## Häfte med extra övningsuppgifter

**Uppgift 1:** Betrakta fyra koner med volym $V$, basyta $A$ och densitet $\rho$ enligt nedanstående tabell.

| | Kon 1 | Kon 2 | Kon 3 | Kon 4 |
|---|---|---|---|---|
| V [m$^3$] | 5.0 | 4.3 | 3.5 | 6.1 |
| A [m$^2$] | 1.7 | 1.9 | 1.5 | 2.2 |
| $\rho$ [kg/m$^3$] | 1000 | 2100 | 2400 | 1200 |

Skriv ett Matlab-program som

a) läser in volymerna $V$, areorna $A$ och densiteterna $\rho$, i tabellen från en textfil `kondata.txt` (som du skapat själv).

b) mha en funktionsfil `kon.m` beräknar de olika konernas höjd $h = 3V/A$ och massa $M = V\rho$

c) till sist ritar ut två stapeldiagram; ett för de olika höjderna [m] och ett för massorna [kg]. Tänk på att namnge axlarna och ange tillhörande enhet.

**Uppgift 2:** En väldigt enkel modell för fondkursen $K(t)$ som funktion av dimensionslös tid $t$ ges av följande samband

$$K(t) = e^{t/5} + 0.3\sin(2\pi t)$$

Om man köper fondandelar för den fasta summan $M$ kr per tidsenhet ges omkostnaden av sambandet

$$O(t) = Mt$$

Det sammanlagda värdet av fondandelarna blir då

$$V(t) = K(t)\int_0^t \frac{M}{K(\tau)}\mathrm{d}\tau$$

Din uppgift är att illustrera dessa samband i en figur. Skapa ett figurfönster med två delfigurer placerade ovanför varandra och rita i det övre delfönstret funktionen $K(t)$ för intervallet $0 \le t \le 2$. I det undre fönstret skall både omkostnaden $O(t)$ och värdet $V(t)$ då $M = 5$ kr ritas in för samma tidsintervall.

**Uppgift 3:** Skriv en egen funktionsfil `B = matrissort(A)` som sorterar elementen i matrisen `A`. Matrisen `A` kan vara godtyckligt stor och matrisen `B` har samma storlek som `A`, men innehåller de sorterade elementen i stigande ordning radvis där element (1,1) är det minsta elementet och det sista elementet $(m,n)$ är det största elementet. För att testa funktionsfilen skall ett huvudprogram skrivas. I huvudprogram skapas en $4{\times}7$ matris med uniformt slumpmässigt fördelade element mellan -10 och 50. Utnyttja Matlabs inbyggda funktion `rand` för att generera matrisen `A` som indata till funktionsfilen.

**Uppgift 4:** Skriv en egen funktionsfil `y = plockaut(x)` som ur vektorn `x` plockar bort alla tal som är jämnt delbara med 3, 4 eller 5. Invariabel till funktionen är vektorn `x` och utvariabel är den nya vektorn `y` som innehåller alla utom de bortplockade talen. Funktionsfilen ska anropas från ett huvudprogram för vektorn `x` given som `x = [1:50]`.

**Uppgift 5:** Skriv en egen funktionsfil `y=downsort(x)` som sorterar elementen i en vektor `x` av godtycklig längd i avtagande ordning. (Det största talet kommer först sedan det näst största osv.) Indata till funktionen är vektorn `x` och utdata är vektorn `y` med de sorterade elementen i avtagande ordning. Du får inte använda Matlabs inbyggda funktion `sort`. För att testa funktionsfilen skall ett huvudprogram skrivas. I huvudprogram skapas en radvektor som innehåller 14 st godtyckliga heltal mellan -30 och 30. Utnyttja Matlabs inbyggda funktion `rand` för att generera matrisen `A` som indata till funktionsfilen.

**Uppgift 6:** Skriv ett program som läser in en textfil som en teckensträng. Därefter skall programmet söka igenom denna teckensträng och byta ut alla bokstäver `a` mot bokstaven `b`. Till sist skall resultatet skrivas till en ny textfil.

Skapa textfilen själv genom att skriva in några raders text i Matlabs editor och spara med filändelsen ”`.txt`”. Använd kommandot `fileread` för att läsa in filen. För att skriva den modifierade teckensträngen till en ny fil kan du använda kommandona `fopen`, `fprintf` och `fclose`.

