

**Instructions:** This homework assignment focuses primarily on C++ scope rules and functions. The answers to the following questions can be determined from Chapters 3 through 8 of the lecture notes and Chapters 2 through 8 of the text. Assume any `#include` directives, variable declarations, etc, which are needed to make the given code syntactically correct.

After you have analyzed the questions and decided what answers you believe are correct, you may find it useful to write some short programs to test your logic.

Opscan forms will be passed out in class. Write your name and code your ID number on the opscan form. Turn in your completed opscan at class on Monday Oct 29 or Tuesday Oct 30. Opscans will not be accepted at any other place or time.

---

1. Which of the following statements concerning the scope of C++ identifiers is correct?
  - 1) The scope of an identifier begins at the point of its declaration and terminates at the end of the file containing the code.
  - 2) The scope of an identifier begins at the beginning of the block in which it is declared and terminates at the end of the block, if any, that contains its declaration.
  - 3) The scope of an identifier begins at the point of its declaration and terminates at the end of the block, if any, which contains its declaration.
  - 4) None of these are correct

---
2. If an identifier `Alpha` is accessible only within a function `F`, then `Alpha` is either:
  - 1) declared with global scope or a formal parameter of `F`.
  - 2) declared locally within `F` or a formal parameter of `F`.
  - 3) declared with global scope or an actual parameter passed to `F`.
  - 4) declared locally within `F` or an actual parameter to `F`.
  - 5) None of these

---
3. If the identifier `Beta` is declared as a formal parameter of a function `F`, then the scope of `Beta`:
  - 1) is the body of the implementation of `F`.
  - 2) extends from the declaration of the function `F` to the end of the file containing the code.
  - 3) is empty; that is, `Beta` has no scope.
  - 4) None of these

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

```
double Calc( double Beta ) {  
    Alpha = 3.8 * Beta;  
    . . .
```

If the code compiles, then the variable `Alpha` must be:
  - 1) a local variable declared later in the body of `Calc()`
  - 2) a global variable
  - 3) a parameter passed to `Calc()`
  - 4) 1 or 2 only
  - 5) 1 or 3 only
  - 6) 2 or 3 only
  - 7) None of these

---

For questions 5 through 9, consider the following program:

```
const int LIMIT = 50;           // Line 1
int AddEm(int x, int y);       // Line 2
int main() {                   // Line 3
    int x = 42,                // Line 4
        y = 35;               // Line 5
    int Sum;                   // Line 6

    Sum = AddEm(x, y);        // Line 7

    return 0;                 // Line 8
}                               // Line 9

int AddEm(int x, int y) {      // Line 10

    int Total;                // Line 11
    Total = x + y;            // Line 12
    if (Total > LIMIT)        // Line 13
        Total = 0;           // Line 14
    return (Total);           // Line 15
}                               // Line 16
```

5. What is the scope of the identifier Sum that is declared in Line 6?
- 1) Line 1 to Line 16
  - 2) Line 6 to Line 16
  - 3) Line 6
  - 4) Line 6 to Line 7
  - 5) Line 6 to Line 9
  - 6) None of these
6. What is the scope of the identifier x that is declared in Line 4?
- 1) Line 1 to Line 16
  - 2) Line 4 to Line 16
  - 3) Line 4
  - 4) Line 4 to Line 7
  - 5) Line 4 to Line 9
  - 6) None of these
7. What is the scope of the identifier x that is declared in Line 10?
- 1) Line 1 to Line 16
  - 2) Line 4 to Line 16
  - 3) Line 10
  - 4) Line 10 to Line 12
  - 5) Line 10 to Line 16
  - 6) None of these
8. What is the scope of the identifier LIMIT that is declared in Line 1?
- 1) Line 1 to Line 16
  - 2) Line 1 to Line 3
  - 3) Line 1
  - 4) Line 10 to Line 13
  - 5) Line 10 to Line 16
  - 6) None of these
9. Which of the following are true?
- 1) LIMIT is local to main()
  - 2) Total is local to AddEm()
  - 3) Sum is local to main()
  - 4) LIMIT is global
  - 5) x is global
  - 6) All of them are true
  - 7) All but 1 are true
  - 8) 2 and 3 only
  - 9) 2, 3 and 4 only
  - 10) None of these

