

Situation

- ❑ Need a list of data be considered both
 - ❑ Individually
 - ❑ Collectively
- ❑ Ordered lists
 - ❑ `ArrayList`
- ❑ Unordered lists
 - ❑ `HashSet`
- ❑ Mappings
 - ❑ `HashMap`



ArrayList – capabilities

```
ArrayList<String> library = new ArrayList<String>();
```

- add(T v)
 - Adds an element with value v to the end of the list

- get(int j)
 - Returns the element with index value j

- indexOf(T v)
 - Returns the index of the element whose value equals v

- size()
 - Returns the number of elements in the list

ArrayList – capabilities

```
ArrayList<String> library = new ArrayList<String>();
```

- ❑ clear()
 - ❑ Removes all elements from the list
- ❑ isEmpty()
 - ❑ Returns whether the list is empty
- ❑ remove(int j)
 - ❑ Shrinks the list by removing the element at index j

ArrayList – ordered list

```
ArrayList<String> library = new ArrayList<String>();
```

```
library.add( "Soldier of the great war" );
```

```
library.add( "Name of the rose" );
```

```
library.add( "Harlot's ghost" );
```

```
library.add( "The sparrow" );
```

```
library.add( "Name of the rose" );
```

HashSet – capabilities

```
HashSet<String> puppies = new Set<String>();
```

- `add(T v)`
 - Adds an element with value `v` to the set
- `contains(T v)`
 - Returns whether the set has an element equal to value `v`
- `size()`
 - Returns the number of elements in the set

HashSet – capabilities

```
HashSet<String> puppies = new Set<String>();
```

- `remove(Object v)`
 - Removes the element with value `v` from the set

- `clear()`
 - Removes all elements from the set

- `isEmpty()`
 - Returns whether the set is empty

HashSet – unordered list

```
Set<String> puppies = new Set<String>();
```

```
puppies.add( "nilla" );
```

```
puppies.add( "galen" );
```

```
puppies.add( "panther" );
```

```
puppies.add( "puffin" );
```

```
puppies.add( "maizie" );
```

```
puppies.add( "galen" );
```

HashMap – capabilities

```
HashMap<String, Integer> numerals  
    = new HashMap<String, Integer> ();
```

- put(K key, V value)
 - Adds the mapping relating key to value

- get(Object key)
 - Returns the mapping for the key

- remove(Object key)
 - Removes the mapping for the specified key

- size()
 - Returns the number of mappings.

HashMap – capabilities

```
HashMap<String, Integer> numerals  
    = new HashMap<String, Integer> ();
```

- clear()
 - Removes all of the mappings from the map

- containsValue(Object value)
 - Returns whether one or more keys maps to the value

- isEmpty()
 - Returns whether there are any mapping

- size()
 - Returns the number of mappings.

HashMap – functional list

```
HashMap<String, Integer> numerals  
    = new HashMap<String, Integer> ();
```

```
numerals.put( "I",    1 );  
numerals.put( "V",    5 );  
numerals.put( "X",   10 );  
numerals.put( "L",   50 );  
numerals.put( "C",  100 );  
numerals.put( "D",  500 );  
numerals.put( "M", 1000 );
```

Collections Framework

- ❑ `max(Collection c)`
 - ❑ Returns the maximum element of collection `c`
- ❑ `min(Collection c)`
 - ❑ Returns the minimum element of collection `c`
- ❑ `swap(List c, int a, int b)`
 - ❑ Swaps the elements the values of the a^{th} and b^{th} elements in list `c`

Collections Framework

- ❑ `reverse(List c)`
 - ❑ Reverses the order of the elements in list `c`
- ❑ `shuffle(List c)`
 - ❑ Randomly arranges the ordering of the elements in list `c`
- ❑ `sort(List c)`
 - ❑ Arranges the ordering of the elements in list `c` to be sorted