

Object Oriented Programming (Practice MCQs)

1. Which of the following is not a feature of OOP in C++?
 - a) Encapsulation
 - b) Inheritance
 - c) Polymorphism
 - d) Compilation

2. What is encapsulation?
 - a) Bundling data and methods that operate on the data into a single unit
 - b) The ability to create a new class from an existing class
 - c) The ability to redefine methods in derived classes
 - d) None of the above

3. Which of the following access specifiers is not available in C++?
 - a) Public
 - b) Private
 - c) Protected
 - d) Friendly

4. Which keyword is used to define a base class in C++?
 - a) base
 - b) super
 - c) class
 - d) parent

5. Which type of inheritance is not supported directly by C++?

- a) Single inheritance
- b) Multiple inheritance
- c) Multilevel inheritance
- d) Hybrid inheritance

6. What is polymorphism in C++?

- a) The ability of a function or operator to behave in different ways
- b) The process of hiding data
- c) The ability to create a new class from an existing class
- d) None of the above

7. What is a virtual function in C++?

- a) A function defined in a base class that can be overridden in a derived class
- b) A function that exists in memory but is not used
- c) A function that is called during object creation
- d) None of the above

8. What is the output of the following code?

```
class Base {
public:
    void show() { cout << "Base" << endl; }
};

class Derived : public Base {
public:
    void show() { cout << "Derived" << endl; }
};

int main() {
```

```
Base* b;
Derived d;
b = &d;
b->show();
return 0;
}
```

- a) Base
- b) Derived
- c) Compilation error
- d) Runtime error

9. What is the purpose of a constructor in C++?

- a) To deallocate memory
- b) To initialize objects
- c) To create a new class
- d) None of the above

10. Which of the following statements about destructors is true?

- a) A class can have multiple destructors
- b) Destructors are called manually by the programmer
- c) Destructors are used to release resources
- d) Destructors can be overloaded

11. What is the output of the following code?

```
class A {
public:
    A() { cout << "A"; }
    ~A() { cout << "~A"; }
};
```

```
int main() {  
    A obj;  
    return 0;  
}
```

- a) A
- b) ~A
- c) A~A
- d) Compilation error

12. Which of the following is not a type of constructor in C++?

- a) Default constructor
- b) Parameterized constructor
- c) Copy constructor
- d) Virtual constructor

13. How is dynamic polymorphism achieved in C++?

- a) Using overloaded functions
- b) Using function overriding
- c) Using function templates
- d) Using default arguments

14. Which of the following can be declared as a friend in C++?

- a) Function
- b) Class
- c) Another object

d) Both a and b

15. What is the output of the following code?

```
Class Base {  
public:  
    virtual void print() { cout << "Base"; }  
};  
  
class Derived : public Base {  
public:  
    void print() { cout << "Derived"; }  
};  
  
int main() {  
    Base* b = new Derived();  
    b->print();  
    delete b;  
    return 0;  
}
```

- a) Base
- b) Derived
- c) Compilation error
- d) Runtime error

16. Which of the following is true about pure virtual functions?

- a) They have no implementation in the base class
- b) They must be implemented in the derived class
- c) They are declared using the syntax = 0
- d) All of the above

17. What is an abstract class in C++?

- a) A class that cannot be instantiated
- b) A class with at least one pure virtual function
- c) A class with all its functions pure virtual
- d) Both a and b

18. What is the use of the *this* pointer in C++?

- a) To access the static members of the class
- b) To differentiate between local and global variables
- c) To access the object's members within the class methods
- d) None of the above

19. What is the default access specifier for members of a class in C++?

- a) Public
- b) Private
- c) Protected
- d) None

20. Which of the following is correct about operator overloading in C++?

- a) It allows defining new operators
- b) It allows using operators with user-defined data types
- c) It changes the syntax of the language
- d) None of the above

21. What is the correct way to define a copy constructor?

```
Class A {  
public:
```

```
A(const A &obj) { /*...*/ }  
};
```

- a) A(const A obj) { /.../ }
- b) A(A &obj) { /.../ }
- c) A(A obj) { /.../ }
- d) A(const A &obj) { /.../ }

22. Which of the following is a correct way to declare an array of objects in C++?

- a) ClassName obj[5];
- b) ClassName obj = new ClassName[5];
- c) ClassName obj{5};
- d) ClassName obj{};

23. Which of the following is true about inheritance in C++?

- a) Derived class inherits private members of the base class
- b) Derived class can access protected members of the base class
- c) Derived class cannot override base class methods
- d) None of the above

24. What does the *protected* access specifier mean?

- a) Members are accessible only within the same class
- b) Members are accessible within the same class and derived classes
- c) Members are accessible within the same class and friend classes
- d) Members are accessible from anywhere in the program

25. What is a virtual destructor in C++?

- a) A destructor that does nothing
- b) A destructor that can be called manually
- c) A destructor that ensures derived class destructors are called
- d) A destructor that can be overridden

26. What is the output of the following code?

```
class Base {
public:
    Base() { cout << "Base"; }
};

class Derived : public Base {
public:
    Derived() { cout << "Derived"; }
};

int main() {
    Derived obj;
    return 0;
}
```

- a) Base
- b) Derived
- c) BaseDerived
- d) DerivedBase

27. Which of the following is true about constructors and inheritance?

- a) Base class constructor is called after derived class constructor
- b) Derived class constructor is called after base class constructor

- c) Constructors are not called in inheritance
- d) Constructors are called in any order

28. How is operator overloading done in C++?

- a) Using the *operator* keyword
- b) Using function overloading
- c) Using the *overload* keyword
- d) Using inheritance

29. What does the *delete* operator do in C++?

- a) Deletes an object from memory
- b) Deletes a class
- c) Deletes a function
- d) Deletes an attribute

30. What is the output of the following code?

```
class A {
public:
    virtual void show() { cout << "A"; }
};

class B : public A {
public:
    void show() { cout << "B"; }
};

int main() {
    A* a = new B();
    a->show();
    return 0;
}
```

- a) A
- b) B
- c) AB
- d) Compilation error

Solutions:

1. d) Compilation
2. a) Bundling data and methods that operate on the data into a single unit
3. d) Friendly
4. c) class
5. d) Hybrid inheritance
6. a) The ability of a function or operator to behave in different ways
7. a) A function defined in a base class that can be overridden in a derived class
8. a) Base
9. b) To initialize objects
10. c) Destructors are used to release resources
11. c) A~A
12. d) Virtual constructor
13. b) Using function overriding
14. d) Both a and b
15. b) Derived
16. d) All of the above
17. d) Both a and b
18. c) To access the object's members within the class methods
19. b) Private
20. b) It allows using operators with user-defined data types
21. d) A(const A &obj) { /.../ }
22. a) ClassName obj[5];
23. b) Derived class can access protected members of the base class
24. b) Members are accessible within the same class and derived classes
25. c) A destructor that ensures derived class destructors are called
26. c) BaseDerived
27. b) Derived class constructor is called after base class constructor
28. a) Using the *operator* keyword
29. a) Deletes an object from memory
30. b) B