Innovators – Assessment

OOPS

## How can you implement multiple inheritance in C#?

A)Interface

2) Select the sequence of execution of function f1(), f2() & f3() in C# .NET CODE?

1. **class** **base**
2. {
3. **public** **void** f1() {}
4. **public** **virtual** **void** f2() {}
5. **public** **virtual** **void** f3() {}
6. }
7. **class** derived :**base**
8. {
9. new **public** **void** f1() {}
10. **public** **override** **void** f2() {}
11. **public** new **void** f3() {}
12. }
13. **class** Program
14. {
15. **static** **void** Main(**string**[] args)
16. {
17. baseclass b = new derived();
18. b.f1 ();
19. b.f2 ();
20. b.f3 ();
21. }

a)

f1() of derived class get executed  
 f2() of derived class get executed  
 f3() of base class get executed

b)

f1() of base class get executed  
 f2() of derived class get executed  
 f3() of base class get executed

c)

f1() of base class get executed  
 f2() of derived class get executed  
 f3() of derived class get executed

d)

f1() of derived class get executed  
 f2() of base class get executed  
 f3() of base class get executed

3) Which of the following, inheritance provides facilities?

1. Inheritance is used to reuse existing facilities of the parent class.
2. We can override the existing functionalities of the superclass.
3. We can also implement new functionality in the child class.
4. We can implement polymorphic behavior using inheritance.
5. Only A
6. Only B
7. Only C
8. A, B and C

4) Which is not a feature of OOP in general definitions?  
a) Efficient Code  
b) Code reusability  
c) Modularity  
d) Duplicate/Redundant data

5)

Define a class to represent a bank account. Include the following members:

Data Members:

1) Name of depositor

1. Account number
2. Type of account
3. Balance amount in the accountMember

Functions: 1) to assign initial values

2) To deposit an amount

3) To withdraw amount after checking the balance

4) To display name and balance. **Write a main program and access the classes**.

using System;

class bankaccount

{

//data members

public double Nameofdepositor;

public double Accountnumber; //if we put private here then we can only access the data member/ member function inside that class only

public double Typeofaccount;

public double Balanceamount;

public double initialva()

{

return initialva();

}

public double depositanamount()

{

return depositanamount();

}

public double checkbalance()

{

return checkbalance();

}

public void Display()

{

Console.WriteLine("to assign initial values", initialva());

Console.WriteLine("To deposit an amount", depositanamount());

Console.WriteLine("Balance is 10k", checkbalance());

}

}//end class

class Bank //main class

{

static void Main(string[] args) //main method it fetch the data from the two derived class as wwell from the base class

{

Bankperson Bankk = new Bankperson();

Bankk.initialva();

Bankk.depositanamount();

Bankk.checkbalance();

Bankk.Display();

//Console.ReadLine(); //if we entered this code it will select as enter to the next line

}

}

6) Wrong statement about run time polymorphism is?  
An abstract inherited property cannot be overridden in a derived class

7) Explain runtime polymorphism with a real time example using class, data members and functions.

Can’t to be done because we need to virtual class

8) When is method overloading determined?

At compile time

9) What makes an interface different from a class?

A) The interface which is used for the multiple inheritance because in the multiple inheritance there we have two parent classes and one child classes so the methods from the base classes will conflict.so solving the issue we are using the interface. The keyword for interface is interface.

Classes is a collection of objects. We can initial the code using the class

10) We have a car class and a tricyclic class and a truck class. Each of these three classes should have a start engine() action. How the "engine is started" for each vehicle is left to each particular class, but the fact that **they must** have a start engine action is the domain of the interface.

Explain using the concept of **Interface**.

using System;

interface car

{

void engine();

}

interface tricyclic

{

void engine();

}

interface truck()

{

void engine();

}

class Demo : car, tricyclic, truck

{

public void engine()

{

Console.Write("Jato ");

}

public void engine()

{

Console.WriteLine("paul");

}

public void engine()

{

Console.WriteLine("qwety");

}

}

class TestInterface

{

static void Main(string[] args)

{

Demo name = new Demo();

name.engine();

name.engine();

name.engine();

}

}