

Piano degli studi (pianostudi [74 punti])

Questo problema è preso dalle OIS (Olimpiadi Italiane a Squadre).

Gabriele vuole laurearsi il prima possibile! Ogni anno deve compilare il proprio “piano degli studi”, dove indica i corsi che intende frequentare in quell’anno accademico. Ogni corso, oltre alla data di inizio e fine lezioni, ha associato anche un numero (intero) di crediti. Quest’anno Gabriele intende raccogliere il maggior numero di crediti possibile ma, per non stressarsi troppo, stabilisce che tra i corsi che sceglierà non dovranno esserci sovrapposizioni, ovvero non dovrà succedere che i periodi di erogazione dei corsi abbiano intersezione, neanche eventualmente per un solo giorno.

Goals: con quanta efficienza sai rispondere alle seguenti domande?

[**opt_val**]: quale è il massimo numero di crediti?

[**opt_sol**]: quale è una scelta ottima di corsi su cui puntare?

Nota: proporranno delle istanze in cui tutti i corsi offrono lo stesso numero di crediti. In tali casi i goal **opt_val** e **opt_sol** si riduce alla competenza di individuare un massimo numero di corsi senza sovrapposizioni.

Input

Si legga l’input da `stdin`. La prima riga contiene T , il numero di testcase (istanze) da risolvere. Seguono T istanze del problema, ciascuna descritta nel seguente formato: ad una prima riga contenente il numero n di corsi disponibili seguono n righe, una per corso. L’ i -esima di queste righe contiene tre numeri interi separati da spazio (nell’ordine: giorno di inizio, giorno di fine, e numero di crediti del corso i -esimo).

Output

Per ciascuna istanza, prima di leggere l’istanza successiva, scrivi su `stdout` il tuo output così strutturato:

[**goal 1 (opt_val)**]: la prima riga contiene il numero massimo di crediti acquisibile quest’anno.

[**goal 2 (opt_sol)**]: la seconda riga contiene i nomi dei corsi che si è scelto di frequentare, ossia dei numeri interi tra 1 e n , separati da spazi.

Assunzioni

1. $1 \leq n \leq 50,000$
2. $1 \leq \text{crediti}[i] \leq 10,000$
3. $1 \leq \text{inizio}[i], \text{fine}[i] \leq 500,000,000$

Esempio di Input/Output

<pre><start in> 3 2 5 8 1 1 5 2 <more></pre>	<pre>4 3 9 30 2 4 10 5 6 15 1 1 0 <more></pre>	<pre>5 3 3 7 4 9 8 1 2 1 7 9 3 9 10 1 <end></pre>	<pre><start out> 2 2 30 1 16 3 1 2 <end></pre>
--	--	---	--

Spiegazione: Nel primo caso di esempio i due corsi si sovrappongono nel giorno 5. Non essendo possibile frequentarli entrambi, conviene scegliere il secondo corso, per un valore totale di 2 crediti. Nel secondo caso di esempio una scelta ottima è frequentare solo il corso da 30 crediti; una seconda scelta ottima sarebbe frequentare i corsi 1 e 4. Nel terzo caso di esempio la soluzione ottima è nuovamente unica.

Subtask

Il tempo limite per istanza (ossia per ciascun testcase) è sempre di 1 secondo.

I testcase sono raggruppati nei seguenti subtask.

1. [0 pts ← 3 istanze da 0 + 0 punti] **esempi_testo:** i tre esempi del testo
2. [12 pts ← 6 istanze da 1 + 1 punti] **small:** $n \leq 10$
3. [12 pts ← 6 istanze da 1 + 1 punti] **medium:** $n \leq 100$
4. [10 pts ← 5 istanze da 1 + 1 punti] **big_same_value:** $n \leq 1000$, ogni corso vale 6 crediti
5. [20 pts ← 10 istanze da 1 + 1 punti] **big:** $n \leq 1000$
6. [20 pts ← 10 istanze da 1 + 1 punti] **large:** $n \leq 50,000$

In generale, quando si richiede la valutazione di un subtask vengono valutati anche i subtask che li precedono, ma si evita di avventurarsi in subtask successivi fuori dalla portata del tuo programma che potrebbe andare in crash o comportare tempi lunghi per ottenere la valutazione completa della sottomissione. Ad esempio, chiamando^{1, 2}:

```
rtal -s <URL> connect -a size=medium pianostudi -- <MY_SOLUTION>
```

vengono valutati, nell'ordine, i subtask:

esempi_testo, small, medium.

Il valore di default per l'argomento size è large che include tutti i testcase.

Nel template di riga di comando qui sopra <MY_SOLUTION> è una qualsiasi scrittura che ove immessa anche da sola al prompt della CLI comporti l'avvio del solver da tè realizzato. Solo alcuni esempi:

- ./a.out per un compilato da C/C++, eventualmente seguito dagli argomenti che prevede
- ./my_solution.py arg1 arg2 ... se il tuo file my_solution.py col codice python ha i permessi di esecuzione e inizia con la riga di shebang
- python my_solution.py o python3 my_solution.py per far eseguire il tuo script da un interprete python

Se vuoi che una tua sottomissione venga conteggiata ai fini di un esame o homework devi allegare il sorgente della tua soluzione con -fsource=<FILENAME> (ad esempio -fsource=my_solution.py subito

¹<URL> server esame: wss://ta.di.univr.it/esame

²<URL> server esercitazioni e simula-prove: wss://ta.di.univr.it/algo2025

a valle del comando `connect`. Inoltre devi aver precedentemente effettuato il login tramite credenziali GIA (lancia prima `rtal -s <URL> login` e segui le istruzioni per impostare il sign-on).

Il comando `rtal` prevede diversi parametri, consigliamo di esplorarne e sperimentarne le potenzialità ed opzioni d'uso. Un tutorial all'uso di `rtal` è esposto alla pagina:

<https://github.com/romeorizzi/AlgoritmiUnivr/tree/main/strumenti>