School of Electronic Engineering
and Computer Science

Mini-Project Coursework: A Network Analyser GUI

Module code: **EBU4201**
Module title: **Introductory Java Programming**
Hand-out date: **7th May 2020**
Hand-in date: **22nd May 2020**
Marks available: **50**
Feedback: **Individual marking sheet including feedback comments and a mark out of 50.**

Introduction: You are asked to write a graphical user interface (GUI) application to execute the fundamental network diagnostic tool ping[1] and display its raw output together with a histogram of Round Trip Time (RTT) values.

Task 1 [30 marks]

Once launched with the command `java NetAnalyser`, your interface should look like Figure 1.

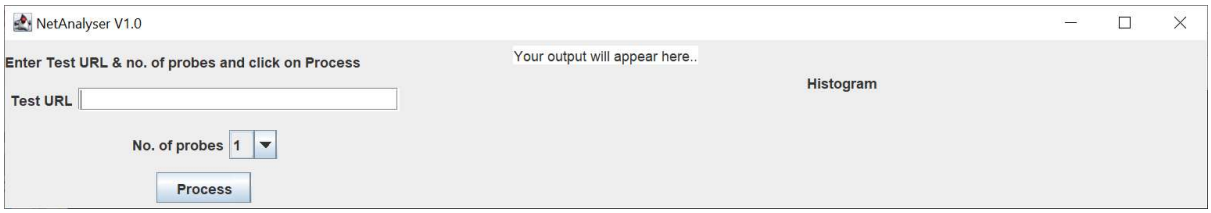

Figure 1

Once the user clicks on the 'Process' button (having entered the Test URL and selected the desired number of probes), your interface should look like Figure 2.

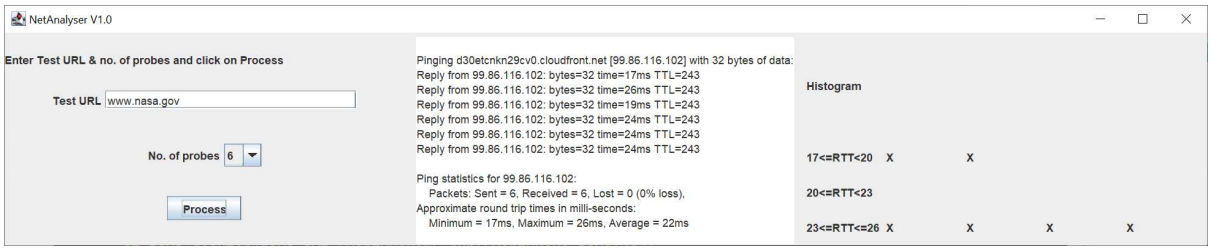

Figure 2

See the notes below for details.

[1] https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/ping

1) The number of probes should allow values from 1 to 10 inclusive. You should make sure the user can only input a value in the range 1-10.
2) The middle section should display the raw output from the ping command (which you will see if you ran `ping` directly on the command window).
3) The right-hand section should display a histogram of the RTT values which are listed as "time=xms" in the raw `ping` output. There would usually be one value for each probe (see note below).
   Note: On some occasions some of the lines may contain a "time-out" (and no RTT value) instead of a "reply" and on certain occasions there will be an error with no "reply" at all (e.g. due to network issues or an incorrect URL). For this task you can assume a valid URL will be provided and at least one probe will receive a reply with an RTT value.

4) Recommended means of calling the `ping` command from within Java[2]:

```
Process p = Runtime.getRuntime().exec("cmd /c ping -n 5
www.bbc.co.uk");

p.waitFor();

BufferedReader reader=new BufferedReader(new
InputStreamReader(p.getInputStream()));
```

   where `reader` will contain the output from the DOS command `ping`.

5) You will need to extract RTT values from the raw output, using appropriate String class methods.
   Note 1: the RTT values do NOT always start at the same position as the length of the IP address would vary.
   Note 2: the number of digits in the RTT values will vary depending on the test URL!

6) Guide to building a histogram:

   bin size = (maxRTT-minRTT)/3 (i.e. aim to have 3 bins)
   e.g. in Figure 2:
   RTT values: 17, 26, 19, 24, 24, 24
   maxRTT = 24
   minRTT = 17
   binSize = (26-17)/3 = 3

Bin (size=3)	**Frequency**
17<=RTT<20	2
20<=RTT<23	0
23<=RTT<=26	4

   Note: you must build the histogram using the steps above and must not use any special classes from the API such as ***javax.media.jai.Histogram.***

[2] Assuming a Windows OS.

