Telling a Random Story

Summary

ArrayList

- Indexable collection, like array, but growable!
  - Access via integer index, start with zero
  - import java.util.ArrayList or java.util.*;

ArrayList

- Indexable collection, like array, but growable!
    - Access via integer index, start with zero
    - import java.util.ArrayList or java.util.*;
    - Create with generic: **`ArrayList<String>`**

ArrayList

- Indexable collection, like array, but growable!
  - Access via integer index, start with zero
  - import java.util.ArrayList or java.util.*;
  - Create with generic: **`ArrayList<Integer>`**

ArrayList

- Indexable collection, like array, but growable!
  - Access via integer index, start with zero
  - import java.util.ArrayList or java.util.*;
  - Create with generic: **`ArrayList<Integer>`**
- Common methods for ArrayList
  - .add(elt) — added to end of ArrayList
  - .size() — returns number of elements in ArrayList
  - .get(index) — returns elements at index
  - .set(index,elt) — assign elt to index location

ArrayList with Indexing Loops

- Access elements via indexing
  - Start with zero, loop to less than **`.size()`**
  - Access via **`.get(index)`**
  - Do not call **`.remove()`** during iteration

```
ArrayList<String> a = new ArrayList<String>();
// add elements

for(int k=0; k < a.size(); k++){
    String s = a.get(k);
    // process s
}
```

Duke
UNIVERSITY

ArrayList via Indexing Loops

- Access elements via iterable loop
  - process elements, in order
  - don't need index of element
  - do not call **`.remove()`** during iteration

```
ArrayList<String> a = new ArrayList<String>();
// add elements

for(String s : a){
    // process s
}
```

Duke
UNIVERSITY