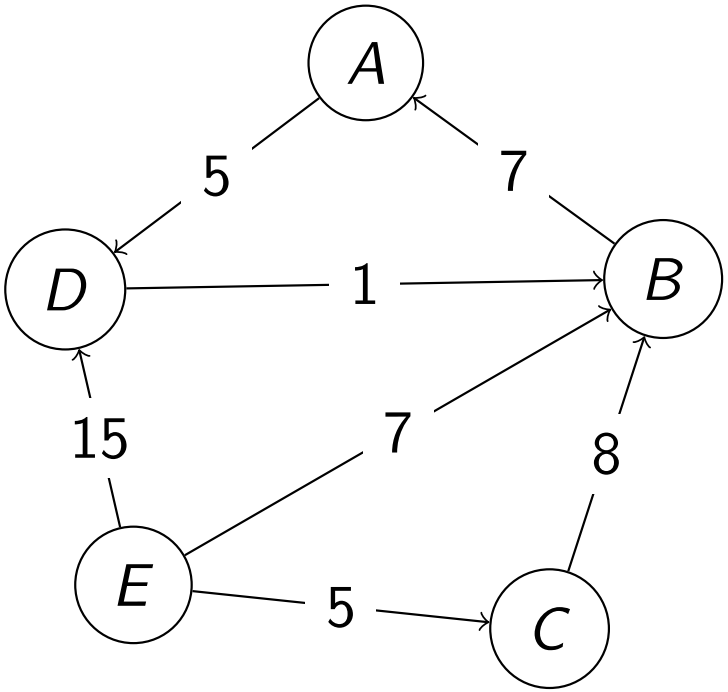
Adj. Matrix - Disadvantages

	1	**2**	**3**	**4**	**5**
1	0	0	1	0	0
2	1	0	1	0	0
3	0	0	0	0	1
4	0	0	0	0	0
5	0	1	0	0	0

1. **Space Complexity:** $O(V^2)$
2. We have to resize the 2d array everytime we create or delete a vertex (like arraylist).

Graphs are a Collection of Vertecies and Edges



Recall, a graph is:

$$G = (V, E)$$

Vertecies (V): A, B, C, D, E

Edges (E)

- $A \to D$, 5
- $B \to A$, 7
- $C \to B$, 8
- $D \to B$, 1
- $E \to D$, 15
- $E \to B$, 7
- $E \to C$, 5

Graph - Adj List Representation

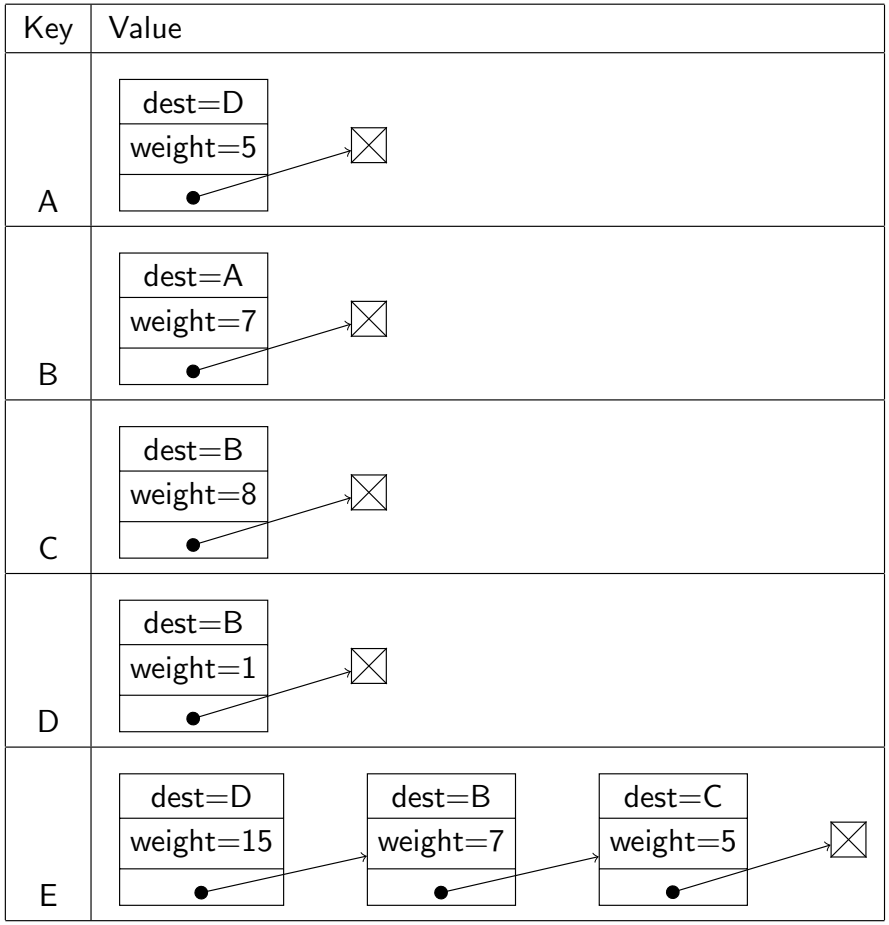
Recall, a graph is:

