`Names` example

- Example: **`Names`** class
- practice with
  - coding array algorithms
  - implementing classes
  - and using good development techniques
- incremental development
- for **`lookup`**, **`remove`**, **`insert`**:
  - design test cases first
  - implement code
    - code refactoring
  - test code

Announcements

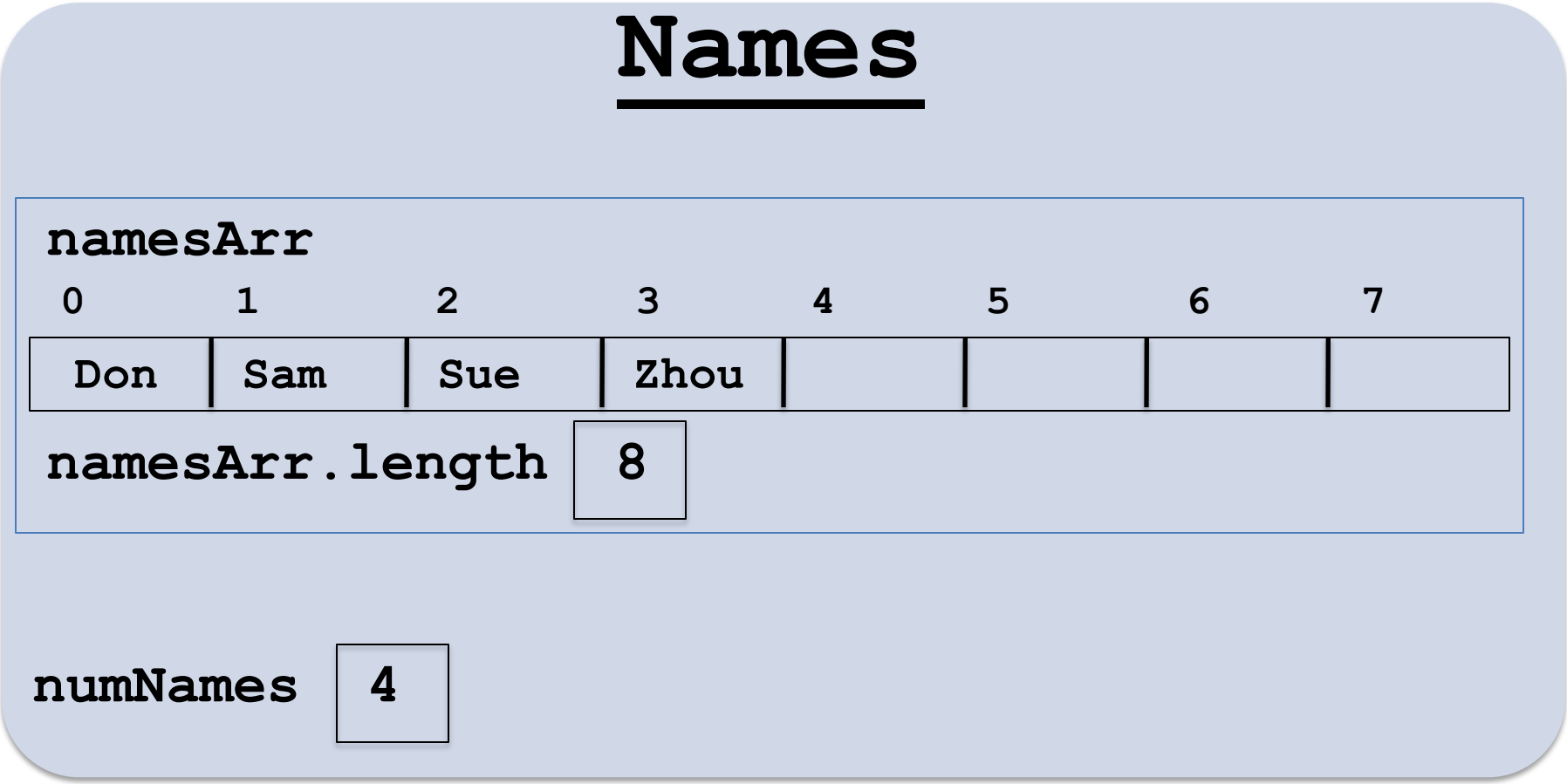
- MT 1: Tues. Sept. 23 9:30 – 10:50am Pacific (alternate time for students in Asia: 6:30pm Pacific)
- IMPORTANT: connect to CS 455 gradescope course soon:
  - email about joining gradescope course
  - on d2l, choose "CS 455 Tools" menu (top of page)
  - choose "gradescope"
  - generally can connect from d2l or via gradescope.com
- Rehearsal exam on gradescope weekend before exam
- PA 2 coming soon
- Lab 5 (next week) no partners: it's a milestone for PA2

Example: **`Names`** class

- Stores a list of unique names in alphabetical order.
- Allows look-up, insert, and removal of names in the list.
- Uses partially-filled array representation

- **`Names.java`** has a partial implementation
- **`MinNamesTester.java`** is a program to test that subset.

