```
class StaticTest {
    public static void main(String[] a)
    {
        StatTest testObj1 = new StatTest();
        StatTest testObj2 = new StatTest();
        StatTest testObj3 = new StatTest();

        testObj1.print();
        testObj2.print();
        testObj3.print();

        testObj1.print();
        testObj2.print();
        testObj3.print();

        // Metoden getAntalObjekt är static -> anrop via olika objekt eller klassen gör ingen skillnad.
        // Modifieraren static gör att metoden bara finns i klassen och inte i objekten av klassen.
        // När metoden anropas via objekten så kommer ändå metoden i klassen att anropas.
        System.out.println("Antal StatTest-objekt: " + testObj1.getAntalObjekt());
        System.out.println("Antal StatTest-objekt: " + testObj2.getAntalObjekt());
        System.out.println("Antal StatTest-objekt: " + StatTest.getAntalObjekt());
    }
}
```

Utskriften:

```
StatTest, utskrift nr 1
StatTest, utskrift nr 2
StatTest, utskrift nr 3
StatTest, utskrift nr 4
StatTest, utskrift nr 5
StatTest, utskrift nr 6
Antal StatTest-objekt: 3
Antal StatTest-objekt: 3
Antal StatTest-objekt: 3
```

```
class StatTest
{
    // Dessa variabler(attribut) kommer bara att finnas i klassen
    // inte i objekt av klassen men objekten kommer åt dem.
    // Man kan säga att de är gemensamma för alla objekt av klassen.
    // Och även för subklasser av klassen.
    protected static int antalPrint=0;
    private static int antalObjekt=0;

    public StatTest()
    {
        antalObjekt++;
    }

    // Denna metod finns endast i klassen och inte i objekt av klassen,
    // men är åtkomlig via objekten,
    // såväl som direkt via klassen och subklasser av klassen.
    public static int getAntalObjekt()
    {
        return antalObjekt;
    }// SLUT PÅ metoden getAntalObjekt()

    public void print()
    {
        antalPrint++;
        System.out.println("StatTest, utskrift nr " + antalPrint);
    }// SLUT PÅ metoden print()

}// SLUT PÅ klassen StatTest


// Nedanstående har ingenting med ovanstående att göra, bara ett exempel på vad man får göra med final och abstract.
// Så här kan man inte göra, en final-klass kan inte ha en abstrakt metod. Fundera på varför!
final class finalTest
{
    public abstract void abstraktMetod();
}
```