$$G = (V, E)$$

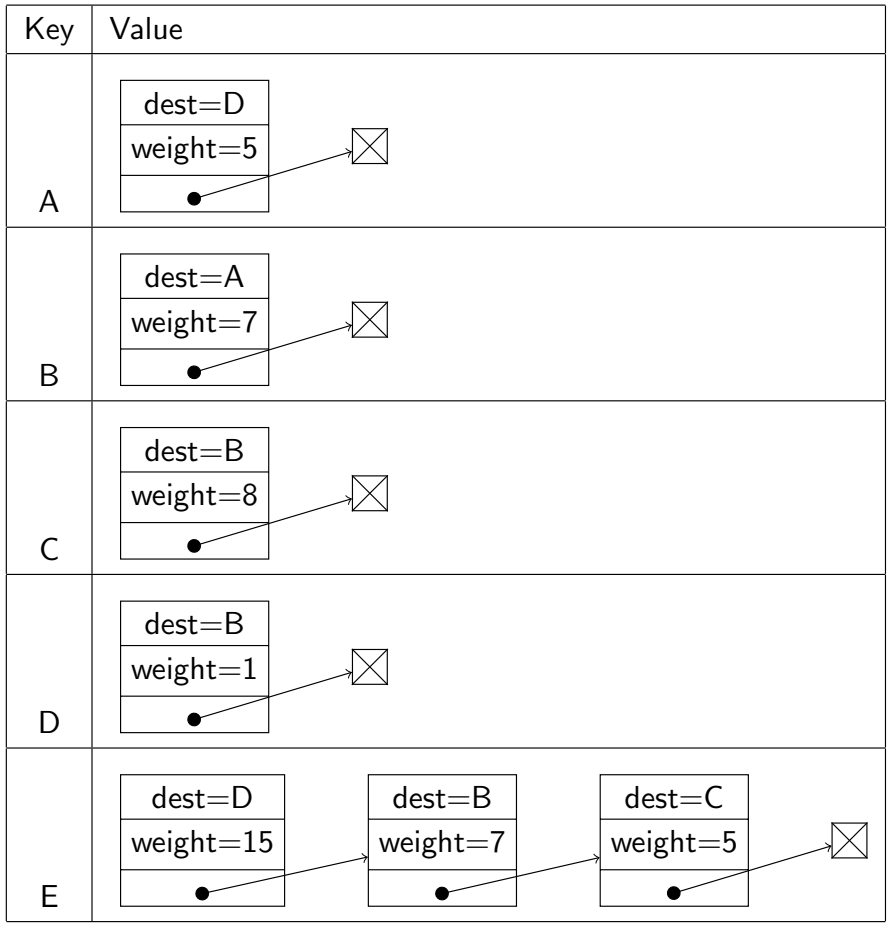
Vertecies (V): A, B, C, D, E

Edges (E)

- $A \rightarrow D$, 5
- $B \rightarrow A$, 7
- $C \rightarrow B$, 8
- $D \rightarrow B$, 1
- $E \rightarrow D$, 15
- $E \rightarrow B$, 7
- $E \rightarrow C$, 5



