University of Technology Sydney
Faculty of Engineering and Information Technology

Subject: 32547 UNIX Systems Programming, Autumn 2017

Assignment

Description

This assignment is an individual programming assignment using Perl. It addresses objectives 2 and 3 as listed in the Subject Outline document.

No limits apply to the number of lines of code of the program.

Assignments are **to be completed individually** (this might be checked with the use of anti-plagiarism tools such as Turnitin). **You should not receive help in the preparation of this assignment, nor ask anyone else to prepare it on your behalf in any way**.

Marks

The assignment corresponds to 30% of the total marks.

Submission

The completed assignment is due by **5:00pm of Friday 9 June 2017**.

PLEASE PAY CAREFUL ATTENTION TO THE FOLLOWING SUBMISSION INSTRUCTIONS:

1. The assignment has to be submitted electronically as a single ZIPPED file attached to an email to the Subject Co-ordinator, Massimo Piccardi (Massimo.Piccardi@uts.edu.au).
2. The email's subject must be: **32547 USP - assignment submission - <*yourStudentID*>**. Example: **32547 - assignment submission - 99887799**.

   Email without this subject or with a different subject may not be accepted as a valid submission. Submit the email **from your UTS email account**, otherwise the submission may get blocked or not be received.
3. The filename for the zipped archive attached to the email must be: **USP_<*yourSurname*>_<*yourStudentID*>.zip**.
   Example: **USP_Piccardi_99887799.zip**. Archives with a different filename may not be accepted as a valid submission.
4. Late assignments will be deducted one mark per day late, more than seven days late the assignment will receive zero. Special consideration, for late submission, must be arranged in advance with the Subject Co-ordinator.
5. If you would like a receipt, just please add a request for a Return Receipt to your email. No other types of receipts can be guaranteed.

If you do not follow the instructions above, your file may not be received or tracked correctly and thus may not be marked.

Academic Standards

The assignments in this Subject should be your own original work. Any code taken from a textbook, journal, the Internet or any other source should be acknowledged. For referencing, please use the standard referencing conventions (http://www.lib.uts.edu.au/help/referencing).

Marking Scheme

Mark Range	
30	All requirements of the assignment are met. The submitted program can be executed by the lecturer "as is" and produces the requested output.
24-29	The program works correctly with any valid file and for all options, but its execution experiences some minor problems.
18-23	The program does not work as expected with one of the options.
0-17	This range goes from no submission at all (0 marks) to a submission which does not meet major requirements of the assignment. Examples: • the program does not work as expected with two or more options; • the program generates unhandled errors; • the program does not solve the assignment problem.

The assignments will be marked within two weeks from the submission deadline.
Important notes:

- **Submission of this assignment is not compulsory to pass the subject; do not feel that you have to submit it "at all costs" and perhaps be tempted to seek unauthorised assistance**.
- There are no minimum requirements on the marks on this assignment to pass the subject.

Title: Summary of network usage statistics with Perl

In this assignment, you will write a Perl program which parses a file containing information about the network usage of a Unix computer and outputs summary information.

These are the specifications for your Perl program:

1. It must be named *net_stat.pl*

2. It must be invoked as:

   `net_stat.pl` *`option net_usage_file`*

   The program must check that: a) it is invoked with the appropriate number of arguments, and b) the values for the *option* argument are as described below, and c) that the *net_usage_file* argument is a readable file. The examples below clarify the valid invocations. In all other cases, the program must print a message to the standard output explaining to the user how the program should be invoked correctly, and exit.

3. File *net_usage_file* can have any arbitrary name. It must be a file of text with the following format:

   a. The first line contains names of fields.

   b. The first line is followed by an arbitrary number of other lines (including, possibly, zero lines).

   c. Each of the lines after the first must contain seven fields aligned as in the example below. The seven fields are:

      - *Interface*: the name of the interface; width: 10 characters;
      - *Rx-ok*: the number of correctly received units; width: 8 characters;
      - *Rx-err*: the number of units received with errors; width: 8 characters;
      - *Rx-drp*: the number of received units dropped; width: 8 characters;
      - *Tx-ok*: the number of correctly transmitted units; width: 8 characters;
      - *Tx-err*: the number of units transmitted with errors; width: 8 characters;
      - *Tx-drp*: the number of transmitted units dropped; width: 8 characters.

   d. The *Interface* field is a string of characters. Acceptable characters include: lower case letters, upper case letters, and digits. The same string can appear only in one line.

   e. Every other field is a string of decimal digits.

