RYERSON UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

CPS 420
MIDTERM 2
WINTER 2019

INSTRUCTIONS

- This exam is 120 minutes long.
- This exam is out of 60 and is worth 15% of the course mark.
- This is a closed book exam. However, one double-sided letter-sized crib sheet is allowed.
- This exam is double-sided and has 8 pages including this front page. The last three pages are blank. Therefore there are 4 pages of questions: pages 2 to 5 inclusive.
- Please answer all questions directly on this exam. If you need extra space to finish answering questions, please do so on pages 6 to 8 and indicate very clearly on the original page of each question on which page the rest of your answer can be found.

PART A – GRAPH THEORY – 20 MARKS

1. Drawing Graphs (10 marks)

In each of the following questions, either draw a graph with the given specifications or explain why no such graph exists.

a) Build a graph with exactly 7 vertices of degrees 1, 2, 3, 4, 5, 6, 7

b) Build a graph with exactly 7 vertices which is connected but is not a tree

c) Build a tree with exactly 7 vertices and 5 edges

d) Build a forest with exactly 7 vertices and 5 edges

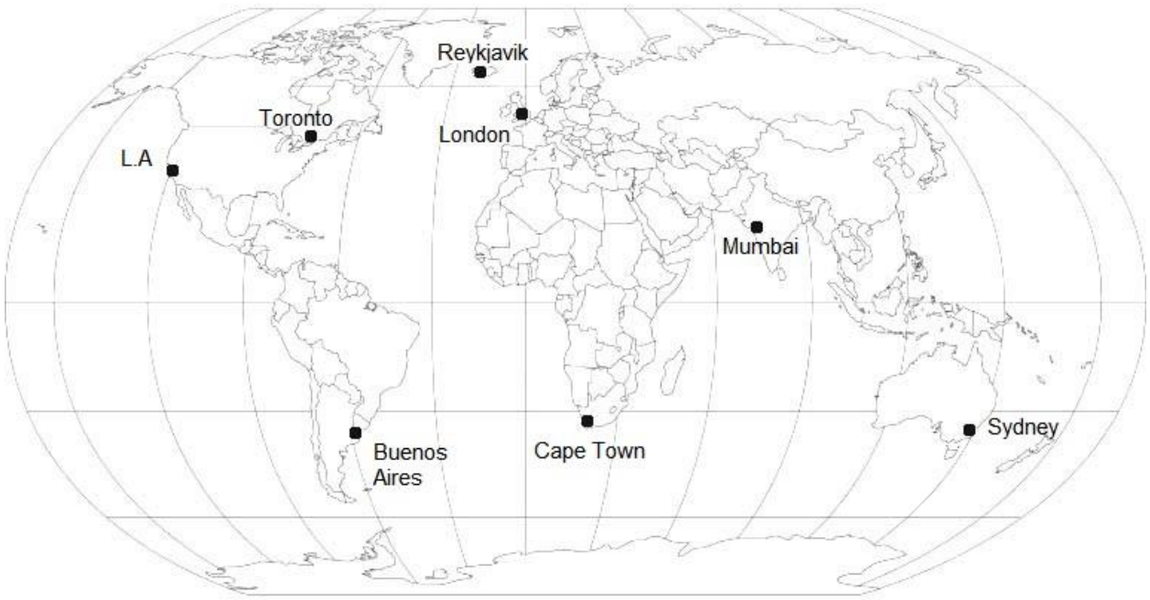
e) Build a forest with exactly 7 vertices which contains a non-trivial circuit

2. Travelling Graphs (10 marks)

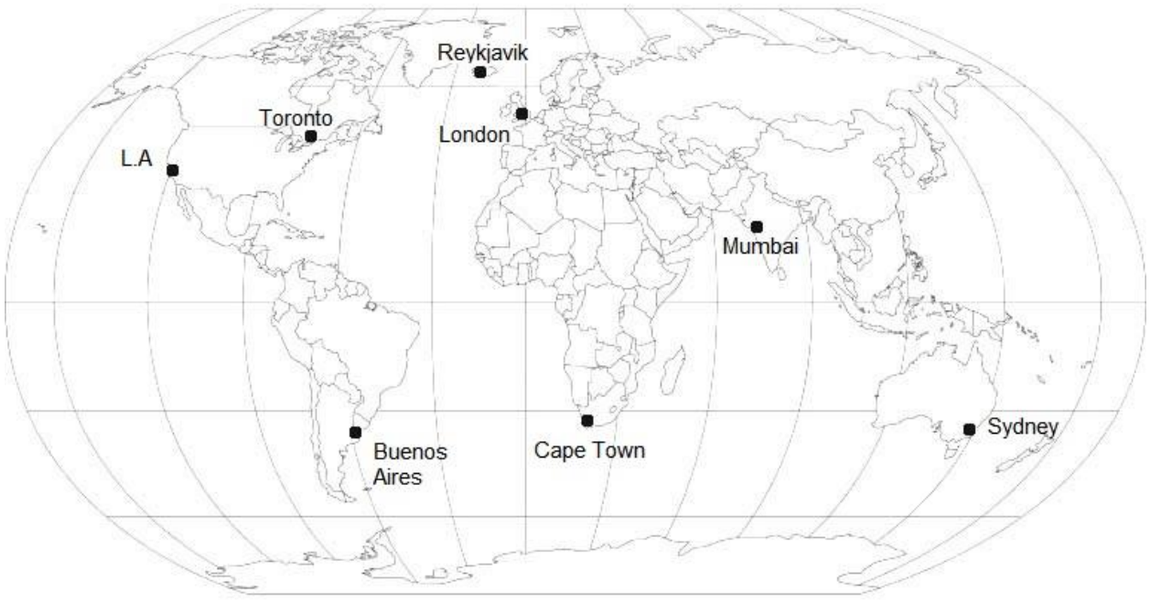
A weighted graph of the cost of travelling between cities is represented by the following cost table. An empty cell means that there is no direct travel between the two cities.

