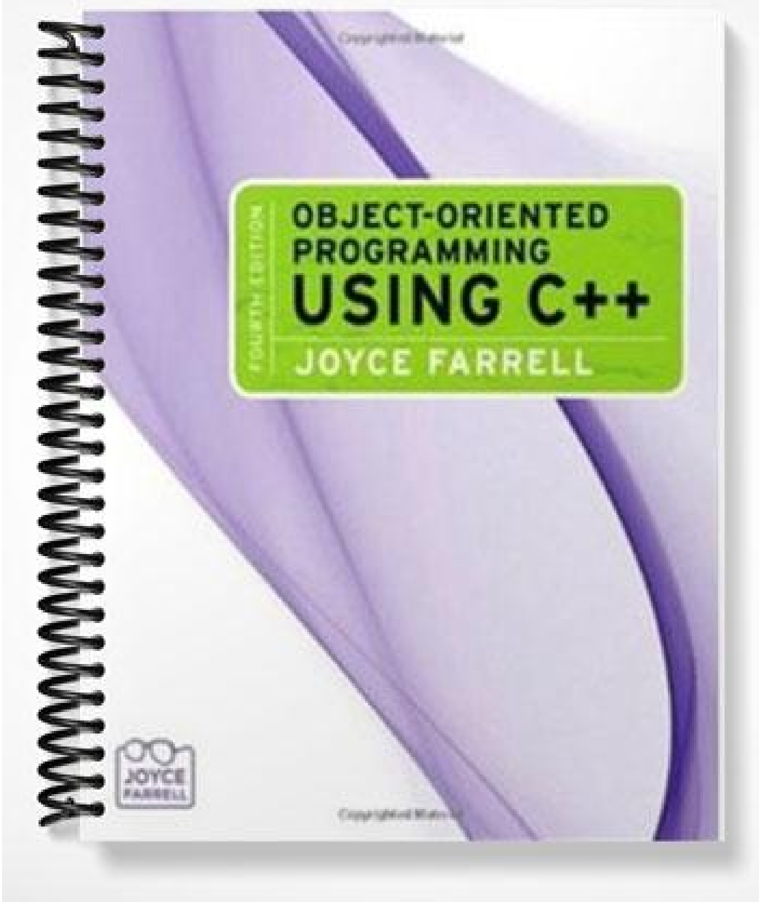
TEST BANK


FOURTH EDITION
OBJECT-ORIENTED PROGRAMMING
USING C++
JOYCE FARRELL
JOYCE FARRELL

ch2

True/False
Indicate whether the statement is true or false.

____ 1. All arithmetic operators are binary.

____ 2. The modulus operator gives the remainder of integer division; it can be used only with integers.

____ 3. When more than one arithmetic operator is included in an expression, then multiplication, division, and modulus operations always occur before addition or subtraction.

____ 4. The associativity of arithmetic operations (if there are no parentheses in the expression) is from right to left.

____ 5. When you use the += operator, you may insert a space between the + and the =.

Multiple Choice
Identify the choice that best completes the statement or answers the question.

____ 6. C++ provides ____ simple arithmetic operators for creating arithmetic expressions.
a. three
b. four
c. five
d. six

____ 7. The C++ modulus operator is ____.
a. &
b. #
c. !
d. %

____ 8. When you mix data types in a binary arithmetic expression, the values on each side of the arithmetic operator are temporarily converted to a ____ type—the data type of the value that occupies more memory and to which all the types in the expression are converted.
a. unifying
b. standard
c. common
d. large

____ 9. To ____ a value is to transform it to another data type.
a. reduce
b. cast
c. convert
d. unify

____ 10. A(n) ____ is a digit added to a number (either at the end or at the beginning) that validates the authenticity of the number.
a. checksum
b. check digit
c. error detection code
d. CRC

____ 11. You can isolate the last digit of a base-10 number by performing modulus with ____.
a. 1
b. 2
c. 10
d. 100

____ 12. Multiplication, division, and modulus are said to have higher ____ than addition or subtraction.
a. priority
b. unifying factor
c. cast value
d. arithmetic precedence

____ 13. When two operations with the same precedence appear in an arithmetic expression, the operations are carried out in order from either left to right or right to left based on their ____.

