LINKÖPINGS UNIVERSITET
Institutionen för datavetenskap
Avdelningen för statistik
Måns Magnusson

2015-08-10
Programmering i R, 7.5 hp
732G33

Tentamen i Programmering i R, 7.5 hp

Skrivtid:	8-12
Hjälpmedel:	Inget tryckt material, dock finns "R reference card v.2" av Matt Baggot tillgängligt 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**.
Lösningen skrivs i en körbar textfil med namnet **Main.R**
Se filen **DocStudent.pdf** för hur tentan ska lämnas in.
Kommentera direkt i Main.R filen 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.

1. Datastrukturer, logik och beräkningar (4p)

   (a) Beräkna $e^{\pi} + \log_2(4^3)$ **1p**

   Lösningsförslag:

   ```
   exp(pi) + log(4^3, base=2)

   [1] 29.1407
   ```

   (b) Läs in datamaterialet `iris` och skapa en ny logisk variabel i datasetet som du kallar `small_petal` och som anger om variabeln `Petal.Length` är mindre än 4. **1p**

   Lösningsförslag:

   ```
   data(iris)
   iris$small_petal <- iris$Petal.Length < 4
   ```

(c) Beräkna den genomsnittliga `Sepal.Length` för de tre olika orkidéearterna som anges i variabeln `Species`. **1p**

Lösningsförslag:

```
aggregate(x = iris$Sepal.Length, by = list(iris$Species), FUN=mean)

     Group.1     x
1     setosa 5.006
2 versicolor 5.936
3  virginica 6.588
```

(d) Plocka ut de orkidéer som har en `Sepal.Width` som är mindre än 3 och en `Petal.Width` som är större än 2. **1p**

Lösningsförslag:

```
iris[iris$Sepal.Width < 3 & iris$Petal.Width > 2, ]

    Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
115          5.8         2.8          5.1         2.4 virginica
119          7.7         2.6          6.9         2.3 virginica
129          6.4         2.8          5.6         2.1 virginica
133          6.4         2.8          5.6         2.2 virginica
    small_petal
115       FALSE
119       FALSE
129       FALSE
133       FALSE
```

---

2. Kontrollstrukturer (4p)

(a) Skapa en kod som loopar över värdena 400 till 500 och skriver ut alla värden för vilka kvadratroten blir ett heltal (med `print()`). Exempel på tal som ska skrivas ut är 1, 4, 9, ... **2p**
Obs! Funktionen måste använda en `for`-loop.

Lösningsförslag:

```r
for (i in 400:500){
  if (sqrt(i) %% 1 == 0) print(i)
}
```

```
[1] 400
[1] 441
[1] 484
```

(b) Skapa nu en `while`-loop som, liknande i uppgift a), med en `while`-loop gå igenom värdena 500 till 600 och summera de värden vars kvadratrot är ett heltal. **2p**
Obs! Funktionen måste använda en `while`-loop.

Lösningsförslag:

```r
i <- 500
mysum <- 0
while (i <= 600){
  if (sqrt(i) %% 1 == 0) mysum <- mysum + i
  i <- i + 1
}
mysum
```

```
[1] 1105
```

---

3. Strängar och datum (4p)

(a) Läs in paketen `lubridate` och `stringr` i R. **0.5p**

Lösningsförslag:

```r
library(stringr)
library(lubridate)
```

(b) Marie Curie var en av vetenskapshistoriens främsta fysiker. Hon föddes den 7 november 1867 och dog den 4 juli 1934. Hon belönades med två nobelpris, 1903 och 1911. Nobelpriset delas ut den 10 december varje år. Använd `lubridate` för att besvara följande frågor: **1.5p**

i. Vilken veckodag föddes Marie Curie?

ii. Hur många veckor hade Marie Curie kvar att leva när hon fick sitt andra Nobelpris?
iii. Hur många dagar levde Marie Curie?

Lösningsförslag:

```
birth <- ymd("1867-11-07")
first_nobel <- ymd("1903-12-10")
second_nobel <- ymd("1911-12-10")
death <- ymd("1934-07-04")
# i.
wday(birth, label=TRUE)

[1] Thurs
Levels: Sun < Mon < Tues < Wed < Thurs < Fri < Sat

# ii.
interval(second_nobel, death) / weeks(1)

[1] 1177

# iii.
interval(birth, death) / days(1)

[1] 24345
```

(c) Läs in dikten `wilde.txt` i R som `poem`. **0.5p**

Lösningsförslag:

```
poem <- readLines("wilde.txt")
```

(d) Räkna ut den genomsnittliga längden på orden i dikten (med ord avses textsträngar som avgränsas med mellanslag). **1.5p**

Lösningsförslag:

```
words <- unlist(str_split(poem, pattern=" "))
words <- words[str_length(words) > 0]
mean(nchar(words))

[1] 4.72174
```

---

4. Funktioner (4p)

   (a) Skapa en funktion du kalla `approx_e()` som approximerar konstanten e på följande sätt. Funktionen ska generera ett värde på e som är korrekt till 4 decimalen som standard. Annars ska det gå att styra `K` nedan. **2p**

$$e \approx \sum_{k=0}^{K} \frac{1}{k!} \text{ där } K \to \infty$$

```
approx_e()

[1] 2.71825

approx_e(K = 2)

[1] 2.5
```

