LINKÖPINGS UNIVERSITET
Institutionen för datavetenskap
Avdelningen för statistik
Josef Wilzén & Måns Magnusson

2014-08-13
Programmering i R, 7.5 hp
732G33

Tentamen i Programmering i R, 7.5 hp

Skrivtid: 14-18
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 (4p)

   (a) Beräkna $\ln(2) + \cos(\pi/\sqrt{3}) + e$ **1p**

   Lösningsförslag:

```
log(2) + cos(pi/sqrt(3)) + exp(1)
```

```
[1] 3.17081
```

   (b) Skapa följande objekt: **1.5p**

      i. `myTextVec`, som innehåller textelementen **''Kalle'', ''Lotta'',''Linda''** repeterat 17 gånger (d.v.s. 51 element).

      Lösningsförslag:

```
myTextVec <- rep(c("Kalle", "Lotta", "Linda"), 17)
```

      ---

ii. `x`: en numerisk vektor innehållande alla tal $(110, 115, ..., 355, 360)$ (också 51 element).

Lösningsförslag:

```
x <- 105 + 1:51 * 5
```

---

iii. `myBoolean`: en logisk vektor som är `TRUE` då vektorn `x` ovan är jämt delbar med sju och annars `FALSE`.

Lösningsförslag:

```
myBoolean <- x%%7 == 0
```

(c) Skapa en `data.frame` kallad `mittData`, där objekten i (b) ska vara variablerna i datasetet. **0.5p**

Lösningsförslag:

```
mittData <- data.frame(myTextVec = myTextVec, x = x, myBoolean)
head(mittData)

  myTextVec   x myBoolean
1     Kalle 110     FALSE
2     Lotta 115     FALSE
3     Linda 120     FALSE
4     Kalle 125     FALSE
5     Lotta 130     FALSE
6     Linda 135     FALSE
```

(d) Skapa en ny variabel i `mittData` som heter `roll`. Om variabeln `myTextVec` är `Linda` eller `Kalle` ska variabeln `roll` ha värdet `Lektor` och om det är `Lotta` ska värdet vara `Studierektor`. **1p**

Lösningsförslag:

```
mittData$roll <- "Lektor"
mittData$roll[mittData$myTextVec == "Lotta"] <- "Studierektor"
head(mittData)
```

```
  myTextVec   x myBoolean         roll
1     Kalle 110     FALSE       Lektor
2     Lotta 115     FALSE Studierektor
3     Linda 120     FALSE       Lektor
4     Kalle 125     FALSE       Lektor
5     Lotta 130     FALSE Studierektor
6     Linda 135     FALSE       Lektor
```

---

2. Kontrollstrukturer (4p)

   (a) För ett givet tal $0 < z < 2$ går logaritmen att räkna ut på följande sätt:

$$\begin{aligned} \ln(z) &= (z-1) - \frac{(z-1)^2}{2} + \frac{(z-1)^3}{3} - \frac{(z-1)^4}{4} + \cdots \\ &= \sum_{i=1}^{n} (-1)^{i-1} \frac{(z-1)^i}{i} \end{aligned}$$

   Skapa en for-loop som räknar ut en approximation för $z = 1.5$ och $n = 7$. **2p**

Lösningsförslag:

```
z <- 1.5
n <- 7
logz <- 0
for (i in 1:n) {
    logz <- logz + (-1)^(i - 1) * ((z - 1)^i)/i
}
logz
```

```
[1] 0.405804
```

```
log(z)  # Jmf med Rs log-funktion
```

```
[1] 0.405465
```

   (b) Implementera en whileloop som gör samma beräkning (för $z = 1.5$ och $n = 7$). **2p**

Lösningsförslag:

```
z <- 1.5
n <- 7
logz <- 0
i <- 1
while (i <= n) {
    logz <- logz + (-1)^(i - 1) * ((z - 1)^i)/i
    i <- i + 1
}
logz
```

```
[1] 0.405804
```

---

3. Strängar och datum (4.5p)

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

   Lösningsförslag:

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

   (b) Andra världskriget inleddes den 1 september 1939 och avslutades 2 september 1945. Använd lubridate för att beräkna hur länge kriget pågick i: **1.5p**

   i. Hela veckor
   ii. Hela månader
   iii. Dagar

   Lösningsförslag:

```
start <- ymd("1939-09-01")
slut <- ymd("1945-09-02")
interval(start, slut)/weeks(1)  # i)
```

```
[1] 313
```

```
interval(start, slut)/months(1)  # ii)
```

```
[1] 72
```

```
interval(start, slut)/days(1)  # iii)
```

```
[1] 2193
```

(c) Läs in artikeln `robot_termites.txt` i R som `robotTermites`. **0.5p**

Lösningsförslag:

```
robotTermites <- readLines("robot_termites.txt")
```

(d) Räkna antalet ord totalt i artikeln (ord definieras som textsträngar som är avgränsade med ett mellanslag, radbörjan eller radslut). **1.5p**

Lösningsförslag:

```
words <- unlist(str_split(robotTermites, pattern = " "))
words <- words[str_length(words) > 0]
length(words)
```

```
[1] 767
```

4. Funktioner: Approximation av π (4p)

(a) Talet π är ett irrationellt tal. För att skapa ett numeriskt värde för π behöver vi använda någon form av approximationsmetod. Nedan är Newtons formel för approximation av π.

$$\pi = 2\sum_{k=0}^{K} \frac{2^k k!^2}{(2k+1)!} \text{ där } K \to \infty$$

Vi kan således få en godtyckligt god approximation genom att välja olika värden för $K \geq 0$. Implementera en funktion som du kallar `myPi(K)` med parametern `K`. **2p**
Exempel:

```
myPi(K = 3)
```

```
[1] 3.04762
```