**Uppgift 7:** En del av en berg-och-dalbana kan beskrivas med kurvan till den parameteriserade funktionen

$$\begin{cases} x(t) = (a-b)\sin(t) - c\sin\left(\left(\frac{a}{b}-1\right)t\right) \\ y(t) = -(a-b)\cos(t) - c\cos\left(\left(\frac{a}{b}-1\right)t\right) \end{cases}$$

a) Plotta funktionskurvan i intervallet $-1.2\pi \leq t \leq 1.2\pi$ för parametervärdena $a = 0.5, b = 2, c = 1.6$.

b) Gör en film som visar ett litet tågs resa genom banan. Gör detta på följande sätt: Låt $x(t)$ och $y(t)$ representeras av två vektorer `x` och `y`. Rita, utöver kurvan i a), fyra röda kvadrater på de koordinater som motsvaras av de första fyra elementen i `x` och `y`. Gör sedan om plottningen, men rita istället kvadraterna för element två till fem i `x` och `y` (de gamla kvadraterna skall inte vara kvar). Upprepa fram till slutet av vektorerna. Lägg in en paus (se `pause`) mellan varje plottning så att hela resan tar runt fem sekunder. OBS: Du behöver inte skapa något filmobjekt med ”`getframe`” eller liknande, det räcker att det ser ut som en film i figurfönstret.

**Uppgift 8:** Skriv en funktionsfil `Vut=mod10(Vin)` som givet vektorn `Vin` skapar vektorn `Vut` där sista talet i `Vut` beräknas enligt Modulo10-algoritmen (även kallad Luhns algoritm) för beräkning av kontrollsiffra. Kontrollera att du gjort rätt genom att som `Vin` ange de första nio siffrorna i ditt personnummer. Vut skall då bli hela ditt personnummer inklusive den sista siffran.

<u>Modulo10-algoritmen:</u>

- Sätt summa till noll
- För varje tal i Vin
  - Om talet är på udda plats
    - Multiplicera med 2
    - Talvärde=Summan av siffrorna (ex 2*7=14 -> 1+4=5)
  - Om talet på jämn plats
    - Talvärde = talet självt
  - Lägg talvärde till summa
- Tag resten vid heltalsdivisionen summa/10
- Tag 10 - resten
- Kontrollsiffran = resten vid heltalsdivision med 10 igen

**Uppgift 9:** Alven Puck och hans vänner tycker om att springa i skogen, speciellt under midsommarnatten. De springer olika långa sträckor, men för att se vem som springer snabbast mäter de även tiden det tar var varje alv att springa sin sträcka. Eftersom de lever i en Shakespearevärld använder de FFF-systemet (i motsats till vårt SI-system). Sträckan mäter de därför i millifurlong och tiden mäter de i mikrofortnight.

a) För att kunna räkna ut hur fort Puck och hans vänner springer, skriv en funktionsfil (`hastighet.m`) som givet en sträcka $s$ (i millifurlong) och en tid $t$ (i mikrofortnight) returnerar medelhastigheten $v$ i meter/sekund.

   1 mikrofortnight = 1.2096 sekunder och 1 millifurlong = 0.201168 m.

b) I filen `Lopresultat.txt` finns sträcka och tid för alverna. Skriv ett huvudprogram som läser in data och beräknar hastigheterna för alla alverna.

c) Utöka huvudprogrammet så att det skriver ut på skärmen de tre bästa resultaten i stigande ordning.

```
På tredje plats kom alv nr 123 med hastigheten 1.111 m/s
På andra plats kom alv nr 456 med hastigheten 2.222 m/s
På första plats kom alv nr 789 med hastigheten 3.333 m/s
```

d) Utöka huvudprogrammet så att det även skriver ut på skärmen vem som sprang närmast långsammare än 3 m/s

```
Alv nr 157 hade hastigheten närmast under 3 m/s (2.567 m/s)
```

**Uppgift 10:** En kastbana med luftmotstånd kan beskrivas av den parameteriserade funktionen

$$\begin{cases} x(t) = \frac{v_0 \cos(\alpha)}{2}\left(1 - e^{-kt}\right) \\ y(t) = \left(\frac{v_0 \sin(\alpha)}{2} + \frac{g}{k^2}\right)\left(1 - e^{-kt}\right) - \frac{gt}{k} \end{cases}$$

a) Plotta funktionenskurvan i intervallet $0 \le t \le 3$ för parametervärdena $v_0 = 30$ m/s, $k = 0.5$ s$^{-1}$, $g$=9.81 m/s$^2$, $\alpha = 45°$.

b) Gör en film som visar en kanonkulas resa genom banan. Gör detta på följande sätt: Låt $x(t)$ och $y(t)$ representeras av två vektorer `x` och `y`. Rita, utöver kurvan i a), en röd cirkel på den koordinat som motsvaras av det första elementent i `x` och `y`. Gör sedan om plottningen, men rita istället cirkeln för element två i `x` och `y`. Upprepa fram till slutet av vektorerna. Lägg in en paus (se `pause`) mellan varje plottning så att hela kastet tar runt fem sekunder. Du behöver inte skapa något filmobjekt med `getframe` eller liknande, det räcker att det ser ut som en film i figurfönstret.

