

Preparazione Midterm

2026/10/28 – Preparazione Parziale

L. Alessandrini

Credits: A. Montenegro per gli esercizi

Vecchi TDE

Esempi di esercizi d'esame

Esercizio 01

TDE Settembre 2024

Testo

Si scriva una funzione **conta** che riceve in ingresso una stringa **S** contenente più parole separate da virgole e contenenti solamente lettere minuscole (oltre ad eventuali parametri aggiuntivi utili per lo sviluppo della funzione).

La funzione **conta** deve restituire al chiamante la lettera dell'alfabeto che compare più frequentemente tra le iniziali delle parole (si supponga per semplicità sia unica).

Si scriva quindi una seconda funzione **elimina** che riceve in ingresso la stringa **S** (contenente più parole e solo caratteri minuscoli), un carattere **C** (oltre ad eventuali parametri aggiuntivi ritenuti utili per lo sviluppo della funzione), e riporta al chiamante una stringa **T** contenente le sole parole di **S** che iniziano per **C**, separate da virgole. Nel main si stampi il vettore restituito dalla funzione **elimina**.

Esercizio 02

TDE Marzo 2017

Testo

Scrivere una funzione che riceve due stringhe: *parola* e *elimina*. La funzione cerca in *parola* tutti i caratteri che compongono la stringa *elimina*, e li rimuove solamente se li trova tutti e nell'ordine in cui compaiono in *elimina*, anche se non consecutivi. L'operazione viene ripetuta finché in *parola* è contenuta un'intera istanza della stringa *elimina*. La funzione, oltre a modificare l'array *parola*, restituisce il numero di volte che ha eliminato i caratteri dell'intera stringa *elimina* da *parola*. Nell'eliminare caratteri non devono essere lasciati buchi, ma *parola* dev'essere ricompattata.

Esempio

parola: amaarrreeemmmmmarrreeaaaarrmae

elimina: mare

La funzione trova tutte le lettere di "mare" in "aMAaRrrEeeemmmmmarrreeaaaarrmae" che diventa "aarreeemmmmmarrreeaaaarrmae"

La funzione trova tutte le lettere di "mare" in "aarreeeMmmmARrrEeaaaarrmae" che diventa "aarreeemmmrreaaaarrmae"

La funzione trova tutte le lettere di "mare" in "aarreeemmMrreAaaaRrrmaE" che diventa "aarreeemmrreaaaarrma"

Poi non trova più tutte le lettere e restituisce 3

Esercizio 03

TDE Gennaio 2022

Testo

Si scriva una funzione **rimuoviValori** che prende in ingresso una matrice M di interi di dimensioni $N \times N$ (con `#define N 4`), un intero x e ulteriori variabili aggiuntive, se necessarie. La funzione identifica i valori di M che sono a distanza minima da x e modifica M mettendo a zero tutti i valori che stanno sulla parallela alla diagonale principale passante per i valori sopra identificati. Si guardino gli esempi sotto per evitare fraintendimenti su quali elementi mettere a zero.

N.B. Qualora ci fossero più valori in M alla stessa distanza da x , la funzione **rimuoviValori** ripete l'operazione per tutti i valori a distanza minima. N.B. Si consiglia fortemente l'uso di funzioni ausiliarie per produrre un codice semplice, leggibile e compatto

Si scriva nel main l'invocazione alla funzione **rimuoviValori** e si stampi la matrice nel main, non in **rimuoviValori**.

Esercizio 04

TDE Gennaio 2023

Testo

Si una funzione riordina che prende in ingresso due vettori **u** e **v** (più eventuali variabili necessarie) e che riordina gli elementi del primo vettore passato in input (**u** nel nostro caso) nel seguente modo:

- tutti gli elementi di **u** che non compaiono in **v** saranno nelle prime posizioni di **u** dopo l'invocazione
- tutti gli elementi di **u** che compaiono in **v** saranno nelle ultime posizioni di **u** dopo l'invocazione

Si faccia riferimento all'esempio sotto per eventuali dubbi su come modificare **u**.
Si invochi quindi la funzione relativamente al seguente codice e si stampi il contenuto del vettore **u** a schermo.

