



READ THIS NOW!

- Choose the single best answer for each question — some answers may be partially correct. If you mark more than one answer to a question, you will receive no credit for any of them.
- Unless a question involves determining whether given C++ code is syntactically correct, it should be; if you think there is a syntax error where there should not be, ask. Unless a question specifically deals with compiler #include directives, you should assume the necessary header files have been included.
- Be careful to distinguish integer values from floating point values (containing a decimal point). In questions or answers that require a distinction between integer and real values, integers will be represented without a decimal point, whereas real values will have a decimal point, [1044 (integer), 1044.0 (floating point)].
- If you are confused about the meaning of one of the questions, email me or come see me. Don't post that to the Forum.
- **This is an open-book, open-notes examination. You may not use a compiler to check your solutions. You may not use other references, including the Web.**
- **You may not collaborate on the test with other students, tutors, etc.**
- **You may not discuss (in any form: written, verbal or electronic) the content of this examination with any student who has not taken it.**
- There are 25 equal-valued multiple-choice questions.
- When you have finished working out your answers, submit them via the Curator as Test2.
- Submitting your answers constitutes an implicit pledge that you have accepted and adhered to the restrictions stated above.

Name (Last, First) _____
printed

Pledge: On my honor, I have neither given nor received unauthorized aid on this examination.

Signature

For the next two questions, consider executing the following code fragment:

```
int j = 4;           // Line 1
while ( j <= 16 ) { // Line 2
    j = 2 * j;      // Line 3
    cout << j << endl; // Line 4
}
```

1. How many times will the body of the loop, lines 3 and 4, be executed?

- | | | | |
|------|------|-------|------------------|
| 1) 0 | 3) 2 | 5) 4 | 7) 13 |
| 2) 1 | 4) 3 | 6) 12 | 8) None of these |

2. What is the last value printed?

- | | | | |
|-----------------------|-------|-------|------------------|
| 1) Nothing is printed | 3) 8 | 5) 32 | 7) 128 |
| 2) 4 | 4) 16 | 6) 64 | 8) None of these |

3. A function, G, has two formal parameters, P1 of type `int` and P2 of type `char`. The data flow (communication) for variable P1 is one-way, into the function. The data flow for variable P2 is one-way, out of the function. Which of the following is the most appropriate prototype for G?

- | | |
|--|-------------------|
| 1) <code>void G(int P1, char P2);</code> | 6) 1 and 2 only |
| 2) <code>void G(int P1, char& P2);</code> | 7) 3 and 4 only |
| 3) <code>void G(int& P1, char P2);</code> | 8) 1 and 3 only |
| 4) <code>void G(int& P1, char& P2);</code> | 9) 2 and 4 only |
| 5) All of them are equally appropriate. | 10) None of these |

4. In C++, which of the following are true about array variables?

- | | |
|---|------------------|
| 1) All the elements of an array must be of the same type. | 5) 1 and 2 only |
| 2) The elements are specified by their location. | 6) 1 and 3 only |
| 3) Aggregate assignment is not supported. | 7) 2 and 3 only |
| 4) All of them | 8) None of these |

5. Consider the following function:

```
int H(int Start, int Stop) {
    int Sum = 0;
    for (int i = Start ; i <= Stop; i++) {
        Sum = Sum + 1;
    }
    return Sum;
}
```

What value is returned from the call `H(3, 5)`?

- | | | |
|-----------------------|------------------------------|----------------------------------|
| 1) 0 | 4) <code>Start + Stop</code> | 7) <code>Stop - Start + 1</code> |
| 2) <code>Start</code> | 5) <code>Stop - Start</code> | 8) None of these |
| 3) <code>Stop</code> | 6) <code>Start * Stop</code> | |

For the next three questions, consider execution of the following program:

```

void Ring(int& Frodo, int Sam);
int Sauron = 3;

int main() {
    int Merry = 7, Pippin = 5;
    Ring(Merry, Pippin);
    cout << "Merry = " << Merry << endl;
    cout << "Pippin = " << Pippin << endl;
    cout << "Sauron = " << Sauron << endl;
    return 0;
}

```

```

void Ring(int& Frodo, int Sam) {
    Sauron = Frodo;
    Frodo = Frodo - 2 * Sam;
    Sam = Sauron;
}

```

6. What value is printed for the variable Merry?

- | | | | |
|-------|------|-------|------------------|
| 1) -5 | 3) 3 | 5) 7 | 7) 21 |
| 2) -3 | 4) 5 | 6) 12 | 8) None of these |

7. What value is printed for the variable Pippin?

- | | | | |
|-------|------|-------|------------------|
| 1) -5 | 3) 3 | 5) 7 | 7) 21 |
| 2) -3 | 4) 5 | 6) 12 | 8) None of these |

