Esercizi 3AIIN 05.03.2021

- Salva in Moodle un file zip che contiene solo i pgm sorgenti.
- Formato nome file: **CognomeNomeGGMMAAAA.estensione**

dove GGMMAAAA è la data di assegnazione del lavoro.

- Terminata ogni lezione di laboratorio devi caricare in Moodle il lavoro svolto.
- A casa eventualmente puoi finirlo e/o correggerlo. Hai tempo una settimana.
- Anche chi è assente il giorno della consegna è tenuto a svolgere il lavoro assegnato.
- Verranno fatti controlli a campione e sicuramente durante le interrogazioni.

Puoi consultare il sito: **https://www.cplusplus.com/**

Esercizio n.1

Definire un **tipo di dato** atto a descrivere un libro (con autore (cognome e nome), titolo, costo, data di pubblicazione) ed uno scaffale di libri.

Codificare le seguenti funzioni:

1) acquisizione dei dati relativi ad un libro e inserimento del libro nella struttura dati (se c'è posto);
2) stampa dei dati relativi a tutti i libri presenti sullo scaffale
3) calcola il costo di tutti i libri presenti sullo scaffale
4) copia tutti i libri con autore che ha il cognome che inizia per determinato carattere (parametro di ingresso) in una seconda variabile di tipo scaffale (parametro con direzionalità di output).

Scrivere un main di prova.

Inizio dichiarazione/definizione tipo di dato

```
#define DIM_STR_NOME 21
#define DIM_STR_TITOLO 31
#define MAX_LIBRI 10

struct data{
        int giorno;
        int mese;
        int anno;
};

struct libro{
        char cognome[DIM_STR_NOME];
        char nome[DIM_STR_NOME];
        char titolo [DIM_STR_TITOLO];
        double costo;
        data dataPubblicazione;
};
```

1^ versione	2^ versione

Nel main():
 	
libro scaffale[MAX_LIBRI];
int numLibri;
	struct **libreria**{
	libro scaffale[MAX_LIBRI];
	int numLibri;
	};
	
	Nel main():
	
	libreria lib;

Fine dichiarazione/definizione tipo di dato

Implementa le funzioni utilizzando la 2^ versione.

```
libro leggiLibro(){
        //legge i dati di un libro e ritorna il record che li contiene
        //il record ritornato sarà inserito dalla procedura inser() nella
        //struttura dati (libreria)
}

int insert(libreria & lib){                    //inserimento in coda
      //usa, al suo interno, leggiLibro()
}
```

La procedura insert() resituisce 1 (inserimento ok), 0 altrimenti.

Esercizio n.2 Vettori di stringhe – Matrici

Inserisci in un vettore di stringhe, di dimensione **MAX_CITTA**, i nomi di **numCitta** città. (numCitta <= MAX_CITTA)
Stampa il vettore di stringhe (che contiene i nomi delle città).
Utilizzando il vettore di stringhe, inserisci in una matrice di interi la distanza tra le città. Ti ricordo che la matrice è simmetrica (se chiedi la distanza tra i e j non devi richiedere la distanza tra j e i e neppure devi chiedere la distanza tra i e i).
Stampa la matrice delle distanze.
Richiedi due città e ricerca la loro distanza. Le città possono anche non essere presenti nella struttura dati.

```
#define MAX_CITTA 5
#define MAX_CAR 20+1

int main(){
char citta[MAX_CITTA][MAX_CAR];
int distanze[MAX_CITTA][MAX_CITTA];
int numCitta;
char partenza[MAX_CAR];
char arrivo[MAX_CAR];
int trovato;
int posizPartenza, posizArrivo;
int i,j;

        //sottpgm che chiede numCitta (≤ DIM_CITTA) e le inserisce nel vettore citta

        //sottpgm che inserisce nella matrice le distanze tra le diverse città

        //sottopgm che stampa le città (per test)

        //sottopgm che stampa la matrice delle distanze (per test)

        //chiedi nel main il nome della città di partenza

        //chiedi nel main la città di arrivo

        //sottopgm che date le due città ritorna la distanza o un valore che indica
        //che una delle due città o tutte e due non sono presenti. Questo
        // sottopgm usa un altro sottpgm che ricerca (in modo sequenziale) nel
        //vettore delle città la posizione della città stessa (può non essere presente).
        //Quest'ultimo sottopgm (ricercaCitta())viene chiamato due volte: una per la
        //città di partenza e l'altra per la città di arrivo. Trovate le posizioni delle due
        // città nel vettore città, attraverso un accesso diretto alla matrice, estraggo
        //la distanza.

        return 0;
}
```