Chapter 1

1. programmed 12. key
2. CPU 13. programmer-defined symbols
3. arithmetic logic unit (ALU) and control unit 14. Operators
4. disk drive 15. Punctuation
5. operating systems and application software 16. syntax
6. instructions 17. variable
7. programming language 18. defined (or declared)
8. Machine language 19. input, processing, output
9. High-level 20. Input
10. Low-level 21. Output
11. portability 22. hierarchy chart

23. Main memory, or RAM, is volatile, which means its contents are erased when power is removed from the computer. Secondary memory, such as a disk or CD, does not lose its contents when power is removed from the computer.

24. An operating system is a set of programs that manages the computer's hardware devices and controls their processes. Application software are programs that users use to solve specific problems or perform general operations.

25. A syntax error is the misuse of a key word, operator, punctuation, or other part of the programming language. A logical error is a mistake that tells the computer to carry out a task incorrectly or to carry out tasks in the wrong order. It causes the program to produce the wrong results.

26. Hierarchy Chart:
27. **Account Balance High Level Pseudocode**
   - Have user input starting balance
   - Have user input total deposits
   - Have user input total withdrawals
   - Calculate current balance
   - Display current balance

   **Account Balance Detailed Pseudocode**
   - Input startBalance  // with prompt
   - Input totalDeposits  // with prompt
   - Input totalWithdrawals  // with prompt
   - currentBalance = startBalance + totalDeposits - totalWithdrawals
   - Display currentBalance

28. **Sales Tax High Level Pseudocode**
   - Have user input retail price
   - Have user input sales tax rate
   - Calculate tax amount
   - Calculate sales total
   - Display tax amount and sales total

   **Sales Tax Detailed Pseudocode**
   - Input retailPrice  // with prompt
   - Input salesTaxRate  // with prompt
   - taxAmount = retailPrice * salesTaxRate
   - salesTotal = retailPrice + taxAmount
   - Display taxAmount, salesTotal

29. 45
30. 7
31. 28
32. 365
33. The error is that the program performs its math operation before the user has entered values for the variables width and length.
34. Some of the questions that should be asked are:
   - What standard ceiling height should be used, or is this figure to be input?
   - How many square feet should be subtracted out for windows and doors, or do you also want this information input since it could vary by room?
   - Are the ceilings also to be painted, or just the walls?
   - How many square feet will 1 gallon of paint cover?
   - How many coats of paint will you use, or should this information be input?
Chapter 2

1. semicolon  5. braces {}
2. iostream   6. constants, or literals
3. main       7. 9.7865E14
4. #          8. 1, 2
9. B
10. A, C
11. B (C is valid, but prints the contents of variable Hello, rather than the string "Hello".)
12. B
13. A) 11     B) 14     C) 3 (An integer divide takes place.)
14. A) 9      B) 14     C) 2
15. double temp, 
         weight, 
         height;
16. int months = 2, 
         days, 
         years = 3;
17. A) d2 = d1 + 2;  
     B) d1 = d2 * 4;  
     C) c = 'K'; 
     D) i = 'K'; 
     E) i = i - 1;
18. A) d1 = d2 - 8.5; 
     B) d2 = d1 / 3.14; 
     C) c = 'F'; 
     D) i = i + 1; 
     E) d2 = d2 + d1;
19. cout << "Two mandolins like creatures in the\n\n\nCreating the agony of ecstasy.\n\n- George Barker\n\n"
20. cout << "L\n"   
    << "E\n"   
    << "A\n"   
    << "F\n";
This can also be written as a single string literal: cout << "L\nE\nA\nF\n";
21. Input weeks // with prompt
    days = weeks * 7
Display days
22. Input eggs // with prompt
cartons = eggs / 12 // perform integer divide
Display cartons

23. Input speed // with prompt
   Input time // with prompt
distance = speed * time
Display distance

24. Input miles // with prompt
   Input gallons // with prompt
milesPerGallon = miles / gallons
Display milesPerGallon

25. A)
   0
   100
B)
   8
   2
C) I am the incredible computing
   machine
   and I will
   amaze
   you.

26. A) Be careful!
   This might/n be a trick question.
B) 23
   1

27. The C-style comments symbols are backwards.
iostream should be enclosed in angle brackets.
There shouldn't be a semicolon after \texttt{int main()}. The opening and closing braces of function main are reversed.
There should be a semicolon after \texttt{int a, b, c}. The comment \texttt{\textbackslash \textbackslash Three integers} should read // Three integers.
The comment \texttt{\textbackslash \textbackslash Three integers} should read // Three integers.
cout begins with a capital letter.
The stream insertion operator (that appears twice in the \texttt{cout} statement)
   should read \texttt{<<} instead of <.
The \texttt{cout} statement uses the variable \texttt{C} instead of \texttt{c}.

