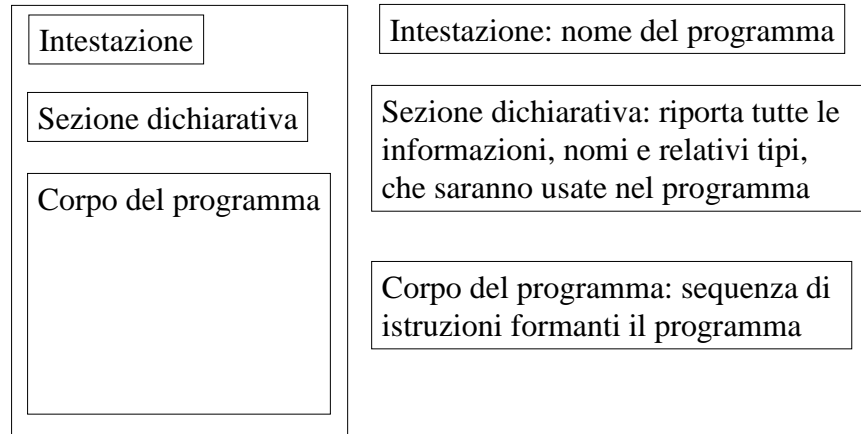


Struttura di un programma

Un programma è tipicamente strutturato nel seguente modo:



Un linguaggio per la definizione di Programmi

.... *regole lessicali e sintattiche*

un insieme di parole chiavi, regole per costruire nuovi nomi, regole per costruire le frasi

- frasi per indicare tipi e nomi delle informazioni (variabili)
- frasi per poter assegnare/modificare valori alle variabili
 - *operazioni di ingresso*
 - *calcolo ed assegnazione*
- frasi per operazioni di uscita
- frasi per la definizione istruzioni

... un ipotetico linguaggio di programmazione generico e generale ...

... Linguaggio di Definizione dei Programmi (LDP) ...

Struttura di un programma

... con riferimento ad un generico LDP ...

```
Program <nome programma>;  
// sezione dichiarativa  
    <nome tipo> <lista nomi_variabili>  
begin  
    <corpo del programma>  
end.
```

In **grassetto** le parole/simboli chiavi (lessico) del linguaggio

Tra < ... > ciò che scrive il programmatore

// indicano un commento: una frase non esecutiva per l'elaboratore ma utile al programmatore

Nomi delle variabili

devono iniziare con un carattere alfabetico possono avere una lunghezza qualsiasi non possono contenere il carattere <spazio>, non possono contenere i simboli degli operatori aritmetici

deve essere indicato il tipo di ogni variabile (dichiarazione della variabile)

```
<nome tipo> <lista variabili>;
```

Es.

```
FLOAT area_quadrato, lato_quadrato;
```

```
INT n.ro_studenti, ore_lezioni;
```

Tipi definiti dal linguaggio

i tipi atomici

INTERO	INT
REALE	FLOAT
CARATTERE	CHAR
LOGICO	BOOL

Istruzioni Semplici

- istruzioni di ingresso
- istruzioni di uscita
- istruzioni di calcolo ed assegnazione

in quasi tutti i linguaggi alle istruzioni di ingresso ed uscita corrisponde una azione elaborativa realizzata attraverso predefinite unità di programma

Istruzioni di ingresso (input)

Permettono di assegnare (definire) il valore di una o più informazioni (variabili) attraverso una unità di ingresso

prevedono:

- definizione dell'unità di ingresso coinvolta
- indicazione delle informazioni (*variabili*) cui assegnare il valore
- definizione del formato dei dati in ingresso

nella loro forma più semplice:

- l'unità di ingresso è individuata per "default"
(tipicamente la tastiera)
- il formato è individuato per "default"
(tipicamente il formato dei dati coincide con la rappresentazione prevista per le costanti del tipo ...)

Istruzioni di ingresso (input)

Hanno un formato del tipo:

read ([<unità ingresso>], [<formato>] <variabile>, ..., [<formato>] <variabile>);

le [...] indicano un qualcosa di opzionale

se [<unità ingresso>] non è indicata si assume l'unità di default

Es:

read ("%f" area_quadrato, "%f" lato_quadrato);

read ("%d" numero_pagine, "%f" prezzo_libro);

%f <==> formato per numero reale

%d <==> formato per numero intero

... quando è incontrata l'istruzione **read**, l'esecuzione del programma si interrompe fin quando non sono immessi i valori dei dati di input

Istruzioni di uscita (output)

permettono di trasferire i valori definiti da espressioni (variabili, costanti) sui supporti di una unità di uscita (schermo del video, foglio stampante etc.)

