# DA339A Laboration L16

## Syfte

Den här laborationens syfte är att öva på att jobba med flera klasser, och array av objekt. Laborationen är också avsedd att ge övning i objektorientering och en av dess huvudprinciper, nämligen inkapsling. Laborationen syftar också på att ge möjligheten att använda konstruktorer som en viktig del av en klass.

Syfte är att studenten ska öva programmering och problemlösning med flera klasser som tillsammans få en applikation att utför sina uppgifter. Lösningarna skall skrivas som körbara Java-program.

## Redovisning

**Den här laborationen ska redovisas.**

Laborationen redovisas muntligt och genom uppvisande av källkod samt kompilering och exekvering av kod. Studenten ska kunna visa upp sin källkod på dator för lärare/assistent som examinerar och kunna visa hur hen kompilerar och exekverar koden på dator. Studenten ska också kunna svara tillfredställande på frågor om sin kod och de lösningar hen implementerat. Studenten ska även kunna visa upp andra lösningar som krävs i laborationen exempelvis diagram eller sanningstabeller.

För godkänd redovisning krävs:

- Väl formaterad källkod.
- Källkod ska gå att kompilera och exekvera med ett resultat som efterfrågas för respektive uppgift.
- Att uppgifterna är lösta på rimligt sätt oavsett om det är i kod eller annan form av dokumentation av lösning som efterfrågas.
- Studenten ska muntligen kunna förklara sina lösningar och beskriva varför lösningen ger det efterfrågade resultatet.

Ingen skriftlig inlämning sker i redovisningen.

# Förberedelser

Den här laborationen är direkt kopplad till det innehåll som finns i föreläsning F13, F14 och F15. För att kunna genomföra laborationen behöver studenten ha bekantat dig med begrepp från de nämnda föreläsningarna, framför allt att strukturera applikationen i olika klasser, skriva och använda konstruktorer och hantera array av objekt (en array som kan fyllas med en viss typ av objekt, till exempel Person[] personArray är en array som innehåller Person objekt).

För denna laboration förutsätts det att du även behärskat tidigare genomgånget innehåll på kursen.

Du behöver ladda ner en zip-fil som behövs för att lösa sista uppgift i laborationen. Zip-filen du behöver ladda ner från Canvas är *ProductManager.zip*.

# Uppgifter

Lösningarna till uppgifterna ska skrivas i Java-kod och köras i IntelliJ eller liknande program (dock kommer alla instruktioner vara baserade på att IntelliJ används).

Skapa ett nytt projekt i IntelliJ för denna labb och spara det någonstans på din dator. Skapa sedan ett paket för varje deluppgift.

## Uppgift 1a: Array av objekt, lägga till och ta bort ett element

Skapa ett paket **Uppgift1** i projektet. I detta projekt skall vi jobba med två klasser, en klass **Login** för att hantera ID och lösenord och en testklass **Main** (brukar skapas automatiskt när du skapar ett nytt projekt) för att starta programmet och testa.

**Att göra:**

- Skriv en klass **Login** och komplettera koden enligt figuren som följer. Ett lösningsförslag till denna deluppgift finns i slutet av dokumentet.

```
public class Login{
    private String id;
    private String password;

    //ToDo 1:  Skriv en konstruktor med 2 parameterar
    //        för initiering av id och password

    //ToDo 2:  Skriv en toString

}
```

- Skapa en klass **Main** och komplettera enligt nedan (om en Main klass redan skapats, skriv endast in de två kodraderna inne i main-metoden):

```
package Uppgift1;

public class Main
{
    public static void main(String[] args)
    {
        Login login = new Login("homer@simpson.se", "homer2022");
        System.out.println(login);
    }
}
```

- Kompilera och testa inifrån IntelliJ. Följande utskrift bör testen visa.

```
ID: homer@simpson.se, Password: homer2022
```

## Uppgift 1b

I denna del skall programmet kunna spara flera instanser (objekt) av klassen **Login**.

- Skapa en ny klass **LoginHandler** och deklarera en array av Login.
- Skriv en konstruktor med en parameter för antalet element som arrayen skall dimensioneras för, dvs. max antalet element. Använd koden nedan för att komma i gång med denna klass:

```
package Uppgift1;

public class LoginHandler
{
    private Login[] loginList;
    private int numOfElements; //Num of registered (saved) elements. Not max
                               // elements

    public LoginHandler(int maxElements)
    {
        loginList = new Login[maxElements];
        numOfElements = 0;
    }
    //resten av koden
```

- Skriv en metod som ska lägga till ett nytt element av Login i arrayen. Använd koden nedan och komplettera där det står ToDo.

```
   //Purpose: to add a new object of Login at the end of the list
   //End of the list is the position equal to numOfElements
   //Input: loginIn - object to add
   //Return: true if success, false otherwise

   public boolean addNew(Login loginIn)
   {
       boolean success = true;
       //validate so loginIn is an object and that the array is not full

        //ToDo:  Komplettera enlig kommentarerna ovanför metoddefinitioen
        //Increment number of elements

     return success;
   }
```

