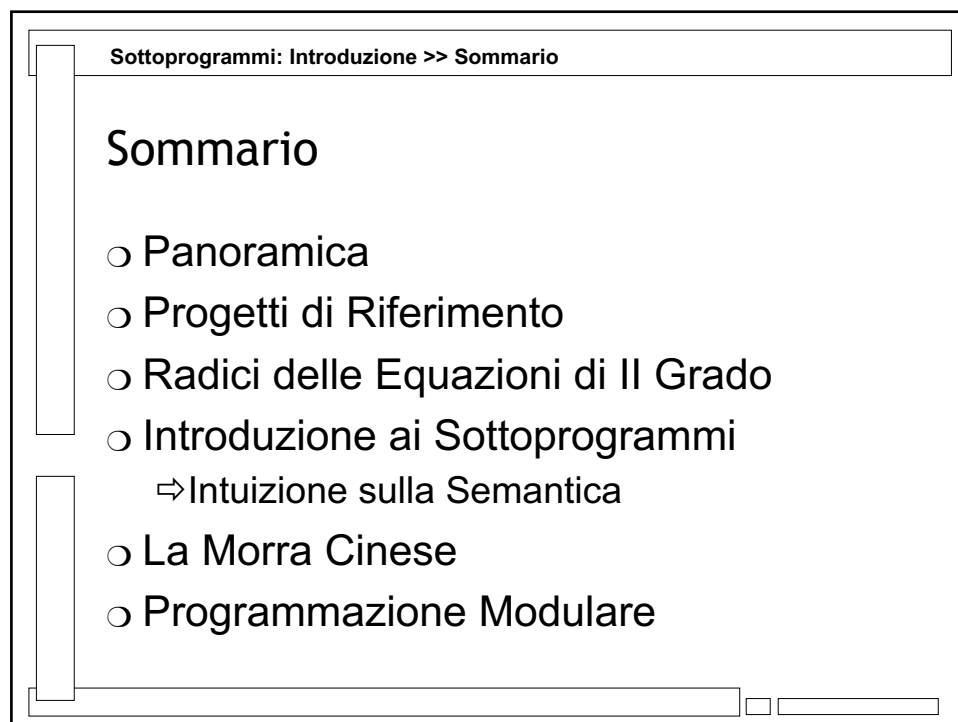




1



2

Sottoprogrammi: Introduzione >> Panoramica

Panoramica

- Istruzioni viste finora
- Istruzioni di base
 - ⇒ dichiarazioni (costanti e variabili, tipi)
 - ⇒ assegnazioni (espressioni)
 - ⇒ lettura e stampa (console e file)
- Strutture di controllo
 - ⇒ istruzioni condizionali (if-then-else, if-then)
 - ⇒ istruzioni iterative o cicli (while, for)

3

Sottoprogrammi: Introduzione >> Panoramica

Panoramica

- In questa parte del corso
 - ⇒ come dividere un programma in più “moduli”, detti sottoprogrammi
 - ⇒ tecnica fondamentale della programmazione per gestire la complessità dei programmi
 - ⇒ alla base delle metodologie di programmazione (cioè del metodo per affrontare la soluzione di problemi complessi)
 - ⇒ e di altri tipi di programmazione (es: POO)

4

Sottoprogrammi: Introduzione >> Introduzione ai Sottoprogrammi

Introduzione ai Sottoprogrammi

- Esistono due categorie di sottoprogrammi
- Funzioni
 - ⇒ servono a definire nuove funzioni per effettuare calcoli non previsti dalle funzioni predefinite
- Procedure
 - ⇒ servono a definire nuove istruzioni per eseguire operazioni non previste dalle istruzioni predefinite

10

Sottoprogrammi: Introduzione >> Introduzione ai Sottoprogrammi

Introduzione ai Sottoprogrammi

- Struttura di un'applicazione
 - ⇒ fatta di vari "moduli"
- Modulo
 - ⇒ un sottoprogramma (funzione o procedura)
 - ⇒ oppure il programma principale (un particolare sottoprogramma da cui comincia l'esecuzione dell'applicazione)

