Lezione 5

Costrutti di scelta condizionale

Lezione 5 - Obiettivi

Al termine di questa lezione saremo in grado di

- Realizzare scelte condizionali nei programmi utilizzando costrutti if.
- Realizzare scelte condizionali nei programmi utilizzando costrutti switch.
- Scrivere espressioni logiche utilizzando operatori relazionali e logici.
- Valutare espressioni logiche correttamente.
- Decidere l'istruzione di scelta condizionale appropriata per il contesto.
- Scrivere un'applicazione utilizzando la classe ListBox del package javabook e la classe Color dal package standard java.awt.

Costrutto if-then-else

```
// messageBox e inputBox sono stati dichiarati e creati
// testScore è dichiarato come int

testScore = inputBox.getInteger(“Inserire il punteggio:");

if (testScore < 18)

    messageBox.show(“Voto insufficiente");

else

    messageBox.show(“Voto sufficiente");
```

Questa istruzione viene eseguita se testScore è inferiore a 18.

Questa istruzione viene eseguita se testScore è maggiore o uguale a 18.

Sintassi del Costrutto if-then-else

```
if ( <espressione logica> )

      <blocco then>
else

      <blocco else>
```

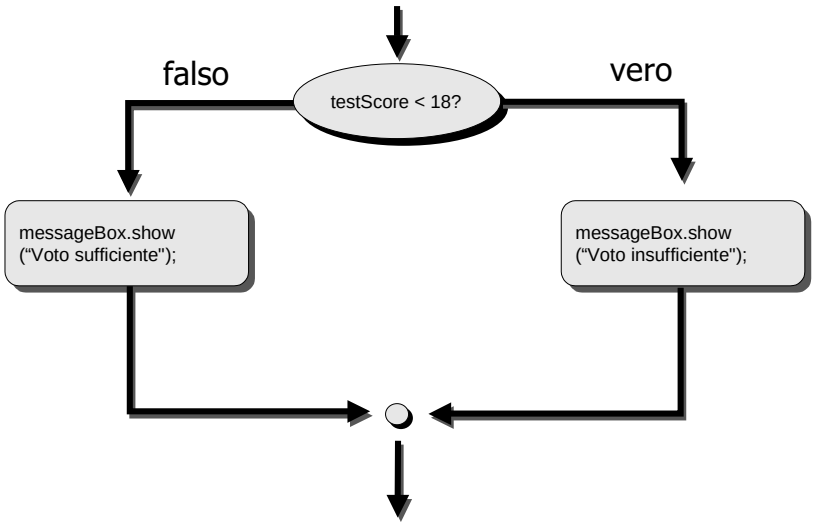
Espressione Logica

```
if (   testScore < 18    )
   messageBox.show(“Voto insufficiente");
else
   messageBox.show(“Voto sufficiente");
```

Blocco <then>

Blocco <else>

Flusso di Controllo if-then-else



 Introduction to Object-Oriented Programming with Java--Wu Chapter 6 - 5

Operatori Relazionali

```
<       // minore
<=      // minore o uguale
==      // uguale
!=      // diverso
>       // maggiore
>=      // maggiore o uguale
```

```
testScore < 80
testScore * 2 >= 350
30 < w / (h * h)
x + y != 2 * (a + b)
2 * Math.PI * radius <= 359.99
```

 Introduction to Object-Oriented Programming with Java--Wu Chapter 6 - 6

Istruzioni Multiple

- Il blocco <then> e il blocco <else> sono racchiusi tra graffe quando constano di più istruzioni.

```
if (testScore < 18)
{
    messageBox.show(“Voto insufficiente");
    messageBox.show(“Prova a studiare!");
}
else
{
    messageBox.show(“Voto sufficiente");
    messageBox.show(“Contunua così!");
}
```

