

Fondamenti di Informatica T-1

modulo 2

Laboratorio 02

1

Obiettivi di questa esercitazione

1. Valutazione in cortocircuito
2. If innestati
3. Switch
4. Cicli
5. Funzioni e Header File

2

Valutazione in cortocircuito (1)

In C, le **espressioni booleane sono valutate in cortocircuito** (short circuit evaluation)

- Appena è possibile determinare il valore logico dell'espressione, **si salta la valutazione degli altri operandi**

Esempio:

- lettura ciclica di due numeri reali
- si esce dal ciclo quando è impossibile effettuare la divisione di essi, oppure il risultato della divisione è negativo

3

Valutazione in cortocircuito (2)

```
float a, b;
do
{
    printf("Inserisci due numeri reali:");
    scanf("%f %f", &a, &b);
}
while(b!=0 && a/b<0);
```

Senza la valutazione in cortocircuito, in caso di divisore==0 si effettuerebbe comunque la divisione (errore concettuale)

4

Valutazione in cortocircuito (3)

```
int main()
{
    int a=5, b=5, c=5;
    if (a>0 || (a=a+1) )
        printf("%d", a);
    if (b>0 && (b=b+1) )
        printf("%d", b);
    if (c>0 && (c=c-5) )
        printf("%d", c);
}
```

Che cosa viene stampato a video? Quanto valgono le variabili?

5

Esercizio 1

Esercizio 1 – Triangoli

- Si scriva un programma, in linguaggio C, che legga da input le **lunghezze dei lati di un triangolo** e determini se il triangolo è equilatero, isoscele o scaleno
- Suggerimento: utilizzare if innestati

6

Esercizio 2

Esercizio 2 - Operazioni aritmetiche

- Realizzare un programma che, presi in input 2 operandi reali e un operatore (+, -, *, /), esegue l'operazione stampandone il risultato
- Nel caso in cui l'operatore sia errato o l'operazione non eseguibile, si stampi "undefined"
- Suggerimento: utilizzare il costrutto switch
 - il comportamento del programma è determinato in base al valore del carattere "operatore"

7

Esempio su cicli iterativi

Esempio - Sequenze di '0' e '1'

- Realizzare un programma che prende in input una sequenza di caratteri '0' e '1' e **conta la lunghezza della più lunga sotto-sequenza di '0' di fila**
- L'inserimento della sequenza **termina** quando si inserisce un **carattere diverso da '0' e '1'**
- A quel punto, si stampa a video il valore trovato

8

Esempio - Soluzione

```
char bit;
int cont = 0, maxlung = 0;
printf("Inserisci la sequenza\n");
do
{
    bit = getchar();
    getchar(); //consumo il newline
    if (bit == '0')
    {
        cont++;
        if(cont > maxlung)
            maxlung = cont;
    }
    else
    {
        cont = 0;
    }
}
while(bit == '0' || bit == '1');
printf("Lunghezza massima sotto-sequenza di 0: %d\n", maxlung);
```

9

Esercizio 3

Esercizio 3 - sequenze di cifre

- Realizzare un programma che prenda in input una **sequenza di cifre (tra 1 e 9)** e calcoli la **somma massima fra le sotto-sequenze di cifre non decrescenti**
- Il programma termina quando viene inserito lo 0

Esempio:

2	2	4	5	3	9	3	1	5	0
	13			12		3		6	

10

Esercizio 3 - Considerazioni

- Ci mettiamo nell'ipotesi di non dover controllare la correttezza del carattere letto
- Come effettuo il passaggio **carattere-cifra numerica**?
 - Quanto fa '8' - '0' (considerato come intero)?
- Di che valori devo tenere traccia?
 - Devo accorgermi di quando il valore appena letto è più piccolo del precedente
 - In tal caso, devo confrontare la somma corrente con quella massima, e ripartire con una nuova somma