28. Whatever problem a pair of students decides to work with they must determine such things as which values will be input vs. which will be set internally in the program, how much precision is required on calculations, what output will be produced by the program, and how it should be displayed. Students must also determine how to handle situations that are not clear cut. In the paint problem many of these considerations are listed in the teacher answer key (Chapter 1, Question 34). In the recipe program students must determine such things as how to handle quantities, like one egg, that cannot be halved. In the driving program, knowing distance and speed are not enough. Agreement should be reached on how to handle delays due to traffic lights and traffic congestion. Should this be an input value, computed as a percent of overall driving time, or handled some other way?
Chapter 3

1. A) cin >> description;
   B) getline(cin, description);

2. char name[35];

3. A) cin >> setw(25) >> name;
   B) cin.getline(name, 25);

4. cin >> age >> pay >> section;

5. iostream and iomanip

6. char city[31];

7. A) price = 12 * unitCost;
   B) cout << setw(12) << 98.7;
   C) cout << 12;

8. 5, 22, 20, 6, 46, 30, 0, 3, 16

9. A) a = 12 * x;
   B) z = 5 * x + 14 * y + 6 * k;
   C) y = pow(x, 4);
   D) g = (h + 12) / (4 * k);
   E) c = pow(a, 3) / (pow(b, 2) * pow(k, 4));

10. Two implicit data type conversions occur. First, because mass is a float, a copy of the int
value stored in units is promoted to a float before the multiplication operation is done. The
result of mass * units will be a float. The second data type conversion occurs when the
float result is promoted to a double in order to be stored in double variable weight.

11. 8

12. Either of these will work:
   unitsEach = static_cast<double>(qty) / salesReps;
   unitsEach = qty / static_cast<double>(salesReps);

13. const int RATE = 12;

14. x += 5;
   total += subtotal;
   dist /= rep;
   ppl *= period;
   inv -= shrinkage;
   num %= 2;

15. east = west = north = south = 1;

16. int sum = 0;
17. No, a named constant must be initialized at the time it is defined. It cannot be assigned a value at a later time.

18. `cout << fixed << showpoint << setprecision(2);`  
   `cout << setw(8) << divSales;`

19. `cout << fixed << showpoint << setprecision(4);`  
   `cout << setw(12) << profit;`

20. A) cmath  B) fstream  C) iomanip

Note: Once students understand that inputs from the keyboard should always be preceded by prompts, the // with prompt comment can be omitted from the pseudocode. Therefore, beginning with Chapter 3, we no longer include it.

21. Input score1  
   Input score2  
   Input score3  
   `average = (score1 + score2 + score3) / 3.0`  
   Display average

22. `discountPct = .15`  
   Input salesAmt  
   `amtSaved = salesAmt * discountPct`  
   `amtDue = salesAmt - amtSaved`  
   Display `amtSaved`, `amtDue`

23. Input maxCredit  
   Input creditUsed  
   `availableCredit = maxCredit - creditUsed`  
   Display `availableCredit`

24. `PI = 3.14`  
   `PRICE_PIZZA12 = 12.00`  
   `PRICE_PIZZA14 = 14.00`  
   `areaPizza12 = PI * (12 / 2)^2`  
   `areaPizza14 = PI * (14 / 2)^2`  
   `pricePerSqIn12 = PRICE_PIZZA12 / areaPizza12`  
   `pricePerSqIn14 = PRICE_PIZZA14 / areaPizza14`  
   Display `pricePerSqIn12`, `pricePerSqIn14`

25. A) Your monthly wages are 3225  // Some compilers display 3225.0000  
       B) 6 3 12  
       C) In 1492 Columbus sailed the ocean blue.
26. A) Hello George
   B) Hello George Washington
   C) Minutes: 612002.0000
       Hours: 10200.0332
       Days: 425.0014
       Months: 13.9726
       Years: 1.1644

27. A) #include <iostream> is missing.
    Each cin and cout statement starts with capital C.
    The << operator is mistakenly used with cin.
    The assignment statement should read:
       sum = number1 + number2;
    The last cout statement should have << after cout and should end with a semi-colon.
    The body of the main function should be indented within the braces.

   B) The cin statement should read:
       cin >> number1 >> number2;
    The assignment statement should read:
       quotient = static_cast<double>(number1) / number2;
    The last cout statement is missing a semicolon.
    There is no return 0;

28. A) The variables should not be declared const.
    The last cout statement is missing a semicolon.

   B) There shouldn’t be a semicolon after the #include directive.
    The function header for main should read:
       int main()
    The combined assignment operators are improperly used.
    Those statements should be:
       number1 *= 50;
       number2 *= 50;
    There is no return 0;
29. A) There shouldn’t be a semicolon after the `#include` directive.
   The function header for main should read:
   ```c
   int main()
   ```
   The variable `number` is defined, but it is called `number1` in the `cin` statement.
   The combined assignment operator is improperly used. The statement should read:
   ```c
   half /= 2;
   ```
   There is a logical error. The value divided by 2 should be `number`, not `half`.
   The results are never output.
   There is no `return 0;`