Blocco <then>

Blocco <else>

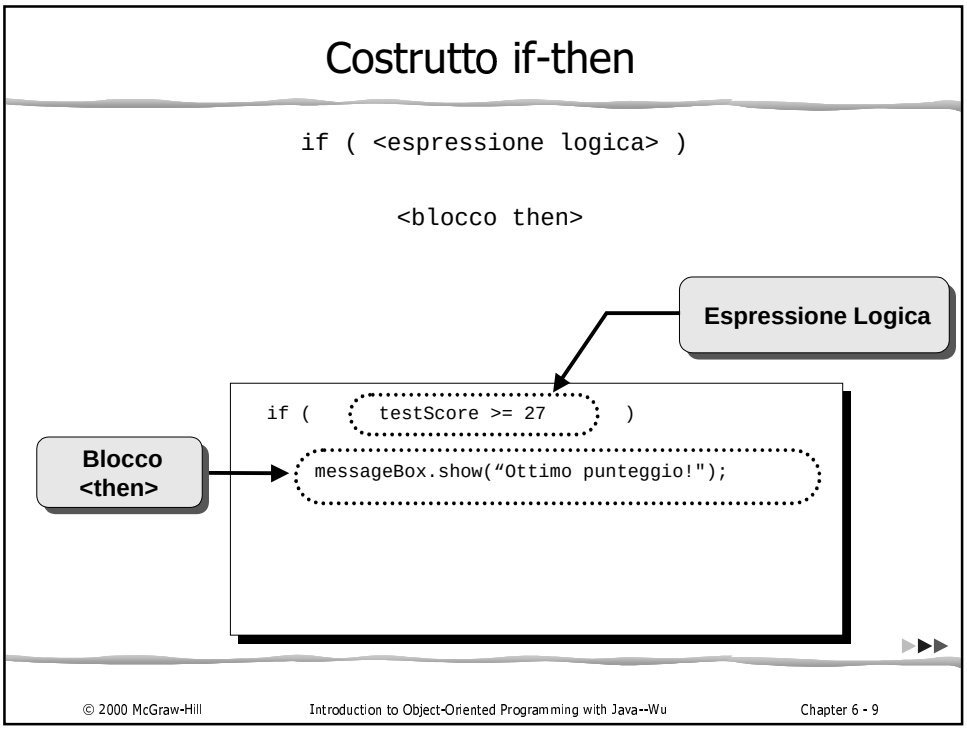
Un po’ di Stile...

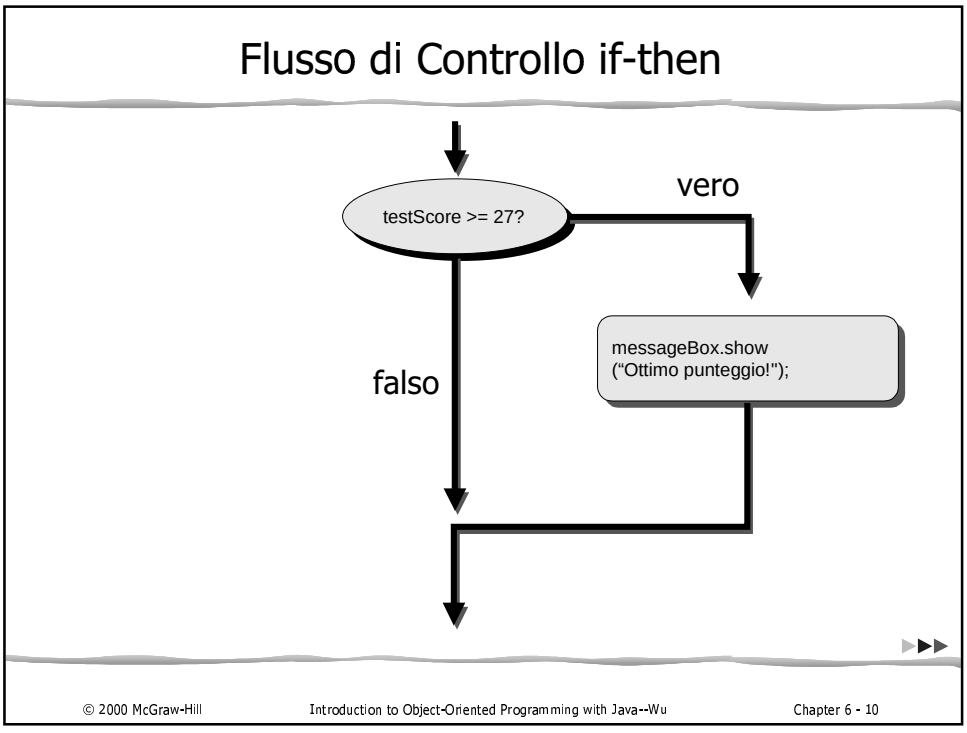
```
if ( <espressione logica> ) {
    …
}
else {
    …
}
```

