## Graph Implementation

Iterable

## Class Diagram

```
V, E
                        Graph
+ Graph()
+ clear()
+ clearEdges()
+ vertexCount(): int
                                                                 DuplicateVertexException
+ edgeCount(): int
+ addVertex(v: Vertex<V>)
+ addVertices(vertices: ArrayList<Vertex<V>>)
+ containsVertex(v: Vertex<V>): boolean
                                                                  DuplicateEdgeException
+ addEdge(e: Edge<V, E>)
+ addEdges(edges: ArrayList<Edge<V, E>>)
+ containsEdge(e: Edge<V, E>): boolean
                                                                  NoSuchVertexException
+ removeVertex(v: Vertex<V>): ArrayList<Edge<V, E>>
+ removeEdge(e: Edge<V, E>)
+ adjacentCount(v: Vertex<V>): int
+ vertices(): ArrayList<Vertex<V>>
                                                                   {\tt NoSuchEdgeException}
+ adjacent(v: Vertex<V>): ArrayList<Vertex<V>>
+ edges(): ArrayList<Edge<V, E>>
+ minSpanningTree(v: Vertex<V>): Tree<V, E>
                                                                             Vertex
+ totalVertexWeight(): int
                                                              + Vertex(label: V)
+ totalEdgeWeight(): int
                                                              + Vertex(label: V, weight: int)
+ hasCycle(): boolean
                                                              + getLabel(): V
                                                              + getWeight(): int
                                                              + setLabel(label: V)
                                                              + setWeight(weight: int)
                                        V, E
                   Tree
                                                              + equals(other: Vertex<V>): boolean
                                                              + toString(): String
+ BinaryTree()
+ getParent(): Tree<V, E>
+ getVertex(): Vertex<V>
                                                                                              √V, E
+ isChild(): boolean
                                                                      Edge
+ isParent(): boolean
                                                Edge(v1: Vertex<V>, v2: Vertex<V>, label: E)
+ isRoot(): boolean
                                                Edge(v1: Vertex<V>, v2: Vertex<V>, lable: E,
+ isLeaf(): boolean
                                                weight: int)
+ isDegenerate(): boolean
                                                getVertex1(): Vertex<V>
+ height(): int
+ depth(): int
                                                getVertex2(): Vertex<V>
+ isSuperTreeOf(t: Tree<V, E>) boolean
                                                getLabel(): E
+ isSubtreeOf(t: Tree<V, E>) boolean
                                                getWeight(): int
                                                setVertex1(v: Vertex<V>)
+ isSiblingOf(): boolean
                                                setVertex2(v: Vertex<V>)
+ isChildOf(): boolean
                                                setLabel(label: E)
+ isParentOf(): boolean
+ makeNewRoot(t: Tree<V, E>)
                                                setWeight(weight: int)
                                                equals(other: Edge<V, E>): boolean
+ iterator(): iterator<V>
                                                toString(): String
                «interface»
```