

Tentamen i Programmering i R, 7.5 hp

Skrivtid: 8.00-12.00
Hjälpmedel: Inget tryckt material, dock finns "R reference card v.2" och några andra referenskort tillgängliga elektroniskt.
Betygsgränser: Tentamen omfattar totalt 20 poäng. 12 poäng ger Godkänt, 16 poäng ger Väl godkänt.
Tänk på följande:
Skriv dina lösningar i **fullständig och läsbar kod**. Kommentera din kod och använd en god kodstil.
Spara filen med namnet **tentaX.R** där **X** är ditt SC-nummer (klientnummer). Det numret kan du se i studentklienten. Exempel: om du har SC-nummer SC12345 så ska filen heta tentaSC12345.R
Filen lämnas sedan in via studentklienten. Notera att du ska lämna in **en** fil med alla dina lösningar.
När du har loggat ut från datorn är tentan avslutad. Frågor kan ställas till lärare via studentklienten.
Spara filer på skrivbordet: ~/Desktop/
Kommentera direkt i din R-fil när något behöver förklaras eller diskuteras.
Eventuella grafer som skapas under tentans gång behöver **INTE** skickas in för rättning, det räcker med att skicka in den kod som producerar figurerna.

OBS: Glöm inte att spara din fil ofta! Om R krashar kan kod förloras.

Programvaror

För att ladda in kursmodulen, kör följande kommando i en terminal

```
module load courses/732G33
```

så kommer du få tillgång till R, RStudio samt de R-paket som behövs för tentan. Öppna en terminal genom att trycka ctrl+alt+T

Kör sedan i terminalen

```
rstudio
```

för att öppna Rstudio.

Övriga program:

- `caja` används som filhanterare
- `pluma` används för att läsa textfiler, kan användas för att titta på datafiler innan inläsning.

Uppgifter

1. Datastrukturer (4p)

- (a) Återskapa data.frame enligt nedan. Den ska heta `myDF`. **1.5p**

```
  e1    e2 e3
1 30  TRUE a
2 29 FALSE b
3 14  TRUE c
4 13 FALSE d
5 57  TRUE e
6 18 FALSE f
```

- (b) Utgå från data.frame `myDF`. Använd funktioner i R för att göra följande: Beräkna medelvärde för `e1` och spara som `m1`. Välj ut de element i `e3` där `e2` är `FALSE`. Spara som `m2`. Skapa sen listan `myList`. Den ska ha `m1`, `m2` och `myDF` som element (i given ordning), elementnamnen ska vara samma som objektnamnen. **1.5p**
- (c) Återskapa matrisen nedan. Döp den till `myMat`. **1p**

```
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
[1,]    1    1    1    1    1    1    1    1    1    1
[2,]    1    4    1    1    1    1    1    1    1    1
[3,]    1    1    9    1    1    1    1    1    1    1
[4,]    1    1    1   16    1    1    1    1    1    1
[5,]    1    1    1    1   25    1    1    1    1    1
[6,]    1    1    1    1    1   36    1    1    1    1
[7,]    1    1    1    1    1    1   49    1    1    1
[8,]    1    1    1    1    1    1    1   64    1    1
[9,]    1    1    1    1    1    1    1    1   81    1
[10,]   10   20   30   40   50   60   70   80   90  100
```

2. Kontrollstrukturer (4p)

- (a) Skapa en matris med nollor som ni kallar **prodMat** av storlek 10×10 . Den matrisen ska fyllas med alla talen i multiplikationstabellerna för heltalen 1 till 10. Detta ska göras med en nästlad for-loop. Första raden ska vara 1:ans tabell, andra raden 2:ans tabell osv. Om ni gjort rätt ska matrisen se ut enligt nedan¹: **2p**

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]
[1,]	1	2	3	4	5	6	7	8	9	10
[2,]	2	4	6	8	10	12	14	16	18	20
[3,]	3	6	9	12	15	18	21	24	27	30
[4,]	4	8	12	16	20	24	28	32	36	40
[5,]	5	10	15	20	25	30	35	40	45	50
[6,]	6	12	18	24	30	36	42	48	54	60
[7,]	7	14	21	28	35	42	49	56	63	70
[8,]	8	16	24	32	40	48	56	64	72	80
[9,]	9	18	27	36	45	54	63	72	81	90
[10,]	10	20	30	40	50	60	70	80	90	100

