

Sottoprogrammi

Particolari programmi NON eseguibili autonomamente, ma soltanto sotto il controllo di un Programma, o un altro sottoprogramma, che li attiva (chiama).

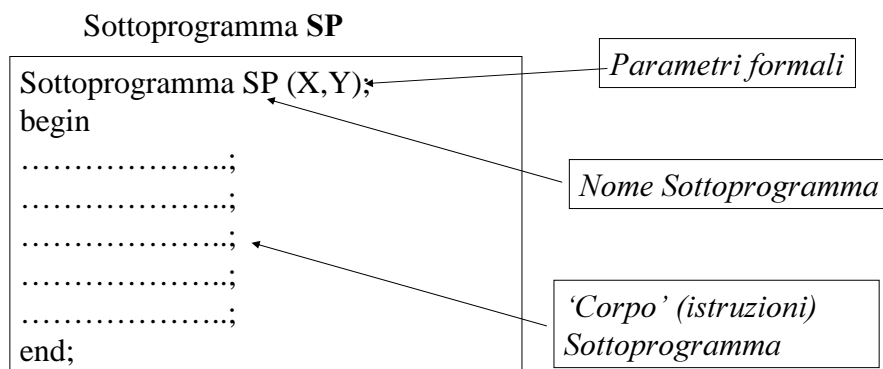
Motivazioni:

- caratteristiche di generalità
 - risolvere un problema di tipo generale molto diffuso: il codice relativo è prodotto una sola volta e riusato chiamando (attivando) lo specifico sottoprogramma
- decomposizione di un problema (programma) in sottoproblemi (sottoprogrammi)
 - permette di ridurre la complessità del problema complessivo che viene risolto più facilmente risolvendo ciascun sottoproblema per volta

... meccanismo di *astrazione funzionale*

Sottoprogrammi

Struttura di un Sottoprogramma:



Sottoprogrammi

L'attivazione di un Sottoprogramma **SP** avviene tramite una *istruzione di chiamata* in un programma **PG** detto *chiamante* o *principale*

All'attivazione di **SP**:

- l'esecuzione di **PG** è sospesa
- è effettuata l'esecuzione di **SP**
- alla fine dell'esecuzione di **SP** viene ripresa l'esecuzione di **PG** eseguendo la prima istruzione successiva a quella di chiamata di **SP**

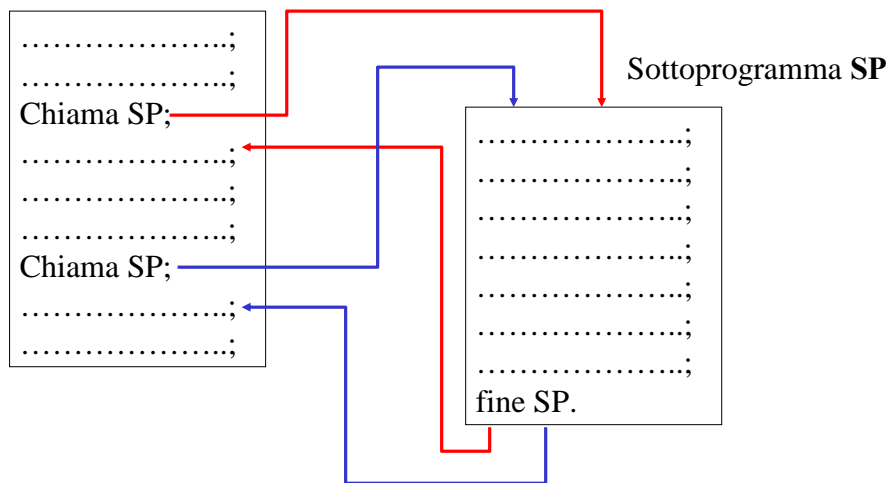
Uno stesso sottoprogramma può essere chiamato più volte sia dal programma principale che da altri sottoprogrammi

Uno stesso sottoprogramma può essere usato in programmi/sottoprogrammi diversi

Sottoprogrammi

... flusso di controllo in chiamate a sottoprogramma

Programma chiamante **PG**



Sottoprogrammi

Un Sottoprogramma opera sui valori di alcuni *parametri di ingresso* e calcola i valori di alcuni *parametri di uscita*, che sono resi disponibili al programma chiamante

Il sottoprogramma agisce (definisce e usa) *variabili locali*, proprie solo del sottoprogramma (sono dichiarate nel sottoprogramma).