8. What value is printed for the variable Sauron?

- | | | | |
|-------|------|-------|------------------|
| 1) -5 | 3) 3 | 5) 7 | 7) 21 |
| 2) -3 | 4) 5 | 6) 12 | 8) None of these |

For the next two questions, consider the execution of the following program:

```

void doIt(string S);

int main() {
    string A = "original";
    doIt(A);
    cout << A << endl;
    return 0;
}

```

```

void doIt(string S) {
    S = "changed";
}

```

9. What value is printed by the cout statement?

- | | |
|-------------|------------------------|
| 1) original | 3) Nothing is printed. |
| 2) changed | 4) None of these |

10. What is the scope of the identifier S?

- | | |
|--------------------|--------------------|
| 1) global | 3) local to doIt() |
| 2) local to main() | 4) None of these |

For the next two questions, assume the declarations below:

```
MAXSIZE = 20;  
char Buffer[3 * MAXSIZE];
```

11. How should the blank preceding MAXSIZE be filled?

- | | | |
|---------------|----------------|------------------|
| 1) char | 4) const int | 7) 2 or 4 only |
| 2) int | 5) All of them | 8) None of these |
| 3) const char | 6) 1 or 3 only | |

12. What is the range of valid index values for the array Buffer[]?

- | | | |
|-----------------|-----------------|------------------|
| 1) 1 through 60 | 4) 0 through 60 | 7) None of these |
| 2) 1 through 61 | 5) 0 through 61 | |
| 3) 1 through 62 | 6) 0 through 62 | |

13. Suppose the first few lines of a function are as follows:

```
string Punctuate(string FName, string LName) {  
    string FullName = LName + COMMA + FName;  
    . . .  
}
```

Assuming the program compiles without any errors, the identifier COMMA must be:

- | | |
|--|--------------------------|
| 1) local to the function Punctuate | 5) Could be any of those |
| 2) a formal parameter in the function Punctuate | 6) 1 or 2 |
| 3) an actual parameter to the function Punctuate | 7) 3 or 4 |
| 4) local to the function that calls Punctuate | 8) None of these |

14. What is the purpose of a *pre-condition* in the header comment for a function?

- 1) To state what conditions are guaranteed to be true after the function is called.
- 2) To state what conditions are guaranteed to be true before the function is called.
- 3) To state what conditions must be true after the function is called.
- 4) To state what conditions must be true before the function is called.
- 5) None of these.

15. Which of the following statements about the lifetime of a variable are true?

- 1) If a variable is local to a function, its lifetime begins when the function is called and ends when the function returns.
- 2) If a variable is global, its lifetime begins when the program begins execution and ends when the program ends.
- 3) For all variables, their lifetime begins when the program begins execution and ends when the program ends.
- 4) All of them are false.
- 5) Only 1 and 2 are true
- 6) All of them are true.
- 7) None of these.

For the next three questions, assume the following declarations:

```
const int SZ = 1000;
string Names[SZ];
```

16. The following loop is supposed to initialize all the cells of the array `Names[]` to hold "Blank".

```
for ( _____; _____; _____ ) {
    Names[Pos] = "Blank";
}
```

How should the three blanks in the loop header be filled?

- | | |
|--|------------------|
| 1) <code>int Pos = 0, Pos <= SZ, Pos++</code> | 5) 1 or 3 only |
| 2) <code>int Pos = 0, Pos < SZ, Pos++</code> | 6) 1 or 4 only |
| 3) <code>int Pos = SZ, Pos > 0, Pos--</code> | 7) 2 or 3 only |
| 4) <code>int Pos = SZ-1, Pos >= 0, Pos--</code> | 8) 2 or 4 only |
| | 9) None of these |
17. The following loop is supposed to read data from an input stream, count the entries, and store it into the array `Names[]`. Assuming the input file is formatted to conform to the logic of the code, does the code contain a logic error?

```
int Read = 0;
In >> Names[Read];
while ( In && Read < SZ ) {
    Read++;
    In >> Names[Read];
}
```

- | | |
|---|-----------------|
| 1) No, the code contains no logic errors. | 5) 2, 3 and 4 |
| 2) Yes, if the input file contains more than 1000 entries, the code will try to store data past the end of the array. | 6) 2 and 3 only |
| 3) Yes, the code will not stop reading if the end of the file is reached. | 7) 2 and 4 only |
| 4) Yes, if the input file contains fewer than 1000 entries, the code will change the first unused cell of the array. | 8) 3 and 4 only |
18. The following loop is supposed to print all data stored in the array `Names[]` by the code in the previous question (with any logic errors that might have occurred there corrected). What value should be used in the blank in the loop header?

```
for (NOT SHOWN; Pos < _____; NOT SHOWN) {
    cout << Names[Pos] << endl;
}
```