11

Introduzione ai Sottoprogrammi

- Per usare i sottoprogrammi
 - ⇒bisogna per cominciare “definirli”
 - ⇒e poi utilizzarli
- Definizione di un sottoprogramma
 - ⇒descrive il comportamento del sottoprogramma
- Utilizzo di un sottoprogramma
 - ⇒“chiamata” del sottoprogramma in un altro modulo dell'applicazione

12

Introduzione ai Sottoprogrammi

- Idea fondamentale
 - ⇒un sottoprogramma è un blocco di istruzioni
 - ⇒prevede una serie di “parametri” su cui eseguire le operazioni specificate (i parametri rappresentano “dati generici” del sottoprogramma)
 - ⇒ai parametri possono essere associati in ogni esecuzione “argomenti” diversi (ovvero “dati specifici” su cui effettuare concretamente le operazioni)

13

Introduzione ai Sottoprogrammi

- Dati nel programma principale
 - ⇒ finora i dati erano esclusivamente costanti e variabili
- Dati nei sottoprogrammi
 - ⇒ costanti
 - ⇒ variabili
 - ⇒ parametri: dati speciali che hanno un legame con argomenti esterni

14

Def. della Funzione “discriminante”

tipo del risultato nome della funzione parametri

float discriminante (float a, float b, float c)

```
{
    float d;
    d = b*b-4*a*c;
    return d;
}
```

istruzione di ritorno

La definizione è fatta di:
 -un'intestazione
 -un corpo

il sottoprogramma lavora su 4 dati:
 - una variabile di tipo float d
 - tre dati di tipo float chiamati a,b,c

15

Uso della Funzione “discriminante”

```
float discriminante (float a, float b, float c) {
    float d;
    d = b*b-4*a*c;
    return d;
}

int main() {
    float a1, b1, c1;
    float a2, b2, c2;
    ...
    if ( discriminante(a1, b1, c1) >= 0 &&
        discriminante(a2, b2, c2) >= 0) {
        ...
    }
}
```

prima chiamata

seconda chiamata

argomenti della I chiamata

argomenti della II chiamata

16

Def. della Procedura “leggiEquazione”

non c'è risultato nome della procedura parametri per modifiche

```
void leggiEquazione (float& a, float& b, float& c)
{
    cout << "Inserisci i coefficienti dell'equazione \n";
    cin >> a;
    cin >> b;
    cin >> c;
    return;
}
```

istruzione di ritorno

il sottoprogramma lavora su tre dati di tipo float chiamati a,b,c attraverso cui è possibile modificare i corrispondenti argomenti

17

Sottoprogrammi: Introduzione >> Introduzione ai Sottoprogrammi

Uso della Procedura “leggiEquazione”

```
void leggiEquazione (float& a, float& b, float& c) {
    cout << "Inserisci i coefficienti dell'equazione \n";
    cin >> a;
    cin >> b;
    cin >> c;
    return;
}

int main() {
    float a1, b1, c1;
    float a2, b2, c2;
    leggiEquazione(a1, b1, c1);
    leggiEquazione(a2, b2, c2);
    ...
}
```

Diagramma di annotazione:

- prima chiamata (punta alla prima chiamata di `leggiEquazione` in `main`)
- seconda chiamata (punta alla seconda chiamata di `leggiEquazione` in `main`)
- argomenti della I chiamata (punta al gruppo `(a1, b1, c1)`)
- argomenti della II chiamata (punta al gruppo `(a2, b2, c2)`)

18

Sottoprogrammi: Introduzione >> Introduzione ai Sottoprogrammi

Intuizione sulla Semantica

- Intuizione sulla semantica
 - ⇒ quando viene incontrata una “chiamata” del sottoprogramma, si interrompe l’esecuzione del programma
 - ⇒ avviene il “passaggio dei parametri” (>>)
 - ⇒ si avvia l’esecuzione del sottoprogramma
 - ⇒ al termine dell’esecuzione del sottoprogramma riprende l’esecuzione del programma

20

Intuizione sulla Semantica

- Passaggio dei parametri: intuizione
 - ⇒ prima di iniziare l'esecuzione del sottoprogramma, viene creata una "identificazione" tra parametri e argomenti
 - ⇒ l'esecuzione del sottoprogramma avviene sulla base dell'identificazione creata (operazioni fatte coi parametri diventano operazioni fatte con i corrispond. argomenti)
 - ⇒ l'identificazione termina al termine dell'esecuzione del sottoprogramma