a. associativity	c. unifying factor
b. precedence	d. modulus

____ 14. The result of evaluating 3 / 4 + 2.2 is ____.

a. 2	c. 2.95
b. 2.2	d. 2.99

____ 15. The result of evaluating 3.0 / 4 + 2.2 is ____.

a. 2	c. 2.95
b. 2.2	d. 2.99

____ 16. All precedence rules can be overridden with appropriately placed ____.

a. colons	c. casts
b. semicolons	d. parentheses

____ 17. ____ is not a valid operator in C++.

a. `&`	c. `++`
b. `%=`	d. `=+`

____ 18. When an expression includes a(n) ____ operator, the mathematical operation takes place before the expression is evaluated.

a. prefix	c. unary
b. postfix	d. binary

____ 19. When an expression includes a(n) ____ operator, the mathematical operation takes place after the expression is evaluated.

a. prefix	c. unary
b. postfix	d. binary

____ 20. The address operator (____) is a C++ unary operator that is not available for use in most other programming languages.

a. `*`	c. `&`
b. `**`	d. `%`

____ 21. ____ operators evaluate the relationship between operands; you use them with Boolean expressions.

a. Relational	c. Boolean
b. Logical	d. Mathematical

____ 22. A ____ expression is one that is interpreted to be true or false.

a. binary	c. Boolean
b. mathematical	d. unary

____ 23. The expression `cout << (9 > 2);` yields ____ as output.

a. 0	c. true
b. 1	d. false

____ 24. The unary operator ____ is the not operator; it means "the opposite of," and essentially reverses the true/false value of an expression.

a. `!`	c. `%`
b. `&`	d. `$`

____ 25. !5 evaluates to ____.

a. -1
b. 0
c. 1
d. N/A

____ 26. (4 – 4) = = 0 evaluates to ____.
a. 0
b. 1
c. true
d. false

Completion

Complete each statement.

27. A(n) ____________________ is a symbol that performs arithmetic.

28. A(n) ____________________ is an operator that takes two operands, one on each side of the operator.

29. A(n) ____________________ is a deliberate cast (as opposed to an automatic one).

30. The two categories of shortcut arithmetic operators are ____________________ operators and increment and decrement operators.

31. ____________________ are those that require only one operand.

32. When you use ____________________ in front of a variable name, you refer to the memory address of that variable rather than the contents of the variable.

Matching

Match each term with the correct statement below.

a. mixed expression
b. long double
c. %=
d. char
e. associativity
f. count++
g. +
h. ++count
i. implicit cast

____ 33. for example, 3.2 * 2

____ 34. has the highest precedence of the unifying types

____ 35. has the lowest precedence of the unifying types

____ 36. is performed automatically and without your intervention

____ 37. rule that dictates the order in which an operator works with its operands

____ 38. modulus and assign operator

____ 39. prefix increment operator

____ 40. postfix increment operator

____ 41. positive value operator when used alone in front of an operand

Short Answer

42. What should you keep in mind when dividing integers?

43. Explain what the following statement means: “If you will use a calculated value later in a program, it is inefficient not to store it.”

44. Write the following operators in order of precedence, from highest to lowest:

```
float
int
doable
char
unsigned long
long double
long
short
unsigned int
```

45. Consider the following line of C++ code:

```
float g = 2.0f, h = 4.4f, i = 12.8f, floatResult;
```

What is the purpose of the lowercase f’s (e.g. 2.0f)?

46. How can you perform an explicit cast?

47. Explain why using modulus with negative numbers can lead to unexpected results.

48. What is the hexadecimal numbering system? When is it typically used?

49. What are the relational operators in C++?

50. What is a common C++ programming error when comparing two values in C++?

51. What are static fields?

ch2
Answer Section

TRUE/FALSE

1.	ANS: F	PTS: 1	REF: 52
2.	ANS: T	PTS: 1	REF: 56
3.	ANS: T	PTS: 1	REF: 58
4.	ANS: F	PTS: 1	REF: 58
5.	ANS: F	PTS: 1	REF: 59

