Exercise 1 The organiser of an event must choose rooms for 6 seminars that will be held on the same day. He has 4 rooms, numbered 1 to 4. Each seminar is composed of presentation that have already been scheduled. The table on the right shows the times of the presentations for the 6 seminars, noted "a" to "f". No seminar can change room, and, of course, there must not be 2 presentations from 2 different seminars at the same time in the same room.

sem.	9h	10h	11h	12h	13h	14h	15h
a		X	X	X			
b		X	X				
c	X	X	X				
d				X	X	X	
e				X	X		
f					X	X	X

Question 1.1 Write a simple program, using the constraint solver of gprolog, that computes an allocation of rooms for the seminars. In particular, you must define a predicate named solution, such that solution(A, B, C, D, E, F) is true if A, B, ... F are the numbers of the rooms chosen for each seminar.

Question 1.2 Write now a program that finds solutions that minimize the number of rooms that are necessary, and also returns the number of rooms used.

The director of a big hotel has a similar problem: allocate rooms for the customers, depending on their arrival and departure dates. But the dates are not given in a small table like the one above, but in a database that contains hundreds of prolog facts:

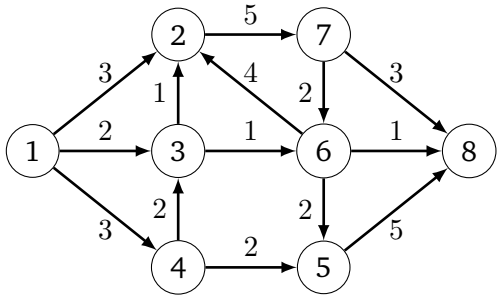
client(N, A, D) indicates that the client named N arrives on day A and leaves on day D.

Each date is a number between 1 et 31 : we consider that the hotel only exists for one month (to make things a little simpler)!

The hotel has 70 rooms, numbered from 1 to 70. Two clients must not share the same room. But a client that arrives on a given day can use a room that has been used by another client who leaves on that day.

Question 1.3 Write a prolog program, again using the constraint solver, that computes an allocation of the rooms to the clients. Try to minimize the number of rooms that are necessary, since, if a room is not used during a month, it saves some cleaning costs.

Exercise 2 A *maximum flow* problem involves finding a maximum feasible flow through a single-source, single-sink flow network like the one opposite. The edges represent possible routes for some goods, they are labelled with the capacity of each route, assumed to be a natural number here. Here, for instance, node (2) can receive 3 units from node (1), 1 unit fro mode (2), and 4 units from node (6); but node (2) can only forward 5 units, to node (7).



The graph can for instance describe pipe-lines between an oil production region and a distribution region. The problem is to find the maximum number of units that can be sent from the source node (1) to the sink node (8).

There are known efficient algorithms to find the maximum flow that can be sent through the network, but the problem can easily represented with constraints.

Question 2.1 Write a simple logic program with constraints that computes the maximum flow from note (1) to (8) on the graph above (you are not asked to write a generic program). In your answer, describe precisely the meaning of the variables that you introduce.