Esercizi per familiarizzarsi con il linguaggio ANSI C.

Stringhe

Per ognuno di questi esercizi, scrivere un programma che contenga l'implementazione della funzione richiesta ed anche un main in cui la funzione viene chiamata e il risultato della funzione viene stampato a video, in modo da valutarne il comportamento. Il main dovrà preliminarmente definire e inizializzare delle variabili e/o strutture dati da usare come argomento della funziona.

1) Implementare la funzione int strlen(char *str); che restituisce la lunghezza della stringa C passata come argomento.

2) Implementare la funzione int strconta(char *str, char ch); che restituisce il numero di caratteri della stringa *str* che siano uguali al carattere *ch* passato come argomento.

3) Implementare la funzione char *strcerca(char *str1, char str2); che verifica se all'interno della stringa puntata da str1 è presente una istanza della sottostringa puntata da str2. La funzione restituisce NULL se la sottostringa non è presente, restituisce il puntatore all'inizio della sottostringa trovata in caso contrario.
PRIMA str1="tanto va la gatta al largo" str2="atta"
Supponendo che str1 inizi all'indirizzo 0x1000,
DOPO il risultato sarà l'indirizzo 0x100D del carattere 'a' in cui inizia ls stringa "atta al largo":

4) Implementare la funzione int strsostitusci(char *str1, char ch, char *str2); che cerca all'interno della stringa puntata da *str1* tutte le istanze del carattere *ch*, e sostituisce questi caratteri con la stringa puntata da str2, restituendo il numero delle sostituzioni effettuate.
PRIMA str1="tanto va la gatta al largo" ch='a' str2="BOF"
DOPO str1="tBOFnto vBOF lBOF gBOFttBOF BOFl lBOFrgo"

5) Implementare la funzione int strelimina(char *str, char ch); che cerca all'interno della stringa puntata da *str1* tutte le istanze del carattere *ch* e lo elimina dalla stringa, restituendo il numero dei caratteri eliminati.

6) Implementare la funzione void strinverti(char *str); che inverte l'ordine dei caratteri che compongono la stringa puntata da str. Ad esempio, se la stringa str punta a "VAF", dopo la chiamata la stringa sarà diventata "FAV".

Strutture Dati Liste

7) Definire la struttura dati che implementa una "lista doppiamente linkata", cioè una lista in cui ogni elemento mantiene un puntatore al successivo (se esiste) ed un puntatore al precedente (se esiste). L'elemento mantiene anche un campo intero di nome key. Tale definizione dovrà essere realizzata in un file .h da includere nei file .c che ne hanno bisogno.

Sfruttando la struttura dati "lista doppiamente linkata" precedentemente definita:

8) Si costruisca un programma che generi una lista doppiamente linkata di almeno 5 o 6 elementi, ordinata in modo crescente secondo il campo key. Per la costruzione della lista, il programma dovrà invocare ripetutamente una opportuna funzione che aggiunge un elemento alla lista inserendolo ordinatamente nella lista stessa secondo il campo key.

9) Si aggiunga a tale programma una funzione che elimina dalla lista l'elemento avente una determinata key specificata tra gli argomenti della funzione.

Strutture Dati Matrici

10) Definire la struttura dati che implementa una "matrice dinamica bidimensionale di elementi double", e che mantiene **anche** dei campi contenenti il numero di righe ed il numero di colonne contenute nella matrice dinamica. Tale definizione dovrà essere realizzata in un file .h da includere nei file .c che ne hanno bisogno.

Sfruttando la struttura dati "matrice dinamica bidimensionale di elementi double" precedentemente definita:

11) Si costruisca un programma che generi due matrici aventi stesso numero di righe e stesso numero di colonne.

12) Si implementi una funzione che stampa a video il numero di righe e di colonne e tutti gli elementi di una matrice passata come argomento.

13) Si implementi in un file .c separato una funzione che prende come argomenti due matrici dinamiche aventi stesso numero di righe e stesso numero di colonne, e che restituisce una nuova matrice dinamica con stesso numero di righe e stesso numero di colonne, i cui valori siano la somma degli corrispondenti elementi delle due matrici originali. La funzione verrà utilizzata dentro il programma principale per creare una nuova matrice che sia la somma delle due matrici precedentemente create.

14) Si implementi una funzione che aggiunge una riga ed una colonna ad una matrice passata come argomento.

Strutture Dati Polivalenti

Si definisca una struttura dati così fatta **typedef struct s { char tipo; void *ptr; }NODO;**
La struttura dati mantiene tipi di dati diversi, in particolare, il puntatore ptr punta al dato, mentre il campo tipo specifica che tipo di dato è quello a cui il puntatore ptr punta. Ad esempio:

se tipo vale 'C' allora il puntatore ptr punta ad un char,
se tipo vale 'I' allora il puntatore ptr punta ad un int,
se tipo vale 'F' allora il puntatore ptr punta ad un float;

15) Si costruisca un programma che utilizza un vettore di 7 elementi di tipo NODO. Tale programma riempie il vettore allocando dinamicamente spazio per 2 elementi di tipo char, 2 di tipo int e 3 di tipo float, facendo poi puntare ciascun puntatore ptr ad uno degli elementi allocati, e settando il campo tipo di ciascun elemento del vettore in accordo col tipo di dato a cui il puntatore ptr di quell'elemento punta.

16) Si costruisca, in un file separato, una funzione che stampa correttamente i valori puntati dai puntatori ptr di ogni elemento di un vettore che viene passato come argomento alla funzione stessa. Il termine "correttamente" indica che la stampa deve essere svolta in accordo col tipo a cui ciascun puntatore punta, cioè usando nella printf %c per i char, %d per gli interi, %f per i float).

17) Si costruisca una funzione che assegna al dato puntato dal puntatore ptr di ciascun elemento del vettore un valore a scelta del programmatore ma che sia diverso da elemento a elemento. Invocarla nel programma principale e poi invocare la funzione che stampa i valori del vettore (quella implementata al punto 16).

18) Si implementi una funzione che restituisce un valore double che sia la somma di tutti i valori puntati dai puntatori di ciascun elemento del vettore. Si noti che la somma dovrà essere fatta con un opportuno type cast, in modo da utilizzare ciascun puntatore ptr come un puntatore al corretto tipo di dato. Tale funzione verrà chiamata nel programma principale ed il risultato restituito verrà stampato a video.