

Instructions: This homework assignment focuses primarily on C++ functions. The answers to the following questions can be determined from Chapters 3 through 9 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. Submit your answers to the Curator collection point HW6.

1. Given the function prototype and declarations:

```
float Fix(int N, float& X);
int someInt = 10;
float someFloat = 4.3;
```

which of the following function calls would be syntactically correct?

- | | |
|---|---------------------|
| 1) <code>Fix(someInt, 6.85);</code> | 5) all of the above |
| 2) <code>someFloat = Fix(24, someFloat);</code> | 6) 1 and 3 only |
| 3) <code>someFloat = 0.3 * Fix(someInt, 6.85);</code> | 7) 2 and 4 only |
| 4) <code>Fix(someInt + 5, someFloat);</code> | 8) None of these |
-

2. For the function definition

```
void Func( int Gamma ) {
    int Delta = Gamma - 1;
}
```

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 |
-

3. 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 |
-

4. 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 |
-

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

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

7. 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. | 4) 1 and 3 only |
| 2) It is initialized each time the function is called. | 5) 2 and 3 only |
| 3) It cannot be reassigned a different value within the function. | 6) None of these are false |
-

For questions 8 and 9, 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
    }

    cout << alpha << ' ' << beta; // Line 11
    return 0;                    // Line 12
}
```

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

- | | |
|-----------------------------------|-----------------------|
| 1) Line 4 through Line 12 | 4) Lines 4 and 5 only |
| 2) Lines 4, 5, 11 and 12 only | 5) Line 4 only |
| 3) Lines 4, 5, 6, 11, and 12 only | 6) None of these |

9. 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 |

10. 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 | |