FONDAMENTI DI INFORMATICA

Prof. Maristella Matera - A.A. 2024/2025

Laboratorio di Programmazione in C

Laboratorio 3

Es. 1 "Grande Giove!"

■ *Cicli, condizioni, printf() e scanf(), struct*

Il Dottor Emmett Brown di "Ritorno al Futuro" ha bisogno del vostro aiuto per sviluppare il software del "conta-chilometri temporale" della sua DeLorean.

Il programma riceve in input la data di partenza e la data di destinazione. Successivamente calcola e stampa a video la distanza tra le due date in anni, mesi, giorni, ore e minuti.

Esempio di funzionamento:

```
Inserisci data e ora di partenza della DeLorean (DD-MM-YYYY hh:mm):
09-02-1988 15:30

Inserisci data e ora di destinazione della DeLorean (DD-MM-YYYY hh:mm):
25-07-2024 19:33

Grande Giove!
Hai viaggiato nel tempo per 36 anni 5 mesi 16 giorni 4 ore e 3 minuti!
```

- le date vanno acquisite direttamente in formato `DD-MM-YYYY hh:mm` e salvate all'interno di una struttura dati di tipo `Data` definita dallo sviluppatore. La struttura deve essere adatta a contenere giorno, mese, anno, ora e minuto che compongono la singola data
- va verificata la validità delle date inserite: se almeno una delle date non è valida, allora il programma stampa a video un messaggio di errore e termina l'esecuzione
- ipotizziamo che la DeLorean possa viaggiare dal 1 gennaio 1800 al 31 dicembre 2100
- le date non devono essere necessariamente inserite in ordine cronologico, in quanto la DeLorean può viaggiare avanti e indietro nel tempo
- il programma deve tenere conto degli anni bisestili e del numero differente di giorni presenti in ciascun mese dell'anno
- non è richiesta l'implementazione del flusso canalizzatore

Suggerimento:

Per verificare la correttezza del risultato è possibile utilizzare questo calcolatore on-line: `https://www.timeanddate.com/date/timeduration.html`

Es. 2 Partitina a Tic-Tac-Toe?

■ *Cicli, condizioni, matrici*

Scrivere un programma in linguaggio C che simula una partita di Tic-Tac-Toe (tris) tra due giocatori.

Il gioco si svolge su una griglia quadrata 3x3. I giocatori si alternano inserendo il proprio simbolo (**X** per il primo giocatore e **0** per il secondo) in una delle celle vuote della griglia. Il programma deve:

- Visualizzare la griglia di gioco dopo ogni mossa.
- Consentire ai giocatori di inserire le loro mosse selezionando la riga e la colonna in cui vogliono inserire il loro simbolo.
- Controllare che la mossa sia valida (cioè che la cella selezionata sia vuota e che i valori di riga e colonna siano validi).
- Determinare se un giocatore ha vinto (cioè se ha completato una linea orizzontale, verticale o diagonale con il proprio simbolo) o se la partita è terminata in parità (cioè se tutte le celle della griglia sono state riempite senza che ci sia un vincitore).
- Visualizzare il risultato della partita, indicando se c'è un vincitore o se la partita è terminata in parità.

Es. 3 Aritmetica dei puntatori

■ *Puntatori, array*

In questo esercizio dovrete dimostrare le relazioni esistenti tra puntatori e array, utilizzando l'aritmetica dei puntatori per scorrere un array.

L'esercizio è da risolvere un passo dopo l'altro, secondo questa sequenza:

- **Passo 1** - dichiarare due variabili: un array di interi e un puntatore a intero a cui assegnare il nome dell'array. Dimostrare l'equivalenza tra l'indirizzo dell'array, l'indirizzo del primo elemento dell'array e il contenuto del puntatore stampandoli;
- **Passo 2** - mostrare come dereferenziando il puntatore si ottiene il valore del primo elemento dell'array;
- **Passo 3** - stampare l'array non accedendo al suo contenuto tramite gli indici, ma incrementando il valore del puntatore all'array o direttamente il nome dell'array avvalendosi dell'aritmetica dei puntatori;

Es. 4 Fusione di array ordinati con Bubble Sort

■ *Array, puntatori*

Scrivere un programma che:

- chiede due volte all'utente di inserire **N** valori interi, e li salva in due array di dimensione **N**;
- ordina ciascun array utilizzando il metodo Bubble Sort;
- effettua la fusione dei due array, in modo che l'array risultante mantenga l'ordinamento;
- stampa il vettore dopo la fusione.