21

Esempio: leggiEquazione

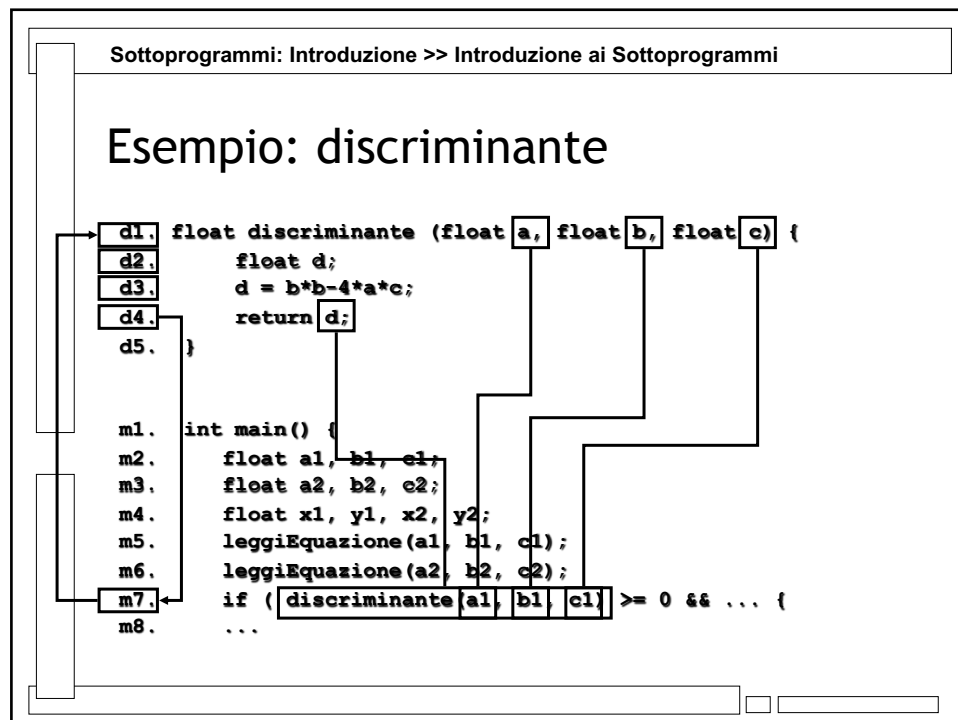
```

void leggiEquazione (float& a, float& b, float& c) {
    cout << "Inserisci i coefficienti dell'equazione \n";
    cin >> a;
    cin >> b;
    cin >> c;
    return;
}

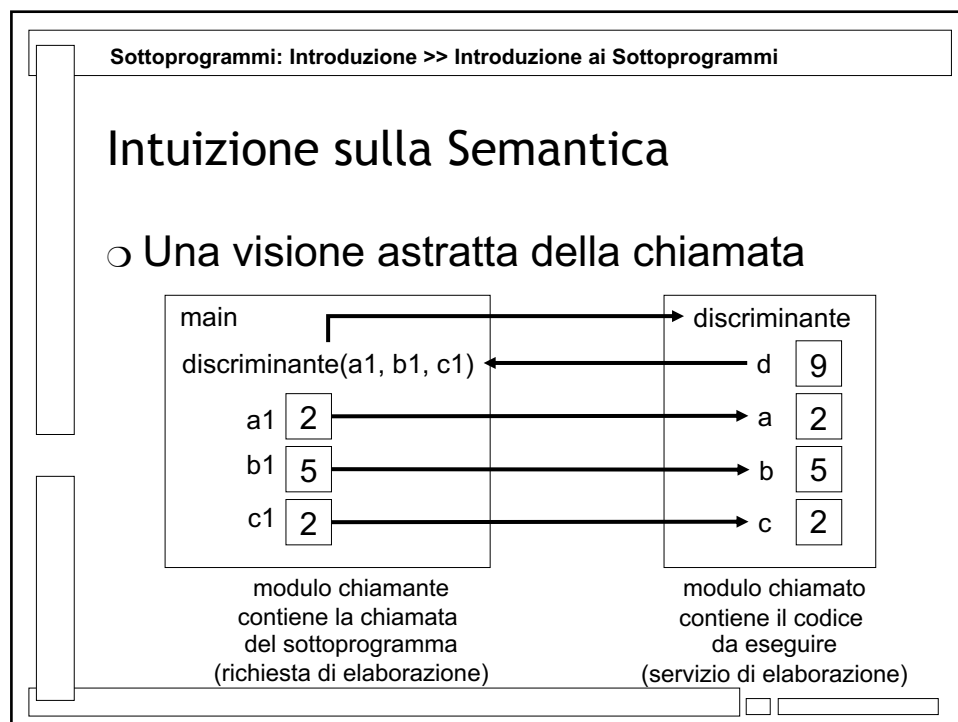
int main() {
    float a1, b1, c1;
    float a2, b2, c2;
    float x1, y1, x2, y2;
    leggiEquazione(a1, b1, c1);
    leggiEquazione(a2, b2, c2);
    if ( discriminante(a1, b1, c1) >= 0 && ... {
        ...
    }
}
  