11

Esempio su funzioni

Esempio 2 - calcolo del logaritmo in base qualunque

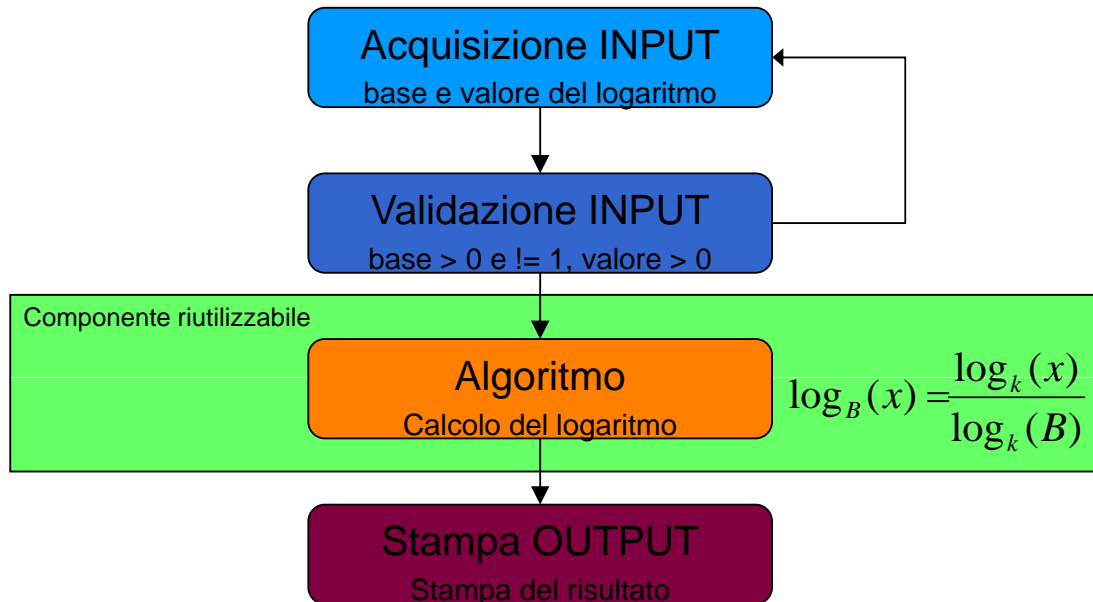
- Incapsulare la logica di calcolo in una funzione
 - PASSO 1: definisco la dichiarazione della funzione (nome, parametri di input e di output)
float mylog(float base, float value)
 - PASSO 2: realizzo la funzione

$$\log_B(x) = \frac{\log_k(x)}{\log_k(B)}$$

12

Funzioni (1)

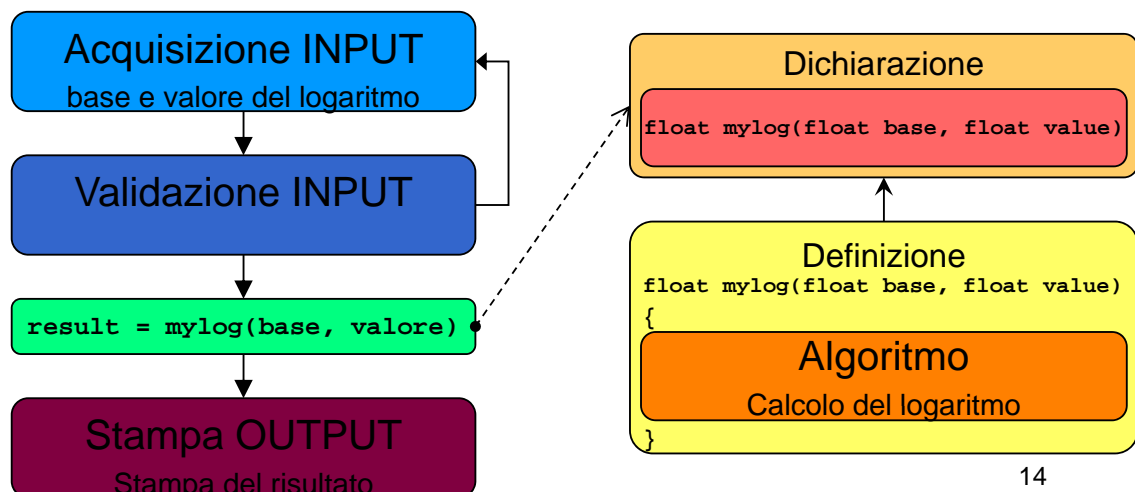
Calcolo del logaritmo in base qualunque - schema di soluzione



13

Funzioni (2)

- Logaritmo come componente: uso di una *funzione*!
 - PASSO 1: **dichiarazione della funzione (nome, parametri di input, parametri di output)**
`float mylog(float base, float value)`
 - PASSO 2: **definizione della funzione (ovvero implementazione)**



