Program Definition:

1. Input: 3 integer test grades from the keyboard
2. Output: On the monitor, a grade report showing:
   the 3 grades and
   the sum and average of the grades
3. Processing: Compute sum and average

Program Logic: (using pseudocode)

1. Input the 3 test grades
2. Compute sum and average of the 3 grades
3. Output the grade report

Program Listing:

// Compute average grade for one student - Version 1
#include <iostream> // for cin, cout
using namespace std;

int main()
{
    int grade1, grade2, grade3, grade_sum;
    double average_grade;

    // Input the three test grades
    cout << "Enter the three test grades,\n";
    cout << "separated by one or more spaces.\n";
    cin >> grade1 >> grade2 >> grade3; // Unfriendly data entry

    // Compute sum and average of the three grades
    grade_sum = grade1 + grade2 + grade3;
    average_grade = grade_sum / 3.0;

    // Output the student report
    cout << '\nTest grades: ' << grade1 < ' ' < grade2 < ' ' < grade3;
    cout << '\nGrade sum: ' << grade_sum;
    cout << '\nGrade average: ' << average_grade;
    cout << '\n\nEnd of Grade Report\n";
    return 0; // Return value of 0 indicates Success
}

Run 1:
Enter the three test grades,
separated by one or more spaces.
100 76 83
Test grades: 100 76 83
Grade sum: 259
Grade average: 86.3333
End of Grade Report
Run 2:
Enter the three test grades, separated by one or more spaces.
100 100 100
Test grades: 100 100 100
Grade sum: 300
Grade average: 100
End of Grade Report

Run 3:
Enter the three test grades, separated by one or more spaces.
0 100 50
Test grades: 0 100 50
Grade sum: 150
Grade average: 50
End of Grade Report

Memory for Run 1:

<table>
<thead>
<tr>
<th>average_grade</th>
<th>86.333 33333 33333</th>
</tr>
</thead>
<tbody>
<tr>
<td>grade_sum</td>
<td>259</td>
</tr>
<tr>
<td>grade3</td>
<td>83</td>
</tr>
<tr>
<td>grade2</td>
<td>76</td>
</tr>
<tr>
<td>grade1</td>
<td>100</td>
</tr>
</tbody>
</table>

Note:
1. header and step comments
2. source code margins
3. default mode for floating-point output;
   In default mode, precision is the maximum number of digits printed. The default precision is six.
1. **char** is short for character; a single character

2. Variable declaration:    
   ```
   char grade; // Stores one character
   char grade = 'A'; // With initial value
   ```
   
   Note the **delimiters** (markers) for a **char constant** are single quotes.

3. ```
   char a, b;
   cin >> a >> b; // char input
   ```
   
   Skips leading whitespace characters.

   **Data entry examples:**
   ```
   Buffer Contents:
   ```
   ```
   a. #$<Enter> #$
      #$\n
   b. ^^^#^^^^$<Enter> ^^^#^^^^$
      ^^^#^^^^$\n
   c. ^^^^#^^^<Enter> ^^^^#^^^$
      ^^^^#^^^\n      ^^^$<Enter>
   ```

4. **String constant:** one or more characters enclosed in double quotes, the delimiters.

   ```
   cout << "Hi^There\n";
   ```

   **Length:** The count of all characters between the double quotes.

   ```
   String constant   Length
   "49ers"           5
   "Let's^go."      8
   "^^^1234^^89^^"   9
   " Wow!  "        6
   ```
5. Escape Sequences in strings: \ is the escape character

\n newline character
\t tab

Escape the usual meaning of the character.

10 \tcout << "\n\tHi\n"
20 \t"\n\ttHi\n";
30 \tcout << "\"Hi There\\"\";

Print position 1 is a tab stop. The next tab stop is 8 positions from the current stop. Thus, the first few tab stops are:

1  9  7  5  3  1
   Hi
   (one blank line)
   Hi
"Hi There"

C++ concatenates multiple consecutive strings that are separated only by whitespace.
// Compute average grade for one student - Version 2

#include <iostream> // for cin, cout
using namespace std;

int main()
{
    const double NUMBER_OF_TESTS = 3.0; // Declared constant
    int grade1, grade2, grade3, grade_sum;
    double average_grade;
    char student_name[80]; // char array to store a string

    cout << "Midterm Grade Report\n";
    cout << "Enter student's name ==> ";
    cin.getline(student_name, 80);
    cout << "Enter first grade =====> ";
    cin >> grade1;
    cout << "Enter second grade =====> ";
    cin >> grade2;
    cout << "Enter third grade =====> ";
    cin >> grade3;

    grade_sum = grade1 + grade2 + grade3;
    average_grade = grade_sum / NUMBER_OF_TESTS;

    cout.setf(ios::fixed); // not in e-notation
    cout.setf(ios::showpoint); // print decimal point and trailing zeros
    cout.precision(1); // print 1 fractional digit
    cout << "\nStudent's name: " << student_name;
    cout << "\nTest grades: " << grade1 << " " << grade2 << " " << grade3;
    cout << "\nGrade sum: " << grade_sum;
    cout << "\nGrade average: " << average_grade;
    cout << "\n\nEnd of Midterm Grade Report\n";
    return 0;
}

const Qualifier:
1. const int CURRENT_YEAR = 2009;

cout << 2009; vs. cout << CURRENT_YEAR;

const is keyword that qualifies the variable declaration

Declared constant:
a) must be initialized const int MAX;
b) cannot be changed MAX = 1000;
c) use UPPERCASE to indicate a constant
2. Advantages:
   Name can suggest what the constant represents; Program is easy to read
   Easy to change the constant

Run 1:
Midterm Grade Report
Enter student's name ==> Carol Wong
Enter first grade =====> 100
Enter second grade =====> 76
Enter third grade =====> 83

Student's name: Carol Wong
Test grades: 100 76 83
Grade sum: 259
Grade average: 86.3
End of Midterm Grade Report

Run 2:
Midterm Grade Report
Enter student's name ==> Jim Garcia
Enter first grade =====> 100
Enter second grade =====> 100
Enter third grade =====> 100

Student's name: Jim Garcia
Test grades: 100 100 100
Grade sum: 300
Grade average: 100.0
End of Midterm Grade Report

Run 3:
Midterm Grade Report
Enter student's name ==> Bich Tran
Enter first grade =====> 0
Enter second grade =====> 100
Enter third grade =====> 50

Student's name: Bich Tran
Test grades: 0 100 50
Grade sum: 150
Grade average: 50.0
End of Midterm Grade Report