Valutare inoltre la complessità dell'algoritmo di ordinamento contando il numero totale di confronti e il numero totale di swap eseguiti e stampare questi valori.

L'algoritmo di ordinamento Bubble Sort

L'algoritmo Bubble Sort confronta ogni coppia di elementi adiacenti del vettore. Se due elementi sono nell'ordine sbagliato, essi vengono invertiti (swap). Come risultato, ad ogni passo l'elemento più grande non ancora ordinato si sposta verso la posizione più alta dell'array. Al passo `p`, gli ultimi `p-1` elementi dell'array saranno nella loro posizione definitiva (non è necessario controllarli di nuovo). L'algoritmo continua a scorrere tutta la lista di elementi finché non vengono più eseguiti scambi, situazione che indica che la lista è ordinata (qui un'interpretazione visiva dell'algoritmo: `https://youtu.be/Iv3vgjM8Pv4?t=51`).

Es. 5 Copia dispari

■ *Cicli, matrici*

Scrivere un programma che, data una matrice `Mat1` di interi `NxN`, con `N` costante opportunamente definita, copia i soli elementi dispari in una nuova matrice `Mat2` della stessa dimensione, senza lasciare posizioni vuote intermedie.

Esempio con `N = 3`:

```
Mat1     Mat2
1 2 3    1 3 5
4 5 6    7 9 0
7 8 9    0 0 0
```

Successivamente, stampare a video la matrice `Mat2`, visualizzando gli elementi per righe.

Es. 6 Copia dispari 2.0

■ *Cicli, matrici, gestione dei file*

Modificare il programma **Copia dispari** in modo che la matrice `Mat1` di interi `NxN` (sempre con `N` costante opportunamente definita) venga letta da un file di testo chiamato `matrice.txt` scritto in formato CSV (Comma Separated Values):

Esempio di file in formato CSV:

```
1,2,3
4,5,6
7,8,9
```

Il programma, prima di procedere con la copia degli elementi dispari, verifica che la matrice definita nel file abbia le dimensioni richieste. Se la matrice non è di dimensioni `NxN` allora deve comparire un messaggio di errore a video e il programma viene terminato.

Se la matrice contenuta nel file ha le dimensioni e il formato corretti, procedere come nell'esercizio **Copia dispari**, stampare a video la matrice di partenza `Mat1` e scrivere in un file di testo `matrice_dispari.txt` la nuova matrice `Mat2`, rispettando il formato riportato sopra.

Es. 7 Esiti degli esami di Fondamenti di Informatica

■ *Gestione dei file, struct*

Creare una struttura dati di tipo `struct` adatta a rappresentare un singolo esito dell'esame di Fondamenti di Informatica.

Ogni esito è caratterizzato da:

- **cognome** dello studente (di tipo stringa)
- **matricola** dello studente (di tipo intero)
- **voto** assegnato (di tipo intero)

Scrivere un programma che richieda all'utente l'inserimento di un massimo di `N` esiti (con `N` definito dallo sviluppatore) e li salvi in un file binario chiamato `esiti.bin`.

Nel corso dell'inserimento, verificare che ogni matricola abbia un solo voto registrato.

Suggerimento

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).

Es. 8 Consultazione esiti degli esami di Fondamenti di Informatica

■ *Struct, gestione dei file*

Scrivere un programma che:

- legga il file `esiti.bin` creato nell'esercizio **"Esiti degli esami di Fondamenti di Informatica"** e lo memorizzi in un array (sapendo che questo avrà dimensione `N`)
- stampi a video il numero di esiti registrati nel file e l'elenco degli esiti sottoforma di tabella
- stampi a video la media dei voti, la mediana, il voto più basso e quello più alto
- permetta all'utente di cercare il voto di uno studente fornendo la matricola (tale funzionalità deve essere ripetuta fino a quando l'utente non inserisce `-1` come valore di matricola)