

CS306 Spring 2009
Homework Assignment #3
Due: Monday, February 16th, 2009 at 4pm

Program 1 – Email mbox Analyzer (150 pts.)

Over the years, emails have been stored on disk in a number of different formats, but one of the most popular is known as the “mbox” format. The mbox format is popular because it is easy – all of the email messages are stored one after another in the same plain text file. The start of a message is indicated by what is known as a “From line”, which is a line that begins with the exact five characters “From “ (including the space after the m). The “From line” is followed by the rest of the email header lines, then a blank line, then the body of the message (which itself may contain one or more blank lines, but these are not of interest). The next message starts when the next “From line” occurs.

A complication arises when a line of an email message body also happens to begin “From ”. To avoid this, all lines in the body that contain this are prepended with a > before they are stored, such that they are stored as “>From “ to avoid false detection as a “From line”. In this way, it can be guaranteed that all “From lines” indicate the start of a new message.

A sample message stored in mbox format looks like:

```
From root@localhost.localdomain Tue Sep 2 04:02:02 2008
Return-Path: <root@localhost.localdomain>
Received: from localhost.localdomain (localhost.localdomain [127.0.0.1])
    by localhost.localdomain (8.13.8/8.13.8) with ESMTTP id m8292209011427
    for <root@localhost.localdomain>; Tue, 2 Sep 2008 04:02:02 -0500
Received: (from root@localhost)
    by localhost.localdomain (8.13.8/8.13.8/Submit) id m82922tM011425;
    Tue, 2 Sep 2008 04:02:02 -0500
Date: Tue, 2 Sep 2008 04:02:02 -0500
Message-Id: <200809020902.m82922tM011425@localhost.localdomain>
To: root@localhost.localdomain
From: logwatch@localhost.localdomain
Subject: Logwatch for localhost.localdomain (Linux)
MIME-Version: 1.0
Content-Transfer-Encoding: 7bit
Content-Type: text/plain; charset="iso-8859-1"
```

```
##### Logwatch 7.3 (03/24/06) #####
    Processing Initiated: Tue Sep 2 04:02:02 2008
    Date Range Processed: yesterday
                        ( 2008-Sep-01 )
                        Period is day.
    Detail Level of Output: 0
    Type of Output: unformatted
    Logfiles for Host: localhost.localdomain
#####
```

----- Disk Space Begin -----

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/mapper/VolGroup00-LogVol00	71G	3.3G	64G	5%	/
/dev/sda1	99M	18M	77M	19%	/boot

----- Disk Space End -----

Logwatch End

```
From root@fran.cis.uab.edu Wed Sep 3 04:02:03 2008
...
```

...where the last From line indicates the start of the next message.

The lines starting with the From line and continuing until the first blank line (more specifically, the first line that contains nothing but whitespace) are known as the mail headers. There are no standard set of mail headers – different mail servers will add different headers, or sometimes have the same headers but in a different order. Mail headers have the format HeaderName: HeaderContent where the name and content are separated by a : and a space. The format of the header content is of course different depending on which header it is.

You can read more about the mbox format at the Qmail man page for mbox - <http://www.qmail.org/man/man5/mbox.html>

Please read this URL to understand more fully how the mbox format works.

The task in this programming assignment is to write a program that can parse an mbox file of arbitrary length that contains 0 or more email messages stored in mbox format. (I have provided a sample file for your convenience on the class website <http://www.cis.uab.edu/cs306/>). The program will analyze the mail headers and body to extract items of interest and compile statistics about the mbox file, producing a final report once parsing and analyzing is complete. The full list of statistics that must be generated is:

1. The total number of messages in the mbox file.
2. The total number of distinct email senders as determined by the sender's email address in the From: header (note: this is -NOT- the “From line” - it's the header that begins “From:”)
3. The number of messages from each sender email address
4. The total number of distinct email recipients as determined by the email address in the To: header
5. The number of messages to each recipient email address
6. The maximum number of body lines for a single message (including any blank lines that were added for padding prior to the next “From line”)
7. The average number of body lines for all of the messages (including any blank lines that were added for padding prior to the next “From line”)
8. The distribution of messages over the hours of the day (in other words, how many messages came from 12:00am-12:59am, from 1:00am-1:59am, etc....) to make it easier, use 24-hour time.
9. The distribution of messages over the days of the week
10. The distribution of messages over the days of the month
11. The average number of hops that the messages took between source and destination (as determined by counting the number of Received: headers in the message header section).

Note that we will grade your program by running it on a different set of mbox files than the sample file.

Helpful Hints and Restrictions:

- For our purposes, assume all From: and To: headers will contain only one sender/recipient. In cases where they contain more than one, just extracting one will be deemed sufficient.
- Writing a regular expression that can capture every possible valid email address is famously hard. However, you can get “pretty close” with a fairly simple regular expression, and if you can successfully capture all of the email addresses that will appear in the sample file, we'll consider that close enough.
- There are no guarantees that a particular header will be present in all messages in an mbox file nor that the headers will appear in a consistent order.
- Do not confuse the “From:” email header with the “From “From line that was placed there by the mbox format. They are two different concepts. The “From line” should only be used to detect the start of the next message. Sender addresses should be parsed from the From: header.
- Similarly, dates and times should be parsed from the Date: header, not the “From line”.
- Your program would benefit from being written like a simple state machine that is aware of when it is in a body vs. when it is in a header. For example, you might want an \$intheheader variable that flips between 1 and 0 at the appropriate lines of input. This will probably save you from doing a lot of needless processing on message body lines.
- Don't try to write this program all at once! Get one requirement working and tested, then add a second requirement, etc... break the problem into smaller subproblems. And think about the algorithm before you just start writing Perl – you may find it helpful to sketch out some pseudocode first.

Sample Output

Your output should look something like this:

```
Total Number of Messages: 180
Distinct Senders: 64
Sender Counts:
  fran@cis.uab.edu: 17
  joe@cis.uab.edu: 5
  bob@gmail.com: 2
  sally@hotmail.com: 12
  ...
Distinct Recipients: 22
Recipient Counts:
  fran@cis.uab.edu: 14
  harry@cis.uab.edu: 3
  tom@cis.uab.edu: 7
  ...
Maximum number of body lines: 1233
Average number of body lines: 23
Hour of Day Distribution:
0: 1
1: 2
2: 2
3: 1
4: 0
5: 2
6: 4
```

```
7: 7
8: 9
9: 17
...
Day of Week Distribution:
Sun: 7
Mon: 31
Tue: 24
Wed: 41
Thu: 33
...
Day of Month Distribution:
1: 3
2: 11
3: 7
4: 19
...
30: 3
31: 21
Average Number of Hops: 2.79
```

BONUS (10 pts)

For items #3 and #5 above (Sender and Recipient Counts), sort the output from maximum number of messages to minimum number of messages.