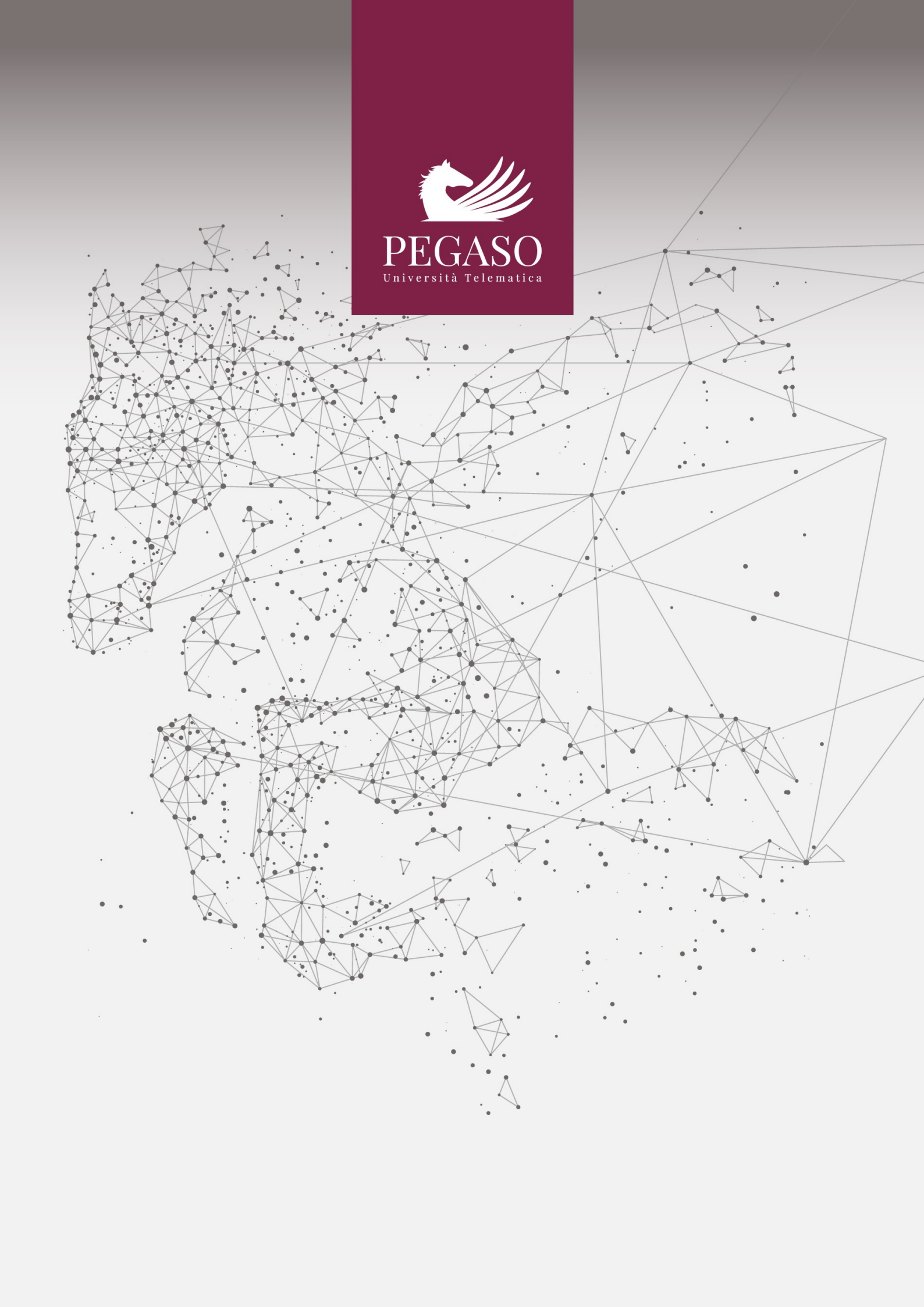
PEGASO
Università Telematica

Indice

1. ITERAZIONI CONTROLLATE DA SENTINELLA 3
2. AFFINAMENTO GRADUALE TOP-DOWN 4
3. PROGRAMMA IN C PER CONTROLLO CON SENTINELLA 9
BIBLIOGRAFIA 12

1. Iterazioni controllate da sentinella

Generalizziamo il problema del calcolo della media di una classe.

Consideriamo il seguente problema:

Sviluppare un programma per il calcolo della media di una classe in grado di elaborare un numero arbitrario di voti ogni volta che è in esecuzione.

Nel primo esempio del calcolo della media di una classe il numero dei voti (10) si sapeva in anticipo.

In questo esempio non si dà alcuna indicazione su quanti voti verranno inseriti.

Il programma deve elaborare un numero arbitrario di voti.

Come può il programma stabilire quando fermare l'inserimento dei voti?

Come saprà quando calcolare e stampare la media della classe?

Un modo per risolvere questo problema è quello di usare un valore speciale, chiamato valore sentinella (o anche valore di segnale, valore fittizio o valore di flag), per indicare la "fine dell'ingresso dei dati".