N.B. non è possibile stampare all'interno della funzione riordina

$u = [4, 1, 3, 7, 0]$ e $v = [1, 6, 8, 4] \rightarrow u = [3, 7, 0, 4, 1]$

Esercizio 05

Alleanze tra Università

Testo

Definire un tipo di dato **University** che descrive un'università. Ogni università è caratterizzata da una stringa che indica il **nome** per **esteso** (ad esempio, "Politecnico di Milano", "École Polytechnique Fédérale de Lausanne", etc.), una stringa che descrive il **nome corto** (ad esempio, "Polimi", "EPFL", etc), due caratteri che contengono la **sigla della nazione** in cui si trova la sede principale (ad esempio, "IT", "CH", etc), un intero che rappresenta il **numero di studenti** immatricolati (ad esempio, 42496, 13445, etc) e un numero reale che descrive il punteggio ottenuto dall'università secondo il **ranking** SDD (ad esempio, 46.7, 277.0, etc).

Definire un tipo di dato **Alleanza** che descrive una lista di **università** che collaborano insieme e il **numero di università** nell'alleanza.

Implementare le seguenti funzioni:

1. **ricerca()**: la funzione verifica se è presente nell'alleanza un'università con un dato nome esteso. Se presente, ritorna le informazioni dell'università, altrimenti segnala al chiamante l'assenza.
2. **filtraPerNazione()**: la funzione restituisce una nuova lista contenente solo le università dell'alleanza che appartengono a una specifica nazione dall'utente.
3. **sostituisci()**: la funzione sostituisce un'università nell'alleanza, identificata dal nome corto, con un'altra università passata in ingresso. La funzione ritorna l'università appena rimossa. La funzione comunica al chiamante se la sostituzione è avvenuta o meno.

massimoStudentiPerNazione(): la funzione ricerca, per ogni nazione nell'alleanza, l'università con il maggior numero di studenti, e restituisce un lista con le università con il numero massimo di studenti per nazione.

Esercizio 07

TDE 2025/01/17 (Es. 1)

Testo

Si consideri un tipo di dato Tavolo composto da due campi che riportano il numero di persone sedute al tavolo (int) e il prezzo totale del cibo ordinato dal tavolo (float):

```
// Definizione del tipo Tavolo  
typedef struct {  
    int commensali; // Numero di persone sedute al tavolo  
    float prezzo;    // Prezzo totale del cibo ordinato dal tavolo  
} Tavolo;
```

La matrice quadrata Tavolo $M[N][N]$ rappresenta i tavoli di un ristorante, dove ciascun elemento contiene i dati relativi ad un tavolo.

Esercizio 07

TDE 2025/01/17 (Es. 1)

Testo

```
Tavolo M[N][N] = {  
    {{2, 100}, {3, 150}, {4, 200}},  
    {{2, 80}, {4, 180}, {3, 120}},  
    {{4, 250}, {2, 90}, {3, 140}}  
};
```

in cui al tavolo (0,0) erano sedute due persone che hanno speso 100 euro, al tavolo (0,1) erano sedute tre persone che hanno speso 150 euro, ...

Esercizio 07

TDE 2025/01/17 (Es. 1)

Testo

Si implementi una funzione “tavoliPiccoli” che riceve in input la matrice M rappresentante il ristorante, più eventuali variabili aggiuntive per la corretta gestione della matrice, e restituisce al chiamante i valori degli indici corrispondenti al tavolo con il minor numero di commensali. Qualora ci fossero due tavoli con lo stesso numero di commensali, la funzione restituisce gli indici del tavolo che ha speso di più. Il chiamante deve stampare a schermo la posizione del tavolo riportato dalla funzione. Nell'esempio i tavoli con meno persone sono i tre tavoli con due persone, tra di essi quello in cui si è speso di più è quello con 100 euro spesi.

