

```
{
    protected:
        int x;
    public:
        Parent(int m)
            { x = m ; }
        friend void display ( );
};
class Child : public Parent
{
    private:
        int y;
    public:
        Child(int m, int n) : Parent(m)
            { y = n ; }
};
void display ()
{
    Child c(3,4);
    cout <<"x="<<c.x<<"y="<<c.y; // Line 1
}

void main ()
{
    display();
}
```

(3 Points)

```
17. class Nice
    { int a ;
      public:
        Nice() { a = 0 ; }
        Nice(Nice & myN)
        { this -> a = myN.a ;
          cout<<"I am the copy constructor " ;
        }
        void setA(int m) { a = m ; }
        int getA() { return a ; }
    };

void show(Nice &obj)
{
    cout<<"I am the show function, value is: " << obj.getA() ;
}

void main()
{ Nice n1;
  n1.setA(15) ;
  show(n1) ;
}
```

(3 Points)

- ☐ I am the show function, value is: 15.
- ☐ I am the show function, value is: 15. I am the copy constructor.
- ☐ I am the copy constructor. I am the show function, value is: 15.

32. Which of the following is true about the function prototype below?

```
void myFunc (int myDef=17, int myVar , int myNormalVar=5) ;
```

(3 Points)

- ☒ We should also give a default value to myVar.
- ☐ We must only give a default parameter for myNormalVar and not the others.
- ☐ The function is correct in that way.

33. Assume you have a member function with the following prototype?

```
void myFunc(int x) ;
```

Which of the following are valid ways to overload it?

(3 Points)

- ☒ void myFunc(char ch) ;
- ☐ int myFunc(int x) ;
- ☒ void myFunc(char c1, char c2) ;
- ☐ float myFunc(int x, int y) ;

25. Multilevel Inheritance is allowed as an object Oriented concept, but it is not allowed in C++;

(1 Point)

- ☒ True
- ☐ False

26. What does the following piece of code do?

```
void main( )
{
    float *ptr ;
    ptr = new float[15];
}
```

(3 Points)

- ☐ Allocate space for a float variable that is not initialized
- ☒ Allocate space for an array of 15 float elements that are not initialized
- ☐ Allocate space for an array of 15 float elements that is initialized by the value 0
- ☐ Allocate space for an array of 15 float elements where all the elements are initialized by the value 15

18. If we did not specify a constructor to the class, then :  
(3 Points)

- ☐ we won't be able to create object of class
- ☐ we won't be able to create object of class, and compiler will give compilation error
- ☐ we won't be able to create object of class, and compiler will give warning
- ☐ it will generate run-time error
- ☐ None of the above

19. What will be the output when you compile and run the following piece of code?

```
class Parent
{ int y;
  static int z;
public:
    Parent( )
    {
        z=0; // Line1
    }
    Parent (int a=5) //Line 2
    {
        v=a;
```

3. What will be the output when you compile and run the following piece of code?

```
class Base
{
    public:
        Base() { cout<<"Welcome    " ; }
};
class Derived : public Base
{
    public:
        Derived() { cout<<"Hello    " ; }
};
void main()
{
    Base myBase ;
    Derived myDerived ;
}
```

(3 Points)

- ☐ Welcome Hello
- ☐ Hello Welcome
- ☐ Welcome Hello Welcome
- ☒ Welcome Welcome Hello

4. When overloading a certain function, it is obligatory that we specify a different



10. class Stack

```
{
    int tos,size;
    int * st;
    Stack( int s=5) { tos=0;size=s; st=new int[size];}
    ~ Stack() { delete []st;}
};

void main()
{
    Stack s1(20);
}
```

what shall we add to the above class Stack to declare another object s2 from class Stack where s2 is declared in terms of s1; Stack s2(s1)?

(3 Points)

- ☐ We must specify overload of assignment operator for class Stack.
- ☐ We must define a copy constructor to class Stack
- ☐ A and B
- ☐ This situation cannot be achieved in C++, however, it has been solved in other programming languages.

