DIETI UNINA 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

Indicazioni

Si consegni un file in **formato editabile (.txt, .docx, .rtf, etc.)** nominandolo "*CognomeNome*", in cui è riportata l'implementazione (nel linguaggio scelto) seguita da una indicazione della complessità temporale dell'algoritmo implementato (complessità nel caso peggiore, è sufficiente il limite superiore O(f(n))). Se si utilizzano librerie di cui non si conosce la complessità, lo si indichi nella spiegazione (ad esempio, "la complessità è O(n log n) al netto della complessità dell'algoritmo *x,* che è non nota"). Se si utilizza la randomizzazione, si indichi anche il tempo di esecuzione atteso.

PROBLEMA 1

E' data una matrice N x M di caratteri. Si scriva un algoritmo per determinare se una parola (una sequenza di caratteri, dunque) data in ingresso è presente nella matrice, assumendo che, a partire dal carattere i-esimo della parole, si può cercare il prossimo carattere in una qualsiasi delle 8 celle adiacenti (sopra, sotto, destra, sinistra e diagonalmente). Può esistere più di una soluzione (ossia la parola può essere ottenuta con diversi percorsi: in tal caso è sufficiente stampare il primo).
Esempio: si cerchi la parola 'hello'

s r z h d
f o a g e
x d n l o
o r a t n
z p o i n

In output si stampino gli indici riga e colonna (conteggio parte da 1) delle celle del percorso, in questo caso:
1 1
2 2
3 3
2 4

Si utilizzi il **backtracking**. Si può creare una matrice della stessa dimensione di quella in ingresso con inizialmente tutti 0. Si inizia ad esplorare (provando tutte le posizioni come celle di partenza), mantenendo un indice che parte da 1 per il carattere i-esimo che si sta cercando. Quando si trova una corrispondenza si scrive il valore dell'indice nella cella.

INPUT

L'input inizia con un numero intero che indica il numero di casi di test. Ogni test, che inizia alla riga successiva, è composta da: una riga che riporta la parola da cercare; una riga che riporta la dimensione delle righe e delle colonne della matrice N x M; da N righe ciascuna con M caratteri separati da uno spazio.

OUTPUT

Per ogni caso di test, si stampi una riga per ogni cella presente nel percorso (con gli indici di riga e colonna della cella), seguite da una riga con la parola END che indica la fine dell'output per quel caso di test. Se la parola non è presente nella matrice, si stampi direttamente END.

Sample Input

2
age
5 5
s r z h d
f o a g e
x d n l o
o r a t n
z p o i n
ora
2 3
r t r
o a r

Sample Output

2 3
2 4
2 5
END
2 1
1 1
2 2
END

PROBLEMA 2

Una progressione aritmetica è una successione di numeri tali che la differenza tra ciascun termine (o elemento) della successione e il suo precedente è costante. La somma degli elementi in una progressione di *n* elementi è data da: $S = ½ * n * (a_1 + a_n)$, dove a_1 ed a_n sono il primo e l'ultimo elemento. E' dato in input un array di interi positivi che rappresenta una progressione aritmetica dove però manca un elemento. La massima lunghezza è 30, il massimo valore è 1000. Si implementi un algoritmo che trova l'elemento mancante.
Suggerimento: una soluzione semplice si ottiene scorrendo il vettore, complessità O(N). Si utilizzi un algoritmo *divide et impera* con complessità O(LogN).

INPUT

La prima riga contiene il numero di casi di test N. Ogni caso di test, che inizia alla riga successiva, è composto da due righe: la prima riportante la dimensione del vettore, la seconda riportante gli elementi del vettore separati da uno spazio.

OUTPUT

Si stampi, per ogni caso di test, il numero mancante.

Sample Input

2
4
1 3 5 9
6
3 6 12 15 18 21

Sample Output

7
9