Chapter 1

1. a. false; b. false; c. true; d. false; e. false; f; false; g. false; h. true; i. true; j. false; k. true; l. false
2. The basic commands that a computer performs are input (get data), output (display result), storage, and performance of arithmetic and logical operations
3. Central processing unit (CPU), main memory (MM), and input/output devices.
4. Secondary storage permanently stores programs and data.
5. An operating system monitors the overall activity of the computer and provides services. Some of these services include memory management, input/output activities, and storage management.
6. The two types of programs are system programs and application programs.
7. In machine language the programs are written using the binary codes while in high-level language the program are closer to the natural language. For execution, a high-level language program is translated into the machine language while a machine language need not be translated into any other language.
8. A program written in a high-level language is called a source program.
9. Because the computer cannot directly execute instructions written in a high-level language, a compiler is needed to translate a program written in high-level language into machine code.
10. A compiler reports syntax errors.
11. Every computer directly understands its own machine language. Therefore, for the computer to execute a program written in a high-level language, the high-level language program must be translated into the computer's machine language.
12. Instructions in a high-level language are closer to a natural language, such as English, and therefore are easier to understand and learn than machine language.
13. In linking an object program is combined with other programs in the library, used in the program, to create the executable code.
14. A well-analyzed problem leads to a well-designed algorithm. Moreover, a program that is well analyzed is easier to modify as well as spot and fix errors.
15. To find the weighted average of the four test scores, first you need to know each test score and its weight. Next, you multiply each test score with its weight, and then add these numbers to get the average. Therefore,

```
1. Get testScore1, weightTestScore1
2. Get testScore2, weightTestScore2
3. Get testScore3, weightTestScore3
4. Get testScore4, weightTestScore4
5. weightedAverage = testScore1 * weightTestScore1 +
                     testScore2 * weightTestScore2 +
                     testScore3 * weightTestScore3 +
                     testScore4 * weightTestScore4;
```

16. a. Get `quarters`

    b. Get `dimes`

    c. Get `nickels`

    d. Get `pennies`

e. `changeInPennies = quarters * 25 + dimes * 10 + nickels * 5`
`+ pennies`

17. To find the price per square inch, first we need to find the area of the pizza. Then we divide the price of the pizza by the area of the pizza. Let `radius` denote the radius and `area` denote the area of the circle, and `price` denote the price of pizza. Also, let `pricePerSquareInch` denote the price per square inch.

```
a. Get radius
b. area = π * radius * radius
c. Get price
d. pricePerSquareInch = price / area
```

18. To calculate the selling price of an item, we need to know the original price (the price the store pays to buy) of the item. We can then the use the following formula to find the selling price:

`sellingPrice = (originalPrice + originalPrice × 0.80) × 0.90`

The algorithm is as follows:

```
a. Get originalPrice
b. Calculate the sellingPrice using the formula:
   sellingPrice = (originalPrice + originalPrice × 0.80) × 0.90
```

The information needed to calculate the selling price is the original price and the marked-up percentage.

19. To calculate the area of a triangle using the given formula, we need to know the lengths of the sides—`a`, `b`, and `c`—of the triangle. Next, we calculate `s` using the formula:

`s = (1/2)(a + b + c)`

and then calculate the area using the formula:

`area = sqrt(s(s-a)(s-b)(s-c))`

where `sqrt` denotes the square root.

The algorithm, therefore, is:

```
a. Get a, b, c
b. s = (1/2)(a + b + c)
c. area = sqrt(s(s-a)(s-b)(s-c))
```

The information needed to calculate the area of the triangle is the lengths of the sides of the triangle.