MULTIPLE CHOICE

6.	ANS: C	PTS: 1	REF: 52
7.	ANS: D	PTS: 1	REF: 52
8.	ANS: A	PTS: 1	REF: 54
9.	ANS: B	PTS: 1	REF: 56
10.	ANS: B	PTS: 1	REF: 57
11.	ANS: C	PTS: 1	REF: 57
12.	ANS: D	PTS: 1	REF: 58
13.	ANS: A	PTS: 1	REF: 58
14.	ANS: B	PTS: 1	REF: 58
15.	ANS: C	PTS: 1	REF: 58
16.	ANS: D	PTS: 1	REF: 58
17.	ANS: D	PTS: 1	REF: 60
18.	ANS: A	PTS: 1	REF: 60-61
19.	ANS: B	PTS: 1	REF: 61
20.	ANS: C	PTS: 1	REF: 62
21.	ANS: A	PTS: 1	REF: 64
22.	ANS: C	PTS: 1	REF: 64
23.	ANS: B	PTS: 1	REF: 64
24.	ANS: A	PTS: 1	REF: 64
25.	ANS: B	PTS: 1	REF: 64
26.	ANS: B	PTS: 1	REF: 64

COMPLETION

27. ANS: arithmetic operator

    PTS: 1 REF: 52

28. ANS: binary operator

    PTS: 1 REF: 52

29. ANS: explicit cast

PTS: 1 REF: 56

30. ANS: compound assignment

PTS: 1 REF: 59

31. ANS: Unary operators

PTS: 1 REF: 60

32. ANS: &

PTS: 1 REF: 62

MATCHING

33.	ANS: A	PTS: 1	REF: 54
34.	ANS: B	PTS: 1	REF: 54
35.	ANS: D	PTS: 1	REF: 54
36.	ANS: I	PTS: 1	REF: 56
37.	ANS: E	PTS: 1	REF: 58
38.	ANS: C	PTS: 1	REF: 60
39.	ANS: H	PTS: 1	REF: 60
40.	ANS: F	PTS: 1	REF: 60
41.	ANS: G	PTS: 1	REF: 61

SHORT ANSWER

42. ANS:
When you work with a program in which you divide integers, you must keep the rules of integer division in mind if you are to avoid serious errors. For example, suppose you write a program for a legislator in which you store survey results from 1,000 voters, and 800 of the voters favor passing a specific law. When you convert the results to percentages, if you divide the integer 800 by the integer 1,000, the result is 0, not .8, because the fractional part of integer division is lost. The legislator to whom you report would be misled as to his or her constituents' wishes.

PTS: 1 REF: 53

43. ANS:
If you will use a calculated value later in a program, it is inefficient not to store it. For example, in the following code, the net-pay calculation is performed twice because the value is used in the last two output statements. It would be more efficient to declare a variable to hold the difference between salary and deductions, perform the arithmetic once, and then use the calculated variable value in both output statements.

```
double salary = 1000.00;
double deductions = 385.00;
cout<<”Salary is “<<salary;
cout<<”Deductions are “<<deductions<<endl;
cout<<”The difference is “<<(salary - deductions)<<endl;
cout<<”So, your net pay is “<<(salary - deductions)<<endl;
```

PTS: 1 REF: 53

44. ANS:

```
long double
double
float
unsigned long
long
unsigned int
int
short
char
```

PTS: 1 REF: 54

45. ANS:
The lowercase f’s following the double constant values that are assigned to the float variables g, h, and i perform an explicit cast, eliminating the warnings that would appear if they were not added.

PTS: 1 REF: 54-55

46. ANS:
You can perform an explicit cast in one of two ways: by typing `static_cast<`*`data type`*`>` in front of an expression in parentheses, or by using a type name within parentheses in front of an expression. For example, each of the following statements explicitly converts a `double` variable named `doubleVariable` to an `int` before assigning its value to `intResult`.

```
intResult = static_cast<int>(doubleVariable);
intResult = (int)doubleVariable;
```

PTS: 1 REF: 56

47. ANS:
Using modulus with negative numbers can lead to unexpected results, because there are two ways of thinking about division remainders with negative numbers. For example, –10 % 8 produces –2, but so does –10 % –8.

PTS: 1 REF: 57

48. ANS:
The hexadecimal numbering system is a numbering system based on powers of 16. It uses 16 different digits—0 through 9 and A through F. By convention, programmers use the hexadecimal system to express computer memory address no matter what programming language they are using.

PTS: 1 REF: 63

49. ANS:
== equivalent to
> greater than
>= greater than or equal to
< less than
<= less than or equal to
! = not equal to

PTS: 1 REF: 64

50. ANS:

A common C++ programming error is to use the assignment operator (=) when you should use the comparison operator (==). Making this mistake causes a logical error that can be very difficult to locate in your programs.

PTS: 1 REF: 65

51. ANS:
When you create a class instead of a structure, some fields can be used without creating an object. These fields are static fields.

PTS: 1 REF: 66