L'utente scrive i voti finché non sono stati inseriti tutti i voti validi, poi scrive il valore sentinella per indicare che l'ultimo voto è stato inserito.

L'iterazione controllata da sentinella è spesso chiamata *iterazione indefinita*, perché il numero di iterazioni non è noto prima che il ciclo inizi l'esecuzione.

Chiaramente, il valore sentinella deve essere scelto in modo tale che non possa essere confuso con un valore di input accettabile.

Poiché i voti su un quiz sono normalmente numeri interi non negativi, –1 è un valore sentinella accettabile per questo problema.

In questo modo, un'esecuzione del programma della media di una classe potrebbe elaborare una sequenza di input come 95, 96, 75, 74, 89 e –1.

Il programma poi calcolerebbe e stamperebbe la media della classe per i voti 95,

96, 75, 74 e 89 (–1 è il valore sentinella, per cui non deve entrare nel calcolo della media).

2. Affinamento graduale top-down

Affrontiamo il programma della media di una classe con una tecnica chiamata affinamento graduale top-down, una tecnica essenziale allo sviluppo di programmi ben strutturati.

Iniziamo con una rappresentazione dello pseudocodice a livello top:

Determina la media della classe per il quiz

Il livello top è un'istruzione singola che rappresenta la funzione complessiva del programma.

Come tale il livello top è, in effetti, una rappresentazione completa di un programma. Purtroppo, il livello top raramente contiene una quantità sufficiente di dettagli per scrivere il programma in C.

Così iniziamo ora il processo di affinamento.

Suddividiamo il livello top in una serie di compiti più piccoli e li elenchiamo nell'ordine in cui devono essere eseguiti.

Ciò risulta nel seguente **primo affinamento**.

Inizializza le variabili

Ricevi in ingresso, somma e conta i voti del quiz

Calcola e stampa la media della classe

Qui è stata usata solo la struttura sequenziale (i passi elencati vanno eseguiti in ordine, uno dopo l'altro).

Osservazione di ingegneria del software: ogni affinamento, come pure lo stesso livello top, è una specificazione completa dell'algoritmo; varia soltanto il livello di dettaglio.

Secondo affinamento

Per procedere al successivo livello di affinamento, cioè al secondo affinamento, useremo alcune variabili specifiche. Avremo bisogno di un totale corrente dei numeri, di un conteggio di quanti numeri sono stati elaborati, di una variabile che riceve il valore di ogni singolo voto quando è inserito e di una variabile per contenere la media calcolata.

L'istruzione in pseudocodice.

Inizializza le variabili

può essere affinata come segue:

Inizializza il totale a zero

Inizializza il contatore a zero

Si noti che vanno inizializzati soltanto il totale e il contatore; le variabili media e voto (rispettivamente, per la media calcolata e l'input dell'utente) non necessitano di essere inizializzate, perché i loro valori saranno calcolati e inseriti dall'utente, rispettivamente.

L'istruzione in pseudocodice

Ricevi in ingresso, somma e conta i voti del quiz

richiede una struttura di iterazione che di volta in volta legge ogni voto. Poiché non sappiamo in anticipo quanti voti ci sono da elaborare, useremo l'iterazione controllata da sentinella.

L'utente inserirà i voti validi uno alla volta.

Dopo aver scritto l'ultimo voto valido, l'utente scriverà il valore sentinella.

Il programma farà il test per questo valore dopo che sarà inserito ogni voto e terminerà il ciclo quando sarà inserita la sentinella.

L'affinamento della precedente istruzione in pseudocodice è allora

Ricevi in ingresso il primo voto (eventualmente la sentinella)
Finché l'utente non ha ancora inserito la sentinella
Aggiungi questo voto al totale corrente
Aggiungi uno al contatore dei voti
Ricevi in ingresso il voto successivo (eventualmente la sentinella)

Si noti che nello pseudocodice non usiamo parentesi graffe intorno all'insieme di istruzioni che forma il corpo dell'istruzione while, ma semplicemente indentiamo tutte queste istruzioni sotto il while per mostrare che tutte quante appartengono al while.

Di nuovo, lo pseudocodice è un ausilio per lo sviluppo informale di un programma.

L'istruzione in pseudocodice

Calcola e stampa la media della classe

si può affinare come segue:

Se il contatore non è uguale a zero
Poni la media uguale al totale diviso per il contatore
Stampa la media
altrimenti
Stampa "No grades were entered"

Si noti che qui siamo stati attenti a verificare la possibilità della divisione per zero, un errore irreversibile che, se non viene scoperto, provoca l'arresto del programma (in questo caso detto "crashing", cioè "arresto anomalo").

Il secondo affinamento completo è mostrato nella Figura 3.7.

☹*Errore comune di programmazione*

Un tentativo di dividere per zero provoca un errore irreversibile.

☺ *Buona pratica di programmazione*

Quando si esegue una divisione con un'espressione il cui valore potrebbe essere zero, controllate esplicitamente questo caso e trattatelo adeguatamente nel vostro programma (come con la stampa di un messaggio di errore) piuttosto che permettere che si verifichi l'errore irreversibile.

Includiamo anche alcune righe completamente vuote nello pseudocodice per migliorarne la leggibilità.

