Laboration 1

Laborationen består av att lösa minst en av följande uppgifter. Du kan själv välja vilken av uppgifterna du vill göra. Det är inte förbjudet att göra mer än en av uppgifterna.

Alt 1.
Skriv en Javaklass som returnerar en lottorad som en sträng. Lottoraden skall vara sorterad när den skrivs ut. Skriv också en mainmetod där du i en meny kan välja hur många lottorader du vill skapa och skriva ut.
Fundera, innan du börjar, på vad som skall vara attribut och vilka metoder som behöver vara publika.

Alt 2.
Skriv en Javaklass som kan hantera en vektor med 100 tal. Talen skall slumpas, sorteras och man skall kunna göra en binärsökning. Man skall även kunna få information om största och minsta tal samt medelvärde och median (jmf lab 1 på C-kursen).
Ni skall styra implementationen så att man inte kan söka i en osorterad vektor eller sortera en vektor utan att först slumpa tal.
Skriv också en mainmetod där du visar hur du kan använda ett objekt ur din klass. Skall menyn vara en del av objektet eller ligga i main?

Alt 3. (lite svårare)
Implementera klassen Bilregister enligt nedanstående specifikation. Klassen ägare kan (bör) ersättas med en String (alltså ägare är en sträng som innehåller ett namn).
Skriv också en mainmetod där du visar hur du kan använda ett objekt ur din klass.

```
public class Bilregister
{
        // defaultkonstruktor
        // pre: true
        // post: Nytt bilregister med storlek 8 skapat
        public Bilregister()

        // konstruktor som sätter storleken på registret till size
        // pre: true
        // post: Nytt bilregister med storlek size skapat
        public Bilregister(int size)

        // add: lägger till ny ägare
        // pre: nyÄgare till ett visst fordon skapad, registret inte fullt
        // post: nyÄgare tillagd i registret
        public void add(Ägare nyÄgare)

        // remove: tar bort ägare
        // pre: position laglig (0 <=pos && pos < maxSize())
        // post: ägare på position pos borttagen
        // ägare på högre positioner har skiftats ner för att undvika
        // "hål" i lagringsutrymmet.
        public void remove(int pos)
```

```
    // getÄgare: hämtar ägare på position pos i registret
    // pre: position laglig (0 <=pos && pos < maxSize())
    // post: ägare på position pos returnerad
    public Ägare getÄgare(int pos)

    // size: returnerar antalet ägare i registret
    // pre: true
    // post: antalet ägare i registret returnerat
    public int size()

    // maxSize: returnerar maximal storlek på registret
    // pre: true
    // post: maximal storlek för registret returnerad
    public int maxSize()
}
```