

Instructions: This homework assignment focuses primarily on C++ functions and arrays. The answers to the following questions can be determined from notes and text chapters discussion functions and arrays. Assume any `#include` directives, variable declarations, etc, which are needed to make the given code syntactically correct.

1) When parameters are passed between the calling code and the called function, formal and actual parameters are matched by:

- 1) their data types
- 2) their relative positions in the formal and actual parameter lists
- 3) their names
- 4) whether they are inputs to or outputs from the function
- 5) none of these

2) A parameter as a simple type for example 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) 1 and 2 above
- 5) 2 and 3 above
- 6) none of these

3) Given the function prototype and declarations:

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

which of the following function calls would be syntactically correct?

- 1) `Fix(someInt, 6.85);`
- 2) `someFloat = Fix(24, 6.85);`
- 3) `someFloat = 0.3 * Fix(6, someFloat);`
- 4) `Fix(someInt + 5, someFloat);`
- 5) all of the above
- 6) 1 and 3 above
- 7) 2 and 4 above
- 8) none of the above

4) A function `SomeFunc` has two formal parameters, `alpha` and `beta`, of type `int`. The data flow for `alpha` is two-way, into and out of 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) `void SomeFunc(int alpha, int beta);`
- 2) `void SomeFunc(int& alpha, int beta);`
- 3) `void SomeFunc(int alpha, int& beta);`
- 4) `void SomeFunc(int& alpha, int& beta);`
- 5) 1 and 2 above
- 6) 3 and 4 above
- 7) none of these

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

- 1) a constant
- 2) a variable name
- 3) an arbitrary expression
- 4) 1 and 2 above
- 5) 1, 2, and 3 above
- 6) none of the above

6) Given the function definition

```
void Twist( int& a, int& b ) {  
    int r;  
  
    r = a + 2;  
    a = a * 3;  
    b = r + a;  
}
```

what is the output of the following code fragment that invokes Twist?

```
int r = 1;  
int s = 2;  
int t = 3;  
Twist(t, s);  
cout << r << ' ' << s << ' ' << t << endl;
```

- | | |
|----------------------|-----------|
| 1) 1 2 3 | 2) 1 2 9 |
| 3) 1 14 3 | 4) 1 10 3 |
| 5) 5 14 3 | 6) 1 14 9 |
| 7) none of the above | |

7) Which of the following statements about value parameters is false?

- 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) The actual parameter can be a literal constant value.
- 5) 2 and 3 above
- 6) none of these

8) Which of the following statements about reference parameters is false?

- 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) The actual parameter cannot be a literal constant value.
- 5) 1 and 2 above
- 6) none of these

9) Which of the following statements about constant reference parameters is false?

- 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) The actual parameter cannot be a literal constant value.
- 5) 1 and 2 above
- 6) none of these

10) Consider the function definition

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

What values are printed by the following code fragment?

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

- 1) myInt = 20 and myFloat = 43.5
- 2) myInt = 40 and myFloat = 4.8
- 3) myInt = 20 and myFloat = 4.8
- 4) myInt = 40 and myFloat = 43.5
- 5) none of the above

11) 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
- 2) one-way, out of the function
- 3) two-way, into and out of the function
- 4) none of the above

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

```
int Power(int& base, int& exp ) {  
    int product = 1;  
  
    while (exp >= 1) {  
        product = product * base;  
        exp--;  
    }  
    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
- 2) 2 to the power 0 is 8
- 3) 0 to the power 0 is 0
- 4) 2 to the power 3 is 1
- 5) none of the above

13) 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
- 2) one-way, out of the function
- 3) two-way, into and out of the function
- 4) none of the above

14) 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
- 2) one-way, out of the function
- 3) two-way, into and out of the function
- 4) none of the above

15) If a variable alpha is accessible only within function F, then alpha is either

- 1) a global variable or a formal parameter of F.
- 2) a local variable within F or a formal parameter of F.
- 3) a global variable or an actual parameter to F.
- 4) a local variable within F or an actual parameter to F.
- 5) none of the above

16) Choose the correct statement:

- 1) A function prototype is global if it is placed outside function definitions in the source file.
- 2) A function prototype is local if it is placed in a function definition.
- 3) The scope of a global prototype begins at the point it is placed and extends until the end of the source file.
- 4) The scope of a local prototype begins at the point it is placed and extends until the end of the function in which it appears.
- 5) All of the above are correct
- 6) None of the above are correct

17) Suppose the first few lines of a function are as follows:

```
void Calc( float beta )  
{  
    alpha = 3.8 * beta;
```

Assuming the code compiles, then the variable alpha is

- 1) a local variable
- 2) a global variable
- 3) a parameter
- 4) none of the above

18) What is the output of the following code fragment?

```
int alpha = 3;
int beta = 20;
if (beta > 10) {
    int alpha = 5;

    beta = beta + alpha;
    cout << alpha << ' ' << beta << ' ';
}
cout << alpha << ' ' << beta;
```

- 1) 3 20
- 2) 3 25 3 25
- 3) 5 25 5 25
- 4) 5 25 3 25
- 5) 5 25 3 20
- 6) none of the above

19) Given the declaration: `int Alpha[99];`

the logically valid range of index values for alpha is:

- 1) 0 through 99
- 2) 0 through 98
- 3) 1 through 99
- 4) 1 through 98
- 5) 1 through 100
- 6) none of these

20) You are writing a program to count the frequencies of decimal digits that are read from a data file. Which of the following array declarations is most appropriate, given that digit values will be used to index into the `freqCount[]` array?

- 1) `char freqCount[10];`
- 2) `char freqCount[9];`
- 3) `int freqCount[10];`
- 4) `int freqCount[int];`
- 5) none of these

21) What is the output of the following program fragment?