Adjacency List

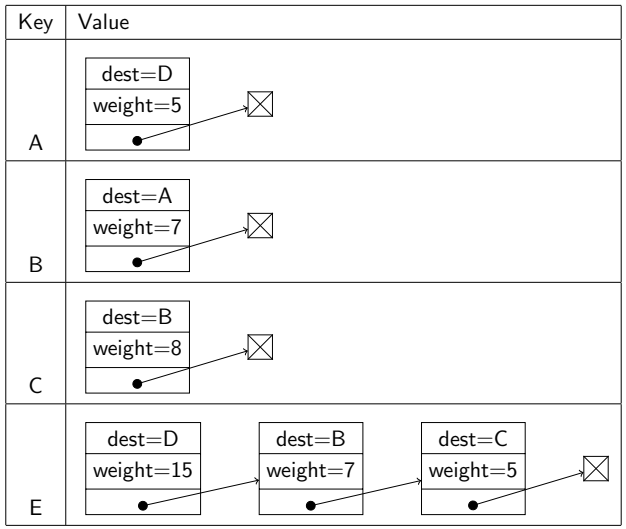



```
private Map<String, List<Edge>> map = new HashMap<>();
```

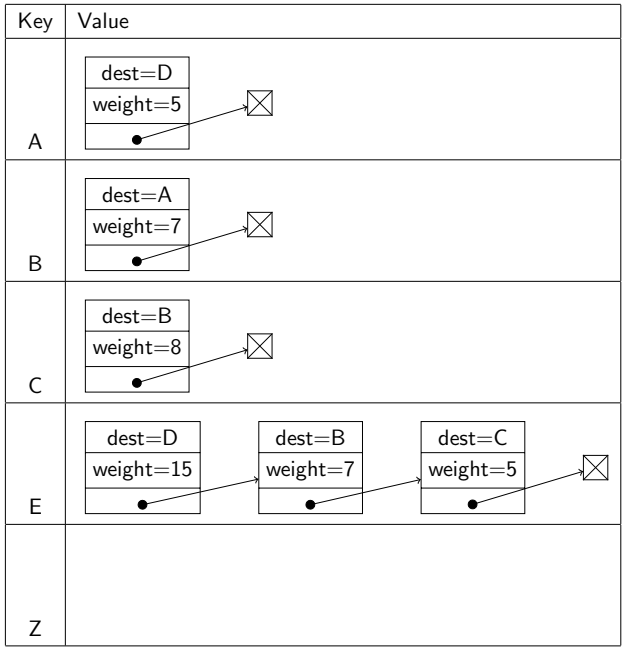
Activity - Constructing the Edge Class

1. For the adjacency list we will be creating lists of edges to associate with vertecies
2. Construct an Edge class in the worksheet with the following attributes
   - Two private final attributes: 1) a string for the destination and 2) a integer for the weight.
   - Getters for both of those attributes.

