TCP2101 Algorithm Design & Analysis

Assignment 40%

Deadline is on 5 June 2023 Week 12 Monday

Instruction section

- This is a group assignment with the minimum of 3 persons to maximum 4 persons per group based on lecture section number.
- Implement the assignment using C++ language. The code should be able to handle the sample set of data given in the topic descriptions.
- Write a report to describe the algorithms implemented, step-by-step algorithm illustrations, results of the experiments, screenshots, code parts, discussions with reasoning for the results obtained and a conclusion on the algorithms such as suitability, possible improvements etc. Include all the citations and references in APA style in your report on the same page you cited.
- Group leader shall zip the submission (one report in docx, one cpp file for each algorithm, input files, output files, one mp4 video file) and following instructions. Start the assignment as soon as possible. No extension of submission deadline will be made. To make testing easier and save time during presentation, your program shall never clear the computer screen and output to the files.
- You are required to follow algorithms and formats based on the given input and output files.
- For all your random input file generations, the seed number is your group leader student id.
- All members must present to explain the algorithms, demonstrate the working programs, present the experimental results, conclude the findings, and etc. No mark will be given to the sleeping member or any instruction is not followed. Book the interview date, time and venue with your respective tutor. There will be an interview and Q&A session for the assignment to validate your work.
- You are reminded that no plagiarism is allowed in any means. If any plagiarism is found, zero mark will be awarded.
- Do not share your code with anyone. If detected, all groups involved will be penalized with zero marks.

Recommendation section

You are to distribute the programming tasks among the group and meet at least 3 times a week to report your progress and discuss the problems you are facing with other group members. You can try to divide the programming tasks accordingly:

- Algorithm 1 (random input file generations, random seed number is your group leader student id) implementation and reporting and other questions
- Algorithm 2 (Kruskal without priority queue) implementation and reporting and other questions
- Algorithm 3 (Kruskal with priority queue) implementation and reporting and other questions
- Algorithm 4 (Huffman coding) implementation and reporting and other questions

You are also advised to integrate your program from the beginning and only share the bug-free program with other members whenever you make changes to the program.

You are also advised to keep your program version properly and store your backup somewhere safe (i.e., pen drive or external drive).

You should add comments whenever possible describing the purpose of a variable or a block of code. A properly commented program can make the maintenance easy and readable.

Deliverables section

You are to submit the following for this assignment:

1. ONE (1) C++ cpp ile for each algorithm.
The lecture section TC1L your group numbers start from group101.
The lecture section TC2L your group numbers start from group201.
The lecture section TC3L your group numbers start from group301.
For examples
(
group105_num01_kruskalwithoutpq_am_0000006_output.cpp
group105_num02_kruskalwithoutpq_kruskalwithpq_am_input_files.cpp
group105_num03_kruskalwithoutpq_am_all_outputs.cpp
group105_num04_kruskalwithpq_am_all_outputs.cpp
group105_num05_huffmancoding_00000003_output.cpp
group105_num06_huffmancoding_input_files.cpp
group105_num07_huffmancoding_all_outputs.cpp
)
This is the source code for your program. Insert the comment below at the beginning of your source code files:

```
// *******************************************************
// Program: YOUR_FILENAME.cpp
// Course: TCP2101 ALGORITHM DESIGN & ANALYSIS
// Class: TC1L
// Trimester: 2220
// Member_1: ID | NAME | EMAIL | PHONE
// Member_2: ID | NAME | EMAIL | PHONE
// Member_3: ID | NAME | EMAIL | PHONE
// Member_4: ID | NAME | EMAIL | PHONE
// *******************************************************
// Task Distribution
// Member_1:
// Member_2:
// Member_3:
// Member_4:
// *******************************************************
```

Your source code must be able to be compiled without errors and warnings with C++ 2017 compiler, CodeBlocks 20.03 and Windows OS. If your source code cannot be compiled, you get a zero mark.

2. One DOCX report.
Example of filename is group105.docx.
* Title page with group number and the table below:

Num	Student ID	Student Name	Task descriptions	Percentage %
1				
2				
3				
4				
			Total	100%

3. One MP4 video for your group presentation with faces and explanations. Each group will be given a maximum of twenty minutes and each person will be given a maximum of five minutes to present their program outputs from input files. You need to record a video to justify your contribution and work. Those who are not in the video will get a ZERO mark for the whole assignment.
Example of filename is group105.mp4.

