



7. Scomposizione dei programmi con le funzioni

L'uso delle funzioni rende il programma complessivamente molto leggibile anche a distanza di tempo e realizza nella pratica il metodo dei raffinamenti successivi (*top-down*). È quindi consigliabile organizzare le diverse parti di un programma come funzioni, anche se vengono utilizzate una sola volta, perché blocchi più piccoli di programma sono più facili da leggere, da comprendere e da correggere.

I programmi risultano così più documentati e maggiormente controllabili.

PROGETTO • Si inseriscono da tastiera i dati riguardanti gli articoli venduti ai clienti di un'azienda: il codice cliente, la data, il codice articolo, il prezzo unitario e la quantità. Supponendo che i dati vengano introdotti ordinati per codice cliente, si vogliono produrre in stampa i dati, con il totale degli articoli e il totale degli importi per ogni cliente.

Si suppone che il cliente venga identificato attraverso un codice e che gli articoli venduti vengano introdotti in modo ordinato secondo il codice del cliente.

Per ogni articolo si inseriscono da tastiera: il codice cliente, la data, il codice articolo, il prezzo unitario e la quantità.

Questi dati vengono stampati in un prospetto: alla fine di ogni gruppo di articoli, che si riferiscono allo stesso cliente, si vuole stampare il totale degli articoli venduti e il totale degli importi.

La stampa della parte finale di ogni cliente deve essere fatta quando viene introdotto un codice cliente diverso dal codice cliente precedente (avendo supposto che gli articoli venduti siano in ordine di codice cliente): quando questo si verifica si ha una situazione di **rottura di codice**, che provoca la stampa dei dati finali del cliente precedente e il passaggio al nuovo cliente.

Per questo motivo un procedimento di questo tipo si chiama anche **stampa con livelli di totalizzazione parziale**.

Il controllo della rottura di codice viene risolto utilizzando una variabile di appoggio nella quale si memorizza il codice del cliente: quando il codice introdotto da tastiera è diverso dal codice contenuto nella variabile di appoggio si è in presenza di una situazione di rottura di codice.

La fine dell'inserimento dei dati da tastiera può essere segnalata introducendo un asterisco * alla richiesta del codice cliente.

PROGRAMMA C++

```
// Clienti.cpp : vendite con rottura di codice
#include <iostream>
#include <string>
#include <iomanip>
using namespace std;

// variabili globali: dati di una vendita
string giorno;           // data vendita
string CodiceArticolo;  // codice articolo
float quantita;          // quantità
float PrezzoUnitario;   // prezzo
```



```

// prototipi delle funzioni
void StampaIntestazione (string descriz, string codice);
void Vendita();
float Calcolo (float unit, float quanti);
void StampaRiga (string descriz, float valore);

// funzione principale
int main()
{
    // input
    string CodiceCliente;    // codice cliente
    // output
    float PrezzoTotale;      // prezzo totale
    float QuantitaTotale;    // quantità totale del cliente
    float TotaleImporti;     // importo totale del cliente
    /* lavoro */
    string appo;             // variabile appoggio

    cerr << "Codice cliente (* = fine): ";
    cin >> CodiceCliente;
    while (CodiceCliente != "*") {
        appo = CodiceCliente;
        TotaleImporti = 0;
        QuantitaTotale = 0;
        StampaIntestazione("Codice cliente", CodiceCliente);

        // articoli venduti al cliente
        while (CodiceCliente == appo && CodiceCliente != "*") {
            Vendita();
            PrezzoTotale = Calcolo(PrezzoUnitario, quantita);
            QuantitaTotale += quantita;
            TotaleImporti += PrezzoTotale;

            // stampa dati di un articolo
            cout << setw(9) << giorno
                 << setw(5) << CodiceArticolo
                 << setw(10) << quantita
                 << setw(12) << PrezzoUnitario
                 << setw(10) << PrezzoTotale << endl;

            // altra vendita
            cerr << "Codice cliente (* = fine): ";
            cin >> CodiceCliente;
        }

        // parte finale di un cliente
        StampaRiga("Quantita' totale", QuantitaTotale);
        StampaRiga("Importo totale del cliente", TotaleImporti);
    }

    return 0;
}

void StampaIntestazione (string descriz, string codice)
{
    cout << endl;
    cout << setw(30) << descriz
         << setw(10) << codice
         << endl;
    cout << endl;
} // StampaIntestazione

```



```

void Vendita()
{
    cerr << "Data          ";
    cin >> giorno;
    cerr << "Codice articolo ";
    cin >> CodiceArticolo;
    cerr << "Quantita' venduta ";
    cin >> quantita;
    cerr << "Prezzo unitario   ";
    cin >> PrezzoUnitario;
} // Vendita

float Calcolo (float unit, float quanti)
{
    float totale;
    totale = unit * quanti;
    return totale;
} // Calcolo

void StampaRiga (string descriz, float valore)
{
    cout << endl;
    cout << setw(30) << descriz
         << setw(10) << valore
         << endl;
} // StampaRiga

```

Le variabili utilizzate sia nella funzione *main* sia nelle altre funzioni sono dichiarate all'inizio del programma, dopo le direttive *#include*, come *variabili globali*.

All'inizio del programma sono descritti i *prototipi delle funzioni* presenti nel programma: si deve fare attenzione all'ordine di scrittura, perché le funzioni chiamate da altre funzioni devono essere dichiarate prima.

Il programma precedente utilizza il flusso **cerr** (*standard error*) per i messaggi da inviare al video come supporto dell'input dell'utente, e il flusso **cout** (*standard output*) per l'output dei risultati dell'elaborazione.

In questo modo, in fase di testing del programma, o nel caso non fosse disponibile una stampante, è possibile effettuare una **ridirezione** dell'output dalla linea comandi:

```
C:\esercizi>Clienti > vendite.txt
```

L'output del programma, normalmente diretto al video (*cout*), viene deviato sul file del disco di nome *vendite*, che può essere successivamente visualizzato (o stampato) con un editor di testi (per esempio, *Blocco Note* di Windows).

Avendo a disposizione una stampante, la ridirezione può essere fatta alla **stampante** parallela collegata al computer, scrivendo dalla linea comandi:

```
C:\esercizi>Clienti >lpt1:
```

Il nome riservato **lpt1** (o *prn*) indica infatti la stampante parallela collegata al computer.



ESERCIZI

1. Dato un elenco di docenti con cognome e materia di insegnamento, supponendo che vengano inseriti in ordine di materia, contare quanti sono i docenti per ogni materia visualizzandone il numero ogni volta che cambia la materia.
2. Le informazioni sugli studenti partecipanti alle gare di atletica leggera sono inseriti in ordine di tipo di gara (corsa, salto in alto, lancio del peso, ecc.). Ogni volta che cambia il tipo di gara, si deve visualizzare il numero degli studenti partecipanti per ciascuna gara.
3. I movimenti effettuati sugli articoli di un magazzino sono inseriti, in ordine di codice articolo, con codice articolo, data e quantità. La quantità può essere positiva se il movimento è un carico, oppure negativo se il movimento è uno scarico. Ogni volta che cambia il codice dell'articolo (rottura di codice) si deve stampare la giacenza calcolata con la differenza tra la somma delle quantità positive e la somma delle quantità negative.