



For questions 8 through 11, assume the following variables have been declared

```
int    anInt;
double aDble;
char   aChar;
```

and assume the standard input stream `cin` contains the following values, separated by tabs:

1.2	4.5	A	-46.32
-----	-----	---	--------

Determine the value of the indicated variable after the execution of the given statement; each question is independent, that is, each starts with the stream contents shown above.

8. `aChar` after `cin >> anInt` // extracts 1, and stops before the '.'  
`>> aChar;` // extracts the next one character, '.'
- 1) 4  
 2) '4'  
 3) '\t' (a tab)  
 4) '.'  
 5) None of these
9. `aChar` after `cin >> aDble` // extracts 1.2, and stops before the tab  
`>> aChar;` // discards tab, extracts next one character
- 1) 4  
 2) '4'  
 3) '\t' (a tab)  
 4) '.'  
 5) None of these
10. `anInt` after `cin >> aDble` // extracts 1.2, as above  
`>> anInt;` // discards tab, extracts next int value
- 1) 2  
 2) '4'  
 3) 4  
 4) 4.5  
 5) None of these
11. `anInt` after `cin >> anInt;` // extracts first int value, 1  
`cin.get(aChar);` // gets next single character, '.'  
`cin >> anInt;` // extracts next int value, 2
- 1) 1  
 2) 2  
 3) 4  
 4) 5  
 5) None of these

12. What is printed by the statement: `cout << "The answer is" << setw(3) << 30 + 12;`
- 1) The answer is 30 + 12  
 2) The answer is42  
 3) The answer is 42  
 4) The answer is30 + 12  
 5) None of these

13. Assuming that all variables are of type `double`, the correct C++ expression for  $\frac{(a+b)c}{d+e}$  is:
- 1) `a + b * c / d + e`  
 2) `(a + b) * c / d + e`  
 3) `(a + b) * c / (d + e)`  
 4) `(a + b * c) / d + e`  
 5) None of these



18. A program specification says that a line of input will start with a text label, followed by a tab character, followed by an integer value; for example:

```
Number of nodes:<tab>293<newline>
```

Here, <tab> and <newline> indicate the occurrence of a single tab character and a single newline character.

Given the specification, which of the following code fragments will successfully read the integer value into the `int` variable `NetSize`? Assume that `In` is an input file stream variable that has been opened on an input file, and that the data in the stream conforms to the specification.

- |   |   |                    |
|---|---|--------------------|
| 1) <code>In.ignore(25, '\t');</code><br><code>In &gt;&gt; NetSize;</code> | 3) <code>In.ignore(25, ':');</code><br><code>In &gt;&gt; NetSize;</code>  |                    |
| 2) <code>In.ignore(30, '\t');</code><br><code>In &gt;&gt; NetSize;</code> | 4) <code>In.ignore(50, '\t');</code><br><code>In &gt;&gt; NetSize;</code> |                    |
| 5) All of the above   | 7) 2 and 4 only   | 9) 1, 2 and 4 only |
| 6) 1 and 2 only   | 8) 1, 2 and 3 only  | 10) None of these  |

There is a problem with the statement. There's no limit on the length of the label text, so none of the calls to `ignore()` can be guaranteed to work properly. However, if you assumed that the label would be no longer than the sample, then answers 1, 2 and 4 all would work. Answer 3 won't work, in general, because there's nothing in the input specification that says there must be a colon after the label.

Because of the misstatement, we wound up throwing this one out.

For questions 19 and 20, assume that the input file variable `Data.txt` is:

```
1234567890
1234567890
1234567890
1234567890
```

19. What output would the following code fragment produce?

```
ifstream In;
In.open("Data.txt");
char Value;
In.ignore(5, '\n');           // ignores up to the '6'
In >> Value;                 // reads the next character, '6'
cout << "Value: " << Value
     << endl;
```

- |             |             |              |
|-------------|-------------|--------------|
| 1) Value: 1 | 5) Value: 5 | 9) Value: 9  |
| 2) Value: 2 | 6) Value: 6 | 10) Value: 0 |
| 3) Value: 3 | 7) Value: 7 |              |
| 4) Value: 4 | 8) Value: 8 |              |

20. What output would the following code fragment produce?

```
ifstream In;
In.open("Data.txt");
char Value;
In.ignore(20, '\n');           // ignores the first line
In.get(Value);                // reads the first char in the 2nd line
cout << "Value: " << Value
     << endl;
```

1) Value: 1

2) Value: 2

3) Value: 3

4) Value: 4

5) Value: 5

6) Value: 6

7) Value: 7

8) Value: 8

9) Value: 9

10) Value: 0