- Skriv en metod som tar bort (nollställer) en viss position i arrayen

```
//Purpose: to delete an existing element from the list
//         An element in array cannot be deleted but the position
//         can be reset.  Let id and pssw be reset to initial values.
//Note:    Keep the list so all elements with data is moved to the
//         left (beginning from 0 to numOfelements-1
//Input: index: position to reset (delete)
//Return:  true if success, false otherwise
public boolean deleteElementAt(int index)
{
    boolean success = true;
    //Check so index is not out of range, if so success = false

   //ToDo:  Komplettera enlig kommentarerna ovanför metoddefinitioen
   //Decrement num of elements
   //Justera arrayen så alla sparade element ligger t.v. i arrayen från
   //position 0 t.o.m. numOfElements-1.Till höger om denna position ska
   //vara tomma element.

    return success;  //true om allt går bra.
}
```

Testa programmet – modifier metoden main:

- Skapa ett objekt av LoginHandler i Main
- Lägg till två element, skriv ut antalet element i arrayen.
- Ta bort ett element från array.
- Testa att ta bort ett element med ett fel index (till exempel 20)

En del av kod i kan användas i metoden main ges här nedan.

```
public static void main(String[] args)
{
    int maxElements = 5;  //test with asmall number
    LoginHandler loginList = new LoginHandler(maxElements);

    Login login = new Login("homer@simpson.se", "homer2022");
    //add an objekt
    boolean ok = loginList.addNew(login);
    if (ok)
        System.out.println(String.format("Object added. Num of elements: %d",
                                         loginList.getNumOfElements()));
    else
        System.out.println("Object could not be added.");

    //add another object
    //fortsätt
}
```

En testkörning ger följande resultat:

```
Object added. Num of elements: 1
Object added. Num of elements: 2
Object could not be deleted.
Object deleted. Num of elements: 1
```

## Uppgift 2a

Skapa ett nytt paket i projektet, ge paketet namnet **Uppgift2**. Utgå från din lösning till L14 Uppgift 2 (uppgift 2a). Klassdiagrammet visar tre klasser som samarbetar för att hantera en kontaktbok.

Din uppgift i denna del blir att du skall skriva klasserna Person och Telefon och därefter testa så att de fungerar bra ihop. Vi inför följande modifikationer och förutsättningar:

- Diagrammet visar att en person har ett antal telefonnummer. Som en förenkling, anta att en person har ett telefonnummer, alltså en instans av klassen Telefon.
- Telefonnumret är antingen ett privat- eller ett arbetsnummer. Använd följande enum för att lagra denna information i klassen Telefon.

```
public enum TelefonTyp
{
    privat,
    arbete
}
```

Enum definitionen kan sparas i egen fil eftersom typen är deklarerad som public.

- Deklarerar instansvariablerna i båda klasser som **private**.
- Skriv getter och setter metoder till alla instansvariabler i båda klasserna.
- Skriv en konstruktor med parametrar för att initiering av instansvariablerna i båda klasser.
- För klassen Telefon kan koden nedan användas:

```
public class Telefon
{
    private String nummer;
    private TelefonTyp telefonTyp;

    public Telefon(String nummer, TelefonTyp telTyp)
    {
        this.nummer = nummer;
        this.telefonTyp = telTyp;
    }
```

I koden ovan ser du användning av nyckelordet ”this” i konstruktorn för Telefon, väldigt kort refererar nyckelordet till den aktuella instansen av klassen. Mer förklaring av detta nyckelord kommer i föreläsning 15.

- Skriv en toString metod i båda klasser. Koden nedan är ett exempel för klassen:

```
public String toString()
{
    String textOut = String.format("%s (%s)",nummer, telefonTyp.name());
    return textOut;
}
```

**Testa programmet**

- Skriv en klass med metoden main som startar applikationen. Du kan kopiera följande kod (om en Main klass redan skapats, skriv endast in de kodraderna inne i main-metoden).

```
public class Main
{
   public static void main(String[] args)
   {
       Telefon telefon = new Telefon("+46 70 71234", TelefonTyp.arbete);
       System.out.println(telefon);

       Person person = new Person("Homer Simpson", telefon);
       System.out.println(person);
   }
}
```

Kör och testa så du får följande utskrift:

```
+46 70 71234 (arbete)
Homer Simpson har numret +46 70 71234 (arbete)
```

## Uppgift 2b

Nu ska du bygga vidare på ditt program för att hantera en lista av personer.

- Först, skall du skapa en ny konstruktor i klassen Person.

```
public Person (String namn, String nummer, TelefonTyp telTyp)
{
    //ToDo: Skriv här om vad som händer i satsen nedan?
    Telefon nyTelefon = new Telefon(nummer, telTyp);

    this.namn = namn;
    this.telefon = nyTelefon;
}
```

- Skapa en ny klass Kontaktbok och spara den.

- Denna klass skall innehålla en privat array av typen Person (tänk på när du namnger arrayen att det kan vara många Person objekt/instanser i arrayen). Skapa arrayen med nyckelordet new i konstruktorn.

- Skriv ingen annan instansvariabel och ingen setter- och getter metod till arrayen **personer**.

- Skriv en metod som lägger till en Person i arrayen.