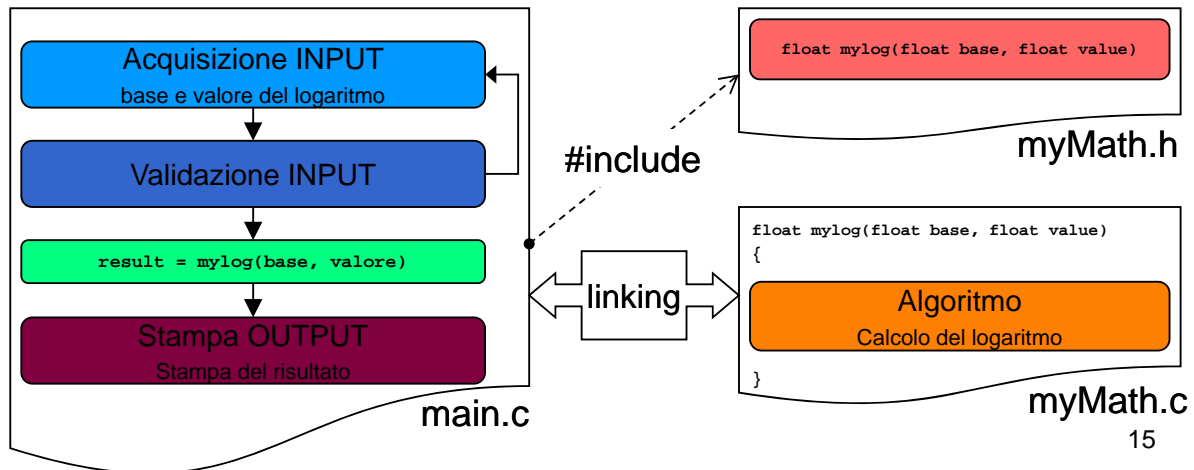
14

Funzioni (3)

Vogliamo rendere la funzione **mylog** davvero utilizzabile da più utenti in più programmi: creazione di un **modulo apposito**

- header file contenente le dichiarazioni (ad es. "myMath.h")
- file C contenente le definizioni (ad es. "myMath.c")
- includiamo "myMath.h" nel modulo che contiene la funzione main
- compiliamo con l'istruzione opportuna:

```
cl myProg.c myMath.c /I myMath.h /o myProg.exe
```



Esempio - Soluzione (1)

myMath.h:

```
float mylog(float base, float value);
```

myMath.c:

```
#include <math.h>
```

```
float mylog(float base, float value)
{
    return log(value) / log(base);
}
```

In realtà, dovrebbe contenere anche la validazione dei dati in input (*mai fidarsi del cliente!*) e restituire **errore** in caso di input non corretto

Esempio 3 - Soluzione (2)

main.c:

```
#include "myMath.h"
#include <stdio.h>

int main ()
{
    double b, x, result;
    ...
    result = mylog(b, x);
    ...
    return 0;
}
```

17

Esercizio 4

Esercizio 4 - ciclo per il calcolo del massimo e del minimo

- Realizzare un programma che calcoli il minimo e il massimo di una serie di valori
- Il numero di valori deve essere costante e definito tramite una opportuna *costante simbolica*
- Se la differenza tra il massimo e il minimo supera 10, il programma termina, altrimenti aspetta una nuova serie di valori
- Incapsulare ***il calcolo del minimo e del massimo in una funzione***

18

Esercizio 4 - Suggerimenti

- Quanti cicli? 2
 - Uno esterno per capire se uscire dal programma o richiedere la serie di valori
 - Uno interno per acquisire i K valori
- Che tipo di cicli?
 - Ciclo esterno: verifica una condizione a posteriori
→ do...while
 - Ciclo interno: conosce a priori il numero di iterazioni
→ for
- Di quanti valori devo tener traccia?
 - Ricordarsi che il minimo ed il massimo si possono calcolare passo passo
- Quando devo re-inizializzare il massimo ed il minimo?

