DIETI UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II
DIPARTIMENTO DI INGEGNERIA ELETTRICA E DELLE TECNOLOGIE DELL'INFORMAZIONE

Corso di Laurea Magistrale in Ingegneria Informatica

CORSO DI ALGORITMI E STRUTTURE DATI

Prof. ROBERTO PIETRANTUONO

Homework set #2

Istruzioni

Si prepari un file PDF riportante il vostro nome e cognome (massimo 2 studenti). Quando è richiesto di fornire un algoritmo, si alleghi un file editabile (ad esempio, .txt, .doc) riportante l'algoritmo in un linguaggio a scelta, corredato da almeno tre casi di test. Laddove opportuno, si fornisca una breve descrizione della soluzione: l'obiettivo non è solo eseguire l'esercizio e riportare il risultato, ma far comprendere lo svolgimento.

PROBLEMA 1

Data un vettore che può contenere numeri interi sia positivi che negativi, trovare il sottoarray di numeri contigui che ha la somma più grande, e riportare tale somma.

INPUT

Ogni riga contiene un caso di test, rappresentato dagli elementi del vettore di cui si vuole calcolare la somma del massimo sottoarray. I casi di test terminano con una riga END

OUTPUT

Ogni riga contiene il valore della somma richiesto per il corrispondente caso di test

Sample Input

-1 -3 4 2
-1 2 -5 7
END

Sample Output

6
7

Si alleghi al PDF un file editabile riportante l'implementazione in un linguaggio a scelta, corredato da almeno due casi di test, oltre quelli del sample input/output, con il corrispondente output atteso. Si forniscano i tre casi di test nello stesso formato del "sample input". Si riporti la complessità.

PROBLEMA 2

Dato un certo importo da pagare di V centesimi ed una lista di *n* monete $\mathrm{coin}[\mathrm{n}]$ che possiamo usare (per esempio n = 3 monete, quelle da 1 centesimo ($\mathrm{coin}[0] = 1$), da 5 centesimi ($\mathrm{coin}[1] = 5$), da 10 centesimi ($\mathrm{coin}[2] = 10$)), si scriva un algoritmo per determinare il numero minimo di monte che dobbiamo usare per arrivare all'importo esatto V o al più piccolo importo maggiore di V. Si assuma di avere un numero illimitato di monete di tutti i tipi.
Ad esempio:

V = 10, n = 2, $\mathrm{coin}[0]$ = 1, $\mathrm{coin}[1]$ = 5. In tal caso si possono usare:

- 10 monete da 1 centesimo (10 x $\mathrm{coin}[0]$). Totale monete usare: 10
- 1 moneta da 5 e 5 monete da 1 (1 x $\mathrm{coin}[1]$ + 5 x $\mathrm{coin}[0]$). Totale monete usate: 6
- 2 monete da 5 (2 x $\mathrm{coin}[1]$). Totale monete usate: 2. La soluzione ottima è questa.

Altro esempio: V = 7, n = 4, $\mathrm{coin}[0]$ = 1, $\mathrm{coin}[1]$ = 3, $\mathrm{coin}[2]$ = 4, $\mathrm{coin}[3]$ = 5. La soluzione ottima è usare due monete $\mathrm{coin}[1]$ + $\mathrm{coin}[2]$ = 3 + 4 = 7.

INPUT

Ogni riga contiene un caso di test, in cui il primo numero rappresenta V, il secondo numero rappresenta n, e gli n numeri successivi rappresentano i valori del vettore coin[].
I casi di test terminano con una riga END

OUTPUT

Ogni riga contiene, per ogni caso di test, il numero di monete minimo richiesto per arrivare all'importo V.

Sample Input

10 2 1 5
7 4 1 3 4 5
END

Sample Output

2
2

Si alleghi al PDF un file editabile riportante l'implementazione in un linguaggio a scelta, corredato da almeno due casi di test, oltre quelli del sample input/output, con il corrispondente output atteso. Si forniscano i tre casi di test nello stesso formato del "sample input". Si riporti la complessità.