Stile 1

```
if ( <espressione logica> )
{
    …
}
else
{
    …
}
```

Stile 2


Costrutto if-then
if (<espressione logica>)
<blocco then>
Espressione Logica
Blocco <then>
if (testScore >= 27)
messageBox.show("Ottimo punteggio!");
© 2000 McGraw-Hill
Introduction to Object-Oriented Programming with Java--Wu
Chapter 6 - 9


Flusso di Controllo if-then
testScore >= 27?
vero
falso
messageBox.show
("Ottimo punteggio!");
© 2000 McGraw-Hill
Introduction to Object-Oriented Programming with Java--Wu
Chapter 6 - 10

Espressioni e Variabili Logiche (Booleane)

A	B	A && B	A || B	!A
falso	falso	falso	falso	vero
falso	vero	falso	vero	vero
vero	falso	falso	vero	falso
vero	vero	vero	vero	falso

Valutazione degli Operatori Logici

- Si consideri la seguente espressione logica:

```
x > y || x > z
```

- L'espressione è valutata da sinistra a destra. Se x > y è vera, allora non è necessrio valutare x > z perché l'intera espressione sarà vera indipendentemente dal valore di verità di x > z.
- Fermare la valutazione quando il risultato dell'intera espressione è noto corrisponde ad una valutazione di tipo *short-circuit* (letteralmente, corto circuito).
- Cosa accadrebbe se la seguente espressione non fosse valutata in maniera short-circuit?

```
z == 0 || x / z > 20
```


Precedenze tra Operatori

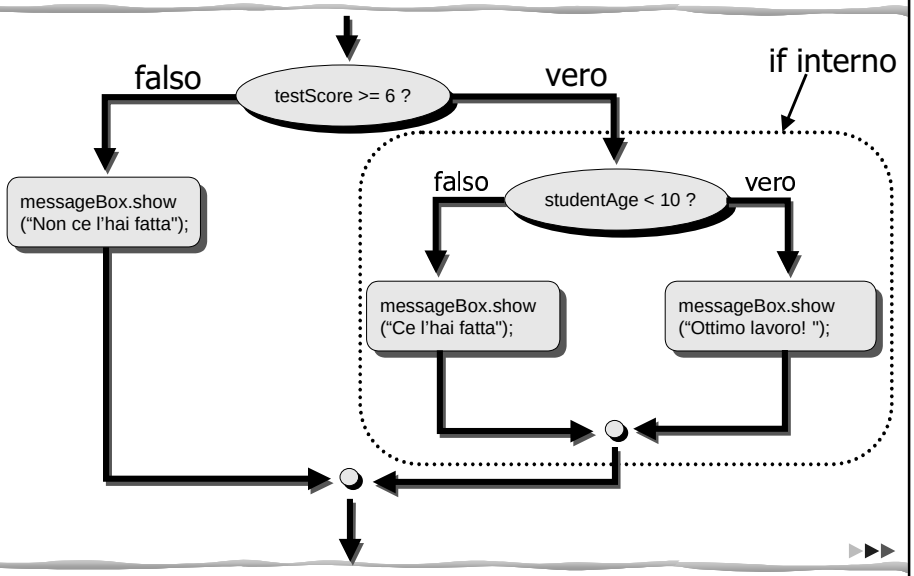
Group	Operator	Precedence	Associativity
subexpression	( )	9 (If parentheses are nested, then innermost subexpression is evaluated first.)	left to right
unary operators	- !	8	right to left
multiplicative operators	* / %	7	left to right
additive operators	+ -	6	left to right
relational operators	< <= > >=	5	left to right
equality operators	== !=	4	left to right
boolean AND	&&	3	left to right
boolean OR	||	2	left to right
assignment	=	1	right to left

