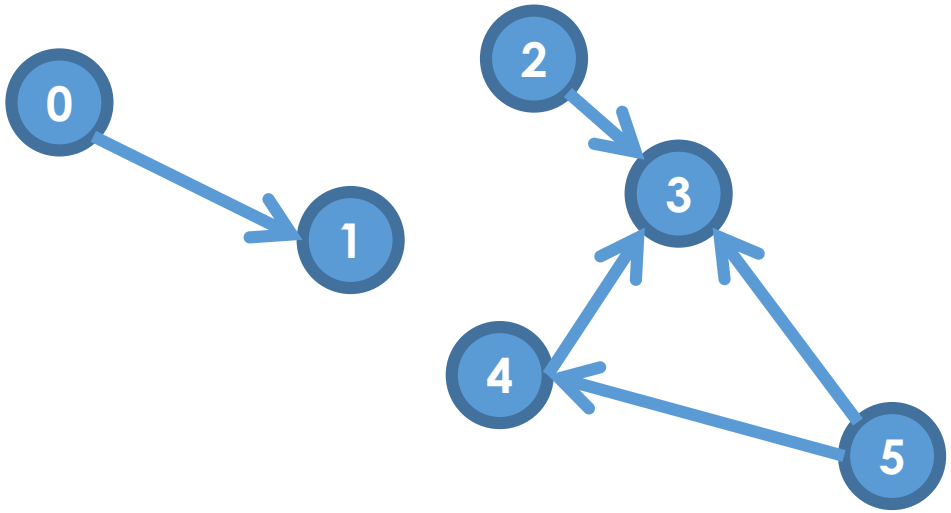
Graphs

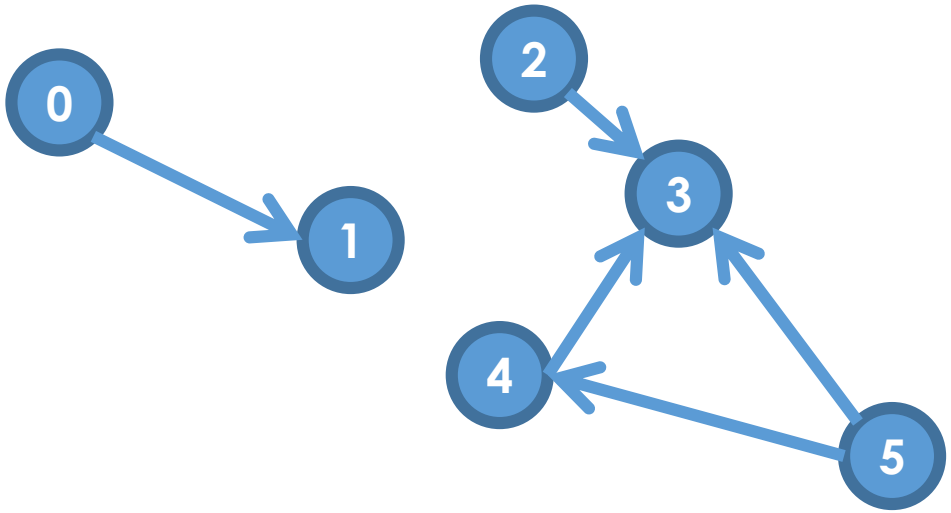
Representation: Adjacency List

By the end of this video you will be able to…

- Implement graphs in Java using adjacency lists
- Describe advantages and disadvantages of adjacency list and adjacency matrix representations.



