CORSO DI FONDAMENTI DI INFORMATICA

Prof. Maristella Matera - A.A. 2021 / 2022

Laboratorio di Programmazione in C – Laboratorio 2

Esercizio 1 – Temperature

Scrivere un programma che:

- chiede all'utente di inserire le misure della temperatura corporea di un paziente rilevate nell'arco di 7 giorni e salva l'input in un array;
- stampa i risultati sotto forma di tabella con allineamento a destra;
- stampa accanto ai risultati un istogramma con allineamento a sinistra (un $*$ per ogni valore di temperatura superiore a 34 con approssimazione per difetto);
- calcola la temperatura media, massima e minima.

Esempio:

```
GIORNO       VALORE       ISTOGRAMMA
      1         37.8       ***
      2         40.6       ******
```

Utilizzare il costrutto `#define` per definire la durata del periodo di osservazione del paziente (7 giorni).

N.B.: nel caso di costanti definite con `#define` per convenzione si consiglia di usare nomi di costante scritti tutti con caratteri maiuscoli per distinguerle dalle variabili.

Esercizio 2 – Temperature Verticali

Modificare l'esercizio precedente in modo che i risultati vengano stampati sotto forma di un istogramma verticale, usando come carattere un asterisco (inserire un * per ogni valore di temperatura superiore a 34 con approssimazione per difetto), scrivendo in basso il valore relativo a ogni colonna.

Esempio:

```
          *          *
    *     *          *
    *     *          *
    *     *          *
    *     *     *    *
    *     *     *    *
   39    40    36   40
```

Esercizio 3 – Linguaggio di Dyck

Nella teoria del linguaggi formali, il linguaggio di Dyck consiste in stringhe bilanciate di parentesi quadre '[' e ']'. Codificare un programma C che sappia riconoscere se una stringa inserita dall'utente appartenga al linguaggio di Dyck o meno.

Esempio: La stringa `[[[][]][]]` appartiene al linguaggio di Dyck.

La stringa `[[[]][]]]` **non** appartiene al linguaggio di Dyck.

Esercizio 4 – Anagrammi

Si codifichi un programma C che legge due stringhe inserite dall'utente che rappresentano due parole e verifica se le parole sono anagrammi, cioè se è possibile ottenere l'una dall'altra tramite permutazione delle loro lettere. Stampa infine a schermo il risultato della verifica.

Esempio: Le parole **POLENTA** e **PENTOLA** sono anagrammi

Le parole **TAPPO** e **PATTO** non sono anagrammi

Esercizio 5 - Mosse su matrice (tratto dal TdE di Novembre 2015)

Sia data una matrice `M` di `N×N` numeri interi positivi (`N` costante) e un array `A` di `K` elementi (`K` costante) organizzato in base alla seguente dichiarazioni di tipo:

```
#define K
...
typedef struct {
      int mossaRiga;
      int mossaColonna;
} Mossa;

Mossa A[K];
```

Gli elementi di `A` rappresentano spostamenti all'interno della matrice `M` a partire dal suo primo elemento `M[0][0]`. I campi `mossaRiga` e `mossaColonna` possono assumere valori nell'insieme `{-1, 0, 1}`. In particolare:

- Il valore -1 indica uno spostamento verso la riga (colonna) precedenti rispetto alla posizione corrente
- Il valore 0 indica di non spostarsi
- Il valore 1 indica uno spostamento verso la riga (colonna) successiva.

Per esempio, se i valori di un elemento dell'array, `A[i]`, sono

`<A[i].mossaRiga = 0, A[i].mossaColonna = 1>`

allora rispetto all'elemento corrente è necessario rimanere nella stessa riga e muoversi di una colonna a destra.

Si scriva un programma che, una volta inizializzate le strutture dati M e A, calcoli e restituisca come valore di ritorno la somma degli elementi della matrice visitati seguendo le mosse memorizzate in A. Nel caso di mosse che portano a posizioni non lecite (cioè oltre i limiti imposti dalle dimensioni della matrice) il programma restituisce il valore -1 e termina immediatamente.

Esercizio 6 – Matrice Simmetrica

Scrivere un programma che riceve in input una matrice A, di dimensione NxN (N è definita all'interno del programma mediante una `#define`). La matrice sarà costituita da numeri interi inseriti dall'utente. Il programma svolge le seguenti operazioni:

- stampa la matrice A
- controlla se A è simmetrica
- stampa il risultato della verifica sulla simmetria.

Si ricorda che una matrice A è simmetrica quando per ogni coppia di indici i e j, A(i,j) = A(j,i).

Esercizio 7 – Crivello di Eratostene

Scrivere un programma che implementi l'algoritmo del Crivello di Eratostene per il calcolo delle tabelle dei numeri primi fino ad un valore N prefissato.

Il procedimento funziona come segue: si scrivono tutti i naturali a partire da 2 fino N in un elenco detto setaccio. Poi si cancellano (setacciano) tutti i multipli del primo numero del setaccio (escluso tale numero). Si prosegue con il numero seguente tra quelli non ancora cancellati e così via fino ad arrivare in fondo alla sequenza dei numeri. I numeri che restano sono i numeri primi minori o uguali a N. Si può provare che il procedimento di setacciatura per ricercare i primi fino ad un certo numero N cessa sempre quando si supera la radice quadrata di N.

Per i più curiosi: è disponibile la rappresentazione animata del procedimento qui:
https://it.wikipedia.org/wiki/Crivello_di_Eratostene

Implementare quindi il programma seguendo i seguenti passi:

- calcolare la tabella dei numeri primi fino al valore N inserito, partendo per il setaccio dal valore 2;
- stampare la lista dei numeri primi individuati fino al valore N.

Definire il valore N utilizzando il costrutto `#define`.

Esercizio 8 – Esami

Scrivere un programma che definisca la struttura di un esame universitario e ne permetta la modifica dei dati da riga di comando.

Gli esami vengono rappresentati con:

- nome corso (di tipo stringa)
- cognome docente (di tipo stringa)
- numero di CFU (di tipo intero)
- voto (di tipo intero)

Per rappresentare in blocco i dati dell'esame si usi una struttura dati di tipo `struct`. Scrivere un `main()` che permetta l'inserimento da linea di comando di un elemento di tipo esame e che stampi poi a schermo il contenuto.

Esercizio 9 – Esami 2.0

Modificare il programma per permettere all'utente di inserire un numero prefissato di esami (es. 5) , stamparne i dati e calcolare la media pesata dei voti.

SUGGERIMENTO (valido per alcuni degli esercizi riportati sopra)

Per acquisire una stringa contenente spazi bianchi utilizzare la funzione `fgets()` specificando un buffer opportunamente grande (se si utilizza la funzione `scanf()` l'acquisizione della stringa termina al primo carattere spazio bianco).