Laboration 2, ISG B18

Din uppgift är att skapa ett system för betalningsregistrering vid en mässa/konsert/event (Jag är den förste att erkänna att bakgrundsstoryn inte håller helt ut).

Du skall alltså skapa ett program där du registrerar betalningarna vid entrén. För en vanlig betalande kostar det 100 kr, för en student (som är volontär) betalas det istället ut 50 kr. För ett företag skickas det en faktura på betalningen och inträdet registreras som 0 kr och ett löpnummer skall genereras. Alla transaktioner (betalningar och utbetalningar) skall lagras i en vektor eller liknande.

För att en student skall kunna komma in (hen skall ju få 50 kr) måste det finnas pengar i kassan. Det betyder att det måste ha kommit minst en betalande kund innan man kan betala en student. Det kan alltså inte vara minus i kassan. Det får maximalt komma 50 personer (totalt av alla kategorier).

Din uppgift är att skapa nödvändiga klasser (se nedan vilka) samt skapa en main-metod som visar hur det skulle kunna fungera. Laborationen är en övning i relationer mellan objekt och du kommer att bygga ut den i nästa laboration. Du skall ha dessa klasser och metoder, inga andra.

Klasser

Följande klasser med specificerat innehåll skall implementeras i ditt program:

Kommentar till pre på de flesta (många) metoderna: De kräver att konstruktorn har körts. Men då metoderna inte kan nås utan att konstruktorn körts blir förvillkoret *true*.

Register

Attribut

```
private LinkedList lista // eller annan typ av lista
private int lopnummer
```

Metoder

```
//pre: -
//Post: Lista (tex typen LinkedList, finns i java.util) skapad. Löpnummer initierat
public Register()

//Pre: -
//Post: Ny balans beräkna och returnera genom att iterera igenom den länkade listan.
public int beraknaSaldo()

//Pre: Kund skapad
//Post: Kund tillagd i listan/registret
public void regKund(kund)

//Pre:-
//Post: Löpnumret ökat och returnerat
public int getLopnummer()
```

//Pre:-
//Post: En sträng har byggts upp genom att iterera igenom den länkade listan och lägga till
//varje transaktion till strängen. Denna sträng returneras.
public String getLista()

//Pre: -
//Post: Returnerar antalet poster i listan
public int listStorlek()

Kund

Attribut

private int belopp
private String datum

Metoder

//Pre: -
//Post: Attributen initierade med lämpliga värden
public Kund()

//Pre: -
//Post: Belopp returnerat
public int getBelopp()

//Pre: Inparametern skall ha ett lämpligt värde
//Post: Belopp har uppdaterats
public void setBelopp(int)

//Pre: -
//Post: Datum returnerat
public String getDatum()

//Pre: Inparametern skall ha ett lämpligt värde
//Post: Datum har uppdaterats
public void setDatum(String)

//Pre: Inparametern skall ha ett lämpligt värde
//Post: Returnerar inparametern + belopp.
Denna metod skall överlagras i Student.
public int nyBalans(int)

//Pre: -
//Post: -
Denna metod skall överlagras i Foretag.
public void setLopnummer(int)

Student

Attribut

-

Metoder

//Pre: -
//Post: Belopp och datum för transaktion returnerat
public String toString()

//Pre: Inparametern skall ha ett lämpligt värde
//Post: Returnerar inparametern - belopp.
public int nyBalans(int)

Normal

Attribut

-

Metoder

//Pre: -
//Post: Belopp och datum för transaktion returnerat
public String toString()

Foretag

Attribut

private int lopnummer

Metoder

//Pre: -
//Post: Attributen initierade med lämpliga värden
public Foretag()

//Pre: -
//Post: Belopp, datum och löpnummer för transaktion returnerat
public String toString()

//Pre: Inparametern skall ha ett lämpligt värde
//Post: Löpnummret returnerat
public int getLopnummer()

//Pre: Inparametern skall ha ett lämpligt värde
//Post: Löpnummer (attrubutet) tilldelat värde
public void setLopnummer(int)

Tips

Börja med att fundera ut vilka relationer de olika klasserna skall ha.
Du kan också, under utvecklingen, lägga in en main-metod i en klass för att testa så att den fungerar, självständigt, som du tänkt dig.
Undvik specialtecken (å, ä och ö) som namn på klasser, metoder, attribut och variabler. Det fungerar med dessa tecken i Java men det blir problem när man byter mellan olika plattformar.