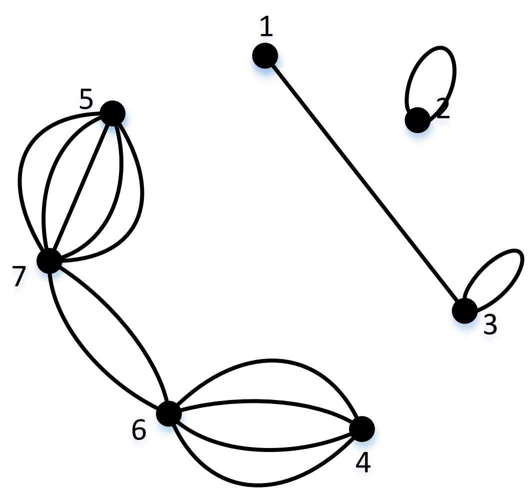
1. <u>Drawing Graphs (10 marks)</u>

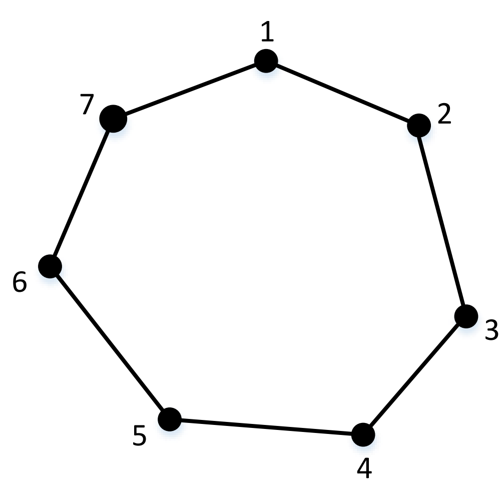
In each of the following questions, either draw a graph with the given specifications or explain why no such graph exists.

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

The total degree of the graph would be 1+2+3+4+5+6+7 = 28, which is even, so it is possible to build many graphs with these degrees. Here is an example.



b) Build a graph with exactly 7 vertices which is connected but is not a tree. There are many possible answers. Here is one:



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

This is not possible: a tree with n vertices has n-1 edges, and therefore a tree with 7 vertices must have 6 edges.

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

This is possible: a forest is a collection of trees, each of which has one fewer edge than vertices. If the forest consists of 2 trees which are not connected, one with n vertices and one with 7-n vertices, then the first tree will have n-1 edges and the second 7-n-1 edges for a total of (n-1)+(7-n-1) = 5 edges.

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

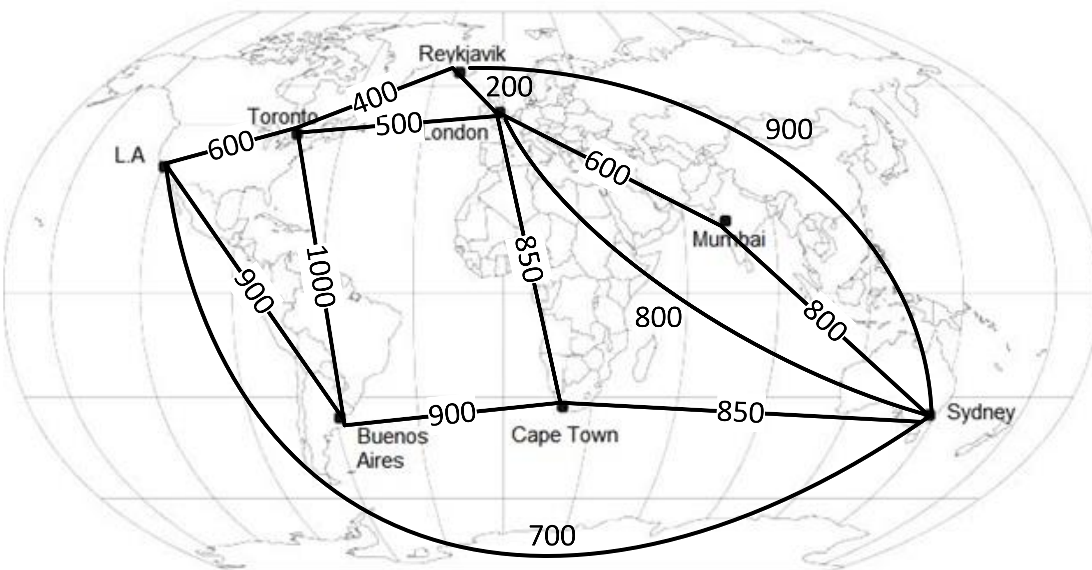
This is not possible: by definition trees do not contain non-trivial circuits.

