WKW SCHOOL

2018

CI6206 Internet Programming

Java Fundamental

Wong Twee Wee

Ver1.4

KEYWORDS IN JAVA

Table 2-1 Java Keywords

abstract	boolean	break	byte
case	catch	char	class
const	continue	default	do
double	else	extends	final
finally	float	For	goto
If	implements	import	instanceof
int	interface	long	native
new	package	private	protected
public	return	short	static
strictfp	super	switch	synchronized
this	throw	throws	transient
try	void	volatile	while

BUILDING A JAVA CLASS

- Each source code file defines a class
  - Class
    - HelloWorldWideWeb
  - File
    - HelloWorldWideWeb.java

RULES

- Java is case sensitive
  - *Public* isn't the same as *public*
- Semicolon (;)
  - All java statements end with a semicolon

```
PrintWriter out = response.getWriter();

try {
      out.println("<!DOCTYPE html>");
      out.println("<html><head>");
}
```

COMMENTS

- Comments
  - Single line
    - // compiler ignores everything to end of line
  - Multi-line
    - /* compiler ignores everything in between */
  - Multi-line (documentation)
    - /** compiler ignores everything in between */
    - Used for JavaDoc

USING JAVA VARIABLES AND DATA TYPES

- Declaring and Initializing Variables
  - Variable data type must be declared prior to initialization
    - Eight available primitive data types
  - Assignment operator (=)
    - Used to assign value to a variable
      - char c = ‘a’;
      - boolean b = true;
      - double d = 1.25;

USING JAVA VARIABLES AND DATA TYPES

Table 2-3 Java Primitive Data Types

	Type	Range of Values	Size
Numeric with no decimals	1. int	+ or – 2.1 trillion	4 bytes
	2. short	+ or – 32,000	2 bytes
	3. long	+ or – 9 E18	8 bytes
	4. byte	+ or – 127	1 byte
Numeric with decimals	5. double	+ or – 1.79 E308	8 bytes, 15 decimals
	6. float	+ or – 3.4 E38	4 bytes, 7 decimals
Other	7. boolean	true or false	
	8. char	any character	2 bytes

USING JAVA VARIABLES AND DATA TYPES

- Using Constants
  - Variable with a value that doesn't change
  - Keyword
    - final
      - Denotes value cannot change
  - Example:
    - final double SALES_TAX_RATE = 4.5;

OPERATORS

Table 2-4 Java Arithmetic Operators

Operator	Description	Example	Result
+	addition	11 + 2	13
–	subtraction	11 –2	9
*	multiplication	11 * 2	22
/	division	11 / 2	5
%	remainder	11 % 2	1

COMPUTING WITH JAVA

- Special Operators
  - For writing shortcut code
    - Increment operator (++)
      - Add one to a variable

      int count = 0 ;

      count ++;
    - Decrement operator (--)
      - Subtract one from a variable
    - Assignment operator with arithmetic operators:

      total = total + 5;

      Total += 5;

WRITING DECISION-MAKING STATEMENTS

- Decision Making Statement
  - Determine whether a condition is true, and take some action based on determination
- Three ways to write decision-making statements:
  - if statement
  - switch statement
  - conditional operator

WRITING DECISION-MAKING STATEMENTS

- Writing *if* Statements
  - *if* statement:
    - Interrogates logical expression enclosed in parentheses
    - Determines whether it is true or false
    - Uses logical operators to compare values:
      - e.g., (studentAge < 21)

LOGICAL OPERATORS

Table 2-6 Java Logical Operators

Operator	Description
&&	And
==	equal to
>	greater than
>=	greater than or equal to
<	less than
<=	less than or equal to
!	Not
!=	not equal to
||	Or