- 
10. Formal parameters are listed in the function \_\_\_\_\_ and actual parameters are listed in the function \_\_\_\_\_.
- 1) call, implementation
  - 2) implementation, call
  - 3) header, body
  - 4) body, header
  - 5) None of these
- 
11. When parameters are passed between the calling code and the called function, formal and actual parameters are matched according to:
- 1) their data types
  - 2) their names
  - 3) their relative positions in the formal and actual parameter lists
  - 4) whether they are inputs to or outputs from the function
  - 5) None of these
- 
12. A parameter of a simple type, such as `int` or `double`, should be passed by value if that parameter's data flow is:
- 1) one-way, into the function.
  - 2) one-way, out of the function.
  - 3) two-way, into and out of the function.
  - 4) None of these
- 
13. Which of the following statements are true when a parameter is passed by value?
- 1) The actual parameter is never modified by execution of the called function.
  - 2) The formal parameter is never modified by execution of the called function.
  - 3) The actual parameter must be a variable.
  - 4) All of these are false.
  - 5) 2 and 3 only
  - 6) None of these
- 
14. Which of the following statements are true when a parameter is passed by reference?
- 1) The actual parameter can be modified by execution of the called function.
  - 2) The formal parameter can be modified by execution of the called function.
  - 3) The actual parameter cannot be a variable.
  - 4) All of these are false.
  - 5) 1 and 2 only
  - 6) None of these
- 
15. Which of the following statements are true when a parameter is passed by constant reference?
- 1) The actual parameter can be modified by execution of the called function.
  - 2) The formal parameter can be modified by execution of the called function.
  - 3) The actual parameter cannot be a variable.
  - 4) All of these are false.
  - 5) 1 and 2 only
  - 6) None of these
-

16. If an ampersand ( ' & ' ) is not attached to the data type of a formal parameter, then the corresponding actual parameter can be:

- |                            |                 |                  |
|----------------------------|-----------------|------------------|
| 1) a constant              | 4) All of these | 7) None of these |
| 2) a variable name         | 5) 1 and 2 only |                  |
| 3) an arbitrary expression | 6) 2 and 3 only |                  |
- 

17. A function `SomeFunc` has two formal parameters, `alpha` and `beta`, of type `int`. The data flow for `alpha` is one-way, into the function. The data flow for `beta` is two-way, into and out of the function. What is the most appropriate function prototype for `SomeFunc`?

- |   |                  |
|---|------------------|
| 1) <code>void SomeFunc( int alpha, int beta );</code>           | 5) 1 and 2 only  |
| 2) <code>void SomeFunc( int&amp; alpha, int beta );</code>      | 6) 3 and 4 only  |
| 3) <code>void SomeFunc( int alpha, int&amp; beta );</code>      | 7) None of these |
| 4) <code>void SomeFunc( int&amp; alpha, int&amp; beta );</code> |                  |
- 

18. For the function definition

```
void Func( int& Gamma ) {  
    Gamma = 245;  
}
```

which of the following comments best describes the direction of data flow for `Gamma`?

- |                                 |  |
|---------------------------------|--|
| 1) one-way, into the function   | 3) two-way, into and out of the function |
| 2) one-way, out of the function | 4) None of these                         |
- 

19. For the function definition

```
void Func( int Gamma ) {  
    cout << 3 * Gamma;  
}
```

which of the following comments best describes the direction of data flow for `Gamma`?

- |                                 |  |
|---------------------------------|--|
| 1) one-way, into the function   | 3) two-way, into and out of the function |
| 2) one-way, out of the function | 4) None of these                         |
- 

20. For the function definition

```
void Func( int& Gamma ) {  
    Gamma = 3 * Gamma;  
}
```

which of the following comments describes the direction of data flow for `Gamma`?

- |                                 |  |
|---------------------------------|--|
| 1) one-way, into the function   | 3) two-way, into and out of the function |
| 2) one-way, out of the function | 4) None of these                         |
-

21. Consider the function definition

```
void Demo( int intVal, double& doubleVal ) {
    intVal    = intVal * 2;
    doubleVal = double(intVal) + 3.5;
}
```

What values does the following code fragment print?

```
int    myInt    = 20;
double myDble = 4.8;
Demo(myInt, myDble);
cout << "myInt = " << myInt
    << " and myDble = " << myDble << endl;
```

- 1) myInt = 20 and myDble = 43.5
  - 2) myInt = 40 and myDble = 4.8
  - 3) myInt = 20 and myDble = 4.8
  - 4) myInt = 40 and myDble = 43.5
  - 5) None of these
- 

22. Consider the function definition

```
void Demo( int& intVal, double doubleVal ) {
    intVal    = intVal * 2;
    doubleVal = double(intVal) + 3.5;
}
```

What values does the following code fragment print?

```
int    myInt    = 20;
float  myDble = 4.8;
Demo(myInt, myDble);
cout << "myInt = " << myInt
    << " and myDble = " << myDble << endl;
```

- 1) myInt = 20 and myDble = 43.5
  - 2) myInt = 40 and myDble = 4.8
  - 3) myInt = 20 and myDble = 4.8
  - 4) myInt = 40 and myDble = 43.5
  - 5) None of these
- 

23. In the following function, the declaration of Beta includes an initialization.

```
void SomeFunc( int Alpha )
{
    int Beta = 25;
    ...
}
```

Which of the following statements about the variable Beta declared above is false?

