G64HLL 2009/2010 Session

Coursework 1 (40%)

Due 10th March 2010, 4:00pm, Demo 17th& 24th March 11:00-13:00

I. Introduction

The log files generated by a Web server are the most useful tools in assisting in understanding of how and when the pages and applications of a website are being accessed. The log file contains, among other things, who and when accessed which page. Nearly all of the major web servers use a common format for their log files. These log files contain information such as the IP address of the remote host, the document that was requested, and a time stamp. The syntax for each line of a log file is:

```
Sit logName FullName [data:time: GMToffset] "req file proto" status lengths
```

Here is an example:

```
128.243.246.63 - - [16/Sep/1999:18:21:18 +0100] "GET /manual/index.html HTTP/1.0" 200 2537
```

The above syntax and the eleven items in the example are explained as follow:

Field Names	Meaning	Items in Example
Site	Either an IP address or the symbolic name of the site making the HTTP request	128.243.246.63
logName	Login name of the user who owns the account that is making the HTTP request. Most remote sites don't give out this information for security reasons. If this field is disabled by the host, you see a dash (-) instead of the login name	-
fullName	Full name of the user who owns the account that is making the HTTP request. Most remote sites don't give out this information for security reasons. If this field is disabled by the host, you see a dash (-) instead of the full name. If your server requires a user id in order to fulfil an HTTP request, the user id will be placed in this field.	-
date	Date of the HTTP request	16/Sep/1999
time	Time of the HTTP request. The time will be presented in 24-hour format	18:21:18
GMToffset	Signed offset from Greenwich Mean Time	+01 one hour ahead of GMT
req	HTTP command. For WWW page requests, this field will always start with the GET command	GET
file (see note)	Path and filename of the requested file	/manual/index.html
proto	Type of protocol used for the request	HTTP 1.0
status	Status code (see list below) generated by the request	200
length	Length of requested document	2537 bytes

Note: There are three types of path/filename combinations: Implied Path and Filename-accesses a file in a user's home directory. For example, /~foo/ could be expanded into /user/foo/homepage.html. The /user/foo directory is the home directory for the user foo. And homepage.html is the default file name for any user's home page. Implied paths are hard to analyze because you need to know how the server is set up and because the server's set up may change. Relative Path and Filename-accesses a file in a directory that is specified relative to a user's home directory. For example, /~foo/cooking.html will be expanded into /user/foo/cooking.html. Full Path and Filename-accesses a file by explicitly stating the full directory and filename. For example, /user/foo/biking/mountain/index.html.

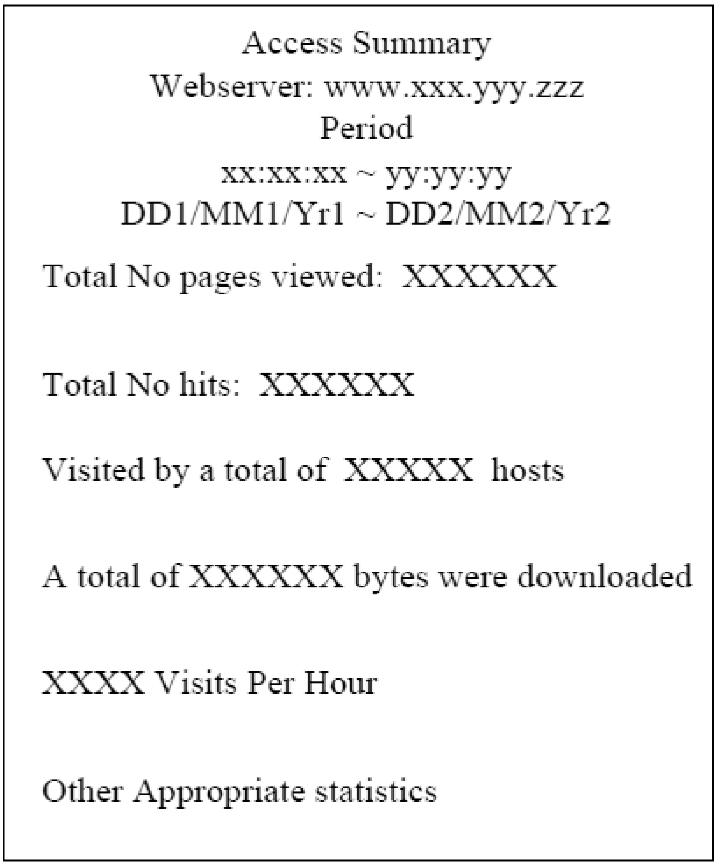
The Most Common Server Status Codes

Status	*Description Code*
200	OK
204	No content
301	Moved permanently
302	Moved temporarily
400	Bad Request
401	Unauthorized
403	Forbidden
404	Not found
500	Internal server error
501	Not implemented
503	Service unavailable

