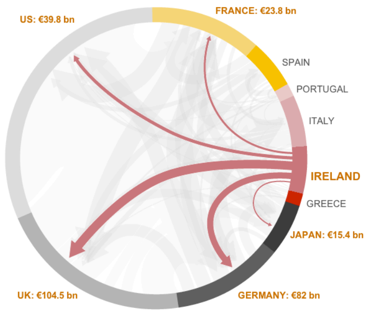
Every death on every road

Who owes how much to whom?



Data Visualization Process

- Acquire - Obtain the data from some source
- Parse - Give the data some structure, clean up,, read in to data structures
- Filter - Remove all but any data of interest
- Mine - Use the data to derive interesting properties
  - Statistical methods/numerical analysis
    - average/median/max/min
    - normalization, rank, percentile rank, correlation coefficient
  - Pattern recognition

Data Visualization Process

- Represent - Chose a visual representation
- Refine – Improve to make it more visually engaging
- Interact - Make it interactive
  - manipulating the data
  - controlling visual features, etc.

Visualizations Methods

- http://www.visual-literacy.org/periodic_table/periodic_table.html

Parsing

- CSV files
  - always split on `","` first
- Special characters
  - `\", \', \\`
- Removing " (or anything else)
  - `int i = str.indexOf("\"");`
  - `String front = str.substring(0, i);`
  - `String back = str.substring(i+1);`
  - `str = front+back;`

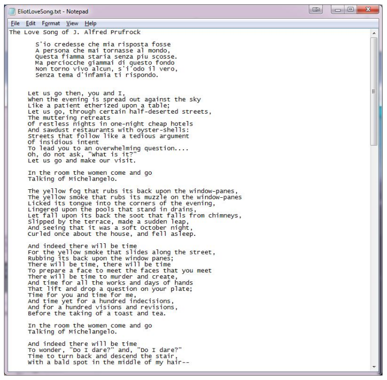
Additional functions

- `join(String[] list, String separator)` - Combines an array of strings into one string, each separated by the character(s) used for the separator parameter.
- `append(array, value)` - Expands an array by one element and adds value to the new position. Type of value must match type of array.

Text Analysis/Text Mining

- Derive high-quality information on patterns and trends in the text via statistical pattern learning
  - Word frequency analysis
  - Sentiment analysis
  - Text categorization
  - Text clustering
- Related fields
  - Computational Linguistics
  - Natural Language Processing
  - Information Retrieval
  - Machine Learning
  - Artificial Intelligence

Acquire



EliotLoveSong.txt - Notepad

```
The Love Song of J. Alfred Prufrock

        S'io credesse che mia risposta fosse
        A persona che mai tornasse al mondo,
        Questa fiamma staria senza piu scosse.
        Ma perciocche giammai di questo fondo
        Non torno vivo alcun, s'i'odo il vero,
        Senza tema d'infamia ti rispondo.


      Let us go then, you and I,
      When the evening is spread out against the sky
      Like a patient etherized upon a table;
      Let us go, through certain half-deserted streets,
      The muttering retreats
      Of restless nights in one-night cheap hotels
      And sawdust restaurants with oyster-shells:
      Streets that follow like a tedious argument
      Of insidious intent
      To lead you to an overwhelming question....
      Oh, do not ask, "What is it?"
      Let us go and make our visit.

      In the room the women come and go
      Talking of Michelangelo.

      The yellow fog that rubs its back upon the window-panes,
      The yellow smoke that rubs its muzzle on the window-panes
      Licked its tongue into the corners of the evening,
      Lingered upon the pools that stand in drains,
      Let fall upon its back the soot that falls from chimneys,
      Slipped by the terrace, made a sudden leap,
      And seeing that it was a soft October night,
      Curled once about the house, and fell asleep.

      And indeed there will be time
      For the yellow smoke that slides along the street,
      Rubbing its back upon the window panes;
      There will be time, there will be time
      To prepare a face to meet the faces that you meet
      There will be time to murder and create,
      And time for all the works and days of hands
      That lift and drop a question on your plate;
      Time for you and time for me,
      And time yet for a hundred indecisions,
      And for a hundred visions and revisions,
      Before the taking of a toast and tea.

      In the room the women come and go
      Talking of Michelangelo.

      And indeed there will be time
      To wonder, "Do I dare?" and, "Do I dare?"
      Time to turn back and descend the stair,
      With a bald spot in the middle of my hair--
```

Parse and Filter

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

Filter

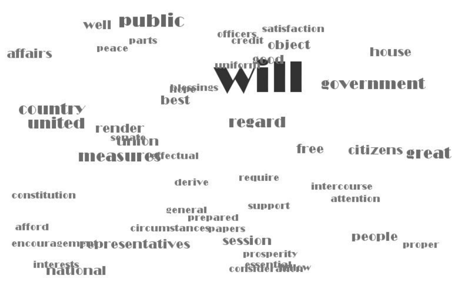
```
String raw;
String delimiters = " ,.?!;:-\'\"()*![]{}|\\~`@#$%^&";
String[] fileText, words, uniqueWords;
int[] freqs;

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.");
}
```



Data Structures

- Ways of storing and organizing data
- Arrays
  - Must know the size ahead of time
  - Can not grow and shrink at will
  - No insertion/deletion without a lot of work
- ArrayList
  - A built-in list that stores and manages an *arbitrary* number of data items of any type (Objects).
  - Objects in an ArrayList are accessed by **index** [0..size-1]

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
```

ArrayList Example – Box Dropper

```
// Box Dropper
ArrayList boxes = new ArrayList();
//ArrayList<Box> = boxes new ArrayList();

void setup() { size(500, 500); }

void draw() {
  background(0);

  for (int i = boxes.size()-1; i>=0; i--) {
    //boxes.get(i).draw();      // Fails. Why?
    Box b = (Box)boxes.get(i);  // Type cast Object->Box
    if(b.update()) {
      boxes.remove(i);
      println(boxes.size() + " boxes remaining");
    }
    else {
      b.draw();
    }
  }
}

void mousePressed() {
  Box b = new Box(mouseX, mouseY);
  boxes.add(b);
  println( boxes.size() + " boxes in ArrayList" );
}
```

```
// A simple Box class
class Box {
  float x, y, v;

  Box(float tx, float ty) {
    x = tx;  // x position
    y = ty;  // y position
    v = 0.0; // y velocity
  }

  void draw() {
    fill(200);
    rect(x, y, 20, 20);
  }

  boolean update(){
    y += v;
    v += 0.02;
    return (y>height);
  }
}
```

- Why can we not call draw() directly on item in ArrayList?
- Why do we loop over ArrayList backwards?

ArrayList Example - Fireworks