# Inlämningsuppgift 2 nr 3

### Beskrivning av uppgiften

Du skall skriva ett program som hanterar ett register för CD-skivor (med artist och titel). Med programmets hjälp skall man kunna utföra följande:

1. Sätta in ett ny skiva med artist och titel.
2. Ta bort alla skivor av en viss artist.
3. Söka upp alla skivor som gäller för en viss artist. Det skall räcka att man anger en del av artistens namn. Samtliga skivor vars artistnamn passar in skall presenteras med både artist och titel.
4. Söka upp den eller de skivor som har en viss titel. Skivorna skall presenteras med både artist och titel.
5. Presentera alla skivor sorterade efter artist.
6. Presentera alla skivor sorterade efter titel.
7. Spara registret på en fil.
8. Hämta uppgifterna till registret från en fil.

### Programmet

Följande klasser skall finnas i lösningen:

*CD:* som beskriver en CD-skiva med artist och titel. Klassen är given nedan.

*Dialog:* som innehåller operationer för dialog med användaren. Här finns operationer för att fråga efter heltal (t ex ett menyval) och strängar samt utskrift av strängar. Klassen är färdigskriven.

*Register:* Denna klass skall du skriva. Klassen skall vara uppbyggd enligt anvisningarna nedan och skall innehålla operationer för insättning respektive borttagning i registret, sökning i registret, utskrift av registret samt operationer för läsning respektive skrivning av registerinnehållet på fil.

*En klass med main-metoden:* Denna klass skall du skriva.

Implementation av den givna klassen CD (kopiera koden till en egen fil CD.java):

```
public class CD {
   private String artist;   // artist
   private String title;    // titel

   /** Skapar en cd med artisten artist och titeln title */
   public CD(String artist, String title) {
      this.artist = artist;
      this.title = title;
   }

   /** Returnerar namnet på artisten */
   public String getArtist() {
      return artist;
   }

   /** Returnerar titeln */
   public String getTitle() {
      return title;
   }

   /** Returnerar en sträng som består av skivans artist och titel */
   public String toString() {
      return artist + "\t" + title;
   }
}
```

Specifikation av den färdiga klassen Dialog (finns att ladda ned från kurshemsidan):

```
/** Skapar ett Dialog-objekt för hantering av popup-dialogrutor. */
Dialog();

/** Visar en dialogruta med hjälptexten s där användaren skall mata in ett
  * heltal som returneras.
  * Om användaren anger ett felaktigt värde eller klickar på "avbryt" så
  * returneras Integer.MAX_VALUE.
  */
int readInt(String s);

/** Visar en dialogruta med hjälptexten s där användaren skall mata in en
  * teckensträng som returneras.
  * Om användaren klickar på "avbryt" så returneras null.
  */
String readString(String s);

/** Visar en dialogruta med textsträngen s. */
void printString(String s);
```

**Anvisningar och ledningar**

- Skapa en egen fil CD.java och lägg in den givna koden för klassen ovan.
- Den givna klassen Dialog hämtas från kurshemsidan. Lägg den i samma katalog som dina egna klasser och kompilera den.
- Så här skall klassen Register vara uppbyggd:

```
import java.util.ArrayList;
public class Register {
   private ArrayList<CD> reg;   // registret skall lagras i en ArrayList

   //...  Konstruktor och alla metoderna för registerhanteringen

   /** Läser registret från filen med namn fileName. */
   public void readFromFile(String fileName) {
      //...  Se ledning för filhanteringen sist i uppgiften
   }

   /** Sparar registret på fil med namnet fileName. */
   public void writeToFile(String fileName) {
      //...  Se ledning för filhanteringen sist i uppgiften
   }
}
```

- Registret skall hela tiden hållas sorterat med avseende på artist. Det går bra att använda linjärsökning för att hitta rätt insättningsposition men då registret är sorterat är det möjligt att istället använda binärsökning för att hitta rätt insättningsposition (se läroboken avsnitt 12.5). Ni som vill utnyttja denna möjlighet får gärna göra det även om avsnitt 12.5 formellt inte ingår i kursen. (Not till avsnitt 12.5: Ni kan dock inte använda den färdiga metoden Collections.binarySearch i java.util då vår klass Person inte implementerar Comparable, vilket gås igenom först i fortsättningskursen.)
- Att registret är sorterat på artist underlättar t ex vid en sorterad utsökning av uppgifterna efter artist. En sorterad utsökning av uppgifterna efter titel (som registret inte är sorterat på) är dock mer komplicerad. Där kan man t ex använda följande algoritm:

```
bilda något att lagra utsökningsresultat i (t ex ett StringBuilder-objekt)
bilda en boolean-vektor help (där alla element är false från början)
för i = 0 till size–1:
   min = stort värde
   för k = 0 till size–1:
      om help[k] är false och värdet på plats k < min:
         index = k;
         min = värdet på plats k;
   lägg till min sist i utsökningresultatet
   help[index] = true;
```