II. Specifications

Write a piece of web server access analysis software using Perl (download the log file from the course web page). The software should produce following outputs:

1. A properly formatted analysis summary page (write it to a file). Here is a possible format



Access Summary
Webserver: www.xxx.yyy.zzz
Period
xx:xx:xx ~ yy:yy:yy
DD1/MM1/Yr1 ~ DD2/MM2/Yr2

Total No pages viewed: XXXXXX

Total No hits: XXXXXX

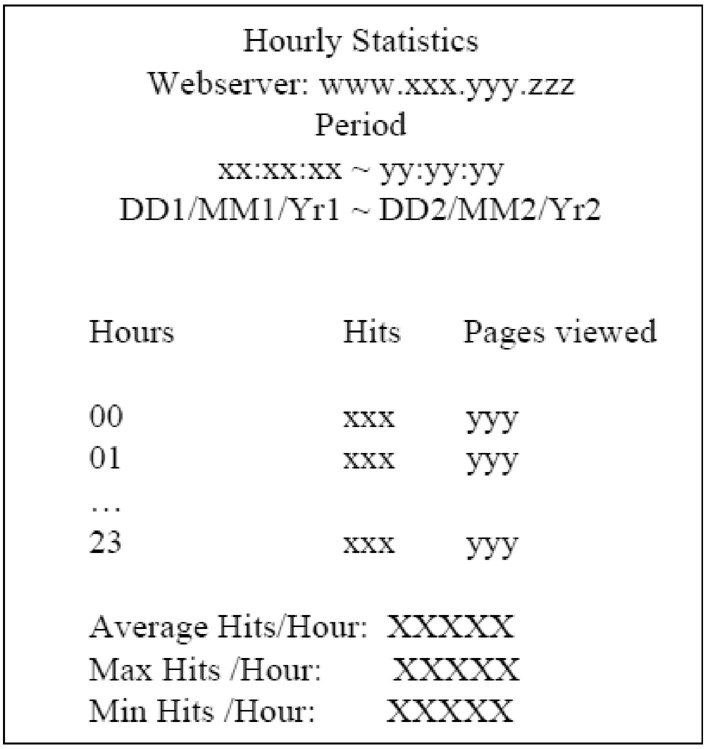
Visited by a total of XXXXX hosts

A total of XXXXXX bytes were downloaded

XXXX Visits Per Hour

Other Appropriate statistics

2. A properly formatted analysis page (write it to a file) of hourly statistics. Here is a possible format

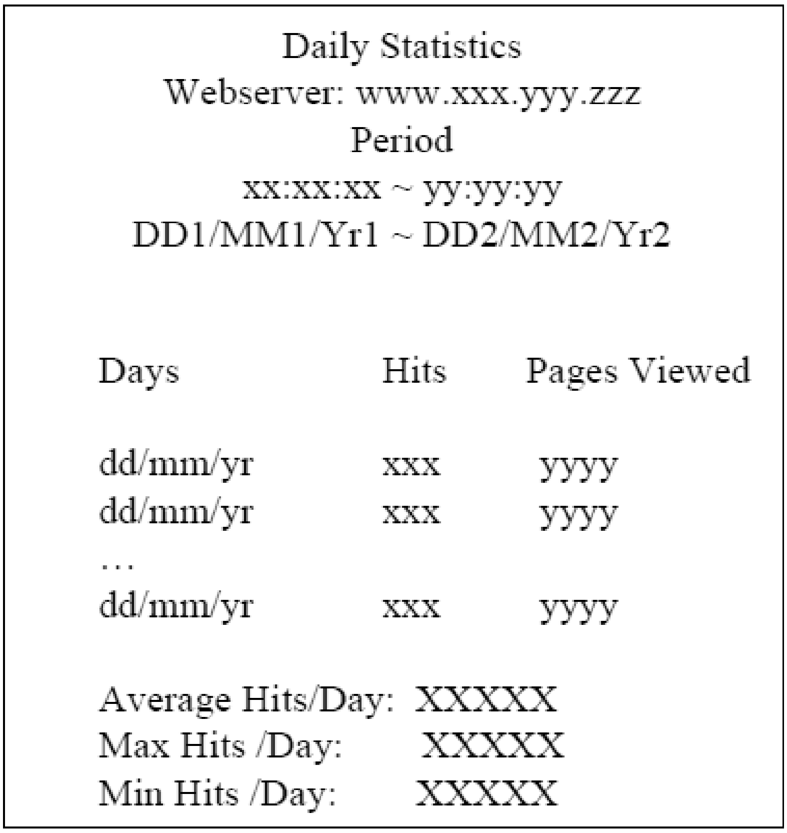


Hourly Statistics
Webserver: www.xxx.yyy.zzz
