Prizes will be awarded for the winning entry in each of the following categories.

- **Featherweight.** At most 128 words of Logo, not including comments and delimiters.
- **Heavyweight.** At most 1024 words of Logo, not including comments and delimiters.

Winners will be selected by popular vote! (Homework 13)

- Static **images** of the output of your programs
- Tonight at midnight: I'll post your Logo implementations!
  - Run them to see these images evolve!
- I will also post a **solution** to the Logo project
  - It runs (almost) all of the contest entries
  - You can use it as a study guide for the final

(Demo)
MapReduce is a framework for batch processing of Big Data.

What does that mean?

- **Framework**: A system used by programmers to build applications.
- **Batch processing**: All the data is available at the outset and results aren't consumed until processing completes.
- **Big Data**: A buzzword used to describe datasets so large that they reveal facts about the world via statistical analysis.

(Demo)

The big ideas that underly MapReduce:

- Datasets are too big to be stored or analyzed on one machine.
- When using multiple machines, systems issues abound.
- Pure functions enable an abstraction barrier between data processing logic and distributed system administration.
Systems research enables the development of applications by defining and implementing abstractions:

- **Operating systems** provide a stable, consistent interface to unreliable, inconsistent hardware

- **Networks** provide a simple, robust data transfer interface to constantly evolving communications infrastructure

- **Databases** provide a declarative interface to software that stores and retrieves information efficiently

- **Distributed systems** provide a single-entity-level interface to a cluster of multiple machines

A unifying property of effective systems:

Hide *complexity*, but retain *flexibility*
The Unix Operating System

Essential features of the Unix operating system (and variants)

- **Portability**: The same operating system on different hardware
- **Multi-Tasking**: Many processes run concurrently on a machine
- **Plain Text**: Data is stored and shared in text format
- **Modularity**: Small tools are composed flexibly via pipes

![Diagram of standard input, process, standard output, standard error, and text input/output]

The *standard streams* in a Unix–like operating system are conceptually similar to Python iterators (Demo)
Python Programs in a Unix Environment

The built-in `input` function reads a line from *standard input*.

The built-in `print` function writes a line to *standard output*.

(Demo)

The values `sys.stdin` and `sys.stdout` also provide access to the Unix *standard streams* as "files."

A Python "file" is an interface that supports iteration, read, and write methods.

Using these "files" takes advantage of the operating system *standard stream* abstraction.

(Demo)
MapReduce Evaluation Model

**Map phase:** Apply a *mapper* function to inputs, emitting a set of *intermediate key–value pairs*

- The *mapper* takes an iterator over inputs, such as text lines.
- The *mapper* yields 0 or more *key–value pairs* per input.

**Reduce phase:** For each *intermediate key*, apply a *reducer* function to accumulate all values associated with that key

- The *reducer takes* an iterator over *key–value pairs*.
- All pairs with a given key are consecutive
- The *reducer yields* 0 or more values for a *key*, each associated with that *intermediate key*. 
MapReduce Evaluation Model

Google MapReduce
Is a Big Data framework
For batch processing

Reduce phase: For each intermediate key, apply a reducer function to accumulate all values associated with that key

- The reducer takes an iterator over key-value pairs.
- All pairs with a given key are consecutive
- The reducer yields 0 or more values for a key, each associated with that intermediate key.
Above-the-Line: Execution model


Tuesday, November 29, 2011
A "task" is a Unix process running on a machine.

http://research.google.com/archive/mapreduce-osdi04-slides/index-auto-0008.html
Python Examples of a MapReduce Application

The mapper and reducer are both self-contained Python programs

• Read from standard input and write to standard output!

```python
#!/usr/bin/env python3
import sys
from ucb import main
from mr import emit

@main
def run():
    for line in sys.stdin:
        emit_vowels(line)

def emit_vowels(line):
    for vowel in 'aeiou':
        count = line.count(vowel)
        if count > 0:
            emit(vowel, count)

Mapper
```

Tell Unix: this is Python

The emit function outputs a key and value as a line of text to standard output

Mapper inputs are lines of text provided to standard input

Mapper inputs are lines of text provided to standard input
The mapper and reducer are both self-contained Python programs:

- Read from standard input and write to standard output!

**Reducer**

```python
#!/usr/bin/env python3
import sys
from ucb import main
from mr import emit, values_by_key

@main
def run():
    for key, value_iterator in values_by_key(sys.stdin):
        emit(key, sum(value_iterator))
```

**Input:** lines of text representing key-value pairs, grouped by key

**Output:** Iterator over (key, value_iterator) pairs that give all values for each key
What Does the MapReduce Framework Provide

**Fault tolerance:** A machine or hard drive might crash
- The MapReduce framework automatically re-runs failed tasks.

**Speed:** Some machine might be slow because it's overloaded
- The framework can run multiple copies of a task and keep the result of the one that finishes first.

**Network locality:** Data transfer is expensive
- The framework tries to schedule map tasks on the machines that hold the data to be processed.

**Monitoring:** Will my job finish before dinner?!?
- The framework provides a web-based interface describing jobs.