# Challenges Week 1

**IMPORTANT:** If you use sorting for any of these challenges (Week 1), **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 all points and the submission will be marked as cheating and forwarded to the BoE.** You are also allowed (and encouraged) to reuse your own implementations from previous courses.

## United Kingdom - 25 points

Howdy traveller!
The year is drawing to a close, and it's time to chart a course for your next adventure. After some friendly debate with your chums, you've decided to forgo the flat Dutch landscape (even though it's easy on the wallet) in favour of a journey to the United Kingdom. Despite protests from your mates that it will be too pricey, you've assured them that you can make it happen for less than £50.

You've done some digging on `http://trust-me-bro.com` and discovered a key piece of information: **the more transport options available on a given day, the lower the cost tends to be**. Essentially, if there are a plethora of flights, trains, and lorries headed from Groningen to the Scottish Highlands, the market will drive down the price. With that in mind, your mission is to find the least expensive day to travel by any mode of transportation to the UK. Lucky for you, you have access to a database that includes all flights, ferries, coaches, and Eurotrains heading to your dream destination in the UK. (For the sake of this task, all modes of transportation are equivalent and offer seamless connections to the majestic Scottish Highlands).

The database includes $n$ entries, with each entry listing the start and end dates (separated by a space) during which the transportation option is available. (For the sake of this task, the transportation operates on all days within that range, including the start date but not the end date). The database also specifies the mode of transportation for each entry.

Your objective is to pinpoint the first day on which you can travel to the UK for the best possible price, and list all of the companies offering transportation on that day in alphabetical order, with each company on a new line (including the last line). Begin your output with the phrase `"Best day: "`. An example input and output can be found below.

Figure 1: The highlands that you'll visit!

**Input:** A single integer $n$ representing the total number of transportation ways recorded. The next $n$ lines each contain:

- the start $s_d$ date in `day.month.year` format
- the end $e_d$ date in in `day.month.year` format
- The mean of transportation

The dates and means of transport are space seperated, the mean itself ends on a newline (for example "`Klemper's cruises`" is a valid mean). You can assume all dates are correct, and there is no leap year (so a year always has 365 days).

**Output:** `"Best day: x"`, where x is the day on which the most companies offer transport, followed by all the company names in ascending alphabetical order (a, b...).
**Constraints:**
$0 \leq n \leq 10^7$
$01.01.2000 \leq s_i \leq e_i \leq 31.12.200420$, for all $0 \leq i < n$

## Example

`<EOF>`, is purely to symbolise where the output should end. That is to say, the last character of your output should be a new line.

| Input | Output |
|---|---|
| `4`<br>`01.01.2023 10.01.2023 Plane`<br>`03.01.2023 07.01.2023 Train`<br>`05.01.2023 09.01.2023 Bus`<br>`17.04.2022 02.01.2023 Eurostar`<br>`<EOF>` | `Best day: 05.01.2023`<br>`Bus`<br>`Plane`<br>`Train`<br>`<EOF>` |

## Programming Hints

- `scanf("%[^\n]", ...)` can be used to read a string until newline
- `%*c` in `scanf`, can be used to read a character and throw it away (anything `*`)
- `printf("%02d", ...)`, can be used to force the number to be made out of two digits (1 would be 01)
- Comparator functions can simplify your sorting algorithms and make them reusable, `https://stackoverflow.com/q/9410`, is a very good resource on this

# Andorra - 45 points

You have been hired by the government of Andorra to help them address a severe drought that is affecting their water supply. As an expert in logistics and transportation, they have asked for your help in transporting water to Andorra from a neighboring country.

While on your holidays in the UK, you have agreed to assist the government of Andorra remotely.

While Andorra is experiencing a severe drought, it's raining heavily in the UK, and they have an abundance of water. Therefore, the UK has agreed to ship water to Andorra to help them address their crisis.

To transport the water, the government of Andorra has hired several transport companies, each with **its own truck**. Each truck can carry $c$ units of water and can make the trip in $t$ days. When the truck arrived it will travel back without any time cost (immediately). Various companies can of course send trucks at the same time.

The Minister of Environment has provided you with the following information and asked you to determine the minimum number of days required to transport at least $D$ units of water to Andorra.

As a thank you for your assistance, the government of Andorra has agreed to pay for an all-expenses vacation for you in Barcelona next week.

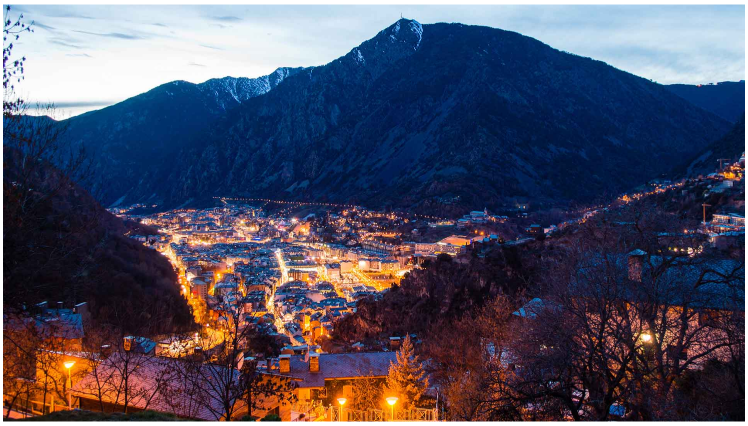

Figure 2: An overview over that tiny nation in the middle of the Pyrenees

**Input:** An integer $D$ representing the number of units of water Andorra needs, an integer $n$ representing the number of transport companies they have hired. Next are $n$ integer pairs $c_i$ and $t_i$ representing the number of units of water $c_i$ a truck from company $i$ can carry and the number of days $t_i$ it takes for a truck from company $i$ to make the trip.

**Output:** The minimum number of days required to transport at least $D$ units of water to Andorra.

**Constraints:**
$0 \leq D \leq 2^{63}$
$0 < n \leq 10^7$
$0 \leq c_i \leq 2^{20}$ for all $0 \leq i < n$
$0 < t_i \leq 2^{20}$ for all $0 \leq i < n$

## Example

| Input | Output |
|---|---|
| `10`<br>`3`<br>`13 20`<br>`3 7`<br>`5 6`<br>`<EOF>` | `12`<br>`<EOF>` |

## Programming Hints

- If you add two `uint64_t` together, the result might overflow so you should use the datatype `__int128` to store intermediary values.

## Changelog

1. 19.04.2023 - 1:42pm
   - Clarified Andorra text, to disallow solution of infinite trucks being sent out at day 0.
   - Added further explanation in text of Andorra
   - Typo fixes in Constraints of Andorra