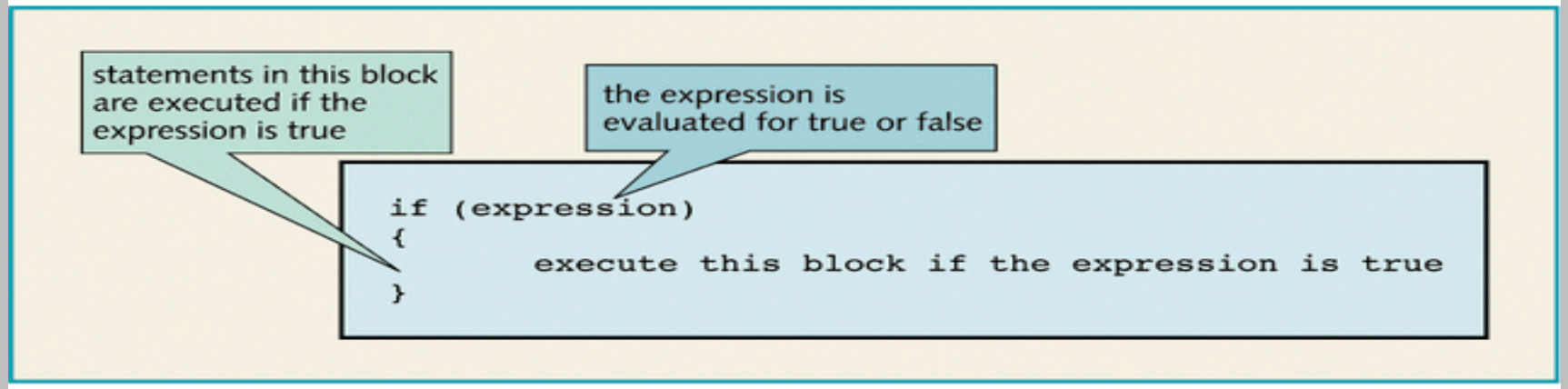

Figure 2-12 if statement format

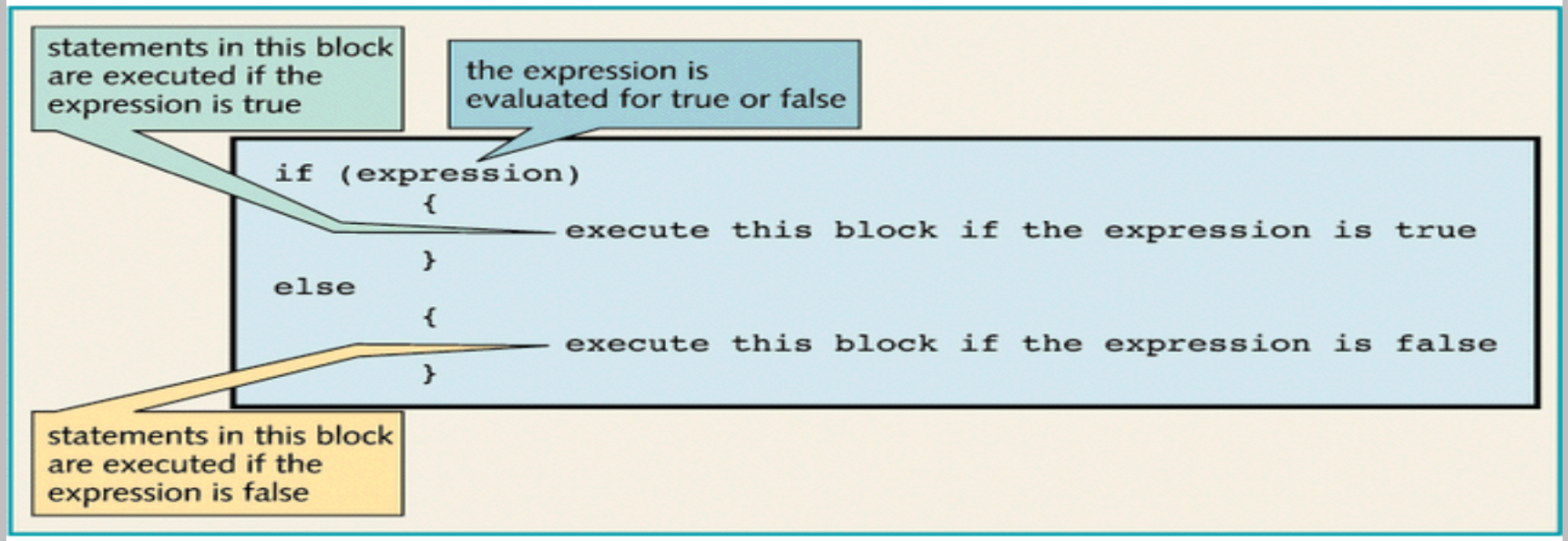

Figure 2-13 if-else statement format

WRITING DECISION-MAKING STATEMENTS

- Writing *if* Statements
  - Compound expression
    - Two expressions joined using logical operators
      - OR → ||
      - AND → &&
  - Nested *if* statement
    - *if* statement written inside another *if* statement