Names representation



Lookup test cases

- Returns true iff **`target`** is present in names

`namesArr`

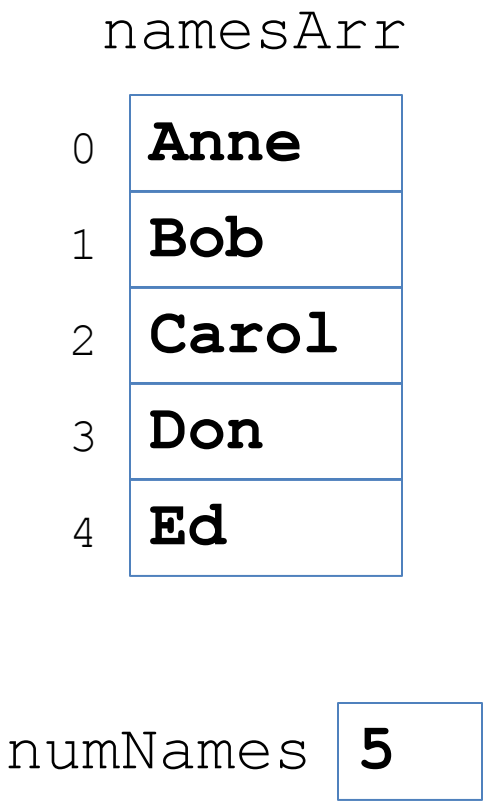
0	**Anne**
1	**Bob**
2	**Carol**
3	**Don**
4	**Ed**

`numNames` **5**

Test cases

Lookup code notes

- Returns true iff **`target`** is present in names



Remove test cases

Removes **`target`** from names object, and returns **`true`**.

If **`target`** wasn't present in names, returns **`false`** and no change made to names.

Test cases

`namesArr`

0	**Anne**
1	**Bob**
2	**Carol**
3	**Don**
4	**Ed**

`numNames` **5**

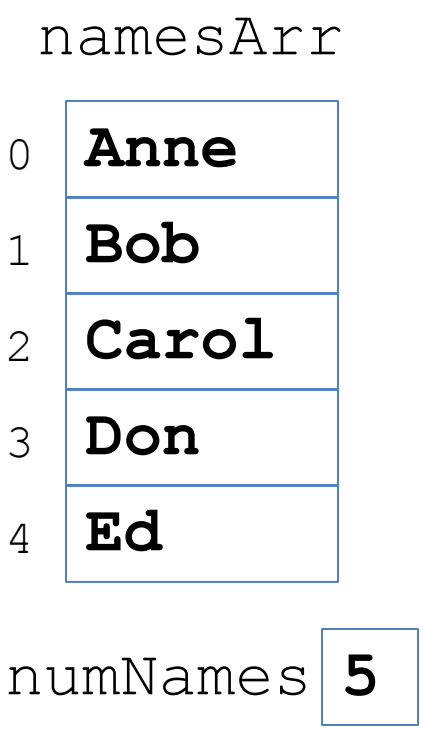
Reuse code to test remove

```
public static void testRemove() {
    Names names = new Names();
    names.loadNames();
    System.out.println("Attempt remove: Scotty");
    boolean removed = names.remove("Scotty");
    if (!removed) {
       System.out.println("Scotty was not present");
    }
    System.out.println(
      "Names in list [exp: Anne Bob Carol Don Ed]: ");
    names.printNames();
    System.out.println(
                "Number of names in list [exp: 5]: "
                + names.numNames());
}
```

Implementing remove: outline

Removes **`target`** from names object, and returns **`true`**.
If **`target`** wasn't present in names, returns **`false`** and no change made to names.

```
public boolean remove(String target) {
```



Minimize amount of code

- Reuse lookup loop?
- It returns boolean
- Refactor!

New helper function

```
/**
    lookupLoc returns index of target in namesArr
    or NOT_FOUND if it is not present
*/
private int lookupLoc(String target)
```

Refactored `lookup` that uses `lookupLoc`

```
public boolean lookup(String target)
```

Implementing remove

Removes **`target`** from names object, and returns **`true`**.
If **`target`** wasn't present in names, returns **`false`** and no change made to names.

```
public boolean remove(String target) {
```