

Programmazione I

A.A. 2002-03

Costrutti di base

(Lezione XII, parte I)

Gestione dell' input

Prof. Giovanni Gallo

Dr. Gianluca Cincotti

Dipartimento di Matematica e Informatica

Università di Catania

e-mail : { [gallo](mailto:gallo@dmf.unict.it), [cincotti](mailto:cincotti@dmf.unict.it) } @dmf.unict.it

Gestione dell' input

- È molto più complessa della gestione dell' *output*!
- Le finalità didattiche del corso di Programmazione 1 ci hanno suggerito di escludere la trattazione della programmazione grafica. Ma ...
 - Se si vuole gestire *l'input* da riga di comando occorre creare un oggetto un po' complesso chiamato "Console".
 - Nel libro di testo viene proposta un'alternativa che è quella che seguiremo in tutto il corso:
 - inserire gli input da "finestre" apposite.

Input di una stringa

```
import javax.swing.*;
public class ImmissioneStringa
{
    public static void main (String[] args)
    {
        String laStringaDiInput =
            JOptionPane.showInputDialog ("Digita una stringa:");
        System.out.println (laStringaDiInput);
        System.exit(0);
    }
}
```

Comando di importazione di una classe di oggetti non standard necessarie al funzionamento della classe che stiamo definendo.

Comando necessario per dire al sistema che può "abbandonare" la finestra di dialogo in quanto si è chiusa correttamente.

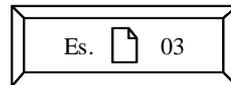
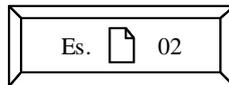
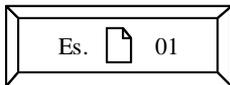
Input di un intero

```
import javax.swing.*;
public class ImmissioneNumero
{
    public static void main (String[] args)
    {
        String laStringaDiInput =
            JOptionPane.showInputDialog ("Digita un intero:");
        int numero = Integer.parseInt (laStringaDiInput);
        System.out.println (numero);
        System.exit(0);
    }
}
```

La stringa di input viene "tradotta" in un intero dal metodo `parseInt(stringa)` della classe predefinita `Integer`.

JOptionPane non ha alternative ?

- Come osservato prima, esiste una classe Console che legge gli input dalla stessa linea di comando in cui si scrive l'output.
 - Una lettura istruttiva... a fine corso.
 - Oppure: compilare e usare!



Quante cose da sapere e ricordare ...

- Ci aiuta la documentazione ON-LINE !!!
 - Installazione:
 - scaricare il file `jdk1_3.docs.zip`, unzipparlo e si otterrà un grande “ipertesto” in cui navigare con un browser.
 - ❖ Attenzione sono circa 30MB !

Un bel po' di informazioni e di roba ...
... può scoraggiare a causa di:

- INGLESE
- ELEMENTI NON ANCORA VISTI
- TROPPI DETTAGLI

SEGRETO: leggere solo ciò che serve!

LEGGE 20-80 : per sapere l'80% di ciò che giova basta leggere solo il 20% del manuale ...

Formattazione dell'output

Non verrà trattato questo argomento !

Leggete i dettagli su un manuale !

Fine

Programmazione I

A.A. 2002-03

Costrutti di base

(Lezione XII , parte II)

Costrutto di selezione “if ...else...”

Prof. Giovanni Gallo

Dr. Gianluca Cincotti

Dipartimento di Matematica e Informatica

Università di Catania

e-mail : { [gallo](mailto:gallo@dmf.unict.it), [cincotti](mailto:cincotti@dmf.unict.it) } @dmf.unict.it

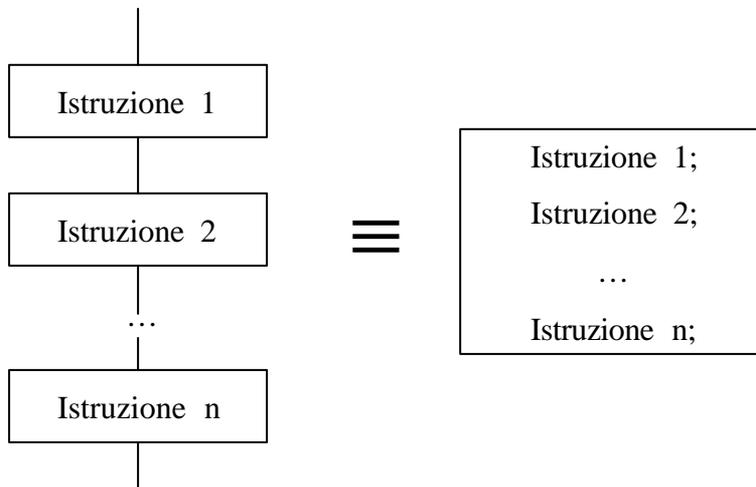
Notazione Lineare Strutturata

➤ Tre costrutti fondamentali :

