Fondamenti di Informatica

Andrea Gussoni
andrea1.gussoni at polimi.it

Politecnico di Milano

May 25, 2021

Table of Contents

1 Ripasso Esercitazione Precedente

Section 1

Ripasso Esercitazione Precedente

Ripasso

Problemi con esercizi della precedente esercitazione?

WC

- Nei sistemi Unix e derivati, il comando wc fornisce il numeri di caratteri, numero di parole, e numero di linee contenute in un file di testo.
- Implementare in linguaggio C una funzione che prende in input il nome di un file e ne restituisce le statistiche del comando wc.

Cifrario di Cesare

- In antichità, il cifrario di Cesare è stato uno dei più antichi algoritmi crittografici conosciuti, usato da Giulio Cesare per cifrare messaggi.
- Funziona tramite uno shift di 3 caratteri, per ogni lettera del messaggio originale. Ad esempio, la parola "CANE", diventa "FDQH".
- Per decifrare il messaggio, si può effettuare la procedura inversa ovviamente.

Movimenti Bancari

- Un file contenente le informazioni sui movimenti di un conto corrente bancario (in Euro) ha il seguente formato.
- La prima riga contiene una stringa con l'IBAN del conto corrente.
- La seconda riga contiene il saldo iniziale (sempre positivo).
- Le righe seguenti formano una sequenza di movimenti.
- Ogni movimento è descritto dalla data del movimento nel formato aaaa-mm-gg, l'importo del movimento espresso in Euro (da rappresentare con un numero `float`, dato che può contenere cifre decimali) e un carattere che può essere il segno + (entrata) oppure - (uscita). Ciascun movimento è posto su di una singola riga del file, con un carattere di spazio, usato come separatore dei valori sulla riga. I movimenti sono riportati nel file in ordine cronologico crescente: non esistono due movimenti effettuati nel medesimo giorno.

Movimenti Bancari

- Scrivere la funzione:

```
void dataLimite(char nomefile[],
  float limCredito,
  char *dataRestituita);
```

- che prende come parametri il nome di un file contenente le informazioni di un conto corrente nel formato sopra descritto e un limite di credito *limCredito* (con $limCredito < 0$), e restituisce tramite l'uso del terzo parametro $char * dataRestituita$ la data in cui il saldo è sceso sotto il valore limite passato come parametro.

Movimenti Bancari

- Dato il tipo di dato seguente:

```
typedef struct mov {
        char data[10+1];
        float importo;
        char segno;
} t_movimento;
```

- scrivere in C la funzione:

```
void leggiEstratto(char nomefile[],
        t_movimento *movimenti,
        int *lung,
        int *saldoIniziale)
```

che prende come parametri il nome di un file contenente le informazioni di un conto corrente nel formato descritto nel precedente esercizio e un puntatore ad un array in cui inseriremo tutti i movimenti presenti all'interno del file.

Movimenti Bancari

- L'array avrà quindi tante celle di tipo `t_movimento` quante sono le righe del file che descrivono un movimento. Il secondo e il terzo parametro sono da intendersi come parametri passati per indirizzo per restituire al chiamante la lunghezza del vettore e il saldo iniziale del conto corrente.

Movimenti Bancari

- Si considerino le ulteriori definizioni di tipo di dato:

```
typedef struct t_entrateUscite{
        float totEntrate;
        float totUscite;
} t_bilancio;
typedef t_bilancio bilancioMensile[12];
```

- Scrivere in C il sottoprogramma:

```
void calcolaBilanciMensili(t_movimento *elenco,
        int lung,
        int anno,
        t_bilancio ris[12]);
```

che prende come parametri un array contenente l'elenco dei movimenti di un conto corrente, la lunghezza dell'array e un valore intero indicante un anno (per esempio: 2019).