19

Esercizio 5

Esercizio 5 - Calcolo del mcm tra numeri interi

- Realizzare un programma che prenda in input una serie di numeri interi, calcolando via via il minimo comune multiplo tra essi; il programma deve terminare quando mcm diventa più grande di 100
 - Ricordarsi che, come per il massimo e il minimo, anche mcm si può calcolare in modo parziale
 - Quindi basta utilizzare, per il calcolo, il valore di mcm calcolato al passo precedente e il numero inserito al passo corrente

$$\text{mcm}(a, b, c) = \text{mcm}(\text{mcm}(a, b), c)$$

20

Esercizio 5 - Requisiti e suggerimenti (1)

- Utilizzare la relazione $mcm(a,b) = \frac{a \cdot b}{MCD(a,b)}$
- Utilizzare l'algoritmo di Euclide per il MCD tra due numeri
 - Finché $M \neq N$:
 - se $M > N$, sostituisci a M il valore $M' = M - N$
 - altrimenti sostituisci a N il valore $N' = N - M$
 - MCD è il valore finale ottenuto quando M e N diventano uguali
- Incapsulare il calcolo di mcm e MCD in due funzioni
 - Ragionare per astrazione!
 - Individuare prima come le funzioni vengono viste dall'esterno (dichiarazione), poi realizzarle

21

Esercizio 5 - Requisiti e suggerimenti (2)

- Procedere per passi
 - Prima definiamo la funzione per MCD e proviamo a testarla su due valori
 - Poi definiamo la funzione per mcm e proviamo a testarla su due valori
 - Poi realizziamo il programma ciclico
- Per ultimo, utilizziamo l'approccio a moduli inserendo il calcolo di MCD e mcm in un modulo di libreria
 - *Header File contenente le dichiarazioni delle funzioni*

22

Esercizio 5 - Esempio di esecuzione

- Esempio di esecuzione

Inserisci il primo valore: 4

Inserisci un valore: 8

mcm corrente: 8

Inserisci un valore: 12

mcm corrente: 24

Inserisci un valore: 10

mcm corrente: 120

23

Esercizio 6

Esercizio 6 - Triangolo di Tartaglia

- Realizzare un programma che, letto in input il massimo livello voluto, mostri a video il contenuto del triangolo di Tartaglia fino a quel livello
- Per la costruzione del triangolo di Tartaglia, si utilizzi la corrispondenza tra i suoi elementi e i coefficienti binomiali

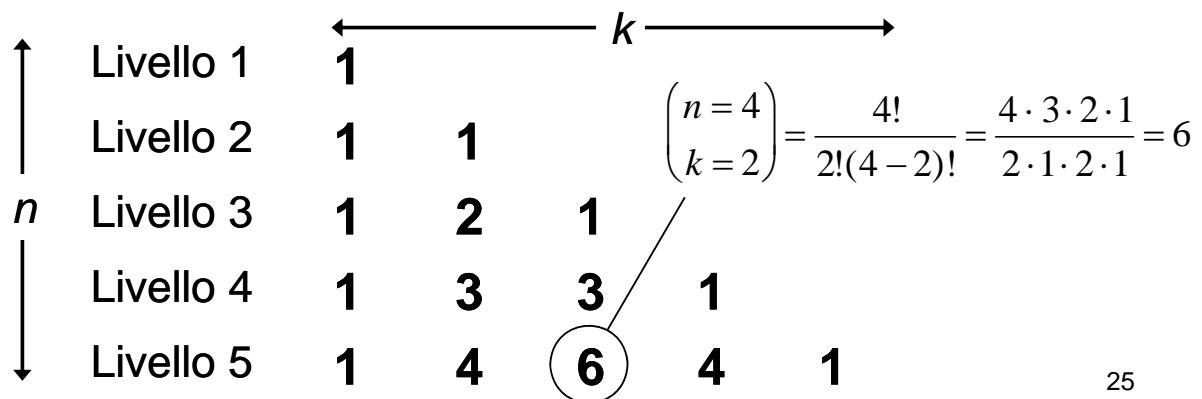
24

Esercizio 6 - Analisi del problema

■ Coefficiente binomiale $\binom{n}{k} = \frac{n!}{k!(n-k)!}$

■ Triangolo di Tartaglia (5 livelli)

- Allineato a sinistra per semplicità di stampa



25

Esercizio 6 - Requisiti

Organizzare il programma in due moduli separati

- I modulo (di libreria)
 - Funzione che calcola il fattoriale
 - Fattoriale di 0 = 1
 - Fattoriale di N = prodotto dei numeri da 1 a N
 - Funzione che calcola il coefficiente binomiale
 - A partire dalla funzione che calcola il fattoriale
 - Prima header file
- Il modulo (main)
 - Acquisizione in input del numero dei livelli
 - Stampa del triangolo di Tartaglia
 - Come utilizzare i cicli? Quanti cicli sono? Che tipo di cicli?
 - Ricordarsi che il coefficiente binomiale è definito solo per $k \leq n$

26