Esercizio 07

TDE 2025/01/17 (Es. 1)

Testo

Si implementi una funzione “speseMedie” che prende in input la matrice M, più eventuali variabili aggiuntive per la corretta gestione della matrice, e che riporta al chiamante un array med contenente le medie dei totali spesi dai tavoli aventi 2, 3 e 4 commensali. Se in M non è presente un tavolo con un particolare numero dei commensali, il corrispettivo elemento di med deve essere zero. Il chiamante deve stampare med riportato dalla funzione. Nel tavolo di esempio il risultato sarà [90, 136.67, 210].

Esercizio 07

TDE 2025/01/17 (Es. 1)

Testo

Si sviluppi la soluzione a partire dal seguente codice:

```
#include <stdio.h>
#define N 3
// Definizione del tipo Tavolo
typedef struct {
    int commensali; // Numero di persone sedute al tavolo
    float prezzo;    // Prezzo totale del cibo ordinato dal tavolo
} Tavolo;

int main() {
    // Matrice di tavoli
    Tavolo M[N][N] = {
        {{2, 100}, {3, 150}, {4, 200}},
        {{2, 80}, {4, 180}, {3, 120}},
        {{4, 250}, {2, 90}, {3, 140}}
    };
    // Chiamate alle funzioni
    // Stampa dei risultati
    return 0;
}
```

Esercizio 07

TDE 2025/01/17 (Es. 1)

Testo

Il programma stamperà:

```
> Indici del tavolo con meno commensali che ha speso di  
più: (0, 0)  
> Prezzo medio per 2, 3, 4 commensali: [90, 136.67, 210]
```

Esercizio 08

TDE 2026/07/17 (Es. 1)

Testo

Si scriva una funzione `vettoreDivisori` che prende in ingresso due vettori di interi `u` e `v` (oltre ad altre variabili, se necessarie) e riporta al chiamante un vettore `d` contenente tutti i divisori comuni di tutte le coppie $(u[i], v[i])$.

Si stampi nel `main` (e solamente nel `main`) il contenuto di `u`, `v`, e `d`

Esercizio 08

TDE 2026/07/17 (Es. 1)

Testo

Si consideri che:

- Quando i vettori u e v non hanno la stessa lunghezza, vengono processate solo le coppie $(u[i], v[i])$ fino a coprire il vettore più corto.
- In ogni caso, la lunghezza dei vettori u, v non è definita a priori e deve poter essere specificata al momento dell'invocazione della funzione

Per risolvere l'esercizio, si implementi obbligatoriamente la funzione ausiliaria `divisoriComuni` che prende in ingresso due interi e riporta al chiamante un vettore di divisori. È possibile implementare altre funzioni ausiliarie per facilitare lo sviluppo e la leggibilità del codice. Si assuma che il numero di elementi inseriti in d non ecceda mai 100, la sua dimensione reale.

Esercizio 08

TDE 2026/07/17 (Es. 1)

Testo

Si parta dal seguente
main:

```
#include<stdio.h>
#define N 10
#define M 100
int main()
{
    int u[N] = {242, 4, 0, 4620};
    int v[N] = {11, 8, 25, 210, 45};
    int d[M];

    // invocare la funzione vettoreDivisori

    // stampare u, v e d

    return 0;
}
```

Esercizio 08

TDE 2026/07/17 (Es. 1)

Testo

Ecco l'output atteso dal programma

```
u = [ 242, 4, 0, 4620, ].
v = [ 11, 8, 25, 210, 45, ].
d = [ 1, 11, 1, 2, 4, 1, 2, 3, 5, 6, 7, 10, 14, 15,
21, 30, 35, 42, 70, 105, 210, ].
```

Thank You!

Mail: luca1.alessandrini@polimi.it

Website: <https://alessandriniluca.github.io/>