Costrutti if Annidati

- Il blocco <then> e il blocco <else> possono contenere qualsiasi istruzione valida, inclusi altri costrutti if. Questi ultimi si dicono costrutti if annidati (nested).

```
if (testScore >= 6) {
    if (studentAge < 10) {
        messageBox.show(“Ottimo lavoro!");
    }
    else {
        messageBox.show(“Ce l’hai fatta"); //test score >= 6
    }                                      //and age >= 10

}
else { //test score < 6

    messageBox.show(“Non ce l’hai fatta");
}
```


Flusso di Controllo di If Annidati



Scrivere Istruzioni If in Modo Corretto

```
if (num1 < 0)
     if (num2 < 0)
          if (num3 < 0)
               negativeCount = 3;
          else
               negativeCount = 2;
     else
          if (num3 < 0)
               negativeCount = 2;
          else
               negativeCount = 1;
else
     if (num2 < 0)
          if (num3 < 0)
               negativeCount = 2;
          else
               negativeCount = 1;
     else
          if (num3 < 0)
               negativeCount = 1;
          else
               negativeCount = 0;
```

```
negativeCount = 0;

if (num1 < 0)
       negativeCount++;
if (num2 < 0)
       negativeCount++;
if (num3 < 0)
       negativeCount++;
```

L’istruzione

negativeCount++;

incrementa di uno la variabile

Costrutto if – else if

Punteggio	Voto
90 ≤ score	A
80 ≤ score < 90	B
70 ≤ score < 80	C
60 ≤ score < 70	D
score < 60	F

```
if (score >= 90)
   messageBox.show(“Il tuo voto è A");

else if (score >= 80)
   messageBox.show("Il tuo voto è B");

else if (score >= 70)
   messageBox.show("Il tuo voto è C");

else if (score >= 60)
   messageBox.show("Il tuo voto è D");

else
   messageBox.show("Il tuo voto è F");
```


Associazione tra if ed else

A e B sono differenti?

A

```
if (x < y)
   if (x < z)
      messageBox.show("Hello");
else
   messageBox.show("Good bye");
```

B

```
if (x < y)
   if (x < z)
      messageBox.show("Hello");
   else
      messageBox.show("Good bye");
```

Sia A che B significano

```
if (x < y) {
   if (x < z) {
      messageBox.show("Hello");
   }
   else {
      messageBox.show("Good bye");
   }
}
```


ListBox

- La classe ListBox fornisce una lista di cui l'utente può selezionare una sola voce
- A livello di interfaccia, è meglio evitare che l'utente possa compiere scelte non valide piuttosto che rilevare l'errore a posteriori.
- Innanzi tutto creiamo un oggetto ListBox

```
MainWindow mainWindow = new MainWindow( );
ListBox    colorList  = new ListBox( mainWindow,
                                     "Select Color" );
```

Possiamo creare colorList come

… = new ListBox(mainWindow);

se non necessitiamo di un titolo.

ListBox – Aggiungere Elementi

- Si possono aggiungere voci ad un oggetto ListBox

```
colorList.addItem("Magenta");
colorList.addItem("Cyan");
colorList.addItem("Red");
colorList.addItem("Blue");
colorList.addItem("Green");
```

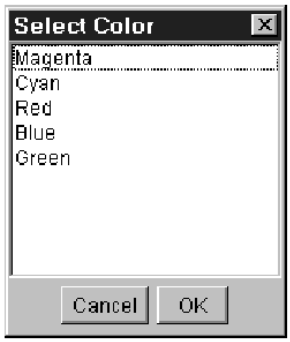
Questi elementi verranno aggiunti alla lista nell'ordine in cui addItem è invocato.

- Eseguendo

```
selection = colorList.getSelectedIndex();
```

si visualizza il contenuto di colorList

ListBox – Voci in colorList



Index value	
0	Magenta
1	Cyan
2	Red
3	Blue
4	Green

Il metodo getSelectedIndex ritorna l'indice della voce prescelta.

Metodi di ListBox

CLASS: ListBox		
Method	**Argument**	**Description**
`<constructor>`	`MainWindow`	Creates a ListBox object.
`addItem`	`String`	Adds the argument String value to the list. Items are added to the list from top to bottom. The top-most item has the index value of zero, the next item's value is one, and so forth.
`getSelectedIndex`	`<none>`	Returns the index value of the selected item in the list. See the explanation of class constants.
Class Constant		**Description**
`NO_SELECTION`		This value is returned by the getSelectedIndex method when the user clicks the OK button without selecting a choice.
`CANCEL`		This value is returned by the getSelectedIndex method when the user clicks the CANCEL button or the dialog's close box.

