We can use final keyword:

1. Before class name
2. Before variable name
3. Before function name

1. Before class name

Can not inherit a final class.

```
final class A{

}

class B extends A{  // This will not work as A is final

}
```

2. Before variable name

final variables can be set once, and can not be changed.

We can set final variables during declaration, or in the constructor;

```
class A{
    final int in1 = 10;
    final int in2;

    A(){
        in2 = 5;
    }
}
```

3. Before function name

final functions can not be overridden. But final does not stop overloading.

```
class A{
    public final void fnc(){
        System.out.println("in A");
    }
}

class B extends A{
    // can not override fnc() as it is final in A
    public void fnc(){
        System.out.println("in B");
    }
}
```

Static keyword

1. Before variables
2. Before functions

1. **Before variables:**

Static variables are linked to the class, rather than objects. Only one copy of static variable is created and it does not depend on the number of objects.

Can access using: classname.staticVariableName.

Can access without creating objects.

```
// static keyword

class A{
    static int stA;
    static int countObj;
    int b;

    A(){
        countObj++;
    }

    void fnc(){
        stA++;
    }
```

```
    void fnc2(int p){
        stA += p;
    }
}

public class Main {
    public static void main(String[] args) {
        A.stA = 20;
        System.out.println(A.countObj);

        A obj1 = new A();
        System.out.println(A.countObj);

        A obj2 = new A();
        System.out.println(A.countObj);

        obj1.fnc();
        obj2.fnc2(15);
        System.out.println(A.stA);

        obj1.stA = 50; // Better not to access
                       // static variable this way
    }
}
```

2. Before functions:

We can call static functions without creating an object of that class.

We can only access static members of a class inside static method.

```
// static keyword

class A{
    static int stA;
    int b;

    static void fnc(){
        stA++;
    }

    void fnc2(int p){
        stA += p;
        b++;
    }
}
```

```java
public class Main {
    public static void main(String[] args) {
        A.fnc();
        System.out.println(A.stA);

        int m = Math.max(20, 10);
        System.out.println(m);
    }
}
```