WRITING DECISION-MAKING STATEMENTS

- Writing *switch* Statements
  - Acts like a multiple-way *if* statement
  - Transfers control to one of several statements or blocks depending on the value of a variable
  - Used when there are more than two values to evaluate
  - Restrictions:
    - Each case evaluates a single variable for equality only
    - Variable being evaluated must be: char, byte, short, or int

WRITING LOOPS

- Loops
  - Provides for repeated execution of one or more statements until a terminating condition is reached
  - Three basic types:
    - while
    - do
    - for

WRITING LOOPS

- Writing *while* Loops
  - Loop counter
    - Counts number of times the loop is executed
  - Two kinds of loops
    - Pre-test loop
      - Tests terminating condition at the beginning of the loop
    - Post-test loop
      - Tests terminating condition at the end of the loop

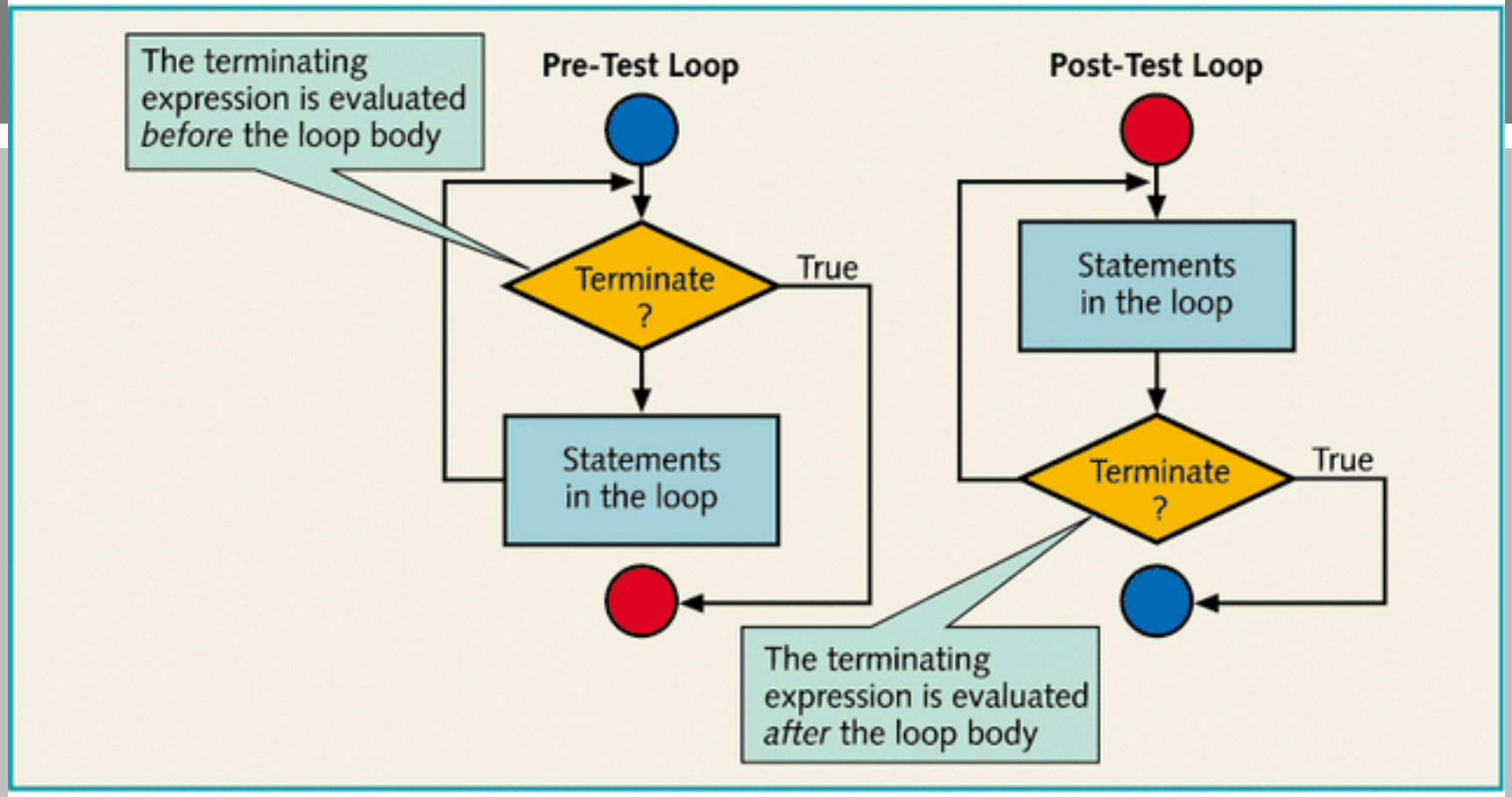

Figure 2-16 Loop structures

WRITING LOOPS

- Writing *do* Loops
  - Loop counter
    - Counts number of times the loop is executed
  - Post-test loop
    - Tests terminating condition at the end of the loop