- Skriv en metod som returnerar en array av String där varje element representerar en person (texten som toString-returnerar i Person-klassen):

```
public String[] getInfoStrings()
{
    String[] infoString = new String[numOfElements];

    //ToDo - komplettera

    return infoString;
}
```

- Skriv följande metod i klassen Kontaktbok för att vi ska använda som input (testdata) för att testa applikationen.

```
public void addTestValues()
{
    addNew(new Person("Namn1", "telnr 1", TelefonTyp.privat));
    addNew(new Person("Namn2", "telnr 2", TelefonTyp.arbete));
    addNew(new Person("Namn3", "telnr 3", TelefonTyp.privat));
    addNew(new Person("Namn4", "telnr 4", TelefonTyp.arbete));
}
```

Testa programmet med följande main-metod.

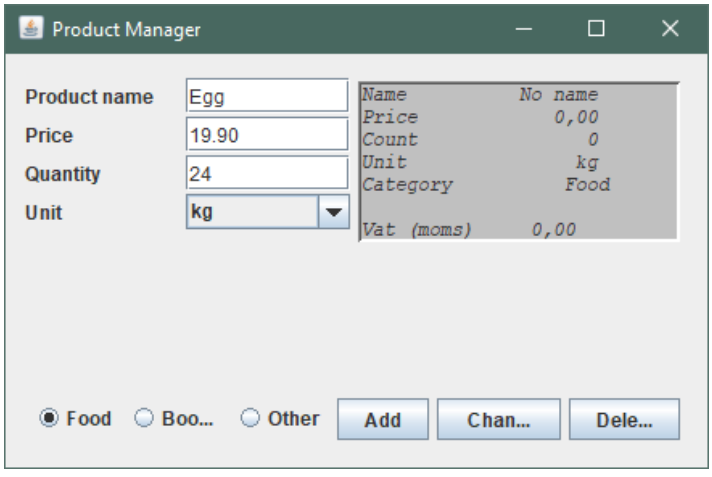
```
public class Main
{
    public static void main(String[] args)
    {
        Kontaktbok kontakt = new Kontaktbok(5);
        kontakt.addTestValues();

        String [] infoStrings = kontakt.getInfoStrings();
        if (infoStrings.length > 0){
            for (String str : infoStrings){
                System.out.println(str);
            }
        }
    }
}
```

En testkörning enligt ovan ger följande utskrift:

```
Namn1 har numret telnr 1 (privat)
Namn2 har numret telnr 2 (arbete)
Namn3 har numret telnr 3 (privat)
Namn4 har numret telnr 4 (arbete)
```

## Uppgift 3

Ladda ner *ProductManager.zip* från Canvas om du inte redan gjort det. Programmet saknar en del kod för att allting ska fungera bra och resultatet blir ett GUI-baserat applikation som i figuren nedan:




I version som du laddar ner från modulen finns det en ny enum-typ, ProductType. Dessutom har det tillkommit en ny variabel i klassen Product, som visas som Quantity (private int count i koden) i användargräns-snittet. GUI-delen är färdigt för dessa ändringar men inte resten av applikationen.

Ditt uppdrag är att studera klassen Controller samt att söka i andra filer där ändringar behövs för att få dessa två nya attribut att fungera. Den bästa vägen är att studera hur det görs med name och price.

## Lösningsförslag för Uppgift 1:

I labbinstruktionerna tidigare i labben står det att man skulle döpa paketet till Uppgift1 med stor bokstav, i bilden nedan råkade det bli med liten bokstav.

```
package uppgift1;

public class Login
{
    private String id;
    private String password;

    //ToDo 1:  Skriv en konstruktor med 2 parameterar
    //       för initiering av id och password

    //ToDo 2:  Skriv en toString

    //reset =  nollställa objektet (eller initiera objektet)
    public void reset()
    {
        id = "";
        password = "";
    }

    //<editor-fold desc=Lösningen">
    public Login(String id, String password)
    {
        this.id = id;
        this.password = password;
    }
    public String toString()
    {
        String textOut = String.format("ID: %s, Password: %s", id, password);
        return textOut;
    }
    //</editor-fold>
}
```

```java
    //Purpose: to delete an existing element from the list
    //         An element in array cannot be deleted but the position
    //         can be resetted.  Let id and pssw be resetted to initial values.
    //Note:    Keep the list so all elements with data is moved to the
    //         left (beginning from 0 to numOfelements-1
    //Input: index: position to reset (delete)
    //Return:  true if success, false otherwise
    public boolean deleteElementAt(int index)
    {
        boolean success = true;

        if  ( (index >= 0) && (index < loginList.length) )
        {
            loginList[index].reset();
            numOfElements--;  //reduce the number of saved elements
            moveElementsOneStepToLeft(index);
        }
        else
            success = false;

        return success;
    }
    private void moveElementsOneStepToLeft(int index)
    {
        for (int i = index+1; i < loginList.length; i++)
        {
            loginList[i-1] = loginList[i];  //move 1 step to left
            loginList[i].reset();
        }
    }

    //getter to return numofElements
    public int getNumOfElements(){
        return numOfElements;
    }
}
```