Assignment 2

The due date for submitting this assignment has passed. Due on 2020-02-12, 23:59 IST.

As per our records you have not submitted this assignment.

1) Consider the following overloaded functions.

```cpp
int fun(int x) { return x; }  \ Function-1
int fun(int x, int y) { return x + y; }  \ Function-2
int fun(int x, int y = 10) { return x + y; }  \ Function-3
```

Choose the appropriate option below for the following function calls:

```cpp
fun(10);
fun(10, 20);
```

- a) `fun(10)` invokes Function-1 and `fun(10, 20)` invokes Function-2.
- b) `fun(10)` invokes Function-3 and `fun(10, 20)` invokes Function-2.
- c) `fun(10)` invokes Function-1 and `fun(10, 20)` is an ambiguous call.
- d) Both function calls are ambiguous

No, the answer is incorrect.

Score: 0

Accepted Answers:

- d) Both function calls are ambiguous
2) Consider the following code segment.

```cpp
#include <iostream>
using namespace std;
#define MULT(x, y) x * y

inline int mult(int x, int y) {
    return x * y;
}

int main() {
    int i = 2, j = 3;
    
    cout << MULT(MULT(i, j - 1), i) << " 
    ";
    cout << MULT(i, MULT(i, j - 1)) << " 
    ";
    cout << mult(mult(i, j - 1), i) << " 
    ";
    cout << mult(i, mult(i, j - 1)) << endl;
    
    return 0;
}
```

What will be the output?

- a) 4, 11, 4, 11
- b) 8, 8, 8, 8
- c) 4, 11, 8, 8
- d) Compilation error on macro expansion

No, the answer is incorrect.
Score: 0
Accepted Answers:
- c) 4, 11, 8, 8

3) Find the correct matching pair of statements (1-5) given below, in association with the manifest constant (X) and constant variable (declared using const keyword) (Y).

1. It is textually replaced by C++ preprocessor.
2. It is type-safe to use.
3. It is visible to compiler and debugger.
4. It has to be evaluated every time it is replaced.
5. It is defined by #define.

- a) X-1, X-4, X-5, Y-2, Y-3
- b) X-1, X-4, Y-2, Y-3, Y-5
- c) X-1, X-5, Y-1, Y-2, Y-3
- d) X-2, X-3, X-4, Y-1, Y-5
4) Consider the following code segment.

```cpp
#include <iostream>
using namespace std;

int fun(int x, int y, int z) {
    return (x + y + z);
}

int main() {
    cout << fun(10);
    return 0;
}
```

What should be the output / error?

- a) 10
- b) 30
- c) error: invalid prototype
- d) error: default argument missing for parameter 2

No, the answer is incorrect.
Score: 0
Accepted Answers:
- d) error: default argument missing for parameter 2

5) Consider the overloaded functions as follows:

```cpp
void f(int i)
    { cout << "f(int i)" << endl; } // LINE-1

void f(double i, double j = 0)
    { cout << "f(double i, double j = 0)" << endl; } // LINE-2

void f(int i = 0, int j = 0)
    { cout << "f(int i = 0, int j = 0)" << endl; } // LINE-3

void f(float i)
    { cout << "f(float i)" << endl; } // LINE-4
```

What will be output for the function call f(10.5)?
a) f(int i)
b) f(double i, double j = 0)
c) f(int i = 0, int j = 0)
d) f(float i)

No, the answer is incorrect.
Score: 0
Accepted Answers:
b) f(double i, double j = 0)

6)
Consider the following code segment.

```
#include <iostream>
using namespace std;

int main() {
    const int i = 10;
    int* p;       // LINE-1
    p = &i;       // LINE-2
    cout << *p;
    return 0;
}
```

LINE-2 generates a compiler error. What is the correct option(s) to declare p at LINE-1 so that compiler errors can be fixed?

- a) const int * p;
- b) int * const p;
- c) int const * p;
- d) int const * const p;

No, the answer is incorrect.
Score: 0
Accepted Answers:
a) const int * p;
c) int const * p;
7) Consider the code segment below.

```cpp
#include <iostream>
using namespace std;

int f(int &x, int c) {
    c--; x--; // LINE-1
    if (c == 0) return 1; // LINE-2
    return f(x, c) * x; // LINE-3
}

int main() {
    int a = 5, b = 5;
    int c = f(a, b);
    cout << a << " " << b << " " << c << endl;
    return 0;
}
```

- a) 5 5 5
- b) 0 0 0
- c) 0 5 0
- d) 5 0 5

No, the answer is incorrect.
Score: 0
Accepted Answers:
- c) 0 5 0
8) Consider the code segment below.

```c++
#include <iostream>
using namespace std;

#define ISEQUAL(X, Y) X == Y

int main()
{
    int x, y, z;
    x = y = z = 4;

    z = ISEQUAL(x, y); // LINE-1
    cout << x << " " << y << " " << z << endl;

    ISEQUAL(z, (x = y)); // LINE-2
    cout << x << " " << y << " " << z << endl;

    return 0;
}
```

What will be the output?

- a) 4 4 4
  4 4 0
- b) 4 4 1
  4 4 0
- c) 4 1 1
  4 1 1
- d) 4 4 1
  4 4 1

No, the answer is incorrect.
Score: 0
Accepted Answers:
- d) 4 4 1
  4 4 1

9) Consider the code segment below.

```
static int var; // LINE-1
void f(void) {
    var = 0;
    while (var != 100); // LINE-2
}
```

LINE-2 represents an infinite loop. What can be the appropriate option to declare var at LINE-1 such that var may be changed by hardware at any time?

- a) static mutable int i;
- b) static volatile int i;
- c) int i;
d) mutable int i;

No, the answer is incorrect.
Score: 0
Accepted Answers:

b) static volatile int i;