

CMSC 330: Organization of Programming Languages

Project 1 – Processing Web Log Files

Overview

- Write Ruby program to process web log files
 - Validate log file format
 - Gather statistics
 - Bytes
 - Time
 - Popularity
- Processing text
 - Common use for scripts
 - Get feel of using scripting languages

CMSC 330

2

Web Log Files

- Produced by web servers to track page access
- Example

```
72.30.61.37 - - [22/Jul/2006:00:00:15 -0400] "GET /~ben/ HTTP/1.0" 200 10175
59.144.5.65 - - [22/Jul/2006:00:00:14 -0400] "GET /~handler/handler.gif HTTP/1.1" 200 23118
85.19.72.153 - - [25/Jul/2006:20:02:02 -0400] "GET /class/fall2005/cmsc417/ HTTP/1.1" 304 0
85.19.72.122 - - [25/Jul/2006:01:02:02 -0400] "GET /class/fall2005/cmsc412/ HTTP/1.1" 304 0
```

CMSC 330

3

Log File Format

- Fields
 1. Numeric IP address
 - 4 numbers separated by a period
 2. Hyphen
 3. Name of the user requesting the page
 - May be "-" if no user
 4. Date the web page was requested
 - [day/month/year:hour:minute:second zone]
 5. Request itself
 - Arbitrary string that begins and ends with double quotes
 6. Status code, a non-negative integer
 7. Number of bytes sent

CMSC 330

4

Weblog.rb Script

- Run as
 - ruby weblog.rb validate <log-file-name>
 - Returns "yes" or "no" for valid/invalid log file
 - ruby weblog.rb bytes <log-file-name>
 - Returns total bytes sent by server (in bytes/KB/MB/GB)
 - ruby weblog.rb time <log-file-name>
 - Returns # of requests for each hour in the day
 - ruby weblog.rb popularity <log-file-name>
 - Returns 10 most popular requests

CMSC 330

5

Project Tasks

- Handle command line arguments
 - ARGV[0]
- Open & read text files
 - file = File.new(<filename>, "r")
- Recognizing text patterns
 - Regular expressions (e.g., /\S+/)
- Processing data
 - Count, compare, add, sort...
- Etc...

CMSC 330

6

Administration

- Project description & files
 - [Download from class web page](#)
- Due midnight Tue, Sept 21st
 - 10% penalty for 1 day late
- Submit weblog.rb to submit server
 - submit.cs.umd.edu
- Public test cases
 - [Sample inputs & outputs available](#)

CMSC 330

7

Getting Started : Counting Words in File

```
file = File.new(ARGV[0], "r")
lines = file.readlines
hash = Hash.new
lines.each { |line|
  words = line.scan(/\S+/)
  words.each { |word|
    if hash[word] == nil
      hash[word] = 1
    else
      hash[word] += 1
    end
  }
}

hash.keys.each {
  |key| puts("word: #{key}, count: #{hash[key]}")
}
```

Get name of
file from
command line
& open file

Use scan to
find words

Use hash to
track words

CMSC 330

8