1. **Without public static final**

If we declare a variable without of these key words, it becomes a regular(normal) variable. That means each object of the class has its own copy of the variable.

**With public statis final**

public keyword makes the variable accessible from outside the class.

static keyword makes the variable associated with the class itself and not with any specific object. Only one copy of the variable exists, regardless of the number of objects created.

final keyword makes the variable constant. Which means it’s value cannot be changed after initialization.

**When to use each approach**

Use regular variables when you need different values for each object of the class, and the value may change between different objects.

Use public static final class variables (constants) when you want to define a value that is shared among all objects, methods of the class and remains constant throughout the program's execution.

2. **With abstract method**

when we use abstract keyword to a method, that method has default implementation in abstract class and must be overridden in the sub class.

Abstract methos dint have body and its end with a semicolon.

Subclasses that extend the abstract class are required to provide an implementation for all the abstract methods inherited from the abstract superclass. If a subclass doesn't provide an implementation for any abstract method, the subclass must also be declared as abstract.

**Without abstract method**

if we don’t use abstract keywords for a method that means that methos has default implementation in abstract class

Sub classes are not required to provide a override method, in this case subclasses are free to override a method

public interface MyFirstInterface {

int x = 10;

void display();

}

public class InterfaceImplemented implements MyFirstInterface {

@Override

public void display() {

x = 20;

System.out.println("Value of x: " + x);

}

}

interface Speaker{

void speak();

}

class Politician implements Speaker{

@Override

public void Speak()

{

System.out.println(“Politician speak”);

}

}

class Priest implements Speaker{

@Override

public void Speak()

{

System.out.println(“Priest speek”);

}

}

class Lecturer implements Speaker{

@Override

public void Speak()

{

System.out.println(“Lecturer speak”);

}

}

class test{

public static void main(String[]args)

{

politician p=new politician;

Priest o=new Priest;

Lecturer d=new Lecturer;

o.Speak();

d.Speak();

p.Speak();

}

}