Algorithm - Adding Vertecies



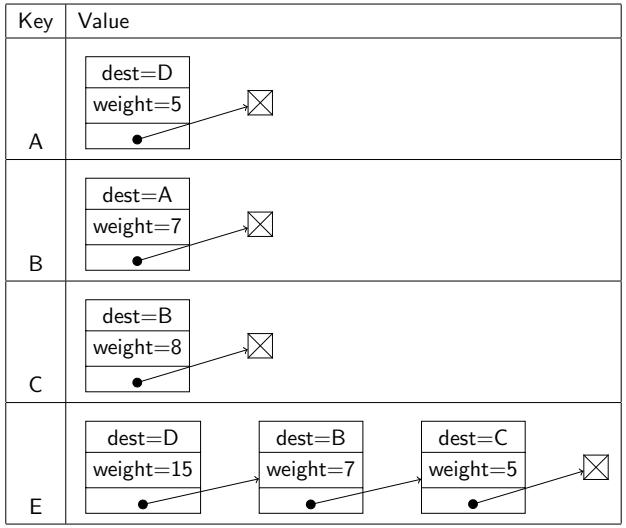
Step 1: Have an Adj List



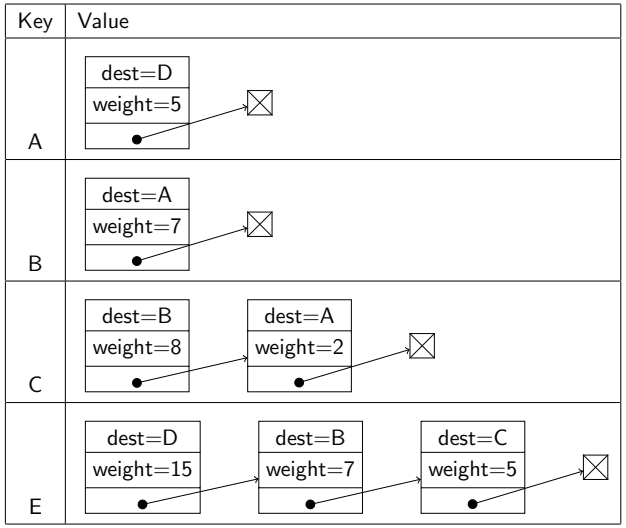
Step 2: After Adding the "Z" Vertex and a list to store its edges.

```
map.contains(key);
map.put(key, val);
map.putIfAbsent(key, val);
```

Algorithm - Adding Directed Edges



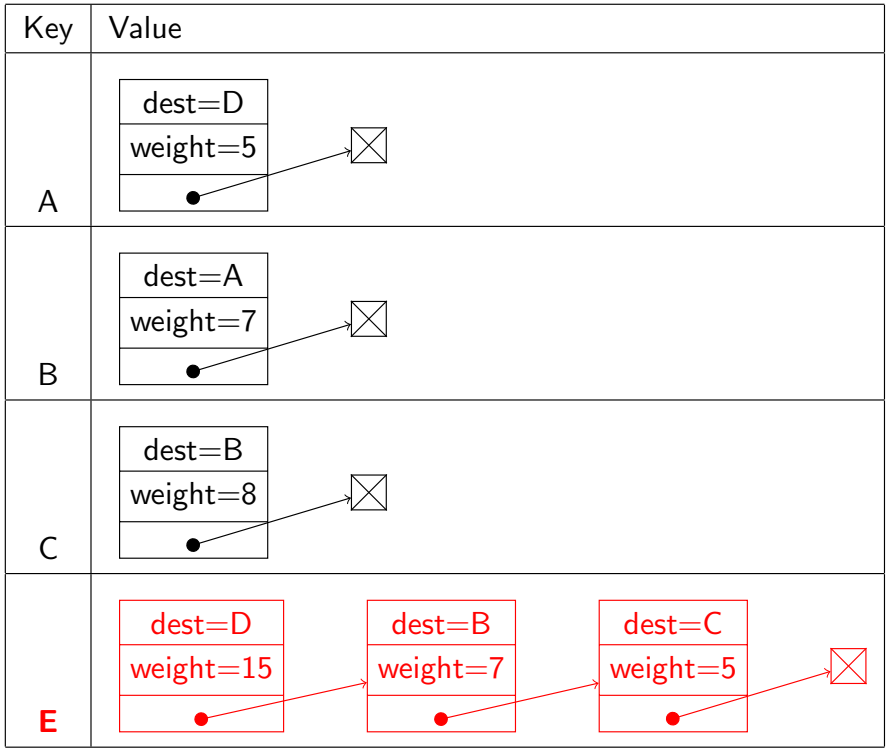
Step 1: Get the list associated with the source vertex. Lets use `C` as an example.



Step 2: Add the edge to the end of that list. In this case, one to A with a weight of 2.

