Inlämningsuppgifter omgång 4

Allmänna instruktioner: Det finns två delar i detta dokument: Del 1 som innehåller uppgifter som måste göras för att få godkänt, och Del 2 som innehåller uppgifter för att få bonuspoäng till tentan för högre betyg. För att bli godkänd på inlämningsuppgifterna räcker det således att göra Del 1.

Döp era Javaprogram till de filnamn som specificeras i uppgifterna.

Spara alla java-filer i en katalog ”<användarnamn>_in4”, t.ex. johthu19_in4. Komprimera denna katalog till en zip-fil och skicka in filen.

Deadline för att lämna in uppgifterna på Blackboard är söndag 29/11 klockan 23:59:59.

Uppgifterna presenteras på examinationstillfället för din grupp 30/11-1/12.

Del 1 – uppgifter för godkänt

Dessa inlämningsuppgifter behandlar: Funktioner, statiska metoder. Kap 2.1-2.2.

Uppgift 1: Max.java

Skriv ett program som innehåller två funktioner som du döper till max2 och max3. Den första funktionen, max2, returnerar det största talet av två angivna tal. Den andra funktionen, max3, returnerar det största talet av tre angivna tal. Programmet ska vara utformat så att max2 används inuti funktionen max3.

I de två uppgifterna nedan ska ni utveckla verktyg för att hitta bedragare. Ni arbetar på ett företag och ska verifiera om data om kunderna är felaktiga. I Uppgift 2 ska ni utveckla ett verktyg för att se om kundens personnummer är felaktigt. I Uppgift 3 ska ni utveckla ett verktyg för att se om kundens kontonummer är felaktigt.

Uppgift 2: Personnummer.java

Skriv ett program som frågar användaren efter dennes personnummer. Personnumret som matas in till programmet ska vara på formatet ÅÅÅÅMMDD-XXXX. Detta personnummer ska sen kontrolleras och om det är korrekt så ska ålder samt födelsedag på personen skrivas ut på skärmen. Mer precist:

a) Skriv en funktion som avgör om personnumret är på rätt format. Dvs. Det är 13 tecken långt, först 8 siffror följt av bindestreck och sedan 4 siffror till.

   Extra (ej obligatoriskt): se till att de åtta första siffrorna representerar ett giltigt datum.

b) Skriv en funktion som först anropar funktionen i a). Om formatet är korrekt så undersöks om personnumret är giltigt. Detta göres genom att multiplicera första siffran i det **10-siffiga** personnumret (inte tolv siffror som innan) med 2, den andra siffran med 1, den tredje siffran med 2, den fjärde med 1 osv. Du får då tio nya siffror. Du summerar siffersummorna av dessa siffror och

får ett tal. Om detta tal är jämnt delbart med 10 så är personnumret giltigt. Se exempel nedan:

```
  8 1 1 2 1 8  9 8  7 6
* 2 1 2 1 2 1  2 1  2 1
----------------------
 16 1 2 2 2 8 18 8 14 6

 1+6+1+2+2+2+8+1+8+8+1+4+6 = 50
```

Tips: En bra metod i detta sammanhang för att undersöka en delsträng av en sträng är metoden substring. Första argumentet till metoden anger vilket index delsträngen ska börja vid och andra argumentet anger vilket index den ska sluta innan. T.ex

"hej på dig".substring(2,8) är ”j på d”.

Följande metoder är också användbara. Om vi har en Sträng string så kan string.toCharArray()användas för att omvandla string till char array; string.charAt(i) returnerar den char som ”är” det i:te elementet i string. String.valueOf(c) omvandlar värdet för en char till String. Character.getNumericValue(c) omvandlar c till ett numeriskt värde.

Uppgift 3: CheckSum.java

En så kallad cheksummaformel används ofta av banker och kreditkortsföretag för att undersöka om kontonummer är korrekt angivna eller tillåtna. Vi kallar första siffran i kontonumret för d0, nästa siffra för d1, tredje sifran för d2 osv. Cheksummametoden går ut på att man först räknar ut

d0 + f(d1) + d2 + f(d3) + d4 + ... = N,

sedan om N%10 = 0 (dvs resten är 0 när man dividerar med 10) så är bankkontot tillåtet (korrekt). f är en funktion som tar ett heltal som argument, multiplicerar detta med 2 och returnerar siffersumman för det nya talet. T.ex. f(7) = 5, eftersom 2*7 = 14 och 1+4 = 5.

Uppgiften består i att:

a) skapa funktionen *f* ovan,

b) skapa funktionen *checksum* som räknar ut cheksumman N för ett bankkontonummer enligt formeln ovan. I denna funktion så används funktionen f.

c) Skapa en funktion *allowed* som undersöker om cheksumman är tillåten (retunerar true eller false).

d) Skapa funktionen *extend_account* som för ett angivet kontonummer med 10 siffror lägger till en siffra i slutet så att checksumman för det nya, 11 siffror långa kontonumret, är tillåtet. I denna funktion så används funktionerna *checksum* och *allowed*.

Programmet ska fråga efter ett tiosiffrigt bankkontonummer och skriva ut ett elvasiffrigt tillåtet bankontonummer på skärmen.

Tips: Se tipps för Uppgift 2.

Del 2 – uppgifter för bonuspoäng

Uppgift 1: PokerAnalysis.java

I denna uppgift ska du konstruera ett verktyg för (meta)pokeranalys. Poker är ett samlingsnamn för ett stort antal kortspel som har ett gemensamt element av vadslagning. Den vanliga kortleken används med fyra olika färger och tretton olika värden för varje färg.

Ett nätkasino har hört av sig som är intresserat av att lansera pokerspel där man har fler kort på handen, t.ex. 10 kort. Din uppgift är att ge denna marknadsaktör ett statistiskt underlag.

Mer specifikt så ska du skriva ett program där användaren anger två heltal. Det första representerar antalet spelare och det andra representerar antalet kort per spelare. Om de 52 korten i en kortlek inte räcker till för att ge varje spelare de antal kort denne behöver, så ska ett felmeddelande skrivas ut på skärmen och programmet ska avslutas. Nedan följer två deluppgifter som beskriver vad programmet ska göra efter att användaren har angett antal spelare och antal kort per spelare (om korten räcker till).

a) (2p) Programmet ska ta reda på, och skriva ut på skärmen, de uppskattade sannolikheterna för att någon av spelarna runt bordet har:

- ett par (dvs. två kort med samma värde, t.ex. två stycken tvåor eller två stycken kungar),
- triss (dvs. tre kort med samma värde),
- tvåpar (dvs. två stycken par)
- kåk (dvs. ett par och en triss där värdet på korten i paret inte är samma värde för korten i triss).

De övriga typerna av pokerhänder får självklart undersökas också men det är inget krav. Om ni vill kan ni också introducera nya typer av händer när antalet kort som delas ut till varje spelare är större än 5. Vad sägs om trepar?

b) (2p) Programmet ska uppskatta hur ofta en spelare vinner med de respektive händerna som gås igenom i a) Dvs. hur ofta vinner man om man har ett par, ett tvåpar, en triss, och en kåk (eller de andra händerna som ni möjligen har introducerat). Rangordningen av de existerande pokerhänderna återfinns här.