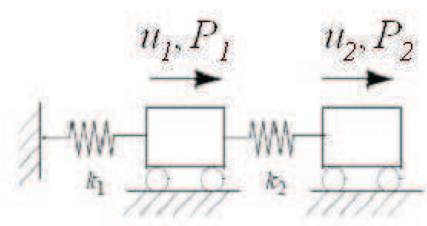
c) Lägg till en ”krasch” i slutet genom att rita in successivt större cirklar. Sätt storleken till 6+3$j$ där $j$ går från 1 till 5.

**Uppgift 11:** En administratör skall välja ut tre av tjugo studenter till att vara kursrepresentanter. Skriv ett program som läser in namn från filen `NamnLista.txt` och skriver ut tre slumpmässigt utvalda namn på skärmen.

**Uppgift 12:** Skriv en funktionsfil som givet matriserna `A` och `B` som invariabler returnerar matrisen `C` som `C=A*B` (matrismultiplikation). Matrismultiplikationen skall i denna uppgift utföras med `for`-loopar. Om dimensionerna på `A` och `B` inte stämmer överens skall programmet ge ett felmeddelande.

**Uppgift 13:** Skriv en funktionsfil som givet matrisen `A` och vektorn `b` returnerar matrisen `x` som lösningen till det linjära systemet `A*x=b`. Gausselimineringen skall utföras med `for`-loopar (ej \). Om dimensionerna på `A` och `b` inte stämmer överens skall programmet ge ett felmeddelande.

**Uppgift 14:** Figur 1 nedan visar en struktur med två frihetsgrader $u_1$ och $u_2$. Vi kan beräkna förskjutningarna $u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$ givet lasten $P = \begin{bmatrix} P_1 \\ P_2 \end{bmatrix}$ (som inte varierar i tiden) med hjälp av sambandet $Ku = P$. Styvhetsmatrisen $K$ fås som $K = \begin{bmatrix} k_1 + k_2 & -k_2 \\ -k_2 & k_2 \end{bmatrix}$




Figur 1: System med två frihetsgrader

a) Skriv en funktionsfil `ekvLosning.m` som givet en godtycklig styvhetsmatris $K$ och en lastvektor $P$ returnerar förskjutningarna $u$.

b) I filen `styvhet.txt` finns det ett antal kombinationer av styvheter $k_1$ och $k_2$. Skriv ett huvudprogram som läser in dessa värden och för varje par av $k_1$ och $k_2$ (rad i `styvhet.txt`) beräknar förskjutningarna $u$. Lasten $P$, som är samma hela tiden, består av en kraft på 3N riktad åt vänster på den högra vagnen och ingen kraft på den vänstra vagnen.

c) Presentera de värden som ligger över gränsvärdet 0.6 m i fallande ordning efter absolutbeloppet på den högra vagnens förskjutningar. Skriv ut resultatet på följande vis:

```
Styvhetspar nr 123 gav förskjutningen 333 m
Styvhetspar nr 456 gav förskjutningen 222 m
Styvhetspar nr 789 gav förskjutningen 111 m
```

**Uppgift 15:** I likhet med kontrollsiffran för svenska personnummer finns det en liknande metod för att ta skapa ISBN-boknummer som är en unik 10-siffrig kod för varje bokutgivning. Skriv en funktionsfil `StrUt=ISBN_siffra(StrIn)` som givet teckensträngen `StrIn` skapar teckensträngen `StrUt` där sista tecknet i `StrUt` beräknas enligt ISBN-algoritmen för beräkning av kontrollsiffra. Kontrollera att du gjort rätt genom att som `StrIn` ange de första nio siffrorna i någon av dina kursböcker. `StrUt` skall då bli hela ISBN-numret inklusive det sista tecknet. Notera att kontrollsiffran i detta fall kan bli 10, och om så är fallet skriver man 'X' som sista tecknet i ISBN.