11.) In order for the following piece of code to compile successfully, what are the constructors that are expected to exist in the Base class?

```
class Parent
{ int y;
  static int z;
public:
    Parent( )
    {
        z=0; // Line1
    }
    Parent (int a=5) //Line 2
    {
        y=a;
    }
};

void main( )
{
    Parent d(4); //Line 3
    Parent m;    //Line 4
}
```

(3 Points)

- ☐ Compilation Error at Line 1, an object member function cannot access a static member
- ☐ Compilation Error at Line 2, constructor should initialize static member (z=0;)
- ☐ Compilation Error at Line 3
- ☐ Compilation Error at Line 4.



```
display();  
}
```

(3 Points)

- ☐ Compilation Error at Line 1, Child::x is inaccessible
- ☐ Compilation Error at Line 1, Child::y is inaccessible
- ☒ A and B
- ☐ The code compiles successfully.

31. Assume you have a class M that contains a pointer to an object of class N. Assume that we declare an object of M in the main( ) function. When will the body of the constructor of class N be executed?

(3 Points)

- ☒ When any member function of the class M is called.
- ☐ After the body of the constructor of class M is executed.
- ☐ Before the body of the constructor of class M is executed.
- ☐ None of the above.

7. The "this" pointer is an array of pointers to all the objects created from a class  
(1 Point)

☒ True

☐ False

8. When inheriting from a Base class, the Derived class will inherit only the protected, and public members of the Base class.  
(1 Point)

☐ True

☒ False

9. The term " Composition " refers to an object of a class that contains another object of another class inside it.  
(1 Point)

☒ True

☐ False

6. What will be the output when you compile and run the following piece of code?

```
class Parent
{
    protected:
        int myVar;
    public:
        Parent(int x) { myVar=x; }
        void powerTwo(){ cout<<myVar*myVar; }
        virtual void powerThree() { cout <<myVar*myVar*myVar; }
};

class Child:public Parent
{
    protected:
        int myData;
    public:
        Child(int a, int b) : Parent(a) {myData= b;}
        void powerTwo() { cout <<myData*myData; }
        void powerThree() { cout<<myData*myData*myData; }
};

void main()
{
    Child myCh(2,3);
    Parent *myPtr;
    myPtr = &myCh;
    myPtr->powerTwo(); //Line1
    myPtr->powerThree(); //Line 2
}
```

22. Which of the following most closely describes the process of overriding?

(3 Points)

- ☐ A class with the same name replaces the functionality of a class defined earlier in the hierarchy.
- ☒ A function with the same name replaces the functionality of a function defined earlier in the inheritance hierarchy.
- ☐ A function with the same name but different parameters gives multiple uses for the same function name.
- ☐ Making a class abstract so that no objects can be declared from it.

23. class Point

```
{
    float x, y;
    Point (float a, float b) { x=a; y=b;}
    Point () { x=0; y=0; }
}
```

To write copy constructor to class point, what would be its signature?

(3 Points)

- ☐ Point (Point)
- ☐ Point (Point &)

14. What will be the output when you compile and run the following piece of code?

```
class Parent
{
    int x;
    Parent(int m){ x = m ; }
};
class Child : public Parent
{ public:
    int y;
    Child(int m, int n) : Parent(m) //Line 1
    { y = n ; }
};
```

```
void main( )
{
    Child obj(3,5,7); //Line 2
    cout<<"Value of x is: "<<obj.x <<endl ; //Line 3
    cout<<"Value of y is: "<<obj.y <<endl ; //Line 4
}
```

(3 Points)

- ☐ Compiler Error at Line 1
- ☐ Compiler Error at Line 2
- ☐ Compiler Error at Line 3
- ☐ The code compiles successfully

20. "A plane is a machine that has a motor and has wings".

"A refrigerator is a machine that has a motor and has shelves".