Period
xx:xx:xx ~ yy:yy:yy
DD1/MM1/Yr1 ~ DD2/MM2/Yr2

Hours	Hits	Pages viewed
00	xxx	yyy
01	xxx	yyy
…		
23	xxx	yyy

Average Hits/Hour: XXXXX
Max Hits /Hour: XXXXX
Min Hits /Hour: XXXXX

3. A properly formatted analysis page (write it to a file) of daily statistics. Here is a possible format



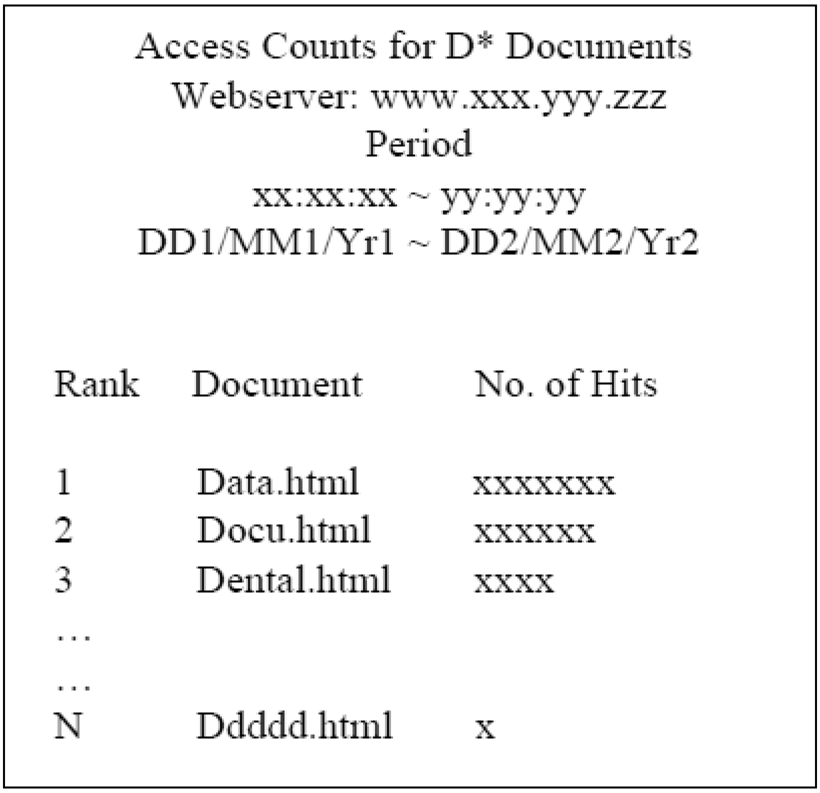
```
                Daily Statistics
          Webserver: www.xxx.yyy.zzz
                     Period
              xx:xx:xx ~ yy:yy:yy
          DD1/MM1/Yr1 ~ DD2/MM2/Yr2


    Days              Hits      Pages Viewed

    dd/mm/yr          xxx         yyyy
    dd/mm/yr          xxx         yyyy
    …
    dd/mm/yr          xxx         yyyy

    Average Hits/Day:  XXXXX
    Max Hits /Day:      XXXXX
    Min Hits /Day:     XXXXX
```