- (b) Skriv en while-loop som upprepade gånger genererar ett slumptal (låt oss kalla det talet x) från en likformig fördelning (uniform distribution) över intervallet $[0, 1]$ och sedan skriver ut följande saker: Om $0 \leq x < 0.4$ så ska meningen 'The random number is smaller than 0.4' skrivas ut på skärmen. Om $0.4 \leq x < 0.9$ så ska meningen 'The random number is bigger than 0.4, but smaller then 0.9' skrivas ut till skärmen. Om $x \geq 0.9$ så ska meningen 'The random number is bigger than 0.9, abort!' skrivas ut till skärmen och loppen ska avbryta. Notera att beroende på seed så kan loppen iterera olika många gånger innan den avbryter. Nedan visas **ett** exempel på när loppen körs, utskriften beror på vilken seed som används innan loopen. **2p**

```
[1] "The random number is bigger than 0.4, but smaller then 0.9"
[1] "The random number is smaller than 0.4"
[1] "The random number is smaller than 0.4"
[1] "The random number is bigger than 0.4, but smaller then 0.9"
[1] "The random number is bigger than 0.4, but smaller then 0.9"
[1] "The random number is smaller than 0.4"
[1] "The random number is bigger than 0.4, but smaller then 0.9"
[1] "The random number is smaller than 0.4"
[1] "The random number is smaller than 0.4"
[1] "The random number is smaller than 0.4"
[1] "The random number is bigger than 0.9, abort!"
```

¹Matrisen kommer vara symmetrisk om ni gjort rätt, dvs den kommer se likadan ut över och under huvuddiagonalen.

3. Strängar och datum (4p)

- (a) Läs in paketen `lubridate` och `stringr` i R. Läs sedan in datamaterialet “BBCarticle201302.txt” och spara som vektorn `myText`. **0.5p**
- (b) Använd `stringr` för att: Räkna ut hur många gånger ordet “Burma” förekommer i `myText`. Spara som `bCount`. Välj sedan ut de element i `myText` där “Burma” nämns minst en gång, spara som vektorn `myTextBurma`. Räkna sedan ut antalet tecken i varje element i `myTextBurma` och spara i vektorn `myChar`. **1.5p**
- (c) Använd funktioner i R för att göra följande: **2p**
- Räkna ut antalet hela veckor mellan riksdagsvalet 2018 (9 september) och Europaparlamentsvalet 2019 (26 maj). Spara talet i variabeln `myWeek`.
 - Räkna ut hur många hela månader som det är mellan datumen 2016-04-12 och 2019-11-08. Spara som `myMonth`.
 - Skapa ett objekt med absolut tid som representerar tiden 2 h, 4 min och 8 sekunder. Döp objektet till `myTime`.
 - Skapa ett intervall-objekt som börjar på dagens datum (2019-06-10) och sträcker sig 5 månader och 4 dagar framåt (relativ tid). Spara som `myInterval`.

4. Funktioner och grafik (4p)

- (a) Skapa en funktion `skewness(x)`, som baserat på en vektor `x` beräknar urvalets skevhet på följande sätt. **2p**

$$\text{skewness}(x) = \frac{\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^3}{\left(\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2\right)^{3/2}}$$

där \bar{x} är medelvärdet för vektorn. Om `x` inte är numerisk så ska funktionen returnera ett fel med meddelandet “Not numeric!”. Se testfallen för hur funktionen ska fungera. Det är inte tillåtet att använda någon inbyggd funktion som beräknar skevhet i denna uppgift.

```
test1<-skewness(x = c(12,42,2,3,12,100))
test1

[1] 1.30058

skewness(x = "hej")

Error in skewness(x = "hej"): Not numeric!

data("trees")
skewness(x = trees[,1])

[1] 0.526316

skewness(x = trees[,2])

[1] -0.374869

skewness(x = trees[,3])

[1] 1.06436
```

- (b) Skapa nu funktionen `polynom(intercept,coef,x)`. `intercept` är ett tal och `coef` och `x` är numeriska vektorer. Funktionen ska beräkna polynom enligt följande:

$$y = intercept + \sum_{i=1}^N coef_i \cdot x^i$$

Tex så ger argumenten `intercept=10` och `coef=(2,-0.5,3)` upphov till ploynomet

$$y = 10 + 2 \cdot x - 0.5 \cdot x^2 + 3 \cdot x^3$$

Ett annat exempel är `intercept=5` och `coef=(-1,2,-3,4)`:

$$y = 5 - 1 \cdot x + 2 \cdot x^2 - 3 \cdot x^3 + 4 \cdot x^4$$

Notera att längden på `x` och `coef` inte behöver vara samma. Funktionen ska returnera en lista med `y` (det uträknade polynomet), `intercept`, `coef` och `x`. Se testfallen för hur funktionen ska fungera. **2p**

```
x1<- -5:5
list1<-polynom(intercept = 3,coef = 1,x = x1)
names(list1)

[1] "y"          "intercept" "coef"       "x"

list1

$y
[1] -2 -1  0  1  2  3  4  5  6  7  8

$intercept
[1] 3

$coef
[1] 1

$x
[1] -5 -4 -3 -2 -1  0  1  2  3  4  5

list2<-polynom(intercept = 3,coef = c(1,-2),x = x1)
list2

$y
[1] -52 -33 -18  -7   0   3   2  -3 -12 -25 -42

$intercept
[1] 3

$coef
[1] 1 -2

$x
[1] -5 -4 -3 -2 -1  0  1  2  3  4  5
```

```
list3<-polynom(intercept = 3,coef = c(2,0.1,-0.3),x = x1)
list3$y

[1] 33.0 15.8 6.0 1.8 1.4 3.0 4.8 5.0 1.8 -6.6 -22.0

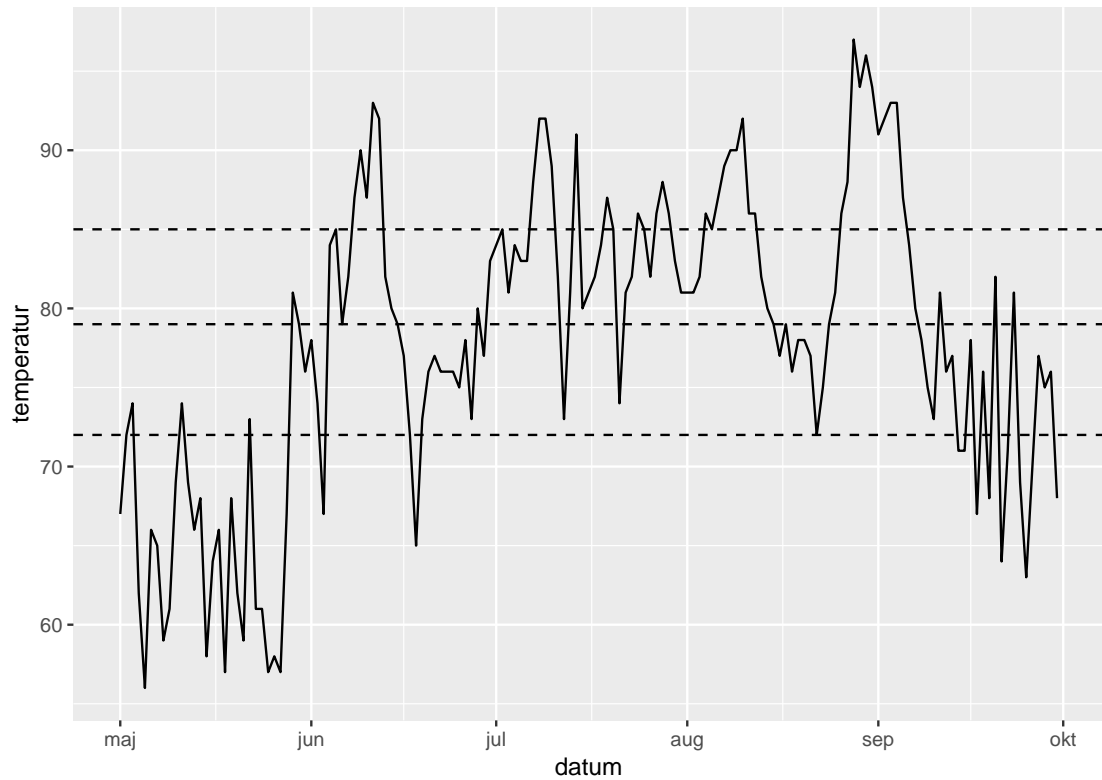
list4<-polynom(intercept = 3,coef = 5:1,x = c(5,-2,pi))
list4$y

[1] 4878.000 -15.000 652.043
```

5. Statistik, grafik (4p)

- (a) Utgår från det inbyggda datasetet `trees`. Beräkna alla parvisa korrelationer mellan variablerna i `trees` (ska vara tre stycken). Spara dessa i vektorn `myCorr`. Testa sen om korrelationen mellan “Girth” och “Height” är skild från noll. Spara p-värdet från testet i variabeln `pValue`. **1p**
- (b) Läs in datamaterialet “`wind_temp_data.csv`” i R och spara som en `data.frame` med namnet `wind`. Datamaterialet innehåller mätningar för temperaturen och vinden för olika dagar på en plats. **1p**
 - i. Ta reda på vilken månad där vinden har som högst standardavvikelse. Spara i variabeln `windMonth`.
 - ii. Gör ett histogram över temperaturen. Det ska vara en **relativ** skala på y-axeln, annars får ni ha valfritt utseende på grafen.
- (c) Återgå nu till din `data.frame` `wind`, och använd `ggplot2` för att göra följande: Återskapa figuren nedan. Figuren är en lineplot mellan variablerna “Date” och “Temp”. Referenslinjerna baseras på den undre kvartilen, medianen och den övre kvartilen² av “Temp”. Notera axeltexter. Skalan på x-axeln ska ha den rätta tidsskalan. **2p**

²Kallas även den 25:e, 50:e och 75:e percentilerna,



Kom ihåg: Skriv alla dina lösningar i en körbar **R-fil**. Spara filen med namnet **tentaX.R** där **X** är ditt SC-nummer (klientnummer). Det numret kan du se i studentklienten. Exempel: om du har SC-nummer SC12345 så ska filen heta **tentaSC12345.R**. Lämna sedan in din fil via studentklienten. När du har loggat ut från datorn är tentan avslutad.

Lycka till!