- 1) It is initialized once only, the first time the function is called.
  - 2) It is initialized each time the function is called.
  - 3) It cannot be reassigned a different value within the function.
  - 4) 1 and 3 only
  - 5) 2 and 3 only
  - 6) None of these are false
-

For questions 24 and 25, consider the short program:

```
#include <iostream>           // Line 1
using namespace std;         // Line 2

int main() {                  // Line 3
    int alpha = 3;           // Line 4
    int beta  = 20;          // Line 5

    if (beta > 10)           // Line 6
    {
        int alpha = 5;       // Line 7

        beta = beta + alpha; // Line 8
        cout << alpha << ' ' // Line 9
             << beta  << ' '; // Line 10
    }                          // Line 11

    cout << alpha << ' ' << beta; // Line 12
    return 0;                  // Line 13
}                               // Line 14
```

24. What is the scope of the identifier alpha declared in Line 4?

- |                                      |                       |
|--------------------------------------|-----------------------|
| 1) Line 4 through Line 14            | 3) Lines 4 and 5 only |
| 2) Lines 4, 5, 6, 12, 13 and 14 only | 4) None of these      |

25. What is the output of the given program?

- |              |              |                  |
|--------------|--------------|------------------|
| 1) 3 20      | 3) 5 25 5 25 | 5) 5 25 3 20     |
| 2) 3 25 3 25 | 4) 5 25 3 25 | 6) None of these |

26. This question demonstrates the hazard of choosing inappropriate parameter-passing mechanisms. Given the function definition

```
int Power(int& Base, int& Exponent ) {
    int Product = 1;
    while (Exponent >= 1) {
        Product = Product * Base;
        Exponent--;
    }
    return Product;
}
```

what is the output of the following code?

```
int N = 2;
int Pow = 3;
int Result = Power(N, Pow);
cout << N << " to the power " << Pow << " is " << Result;
```

- |                          |                          |
|--------------------------|--------------------------|
| 1) 2 to the power 3 is 8 | 4) 2 to the power 3 is 1 |
| 2) 2 to the power 0 is 8 | 5) None of these         |
| 3) 0 to the power 0 is 0 |                          |

27. Which of the following would be the most appropriate prototype for a function that computes the tax, in cents, on a purchase? Assume that the tax rate will be declared within the function.

- |                          |                         |
|--------------------------|-------------------------|
| 1) void Tax(int& Price); | 3) void Tax(int Price); |
| 2) int Tax(int& Price);  | 4) int Tax(int Price);  |

28. Which of the following things must be specified in a function prototype?

- |                                   |                    |
|-----------------------------------|--------------------|
| 1) name of the function           | 6) 1 and 2 only    |
| 2) types of the formal parameters | 7) 2 and 3 only    |
| 3) return type of the function    | 8) 1 and 3 only    |
| 4) names of the formal parameters | 9) 1, 2 and 3 only |
| 5) All of the above               | 10) None of these  |

For questions 29 and 30, consider this program, which illustrates one of the hazards of polluting global scope with variable declarations:

```
#include <fstream>
#include <iostream>
#include <string>
using namespace std;
const char SPACE = ' ';
int lCount;

void countSpaces(string Line);

int main( ) {
    ifstream In("Data.txt");
    string Line;
    lCount = 0;
    getline(In, Line);
    while (In) {
        lCount++;
        countSpaces(Line);
        getline(In, Line);
    }

    cout << "Lines: "
         << lCount << endl;
    In.close();
    return 0;
}

void countSpaces(string Line) {
    int sCount = 0;
    int Idx = 0;
    while ( Idx < Line.length() ) {
        if ( Line[Idx] == SPACE )
            lCount++;
        Idx++;
    }
    cout << "Spaces on line: "
         << sCount << endl;
}
```

29. If the program is run on the input file shown below, what value will be reported for the number of lines? (The first character in the file is the 'S' and the last is the '.'.)

```
Software test teams appreciate the use of global variables.
```

- |      |      |                  |
|------|------|------------------|
| 1) 0 | 4) 3 | 8) 9             |
| 2) 1 | 5) 5 | 9) None of these |
| 3) 2 | 6) 7 |                  |

30. If the declaration of lCount were local to main( ), how would the error in this program have been detected?

- 1) By the discovery of an incorrect result when the output was examined.
- 2) By the occurrence of a run-time error.
- 3) By the occurrence of a compile-time error message.
- 4) None of these.