Rede

Uma rede de comutação de pacotes manipula informações em pequenas unidades, quebrando mensagens longas em pacotes antes de realizar o roteamento (envio). Embora os pacotes, talvez, sejam enviados por caminhos diferentes ou cheguem ao seu destino em tempos diferentes ou fora de ordem, o computador que recebe os pacotes remonta a mensagem original corretamente.

O computador que recebe os pacotes usa um *buffer* de memória para manter os pacotes que chegam fora de ordem. Quando cada pacote chega, ele pode ser colocado no *buffer* ou movido diretamente para a área de saída. Todos os pacotes que são mantidos no *buffer* ficam disponíveis para serem movidos para a área de saída a qualquer momento.

É dito que um pacote passou do *buffer* quando ele é movido para a área de saída. É dito que uma mensagem passou do *buffer* quando todos os seus pacotes passaram do *buffer*.

Os pacotes de qualquer mensagem devem ser ordenados para que a passagem dos dados do *buffer* para a área de saída seja de forma ordenada.

Por exemplo, o pacote contendo a parcela [3, 5] da mensagem completa deve passar do *buffer* antes do pacote que contém a parcela [6, 10] da mesma mensagem. Mensagens podem passar pelo *buffer* em qualquer ordem, mas todos os pacotes de uma mesma mensagem devem passar o buffer de forma consecutiva e em ordem.

Você deve escrever um programa que calcula o tamanho mínimo necessário do *buffer* em *bytes* para remontar as mensagens. Cada pacote possui as seguintes informações: o número da mensagem que ele pertence, o *byte* de começo do pacote e o *byte* de fim do pacote. O primeiro *byte* de qualquer mensagem é 1.

Note que para o computador remontar todas as mensagens de forma correta, as mensagens devem passar para área de saída uma por vez.

ENTRADA

A primeira linha da entrada consiste de dois números inteiros **N**, **M** ($1 \leq \mathbf{N} \leq 6$) e ($1 \leq \mathbf{M} \leq 1000$). A segunda linha contém **N** inteiros que são os tamanhos das mensagens em bytes. Cada uma das seguintes **M** linhas descrevem um pacote com três inteiros: o número da mensagem, o byte de começo e de fim do pacote. Nenhum pacote contém mais do que 64 bytes de dados.

SAÍDA

A saída deve conter um número inteiro que é o mínimo tamanho necessário do *buffer* para remontar a mensagem original.

ENTRADA	SAÍDA
3 5 10 20 5 2 16 20 1 6 10 3 1 5 1 1 5 2 1 15	10

ENTRADA	SAÍDA
3 3 5 5 5 1 1 5 2 1 5 3 1 5	0

Explicação para o caso de teste 1:

O primeiro a chegar é o pacote (2, 16, 20), porém, ele é armazenado no *buffer* (pelo fato de ser um pacote que começa em 16 e termina em 20), o segundo é (1, 6, 10) que também é armazenado no *buffer* (pelo mesmo fato do pacote 1), então já temos um buffer de tamanho 10.

O terceiro pacote é (3, 1, 5) que pode ser mandado para área de saída (por começar no byte 1), note que assim que enviamos o pacote (3, 1, 5) a mensagem 3 já é completamente enviada, então podemos enviar pacotes de outras mensagens. O quarto pacote é (1, 1, 5), logo podemos enviá-lo, além disso o pacote (1, 6, 10) ainda está esperando para ser enviado no buffer, então, enviamos, o buffer agora é de tamanho 5.

Na chegada do quinto pacote, (2, 1, 15) já podemos enviá-lo, pois, a mensagem 1 já passou pelo buffer. Após, enviar o quinto pacote, podemos enviar o primeiro que estava esperando no buffer.