20. Suppose that `billingAmount` denotes the total billing amount, `numOfItemsOrdered` denotes the number of items ordered, `shippingAndHandlingFee` denotes the shipping and handling fee, and `price` denotes the price of an item. The following algorithm computes and outputs the billing amount.

```
a. Enter the number of items bought.
b. Get numOfItemsOrdered
c. billingAmount = 0.0;
d. shippingAndHandlingFee = 0.0;
e. Repeat the following for each item bought.
```

```
        i. Enter the price of the item
        ii. Get price
        iii. billingAmount = billingAmount + price;
    f. if billingAmount < 200
           shippingAndHandlingFee = 10 * numOfItemsOrdered;
    g. billingAmount = billingAmount + shippingAndHandlingFee
    i. Print billingAmount
```

21. Suppose that `numOfPages` denoes the number of pages to be faxed and `billingAmount` denotes the total charges for the pages faxed. To calculate the total charges, you need to know the number of pages faxed.

    If `numOfPages` is less than or equal to ten, the billing amount is services charges plus (`numOfPages × 0.20`); otherwise, billing amount is service charges plus `10 × 0.20` plus (`numOfPages - 10`) × `0.10`. That is,

    You can now write the algorithm as follows:

    a. Get `numOfPages`.

    b. Calculate billing amount using the formula:

```
    if (numOfPages is less than or equal to 10)
        billingAmount = 3.00 + (numOfPages × 0.20);
    otherwise
        billingAmount = 3.00 + 10 × 0.20 + (numOfPages - 10) × 0.10;
```

22. Suppose `amountWithdrawn` denotes the amount to be withdrawn, `serviceCharge` denotes the service charges, if any, and `accountBalance` denotes the total money in the account.

    You can now write the algorithm as follows:

    a. Get `amountWithdrawn`.

    b. `if amountWithdrawn > 500`

```
      Print "The maximum amount that can be drawn is $500"
    otherwise (if accountBalance <= 0)
        Print "Account balance is <= 0. You cannot withdraw any money. "
    Otherwise
    {
        if (amountWithdrawn > accountBalance)
        {
            Print "Insufficient balance. If you withdraw, services charges
                   will be $25.00. Select Yes/No."

            if (Yes)
            {
                if (amountWithdrawn > 300)
                    serviceCharge = (amountWithdrawn – 300) * 0.04;
                otherwise
                    serviceCharge = 0;
```

```
                  accountBalance = accountBalance - amountWithdrawn
                                   - serviceCharge - 25;
                  Print "Collect your money. "
              }
      }
      othwewise
      {
           if (amountWithdrawn > 300)
               serviceCharge = (amountWithdrawn - 300) * 0.04;
          otherwise
               serviceCharge = 0;

          accountBalance = accountBalance - amountWithdrawn
                           - serviceCharge;
          Print "Collect your money."
      }
```

23. Suppose `averageTestScore` denotes the average test score, `highestScore` denotes the highest test score, `testScore` denotes a test score, `sum` denote the sum of all the test scores, and `count` denotes the number of students in class, and `studentName` denotes the name of a student.

    a. First you design an algorithm to find the average test score. To find the average test score, first you need to count the number of students in the class and add the test score of each student. You then divide the sum by count to find the average test score. The algorithm to find the average test score is as follows:

       i. Set `sum` and `count` to `0`.

       ii. Repeat the following for each student in class.

           1. Get `testScore`

           2. Increment `count` and update the value of `sum` by adding the current test score to `sum`.

       iii. Use the following formula to find the average test score.

```
if (count is 0)
    averageTestScore = 0;
otherwise
    averageTestScore = sum / count;
```

    b. The following algorithm determines and prints the names of all the students whose test score is below the average test score.

