Blocco 7: Funzioni

F_1 – Quadrato
Scrivere una funzione per il calcolo del quadrato di un intero.
La funzione riceve in ingresso un intero e restituisce l'intero corrispondente al quadrato.

F_2 – Somma di due interi
Scrivere una funzione per calcolare la somma di due interi passati per valore.

F_3 – Stampa *
Scrivere una funzione che stampa un numero di asterischi passato per valore.
Richiamare la funzione in un programma che stampa 20 sequenze casuali di '*'.

F_4 – Numero primo
Scrivere due funzioni per verificare se un numero è primo, utilizzando i prototipi forniti; la prima funzione chiama la seconda.

- typedef enum {falso,vero} SiNo;
- SiNo primo (int);
- SiNo divisibile (int,int);

F_5 – Interessi
Scrivere una funzione che calcoli gli interessi maturati su un capitale in N anni.

F_6 – Maggiore di tre
Scrivere una funzione che riceve tre parametri di tipo puntatore a intero e restituisce il puntatore del parametro con valore maggiore.

F_7 – Ordina 3 parametri
Scrivere una funzione che scambi i contenuti dei tre parametri ricevuti in modo che il primo contenga il valore maggiore. La funzione restituisce inoltre la somma dei tre valori.

F_8 – Esempi
Simulazione di chiamate a funzioni.

F_9 – Esempi
Simulazione di chiamate a funzione

F_10 – Prima occorrenza
Ricerca della prima occorrenza di un elemento in un vettore tramite funzione.
Il vettore è globale.
In caso di esito positivo della ricerca, nella cella relativa all'elemento trovato bisogna inserire la differenza, in valore assoluto, tra l'elemento precedente e quello successivo ad esso, oppure 0 in caso di errore.

F_11 - Cerca ripetizione
Scrivere un programma che verifichi la ripetizione di un elemento in un array (definito globalmente) sfruttando una funzione che trova la prima occorrenza dell'elemento "corrente" nelle posizioni successive dell'array (se presente).

F_12 - Pari o Dispari
Scrivere un programma che, generato un numero casuale, determini se esso è pari oppure dispari e lo memorizzi tramite un'apposita funzione in uno di due array (creare array dei pari ed array dei dispari).

F_13 - Ordinamento per selezione
Ordinamento per selezione tramite funzioni.

F_14 - Confronto di vettori
Scrivere una funzione che riceva 2 array di interi e la loro lunghezza e restituisca "1" se essi hanno lo stesso contenuto.

F_15 - Lunghezza stringa
Scrivere una funzione che calcoli la lunghezza di una stringa passata come parametro, sfruttando l'aritmetica dei puntatori.

F_16 - Percentuali di stringa
Scrivere una funzione che calcoli la percentuale di lettere e di cifre numeriche presenti nella stringa passata come parametro.
Il risultato (ovvero le due percentuali) viene restituito utilizzando un'unica struttura.

F_17 - To Upper
Implementare una funzione che converta in maiuscolo una stringa passata come parametro.

F_18 - Elimina punti
Scrivere una funzione che, riceva come parametro una stringa e, nel caso essa inizi con dei punti, la modifichi eliminando i punti stessi, e restituisca il numero di punti tolti.

F_19 - Biblioteca
Definire le strutture dati e le funzioni necessarie (inserimento e cancellazione libri) per gestire molteplici biblioteche. Una biblioteca contiene uno scaffale composto da vari libri (possono esserci posizioni vuote).
Ogni libro è caratterizzato da autore e titolo.

F_20 - Archivio
Implementare le funzioni necessarie ad inserire ed eliminare interi da un archivio con stato pieno/vuoto.
Il vettore dell'archivio viene passato come parametro.

In particolare si consideri il problema di memorizzare N numeri interi in un archivio, sfruttando un apposito flag per memorizzare lo stato delle singole celle (usata, libera).
Si definiscano le strutture dati richieste per generare tale archivio.
Si implementino, tramite apposite funzioni/procedure, le seguenti funzionalità:

1. Aggiunta di un elemento all'archivio
2. Eliminazione di un elemento dall'archivio
3. Visualizzazione degli elementi presenti nell'archivio
4. Ricerca di un elemento nell'archivio
5. Inserimento di un elemento solo se esso non è già presente nell'archivio
6. Incremento di una unità degli elementi presenti nell'archivio
7. Raggruppamento degli elementi validi all'inizio dell'archivio

Ottimizzare il codice implementando funzioni che possono essere utilizzate in più punti del programma.

F_21 - Game of Life
Il gioco della vita mostra l'evoluzione di un mondo molto semplice costituito da cellule che possono essere vive o morte.

Il campo di gioco è una matrice le cui celle possono contenere ciascuna un'unica cellula.
Le regole dell'evoluzione sono le seguenti:

- Una cellula muore per sovraffollamento nel caso in cui abbia un numero di cellule vicine compreso tra 4 e 8
- Una cellula muore per "desertificazione" nel caso in cui abbia 0 o 1 cellula vicina
- Una nuova cellula nasce se ha tre cellule vicine
- Una cellula rimane nel suo stato precedente nel caso abbia due cellule vicine

Si sviluppi un programma che mostri all'utente l'evoluzione del gioco ad ogni passo.
Si supponga che l'utente definisca lo stato iniziale del gioco

- indicando la dimensione del campo di gioco
- identificando il numero e la posizione delle cellule vive

...e che possa controllare l'avanzamento del gioco e la sua terminazione inserendo da tastiera opportuni comandi