```

The diagram illustrates the execution flow between the `leggiEquazione` function and the `main` function. On the left, a vertical list of labels represents the call stack: `le1.` through `le7.` for the function and `m1.` through `m8.` for the main function. Arrows show the sequence of execution: from `m1.` to `m2.`, then to `m3.`, which calls `leggiEquazione` at `le1.`. The function executes through `le2.` to `le6.` and returns to `m4.`. This process repeats for the second call to `leggiEquazione` at `m5.`, which returns to `m6.`. Finally, execution continues from `m7.` to `m8.` in the `main` function.

22



23



24

Intuizione sulla Semantica

- Vantaggi dei sottoprogrammi
 - ⇒ il codice viene scritto una sola volta
 - ⇒ e utilizzato tutte le volte che serve
 - ⇒ semplificano la lettura e la manutenzione del programma
- Ma c'è anche un vantaggio metodologico
 - ⇒ suggeriscono un metodo per affrontare la scrittura di programmi complessi

26

Intuizione sulla Semantica

- In sintesi quindi
 - ⇒ i sottoprogrammi sono un elemento fondamentale della programmazione
 - ⇒ da questo momento in poi programmeremo solo con i sottoprogrammi (anche per problemi semplici)
- In particolare
 - ⇒ approfondiremo le tecniche della “programmazione modulare”

33

Sottoprogrammi: Introduzione >> Programmazione Modulare

Programmazione Modulare

- Idea
 - ⇒ un programma è fatto di moduli che si “chiamano” a vicenda
 - ⇒ il programma principale è uno dei moduli (un particolare sottoprogramma da cui comincia l'esecuzione)
- La metodologia associata
 - ⇒ basata sull'uso di sottoprogrammi
 - ⇒ scomposizione del problema in sottoproblemi

34

Sottoprogrammi: Introduzione >> Programmazione Modulare

Programmazione Modulare

- Cosa bisogna sapere sui sottoprogrammi
 - ⇒ come si definiscono (come si creano): sintassi e semantica della definizione di sottoprogrammi
 - ⇒ come si utilizzano: sintassi e semantica delle chiamate
- In particolare
 - ⇒ la parte delicata è il meccanismo attraverso cui i sottoprogrammi “comunicano” tra di loro

35

Riassumendo

- Sottoprogrammi
 - ⇒ elemento fondamentale della programmazione
- Funzioni
 - ⇒ blocchi di istruzioni per il calcolo di un valore
- Procedure
 - ⇒ blocchi di istruzioni per realizzare operazioni varie
- Meccanismo Fondamentale
 - ⇒ “passaggio dei parametri”
- Programmazione Modulare

36

Termini della Licenza

- This work is licensed under the Creative Commons Attribution-ShareAlike License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/1.0/> or send a letter to Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.
- Questo lavoro viene concesso in uso secondo i termini della licenza “Attribution-ShareAlike” di Creative Commons. Per ottenere una copia della licenza, è possibile visitare <http://creativecommons.org/licenses/by-sa/1.0/> oppure inviare una lettera all'indirizzo Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

37

Programmazione Modulare

- Metodo per scegliere l'algoritmo
 - ⇒ l'algoritmo viene concepito per "raffinamenti successivi"
 - ⇒ si parte dividendo il problema complessivo in sottoproblemi più semplici e immediatamente identificabili
 - ⇒ poi si applica lo stesso metodo successivamente a ciascun sottoproblema