

The following exam is pledged. All answers are to be done on the answer sheet that is provided. The test is closed book and closed note.

1. The parameters in a function invocation are called the _____ parameters. The parameters are represented in the invoked function by the _____ parameters.
2. The values of the formal parameters and other objects defined in a function are kept in its _____.
3. A _____ is a description of a function's interface.
4. A _____ statement supplies a value from the invoked function to the invoking function.
5. A _____ object is an object defined within a statement block.
6. A _____ object is an object defined outside of any function interface or function body.
7. C++ supports two forms of parameter passing: _____ and _____.
8. A _____ modifier applied to a parameter declaration indicates that the function may not change the object.
9. _____ parameters can be specified as trailing parameters only.
10. _____ is when two or more functions have the same name.
11. Casting expressions provide a facility to explicitly convert one _____ to another.
12. Member functions that return the value of an attribute of an object are called _____.
13. Member functions that set or change the value of an attribute of an object are called _____.
14. A _____ member function is one that cannot change its objects' attributes.
15. A _____ constructor is a constructor that requires no parameters.
16. A _____ constructor initializes a new object to be a duplicate of a previously defined source object. If a class does not define such a constructor, the compiler automatically supplies a version.
17. The client interface to a class object occurs in the _____ section of the class definition.
18. Data members are normally declared in a _____ section. By restricting direct client program access to the data members in an ADT, it is easier to ensure the integrity and consistency of their values.
19. An _____ in the return type for a function or operator indicates that a reference return is being performed.
20. _____ is the reason class objects that are not modified in a function or operator are typically passed as constant reference parameters rather than as value parameters.

21. Prototype a function `f()` that takes a double-precision floating-point value `x` as its parameter and returns a value of that same type.
22. Consider the following program:

```
#include <iostream.h>
// prototype of f belongs here
int main() {
    int x = 1;
    f(x);
    cout << "x is " << x << endl;
    return 0;
}
```

When this program runs, the output is

x is 2

Write the function prototype for **void** function `f()`.

23. What is the output of the following program?

```
#include <iostream.h>
int counter = 10;
void f() {
    ++counter;
}
void g() {
    f();
    f();
}
int main() {
    f();
    cout << "output1: " << counter << endl;
    g();
    cout << "output2: " << counter << endl;
    return 0;
}
```

24. What is the output of the following program?

```
#include <iostream.h>
void f() {
    int i = 10;
    cout << "f: " << i << endl;
    return;
}
int main() {
    int i = 1;
    f();
    cout << "main: " << i << endl;
    return 0;
}
```

25. Consider the following overloaded functions:

```
#include <iostream.h>
void f(int a, double b) {
    cout << "f(int, double) says a is "
        << a << endl;
    return;
}
void f(int a, int b) {
    cout << "f(int, int) says b is "
        << b << endl;
    return;
}
void f(double a, double b) {
    cout << "f(double, double) says b is "
        << b << endl;
    cout << endl;
    return;
}
```

What is the output when the following program is executed using these overloaded functions?

```
int main() {
    int i = 1;
    int j = 2;
    double x = 3.5;
    double y = 10.2;
    f(y, x);
    f(i, y);
    f(2, 4);
    f(2.6, 10.5);
    return 0;
}
```

26. Consider the following function `f()`:

```
void f(int &i, int j, int &k) {
    i = 1;
    j = 2;
    k = 3;
    return;
}
```

(a) What is the output of the following program fragment?

```
int main() {
    int i = 10;
    int j = 20;
    int k = 30;
    f(k, j, i);
    cout << "i = " << i << " j = " << j
        << "k = " << k << endl;
    return 0;
}
```

(b) What is the output of the following program fragment?

```
int main() {
    int i = 10;
    int j = 20;
    int k = 30;
    f(j, j, j);
    cout << "i = " << i << " j = " << j
        << "k = " << k << endl;
    return 0;
}
```

27. Consider the following function `f()`:

```
void f(const int x, int &y, int z) {
    x = 1;
    y = 2;
    z = 3;
}
```

When the function is compiled which assignment statements generate a compilation error? Write Yes beside the assignment statement if a compilation error occurs, write No if a compilation error does not occur.

- (a) `x = 1;`
- (b) `y = 2;`
- (c) `z = 3;`

28. What is the output of the following C++ program?

```
#include <iostream.h>
void f(char &a) {
    cout << "char " << a << "\n";
    return;
}
void f(int &a) {
    cout << "int " << a << "\n";
    return;
}
int main() {
    int i = 1;
    char c = 'c';
    f(i);
    f(c);
    return 0;
}
```

29. How does a reference return differ from the standard return?

Suppose the following definitions are in effect.

```
class Widget {
public:
    int x;
    Widget();
    Widget(int a, int b, int c);
    int GetValue() const;
protected:
    int y;
    void SetValue(int Value);
private:
    int z;
    int PutValue(int Value);
};
```

30. How many constructors does class `Widget` have?
31. How many non-constructor member functions does class `Widget` have?
32. How many data members does class `Widget` have?
33. Is `SetValue()` defined in the class `Widget` an inspector?
34. Can the `Widget` public member functions access the `Widget` data member `x`?
35. Can a client function access the `Widget` member `x`?
36. Can a `Widget` member function access the `Widget` member `z`?
37. Does the class `Widget` support information hiding? Why?

38. Provide a class definition for a class `Coordinate` for representing two-dimensional points. There should be a single public constructor with two parameters representing the x- and y-coordinates of the object. The default value for each of these parameters is 0. The class should have two public inspectors that provide access to the individual coordinate components. The class should have two protected mutators that allow the individual coordinate components to be updated. The data members should be declared in a private section.

For the following questions make sure your answers follow the conventions developed in class and lab.

39. Write a **void** function `Return( )` with a single parameter `m` with a default value of 1. The function inserts `m` newline characters (`'\n'`) to standard output stream `cout`.
40. Write a **bool** function `IsEnd( )` that has a single value **char** parameter `c`. The function returns **true** if `c` is either a period, a question mark, or an exclamation point; otherwise, the function returns **false**.
41. Write a function `power( )` with two parameters `r` and `n`. Parameter `r` is a `Rational` and `n` is an **int**. The function should return r^n . Assume the `Rational` operators `+` and `*` are defined as in class lectures.