4. input txt files, output txt files

5. Zip all the files.
Example of a zipped filename is group105.zip.

Submission section

Submit the SOFTCOPY of your deliverables according to your tutor's instructions.
No late submission.

Video presentation and interview section

Assignment presentation and interview is to be held in week 13 and week 14. One MP4 video for your group presentation with faces and explanations. Each group will be given a maximum of twenty minutes and each person will be given a maximum of five minutes to present their program outputs from input files. Please practice and time your presentation.
You need to attend the presentation and interview to justify your contribution and work. Those who are absent from the presentation and interview will get a ZERO mark for the whole assignment.
To make testing easier and save time during presentation, your program shall have the following features:
* Never clear screen.
* Read input files
* Write output files

Your tutor will announce the arrangement and timeslot for the presentation and interview.

Plagiarism section

It is fine to seek help from friends or from online resources when you do the assignment, However, seeking help should not go overboard, to the point of getting (or even paying) someone to complete the assignment partly or fully for you, copying from online resources without understanding, or doing any means with the intention to cheat. For this assignment, plagiarism means the following:
* Turning in a work that, from the examiner's point of view, you do not sufficiently understand.
* Turning in someone else's work (whether partly or fully) as your own.
* To use another's work (whether partly or fully) without crediting the source.
* Any means of cheating.

Plagiarism is a serious offense.
We will give ZERO (0) marks to students who plagiarize AND to students who intentionally or unintentionally help other students to plagiarize by giving all or some of their code.

Question section

Q01 5m
Kruskal algorithm without priority queue and adjacency matrix graph of n number of vertices implementation for file inputs and file outputs with screen outputs with step-by-step illustration for the minimum spanning tree problem.

Sample run

input filename: kruskalwithoutpq_am_0000006_input.txt	description	output filename: kruskalwithoutpq_am_0000006_output.txt	description

6
0 A
1 B
2 C
3 D
4 E
5 F
i 2 8 i 7 i
2 i 5 7 i i
8 5 i 9 8 i
i 7 9 i i 4
7 i 8 i i 3
i i i 4 3 i
	// num of vertices
	// vertex indexes and vertex names
	
	// adjacency matrix graph, i is infinity
		6
		0 A
		1 B
		2 C
		3 D
		4 E
		5 F
		A B 2
		E F 3
		D F 4
		B C 5
		A E 7
		21
		10s
			// num of vertices
			// vertex indexes, vertex names
			
			// edge vertex pairs, edge weights, in alphabetical order
			
			// total weight
			// example total time taken

Q02 5m
Adjacency matrix complete graphs for Kruskal algorithm of n number of vertices implementation for dataset generation of input files that contain random edge weight integers in each file (10 vertices, 100 vertices, 1000 vertices, 10000 vertices, 100000 vertices, etc.). Write a function to generate all the input files.
The filenames are:
- kruskalwithoutpq_kruskalwithpq_am_00000010_input.txt
- kruskalwithoutpq_kruskalwithpq_am_00000100_input.txt
- kruskalwithoutpq_kruskalwithpq_am_00001000_input.txt
- kruskalwithoutpq_kruskalwithpq_am_00010000_input.txt
- kruskalwithoutpq_kruskalwithpq_am_00100000_input.txt

For all your output files, follow the algorithms and formats given based on the output files.
Your output files generated from the algorithm can process input files of any values.
Screenshot your algorithm time or space for each input size n to provide evidence.
Record the algorithm time or space in the table for each input size n and then plot graphs of the algorithms, recall the first lab.

Q03 5m
Adjacency matrix complete graphs for Kruskal algorithm without priority queue of n number of vertices for input files of different problem sizes that have been generated previously and output files with screen outputs with algorithm times for the minimum spanning tree problem.
Write a function to generate all output files for each input size n.
The filenames are:
- kruskalwithoutpq_am_00000010_output.txt
- kruskalwithoutpq_am_00000100_output.txt
- kruskalwithoutpq_am_00001000_output.txt
- kruskalwithoutpq_am_00010000_output.txt
- kruskalwithoutpq_am_00100000_output.txt

Q04 5m
Adjacency matrix complete graphs for Kruskal algorithm with priority queue of n number of vertices for input files of different problem sizes that have been generated previously and output files with screen outputs with algorithm times for the minimum spanning tree problem.
Write a function to generate all output files for each input size n.
The filenames are:
- kruskalwithpq_am_00000010_output.txt
- kruskalwithpq_am_00000100_output.txt
- kruskalwithpq_am_00001000_output.txt
- kruskalwithpq_am_00010000_output.txt
- kruskalwithpq_am_00100000_output.txt

Q05 5m
Huffman coding of n number of words implementation for file inputs and file outputs with screen outputs with step-by-step illustration for the lossless data compression problem.

