British Informatics Olympiad Final

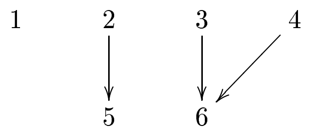
6–8 April, 2001

Sponsored by Data Connection & e-competitions

Widget, Whatsit & Doodah

The family firm of *Widget, Whatsit & Doodah* (est. 1862) have been making all manner of things for well over a hundred years. It is a complicated business and the scheduling of the manufacturing steps requires a lot of work. Mr Widget senior still delivers a good line in after-dinner speeches, recounting tales his grandfather told him about the early days in the company, when the three founding partners worked the production line themselves.

As early pioneers in the field, the production of thingum-a-jigs was painstakingly worked out. The production was broken into steps, each of which took 1 hour to complete (independent of who did the work). Some of the steps required other steps to have been completed first; in these cases the components from the earlier steps got used up, hence no step was a direct requirement for more than one other step. Once any step was begun it could not be interrupted before it was completed. A schedule for the steps was then produced (by Widget grand-senior) that optimised (i.e. minimised) the completion time.



For example, consider the case (pictured above) with 6 tasks. Arrows indicate precedence, so 2 must be completed before 5; 3 and 4 must be completed before 6. Suppose only two people are available. One schedule would be for jobs 1 & 2 to be done during the first hour, 3 & 5 during the second, 4 during the third (6 is not yet ready), finishing with 6 during the fourth. A better schedule (which is optimal) would be for jobs 1 & 2 to be done during the first hour, 3 & 4 during the second and 5 & 6 during the third.

Write a program which, given a list of precedence constraints and the number of available people, outputs an optimal schedule. The first line of input will consist of two numbers, p ($1 \leq p \leq 3$) then j ($1 \leq j \leq 100$), indicating the number of available people and the number of jobs respectively. Each successive line will consist of two integers indicating a precedence order; the first job on a line must be completed before the second job. This list will be terminated with the line "`-1 -1`".

The first line of your output should be the time it takes your optimal schedule to produce a thingum-a-jig. The next j lines should each consist of a single integer; the i^{th} of these lines indicating the finishing time for job i. A solution will always exist for all the given test data.

Sample Input	Sample Output
`2 6`	`3`
`2 5`	`1`
`4 6`	`1`
`3 6`	`2`
`-1 -1`	`2`
	`3`
	`3`