4. An analysis page (write it to a file) that reports the ranking of a particular type of documents, for example, documents beginning with the letter D, according the number of times they are visited in the period. Here is a possible format

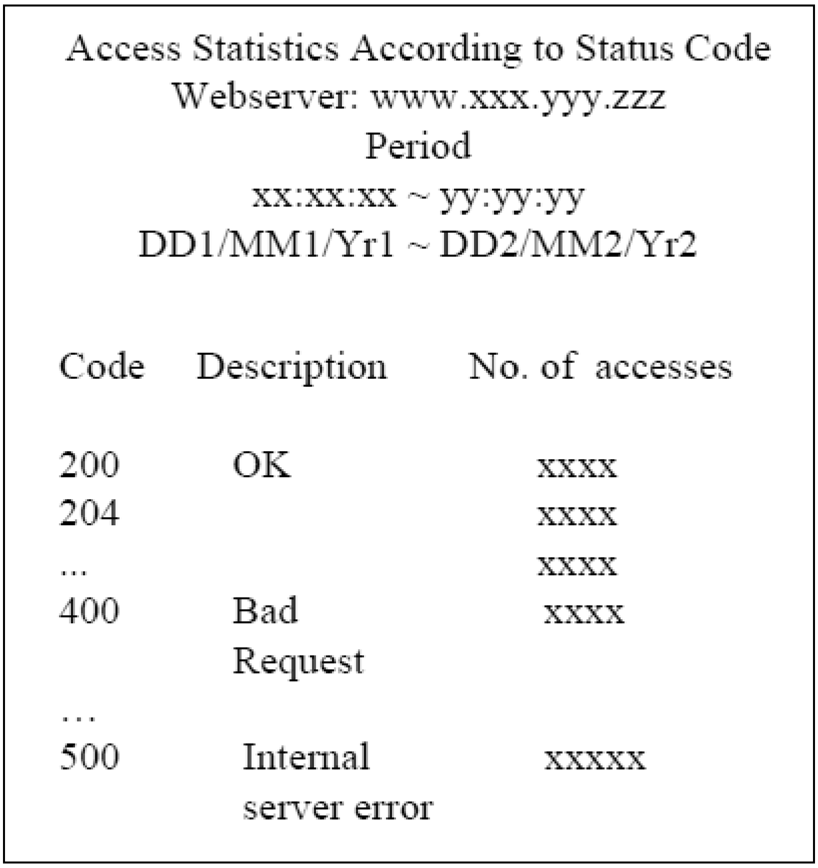


```
          Access Counts for D* Documents
           Webserver: www.xxx.yyy.zzz
                      Period
               xx:xx:xx ~ yy:yy:yy
           DD1/MM1/Yr1 ~ DD2/MM2/Yr2


 Rank     Document          No. of Hits

 1         Data.html        xxxxxxx
 2         Docu.html        xxxxxx
 3         Dental.html      xxxx
 …
 …
 N         Ddddd.html       x
```

5. An analysis page (write it to a file) gives out statistics according to status Code. Here is a possible format



```
      Access Statistics According to Status Code
           Webserver: www.xxx.yyy.zzz
                      Period
                 xx:xx:xx ~ yy:yy:yy
            DD1/MM1/Yr1 ~ DD2/MM2/Yr2


Code   Description      No. of  accesses

200       OK                  xxxx
204                           xxxx
...                           xxxx
400       Bad                  xxxx
          Request
...
500        Internal            xxxxx
           server error
```

III. Tasks

1. Use appropriate UML diagrams to model your programme.
2. Implement the programme using Perl

IV. What to hand in

1. Hand in a hard copy of your design (UML diagrams), your code which must be appropriately commented, and print outs of your analysis output pages.
2. Submit the source code and output files through the CW system (http://support.cs.nott.ac.uk/coursework/cwstud/)

V. Demo

You should demonstrate the running of your program in the Lab on 17th and 24th March. Demo timetable will be given out later. The demonstration will be based on the code you submitted (so any new version after the deadline will not count).

VI. Assessment

Assessment will be based on design/UML diagrams (10%), correctness and style of analysis outputs (30%), clearly explained and commented code (30%) and demo (30%).