- | | | |
|------------------------|--------------------------|------------------|
| 1) <code>SZ - 1</code> | 4) <code>Read - 1</code> | 7) None of these |
| 2) <code>SZ</code> | 5) <code>Read</code> | |
| 3) <code>SZ + 1</code> | 6) <code>Read + 1</code> | |

For the next three questions, consider the incomplete function definition given below:

```
// Mapper traverses an array of integers from front to back, and compares each
// element to its successor, if it has one, replacing the element with half its
// value, if the difference between it and its successor is even. For example,
// given the array:
//
//      9   17   23   21   4   9   12   18
//
// Mapper would transform it into the array:
//
//      4   8   11   21   4   9   6   18
//
// Parameters:
//      List[]  array to be reversed
//      Usage  number of values stored in List[]

void Mapper(int List[], int Usage) {           // Line 1
    int Pos = 0;                               //      2
    while ( _____ ) {                     //      3
        if ( _____ ) {                   //      4
            _____;                       //      5
        }
        Pos++;                                 //      6
    }
}
```

19. How should the blank in Line 3 be filled to properly terminate the loop?

- | | | |
|--------------------|------------------------------|------------------|
| 1) Pos < Usage - 1 | 3) Pos <= Usage | 5) true |
| 2) Pos < Usage | 4) List[Pos] < List[Pos + 1] | 6) None of these |

20. How should the blank in Line 4 be filled to decide whether the list entry needs to be changed?

- 1) Pos % 2 == 0
- 2) List[Pos] % 2 == 0
- 3) (List[Pos + 1] - List[Pos]) % 2 == 0
- 4) List[Pos] % 2 == 1
- 5) 2*List[Pos] == List[Pos + 1]
- 6) None of these

21. How should the blank in line 5 be filled to carry out the specified transformation of the list elements?

- | | |
|----------------------------------|------------------------------|
| 1) List[Pos] = Pos / 2 | 5) List[Pos] = List[Pos + 1] |
| 2) List[Pos] = List[Pos / 2] | 6) 2 or 3 only |
| 3) List[Pos] = List[Pos] / 2 | 7) 3 or 4 only |
| 4) List[Pos] = List[Pos + 1] / 2 | 8) None of these |

Consider a data file that contains lines of score information, so that each line contains one or more integer values, followed by a zero that marks the end of the data on that line. The first line of the file will specify the number of data lines that follow.

For the next four questions, consider the function given below, which is intended to read a data file as described above and determine the largest sum that can be obtained from any data line.

<pre> int hiScore(ifstream& Data) { int maxSoFar = INT_MIN; int numLines, lineSum, Value; Data >> numLines; for (int Line = 1; Line <= numLines; Line++) { while (Value != 0) { lineSum = lineSum + Value; } } return maxSoFar; } </pre>	<pre> // 1 // 2 // 3 // 4 // 5 // 6 // 7 // 8 // 9 // 10 // 11 // 12 // 13 // 14 // 15 // 16 </pre>	<p>Data file:</p> <pre> 10 23 14 9 0 31 0 12 2 7 20 19 0 22 14 2 0 8 3 21 9 4 3 2 0 25 3 0 47 9 0 5 5 5 5 5 5 0 1 2 34 8 0 20 5 13 0 </pre>
---	---	--

22. How should the variable `lineSum` be initialized?

- | | | |
|------------------------|------------------------|---------------------|
| 1) set to 0 in line 5 | 4) set to 1 in line 5 | 7) It shouldn't be. |
| 2) set to 0 in line 7 | 5) set to 1 in line 7 | 8) None of these |
| 3) set to 0 in line 12 | 6) set to 1 in line 12 | |

23. Where should the statement `Data >> Value` appear?

- | | | |
|------------|-----------------------|-----------------------|
| 1) line 5 | 4) line 12 | 7) line 8 and line 10 |
| 2) line 8 | 5) line 5 and line 10 | 8) line 8 and line 12 |
| 3) line 10 | 6) line 5 and line 12 | 9) None of these |

24. Where should the following if-statement appear: `if (lineSum > maxSoFar)`
`maxSoFar = lineSum;`

- | | | |
|------------|------------------------------|------------------|
| 1) line 12 | 3) either line 12 or line 14 | 5) None of these |
| 2) line 14 | 4) It should NOT appear. | |

25. What would happen if the input file contained more lines of data than were promised by its first line?

- 1) The function would process the extra lines of data.
- 2) The function would process as many lines as were promised, and return an answer based only on those lines.
- 3) The function would fail to terminate.
- 4) None of these