Movimenti Bancari

- Il sottoprogramma riempie l'array passato come quarto parametro memorizzando nella 1ma cella i totali delle entrate e delle uscite relative al mese di Gennaio, nella seconda cella i totali delle entrate e delle uscite relative al mese di Febbraio ecc.
- Nota: potrebbe essere utile definire e usare due sottoprogrammi:

```
int numAnno(char data[]); int numMese(char data[]);
```

per ottenere dalla stringa con formato aaaa-mm-gg, passata come parametro, un numero con il valore dell'anno e un numero con il valore del mese, rispettivamente. Si assuma 1=Gennaio, 2=Febbraio, ecc.

Prodotto interno lordo

- il prodotto interno lordo (pil) è una delle grandezze macro-economiche fondamentali e rappresenta il valore di mercato di tutti i beni e servizi finali prodotti in una nazione, in un dato periodo di tempo. per esempio, la farina è un "bene finale" se venduta come farina; un "bene intermedio" se venduta al panettiere per fare il pane; in questo caso il valore della farina è incorporato nel valore del pane.
- limitandosi a considerare solo beni di consumo (trascurando quindi i beni e i servizi di investimento), si considerino due file di testo, chiamati rispettivamente `benifinali.txt` e `materiali.txt`.

Prodotto interno lordo

- Il file BeniFinali.txt include un elenco di righi, dove ciascun rigo riporta una stringa alfanumerica (univoca) di dieci caratteri per identificare un bene (ID_bene), una stringa senza spazi all'interno e lunga al massimo cinquanta caratteri contentente il nome del bene (nome), e un prezzo di vendita unitario espresso in euro (prezzo_unitario), separati da un carattere di spazio.
- Il file Materiali.txt include un elenco di righi, dove ciascun rigo riporta una stringa alfanumerica lunga dieci caratteri per identificare un bene finale, una stringa alfanumerica lunga dieci caratteri per identificare un bene intermedio e la quantità necessaria di prodotto intermedio (indicata come un numero frazionario), per la produzione del primo prodotto.

Prodotto interno lordo

- Dichiarare in C due tipi di dato, chiamati bene_finale_t e materiale_t, per rappresentare in memoria ciascun un rigo del primo e del secondo file, rispettivamente.

Prodotto interno lordo

- Si consideri la seguente definizione di tipo di dato:

```
typedef struct {
  bene_finale_t b;
  float quantita;
} vendita_t;
```

- Scrivere in C il codice della funzione int main()... che apra il file BeniFinali.txt, conteggi il numero complessivo di beni di consumo da considerare per il calcolo del PIL (cioè calcoli il numero di righi nel file) e dichiarando un vettore di tipo vendita_t di una lunghezza massima prefissata (superiore sicuramente al numero di beni descritti nel file BeniFinali.txt), lo inizializzi acquisendo da tastiera la quantità di prodotto venduta per ciascun bene.

Prodotto interno lordo

- Scrivere in C la funzione:

```
float calcola_valore_aggiunto(char ID[],
        char nomeFileMateriali[],
        vendita_t elenco[],
        int lungElenco);
```

  avente come argomenti l'identificativo di un bene, una stringa contenente il nome del file con le informazioni sui materiali necessari per produrlo (quindi un file di testo organizzato come Materiali.txt), un array di tipo vendita_t e la lunghezza di quest'ultimo.

- La funzione restituisce un numero frazionario calcolato come la differenza tra il prezzo di vendita del bene (con identificativo passato come parametro) meno i prezzi delle quantità di materiale impiegate per produrlo. La funzione restituisce il valore 0.0 nel caso in cui l'identificativo passato come parametro non corrisponda ad alcun bene.

Prodotto interno lordo

- Scrivere in C la funzione:

  `...calcola_PIL_consumi(...)`

  avente come parametri un array di tipo vendita_t e la sua lunghezza, e che restituisca il PIL dei consumi, calcolato come la somma dei "valori aggiunti" dei beni presenti nell'array passato come parametro e tenendo conto delle quantità dei beni in esso riportate.