Çoklu Kalıtım Nedir?

Çoklu kalıtım bir child class ın birden fazla sınıftan miras almasıyla ortaya çıkıyor.Java çoklu kalıtımı desteklemiyor.Javada aynı anda birden fazla sınıftan miras alınamıyor.Mesela iki print fonksiyonuna sahip sınıfımız var.Bu sınıflar abstract değil.Yani iki sınıftaki print fonksiyonlarının da body kısmı var.Bu iki sınıftan miras alan alt sınıfın print fonksiyonu olmadığı durumda hangi üstsınıfın print fonksiyonu çağrılır?Bu derleyici için bir karışıklık buna karar veremez.Eğer altsınıfa ait print fonksiyonu olsaydı program onu kullanacaktı.Olmadığı durumda üst sınıfa gidip çağırması gerekiyor.Bu durumda iki üst sınıftan hangi print fonksiyonu çalıştırılacak?Bu çoklu kalıtımın temel problemi ve java bu yüzden izin vermiyor.

```
public class A {
	public void print() {
		System.out.println("A class print function");
	}

}
public class B {
	public void print() {
		System.out.println("B class print function");
	}

}
public class C extends A,B {


}
```

Java bu ihtiyaca interface yapısı yardımıyla çözüm buluyor.Java sınıflar istediği kadar interface kullanabilir.A ve B yi interface olarak tanımlayıp print fonksiyonları çalıştırılabilir.Bu durumda iki sınıfın fonksiyonun body kısmı olmayacak.Yani fonksiyonun tanımlamasını A ve B yi implement eden C sınıfında tanımlamak zorundayız.C deki print fonksiyonu çalıştırmak istediğimizde fonksiyonu burda tamamladığımız için derleyici bir karışıklıkla karşılaşmıyor.

```
{
        C obj;
        obj.printMessage();
}


public interface A {
    public void print();
}

public interface B {
   public void print();
}
public class C implements A,B{
        public void print() {
                System.out.println("Hello");

}
```

Java nın aksine c++ çoklu kalıtımı destekliyor.Fonksiyonları çağırırken karşılaşılan karışıklığa çözüm olarak fonksiyon çağırılırken scope belirtilip hangi üst sınıftan çağrılacağına karar verilmiş oluyor.

```
class A{
        public:
                void printMessage(){
                   cout<<"Class A print message function"<<end1;
                }
};
class B{
        public:
                void printMessage() {
                cout<<"Class B print message function"<<endl;"
        }
};

class C : public A,public B{
        public:
                void printMessage() {
                A::printMessage();
                //scope belirleniyor.Böylece hangi sınıfın
                //fonksiyonunun çağrılacağı belirleniyor.
        }
};

int main()
{
        C obj;
        obj.printMessage();
}
```