Newcastle University

School of Engineering

EEE2007 – COMPUTER SYSTEMS AND PROGRAMMING
PROJECT 1: DATA GENERATION, INFERENCE AND ANALYSIS

Course work Objectives

1. To develop C++ programs that can generate data and analyse them, and
2. To understand how memory accesses affect the program performance.

Project Description:

PART 1:

Total: 40 Marks

1. Write a program that can generate a one-dimensional matrix of characters randomly. The allowed characters are 'A', 'C', 'T' and 'G' (all capitals). Your program should be named as '*part1.cpp*'.

   15 Marks

2. Your program should ask the user for the number of characters to be generated. Your program should ideally allow the maximum number of characters possible by your memory subsystem. The higher it can allow the better.

   10 Marks

3. Now generate all 4-character combinations for 'A', 'C', 'T' and 'G', including repetitions of any character, and generate a statistics for each combination, stating how many times it appears in the random matrix of characters. You should ideally print the statistics on the standard console; each entry must be followed by a line feed. For example, three lines in the console could look like this:

   ```
   ACCC, 42421
   ACCG, 44211
   ACCT, 33221
   ```

   Use arrays, and nested loops within the main function, where permissible.

   15 Marks

PART 2:

Total: 60 Marks

4. Using your program in PART 1, write a separate library called 'data.hpp', and write another program named as 'part2.cpp'. The file 'part2.cpp' should include the 'data.hpp' library, together with others. The 'data.hpp' library should include all data related functions, e.g.:
   a. The function to generate data
   b. The function to create the next combination using the current combination
   c. The function to count the number of occurrences in the data for a given combination
   d. The function to print the outputs in a separate file called 'analysis.csv', in a comma separated format; each entry must be followed by a line feed. For example, three lines in the file could look like this:

   ```
   ACCC, 42421
   ACCG, 44211
   ```

ACCT, 33221

15 Marks

e. You should use call by reference for all functions above, when permissible, and use loops and arrays, where possible.

15 Marks

f. Make your main function parameterizable, so that you can instruct the internal functions to generate and analyse the data as follows:

```
./part2.exe –n 87385783 –t 5
```

Where 'part2.exe' is the executable generated from the program, '–n' denotes the number of characters, followed by its value, and '–t' denotes the size of character tokens, followed by its value. Note, instead of 4 tokens as in part 1, this parameter suggests creating the tokens of five characters each, e.g. 'AACCT' and 'AACGA'.

Your solution to this should include passing arguments from 'part2.cpp' and using variadic functions, if possible, in the 'data.hpp', where you declare and define the functions.

20 Marks

g. Generate some timing statistics for each function using clock(..) included in 'time.h' file (see examples in the class or elsewhere), and record how their execution times vary with increasing 'n' and 't' values. Save the execution times for each n and t value in an organised table within an Excel file named 'stats.xlsx'.

10 Marks

Deliverables

You should submit all source code organised in folders.
'part1' folder should contain only 'part1.cpp'.
'part2' folder should contain 'data.hpp', 'part2.cpp' and 'stats.xlsx'.
You should create an archive (e.g. a zip file) and attach this as a single file for submission on the ***Turnitin*** potal (this will be made available nearer the deadline).

Marking

This part of the module is assessed by 20% programming assessment. This assignment will constitute 10% of the total mark for the module. The marks' distribution is already shown in the problem descriptions, totalling 100. The actual mark will be scaled to the module's marks afterwards.

MARKING INSTRUCTIONS

Simplified and correct codes, together with proper comments and annotations will merit up to 80% marks. But, additional marks will be awarded if the following feature is observed:

* Memory-efficient solution, that also optimises for performance of both programs.

Plagiarism is strictly prohibited as it may result in serious penalties. The university uses turnitin software, which has extensive database containing internal (code and reports) and external (codes and reports on the internet) sources. There would be informal tutorial discussions on this assignment after the regular lecture hours. The submission deadline is strictly **23rd Nov 2018**, after which the submissions would be considered late and usual late submission rules would apply.

Feedback

After your assignment has been marked, feedback is provided as follows:

1) Blackboard: the document submitted has been annotated with little 'blue balloons' in the usual manner. I will notify you when this information becomes available.

2) Email: Shortly after the Blackboard feedback becomes available, a feedback sheet (a single page pdf file) with a detailed breakdown of your marks will be communicated to everyone.