```
int Alpha[5] = {500, 400, 300, 200, 100};
int i;

for (i = 4; i > 0; i--)
    cout << Alpha[i] << ' ';
```

- 1) 400 300 200 100
- 2) 100 200 300 400 500
- 3) 100 200 300 400
- 4) 4 3 2 1
- 5) It cannot be answered from the information given.
- 6) None of these

22) Given a 5000-element integer array `Beta []`, which of the code fragments below could be used to print out the values of `Beta[1]`, `Beta[3]`, `Beta[5]`, and so forth?

1)

```
for (int i = 0; i < 5000; i = i + 2)
    cout << Beta[i] << endl;
```

2)

```
for (int i = 0; i < 2500; i++)
    cout << Beta[2*i + 1] << endl;
```

3)

```
for (int i = 0; i < 2500; i++)
    cout << Beta[i]*2 - 1 << endl;
```

- 4) all of the above
 5) 1 and 2 only
 6) none of these

23) You are writing a program to keep track of majors for a collection of students. Given the following declarations

```
const int MAXSTUDENTS = 100;
const int NUMMAJORS   = 4;
const string MAJORS[NUMMAJORS] = {"CS", "CpE", "Math", "Other"};
```

which of the following will properly declare two parallel arrays for storing student names and majors?

- | | |
|--|-----------------|
| 1) <pre>string Name[MAXSTUDENTS]; string Major[NUMMAJORS];</pre> | 4) All of them |
| 2) <pre>string Name[MAXSTUDENTS]; string Major[MAXSTUDENTS];</pre> | 5) 1 and 2 only |
| 3) <pre>string Name[NUMMAJORS]; string Major[NUMMAJORS];</pre> | 6) 1 and 3 only |
| | 7) 2 and 3 only |
| | 8) None of them |

24) Given the declarations below, how many components of type `double` does `Depth` have?

```
const int HEIGHT = 50;
const int WIDTH  = 100;
double Depth[HEIGHT][WIDTH];
```

- | | |
|------------|--|
| 1) 100 | 4) 100 * 50 |
| 2) 50 | 5) None, they are of type <code>int</code> . |
| 3) 99 * 49 | 6) None of these |

25) Given the declarations from question 24, the expression `Depth[5][3]` refers to:

- 1) the third element of the fifth row of `Depth`
 2) the fifth element of the third row of `Depth`
 3) the fourth element of the sixth row of `Depth`
 4) the sixth element of the fourth row of `Depth`
 5) The expression is not allowed.
 6) None of these

Questions 26 through 28 refer to the following incomplete program:

```

void FillEm(_____ arr1[], _____ arr2[], int length); // line 1
void Copy(_____ arr1[], _____ arr2[], int length); // line 2

void main() {
    const int dim = 200;
    char alpha[dim];
    char beta[dim];

    FillEm(alpha, beta, dim); // Initialize arrays alpha and beta
    Copy(alpha, beta, dim); // Copy all components of beta into alpha
}

void Copy(_____ arr1[], _____ arr2[], int length) {
    for (int Idx = 0; Idx < length; Idx++) {
        arr1[Idx] = arr2[Idx];
    }
    return;
}

void FillEm(_____ arr1[], _____ arr2[], int length) {
    // some initialization code goes here
}

```

26) What is the most appropriate way to fill the blank preceding the first formal parameter of FillEm() in line 1?

- | | |
|---------------|-------------------------|
| 1) int& | 5) int |
| 2) char& | 6) const int |
| 3) char | 7) all are equally good |
| 4) const char | 8) none are good |

27) What is the most appropriate way to fill the blank preceding the first formal parameter of Copy() in line 2?

- | | |
|---------------|-------------------------|
| 1) int& | 5) int |
| 2) char& | 6) const int |
| 3) char | 7) all are equally good |
| 4) const char | 8) none are good |

28) What is the most appropriate way to fill the blank preceding the second formal parameter of Copy() in line 2?

- 1) int&
- 2) char&
- 3) char
- 4) const char
- 5) int
- 6) const int
- 7) all are equally good
- 8) none are good

29) Given the declarations below, the elements of X could be copied to the corresponding locations in Y by:

```
char X[100], Y[100];
```

- 1) the statement: `Y = X;`
 - 2) the statement: `Y[] = X[];`
 - 3) a user-defined function to which X and Y are passed
 - 4) All of these.
 - 5) 1 and 3 only
 - 6) The elements of Y cannot possibly be copied into X.
 - 7) None of these
- 30) Which of the following code fragments can be used to input values into a 3-element string array named Alpha?

1) `cin >> Alpha[0] >> Alpha[1] >> Alpha[2];`

2) `cin >> Alpha;`

3) `for (i = 0; i < 3; i++)
 cin >> Alpha[i];`

4) `cin >> Alpha[0];
cin >> Alpha[1];
cin >> Alpha[2];`

- 5) All of them
- 6) 1 and 4 only
- 7) 1, 3 and 4 only
- 8) None of them