<u>ISBN:s-kontrollsiffra:</u> Multiplicera talen i StrIn med respektive 10, 9, 8,...2

Summer alla talen

Tag resten av summan heltalsdividerat med 11 (`mod`)

Kontrollsiffra=11-resten efter heltalsdivisionen

Skapa `StrUt` genom att lägga kontrollsiffra till `StrIn`

Not: Från 2007 har man infört om ett system med 13 siffror i ISBN-numret.

**Uppgift 16:** Ett sågverk har installerat ett automatiskt system för att mäta storleken på de stockar man får in. Systemet mäter längden på stocken samt topp- och rotdiameter. Mätningarna finns lagrade i filen `GranData.txt`. Mätsystemet är lite instabilt, varpå en del mätningar kan falla bort. Missade mätningar representeras av värdet 0 i `GranData.txt`.

a) Skriva ett program som läser in data från filen GranData.txt. Därefter skall programmet beräkna volymen för varje stock enligt

$$V = \frac{L\pi}{12}\left(d_{\text{topp}}^2 + d_{\text{topp}}d_{\text{rot}} + d_{\text{rot}}^2\right)$$

De stockar med någon missad mätning skall inte tas med i beräkningarna.

b) Utöka programmet genom så att det även beräknar medelvärde och standardavvikelse för volymerna. Resultatet skall skrivas ut på skärmen. Återigen skall de stockar med någon missad mätning inte tas med i beräkningarna.

c) Rita ett stapeldiagram över volymerna. Missade värden skall representeras av staplar med höjden noll (sista stapeln skall alltså ha samma ”x-koordinat” som antalet rader i `GranData.txt`). Rita även in tre horisontella linjer i diagrammet för medelvärdet samt medelvärde plus/minus en standardavvikelse.

**Uppgift 17:** En bil är utrustad med farthållare. Bilen färdas med hastigheten $y$ och som beror på trottelvinkeln på förgasaren $u$ (trottelvinkeln bestäms av gaspedalen eller av farthållaren). Genom att ställa in en önskad hastighet $y_0$ kan farthållaren bestämma $u$ så att bilens hastighet kommer så nära den önskade hastigheten $y_0$ som möjligt. Med hjälp av en s.k. PI-regulator bestäms $u$ utifrån hur mycket $y$ skiljer sig från $y_0$. En bil som startar från stillastående med farthållaren inställd på $y_0$ får följande hastighet som funktion av tiden (här har vi gjort flera förenklande antaganden) enligt

$$\begin{cases} y(t) = y_0\left(1 - \frac{e^{-\omega\zeta t}}{\sqrt{1-\zeta^2}}\sin(\omega t + \varphi)\right) \\ u(t) = \left(K_p y_0 + \frac{t^2}{2T_i}\right)\frac{e^{-\omega\zeta t}}{\sqrt{1-\zeta^2}}\sin(\omega t + \varphi) \end{cases}$$

med parametervärdena $y_0$=25 m/s, $K_p$=0.2, $T_i$=5 s, $\zeta = 0.5/\sqrt{10K_p}$, $\omega = \sqrt{10K_p}$ och $\varphi = \sin^{-1}\left(1-\zeta^2\right)$.

a) Rita funktionen $y(t)$ och $u(t)$ i varsitt delfönster placerade ovanför varandra för tidsintervallet $0 \le t \le 20$ s. Förse axlar och titlar med lämpliga beteckningar.

b) Farthållarsystemet har två begränsningar. Dels får inte bilens hastighet $y$ överskrida den önskade hastigheten $y_0$ med mer än 10 % och dels får inte absolutvärdet av styrsignalen $u$ överskrida värdet $u_{\max} = 6$. Utvidga programmet från a) enligt följande: Rita in dessa begränsningar som horisontella streckade linjer i respektive delfönster. Om någon del av $y$ eller $u$ överskrider sitt respektive gränsvärde skall hela denna funktion ritas med röd linje och linjetjocklek 2, men om villkoren är uppfyllda skall linjen vara grön med linjetjocklek 1. Alla parametrar ($y_0$, $u_{\max}$, etc.) skall anges i början av programmet så att de är lätta att hitta och ändra.