       Repeat the following for each student in class:

       i. Get `studentName` and `testScore`

       ii.

```
if (testScore is less than averageTestScore)
    print studentName
```

    c. The following algorithm determines and highest test score

       i. Get first student's test score and call it `highestTestScore`.

       ii. Repeat the following for each of the remaining student in class

1. Get `testScore`
2.
```
if (testScore is greater than highestTestScore)
    highestTestScore = testScore;
```

d. To print the names of all the students whose test score is the same as the highest test score, compare the test score of each student with the highest test score and if they are equal print the name. The following algorithm accomplishes this

Repeat the following for each student in class:

i. Get `studentName` and `testScore`

ii.
```
if (testScore is equal to highestTestScore)
    print studentName
```

You can use the solutions of the subproblems obtained in parts a to d to design the main algorithm as follows:

1. Use the algorithm in part a to find the average test score.
2. Use the algorithm in part b to print the names of all the students whose score is below the average test score.
3. Use the algorithm in part c to find the highest test score.
4. Use the algorithm in part d to print the names of all the students whose test score is the same as the highest test score

Chapter 2

1. a. false; b. false; c. false; d. true; e. true; f. false; g. true; h. true; i. false; j. true; k. false
2. a, b, d, e, j
3. b, d, e
4. A keyword is a reserved word and is defined by the system. A keyword cannot be redefined in a program. A user-defined identifier can be redefined.
5. The identifiers `firstName` and `FirstName` are not the same. C++ is case sensitive. The first letter of `firstName` is lowercase `f` while the first character of `FirstName` is uppercase `F`. So these identifiers are different
6. a. `7`

   b. `3`

   c. `3`

   d. `-2`

   e. `4.4`

   f. `25.5`

   g. `26`

   h. `21.75`
7. a. `3`

   b. Not possible. Both the operands of the operator `%` must be integers. Because the second operand, `w`, is a floating-point value, the expression is invalid.

   c. Not possible. Both the operands of the operator `%` must be integers. Because the first operand, which is `y + w`, is a floating-point value, the expression is invalid .

   d. `38.5`

   e. `1`

   f. `2`

   g. `2`

   h. `420.0`
8. a, b, c, e, i, j, and k are valid;

   d, f, and g are invalid because the left side of an expression must be a variable. h is invalid because the operands of the mod operator must be integers.
9. `7`
10. Variable declarations in Lines 2, 4, 5 and 6 are correct.

    Variable declaration in Line 1 is incorrect because the left side of the assignment operator must be a variable, and the data type of the variable must be specified. A correct declaration is:

    ```
    int age = 55;                                  //Line 1
    ```
    The variable declaration in Line 3 is incorrect because strings are enclosed in double quotation marks, and the semicolon at the end of the statement is missing.

```
string message = "First test is on Monday";    //Line 3
```

11. `a` and `c` are valid

12. a.
```
int x, y;
x = 25;
y = 18;
```
b.
```
int temp = 10;
char ch = 'A';
```
c. `x = x + 5;`

d. `double payRate = 12.5;`

e. `tempNum = firstNum;`

f.
```
temp = x;
x = y;
y = temp;
```
g. `cout << x << "  " << y << "  " << x + 12 / y - 18 << endl;`

h. `char grade = 'A';`

i. `int num1, num2, num3, num4;`

j. `x = static_cast<int>(z + 0.5);`

13. a. `32 * a + b`

b. `'8'`

c. `"Julie Nelson"`

d. `(b * b – 4 * a * c) / (2 * a)`

e. `(a + b) / c * (e * f) – g * h`

f. `(–b + (b * b – 4 * a * c)) / (2 * a)`

14.
```
x = 6
y = 29
z = 3
w = -10
```

15.
```
x = 28
y = 35
z = 1
w = 22.00
t = 6.5
```

16. a. `x = 2, y = 5, z = 6`

b. `x + y = 7`

c. `Sum of 2 and 6 is 8`

d. `z / x = 3`

e. `2 times 2 = 4`

17. a. `0.50`

 b. `24.50`

 c. `37.6`

 d. `8.3`

 e. `10`

 f. `38.75`

18. a. `cout << endl;` or `cout << "\n";` or `cout << '\n';`

 b. `cout << "\t";`

 c. `cout << "\"";`

19. a and c are correct

20. a. `firstName`

 b. `discountedPrice`

 c. `numOfJuiceBottles`

 d. `milesTravelled`

 e. `highestTestScore`

21. a.
```
int num1;
int num2;
```

 b. `cout << "Enter two numbers separated by spaces." << endl;`

 c. `cin >> num1 >> num2;`

 d.
```
cout << "num1 = " << num1 << "num2 = " << num2
       << "2 * num1 - num2 = " << 2 * num1 - num2 << endl;
```