ListBox – Gestire la Scelta dell'Utente

```
selection = colorList.getSelectedIndex();
if (selection == ListBox.NO_SELECTION)
   messageBox.show("Nessuna selezione");

else if (selection == ListBox.CANCEL)
   messageBox.show("Hai premuto Cancel");

else if (selection == 0)
   messageBox.show("Hai scelto Magenta");

 • • •

else if (selection == 3)
   messageBox.show("Hai scelto Blue");

else if (selection == 4)
   messageBox.show("Hai scelto Green");
```

L'uso delle costanti NO_SELECTION e CANCEL rende il codice più leggibile. Rende anche più facile modificare il codice.

L'Istruzione switch

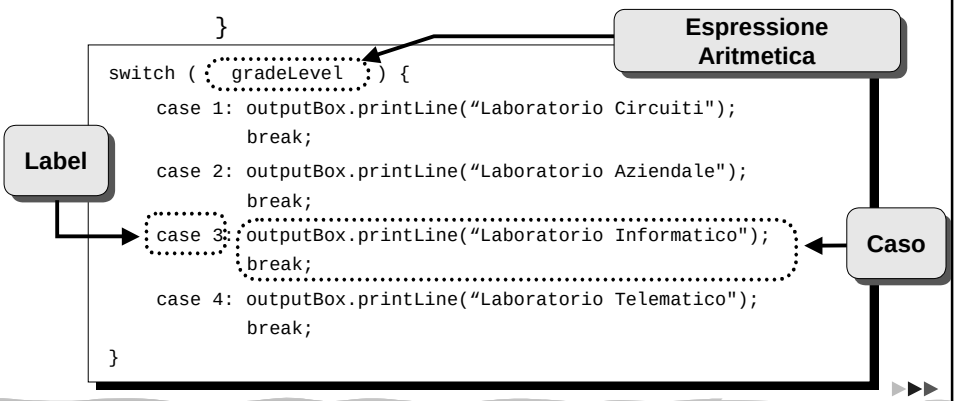
```
int gradeLevel;
gradeLevel = inputBox.getInteger("Corso (ELE-1,GEST-2,...):" );

switch (gradeLevel) {

    case 1: outputBox.printLine("Laboratorio Circuiti");
            break;

    case 2: outputBox.printLine("Laboratorio Aziendale");
            break;

    case 3: outputBox.printLine("Laboratorio Informatico");
            break;

    case 4: outputBox.printLine("Laboratorio Telematico");
            break;
}
```

Viene eseguita se gradeLevel è uguale a 1.

Viene eseguita se gradeLevel è uguale a 4.

Sintassi del Costrutto switch

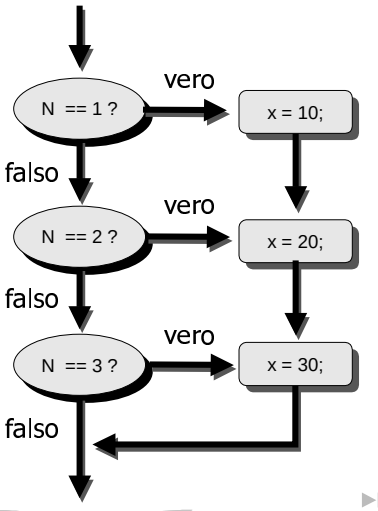
```
switch ( <espressione aritmetica> ) {
        <label 1> : <caso 1>
        …
        <label n> : <caso n>
}
```



