SNS COLLEGE OF ENGINEERING

Kurumbapalayam(Po), Coimbatore – 641 107
Accredited by NAAC-UGC with ‘A’ Grade
Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

Department of Information Technology

19CS204 OBJECT ORIENTED PROGRAMMING

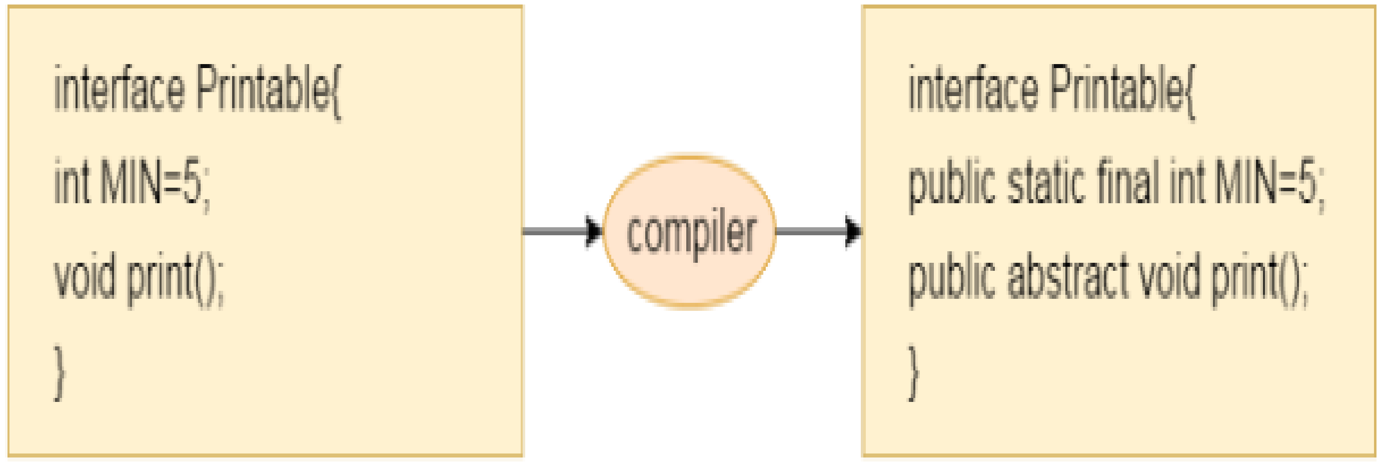
I YEAR /II SEMESTER

Topic – Interface

Interface

Rules

- Like abstract classes, interfaces cannot be used to create objects.
- Interface methods do not have a body - the body is provided by the "implement" class
- On implementation of an interface, you must override all of its methods
- Interface methods are by default abstract and public
- Interface attributes are by default public, static and final
- An interface cannot contain a constructor (as it cannot be used to create objects)



```
interface Printable{
int MIN=5;
void print();
}
```

```
interface Printable{
public static final int MIN=5;
public abstract void print();
}
```

Interface

Interface Uses

- To Achieve Abstraction (This also adds security)

- To Achieve Multiple Inheritance (Java does not support multiple inheritance but using interface we can achieve it)

What’s allowed?

- JDK 8 accepts
- Method body but it should be default or static
- Default methods, Static methods and Private methods
- Private methods from java 9

Interface

Example

```
interface example
{
    int a=10;// 4. By default all the variable are public, static and final
    void show();
    static void display() // 2. Methods do not have body 3.Shoud be overridden in sample class
    {
        System.out.println("Hello");
    }
}
class sample implements example
{
    public void show()
    {
        System.out.println("Value of a is " +a);
    }
    /*public void display()
    {
        System.out.println("Hello");
    }*/
}
```

Interface

Example

```
public class test
{
    public static void main(String[] args)
    {
        example x=new sample(); // 1. cannot create objects for interface
        x.show();
        example.display();
    }
}
```

Multiple Interface

```
interface square
{
   void square(int a);
}
interface rectangle
{
   void rectangle(int l,int w);
}
class area implements square, rectangle
{
   public void square(int a)
   {
      a = a*a;
      System.out.println("Area of square is " + a);
   }
   public void rectangle(int l, int w)
   {
      int a;
      a=l*w;
      System.out.println("Area of rectangle is " +a);
   }
}
```

```
public class test
{
   public static void main(String[] args)
   {
      area a1=new area();
      a1.square(5);
      a1.rectangle(5,4);
   }
}
```

Extending Interface

- One interface can inherit another by use of the keyword extends.
- Similar to classes, interfaces can extend other interfaces.
- The extends keyword is used for extending interfaces.
- The syntax is the same as for inheriting classes.
- When a class implements an interface that inherits another interface, it must provide implementations for all methods required by the interface inheritance.

Extending Interface

```
interface square
{
    int a=2;
    void square();
}
interface cube extends square
{
    void cube();
}

class area implements cube
{
    public void square()
    {
        int b;
        b = a*a;
        System.out.println("Area of square is " + b);
    }
    public void cube()
    {
        int b;
        b=6*(a*a);
        System.out.println("Area of cube is " +b);
    }
}

}
```

```
public class test
{
    public static void main(String[] args)
    {
        area a1=new area();
        a1.square();
        a1.cube();
    }
```

THANK YOU