Sample run

input filename: huffmancoding_00000003_input.txt	description	output filename: huffmancoding_00000003_output.txt	description

6
A
B
C
G
K
T
CBKTG CACGA
GCTA
	// num of unique char
	// characters, one character is 7-bit
	
	// number of words with spaces
		6
		A 3 00 6
		B 1 1110 4
		C 4 10 8
		G 3 01 6
		K 1 1111 4
		T 2 110 6
		34-bit out of 98-bit
		total space 34%
		10s
			// num of unique char
			// characters, frequencies, code words, num of character bits, in alphabetical order
			
			// num of total bits
			// total space int percentage
			// example total time taken

Q06 5m
Random words for Huffman coding algorithm of n number of words implementation for dataset generation of input files that contain random words in each file (10 words, 100 words, 1000 words, 10000 words, 100000 words, etc.). Write a function to generate all the input files.
The filenames are:
- huffmancoding_00000010_input.txt
- huffmancoding_00000100_input.txt
- huffmancoding_00001000_input.txt
- huffmancoding_00010000_input.txt
- huffmancoding_00100000_input.txt

Q07 5m
Huffman coding algorithm of n number of words for input files of different problem sizes that have been generated previously and output files with screen outputs with algorithm space percentages for the lossless data compression problem. Write a function to generate all output files for each input size n.
The filenames are:
- huffmancoding_00000010_output.txt
- huffmancoding_00000100_output.txt
- huffmancoding_00001000_output.txt
- huffmancoding_00010000_output.txt
- huffmancoding_00100000_output.txt

Q08 Q09 10m
Your report of screenshots, code parts, explanations, input files, output files and step-by-step illustration contains the following.
* Kruskal algorithm without priority queue and Kruskal algorithm with priority queue using adjacency matrix complete graph for the minimum spanning tree
** perform numerous experiments of different input sizes using the algorithms, get the total times for the algorithms
** the above experiment results that can be used to perform a comparative analysis between the two implementations in table form and graph form
** conclude your findings in the report
* Lossless data compression using Huffman coding algorithm
** perform numerous experiments of different input sizes using the algorithm, get the total space percentages for the algorithm
** the above experiment results that can be used to perform a comparative analysis between with data compression and without data compression in table form and graph form
** conclude your findings in the report

Q10 5m
Group video presentation with faces with a maximum of twenty minutes.
Make an appointment for your group interview and meeting with your tutor to validate your work.

To 40m

Tutor total mark 50m	Assignment mark 40m

RUBRIC COMPONENTS

PROGRAM – DESIGN / ARCHITECTURE / INTERFACES / FILES / IMPLEMENTATION

Descriptive Elements	Very Weak (1)	Weak(2)	Average(3)	Good(4)	Excellent(5)
Program implementation results	Program implemented poorly and minimal results were produced.	Program implemented with major defects and some results produced.	Program implemented minor defect and sufficient results produced.	Program implemented quite effectively and results produced quite efficiently	Program implemented effectively and results produced efficiently

Note: 0 = no submission

REPORT

Descriptive Elements	Very Weak (1)	Weak(2)	Average(3)	Good(4)	Excellent(5)
Report contents and quality	Report is poorly written and is difficult to understand	Report is written with limited clarify and limited coherence and require major improvements	Report is written moderately coherence and moderately clear but require minor improvement	Report is written coherently and clearly	Report is written with excellent coherence and excellent clarity

Note: 0 = no submission

INTERVIEW AND PRESENTATION

Descriptive Elements	Very Weak (1)	Weak(2)	Average(3)	Good(4)	Excellent(5)
Presentation, Demonstration, Interview, Explanations	Not able to demonstrate and explain clearly	Able to demonstrate and explain but require major improvements	Able to demonstrate and explain moderately clear and require minor improvement	Able to demonstrate and explain with good clarity	Able to demonstrate and explain with excellent clarity

Note: 0 = no submission