Plan

- Control Structures
  - if Single-Selection Statement
  - if else Selection Statement
  - while Repetition Statement
  - for Repetition Statement
  - do...while Repetition Statement
  - switch Multiple-Selection Statement
  - break and continue Statements
  - Structured Programming Summary

Control Structures

- Java has a sequence structure “built-in”
- Java provides three selection structures
  - if
  - if else
  - switch
- Java provides three repetition structures
  - while
  - do...while
  - do
- Each of these words is a Java keyword

if Single-Selection Statement

- Single-entry/single-exit control structure
- Perform action only when condition is true
- Action/decision programming model

if...else Selection Statement

- Perform action only when condition is true
- Perform different specified action when condition is false
- Conditional operator (?:)
  - System.out.println( grade >= 60 ? “Passed” : “Failed” );

while Repetition Statement

- Repeat action while condition remains true
Formulating Algorithms

- Counter-controlled repetition
  - Variable that controls number of times set of statements executes
- Sentinel-controlled repetition
  - User enters sentinel value (-1) to end repetition

Outline 8

Average1.java

1   // Fig. 4.7: Average1.java
2   // Class-average program with counter-controlled repetition.
3   import javax.swing.JOptionPane;
4
5   public class Average1 {
6
7   public static void main( String args[] )
8      {
9      int total;          // sum of grades input by user
10     int gradeCounter;   // number of grade to be entered next
11     int grade;          // grade value
12     int average;        // average of grades
13
14     String gradeString; // grade typed by user
15
16     // initialization phase
17     total = 0;          // initialize total
18     gradeCounter = 1;   // initialize loop counter
19
20     // processing phase
21     while ( gradeCounter <= 10 ) {  // loop 10 times
22
23     // prompt for input and read grade from user
24     gradeString = JOptionPane.showInputDialog(
25     "Enter integer grade: " );
26
27     // convert gradeString to int
28     grade = Integer.parseInt( gradeString );
29
30     total = total + grade;            // add grade to total
31     gradeCounter = gradeCounter + 1;  // increment counter
32
33     } // end while
34
35     // termination phase
36     average = total / 10;  // integer division
37
38     // display average of exam grades
39     JOptionPane.showMessageDialog( null, "Class average is " + average,
40     "Class Average", JOptionPane.INFORMATION_MESSAGE );
41
42     System.exit( 0 );  // terminate the program
43
44      } // end main
45
46   } // end class Average1

Outline 10

Average1.java

Assignment Operators

- Assignment Operators
  - Abbreviate assignment expressions
  - Any statement of form
    - `variable = variable operator expression`
  - Can be written as
    - `variable operator expression`;
  - e.g., addition assignment operator `+=`
    - `c += 3`
  - can be written as
    - `c = c + 3`

Assignment Table
Increment and Decrement Operators

- Unary increment operator (++)
  - Increment variable’s value by 1
- Unary decrement operator (--)
  - Decrement variable’s value by 1
- Preincrement / predecrement operator
- Post-increment / post-decrement operator

<table>
<thead>
<tr>
<th>Operator</th>
<th>Called</th>
<th>Sample Expression</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>++</td>
<td>Increment</td>
<td>++a</td>
<td>Increment a by 1, then use the new value of a in the expression in which a resides.</td>
</tr>
<tr>
<td>--</td>
<td>Decrement</td>
<td>--b</td>
<td>Decrement b by 1, then use the current value of b in the expression in which b resides.</td>
</tr>
<tr>
<td>++</td>
<td>Preincrement</td>
<td>a++</td>
<td>Use the current value of a in the expression in which a resides, then increment a by 1.</td>
</tr>
<tr>
<td>--</td>
<td>Predecrement</td>
<td>b--</td>
<td>Use the current value of b in the expression in which b resides, then decrement b by 1.</td>
</tr>
</tbody>
</table>

Fig. 4.13 The increment and decrement operators.

```java
// Fig. 4.14: Increment.java
// Preincrementing and postincrementing operators.
// Fig. 4.14: Increment.java
// Preincrementing and postincrementing operators.
public class Increment {
    public static void main( String args[] )
    {
        int c;

        // demonstrate postincrement
        c = 5;                     // assign 5 to c
        System.out.println( c );   // print 5
        System.out.println( c++ ); // print 5 then postincrement
        System.out.println( c );   // print 6

        System.out.println();      // skip a line

        // demonstrate preincrement
        c = 5;                     // assign 5 to c
        System.out.println( c );   // print 5
        System.out.println( ++c ); // preincrement then print 6
        System.out.println( c );   // print 6

    } // end main
} // end class Increment
```

```
Line 13 postincrements c
Line 21 preincrements c

for Repetition Statement

- Handles counter-controlled-repetition details

```
```
**switch Multiple-Selection Statement**

- **switch statement**
  - Used for multiple selections

```
switch (choice) {
    case 1: // draw a line
        g.drawLine(10, 10, 250, 10 + i * 10);
        break; // done processing case
    case 2: // draw a rectangle
        g.drawRect(10 + i * 10, 10 + i * 10, 50 + i * 10, 50 + i * 10);
        break; // done processing case
    case 3: // draw an oval
        g.drawOval(10 + i * 10, 10 + i * 10, 50 + i * 10, 50 + i * 10);
        break; // done processing case
    default: // draw string indicating invalid value entered
        g.drawString("Invalid value entered", 10, 20 + i * 15);
}
```