22. A correct answer is:

```
#include <iostream>

using namespace std;

const int TOP_NUM = 753409;
const double PAY_RATE = 18.35;

int main()
{
    int testScore, projectScore;
    double temp;
    double payCheck;
    int newTemp;
    int first;
    double hoursWorked;


    testScore = 88;
    projectScore = 22;

    cout << testScore << " " << projectScore << endl;
```

```
        temp = 82;
        newTemp = testScore + 2 * projectScore;

        first = 2 * TOP_NUM;

        cout << first << " " << TOP_NUM << endl;

        cout << "Enter hours worked: ";
        cin >> hoursWorked;
        cout << endl;

        payCheck = hoursWorked * PAY_RATE;

        cout << "Wages = " << payCheck << endl;

       return 0;
    }
```

23. A correct answer is:

```
    #include <iostream>

    using namespace std;

    const char STAR = '*';
    const int PRIME = 71;

    int main()
    {
       int count, sum;
       double x;

       int newNum;   //declare newNum

       count = 1;
       sum = count + PRIME;
       x = 25.67;   // x = 25.67;
       newNum = count * 1 + 2; //newNum = count * ONE + 2;
       sum = sum + count;   //sum + count = sum;
       x = x + sum * count; // x = x + sum * COUNT;
       cout << " count = " << count << ", sum = " << sum
            << ", PRIME = " << PRIME << endl;
       return 0;
    }
```

24. A correct answer is:

```
  #include <iostream>
  #include <string>

  using namespace std;

  int main()
  {
      int temperature;    // int temp;
      string first, last; // string first;
      cout << "Enter first name: ";  // cout << "Enter first name: ";
      cin >> first;  // cin >> first
      cout << endl;

      cout << "Enter last name: ";  // cout << "Enter last name: ;
      cin >> last;
      cout << endl;

      cout << "Enter today’s temperature: ";
      cin >> temperature;
      cout << endl;

      // cout << first << " " << last << today’s temperature is: ";
      //     << temperature << endl;

      cout << first << " " << last << " today’s temperature is: "
           << temperature << endl;


      return 0;
  }
```

25. An identifier must be declared before it can be used.

26. b.

27. a. `x *= 2;`
    b. `x += y - 2;`
    c. `sum += num;`
    d. `z *= x + 2;`
    e. `y /= x + 5;`

28. a. `x = x + 5 – z;`
    b. `y = y * (2 * x + 5 – z);`
    c. `w = w + 2 * z + 4;`
    d. `x = x – (z + y – t);`
    e. `sum = sum + num;`

29.

	a	b	c
`a = (b++) + 3;`	9	7	und
`c = 2 * a + (++b);`	9	8	26
`b = 2 * (++c) – (a++);`	10	45	27

30.

	a	b	c	sum
`sum = a + b + c;`	3	5	14.1	22
`c /= a;`	3	5	4.7	22
`b += c – a;`	3	6	4.7	22

```
a *= 2 * b + c;             50    6    4.7    22
```

31. (The user input is shaded.)

```
a = 25
Enter two integers: 20 15

The numbers you entered are 20 and 15
z = 45.5
Your grade is A
The value of a = 65
```

32. (The user input is shaded.)

```
Enter last name: Miller

Enter a two digit number: 34

Enter a positive integer less than 1000: 340

Name: Miller
Id: 3417
Mystery number: 3689
```

