IOI Team Training 2013 – Online Test 9, 9–10 June, 2013

Order

Carpenter Sam receives N orders. While reading the orders she realizes that she is missing M machines necessary to complete the orders. Not all orders require all missing machines, but every order requires at least one of them.

To complete an order, Sam needs to either buy or rent each of the machines the order requires. Since different orders need different amounts of work (and thus time) on each machine, the rent for a machine may depend on the order that is completed on it. The purchase cost for a machine does not depend on the orders, though. A machine that is purchased once can be used to work on any number of orders at no extra cost.

If the cost caused by an order seem too high to Sam, she may choose to reject an order; this will lead to no cost (and no profit)

Example

$N = 2$, $M = 3$

Order	Sam's Income if Completed
O_1	100
O_2	100

Machine	Purchase Price
M_1	50
M_2	80
M_3	110

Order	Machine Required by Order	Rent to Complete Order on Machine
O_1	M_1	30
	M_2	20
O_2	M_1	40
	M_3	80

There are two solutions leading to the maximum profit of 50:

- Reject O_2, complete O_1, rent both M_1 and M_2.
- Complete both O_1 and O_2, buy M_1, rent M_2 and M_3

Program and Grader

Implement one function in the file `order.cpp`

```
getProfit(int N, int M, int** rent, int* P, int* C);
```

that returns the maximum profit you can get from the orders assuming that `N` is the number of orders, `M` is the number of machines, `rent` is an `N` × `M` array, where `rent[i][j]` = cost of renting machine `j` to do order `i` (if you do not need machine `j` for order `i`, `rent[i][j]` = 0), `P` is an `N`-element array, where `P[i]` is the profit got by completing order `i`, and `C` is an `M`-element array, where `C[j]` is the cost of buying machine `j`.

Input and Output

The program `grader_order.cpp` takes input in the following format:

- The first line has two integers: N and M.
- This is followed by N blocks of lines, each describing an order.
  - The first line of block has two integers: `P[i]`, and the number of machines that order `i` requires, which we call `m[i]`.
  - The next `m[i]` lines specify two integers `j` and `r`: `j` is the machine needed for the order, and `r` is the rental cost of the machine for that order.
- This is followed by `M` lines containing one integer each: `C[j]`.

`grader_order.cpp` then calls your function `getProfit()` and outputs the integer that your function returns.

Test data

In all test cases, $1 \leq P[i] \leq 5000$, $1 \leq rent[i][j] \leq 20000$ and $1 \leq C[j] \leq 20000$.

- Subtask 1 (20 marks) : $1 \leq N, M \leq 10$
- Subtask 2 (20 marks) : $1 \leq \min(N, M) \leq 16$, $1 \leq \max(N, M) \leq 50$.
- Subtask 2 (60 marks) : $1 \leq N, M \leq 1200$.

Sample input

```
2 3
100 2
1 30
2 20
100 2
1 40
3 80
50
80
110
```

Sample output

```
50
```

Limits

- *Memory limit* : 128 MB
- *Time limit* : 4s