Il posto fisso

Sei stato assunto come unico impiegato in un piccolo ufficio postale del tuo quartiere.
Il lavoro non ti piace molto e lo fai controvoglia. Tuttavia, come dice tua madre, è un **posto fisso**, perfetto per te che, come lei sottolinea sempre, sei una persona pigra.

Descrizione del problema

Ogni giorno devi servire un certo numero di clienti, diciamo n.
Poiché l'ufficio è piccolo e tu sei l'unico impiegato, i clienti si prenotano in anticipo tramite un'app, così puoi conoscere in anticipo l'orario di arrivo di ciascuno.

- Il cliente i-esimo arriva all'istante di tempo t_i.
- I clienti, se trovano lo sportello occupato, fanno la coda ed aspettano il proprio turno.

Le pratiche da svolgere sono tutte uguali. Se lavorassi alla massima velocità, ogni pratica richiederebbe Δ minuti.

Tuttavia, tu sei pigro e vuoi lavorare il più lentamente possibile. Puoi decidere, all'inizio della giornata, di lavorare ad una velocità ridotta, impiegando $\Delta' \geq \Delta$ minuti per completare **ognuna** delle pratiche della giornata.

Vincolo di tempo

Hai però una scadenza: l'ufficio chiude a tempo M.
Se non completi tutte le pratiche entro il tempo M, verrai licenziato (e tua madre ne soffrirebbe troppo).

Obiettivo

Determinare il **massimo valore Δ'** ammissibile tale che puoi servire tutti i clienti senza sforare il tempo di chiusura M.

Input

- n: numero di clienti
- M: tempo di chiusura dell'ufficio
- Δ: il tempo minimo (massima velocità) per completare una pratica
- $t_1, t_2, \ldots, t_n$: gli istanti di arrivo dei clienti, con $t_1 \leq t_2 \leq \cdots \leq t_n$.

Output

- Il massimo Δ' (intero) tale che servendo ogni cliente in Δ' minuti si finisce tutto entro M.

Soluzione (idea)

1. **Test di fattibilità per un dato Δ'**:
   - Si simula la giornata:
     - Ordinare i clienti in base al tempo di arrivo.
     - Mantenere un tempo corrente $T = 0$.
     - Per ogni cliente i:
       - Iniziare a servirlo non prima del suo arrivo: $\text{inizio}_i = \max(T, t_i)$.
       - Servire il cliente richiede Δ' minuti: $T = \text{inizio}_i + \Delta'$.
   - Alla fine, se $T \leq M$, allora Δ' è fattibile; altrimenti no.
2. **Ricerca del massimo Δ'**:
   - Poiché se Δ' è fattibile, lo sono tutti i valori inferiori, e se Δ' non è fattibile, nessun valore superiore lo sarà, possiamo usare la **ricerca binaria**:
     - Impostare `low = Δ` e `high = M`.
     - Finché `low <= high`:

        - `mid = (low + high) // 2`
        - Se `mid` è fattibile, `low = mid + 1`.
        - Altrimenti, `high = mid - 1`.
    - Al termine, `high` conterrà il massimo Δ' fattibile.

Complessità

- Ordinamento: $O(n \log n)$
- Test di fattibilità: $O(n)$
- Ricerca binaria su Δ' in $[\Delta, M]$: $O(\log M)$ test

Complessità totale: $O(n \log n + n \log M) \approx O(n \log M)$.

Correttezza

La correttezza del metodo proposto è garantita dalla **monotonicità** del problema:

- Se un valore Δ' è fattibile (cioè consente di completare tutte le pratiche entro M), allora qualunque valore $\Delta'' < \Delta'$ sarà certamente fattibile, poiché un minore tempo di servizio per cliente non può che ridurre il tempo totale.
- Se un valore Δ' non è fattibile, allora nessun valore $\Delta'' > \Delta'$ potrà esserlo, poiché un tempo di servizio più lungo per ogni pratica non farà che aumentare il tempo totale necessario.

Questa struttura monotona dei valori fattibili di Δ' permette di applicare la **ricerca binaria** sullo spazio dei possibili valori di Δ'. La ricerca binaria, partendo da un intervallo $[\Delta, M]$, restringe sistematicamente lo spazio delle possibili soluzioni fino ad individuare il valore massimo di Δ' che soddisfa il vincolo di chiusura.

In altre parole, la correttezza dell'algoritmo discende direttamente dalla proprietà monotona del problema: il test di fattibilità determina in modo chiaro, per ogni Δ', se si può completare entro M o meno, e la ricerca binaria sfrutta questa risposta per convergere sulla soluzione ottimale.

Ottimalità

Il metodo proposto sfrutta la monotonicità del problema:

- Se per Δ' è possibile completare le pratiche in tempo, allora per qualsiasi $\Delta'' < \Delta'$ è sicuramente possibile.
- Viceversa, se non è possibile completare in tempo per Δ', lo stesso vale per tutti i $\Delta'' > \Delta'$.

Questa proprietà monotona permette di utilizzare efficientemente la **ricerca binaria**, riducendo notevolmente il costo rispetto alla ricerca lineare. Inoltre, la verifica della fattibilità in $O(n)$ è necessaria perché si deve considerare la sequenza dei clienti e i loro tempi di arrivo. Il ricorso alla ricerca binaria limita a $O(\log M)$ il numero di tali verifiche, ottenendo così una soluzione asintoticamente ottimale sotto le condizioni date.
Non esiste un metodo più rapido che verifichi la fattibilità senza considerare i singoli clienti, quindi la combinazione di verifica lineare e ricerca binaria produce una soluzione bilanciata tra semplicità e velocità.