**141. What are interfaces?**

\* What is interface?

\* Multiple Inheritance vs Interface.

\* Example of Interface.

\* Dos and Don’ts of interface.

1. Abstract is useful for polymorphism.

2. Interface is completely used for polymorphism.

3. Interface can be called as abstract class with all abstract methods.

4. Interface helps us achieve 100% abstraction.

5. Classes are extended. Interface are implemented.

6. The purpose of interfaces is to achieve polymorphism.

7. We can have reference of interface and object of sub class.

8. In java a class can extend only one class.

9. A class can implement multiple interfaces.

9. Members of interface are always declare as const, their values are final.

**142. Practicing Interfaces.**

1. Q1:

2. You cannot create an object of interface.

3. Interface are implemented, not extended.

4. A class becomes abstract if it does not override all methods inherited from interface.

5. A reference of an interface can an object of class which implements that interface.

6. Interfaces are meant only for overriding.

7. They are meant for achieving runtime polymorphism and dynamic dispatch.

8. A interface is the collection of abstract methods.

**143. Example of Interface.**

1. Q1.

class Phone.

interface ICamera. //I stands for interface as per naming convention.

interface Music Player.

2. Interfaces can have references but not object.

3. References interface can hold object of sub class and call methods which are defined in that interface.

**144. Demo: Example For Interface.**

**145. Student Challenge: Call Back Methods.**

1. The common use of interface is to define call back methods.

2. Call back methods are used in event handling.

3. Q1.

class Store

{

Member mem[] = new Member[10];

int count = 0;

}

void inviteSale()

{

for(int = 0; I <count; i++

}

interface member

{

void callback();

}

class Customer implements Member

{

String name;

Customer(String n)

{

name = n;

}

void callback()

{

System.out.println(“Ok, I will visit.”+name);

}

}

**146. Dos and Don’ts.**

1. By default, the methods are public and abstract.

2. Methods cannot be private.

3. We can have identifiers inside interface which are named in capital letters. These identifiers are by default static and final.

4. Methods cannot have body.

5. An interface can have static methods with body.

6. A interface can extend from another interface.

7. All method should be overridden.

8. Default methods are also allowed in interface.

default void meth3()

{

S.o.p(“Meth 3”);

}

9. Default methods are used to modify the interface so other sub classes are not disturbed. (Think in terms of long and big project.) To update and modify existing interface without disturbing the existing child classes.

10. We can write private which is not abstract.

private void meth3()

{

S.o.p(“/…”);

}

Can be used only inside default methods of interface.

**147. Interface vs Multiple Inheritance.**

1. Java doesn’t support multiple inheritance because of its design.

2. In C++ we can inherit multiple classes. It allows multiple inheritance.

3. In Java, a smartphone is a phone having features like camera, music player.

4. This is the right thinking in context of OOP’s.

5. It like a class inherit important properties from one class/ interface but additional features from other interfaces.

6. Java is righter and C++ is wrong in context of OOP’s. Interface are perfect than multiple inheritance.

7. For example:

Suzuki car is a car having features like music players.

8.