COMP9319 YYT2 Sample Exam

Thursday DDth August YYYY

Marks: 100

Time: 10 minutes reading + 2 hours working

Please read **all** of the instructions below while you are waiting.

Instructions

Start-of-exam Instructions

- Place your student card/other photo ID on the desk.
- Wait for the supervisor to announce the start of reading time.
- Once reading time has started, click the button below to reveal the questions.

End-of-exam Instructions

- Stop working when your supervisor tells you to do so.
- Log out from your workstation.
- Take all of your belongings.

Exam Instructions

- All the files you need to work on the exam are in your home directory.
- Right click on the desktop to bring up the menu and open a terminal.
- Follow the submission instructions under each question to submit your work.
- You can submit your answers multiple times. Only your last submission will be marked.
- **SAVE your changes and SUBMIT each question as you finish working on it.** Do not wait until just before the end of the exam to submit all your answers.
- **To check what you have submitted so far**, use the `submit` command (**with no arguments**).
- If you have any questions during the exam, you must raise your hand and ask a supervisor.
- The following conditions apply to programming questions:
  - Solutions which do not attempt to solve the question generally but instead only hardcode return values for specific tests will receive zero marks.
  - Code style is not marked. However, good style may help a marker understand your code better, which will give you a greater chance of being awarded partial marks if your code does not work.
  - We will not measure the timing of your program, however, any test that runs more than 15 seconds will be terminated (e.g., in case it may have gone into an infinite loop).

Structure

This exam consists of two parts:

- Programming questions (2 questions)
- Written short-answer questions (6 questions)

There are **8 questions** in total.

Resources

The lecture slides from this term are available for you to use. You do not need to reference them.

You can access them via this page: **lecture slides**

You can also access the C/C++ quick reference via this page: **C/C++ Reference**

Question 1 (20 marks)

Write an encoder (called q1) the static Arithmetic Coding algorithm as presented in the lecture week 1 in C or C++. The encoder takes a filename as a commandline argument. The input file may contain at least 1 and at most 10 characters with ASCII value between 0 and 126 inclusively. Assume that the symbols will divide the number line [0, 1) according to their lexicographical order, the encoder will output the low and the high (separated by a space) of the final result to the standard output, as shown in the example below. No penalty to your solution if more than necessary number of decimal digits are printed.

```
%vx06> cat test1.txt
BILL GATES%vx06>
%vx06> ./q1 test1.txt
0.2572167752 0.2572167756
%vx06>
```

You are given a string sorting program, `q1/q1.cc`, and you shall modify this program to fulfill the requirements of this question.

Files (in q1/)

Makefile	A Makefile to compile your code
q1.cc	A program currently read a string from stdin, sort the characters of the string, and outputs the result to stdout. This is the file you should modify (you may also need to modify Makefile for gcc/g++ and other flags).

Testing

You can compile and test your functions using the following commands:

```
$ make                  # compiles the program
$ ./q1 input-file       # tests with an input file, outputs to terminal
```

Note that you are expected to devise your own tests and check the output yourself.

Instructions

Modify and put all your code in the provided `q1.cc`. Modify the makefile accordingly.

Submission

Submit using the following command:

```
$ submit q1
```

Question 2 (15 marks)

Suppose T is a text file that contains at least 1 and at most 5120 characters with ASCII value between 0 and 126 inclusively, and it is always ended with a newline \n (the only newline in the file). Write a BWT decoder (called q2) that reverses the BWT encoded T back to the original T. q2 accepts two commandline arguments: the name of a BWT encoded file; the output filename. The following is a simple test of q2 using the provided sample test:

```
vx06 % cat tests/q2a.txt
%vx06> ./q2 tests/test2.bwt myoutput.txt
%vx06> diff myoutput.txt tests/test2.txt
%vx06>
```

You are given a string sorting program, `q2/q2.cc`, and you shall modify this program to fulfill the requirements of this question.

Files (in q2/)

Makefile	A Makefile to compile your code
q2.cc	A program currently read a string from stdin, sort the characters of the string, and outputs the result to stdout. This is the file you should modify (you may also need to modify Makefile for gcc/g++ and other flags).
tests/	A directory containing the 2 sets of test files: test1.txt test1.bwt test2.txt test2.bwt

Testing

You can compile and test your functions using the following commands:

```
$ make                          # compiles the program
$ ./q2 input-file output-file   # tests with an input file, outputs to an output file
$ ./q2 tests/test2.bwt output.txt   # for example, tests with input from tests/test2.bwt
```

It is possible to devise your own tests by creating your own input files. See the existing input files for examples. Note that you will need to derive and check the output yourself.

Instructions

Modify and put all your code in the provided `q2.cc`. Modify the makefile accordingly.

Submission

Submit using the following command:

```
$ submit q2
```

Question 3 (10 marks)

Write your answers for this question in **q3.txt**.

Using the complete Boyer-Moore algorithm (i.e., with bad character and good suffix rules) to find the pattern: **abacab** from the text: **abacaabaccabacab**

How many comparisons are needed to find the match?

Submission

Submit using the following command:

```
$ submit q3
```

Question 4 (12 marks)

Write your answers for this question in **q4.txt**.

Given an Adaptive Huffman (Vitter's) encoded bitstream:

0110000100011000100111101

The initial coding before any transmission is: a=**01100001**, b=**01100010**.

Derive its corresponding decoded message (i.e., the output produced by its corresponding decoder).

Submission

Submit using the following command:

```
$ submit q4
```

Question 5 (12 marks)

Write your answers for this question in **q5.txt**.

Consider the XBW representation of an XML tree T using the two arrays:

CbaDDcDaBABccab

100101010011011

where letters in lowercase are leaf nodes; uppercase letters are internal nodes.

How many leaf nodes will be in the result set when we evaluate the XPath **//B/D** on this XML tree ?

Submission

Submit using the following command:

```
$ submit q5
```

Question 6 (10 marks)

Write your answers for this question in **q6.txt**.

Given character sequence S:

```
database$
```

Derive the Move-to-Front transform of the BWT(S), assuming we use the 255 ASCII symbols as the symbol table. Note: If you need an ASCII table, use the "ascii" command available on the CSE lab machines.

Submission

Submit using the following command:

```
$ submit q6
```

Question 7 (11 marks)

Write your answers for this question in **q7.txt**.

Suppose that the RLFM encoded string of text T is **cgc$agagatc** where $ is the last character of T. Its corresponding bit array B is **11010111011110011**

Derive the last 4 characters of T.

Submission

Submit using the following command:

```
$ submit q7
```

Question 8 (10 marks)

Write your answers for this question in **q8.txt**.

What is the suffix array for string S = **enhancement$**?

Submission

Submit using the following command:

```
$ submit q8
```

This is the end of the exam.

We wish you all the best in your future studies!