33.

```
#include <iostream>
#include <string>

using namespace std;

const double X = 13.45;
const int Y = 34;
const char BLANK = ' ';

int main()
{
    string firstName, lastName;
    int num;
    double salary;

    cout << "Enter first name: ";
    cin >> firstName;
    cout << endl;

    cout << "Enter last name: ";
    cin >> lastName;
    cout << endl;

    cout << "Enter a positive integer less than 70: ";
    cin >> num;
    cout << endl;

    salary = num * X;

    cout << "Name: " << firstName << BLANK << lastName << endl;
```

```
        cout << "Wages: $" << salary << endl;
        cout << "X = " << X << endl;
        cout << "X + Y = " << X + Y << endl;

        return 0;
    }
```

34. The program requires four inputs in the following order:

```
string decimal_number decimal_number integer
```

Chapter 3

1. a. true; b. true; c. false; d. false; e. true; f. true
2. a. `num1 = 47, num2 = 8, symbol = '1'`

   b. `num1 = 7, num2 = 18, symbol = '4'`

   c. `num1 = 47, num2 = 18, symbol = ' '`

   d. `num1 = 47, num2 = 18, symbol = ' '`

   e. `num1 = 7, num2 = 18, symbol = '4'`
3. a. `x = 37, y = 86, z = 0.56`

   b. `x = 37, y = 32, z = 86.56`

   c. Input failure: `z = 37.0, x = 86,` trying to read the `.` (period) into `y`.
4. a. `x = 38, y = 26, symbol = '2'`

   b. `x = 38, y = 67, symbol = ' '`

   c. `x = 24, y = 38, symbol = '$'`

   d. `x = 67, y = 24, symbol = '3'`

   e. `x = 24, y = 63, symbol = '$'`
5. Input failure: Trying to read `A` into `y`, which is an `int` variable. `x = 46, y = 18, and z = 'A'`. The values of `y` and `z` are unchanged.
6. a. `x = 35, y = 62, ch = '.', z = 78.0`

   b. `x = 86, y = 32, ch = 'A', z = 92.6`

   c. `x = 12,` input failure, trying to read `'.'` into `y`, which is an `int` variable.

7. `iomanip`

8. `cmath`

9. `cmath`

10.
```
16
15
32
6
243
Length of message = 29
```

11. To use the function `putback`, the program must include the header file `iomanip`. To use the function `peek`, the program must include the header file `iostream`.

12. a. `num = 34, discard = '#'`

    b. Input failure. After peeking into the input stream, `cin` tries to input `#` into `num`. However, `num` is an `int` variable, so the stream enters the fail state.

    c. `num = 34, discard = '#'`

13. `getline(cin, name);`

14. `cout << setfill('*') << setw(35) << '*' << endl;`

15. a. `name = " Lance Grant", age = 23`

    b. `name = " ", age = 23`

16. a. `name = "Lance Grant", age = 23`

    b. `name = "Lance Grant", age = 23`

17.
```
#include <iostream>
#include <fstream>

using namespace std;

int main()
{
    int num1, num2;
    ifstream infile;
    ofstream outfile;

    infile.open("input.dat");
    outfile.open("output.dat");

    infile >> num1 >> num2;
    outfile << "Sum = " << num1 + num2 << endl;

    infile.close();
    outfile.close();

    return 0;
}
```

18. Invalid data may cause the input stream to enter the fail state. When an input stream enters the fail state, all further inputs associated with that input stream are ignored. The program continues to execute with whatever values the variables have.

19. `fstream`

20. `infile.open("employee.dat ");`

21. a. Same as before.

    b. The file contains the output produced by the program.

    c. The file contains the output produced by the program. The old contents are erased.

    d. The program would prepare the file and store the output in the file.

22.
```
infile >> acctNumber;
infile >> accountType;
infile >> balance;
```

23. a. `outfile.open("travel.dat ");`

    b. `outfile >> fixed >> showpoint >> setprecision(2);`

    c. `outfile >> day >> " " >> distance >> " " >> speed >> endl;`

    d.
    ```
    travelTime = distance / speed;
    outfile >> travelTime;
    ```

    e. `fstream` and `iomanip`.