7) Display the histogram using a marker (e.g. 'X' or '*') or by using a colour. Only basic components such as JLabel or JTextField should be used (i.e. do NOT use Graphics or any other special classes from the API.)
   e.g. the implementation shown in Figure Figure 2 has used JLabel components displaying an appropriate number of 'X' according to each frequency .
8) It should be possible to repeat the experiment without having to relaunch the application, i.e. providing new input values and clicking on the Process button should clear all previous results and display new results.
9) The interface should have a good layout even when resized; e.g. the button should NOT resize when the frame is resized.

Note: *All the necessary files should be placed in a directory called Task1.*

Task 2 [8 marks]

1) Extend your program so that the maximum number of probes can be provided as a command line argument, between 10 and 20 inclusive (e.g. `java NetAnalyser 15` should display an interface allowing the user to select a number of probes between 1 and 15 inclusive). The program should terminate if no valid argument is provided.
   Note: You may need to change the maximum frequency of a bin accordingly.

2) Write the histogram data to a text file along with date and time stamps as shown in Figure 3. The filename should consist of the URL, date and time.

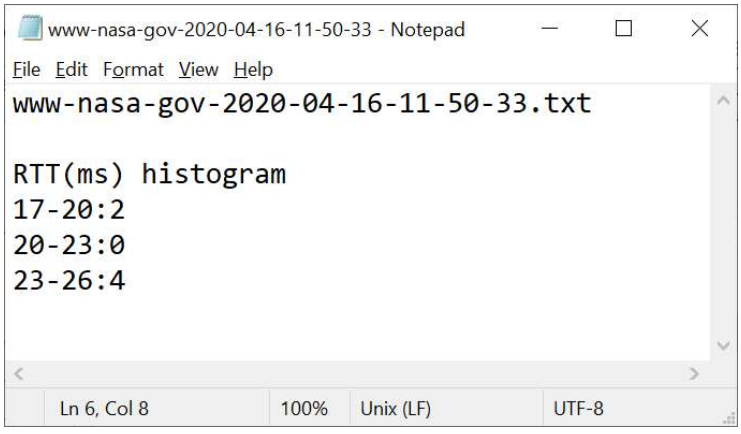

Figure 3

IMPORTANT: The extended application must still provide the functionality developed for **TASK 1**.

Note 1: *you must use the java.time package in order to obtain the date and time.*

Note 2: *All the necessary files (including any reused ones from Task 1) should be placed in a directory called Task2.*

Documentation [12 marks]

Your submitted work must include:

a. Automatically generated Javadoc files.

b. Comments (both internal and Javadocs) in your code.

c. User Manual; this should be a document[3] with no more than two A4 pages which must include instructions on how to run the program (i.e. both how to start it, and how to use it).

Note: *All documentation files must be placed in a directory called Documentation.*

Extra credit [4 marks]

Extra marks from this section can be used to top up your final grade for this project. Maximum mark is still 50.

1) Perform input validation of the test URL, e.g. checking it is non-empty and contains at least one dot. Program should gracefully terminate if the input URL is invalid.

2) Improve your program so it gracefully terminates where the ping output results in an error, i.e. with no "reply" lines (e.g. server cannot be reached).

IMPORTANT: The extended application must still provide the functionalities developed for **TASK 1 and TASK 2**.

Note: *All the necessary files (including any reused ones from Task 1 and Task 2) should be placed in a directory called ExtraCredit.*

Marking Scheme

Marks will be awarded for the following:

1. a clearly laid out interface,
2. correctly functioning code and,
3. clearly structured code including comments and sensible variable, class, and method names.
4. properly designed classes following object oriented principles. E.g. do NOT write everything in the main method, keep code repetition to a minimum (i.e. use methods), do NOT use static methods unless there is a good reason.

Note 1: *The main program file must be called* ***NetAnalyser.java****, and it must compile and run from the command line; otherwise, it will not be possible to give marks for any implemented functionality.*

Note 2: *OpenJDK 13 must be used, as instructed in TW1 at* *https://qmplus.qmul.ac.uk/pluginfile.php/1552074/mod_label/intro/EBU4201-2019.20_JDKandCommandLine.pdf.*

[3] Accepted document formats include: .txt, .docx and .pdf.

Note 3: *All documentation files should be placed in a directory called Documentation.*

Submission Instructions

You must zip all the following files:

1. The **directories created in Task1, Task2 and Extra Credit (if applicable)**, including all **.java** files produced[4].
2. The **Documentation** directory, including all *Javadoc* comments and the **User Manual**.

Notes:

- You must name your .zip file **201821xxxx.zip**, where **201821xxxx** is your BUPT student number.
- You must submit your .zip file to the EBU4201 course area in QMplus[5], under the assignment activity **Mini-Project (submission)**.

IMPORTANT: This is an individual piece of assessed coursework; therefore, students must not work in groups and must not share code solutions. Students must not post (partial) code solutions when asking questions about the Mini-Project in QMplus.

[4] It will not be possible to mark your work if you only provide .class files.

[5] Any work sent via email will be ignored and not marked.