B) There shouldn’t be a semicolon after the `#include` directive.
   ```c
   name
   ```
   should be declared as a `string` or a `char` array. If declared as `string`, a
   `#include <string>` directive is needed.
   The statement `cin.getline >> name;` should read
   ```c
   cin >> name;
   ```
   The statement `cin >> go;` should read
   ```c
   cin.get(go);
   ```

30. Before the price per square inch of a pizza can be calculated, we need to know both the
    number of square inches it contains and its price. The price for each size pizza can be set at
    the beginning of the program as constants, since they are known. This can also be done with
    PI, which is needed for the pizza area calculation. We will use 3.14 for PI because that is
    precise enough for our calculations. The area of each pizza can be calculated with the
    formula \( \text{area} = \pi \times \text{radius}^2 \), where the radius of each pizza is half of its diameter.
    Now that the price of each pizza and its area are known, the price per square inch for each
    pizza can be found by dividing the price by the area.

    If you are unsure what to divide by what to get the answer, try thinking of a simple example
    using actual numbers. Suppose a pizza contained only 12 square inches and cost $12.00,
    then it would cost \( 12 / 12 \) or $1.00 per square inch. But if it were twice that big for the same
    price, it would only cost half as much per square inch. Right? Since \( 24/12 = 2.00 \) per
    square inch, that can’t be right. But \( 12 / 24 = .50 \) per square inch. That is clearly correct.
    So you can see that we need to divide the price by the square inches to get the correct result.
Chapter 4

1. relational 9. !
2. false, true 10. lower
3. false, true 11. &&
4. braces 12. ||
5. true, false 13. block (or local)
6. default 14. integer
7. false 15. break
8. true 16. 1, 0, 0, 1

17. if (y == 0)
   x = 100;

18. if (y == 10)
   x = 0;
else
   x = 1;

19. if (score >= 90)
    cout << "Excellent";
else if (score >= 80)
    cout << "Good";
else
    cout << "Try Harder";

20. if (minimum)
    hours = 10;

21. if (x < y)
    q = a + b;
else
    q = x * 2;

22. switch (choice)
{
    case 1: cout << fixed << showpoint << setprecision(2);
            break;
    case 2:
    case 3: cout << fixed << showpoint << setprecision(4);
            break;
    case 4: cout << fixed << showpoint << setprecision(6);
            break;
    default: cout << fixed << showpoint << setprecision(8);
             break;
}

23. T, F, T
24. T, F, T

25. if (grade >= 0 && grade <= 100)
    cout << "The number is valid."
;

26. if (temperature >= -50 && temperature <= 150)
    cout << "The number is valid."
;

27. if (hours < 0 || hours > 80)
    cout << "The number is not valid."
;

28. When using string objects, use the following code:
    if(title1 <= title2)
        cout << title1 << " " << title2 << endl;
    else
        cout << title2 << " " << title1 << endl;

    With using C-strings, you must replace the above if statement with:
    if (strcmp(title1, title2) <= 0)

29. if(sales < 10000)
    commission = .10;
else if (sales <= 15000)
    commission = .15;
else
    commission = .20;

30. There are several correct ways to write this. Here is one way.
    if(dept == 5 && price >= 100)
        discount = .20;
    else if (price >= 100) // but dept is not 5
        discount = .15;
    else if(dept == 5) // but price < 100
        discount = .10;
    else // dept is not 5 and price < 100
        discount = .05;

31. It should read
    if (!(x > 20))

32. It should use && instead of ||.

33. It should use || instead of &&.

34. The statement will always be true. It x equals neither 1 nor 2, it is clearly true. If x equals 1 it is true because x != 2 is true. If x equals 2 it is true because x != 1 is true. The statement should use && instead of ||.
35. A) The first cout statement is terminated by a semicolon too early.
   The definition of score1, score2, and score3 should end with a semicolon.
   The statement: if(average = 100) should read: if(average == 100)
   perfectScore is used before it is declared.
   The if (perfectScore); statement should not be terminated with a semicolon.
   The conditionally-executed block in the if statement shown above should end with a closing brace.

B) The conditionally-executed blocks in the if/else construct should be enclosed in braces.
   The statement cout << "The quotient of " << num1 <<
   should end with a semi-colon, rather than with a <<.

C) The trailing else statement should come at the end of the if/else construct.

D) A switch case construct cannot be used to test relational expressions.
   An if/else if statement should be used instead.

36. A) An if/else if is more appropriate than a switch statement when all test expressions do not involve the same variable or when test expressions need to test more than one condition, work with non-integer values, or use relational operators that test for something other than equality.

B) A switch statement is more appropriate than an if/else if statement when all tests are comparing a variable for equality with just 1 or a small set of integer values. It is a particularly useful construct to use when you want to utilize the “fall through” feature to carry out more than 1 set of actions when a particular condition is true.

C) A set of nested if/else statements is more appropriate than either of the other two constructs when the test conditions that determine the actions to be carried out do not fall into a neat set of mutually exclusive cases. For example, if one condition is true, then which set of actions you wish to take may depend on the outcome of a second test.