2. Travelling Graphs (10 marks)

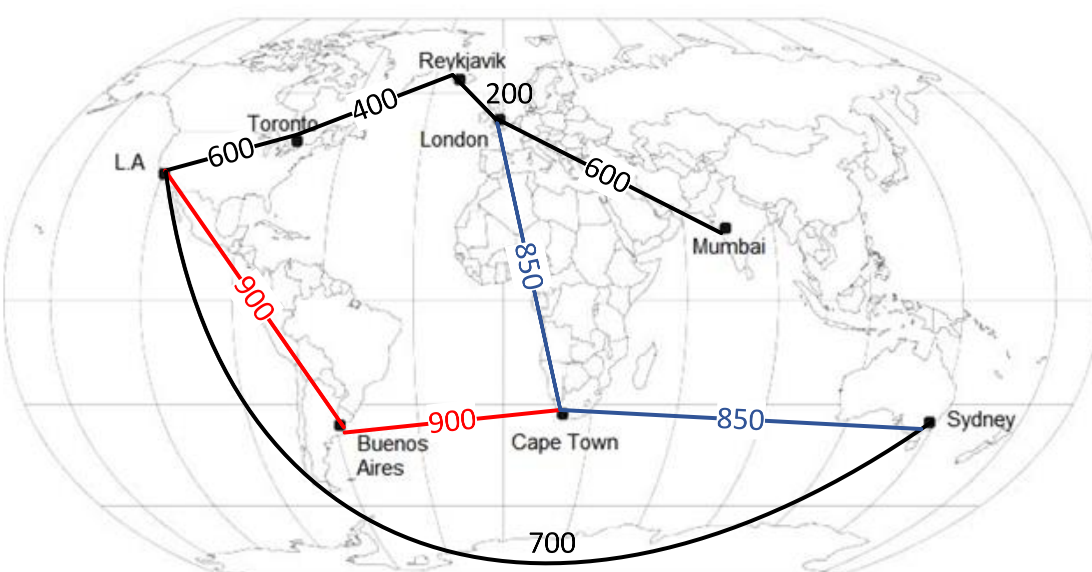
A weighted graph of the cost of travelling between cities is represented by the following cost table. An empty cell means that there is no direct travel between the two cities.

	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.
There are four possible MSTs: the black edges are in all four of them and then each MST includes exactly one of the two red edges and exactly one of the two blue edges.



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 $L_1 \cap L_2$

{ a, b }

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

{ a, b, ab, bb, ε }

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

{ (a,ε), (a, a), (a,b), (b,ε), (b, a), (b,b), (ab,ε), (ab, a), (ab,b), (bb,ε), (bb, a), (bb,b) }

d) List all the elements of $L_1 L_2$

{ a, aa, ab, b, ba, bb, aba, abb, bba, bbb }

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.

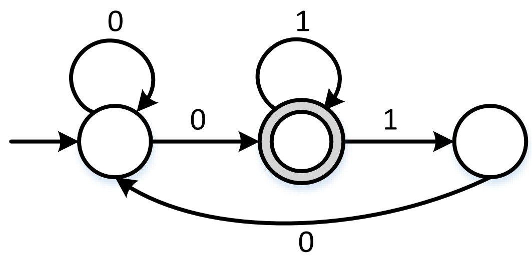
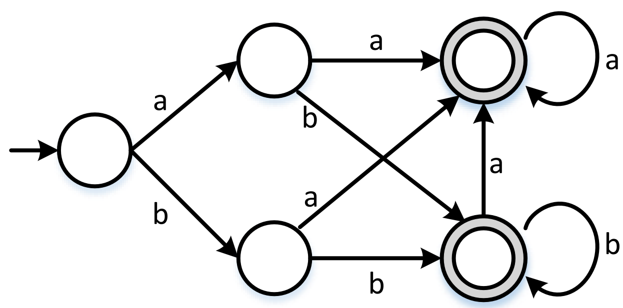
0 | [1-9][0-9]*

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. (The colour coding in the solution has been added to enhance legibility).

(+ | -)? **int** (x **int**?)? ((+|-) **int** (x **int**?)?)*
or without using the ? shorthand, this is:
(ε | + | -) **int** (ε | x (ε | **int**)) ((+| -) **int** (ε | x (ε | **int**)))*

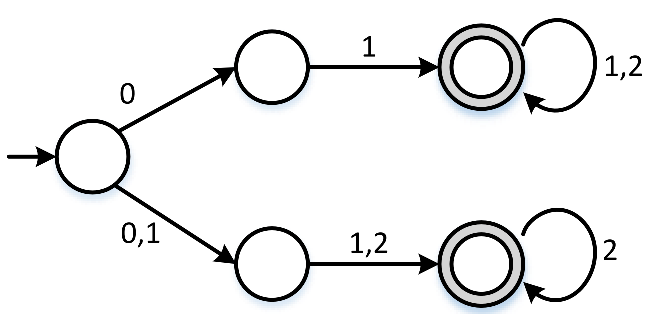
3. Finite State Automata (20 Marks)

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

Automaton	Regular expression
	$0^+ (1 (0^+0)?)^*$
	$(a\|b) (a^+ \| b^+) a^*$

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:
$01^+(1|2)^* \mid (0|1)(1|2)2^*$