3 Problemi su Vettori

3.1 Comparatore binario (I esonero, 25/3/22)

Indichiamo con $\bar{a}$ l'intero rappresentato da un vettore a di k di cifre binarie, con la cifra più significativa in posizione $k-1$.

1. Scrivere una funzione:

```
u, m = confronta(a, b, k)
```

che confronta due vettori di k cifre binarie a e b, e ritorna 1, # se $\bar{a} = \bar{b}$, 0, 1 se $\bar{a} > \bar{b}$, e 0, 0 se $\bar{a} < \bar{b}$.

2. Valutare la complessità, indicando il caso *ottimo* e *pessimo*.

3. ★ Valutare la complessità del caso *medio*, assumendo le sequenze siano distribuite uniformemente (se uscisse una sommatoria, basta impostarla).

3.2 Sequenza quasi crescente (I esonero, 25/3/22)

Una segmento di un vettore $v[inf, sup)$ è *quasi crescente* se esiste al più un indice $j \in (inf, sup)$ tale che $v[j] < v[j-1]$.

Scrivere una funzione `k, l = maxSeqCresc(v)` che restituisce la più lunga sequenza quasi crescente di v, tornando l'indice di inizio k e la sua lunghezza l.

3.3 Segmento di somma s (I esonero, 17/10/22)

Dare un algoritmo che dato un vettore v di interi e un numero intero s verifica se esiste un segmento $v[inf, sup)$ di v tale che $\mathbf{S}(v[inf, sup)) = s$ e in tal caso ritorna True, inf, sup mentre torna False, #, # altrimenti. Calcolare la complessità del vostro algoritmo.

Ricordiamo che $\mathbf{S}(v[inf, sup)) = \sum_{i=inf}^{i<sup} v[i]$ e che convenzionalmente i segmenti vuoti hanno somma nulla, quindi $\forall i.\mathbf{S}(v[i, i)) = 0$.

3.4 Segmento di somma s, caso positivo (I esonero, 17/10/22)

Dare un algoritmo che risolve il problema 3.3 sotto la precondizione che il vettore v contenga solo valori *strettamente positivi*. Rispetto al caso generale, è possibile:

1. semplicemente migliorare l'efficienza, senza migliorare la complessità asintotica del caso pessimo: fornire esempi di casi in cui si guadagna molto e di casi in cui non si guadagna niente rispetto all'algoritmo del punto precedente; illustrare con esempi, casi ottimo, medio e pessimo.

2. **oppure** ★dare un algoritmo di complessità $\Theta(n)$ argomentando la correttezza (anche senza una vera e propria dimostrazione), aiutandosi con asserzioni logiche (precondizioni, invarianti, etc.).

3.5 Baricentro (I esonero, 2012)

Diciamo che un indice b di un vettore di interi v lungo n ($0 \leq b < n$) è il *baricentro* di v, se $\mathbf{S}(v[0,b)) = \mathbf{S}(v[b,n))$, ossia se la somma degli elementi di v prima di b è uguale alla somma degli elementi a cominciare da b fino ad n.

1. Si scriva una funzione `b = baricentro(v)` che restituisce in b un eventuale baricentro di v oppure -1 se non esiste nessun baricentro in v. Si valuti la complessità della soluzione.

2. Sotto la precondizione che gli elementi del vettore siano tutti positivi, è possibile migliorare l'efficienza della funzione (anche senza migliorare la complessità asintotica del caso pessimo)? Motivare la risposta (eventualmente con il codice di una funzione `baricentroPos` che sfrutta questa precondizione).

3. ★ Scrivere una funzione ricorsiva che risolve lo stesso problema con un'*unica* scansione del vettore (SUGG: scrivere una funzione ausiliaria con parametri e/o valori di ritorno aggiuntivi, ricordando che la scansione ricorsiva, di fatto, percorre il vettore 2 volte, una all'*andata* e una al *ritorno* dalle chiamate ricorsive).

3.6 ★ Sottoinsiemi di somma s (I esonero, 17/10/22)

1. Dato un vettore v di interi e un intero s, scrivere un algoritmo che verifica se esiste un *qualsiasi insieme di indici* di v di somma s (non necessariamente un segmento), cioè risponde TRUE se esiste un insieme $I \subseteq [0, len(v))$ tale che $\sum_{i \in I} v[i] = s$. Dare la complessità.

2. ★★ Rispetto al punto precedente, dare un algoritmo che restituisce in caso affermativo anche l'insieme di indici I. A tal fine, proporre un modo di codificare l'insieme di indici I con un vettore e dare un algoritmo che torna la coppia n, u dove n è la cardinalità di I (-1 se non esiste un tale insieme di indici) e u è un vettore che codifica opportunamente I.

3.7 Segmento di somma massima♣

Considerare il problema di trovare il segmento di vettore di somma massima in un vettore v di numeri. Il problema è interessante solo se v contiene sia numeri positivi che negativi (se contenesse solo positivi, il segmento di somma massima è v stesso, mentre se contenesse solo negativi, sarebbe il segmento vuoto di somma 0).

1. Supponete di avere a disposizione un esecutore la cui unica capacità aritmetica sia una funzione `sumV(v, inf, sup)` che calcola $\mathbf{S}(v[inf, sup))$. Avete ovviamente a disposizione gli usuali operatori di confronto (`<`, `<=` etc.). Scrivere una funzione `s, b, e = svSommaMax(int v)` che dato il vettore a di interi di lunghezza n, restituisce la somma del sotto-vettore di somma massima in `s` e carica nelle variabili `b` ed `e` gli indici di dove comincia e finisce tale sotto-vettore.

Supponendo `sumV` abbia complessità $\Theta(sup - inf)$, dare la complessità della funzione `svSommaMax`, giustificandola (anche con un argomento informale).

2. Avendo a disposizione la libertà di sommare gli elementi del vettore scrivere una funzione asintoticamente più efficiente.

(a) È sufficiente un algoritmo $\mathcal{O}(n^2)$, che usa tecniche standard (riflettere su come sia possibile calcolare incrementalmente le somme parziali).

(b) ★ Se siete amanti del *divide-et-impera*, potete cimentarvi in un sofisticato algoritmo di complessità $\mathcal{O}(n \log n)$.

(c) ★★ Se, infine, avete un po' di talento visionario per la semplicità, potreste scoprire un semplicissimo (ma affatto ovvio da trovare) algoritmo $\Theta(n)$.

3.8 Segmento di spessore s (esame 23/6/23)

Dato un vettore v di lunghezza n, definiamo lo *spessore* di v, notazione $\mathbf{spess}(v)$ come $\max_{i,j \in [0,n)} v[i] - v[j]$. La definizione si estende naturalmente ai *segmenti* di vettore su un intervallo di indici $[inf, sup)$ come segue: $\mathbf{spess}(v[inf, sup)) = \max_{i,j \in [inf,sup)} v[i] - v[j]$.

Dare un algoritmo che preso in input un vettore di interi v e un intero k determina lo spessore massimo dei segmenti di v di lunghezza al più k. Ritornare come risultato sia lo spessore che l'indice di inizio del segmento di spessore massimo. Argomentare la correttezza e determinare la complessità del vostro algoritmo.

Esempio: Dato il vettore $\{2, 6, 1, 4, 11, 5, 7, 8, 0\}$ e l'intero $k = 3$, il segmento di spessore massimo di lunghezza 3 è $\{1, 4, 11\}$. L'algoritmo deve tornare 10 (spessore massimo) e 2 (indice di inizio del segmento di spessore massimo).