Lösningsförslag:

```
function(K = 7){
  k <- 0:K
  elem <- 1/factorial(k)
  resultat <- sum(elem)
  return(resultat)
}
```

   (b) Vi ska nu på ett liknande sätt implementera en funktion du kallar `approx_ln()` för att godtyckligt approximera $\ln(z)$. För att approximera använd följande funktion där approximationen blir godtyckligt god när $K \to \infty$. **2p**

$$\ln(z) = 2 \sum_{k=0}^{K} \frac{1}{2k+1} \left(\frac{z-1}{z+1} \right)^{2k+1}$$

```
approx_ln(exp(1), K = 2)

[1] 0.998455

approx_ln(exp(1), K = 5)

[1] 0.999992
```

Lösningsförslag:

```
function(z, K){
  k <- 0:K
  elem1 <- 1/(2*k + 1)
  elem2 <- ((z-1)/(z+1))^(2*k + 1)
  resultat <- 2 * sum(elem1 * elem2)
  return(resultat)
}
```

---

5. Statistik och grafik (4p)

   (a) Skapa en vektor $\mathbf{x}$ genom att upprepa talen 1 till 10, 20 ggr (d.v.s. totalt en vektor av längd 200). Simulera sedan data från vektorn $\mathbf{y}$ på följande sätt: **1p**

   $$y_i = 5 + 0.5 \cdot x_i + \epsilon_i$$

   där

   $$\epsilon_i \sim \mathcal{N}(\mu = 0, \sigma = 2)$$

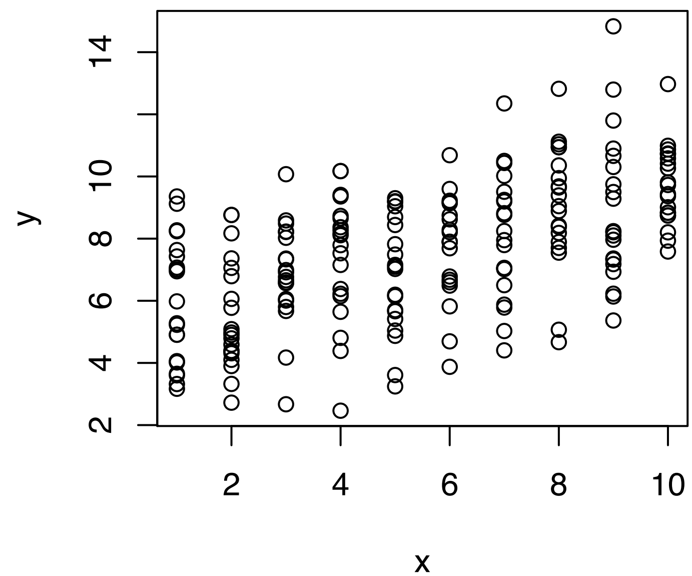
Lösningsförslag:

```
x <- rep(1:10, 20)
y <- 5 + 0.5*x + rnorm(length(x), 0, 2)
```

   (b) Visualisera variabeln x mot y i en scatterplot. **1p**

Lösningsförslag:

```
plot(x, y)
```



(c) Anpassa en linjär regressionsmodell mellan x mot y. **1p**

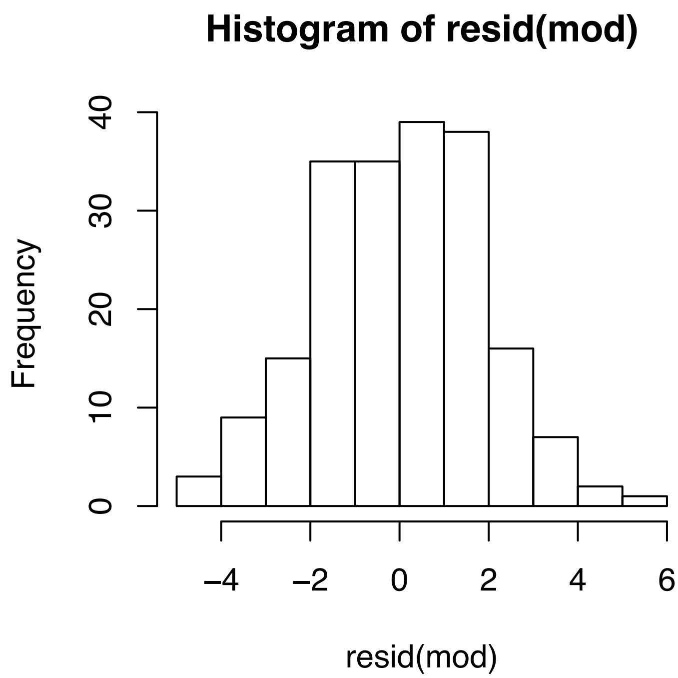
Lösningsförslag:

```
mod <- lm(y~x)
mod


Call:
lm(formula = y ~ x)

Coefficients:
(Intercept)            x
      5.216        0.434
```

(d) Plocka ut och visualisera residualerna i ett histogram. **1p**

Lösningsförslag:

```
hist(resid(mod))
```



Lycka till!