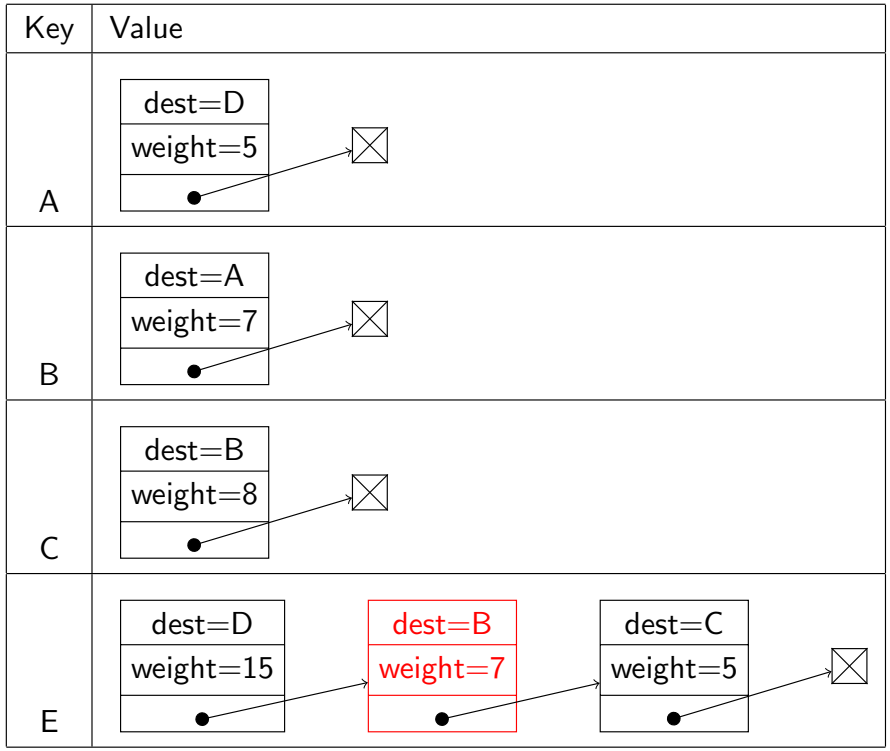
```
map.get(key);
```

Algorithm - Removing Edges



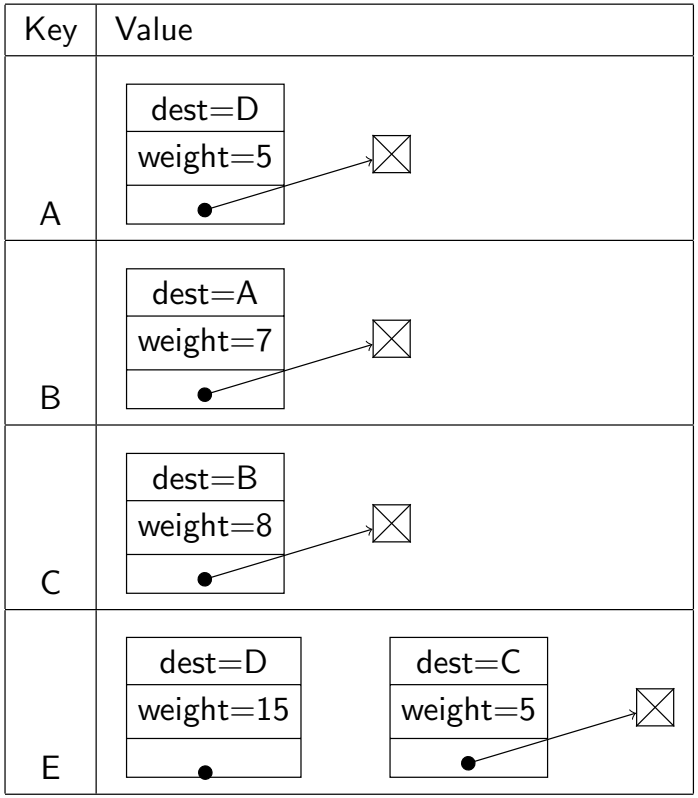
Step 1: Get the list associated with the source of the edge you want to remove. For example E.

Algorithm - Removing Edges



Step 2: Iterate over the list to get the index of the Edge object with the matching destination. If we want to remove B, that would be index 1.

Algorithm - Removing Edges



Step 3: Remove the element at that index from the list.

Activity - Constructing a Digraph

Go to the worksheet and implement the Digraph class with the algorithms we just went over.