Challenges Week 2

IMPORTANT: If you use sorting for any of these challenges, **you must implement it yourself**. You cannot use built in functions or existing implementations in standard libraries. Even if you pass all the test cases on Themis, if you break this rule, **you will lose points.** You are also allowed (and encouraged) to reuse your own implementations from previous courses.

SPOTS - 45 points

The weather in the Netherlands is not known for its sunny days. So, in anticipation of the sunny days during summer in the Netherlands, you were looking for a way to spend your free time relaxing at some of the best spots in Groningen.

You have downloaded a collection of n recommended spots from the internet. In that collection, each spot has an identifier (integer) as well as an integer that represents the spot's quality.

Because, you want to visit the best places, you will make a program that finds a list from the founded collection with the m best places to relax at.

Input: The first line contains the integers n and m, separated by a space. n is the total number of spots in the collection, and m is the number of spots you want to visit. The next n lines each contain two integers separated by a space, representing the identifier and quality level of a spot. Both the identifier and quality level are unique.
Output: m lines, each containing a spot of top quality levels (the identifier and quality level of the spot). The spots should be sorted by quality level, from highest to lowest.

Constraints:
$0 \leq n \leq 2 \cdot 10^7$
$0 \leq m \leq 5 \cdot 10^4$ and $m \leq n$
$0 \leq$ spot's identifier and quality level $\leq 2^{31} - 1$

Attention: The memory complexity should be $\mathcal{O}(m)$

Example

Input	Output
7 4 5716 3698 8164 4402 4169 6194 3564 2226 7791 3835 1225 4204 5164 298	4169 6194 8164 4402 1225 4204 7791 3835

TOLLGATES - 70 points

Note: This assignment requires knowledge of lecture 4 about dynamic programming.

The TAs are looking forward to their upcoming summer holiday, and have decided to make a road trip with a car to their very local holiday destination. Unfortunately, their road trip crosses a bunch of tollgates. The TAs have decided to be rebels, and speed past the tollgates every so often. Of course, since they feel a sense of guilt they do not want to do this for at least c attempts in a row. Can you help the TAs break the law (with moderation) and get them to their destination for as cheap as possible by skipping the most expensive tollgates?

The road that the TAs will be driving on is split up at each tollgate for your convenience. There are n tollgates on their entire trip, and the cost at each tollgate will be given as a list of integers $p_0, \ldots, p_{n-1}$. Note that these numbers can also be negative: some tollgate booth employees will be happy to see the TAs as they have had the pleasure of following some of their exquisite courses, so they give them money for driving on their road.

During the road trip the TAs have run into a small argument. Some TAs insist on skipping all of the tollgates, while some TAs insist on paying for all of the tollgates. They decide to settle their argument by making a compromise: after skipping h tollgates in a row, they cannot skip *at least* the next c tollgates. Since the greedy TAs are very Dutch, they must always skip h tollgates if they do decide to break the law. Never more, and never less. This also means that if they are approaching the end of their trip, they refuse to skip less than h tollgates, even if those are the last few gates.

For every tollgate where the TAs will stop and pay, the price they have to pay will be summed into s. It is your task to select which tollgates the TAs should skip in order to *minimise* this cost s. Keep the constraints of skipping exactly h tollgates, and not being able to skip c tollgates directly afterwards in mind. Also keep in mind that some tollgates are beneficial to go through, as the TAs will be rewarded. These tollgates have a price of -1.

Note: it is allowed that after skipping h tollgates that you end up with fewer than c tollgates before you reach the end of the road. In other words, for $h = c = 2$ the input `50 50 50` should give you the cost of 50, as you can skip the first two and only pay for the last tollgate.

Figure 1: Note that some of the roads where the TAs will be driving are not so good.

Input: On the first line you are given two integers: the number of tollgates the greedy TAs will skip in a row, h, and the number of tollgates that the lawful TAs will refuse to skip afterwards, c. After this, an input n is given, which denotes the number of tollgates. This is followed by n rows, each row denoting the price p_i for a tollgate i.
Output: Output the total cost s that the TAs will have to pay with the optimal skip/pay configuration, such that the cost s is minimised.
Constraints:
$0 < h < 50$
$0 < c < 50$
$0 \leq n \leq 10^8$
$-10^6 \leq p_i \leq 10^6$ for $0 \leq i < n$

Example

Input	Output

`2 2`
`10`
`5`
`2`
`1`
`3`
`6`
`8`
`20`
`-1`
`9`
`5`
	`23`

Example explanation

The TAs can skip the first two tollgates, and prevent paying `5` and `2` units of currency respectively. After these tollgates, the lawful TAs are back in charge, and they will have to pay for the next two tollgates: `1` and `3`. After they have paid at these tollgates, they could decide to skip the next tollgates again (`6` and `8`) but it is actually cheaper to also pay `6` at the next tollgate, and skip through `8` and `20`. After having skipped `8` and `20` they will have to obey the law for the coming two tollgates, and pay `-1` and `9`. After having done this, they cannot skip the last tollgate (`5`) since this would mean they are skipping fewer than $h = 2$ tollgates, and the naughty TAs would be upset.

In total, the TAs have spent $1 + 3 + 6 + -1 + 9 + 5 = 23$ units of currency, so this should be the output of the optimisation program.

Hint

The optimal solution has a memory complexity of $\mathcal{O}(h + c)$. Only submissions with this memory complexity will be able to pass the test cases 10-13 + valgrind (which has the same input as test case 13).