- All dialog med användaren (inläsning från tangentbordet och utskrift på skärmen) skall ske via klassen Dialog i huvudprogrammet. Dvs inga utskrifter på skärmen inifrån klassen Register.
- Innehållet i registret lagras i form av strängar liksom alla hjälptexter till metoderna i klassen Dialog. Följande kan vara bra att känna till om klassen String:
  - Det går bra att lägga in en radbrytning (tecknet '\n') mitt i en sträng, vilket t ex kan utnyttjas om man från en String-metod (en metod med returtyp String) vill returnera "flera" strängar eller om man i en hjälptext till metoderna i klassen Dialog vill ha flera svarsalternativ. T ex kan man göra så här för att skapa en menysträng (i exemplet finns endast tre menyalternativ, lägg själv till ytterligare alternativ):

```
String menuItems = "Meny" + "\n"
                 + "1: Sätt in en ny skiva" + "\n"
                 + "2: Tag bort en artists skivor" + "\n"
                 + "3: Sök skivor från artistnamn/del av artistnamn";
```

  - För att bygga upp strängar som består av flera registerelement kan man t ex använda klassen StringBuilder.
  - Två strängar kan jämföras med varandra med metoden `int compareTo(String s)`. Se läroboken avsnitt 11.2.
  - För att undersöka om en sträng ingår i en annan sträng kan man använda metoden `int indexOf(String s)`. Exempel:

```
String name = "Kalle Anka";
String soughtName = "Anka";
if (name.indexOf(soughtName) >= 0) {
   System.out.println(soughtName + " ingår i namnet " + name);
}
```

    Metoden indexOf returnerar –1 om strängen soughtName inte ingår i name, annars returnerar den positionen för början av den första förekomsten av soughtName i name.
  - Metoderna compareTo och indexOf skiljer båda på stora och små bokstäver. På följande sätt kan man konvertera strängar till små bokstäver vid jämförelser:

```
String a = ...;
String b = ...;
if (a.toLowerCase().compareTo(b.toLowerCase()) == 0) ...
if (name.toLowerCase().indexOf(soughtName.toLowerCase()) >= 0) ...
```

    Alternativt kan man utnyttja konverteringsmöjligheterna redan när man lägger in element i registret.
- Main-metoden kan ha följande struktur:

```
...
while (true){
   ...
   int command = önskat kommando;
   switch (command) {
      case 1: Fråga efter artist;
              Fråga efter titel;
              Sätt in skivan i registret;
              break;
      case 2: ...
      ...
   }
}
```

**Filhanteringsanvisningar**

- I metoden readFromFile skall du läsa in ett sparat register från en textfil och i metoden writeToFile skall du skriva ut det befintliga registret på en textfil. Grundtanken är att man i början av programmet läser in ett sparat register från fil, därefter arbetar man med registret (söker, lägger till, tar bort, osv.). Precis innan man avslutar sitt program så skriver man ut programmet på filen igen så att det aktuella registret finns tillgängligt nästa gång man startar programmet. För att detta skall fungera måste man bestämma sig för i vilken ordning man skriver ut registerinnehållet på filen, det måste ju sedan läsas i samma ordning. Ett lämpligt format kan vara att varje skiva skrivs på två rader enligt:

```
Beatles
Abbey Road
ABBA
Waterloo
...
```

- Tips: Vänta med att skriva metoden readFromFile tills du skrivit och verkligen testat din insättningsmetod. Implementera sedan readFromFile så att man där anropar insättningsmetoden för varje skiva man läser in.

- Anvisningar för hur man läser från fil finns i läroboken avsnitt 7.8. Jämför följande program:

```
import java.util.Scanner;
import java.io.File;
import java.io.FileNotFoundException;
public class ScannerFileExample {
   public static void main(String[] args) {
      Scanner scan = null;
      try {
         scan = new Scanner(new File("data.txt"));
      } catch (FileNotFoundException e) {
         System.err.println("Filen kunde inte öppnas");
         System.exit(1);
      }
      ... läs nu som vanligt med scan
   }
}
```

- Anvisningar för hur man skriver till fil finns i läroboken avsnitt 7.9. Jämför följande program:

```
import java.io.PrintWriter;
import java.io.File;
import java.io.FileNotFoundException;
public class PrintWriterFileExample {
   public static void main(String[] args) {
      PrintWriter out = null;
      try {
         out = new PrintWriter(new File("data.txt"));
      } catch (FileNotFoundException e) {
         System.err.println("Filen kunde inte öppnas");
         System.exit(1);
      }
      ... utskrifter med out.print hamnar nu på filen
      out.close();      // stänger filen
}
```