Lab4: Greedy Algorithms
Deadline: April 22

INSTRUCTIONS:

1. **Submit in LMS a zip file containing the following:**
    a. python code files (problem1.py, problem2.py)
    b. lab report

    SUBMISSION DETAILS
    1. **The implemented program code should be written in python**. The program should match the pseudocode that is provided in item #2.
    2. **A short report in pdf format.** Provide a short explanation of your code/solution. Include references, if any.
    3. If AI was used, disclose the parts of your code where AI was used. List down the prompts.
    4. Please do not submit code that you copied from another group or from a source you found online (and slightly modified). Those who submit copied codes will be automatically given a score of zero, and the source of the copied codes will also be given deductions.

Problem 1:
Professor Fluffy has now become conscious of his body ever since body-conscious after the girl of his dreams crushed his heart when she told him *"Ayoko sayo, ang taba mo kasi. Workout-workout din pag may time."* (Disclaimer: I'm not promoting fat-shaming). On that day, he vowed to workout daily in the school gym. Unfortunately, he has a very busy daily schedule, so he rarely has the time to workout. He wants to workout once a day. It is also the school gym's policy to allow employees to use the gym only once daily. Naturally, Professor Fluffy wants to take the longest possible time to use the gym for exercise, that will fit his schedule. He is in the campus from 10 AM to 6 PM only, and he wants to workout within that period. Help Professor Fluffy achieve his body goals by creating a program that will determine the longest possible time he can use for working out, given his daily schedule.

INPUT
- You will be reading input from the command line.
- First input is the number of test cases / days he wants to work out, say N.
- This is followed by N chunks of data, for each day, in this format:
    - Next input contains the number of appointments Professor Fluffy has for that day, say A.
    - This is followed by A lines, each containing the appointment he has for that day.
    - Each appointment data has 2 numbers (start_time, end_time) separated by space.
- Start and end times follow the 24-hour notation, but in float. (e.g. 1:45PM = 13.75)
- You are assured that the start time will be ≥ 10 (10 AM) and the end time will be ≤ 18 (6PM)

OUTPUT
- For each day, print the start time and end time of the longest possible workout time on that day.
- Both start and end times should be in float, with 2 decimal places.
- Separate the start and end times by a single space.

EXAMPLE INPUT
1
4
10 12
12 13
13 15
15.5 17.75
EXPECTED OUTPUT
15.00 15.50
EXPLANATION
Professor Fluffy can only workout from 15.00 - 15.50 and 17.75 - 18.00, since he has appointments in all the other time intervals. The maximum time interval among the choices is 15.00 - 15.50 (30 minutes).

PROBLEM 2: Slowie the Shoemaker

Mr. Slowie, a shoemaker, has a lot of orders from customers. He can only work on one order per day, due to old age, and some orders usually take several days to finish. But, because of the quality of his work, Mr. Slowie is quite popular and has many customers. Mr. Slowie also has a policy about delays: he agrees to deduct a certain amount from the total price for each delayed day. This daily penalty amount is agreed upon by Mr. Slowie and the customer beforehand.

Help Mr. Slowie by writing a program that will find the optimal sequence of job orders to follow, such that the total penalty is minimized. If there are ties in the sequence, the job that appeared first should be prioritized.

INPUT
- You will be reading input from the command line.
- The first input is the number of problems to be solved, say N.
- This is followed by N chunks of data, formatted as follows:
    - Next input contains the number of job orders for Mr. Slowie, say J
    - This is then followed by J lines of data, one for each job order, formatted this way: *name_of_customer number_of_days_to_complete daily_penalty*
    - e.g. : A 3 4 = customer A's order takes 3 days to complete, with daily penalty of P4 per late day
    - The order in which the job order appears in the input will be used as tie-breaker.

OUTPUT
- For each problem, print in one line the minimum total penalty and the sequence of job orders (customer names only), that minimizes the total penalty, separated by a single space.
- The names of the customers in the job order sequence will be separated by a single dash (-).

EXAMPLE INPUT
1
4
A 3 4
B 1 1000
C 2 2
D 5 5
EXPECTED OUTPUT
42 B-A-C-D

EXPLANATION:

Day	1	2	3	4	5	6	7	8	9	10	11
Job	B	A	A	A	C	C	D	D	D	D	D

Job	Expected Finish	Actual Finish	Late Days	Daily Penalty	Penalty
A	Day 3	Day 4	1	P4	P4
B	Day 1	Day 1	0	P1000	P0
C	Day 2	Day 6	4	P2	P8
D	Day 5	Day 11	6	P5	P30
Total					P42

Note: The sequence B A D C also has a total penalty of P42 (A = P4, B = P0, C = P18, D = P20), but because of the tie-breaker, we will prioritize job C over job D.