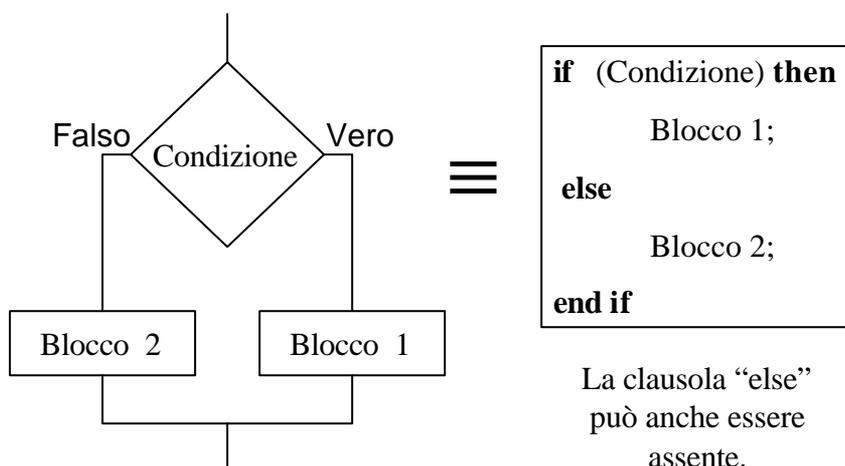
- Sequenza,
- Selezione,
- Iterazione.

➤ Servono a controllare il flusso del programma !

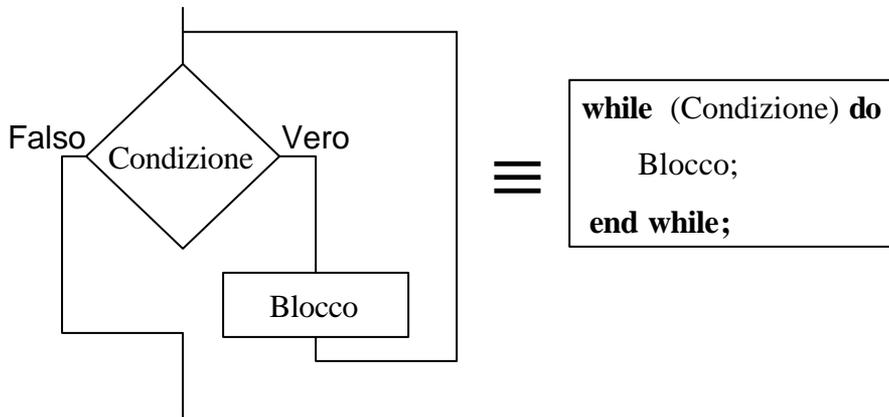
Sequenza



Selezione (o condizione)



Iterazione (o ciclo “while”)



Un'istruzione particolare ...

- L'istruzione *blocco* !
 - È costituita da una sequenza di istruzioni racchiuse tra parentesi graffe { ... }
- Un blocco viene usato per indicare di trattare un insieme di istruzioni come se fossero un'unica istruzione.
 - può essere usato là dove la sintassi di Java vuole un istruzione !
 - Molto utile in combinazione con i costrutti di selezione ed iterazione.

Esempio

```
{  
    n=1;  m=1;  
    System.out.print (n);  
    m += n++ + 3;  
    System.out.print (n+m);  
}
```

Blocchi composti da una sola istruzione

```
{  
    n=n+17;  
}
```

 \equiv

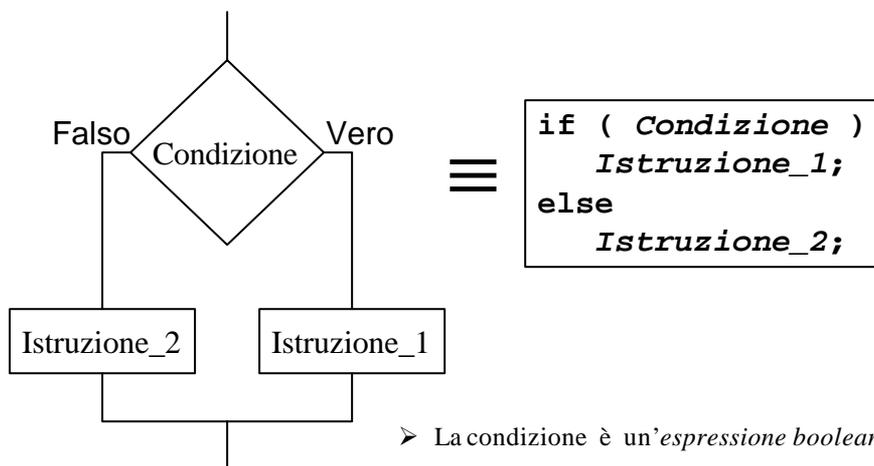
```
n=n+17;
```

è questione di
"gusti"

Istruzioni condizionali

- Il costrutto di *selezione* consente di prendere decisioni (cioè di scegliere l'istruzione successiva) in base al valore di un'espressione booleana.
- Tale costrutto viene realizzato in Java mediante le *istruzioni condizionali*:
 - *if ... else*
 - *switch*

Istruzione "if ... else ..."



