# True or False

1. Mark the following statements as true or false.  
     
   a. The extraction operator >> skips all leading whitespace characters when searching for the next data in the input stream.

**True**

b. In the statement cin >> x;, x must be a variable.

**True**

c. The statement cin >> x >> y; requires the input values for x and y to  
appear on the same line.

**False**

d. The statement cin >> num; is equivalent to the statement num >> cin;.

**False**

e. You generate the newline character by pressing the Enter (return) key on the keyboard.

**True**

f. The function ignore is used to skip certain input in a line.

**True**

# Reading input and assignment of values to variables

1. Suppose x and y are int variables and ch is a char variable. Consider the  
   following input:  
   5 28 36  
   What value (if any) is assigned to x, y, and ch after each of the following  
   statements executes? (Use the same input for each statement.)  
   a. cin >> x >> y >> ch;

**x = 5;**

**y = 28;**

**ch = ‘3’;**

b. cin >> ch >> x >> y;

**x = 28;**

**y = 36;**

**ch = ‘5’**

c. cin >> x >> ch >> y;

**x = 5;**

**y = 8;**

**ch = ‘2’;**

d. cin >> x >> y;  
 cin.get(ch);

**x = 5;**

**y = 28;**

**ch = ‘ ’;**

1. Suppose x and y are int variables and z is a double variable. Assume the following input data:  
   37 86.56 32  
   What value (if any) is assigned to x, y, and z after each of the following statements executes? (Use the same input for each statement.)  
   a. cin >> x >> y >> z; **x = 37, y = 86, z = 0.56**  
   b. cin >> x >> z >> y; **x = 37, y = 32, z = 86.56**  
   c. cin >> z >> x >> y; **x = 86, y = 0, z = 37**
2. Suppose x and y are int variables and ch is a char variable. Assume the following input data:  
   13 28 D  
   14 E 98  
   A B 56  
   What value (if any) is assigned to x, y, and ch after each of the following statements executes? (Use the same input for each statement.)  
   a. cin >> x >> y;  
    cin.ignore(50, '\n');  
    cin >> ch;

**x = 13;**

**y = 28;**

**ch = ‘1’;**

b. cin >> x;  
 cin.ignore(50, '\n');  
 cin >> y;  
 cin.ignore(50, '\n');  
 cin.get(ch);

**x = 13;**

**y = 14;**  
**ch = ‘A’;**

c. cin >> y;  
 cin.ignore(50, '\n');  
 cin >> x >> ch;

**x = 14;**

**y = 13;**  
**ch = ‘E’;**

d. cin.get(ch);  
 cin.ignore(50, '\n');  
 cin >> x;  
 cin.ignore(50, 'E');  
 cin >> y;

**x = 14;**

**y = 98;**  
**ch = ‘1’;**

1. Given the input:  
   46 A 49  
   and the C++ code:  
   int x = 10, y = 18;  
   char z = '\*';  
   cin >> x >> y >> z;  
   cout << x << " " << y << " " << z << endl;  
   What is the output?

**46 0 \***

Suppose that x and y are int variables, z is a double variable, and ch is a  
char variable. Suppose the input statement is:  
cin >> x >> y >> ch >> z;  
What values, if any, are stored in x, y, z, and ch if the input is:  
a. 35 62.78 **x = 35, y = 62, ch = ‘.’, z = 0.78**   
b. 86 32A 92.6 **x = 86, y = 32, ch = ‘A’, z = 92.6**  
c. 12 .45A 32 **x = 12, y = 0, ch = undefined, z = undefined**

# Manipulators

1. Which header file must be included to use the function setprecision?

**iomanip**

1. Which header file must be included to use the function pow?

**cmath**

1. Suppose that name is a variable of type string. Write the input statement to read and store the input Brenda Clinton in name. (Assume that the input is from the standard input device.)

**getline(cin, name);**

1. Write a C++ statement that uses the manipulator setfill to output a line  
   containing 35 stars, as in the following line:  
   \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**cout << setfill(‘\*’) << setw(35);**

1. What output will be produced when the following lines are executed (assuming the lines are embedded in a complete and correct program with the proper include directives)?  
   cout << "\*";  
   cout.width(5);  
   cout << 123 << "\*" << 123 << "\*" << endl;  
   cout << "\*" << setw(5) << 123 << "\*" << 123 << "\*" << endl;

**\* 123\*123\***

**\* 123\*123\***

1. What output will be produced when the following lines are executed (assuming the lines are embedded in a complete and correct program with the proper include directives)?  
   cout << "\*" << setw(5) << 123;  
   cout.setf(ios::left);  
   cout << "\*" << setw(5) << 123;  
   cout << right;  
   cout << "\*" << setw(5) << 123 << "\*" << endl;

**\* 123\*123 \* 123\***

1. What output will be produced when the following lines are executed (assuming the lines are embedded in a complete and correct program with the proper include directives)?   
   cout << "\*" << setw(5) << 123 << "\*" << 123 << "\*" << endl;  
   cout << showpos;  
   cout << "\*" << setw(5) << 123 << "\*" << 123 << "\*" << endl;

cout.unsetf(ios::showpos);  
cout.setf(ios::left);  
cout << "\*" << setw(5) << 123 << "\*" << setw(5) << 123

<< "\*" <<endl;

**\* 123\*123\***

**\* +123\*+123\***

**\*123 \*123 \***

# Input with Strings and Chars

1. Suppose that age is an int variable and name is a string variable. What are the values of age and name after the following input statements execute:  
   cin >> age;  
   getline(cin, name);  
   if the input is:  
   a. 23 Lance Grant **=> age = 23, name = “Lance Grant”**

b. 23 **=> age = 23, name = “”**

Lance Grant

1. Suppose that age is an int variable, ch is a char variable, and name is a string variable. What are the values of age and name after the following input statements execute

cin >> age;  
cin.get(ch);  
getline(cin, name);  
if the input is:  
a. 23 Lance Grant **=> age = 23, ch = ‘ ’, name = “Lance Grant”**

b. 23 **=> age = 23, ch = ‘\n’, name = “Lance Grant”**

Lance Grant

**Recall:** that cin statements ignore all whitespace characters. If we want to actually read in a whitespace character we need to use cin.get(). We have two versions:

cin.get(); //we can output the character we get, or //use an = to assign to a variable

cin.get(c); //we can output the character and assign it //to a variable

1. Suppose c is a variable of type *char*. What is the difference between the following two statements?  
   cin >> c;  
   and  
   cin.get(c);

**In the first statement, any whitespace in the stdin will be ignored and only the first non-whitespace char will be assigned to c. The second statement will not ignore any whitespace.**

1. Consider the following code (and assume that it is embedded in a complete and correct program and then run):  
   charc1, c2, c3, c4;  
   cout << "Enter a line of input:\n";  
   cin.get(c1);  
   cin.get(c2);  
   cin.get(c3);  
   cin.get(c4);  
   cout << c1 << c2 << c3 << c4 << "END OF OUTPUT";  
   If the dialogue begins as follows, what will be the next line of output?

Enter a line of input: **a b c d e f g**

**a b END OF OUTPUT**