Chapter 4

1. a. false; b. false; c. false; d. true; e. false; f. false; g. false; h. false; i. false; j. true
2. a. 0 (false) b. 1 (true) c. 0 (false) d. 1 (true)
   e. 1 (true) f. 0 (false)
3. a. true; b. false; c. true; d. true; e. false
4. a. true; b. true; c. false; d. true
5. a. `x = y: 0`

   b. `x != z: 1`

   c. `y == z - 3: 1`

   d. `!(z > w): 0`

   e. `x + y < z: 0`

6. a and b.

7. a.
```
%%
```
b.
```
10 2 * 5
```
c.
```
A
```
d.
```
C--
```
e.
```
Sam Tom
Tom Sam
```
f.
```
-6
**
```

8. Only a

9. a.
```
R&
```
b.
```
1 2 3 4
$$
```
c.
```
Jack Accounting
John Business
```

10. `The value of found is: 0`

11. `The value of found is: 1`

12.
```
25
0
1
1
1
```

13. Omit the semicolon after `else`. The correct statement is:
```
if (score >= 60)
    cout << "You pass." << endl;
else
    cout << "You fail." << endl;
```

14.
```
if (gender == 'M')
    cout << "Male" << endl;
else if (gender == 'F')
```

```
        cout << "Female" << endl;
    else
        cout << "Invalid gender code." << endl;
```

15. The correct code is:

```
    if (0 < numOfItemsBought && numOfItemsBought < 5)
        shippingCharges = 5.00 * numOfItemsBought;
    else if (5 <= numOfItemsBought && numOfItemsBought < 10)
        shippingCharges = 2.00 * numOfItemsBought;
    else
        shippingCharges = 0.0;
```

16. `2 20 404`

17. `3 1`

18. a. `2 2`

    b. `1 3`

19.
```
    if (sale > 20000)
        bonus = 0.10
    else if (sale > 10000 && sale <= 20000)
        bonus = 0.05;
    else
        bonus = 0.0;
```

20.
```
    if (0 < overSpeed && overSpeed <= 5)
        fine = 20.00;
    else if (5 < overSpeed && overSpeed <= 10)
        fine = 75.00;
    else if (10 < overSpeed && overSpeed <= 15)
        fine = 150.00;
    else if (overSpeed > 15)
        fine = 150.00 + 20.00 * (overSpeed - 15);
```

21. a. The output is: `Discount = 10%`. The semicolon at the end of the `if` statement terminates the `if` statement. So the `cout` statement is not part of the `if` statement. The `cout` statement will execute regardless of whether the expression in the `if` statement evaluates to `true` or `false`.
    b. The output is: `Discount = 10%`. The semicolon at the end of the `if` statement terminates the `if` statement. So the `cout` statement is not part of the `if` statement. The `cout` statement will execute regardless of whether the expression in the `if` statement evaluates to `true` or `false`.

22. a. (i) The output is: `Grade is C`. The value of `score` after the `if` statement executes is `70`.
    (ii) The output is: `Grade is C`. The value of `score` after the `if` statement executes is `70`.

    b. (i) No output. The value of `score` after the `if` statement executes is `80`.
    (ii) The output is: `Grade is C`. The value of `score` after the `if` statement executes is `70`.

23. a. `(x >= y) ? z = x - y : z = y - x;`

    b.
```
    (hours >= 40.0) ? wages = 40 * 7.50 + 1.5 * 7.5 * (hours - 40)
                    : wages = hours * 7.50;
```
    c. `(score >= 60) ? str = "Pass" : str = "Fail";`

24. a.
```
    if (x < 5)
        y = 10;
```

```
     else
         y = 20;
b. if (fuel >= 10)
         drive = 150;
   else
         drive = 30;
c. if (booksBought >= 3)
       discount = 0.15;
   else
       discount = 0.0;
```

25. a. 40.00
    b. 40.00
    c. 55.00

26. a, c, and d are valid. b is invalid; a `case` value cannot appear more than once and there should be a colon after a case value.

