PROGRAMMAZIONE I E LABORATORIO DI PROGRAMMAZIONE I

Esercizi per la prova all'impronta di Programmazione I e Lab. Programmazione I

Di seguito sono mostrate le tracce degli esercizi tra le quali viene estratta la prova individuale all'impronta.

TRACCIA 1

Sviluppare una function C che, dati come parametri di input un *array* di **int** e il suo **size**, determina e restituisce come parametro di output il secondo **più grande elemento** dell'array (N.B.: non bisogna seguire l'idea di ordinare prima l'*array*).

TRACCIA 2

Sviluppare una function C che, dato come parametro di input un *array* di tipo **struct punto {double x; double y; }** e il suo **size**, determina e restituisce come parametri di output gli **indici** dei due punti che hanno **distanza minima** tra loro. I campi **x** e **y** contengono l'ascissa e l'ordinata, rispettivamente, di un punto.

TRACCIA 3

Sviluppare una function C che, dato come parametro di input un *array* di tipo **struct punto {double x; double y; }** e il suo **size**, determina e restituisce come parametro di output la **massima distanza tra i punti**. I campi **x** e **y** contengono l'ascissa e l'ordinata, rispettivamente, di un punto.

TRACCIA 4

Sviluppare una function C che, dati come parametri di input un *array* 2D di **double**, il numero delle righe e il numero delle colonne, determina e restituisce come parametro di output il **massimo** tra le somme degli elementi di ogni colonna.

TRACCIA 5

Sviluppare una function C che, dati come parametri di input un *array* 2D di **int**, il numero delle righe e il numero delle colonne, determina e restituisce come parametro di output il **massimo** tra le somme degli elementi di ogni riga.

TRACCIA 6

Scrivere una funzione che dati in input due *array* ordinati (rispetto al campo matricola) di elementi della seguente struttura

```
struct studente {char *nome; char *cognome; int matricola;};
```

restituisca in output l'*array* **fusione** dei due *array*. La fusione deve avvenire in base al campo matricola.

TRACCIA 7

Scrivere una funzione che dati in input due *array* di strutture del seguente tipo

```
struct prodotto {char *nome; int codice; double prezzo;};
```

restituisce in output 1 se i due *array* di struct sono uguali, 0 se non lo sono. Si noti che due dati struct sono uguali se sono uguali tutti i loro campi.

TRACCIA 8

Sviluppare una function C che, data come parametro di input una stringa che rappresenta un testo in italiano, determina e restituisce come parametro di output il numero di parole di **tre lettere** contenute nel testo. Nel testo le parole sono separate da un unico *spazio*.

TRACCIA 9

Sviluppare una function C che, data come parametro di input una stringa che rappresenta un testo in italiano, determina e restituisce come parametro di output il numero di parole che terminano in **are** contenute nel testo. Nel testo le parole sono separate da un unico *spazio*.

TRACCIA 10

Sviluppare una function C che, data come parametro di input una stringa che rappresenta un testo in italiano, determina e restituisce come parametro di output il numero di parole che iniziano con **a** e terminano con **e** contenute nel testo. Nel testo le parole sono separate da un unico *spazio*.

TRACCIA 11

Sviluppare una function C che, data come parametro di input una stringa che rappresenta un testo in italiano, determina e restituisce come parametro di output il numero delle parole contenute nel testo che hanno almeno **5 vocali**. Nel testo le parole sono separate da un unico *spazio*.

TRACCIA 12

Sviluppare una function C che, data come parametro di input una stringa che rappresenta un testo in italiano, determina e restituisce come parametri di output la **parola di lunghezza massima** contenuta nel testo e la sua lunghezza. Nel testo le parole sono separate da un unico *spazio*.

TRACCIA 13

Sviluppare una function C che, data come parametro di input una stringa che rappresenta un testo in italiano, determina e restituisce come parametri di output la parola di **lunghezza minima contenuta** nel testo e la sua lunghezza. Nel testo le parole sono separate da un unico *spazio*.

TRACCIA 14

Sviluppare una function C che, data come parametro di input una stringa che rappresenta un testo in italiano, determina e restituisce come parametri di output la parola di **lunghezza minima contenuta** nel testo e la posizione di inizio della parola nella stringa. Nel testo le parole sono separate da un unico *spazio*.

TRACCIA 15

Sviluppare una function C che, dati come parametri di input un *array* di **char** e il suo **size**, determina e restituisce come parametro di output l'array (di size 21) del **numero delle occorrenze** delle 21 lettere dell'alfabeto italiano (per es. il numero di occorrenze della lettera a è il numero di volte in cui la lettera a compare nel testo).

TRACCIA 16

Sviluppare una function C che, dati come parametri di input un array di **char** e il suo **size**, determina e restituisce come parametro di output l'array **occorrenze** (di size 21) del numero delle occorrenze dell'evento **a** precede ognuna delle 21 lettere dell'alfabeto italiano (cioè **occorrenze[0**] è il numero di volte in cui accade che "**a** precede **a"**, cioè che nel testo compare aa, **occorrenze[1]** è il numero di volte in cui accade che "**a** precede **b"**, cioè che nel testo compare ab, **occorrenze[2]** è il numero di volte in cui accade che **a** precede **c**, cioè che nel testo compare ac, …).

TRACCIA 17

Sviluppare una function C che, dati come parametri di input un array di **char** e il suo **size**, determina e restituisce come parametro di un dato logico che indica se il testo nell'array è un **pangramma**, ovvero è un testo che contiene, almeno una volta, tutte le 21 lettere dell'alfabeto italiano.

TRACCIA 18

Sviluppare una function C che, dati come parametri di input un *array* di **char** e il suo **size**, determina e restituisce come parametro di output il **carattere più frequente**.

TRACCIA 19

Sviluppare una function C che, dati come parametri di input un *array* di **char** e il suo **size**, determina e restituisce come parametro di output il **carattere meno frequente**.

TRACCIA 20

Dato un elenco (array) di persone partecipanti a un concorso, ordinare l'elenco in ordine alfabetico in base al campo cognome. La struttura che identifica il partecipante è

```
struct persona {char *nome; char *cognome;};

typedef struct persona id;

struct partecipante {id *utente; unsigned short codice; };
```

TRACCIA 21

Scrivere un programma per simulare l'inserimento di un PIN per il telefonino.

Nella prima fase viene costruito il PIN di riferimento, che deve essere di lunghezza 5 e deve essere costruito in modo casuale. Nella seconda fase l'utente inserisce il codice di tentativo e ha al massimo 3 tentativi per indovinarlo.

TRACCIA 22

Due giocatori si sfidano lanciando un "dado truccato". Il dado ha dei valori interi nell'intervallo [5 , 15]. A ogni turno vince il giocatore che ottiene un punteggio maggiore. In caso di parità il punto viene assegnato a entrambi. Simulare 10 sfide e visualizzare il giocatore che vince più volte.

TRACCIA 23

Scrivere una function C che ha come input i dati che identificano uno studente (nome, cognome, matricola) e che restituisce in output una struttura dati opportuna, che contiene i dati di identificazione e il libretto universitario con al massimo 20 esami. Ogni esame è caratterizzato da denominazione, cfu e voto.