Tra programma chiamante e sottoprogramma vi è uno scambio di informazioni effettuato tramite una *lista di parametri*

Sottoprogrammi

Parametri effettivi:

la lista delle variabili che il programma chiamante indica (*passa*) al sottoprogramma; sono proprie del programma chiamante (sono dichiarate in esso) e contengono i valori effettivi dell'elaborazione (i parametri *effettivi* sono detti anche *attuali*).

Parametri formali:

la lista delle variabili usate dal sottoprogramma nella sua esecuzione; sono proprie del sottoprogramma e contengono i valori scambiati con il programma principale corrispondenti ai parametri effettivi

I parametri effettivi possono essere nomi di variabili, costanti, espressioni

Sottoprogrammi

La lista dei parametri effettivi è indicata tra parentesi nella istruzione di chiamata del sottoprogramma

La lista dei parametri formali è indicata tra parentesi nell'intestazione del sottoprogramma. Ad ogni parametro effettivo deve corrispondere un parametro formale, e viceversa

La corrispondenza tra parametri è per posizione: al primo effettivo corrisponde il primo formale, e così via

All'attivazione di un sottoprogramma a ciascun parametro formale di questo è 'sostituito' il corrispondente parametro effettivo

Sottoprogrammi

Un esempio:

Programma chiamante **PG**

```
Programma PG;  
.....;  
Chiama SP(A,B);  
.....;  
.....;  
.....;  
.....;  
Chiama SP(M,24);  
.....;  
.....;  
.....;
```

Sottoprogramma **SP**

```
Sottoprogramma SP (X,Y);  
.....;  
.....;  
Somma=X+Y;  
write("Somma= ", Somma);  
.....;  
.....;  
.....;  
fine SP.
```

Parametri effettivi

Parametri formali

*Prima Attivazione: A ↔ X, B ↔ Y,
se A=5, B= 8 nell'esecuzione di SP è stampato Somma=13*
*Seconda Attivazione: M ↔ X, 24 ↔ Y,
se M=10, nell'esecuzione di SP è stampato Somma=34*

Sottoprogrammi

Principali meccanismi di scambio dei parametri:

per VALORE:

all'atto della chiamata il valore dei parametri attuali viene copiato nei corrispondenti parametri formali.

Il valore dei parametri effettivi NON viene modificato, anche se quello dei parametri formali lo è.

Sono utilizzate locazioni di memoria diverse per i parametri attuali e formali

Sottoprogrammi

Principali meccanismi di scambio dei parametri:

per RIFERIMENTO:

è detto anche per indirizzo:

all'atto della chiamata l'indirizzo di memoria di ciascun parametro effettivo viene ricopiato in quello del corrispondente parametro formale.

Qualsiasi variazione del valore del parametro formale si riflette sul corrispondente parametro formale.

E' utilizzata la stessa locazione di memoria per ciascun parametro effettivo ed il corrispondente formale

Sottoprogrammi

In una lista di parametri è possibile avere parametri scambiati sia per valore che per riferimento

Il tipo di scambio è indicato nella lista dei parametri formali, cioè nell'intestazione del sottoprogramma.

Ciascun linguaggio di programmazione ha proprie regole per indicare lo scambio per valore o per riferimento

Sottoprogrammi

In un Sottoprogramma, i parametri possono essere:

- *solo di ingresso*,
ovvero il valore dei parametri effettivi non deve essere modificato
==> Scambio per *VALORE*
- *solo di uscita*,
ovvero il valore del parametro calcolato nel sottoprogramma deve essere noto al programma chiamante
==> Scambio per *RIFERIMENTO*
- *di ingresso e di uscita*,
ovvero il valore di ingresso del parametro effettivo modificato dal sottoprogramma deve essere noto al programma chiamante
==> Scambio per *RIFERIMENTO*

Sottoprogrammi

Tipi di sottoprogrammi:

PROCEDURE:

simile ad un programma;

è definito il nome ed indicati i nomi dei parametri formali ed modo con cui questi sono scambiati con il chiamante

L'attivazione è fatta indicando il nome della procedure e la lista dei parametri effettivi

Schema di una procedure in LDP:

```
procedure <nome> (<lista parametri formali>);  
<parte dichiarativa>  
<corpo della procedure>  
end;
```

