

## ArrayList

- Constructors

```
ArrayList lst1 = new ArrayList();
ArrayList lst2 = new ArrayList(int initialSize);
ArrayList<String> strList = new ArrayList();
```

- Parameterized type

- use if you know the type of the list and the list type is not mixed

- Methods

```
size() // Returns the num of items held.
add(Object o) // Appends o to end.
add(int idx, Object o) // Inserts o at pos idx.
remove(int idx) // Removes item at pos idx.
get(int idx) // Gets items at idx. No removal.
set(int idx, Object o) // Replaces item at idx with o.
clear() // Removes all items.
isEmpty() // true if empty.
toArray() // returns an array that contains
// the contents of the list
```

## Removing items from ArrayList while iterating

- When an item is removed from an ArrayList, the list shrinks and the indices are renumbered behind the removed item

- Why doesn't this removal work?

```
for (int i=0; i<lst.size(); i++) {
    lst.remove(i);
}
```

- Must remove from the back to the front

```
for (int i=lst.size()-1; i>=0; i--) {
    lst.remove(i);
}
```

## Word class

```
class Word {
    String word;
    int freq;

    Word (String word) {
        this.word = word;
        freq = 1;
    }

    void inc() {
        freq++;
    }

    String getWord() {
        return word;
    }

    int getFreq() {
        return freq;
    }

    String toString() {
        return "<" + word + ",
        "+ freq + ">";
    }
}
```

## Filter

```
String raw;
String delimiters = " ,.?!;:-\\'\"()*![]{}|\\~`@#%&";
String[] fileText, words, uniqueWords;
int[] freqs;

void setup() {
    fileText = loadStrings("EliotLoveSong.txt");
    println("Read " + fileText.length + " lines.");

    raw = join(fileText, " ");
    raw = raw.toLowerCase();

    words = splitTokens(raw, delimiters);
    println("Found " + words.length + " words.");

    freqs = makeUnique(words);
    println("Found " + uniqueWords.length + " unique words.");
}
```

## Filter

```
String raw;
String delimiters = " ,.?!;:-\\'\"()*![]{}|\\~`@#%&";
String[] fileText, words;
ArrayList<Word> uniqueWords;

void setup() {
    fileText = loadStrings("EliotLoveSong.txt");
    println("Read " + fileText.length + " lines.");

    raw = join(fileText, " ");
    raw = raw.toLowerCase();

    words = splitTokens(raw, delimiters);
    println("Found " + words.length + " words.");

    makeUnique(words);
    println("Found " + uniqueWords.size() + " unique words.");
}
```

## makeUnique

```
void makeUnique(String[] words) {
    uniqueWords = new ArrayList();

    for (int i=0; i<words.length; i++) {
        int idx = contains(words[i], uniqueWords);

        if (idx < 0) {
            uniqueWords.add(new Word(words[i]));
        }
        else {
            uniqueWords.get(idx).inc();
        }
    }
}
```

```
int contains(String t, ArrayList<Word> lst) {
    for (int i=0; i<lst.size(); i++) {
        if (t.equals(lst.get(i).getWord()))
            return i;
    }
    return -1;
}
```



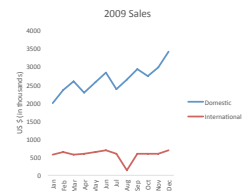


### Chris Rock @ Oscars



### Sales Data (US \$ in thousands)

Region	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Domestic	1983	2343	2593	2283	2574	2838	2382	2634	2938	2739	2883	3413
International	574	636	573	593	644	679	593	139	599	583	602	690



4/25/16

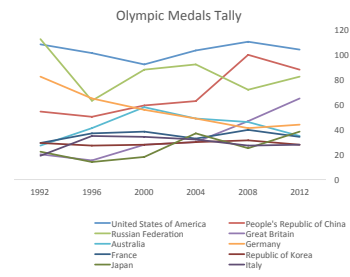
Permeating Data Visualization in CS Courses

20

### Top Medals in Olympics by Country (1992-2012)

Country	2012	2008	2004	2000	1996	1992
United States of America	104	110	103	92	101	108
People's Republic of China	88	100	63	59	50	54
Russian Federation	82	72	92	88	63	112
Great Britain	65	47	30	28	15	20
Australia	35	46	49	58	41	27
Germany	44	41	49	56	65	82
France	34	40	33	38	37	29
Republic of Korea	28	31	30	28	27	29
Japan	38	25	37	18	14	22
Italy	28	27	32	34	35	19

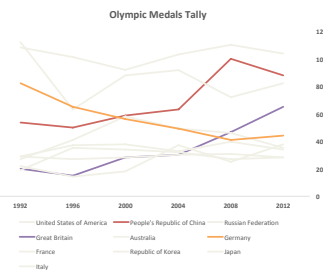
### Top Medals in Olympics (1992-2012)



### Top Medals in Olympics (1992-2012)

Focus on

- Germany
- China
- Great Britain

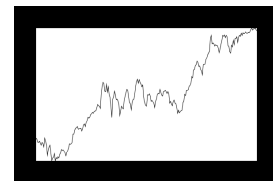


### Example: Visualizing Time Series

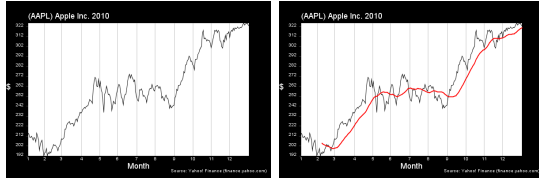
#### Raw Data

```

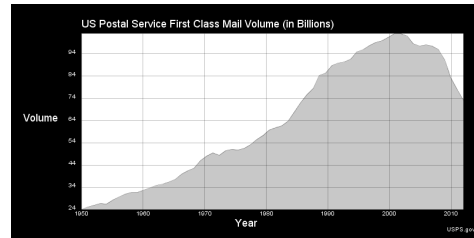
1,4,2010,213.43,214.50,214.01,17633200
1,5,2010,214.60,215.59,214.38,21496600
1,6,2010,214.38,215.23,210.97,19720000
1,7,2010,211.75,212.00,210.58,17040400
1,8,2010,210.30,212.00,211.98,15986100
...
12,27,2010,322.85,325.44,324.68,8922000
12,28,2010,325.91,326.66,325.47,6283000
12,29,2010,326.22,326.45,325.29,5826400
12,30,2010,325.48,325.51,323.66,5624800
12,31,2010,322.95,323.48,322.56,6911000
    
```



### Visualizing Time Series



### Visualizing Time Series



### Data Visualization - Horizons