V = {0,1,2,3,4,5}

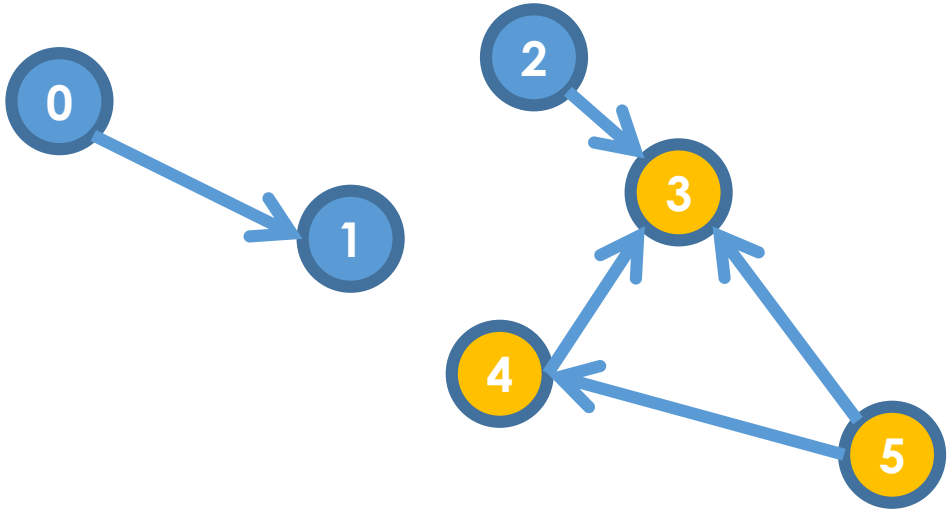


V = {0,1,2,3,4,5}

0	1	0	0	0	0
0	0	0	0	0	0
0	0	0	1	0	0
0	0	0	0	0	0
0	0	0	1	0	0
0	0	0	1	1	0

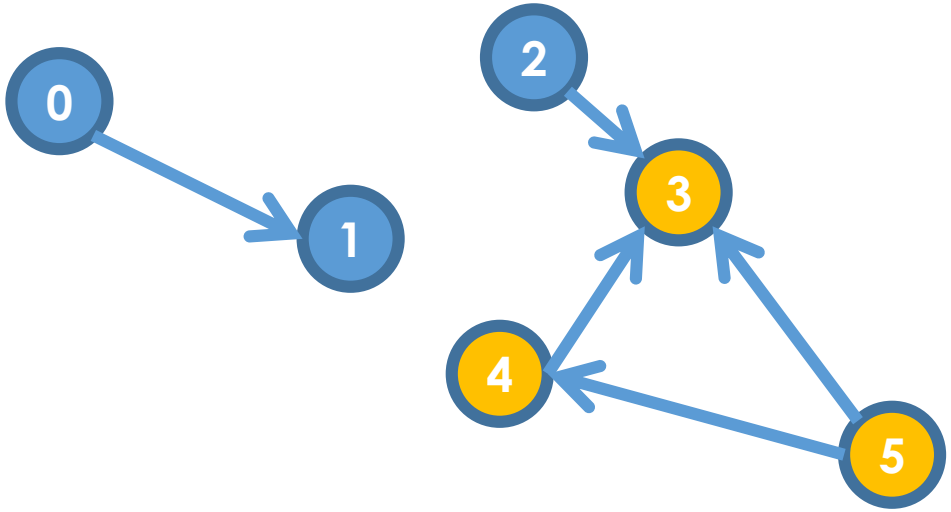
```
public class GraphAdjList extends Graph {

   private Map<Integer,ArrayList<Integer>> adjListsMap;
```

```
public class GraphAdjList extends Graph {

   private Map<Integer,ArrayList<Integer>> adjListsMap;
```



5 → {3,4}

```
public class GraphAdjList extends Graph {

   private Map<Integer,ArrayList<Integer>> adjListsMap;
```



0 → {1}

1 → null

2 → {3}

3 → null

4 → {3}

5 → {3,4}

```
public class GraphAdjList extends Graph {

...
  public void implementAddVertex() {
     int v = getNumVertices();
     ArrayList<Integer> neighbors = new ArrayList<Integer>();
     adjListsMap.put(v,  neighbors);
  }

  public void implementAddEdge(int v, int w) {
     (adjListsMap.get(v)).add(w);
  }
```

Recap: Adjacency lists

- Easy to add vertices.
- Easy to add/remove edges.