- La condizione è un'espressione booleana.
- La clausola *else* è opzionale.

Modalità d'impiego

- 1) *if semplice* : if (condizione) istruzione;
- 2) *if ... else* : if (condizione) istruzione1;
 else istruzione2;
- 3) *if annidati* :
 if (condizione 1) istruzione1;
 else if (condizione 2) istruzione2;
 else if (condizione 3) istruzione3;
 else istruzione4;

Costrutto "if" semplice

```
System.out.print ("Ho vinto " + e);  
if ((e > MAX) && !sfortunato)  
    System.out.print ("000");  
System.out.println (" euro !!!");
```

Prima si valuta la condizione e poi si decide quale sarà la prossima istruzione da eseguire.

Costrutto "if" semplice (cont.)

```
System.out.print ("Ho vinto " + e);  
if ((e > MAX) && !sfortunato)  
{  
    Capitale += e;  
    System.out.print ("000");  
}  
System.out.println (" euro !!!");
```

Costrutto "if" semplice (cont.)

```
System.out.print ("Ho vinto " + e);  
if ((e > MAX) && !sfortunato)  
    Capitale += e;  
    System.out.print ("000");  
System.out.println (" euro !!!");
```



Problema !!!

Consiglio : Usare sempre le parentesi !

Costrutto “if ... else”

```
System.out.print("Hai digitato un numero");  
if (numero <= 5)  
    System.out.println (  
        "minore o eguale a 5");  
else  
    System.out.println ("maggiore di 5");
```

Costrutto “if ... else” (cont.)

```
if (numero > MAX)  
{  
    totale -= MAX;  
    numero--;  
}  
else  
{  
    totale += MAX;  
    numero++;  
}
```

Costrutto “if ... else” (cont.)

```
System.out.print ("Hai digitato un numero ");
if (numero <= 5)
    System.out.println ("minore o eguale a 5");
else
    {
        if (numero > 30)    numero--;
        System.out.println ("maggiore a 5");
    }
```

Costrutti “if” annidati

- L’istruzione da eseguire come risultato della valutazione di una condizione potrebbe essere a sua volta un’istruzione *if-else*.
 - Queste istruzioni sono dette *annidate*.

- **REGOLA** : Ogni clausola *else* è associata alla istruzione *if* immediatamente precedente ad essa.
 - non fatevi ingannare dall’indentazione !

Costrutti "if" annidati (cont.)

```
if (condizione 1) istruzione1;  
    else if (condizione 2) istruzione2;  
        else if (condizione 3) istruzione3;  
            else istruzione4;
```

```
if (condizione 1)  
    if (condizione 2) istruzione2;  
else if (condizione 3) istruzione3;  
else istruzione4;
```

Problema!!!

Es.  04

Es.  05

Es.  06

Esempio: Confronto tra caratteri

- Gli operatori relazionali possono essere usati anche sui dati di tipo carattere.
 - Il risultato dipende dalla posizione nell'insieme Unicode

```
if ( 'A' < 'a' )  
    System.out.print ( "A è minore di a" );
```

- La condizione è vera perché le lettere maiuscole sono antecedenti le minuscole in ASCII (e quindi Unicode)

Esempio: Confronto tra valori in virgola mobile

- Particolare attenzione va posta per il confronto di uguaglianza tra valori in virgola mobile (**float** o **double**)
 - Raramente si usa l'operatore di uguaglianza (**==**) per confrontare due numeri di questo tipo.
- È meglio considerare se i due valori sono sufficientemente vicini, anche se non identici, a causa
 - delle approssimazioni introdotte nella rappresentazione, e
 - degli errori prodotti dalle operazioni applicate ad essi.

```
if (Math.abs (f1 - f2) < 0.00001)
    System.out.println ("Praticamente uguali.");
```

Operatore condizionale

- L'operatore ternario *condizionale* valuta una condizione booleana che determina quale espressione, tra due possibili, valutare.
 - Il risultato dell'espressione selezionata diventa il risultato dell'operatore condizionale
- La sintassi:

condizione ? espressione_1 : espressione_2

- Se *condizione* è vera, allora viene valutata *espressione_1* altrimenti si valuta *espressione_2*

Operatore condizionale (cont.)

➤ L'operatore condizionale è simile all'istruzione *if...else*, tranne che riporta il valore di un'espressione.

➤ Esempio:

```
larger = (num1>num2) ? num1 : num2;
```

è equivalente a:

```
if (num1>num2) larger = num1;
    else larger = num2;
```

Operatore condizionale (cont.)

➤ Esempio:

```
System.out.println (
    "Il resto è di " + valore +
    (valore == 1) ? "Lira" : "Lire");
```

- se `valore` è 1, allora si stampa "Lira"
- per qualunque altro valore di `valore`, si stampa "Lire"

Fine