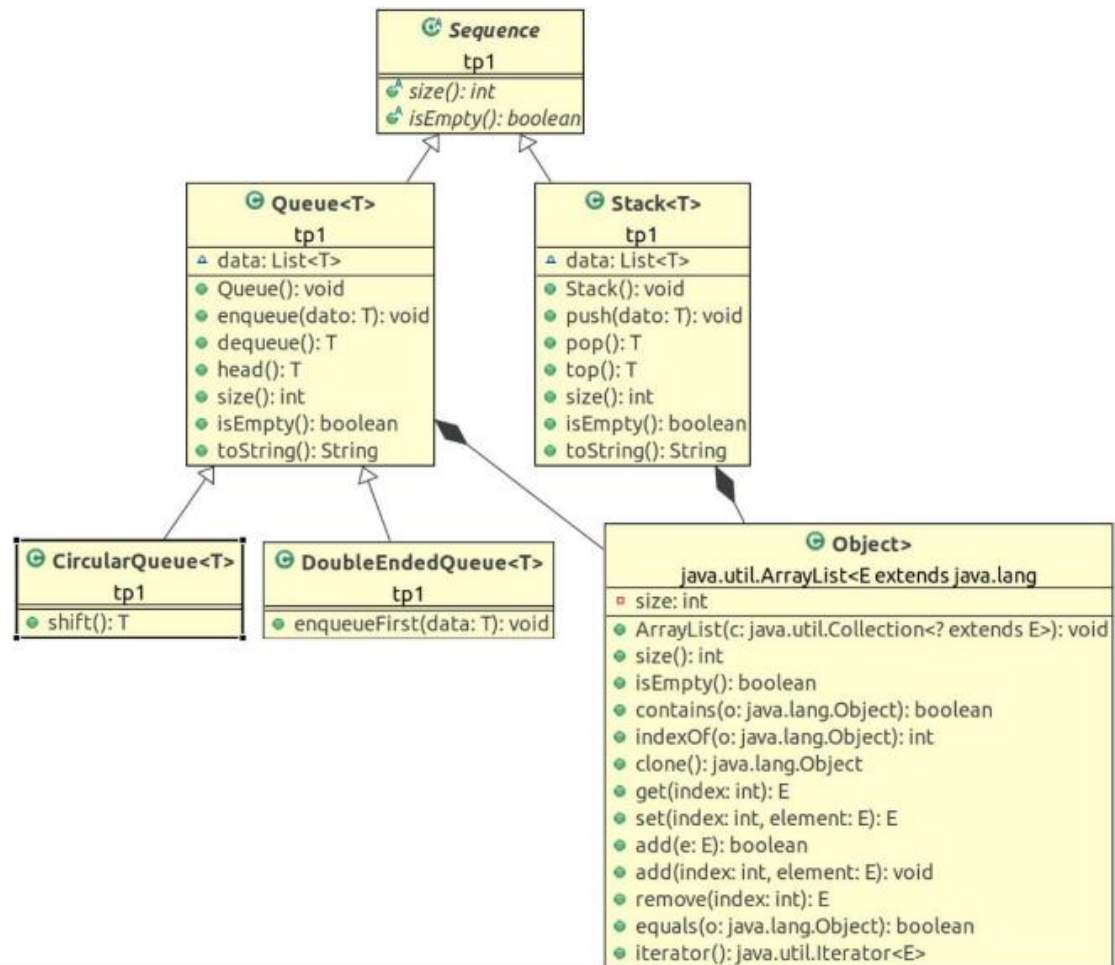


List, Stack y Queue



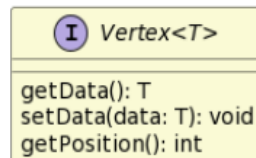
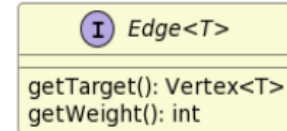
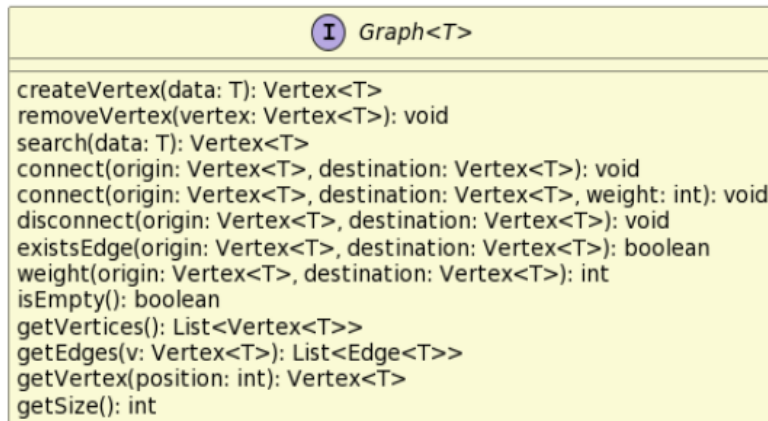
BinatyTree

BinaryTree<T>
<ul style="list-style-type: none"> data: T leftChild: BinaryTree<T> rightChild: BinaryTree<T>
<ul style="list-style-type: none"> BinaryTree(): void BinaryTree(T): void getData(): T setData(T): void getLeftChild(): BinaryTree<T> getRightChild(): BinaryTree<T> addLeftChild(BinaryTree<T>): void addRightChild(BinaryTree<T>): void removeLeftChild(): void removeRightChild(): void isEmpty(): boolean isLeaf(): boolean hasLeftChild(): boolean hasRightChild(): boolean toString(): String contarHojas(): int espejo(): BinaryTree<T> entreNiveles(int, int): void

GeneralTree

GeneralTree<T>
<ul style="list-style-type: none"> data: T children: List<GeneralTree<T>>
<ul style="list-style-type: none"> GeneralTree(): void GeneralTree(T): void GeneralTree(T, List<GeneralTree<T>>): void getData(): T setData(T): void getChildren(): List<GeneralTree<T>> setChildren(List<GeneralTree<T>>): void addChild(GeneralTree<T>): void isLeaf(): boolean hasChildren(): boolean isEmpty(): boolean removeChild(GeneralTree<T>): void altura(): int nivel(T): int ancho(): int

Graph



Graph - Las interfaces y clases