	Toronto	L.A.	Buenos Aires	London	Reykjavik	Cape Town	Mumbai	Sydney
Toronto		600	1000	500	400			
L.A.	600		900					700
Buenos Aires	1000	900				900		
London	500				200	850	600	800
Reykjavik	400			200				900
Cape Town			900	850				850
Mumbai				600				800
Sydney		700		800	900	850	800	

a) Draw an **undirected weighted graph** of the above travel costs into the map below.



b) Draw a **minimum spanning tree** of the graph above into the map below.



PART B – REGULAR EXPRESSIONS AND FINITE STATE AUTOMATA – 40 MARKS

1. Operations on Languages (10 marks)

Let the following two languages L_1 and L_2 over the alphabet $\Sigma=\{a,b\}$ be defined as:
$L_1 = \{a, b, ab, bb\}$ $L_2 = \{\varepsilon, a, b\}$

a) List all the elements of $L_1 \cap L_2$

{ }

b) List all the elements of $L_1 \cup L_2$

{ }

c) List all the elements of $L_1 \times L_2$

{

}

d) List all the elements of $L_1 L_2$

{

}

2. Regular Expression (10 marks)

In this question you will be asked to write a regular expression to match all *polynomials* in a new programming language. In this language *polynomials* are strings like **-3x2+5x4-2x+3** (this string represents $-3x^2 + 5x^4-2x+3$). *Polynomials* also include simpler strings like **1x** or **-5.**

Polynomials are defined as follows:

- A *polynomial* is a sequence of one of more *terms*.
- A *term* consists of a *sign* followed by an *integer* (the coefficient of the term), optionally followed by a *power of x*.
- A *sign* is either the symbol + or **-.** For the first term of the polynomial the sign is optional, but it is compulsory for all the other terms.
- An *integer* is either the digit **0** or a string of one or more digits which does not start with the digit **0**
- A *power of x* is the symbol **x** optionally followed by an *integer* (the degree of the term)

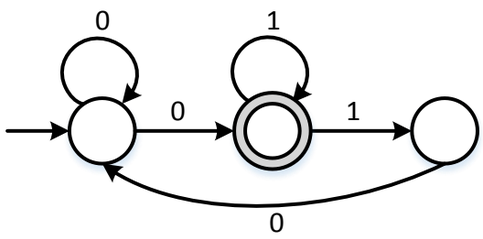
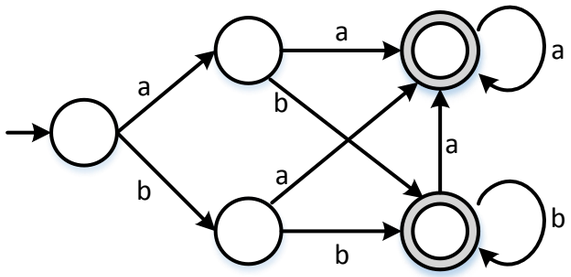
In the two questions that follow, you do **not** need to simplify your regular expressions. You may use the [ ], +, and ? shorthand notations if you wish.

a) Write a regular expression for an *integer* as described above.

b) Assuming that your regular expression for integers in part a) is called **int**, write a regular expression for a polynomial. You can use the name **int** in this regular expression in the place of the regular expression for an integer.

3. Finite State Automata (20 Marks)

a) Give a regular expression for each of the following finite state automata.
Make these regular expressions as simple as possible.

Automaton	Regular expression
	
	

b) In the next two questions the simplest possible automaton refers to an automaton with as few states as possible.

Draw the simplest possible NFA (**non-deterministic** finite state automaton) on an input alphabet I={0,1,2} which recognizes the following regular expression:
$01^{+}(1|2)^{*} \mid (0|1)(1|2)2^{*}$

Draw the simplest possible DFA (**deterministic** finite state automaton) on an input alphabet I={0,1,2} which recognizes the following regular expression:
$01^{+}(1|2)^{*} \mid (0|1)(1|2)2^{*}$.

THIS PAGE IS INTENTIONALLY LEFT BLANK AND CAN BE USED FOR ROUGH WORK OR TO CONTINUE ANSWERING AN EARLIER QUESTION.

WORK ON THIS PAGE WILL ONLY BE GRADED IF SPECIFICALLY REQUESTED ON ONE OF PAGES 2 TO 6.

THIS PAGE IS INTENTIONALLY LEFT BLANK AND CAN BE USED FOR ROUGH WORK OR TO CONTINUE ANSWERING AN EARLIER QUESTION.

WORK ON THIS PAGE WILL ONLY BE GRADED IF SPECIFICALLY REQUESTED ON ONE OF PAGES 2 TO 6.

THIS PAGE IS INTENTIONALLY LEFT BLANK AND CAN BE USED FOR ROUGH WORK OR TO CONTINUE ANSWERING AN EARLIER QUESTION.

WORK ON THIS PAGE WILL ONLY BE GRADED IF SPECIFICALLY REQUESTED ON ONE OF PAGES 2 TO 6.