- May use a lot less memory than adjacency matrices.

Recap: Adjacency lists

- Easy to add vertices.
- Easy to add/remo

But is it fast?

- May use a lot less memory than adjacency matrices.

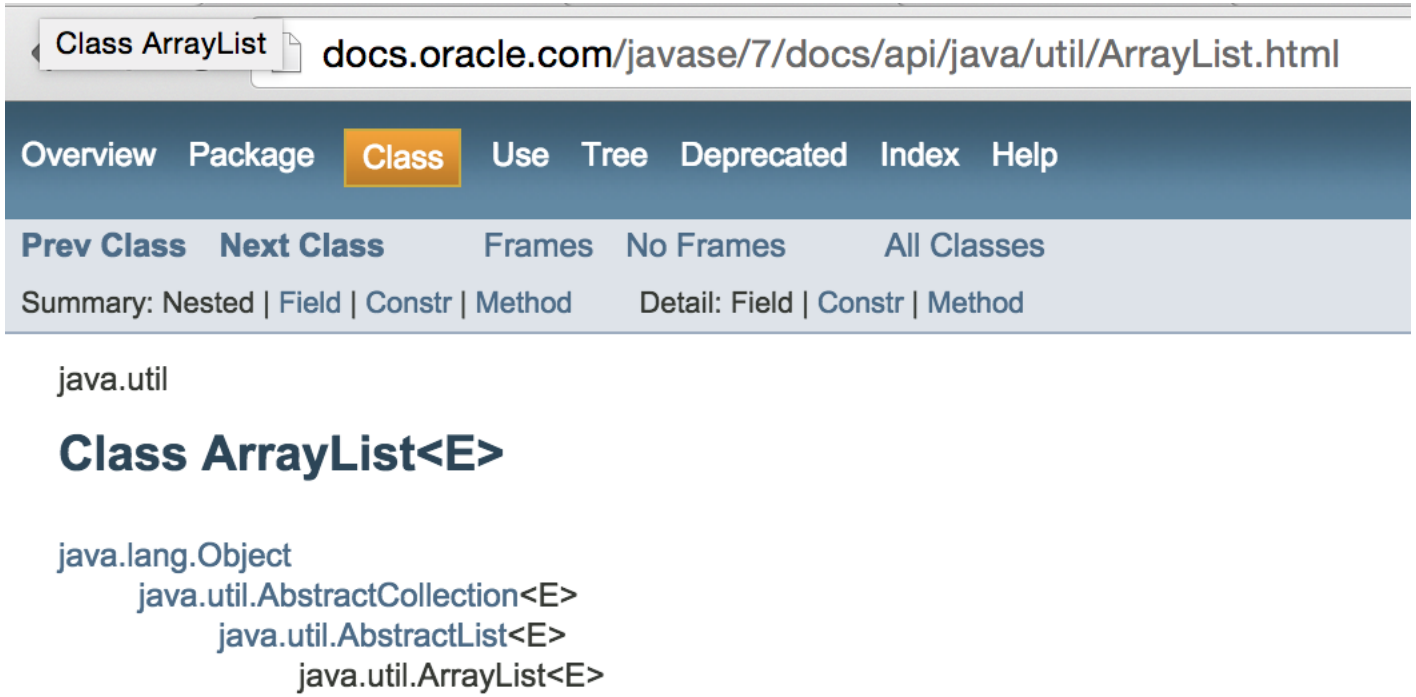
Recap: Adjacency lists

- Easy to add vertices.
- Easy to add/remo

But is it fast?

- May use a lot less memory than adjacency matrices.

```
public class GraphAdjList extends Graph {

   private Map<Integer,ArrayList<Integer>> adjListsMap;
```



Constant time operations:

size, isEmpty, get, set, add*

Why use anything else?