    - Forces execution of statements in the loop body at least once

WRITING LOOPS

- Writing *for* Loops
  - Loop counter
    - Counts number of times the loop is executed
  - Pre-test loop
    - Tests terminating condition at the beginning of the loop
  - Includes counter initialization and incrementing code in the statement itself

DECLARING AND ACCESSING ARRAYS

- Arrays
  - Allows the creation of a group of variables with the same data type
  - Consist of elements:
    - Each element behaves like a variable
  - Can be:
    - One dimensional
    - Multi-dimensional

DECLARING AND ACCESSING ARRAYS

- Using One-Dimensional Arrays
  - Keyword
    - new
      - Used to create a new array instance
      - int testScores[] = new int[10];
  - Use brackets ([]) and indices to denote elements:
    - testScores[5] = 75;

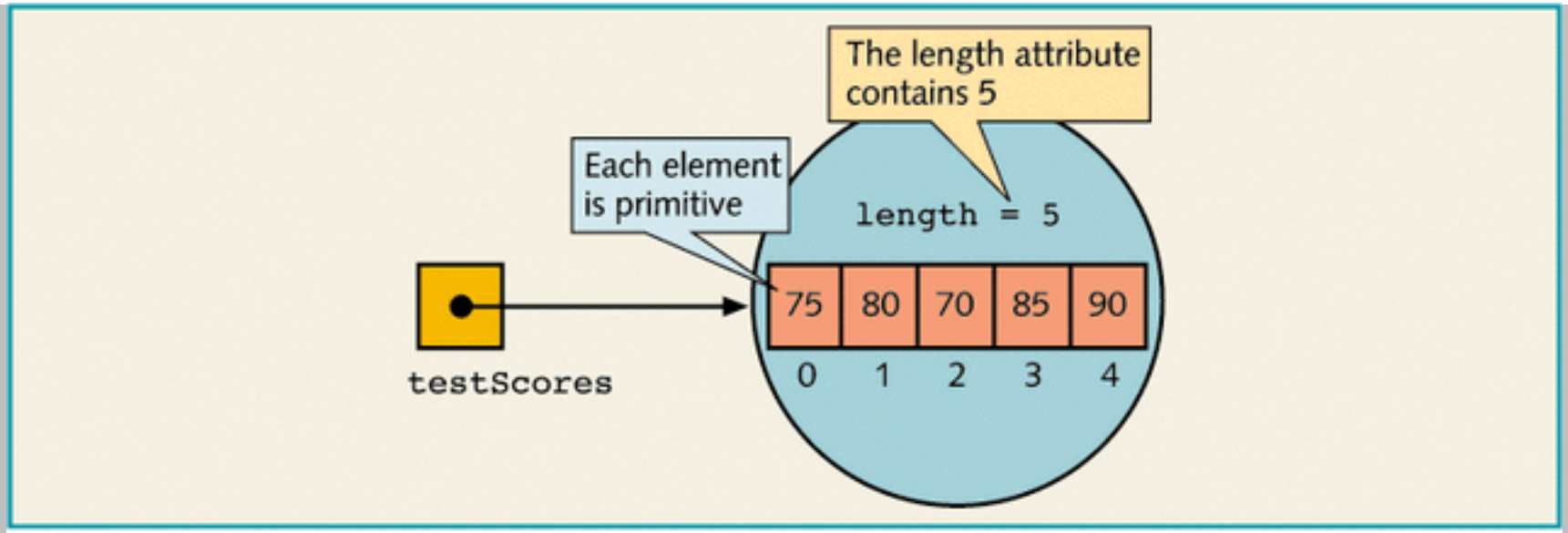

Figure 2-21 A five-element `int` array

DECLARING AND ACCESSING ARRAYS

- Using Multidimensional Arrays
  - Array of arrays
    - Three dimensions → cube
    - Four dimensions → ???
  - Each dimension has its own set of brackets:
    - testScoreTable[5][5] = 75;