prevedono:

- individuazione dell'unità di uscita coinvolta
- indicazione delle espressioni (informazioni) coinvolte
- definizione del formato dei dati in uscita

nella loro forma più semplice:

- l'unità di uscita è individuata per "default"
(tipicamente il video)
- il formato è individuato per "default"
(tipicamente il formato dei dati coincide con la rappresentazione prevista per le costanti del tipo delle espressioni)

Istruzioni di uscita (output)

Hanno un formato del tipo:

write ([<unità uscita>], [<formato>] <espressione>, ... , [<formato>] <espressione>);

se [<unità uscita >] non è indicata si assume l'unità di default
<espressione> può essere: una variabile, una costante, una espressione

Es:

write ("%f" area_quadrato, "%f" lato_quadrato);

write ("%d" numero_pagine, "%f" prezzo_libro);

write ("%d" numero_pagine*300, "%f" prezzo+IVA, "%d" 900, "%c" 'B');

%c <==> formato per carattere

... l'esecuzione dell'istruzione **write**, ha come effetto quello di visualizzare sull'unità di output indicata i valori di <espressione>

Istruzioni di Calcolo ed Assegnazione

permettono di calcolare il valore di una espressione ed assegnarlo ad una informazione

informazione ed espressione devono essere dello stesso tipo

l'istruzione richiede:

- indicazione del nome della variabile che deve ricevere il valore
- operatore di assegnazione (=)
- definizione della espressione

Hanno la forma:

<nome variabile> = < espressione>;

<espressione> può essere: una variabile, una costante, una espressione

Esempio:

x = w + a * (y + 2);

y = 3;

Esempio

Definizione del problema: Calcolare la superficie di una stanza di forma rettangolare,

Definizione dei dati del problema:

I: base, altezza del rettangolo

Pi: base, altezza devono essere maggiori di zero

U: area

Pu: area maggiore di zero

Esempio

Tabella delle variabili di ingresso

Nome variabile	Descrizione	Tipo
base	base del rettangolo	FLOAT
altezza	altezza del rettangolo	FLOAT

Tabella delle variabili di uscita

Nome variabile	Descrizione	Tipo
area	area del rettangolo	FLOAT

Esempio

Precondizioni informazioni di ingresso

$(base > 0) \text{ and } (altezza > 0)$

Precondizioni informazioni di uscita:

$area > 0$

Modello della soluzione:

$area = base * altezza$

Esempio

... tralasciando per ora precondizioni e postcondizioni ...

Descrizione del metodo di elaborazione:

- sono letti (dalla tastiera) i valori di base ed altezza;
- è calcolato il valore di $area = base * altezza$
- è visualizzato (sul monitor) il valore di area calcolato;

Esempio

Codifica

- sono letti (dalla tastiera) i valori di base ed altezza;

read (base);

read (altezza);

...però in tale modo all'utente comparirà uno schermo vuoto con un cursore che lampeggia, meglio se ...

write ('immetti valore della base');

read (base);

write ('immetti valore della altezza');

read (altezza);

Esempio

Codifica

- è calcolato il valore di area = base * altezza

area = base * altezza;

- è visualizzato (sul monitor) il valore di area calcolato;

write ('Area= ', "%f" area);

Il programma di esempio

```

Program area Rettangolo;
// questo programma calcola l'area di un rettangolo
Float base, altezza, area;
begin
write ('immetti valore della base');
read ("%f" base);
write ('immetti valore della altezza ');
read ("%f" altezza);
area = base * altezza;
write ('Area= ', "%f" area);
end.
    
```

Diagramma di annotazione:

- intestazione: `Program area Rettangolo;`
- commento: `// questo programma calcola l'area di un rettangolo`
- Dichiarazioni variabili: `Float base, altezza, area;`
- Operazioni input: `write ('immetti valore della base');` e `read ("%f" base);`
- Operazioni output: `write ('immetti valore della altezza ');` e `write ('Area= ', "%f" area);`
- Calcolo e assegnazione: `area = base * altezza;`

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

Il programma di esempio

Cosa accadrà quando sarà eseguito ... dal punto di vista dell'utente

```

Program area Rettangolo;
Float base, altezza, area;
begin
write ('immetti base');
read ("%f" base);
    l'utente digita 7
    base = 7
write ('immetti altezza ');
read ("%f" altezza);
    l'utente digita 12
    altezza = 12
area = base * altezza;
    area = 7 * 12 = 84
write ('Area= ', "%f" area);
end.
    
```

immetti base
7
immetti altezza
12
Area= 84

Schermo del video

... ma se base o altezza =< 0 ?

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