**Uppgift 18:** Tack vare att SI-systemet har decimalenheter är det mycket enkelt att uttrycka till exempel 243 centimeter som 2 meter och 43 centimeter. Tyvärr är inte våra tidsenheter decimala då 243 sekunder blir 4 minuter och 3 sekunder. Din uppgift är att skriva ett program som översätter ett (stort) antal sekunder i antal veckor, dagar, timmar, minuter och återstående sekunder.

Själva omvandlingen skall göras i en funktionsfil, `vdtms=tidsomvandlare(s)`, som givet antal sekunder `s` beräknar vektorn `vdtms` på formen `[veckor, dagar, timmar, minuter, sekunder]` där det första elementet anger antal veckor, andra elementet antal dagar osv. 243 sekunder motsvaras då av vektorn `[0 0 0 4 3]`.

Skriv också ett huvudprogram som ber användaren knappa in ett antal sekunder och efter anrop av `tidsomvandlare` skriver ut formaterad text på skärmen:

```
Ange antal sekunder: 243

243 sekunder motsvarar 0 veckor 0 dagar 0 timmar 4 minuter 3 sekunder.
```

Obs: Det är inte tillåtet att använda någon av Matlabs inbyggda funktioner för datum såsom `datenum`, `datevec` eller liknande.

**Uppgift 19:** År 2008 inföll påsksöndagen den 23:e mars, vilket är ovanligt tidigt, det enda datum tidigare som påsken kan infalla är den 22:a mars. Vid en närmare granskning visar det sig att påskdagens olika datum inte är lika vanligt förekommande, vilket vi skall titta närmare på i denna uppgift.

I filen `Datum_ored.txt` finns en tabell över påsksöndagens datum för åren 1700-2299 som får tjäna som underlag (datumen är hämtade från http://www.assa.org.au/edm.html och skiljer sig i några enstaka fall från svenska förhållanden). För att förenkla hanteringen finns en motsvarande fil `Datum.txt` som innehåller samma information men är lättare att läsa in i Matlab.

Din uppgift är att skriva ett program som läser in innehållet i filen `Datum.txt` och skapar ett histogram över de olika datumen, dvs. ett stapeldiagram där stapelns höjd anger antalet gånger påsken infaller ett specifikt datum. Programmet skall dessutom ta reda på när påsken infaller den 23:e mars nästa gång (dvs. vilket år).

Ledning: Utnyttja hist för att rita diagrammet. Det är tillåtet att utgå från att det finns precis 35 möjliga datum (22 mars - 25 april).

**Uppgift 20:** På nätet finns massor av relationstestare som mer eller mindre intrikat räknar ut en sannolikhet att två personer passar ihop. Detta är naturligtvis rent nonsens, men roligt ändå. Vi skall testa några olika algoritmer som för givet två textsträngar räknar ut ett tal mellan 0 och 100 som skrivs ut på skärmen, vilket är sannolikheten i % att de två personerna passar ihop.

Skriv en funktionsfil som har utvariabeln `prob` och invariabler (namn1, namn2, alg) där `namn1` och `namn2` är teckensträngar för namnen (små bokstäver) och `alg` är ett tal som via if eller switch-satser väljer vilken algoritm som skall användas. Programmet skall skriva ut en text på skärmen som anger vad sannolikheten mellan ”`namn1`” och ”`namn2`” blev, uttryckt i procent.

a) Algoritm 1 Tag summan av ASCII-värdena för de båda textsträngarna tillsammans. Sökta talet är resten vid heltalsdivision med 100.

b) Algoritm 2 Tag summan av ASCII-värdena för de båda teckensträngarna var för sig. Beräkna kvoten minsta värdet delat med största värdet. Sökta talet är kvoten multiplicerat med hundra avrundat till heltal.

c) Algoritm 3 Hitta på en egen algoritm. Helst en som gynnar dig själv t.ex. genom att ditt plus någon speciell persons namn får ett högt värde (ej 100 - det verkar misstänkt, runt 95 är bättre!).

Ledning: ASCII-värdena för bokstäver fås med `double` enligt

```
>>a='banan'
>>double(a)
ans =
98    97   110    97   110
```

Funktionsfilen anropas enligt

```
>>Prob=RelationTester('anders','anna',1);
```

Som enligt Algoritm 1 ger sannolikheten 51%.

**Uppgift 21:** Skriv ett program som löser motorvägsgåtan, enligt notisen i NyTeknik på nästa sida. Läs notisen noga innan du fortsätter med denna uppgift. Decimalutvecklingen för $e$ kan man hitta på adressen http://antwrp.gsfc.nasa.gov/htmltest/gifcity/e.2mil. Tillräckligt många siffror har kopierats över till en textfil med namnet `Edecimal.txt`.