27.	a. 16	b. 3	c. 18	d. 23
28.	a. 14	b. -10	c. 0	d. 17
29.	a. 3	b. -20	c. 3	d. 5

30. A correct code is:

```
#include <iostream>

using namespace std;

int main()
{
    int num1, num2;
    int temp;
    bool found;

    cout << "Enter two integers: ";
    cin >> num1 >> num2;
    cout << endl;

    if (num1 >= num2  &&  num2 > 0)
        switch (num1 % num2)
        {
        case 1:
            found = (num1 / num2) >= 6;
            break;
        case 2: case 3:
            num1 = num2 / 2;
            break;
        default:
            num2 = num1 * num2;
        }
    else
    {
        found = (2 * num2 < num1);
        if (found)
        {
            cout << "Enter an integers: ";
            cin >> num2;
            cout << endl;
```

```
                num1 = num2 - num1;
                temp = (num1 + num2) / 10;
                if (num2)
                {
                    num1 = num2;
                    num2 = temp;
                }
            }
        }

        cout << num1 << " " << num2 << endl;

        return 0;
    }
```

a. 4 8
b. 4 9

31.
```
#include <iostream>

using namespace std;
const int SECRET = 5;

int main()
{
    int x, y, w, z;

    z = 9;

    if (z > 10)
    {
        x = 12;
        y = 5;
        w = x + y + SECRET;
    }
    else
    {
        x = 12;
        y = 4;
        w = x + y + SECRET;
    }

    cout << "w = " << w << endl;

    return 0;
}
```

32.
```
#include <iostream>

using namespace std;

int main()
{
    double firstNum, secondNum;

    cout << "Enter two nonzero numbers: ";
    cin >> firstNum >> secondNum;
```

```
        cout << endl;

        if (firstNum == 0 || secondNum == 0)
            cout << "Both the numbers must be nonzero." << endl;
        else if (firstNum > secondNum)
            cout << firstNum / secondNum << endl;
        else if (firstNum < secondNum)
            cout << secondNum / firstNum << endl;
        else
            cout << firstNum * secondNum << endl;

        return 0;
    }
```

33.

```
switch (classStanding)
{
case 'f':
    dues = 150.00;
    break;
case 's':
    if (gpa >= 3.75)
        dues = 75.00;
    else
        dues = 120.00;
        break;
case 'j':
    if (gpa >= 3.75)
        dues = 50.00;
    else
        dues = 100.00;
    break;
case 'n':
    if (gpa >= 3.75)
        dues = 25.00;
    else
        dues = 75.00;
    break;
default:
    cout << "Invalid class standing code." << endl;
}
```

34. Suppose that we have the following variables:

```
double billingAmount;
double payment;  //payment made by the customer
double credit;   //credit for the next bill
double penalty;  //penalty to be added the next month's bill
double balance; //unpaid balance
double paymentPercent; //percent of the billing amount paid
```

The following algorithm determines the credit or penalty, and the unpaid balance.

1. Prompt the user to enter the billing amount.
2. Input the billing amount into the variable `billingAmount`.
3. Prompt the user to input the payment.

4. Input the payment into the variable `payment`.
5. Determine the unpaid balance using the formula:
```
    balance = billingAmount - payment;
```
6. Determine the percent of the billing amount made by the customer:
```
if (billingAmount != 0.0)
     paymentPercent = payment / billingAmount;
else
     paymentPercent = 0.0;
```
7. Determine the credit or the penalty, and the unpaid balance including penalty, if any, using the following `if/else` statement.

```
if (paymentPercent == 1.0)
{
     credit = billingAmount * 0.01;

     if (credit > 10.00)
         credit = 10.00;
}
else
{
     if (paymentPercent >= 0.50)
         penalty = balance * 0.05;
     else if (paymentPercent >= 0.20 && paymentPercent < 0.50)
         penalty = balance * 0.10;
     else
         penalty = balance * 0.20;

     balance = balance + penalty;
}
```