   The following is an example of file *net_usage_file*:

```
Interface    Rx-ok  Rx-err  Rx-drp   Tx-ok  Tx-err  Tx-drp
eth0       2450012      12      20  640000       0       0
lo           16356       0       0   44333       0       0
wlan0      4800000     898     645 7812167     345       0
```

Very important note: your program **is not expected to verify** that file *net_usage_file* complies with the above specifications. It will only be tested with compliant files.

4. Your program can be invoked with option: -n. In this case, it must print the following:

   `Interfaces:`
   <first interface's name in appearance order>
   <second interface's name in appearance order>
   …
   <last interface's name in appearance order>

   Example with the example *net_usage_file* given above:

   Command line:

   `net_stat.pl -n net_usage_file`

   Output:

```
Interfaces:
eth0
lo
wlan0
```

   In the case in which file *net_usage_file* contains only the first line, your program must instead only print:

   `No interfaces present`

5. Your program can be invoked with option: -r. In this case, it must only print the following string:

   `Total number of units received:` *<total number of units received for all interfaces, including ok, err and drp>*

   Example with the example *net_usage_file* given above:

   Command line:

   `net_stat.pl –r net_usage_file`

   Output:

   `Total number of units received: 7267943`

   In the case in which file *net_usage_file* contains only the first line, your program must print:

   `Total number of units received: 0`

6. Your program can be invoked with option: -t. In this case, it must only print the following string:

   `Total number of units transmitted:` *<total number of units transmitted for all Interfaces, including ok, err and drp>*

   Example with the example *net_usage_file* given above:

Command line:

`net_stat.pl –t net_usage_file`

Output:

```
Total number of units transmitted: 8496845
```

In the case in which file *net_usage_file* contains only the first line, your program must print:

```
Total number of units transmitted: 0
```

7. Your program can be invoked with option: -i <*interface's name*>. In this case, it must print:

`Interface` <*interface's name*>`:`
`Total number of units received:` <*total number of units received for that Interface, including ok, err and drp*>
`Percentage of correctly received units:` <*percentage of correctly received units for that Interface, computed as Rx-ok * 100/(Rx-ok + Rx-err + Rx-drp) and printed truncated to 3 decimal digits after the point and followed by '%'*>
`Total number of units transmitted:` <*total number of units transmitted for that Interface, including ok, err and drp*>
`Percentage of correctly transmitted units:` <*percentage of correctly transmitted units for that Interface, computed as Tx-ok * 100/(Tx-ok + Tx-err + Tx-drp) and printed truncated to 3 decimal digits after the point and followed by '%'*>

Example with the example *net_usage_file* given above:

Command line:

`net_stat.pl –i wlan0 net_usage_file`

Output:

```
Interface wlan0:
Total number of units received: 4801543
Percentage of correctly received units: 99.968%
Total number of units transmitted: 7812512
Percentage of correctly transmitted units: 99.996%
```

If the percentage of correctly received units or correctly transmitted units is below 99%, the program must also add the following line to the output above:

```
Attention: significant errors over this interface
```

In the case in which interface *interface's name* is not present in *net_usage_file*, your program must print:

`Interface` <*interface's name*> `not found`

8. Your program can be invoked with option: -v. In this case, it must only print your name, surname, student ID and date of completion of your assignment, using a format of your choice.

9. No options can be used simultaneously. This means that your program can only be invoked with one of the options at a time.

10. Zip your file *net_stat.pl* into a file named USP_<*yourSurname*>_<*yourStudentID*>*.zip* and submit it with the modalities specified above. Several free zip utilities are available on the WWW.

Please be reminded that:

- **this assignment must be your own work and you should not be helped by anyone to prepare it in any way; your assignment may be tested by anti-plagiarism software that detects superficial changes such as changing variable names, swapping lines of code and the like.**
- **the submission of the assignment is not compulsory to pass the subject and it only counts for ten marks;**
- **understanding the assignment specifications is part of the assignment itself and no further instructions will be provided; on the other hand, whatever is not constrained you can implement it according to your own best judgment.**