In realtà, le righe vuote separano questi programmi nelle loro varie fasi.

```
1  Inizializza il totale a zero
2  Inizializza il contatore a zero
3
4  Ricevi in ingresso il primo voto (eventualmente la sentinella)
5  Finché l'utente non ha ancora inserito la sentinella
6  Aggiungi il voto al totale corrente
7  Aggiungi uno al contatore dei voti
8  Ricevi in ingresso il voto successivo (eventualmente la sentinella)
9
```

10 Se il contatore non è uguale a zero

11 Poni la media uguale al totale diviso per il contatore

12 Stampa la media

13 Altrimenti

14 Stampa "No grades were entered"

L'algoritmo in pseudocodice risolve il problema del calcolo della media di una classe nella sua forma più generale. Questo algoritmo è stato sviluppato dopo due soli livelli di affinamento. Talvolta sono necessari più livelli.

Osservazione di ingegneria del software

Molti programmi possono essere divisi logicamente in tre fasi: una fase di inizializzazione, che inizializza le variabili del programma; una fase di elaborazione, che riceve in ingresso i valori dei dati e modifica conseguentemente le variabili del programma e una fase di chiusura, che calcola e stampa i risultati finali.

Osservazione di ingegneria del software

Il processo di affinamento graduale top-down termina quando l'algoritmo in pseudocodice è specificato con sufficiente dettaglio, tale da poter convertire lo pseudocodice direttamente in C.

A quel punto, implementare il programma in C è normalmente molto semplice.

3. Programma in C per controllo con sentinella

Il programma in C e un esempio di esecuzione sono mostrati di seguito.

Sebbene vengano inseriti solo voti interi, è probabile che il calcolo della media produca un numero con un punto decimale (che sostituisce la virgola nella notazione del C).

Il tipo int non può rappresentare un tale numero.

Il programma introduce il tipo di dati float per trattare i numeri con punto decimale (chiamati numeri in virgola mobile) e introduce un operatore speciale chiamato operatore cast per trattare il calcolo della media.

```
01    // Fig. 3.8: fig03_08.c
02     // Media di una classe con iterazione controllata da sentinella.
03     #include <stdio.h>
04
05     // la funzione main inizia l'esecuzione del programma
06     int main( void )
07     {
08     unsigned int counter; // numero di voti letti
09     int grade; // valore del voto
10     int total; // somma dei voti
11
12     int average; // numero con punto decimale per la media
13
14     // fase di inizializzazione
15     total = 0; // inizializza il totale
16     counter = 0; // inizializza il contatore del ciclo
17
18       // fase di elaborazione
19       // ricevi il primo voto dall'utente
```

```
20       printf( "%s", "Enter grade, -1 to end: " ); // prompt per l'input
21        scanf( "%d", &grade ); // leggi il voto dall'utente
22
23        // ripeti finche' non viene letto il valore sentinella
24        while ( grade != -1 ) {
25        total = total + grade; // aggiungi il voto al totale
26        counter = counter + 1; // incrementa il contatore
27
28        // ricevi il voto successivo dall'utente
29        printf( "%s", "Enter grade, -1 to end: " ); // prompt
30        scanf("%d", &grade); // leggi il prossimo voto
31        } // fine di while
32
33       // fase di chiusura
34       // se l'utente ha inserito almeno un voto
35       if ( counter != 0 ) {
36
37        // calcola la media di tutti i voti inseriti
38        average = total / counter; // divisione intera
39
40        // stampa la media con la precisione di due cifre
41        printf( "Class average is d\n", average );
42        } // fine di if
43       else { //  se non sono stati inseriti voti, stampa un messaggio
44       puts( "No grades were entered" );
45     } // fine di else
46  } // fine della funzione main
```

```
Enter grade, -1 to end: 75
Enter grade, -1 to end: 94
Enter grade, -1 to end: 97
Enter grade, -1 to end: 88
Enter grade, -1 to end: 70
Enter grade, -1 to end: 64
Enter grade, -1 to end: 83
Enter grade, -1 to end: 89
Enter grade, -1 to end: -1
Class average is 82
Enter grade, -1 to end: -1
No grades were entered
```

Si noti l'istruzione composta nel ciclo while (riga 24).

Ancora una volta, le parentesi graffe sono necessarie per assicurare che tutte e quattro le istruzioni siano eseguite all'interno del ciclo.

Senza le parentesi graffe le ultime tre istruzioni nel corpo del ciclo cadrebbero al di fuori di esso, facendo sì che il computer interpreti scorrettamente questo codice come segue.

```
while ( grade != -1 )
   total = total + grade; // aggiungi il voto al totale
counter = counter + 1; // incrementa il contatore
printf( "%s", "Enter grade, -1 to end: " ); // prompt per l'input
scanf( "%d", &grade ); // leggi il prossimo voto
```

Questo provocherebbe un ciclo infinito se l'utente non inserisse –1 per il primo voto.

Bibliografia

- Paul Deitel, Harvey Deitel, "Il linguaggio C – Fondamenti e tecniche di programmazione", Libro edito da Pearson Italia. Include anche utili esercizi di autovalutazione.