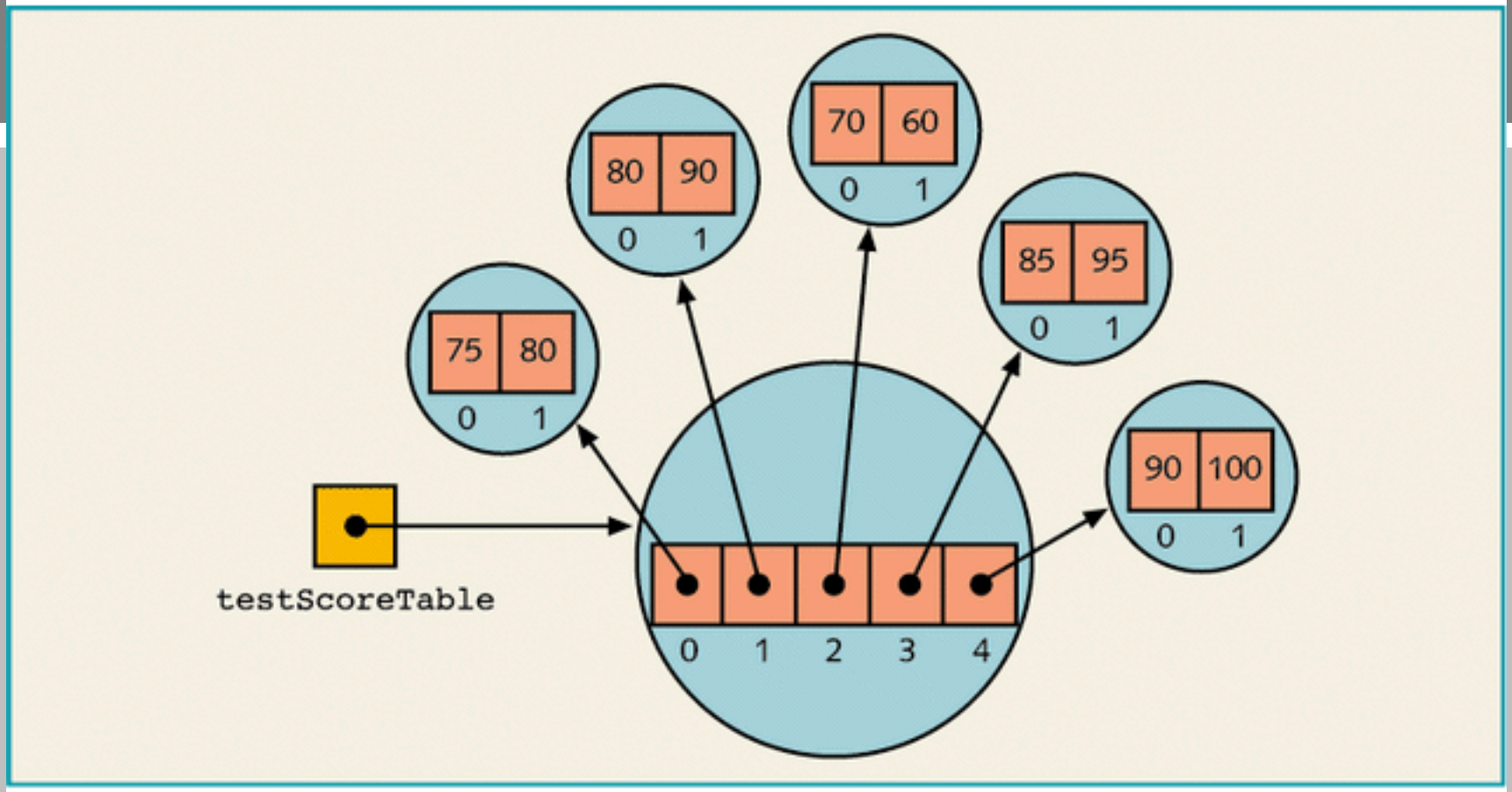

Figure 2-24 An array of arrays