Esercizio 1: Bandwidth Allocation (Allocazione Larghezza di Banda)

Dato un insieme di richieste di trasmissione $R = \{r_1, r_2, ..., r_n\}$, dove ogni richiesta r_i è caratterizzata da:

- tempo di inizio s_i
- tempo di fine f_i
- larghezza di banda richiesta $b_i \in \{1, 2, 3\}$

La capacità totale del canale è $B = 3$ unità. L'obiettivo è selezionare il massimo numero di richieste compatibili tali che in ogni istante temporale la somma delle larghezze di banda non superi B.

Parte A: Progettare un algoritmo greedy e dimostrarne la correttezza tramite le proprietà di scelta greedy e sottostruttura ottima.

Parte B: Fornire un controesempio che mostri come l'algoritmo greedy che seleziona per "densità di larghezza di banda" ($b_i/(f_i-s_i)$) non sia sempre ottimo.

Esercizio 2: Minimum Checkpoint Coverage

Sia $G = (V, E)$ un grafo orientato aciclico che rappresenta una rete di strade. Ogni arco $(u,v) \in E$ ha un peso $w(u,v)$ che rappresenta la lunghezza della strada.

Un **checkpoint** è un nodo che monitora tutti i cammini che lo attraversano. L'obiettivo è trovare il numero minimo di checkpoint tale che ogni cammino semplice da s a t passi attraverso almeno un checkpoint.

Vincolo: La distanza tra due checkpoint consecutivi su qualsiasi cammino deve essere $\leq D$.

Parte A: Formulare il problema come greedy e fornire pseudocodice.

Parte B: Dimostrare la proprietà di scelta greedy utilizzando l'exchange argument.

Esercizio 3: Task Scheduling con Penali Variabili

Dato un insieme di task $T = \{t_1, t_2, ..., t_n\}$ dove ogni task t_i ha:

- durata d_i

- deadline D_i
- penale $p_i(x) = \alpha_i \cdot x^2$ se completato con ritardo $x > 0$

L'obiettivo è trovare una schedulazione che minimizza la somma totale delle penali.

Domanda: La scelta greedy "ordina per D_i/α_i crescente" produce sempre la soluzione ottima? Dimostrarlo o fornire controesempio.

Esercizio 4: Multi-Level Caching

Un sistema ha una gerarchia di cache a 3 livelli:

- L1: capacità C_1, costo accesso c_1
- L2: capacità C_2, costo accesso c_2
- L3: capacità C_3, costo accesso c_3

Dove $c_1 < c_2 < c_3$ e $C_1 < C_2 < C_3$.

Dato un insieme di oggetti $O = \{o_1, o_2, ..., o_n\}$ con frequenze di accesso $f_1, f_2, ..., f_n$, determinare l'allocazione ottima negli k livelli di cache per minimizzare il costo totale di accesso.

Vincolo: Ogni oggetto può essere memorizzato in al più un livello.

Richiesta:

1. Progettare algoritmo greedy O(n log n)
2. Dimostrare ottimalità con sottostruttura e scelta greedy
3. Analizzare caso particolare quando tutti gli oggetti hanno la stessa dimensione

---

Note per le Dimostrazioni

Per ogni esercizio, ricordare di seguire il paradigma generale greedy:

Proprietà di Scelta Greedy: Dimostrare che esiste sempre una soluzione ottima che contiene la scelta greedy, spesso utilizzando il **cut&paste** (exchange argument).

Sottostruttura Ottima: Dopo aver fatto la scelta greedy, il sottoproblema rimanente deve avere struttura ottima, preferibilmente con spazio dei sottoproblemi **lineare** anziché quadratico.

Template dimostrazione Cut&Paste:

1. Sia OPT una soluzione ottima qualsiasi
2. Se OPT contiene già la scelta greedy → finito
3. Altrimenti, sostituire la scelta di OPT con quella greedy
4. Dimostrare che la nuova soluzione è ammissibile e ha stesso costo
5. Concludere che esiste una soluzione ottima con la scelta greedy