Which of the following best describes the previous statements as a set of classes?

(3 Points)

- ☐ 1 class: A machine class that has an attribute for the type of machine.
- ☐ 2 classes: A plane class that has two attributes, and a refrigerator class that also has two attributes.
- ☐ 3 classes: A machine class that has one attribute: motor. A plane class that inherits from the machine class. And a refrigerator class that inherits from the plane class.
- ☐ 3 classes: A machine class that has one attribute: motor. A plane class that inherits from the machine class. And a refrigerator class that also inherits from the machine class.

21. We can overload Destructor in the class

(1 Point)

- ☐ True
- ☐ False



29. If a certain function is made friend for class A, then that function can access only the private members of class A.

(1 Point)

☐ True

☒ False

30. What will be the output when you compile and run the following piece of code?

```
class Parent
{
    protected:
        int x;
    public:
        Parent(int m)
            { x = m ; }
        friend void display( );
};
class Child : public Parent
{
    private:
        int y;
    public:
        Child(int m, int n) : Parent(m)
            { y = n ; }
```



23. class Point

```
{  
    float x, y;  
    Point (float a, float b) { x=a; y=b;}  
    Point () { x=0; y=0; }  
}
```

To write copy constructor to class point, what would be its signature?  
(3 Points)

- ☐ Point (Point)
- ☐ Point (Point &)
- ☐ Point & Point (Point &).
- ☐ Point & Point (Point)
- ☐ void Point (Point)
- ☐ void Point (Point &)
- ☐ None of the above

24. What will be the output when you compile and run the following piece of code?



#### Screenshot saved

The screenshot was added to your  
OneDrive.

OneDrive

```
13. class Parent
    { public:
        int x;
        Parent(int m) { x = m ; }
    };
class Child : protected Parent
{
    public:
        int y;
        Child(int m, int n) : Parent(m) { y = n ; }
};
class GrandChild : public Child
{
    int z ;
    public:
        GrandChild(int a, int b, int c) : Child(a,b) { z = c ; }
};
void main( )
{
    GrandChild obj(3,5,7);
    cout<<"Value of x is: "<<obj.x <<endl ;           //Line 1
    cout<<"Value of y is: "<<obj.y <<endl ;           //Line 2
    cout<<"Value of z is: "<<obj.z <<endl ;           //Line 3
}
```

(3 Points)

☐ Compiler Error at Line 1☐ Compiler Error at Line 2

34. Which of the following is true about an object member function?

(4 Points)

- ☐ It can be called using the name of the class.
- ☒ It can access static variables of the class.
- ☒ It has a "this" pointer as an implicit parameter passed to it.
- ☐ It can access the instance variables.
- ☐ It cannot be overloaded.
- ☐ It can call other member functions from inside it.

35.

```
class Super
{
    protected:
        Super(int a)
        {
            this.a = a; // Line 1
        }
    private:
        int a; // Line 2
}
```

16. A protected member of a class can be directly accessed by its name inside another class if and only if that other class is a child of that class.  
(1 Point)

☒ True

☐ False

17. class Nice  
    { int a ;  
      public:  
        Nice() { a = 0 ; }  
        Nice(Nice & myN)  
          { this -> a = myN.a ;  
            cout << "I am the copy constructor " ;  
          }  
        void setA(int m) { a = m ; }  
        int getA() { return a ; }  
    };  
void show(Nice &obj)  
{  
  cout << "I am the show function, value is: " << obj.getA() ;  
}

15. What will be the output when you compile and run the following piece of code?

```
class Test
{
    int x;
    public:
        Test() { x = 0; }
        Test(int y) { x = y++; }
        Test(Test &r) { x = ++r.x; }
        void print() { cout<<x; }
};

void main()
{
    Test t(1);
    t.print();
    Test x(t);
    x.print();
    t.print();
}
```

(3 Points)

- ☐ 121
- ☐ 122
- ☐ 222
- ☐ 233