Uppgiften är nu att skriva ett program som utifrån filen `Edecimal.txt` skriver ut på skärmen det första 10-siffriga primtalet och dess position i talutvecklingen samt de fem första 10-siffrorsserier med radsumman 49 med respektive positioner i talserien.

Programmet skall alltså skriva ut

```
First prime found at position 99: 7427466391

Sum 49 found at position 1: 7182818284

Sum 49 found at position 5: 8182845904

Sum 49 found at position 23: 8747135266

Sum 49 found at position 99: 7427466391

Sum 49 found at position 127:  XXXXXXXXXX
```

Ledning: Dela upp problemet i mindre delproblem:

1) Läs in filen `Edecimal.txt` som en lång vektor där varje element utgörs av en siffra i decimalutvecklingen. Utnyttja t.ex. `fscanf` med formatet `'%1d'`.
2) Stega upp i vektorn till första primtalet hittas.
3) Börja om från början och stega upp i vektorn tills femte gången radsumman 49 hittas.

Det visar sig att vi måste skriva en egen funktion som testar om ett tal är ett primtal eftersom matlabs inbyggda `isprime` är begränsad till $23^2$ $\approx$4.2e9.

NyTeknik nr 43 2004 (20/10)

20 OKTOBER 2004 NR 4

EFTER JOBBET

# Motorvägsgåta var platsannons

**Amerikanska bilister har under hösten hajat till när de passerat en stor skylt vid sidan av motorväg 101 söder om San Francisco. Den såg ut som en reklamskylt, men för vad? Så här löd texten:**

{First 10 digit prime in consecutive digits of e}.com

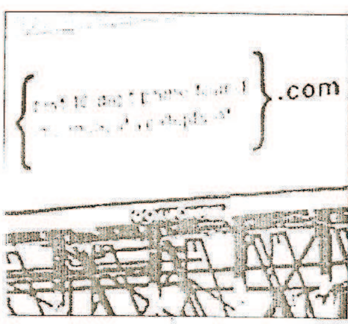
**En likadan skylt dök sedan upp vid väg 128 norr om Boston, inte långt från MIT – Massachusetts Institute of Technology. Vad var nu detta?**

**DEN SKARPSINNIGE** insåg omedelbart att det var fråga om en webbadress, eftersom det hela slutade med ".com". Uppenbarligen gällde det att finna det första tiosiffriga primtalet i decimalutvecklingen av talet *e*.

*e* är en matematisk konstant som har en rad märkliga egenskaper, och det är också basen för de naturliga logaritmerna. Värdet är drygt 2,7.

Men *e* är ett irrationellt tal, och har alltså liksom π ett oändligt antal decimaler. Så här ser utvecklingen av *e* ut:

e=2.71828182845904523536028
747135266249775724709369995
957496696762772407663035354
759457138217852516642742746
639193200305992181741359662
904357290033429526059563073
813232862794349076323382988
075319525101901157383418793
070215408914993488416750...




Från tjallarens webbsajt

Den märkliga motorvägsskylten.

Det första tiosiffriga primtalet i denna serie siffror är 7427466391, och det finns i position 99 i utvecklingen. Den sökta webbadressen skulle alltså bli: http://www.7427466391.com

Den som surfade iväg till denna adress möttes av en ny gåta:

**"Congratulations! You've made it to level 2. Go to** www.Linux.org **and enter Bobsyouruncle as the login and the answer to this equation as the password.**

**f(1)= 7182818284**
**f(2)= 8182845904**
**f(3)= 8747135266**
**f(4)= 7427466391**
**f(5)= ___________"**

Jaha. Vad är nu detta? Vad har de här sifferföljderna gemensamt, och hur kommer vi fram till $f(5)$?

Jo, det visar sig att alla de fyra talen har tvärsumman 49. Det femte talet med denna egenskap i utvecklingen av *e* hittar man i position 127, och ni kan själva roa er med att finna det.

Vad händer då när vi loggar in på www.Linux.org med vårt lösenord?

I dag ingenting, för nu är sajten stängd. Men hade vi svarat tidigare hade vi fått veta att vi tillhörde världseliten bland ingenjörer och erbjudits jobb hos dataföretaget Google. Men det var innan en tjallare la ut lösningen på webben.

**Kaianders Sempler** 08-796 65 67
kaianders.sempler@nyteknik.se

TEKNIKFRÅGAN