Sottoprogrammi

Esempio di chiamata a Procedura

Programma chiamante **PG**

```
Program PG;  
.....;  
SP(<lista1_par_effettivi>);  
.....;  
.....;  
.....;  
SP(<lista2_par_effettivi>);  
.....;  
.....;  
end.
```

Procedura **SP**

```
procedure SP (<lista_par_formali>);  
.....;  
.....;  
.....;  
.....;  
end;
```

*Istruzioni di chiamata a
Procedura*

Sottoprogrammi

Tipi di sottoprogrammi:

FUNZIONE:

- ha un tipo
- è identificata da un nome ed assume un valore, associato al nome della Funzione (quindi è un'informazione);
- è indicato il nome dei parametri formali ed il modo con cui questi sono scambiati con il chiamante

L'attivazione è fatta usando il nome della Funzione in una espressione (che userà il valore scauturante dall'esecuzione della Funzione) e la lista dei parametri effettivi

Sottoprogrammi

Schema di una Funzione in LDP:

```
<tipo> function <nome> (<lista parametri formali>);  
<parte dichiarativa>  
<corpo della function>  
return <valore calcolato>;  
end;
```

Sottoprogrammi

Esempio di chiamata a Funzione:
Programma chiamante **PG**

Program PG;
.....;
A= B + C * F(<lista_par_effettivi>);
.....;
.....;
end.

Funzione **F**

int function F (<lista_par_formali>);
.....;
.....;
.....;
return F;
end;

Elementi di Informatica
Prof. G. A. Di Lucca - Univ. del Sannio 17

Sottoprogrammi

Indicazione del meccanismo di scambio parametri in PDL:
**procedure <nome> ([in <lista parametri solo di ingresso>];
[out <lista parametri solo di uscita>];
[inout <lista par. ingresso/uscita>]);**

per VALORE

↙ ↘

per RIFERIMENTO

in maniera analoga per la Function

N.B. Per ogni parametro nella lista deve essere indicato il tipo
procedure <nome> (in float A,B; out float C, char S);

per VALORE

↙ ↘

per RIFERIMENTO

Elementi di Informatica
Prof. G. A. Di Lucca - Univ. del Sannio 18

Sottoprogrammi: effetto dei diversi tipi di scambio

```

Program PG;
.....;
C = 7;
SP(A,B,C);
write("C= ", C);
.....;
.....;
SP(M,24,C);
write("C= ", C);
.....;
end.
        
```

per VALORE per RIFERIMENTO

```

procedure SP (in int X,Y; out Somma);
.....;
.....;
Somma=X+Y;
write("Somma = ", Somma);
.....;
.....;
end;
        
```

Prima Attivazione: $A \leftrightarrow X, B \leftrightarrow Y, C \leftrightarrow \text{Somma}$
se $A=5, B= 8$ dopo l'esecuzione di SP è stampato $C=13$

Seconda Attivazione: $M \leftrightarrow X, 24 \leftrightarrow Y, C \leftrightarrow \text{Somma}$
se $M=10$, dopo l'esecuzione di SP è stampato $C=34$

Elementi di Informatica
Prof. G. A. Di Lucca - Univ. del Sannio 19

Sottoprogrammi: effetto dei diversi tipi di scambio

```

Program PG;
.....;
C = 7;
SP(A,B,C);
write("C= ", C);
.....;
.....;
SP(M,24,C);
write("C= ", C);
.....;
end.
        
```

tutti per VALORE

```

procedure SP (in int X,Y, Somma);
.....;
.....;
Somma=X+Y+Somma;
write("Somma = ", Somma);
.....;
.....;
end;
        
```

In SP: *Prima Attivazione:* $A \leftrightarrow X, B \leftrightarrow Y, C \leftrightarrow \text{Somma}$
se $A=5, B= 8, C=7 \Rightarrow \text{Somma} = 20$

Seconda Attivazione: $M \leftrightarrow X, 24 \leftrightarrow Y, C \leftrightarrow \text{Somma}$
se $M=10, \Rightarrow \text{Somma} = 41$

***In PG, dopo entrambe le attivazioni sarà sempre $C = 7$
perché C è scambiato per VALORE***

Elementi di Informatica
Prof. G. A. Di Lucca - Univ. del Sannio 20