 Introduction to Object-Oriented Programming with Java--Wu Chapter 6 - 25

switch senza Istruzioni break

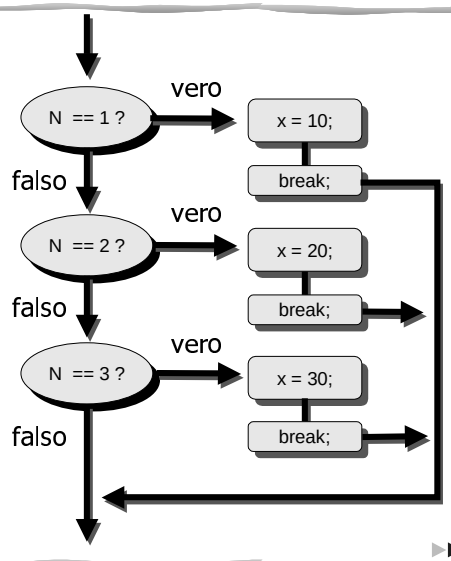
```
switch ( N ) {
   case 1: x = 10;
   case 2: x = 20;
   case 3: x = 30;
}
```



 Introduction to Object-Oriented Programming with Java--Wu Chapter 6 - 26

switch con Istruzioni break

```
switch ( N ) {
   case 1: x = 10;
           break;
   case 2: x = 20;
           break;
   case 3: x = 30;
           break;
}
```



switch con Blocco default

```
switch (ranking) {

    case 10:
    case  9:
    case  8: messageBox.show("Maestro");
             break;

    case  7:
    case  6: messageBox.show(“Anziano");
             break;

    case  5:
    case  4: messageBox.show("Apprendista");
             break;

    default: messageBox.show(“Errore: dato non valido");
             break;
}
```


Esempio: Disegnare Figure

- Problema

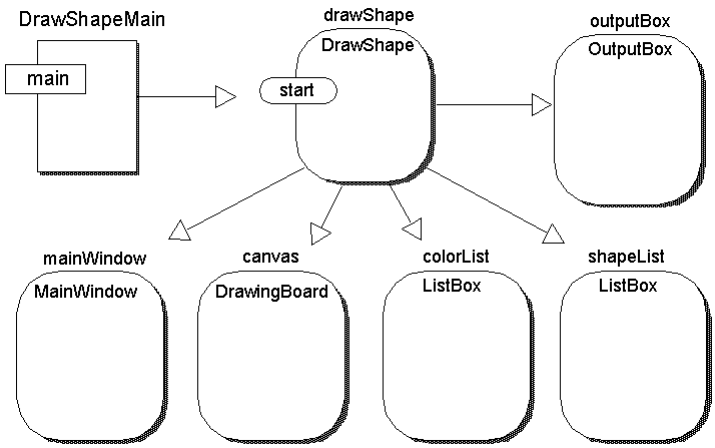
  Scrivere un'applicazione che disegni una figura geometrica specificata dall'utente. L'utente può specificare la posizione, la dimensione e il colore della figura. Il programma può disegnare linee, rettangoli e cerchi, nei colori magenta, cyan, red, blue oppure green.

- Compiti Principali
  1. Chiedere all'utente la figura desiderata.
  2. Chiedere all'utente il colore desiderato.
  3. Chiedere la posizione e la dimensione della figura selezionata.
  4. Disegnare la figura selezionata.

DrawShape – Progetto

Design Document: DrawShape	
Class	**Purpose**
`DrawShapeMain`	The main class of the program.
`DrawShape`	The top-level control object that manages other objects in the program.
`ListBox`	One ListBox object is used for listing the available shapes: rectangle, circle, and line. Another ListBox object is used for listing the available colors: magenta, cyan, red, blue, and green. This class is from javabook.
`InputBox`	An InputBox object is used to get values for the position and size of the shape to be drawn. This class is from javabook.
`OutputBox`	An OutputBox object is used to display the program description. The object is used also for printing out temporary messages during the program development. This class is from javabook.
`DrawingBoard`	A core object that does the actual drawing.
`MainWindow`	The main window of the application.

DrawShape – Diagramma



DrawShape – Sviluppo

1. Iniziare con uno scheletro del programm. Definire le classi DrawShapeMain e DrawShape.
2. Aggiungere le istruzioni per consentire all'utente di selezionare una figura.
3. Aggiungere le istruzioni per consentire all'utente di selezionare un colore.
4. Aggiungere codice per consentire all'utente di specificare la dimensione e la posizione della figura selezionata.
5. Aggiungere il codice per disegnare la figura desiderata.
6. Finalizzare il programma assemblando le componenti.