Lösningsförslag:

```
myPi

function(K){
  k <- 0:K
  taljare <- 2^k * factorial(k)^2
  namnare <- factorial(2*k + 1)
  resultat <- 2 * sum(taljare / namnare)
  return(resultat)
}
```

(b) Vi ska nu använda funktionen ovan för att beräkna volymen av ett klot. Volymen beräknas på följande sätt:

$$V = \frac{4}{3}\pi r^3$$

där r är klotets radie. Använd din approximation av pi ovan (`myPi()`) och skapa en funktion du kallar `mySphere(K,r)` med argumenten `K` och `r` som beräknar klotets volym där `K` används för approximationen av π. **2p**

Exempel:

```
mySphere(K = 3, 1)

[1] 4.06349

mySphere(K = 10, 1)

[1] 4.18814
```

Lösningsförslag:

```
mySphere

function(K, r){
  resultat <- (4 / 3) * myPi(K) * r^3
  return(resultat)
}
```

---

5. Funktioner: Statistik och grafik (4p)

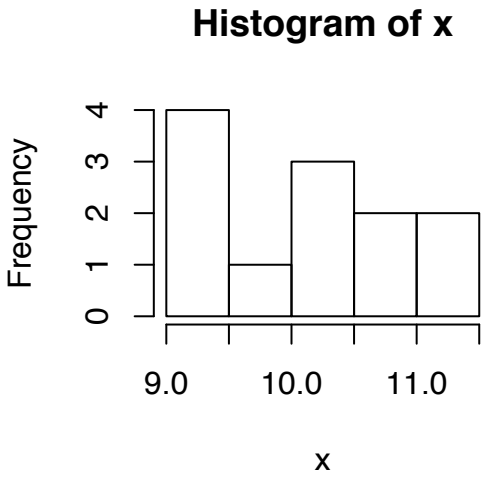
(a) Simulera två variabler X ($n_1 = 12$) och Y ($n_2 = 31$) från

$$X \sim \mathcal{N}(10, 1) \text{ och } Y \sim \mathcal{N}(11, 1)$$

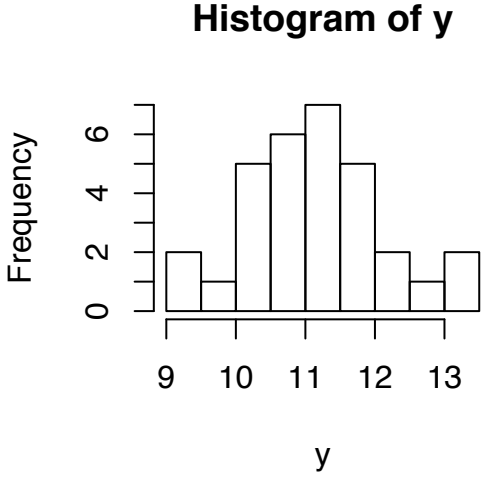
och skapa ett histogram för varje variabel. **1p**

Lösningsförslag:

```
set.seed(20140101)
x <- rnorm(12, 10, 1)
y <- rnorm(31, 11, 1)
hist(x)
```




```
hist(y)
```



(b) Vi ska nu skapa en egen funktion för att göra ett vanligt "two sample t-test". Implementera följande två funktioner i R som `mys(x,y)` och `myt(x,y)` där `x` och `y` är vektorer och där

`myt(x,y)` beräknar t-statistikan för två godtyckliga vektorer. **1p**

$$s(\mathbf{x},\mathbf{y}) = \sqrt{\frac{(n_x-1)s_x^2 + (n_y-1)s_y^2}{n_x+n_y-2}}$$

$$t(\mathbf{x},\mathbf{y}) = \frac{\bar{x}-\bar{y}}{s(\mathbf{x},\mathbf{y}) \cdot \sqrt{\frac{1}{n_x}+\frac{1}{n_y}}}$$

där n_x är längden av vektorerna $\mathbf{x}$, $\bar{x}$ är medelvärdet för $\mathbf{x}$ och s_x^2 är variansen för vektorn $\mathbf{x}$.

Exempel:

```
mys(x = 1:10, y = 11:20)

[1] 3.02765

myt(x = 1:10, y = 11:20)

[1] -7.38549
```

Lösningsförslag:

```
mys

function(x,y){
  taljare <- (length(x) - 1) * var(x) + (length(y) - 1) * var(y)
  namnare <- length(x) + length(y) - 2
  resultat <- sqrt(taljare / namnare)
  return(resultat)
}

myt

function(x,y){
  taljare <- mean(x) - mean(y)
  namnare <- mys(x,y) * sqrt(1 / length(x) + 1 / length(y))
  resultat <- taljare / namnare
  return(resultat)
}
```

(c) Skapa en funktion `myTTest(x,y)` (**Obs!** t-test med lika varians) som beräknar t-statistikan ovan, antal frihetsgrader och p-värdet för t-statistikan och skriver detta till skärmen. Testet ska vara ensidigt (μ_x är mindre än μ_y). Antalet frihetsgrader för testet beräknas på följande sätt:

$$\text{df} = n_x + n_y - 2$$

Pröva funktionen på de simulerade vektorerna i a) ovan. **1.5p**

Exempel:

```
myTTest(x = 6:10, y = 11:15)

My t-test:
t: -5
df: 8
p-value: 0.000526413
```

Lösningsförslag:

```
myTTest

function(x,y){
  tValue <- myt(x,y)
  df <- length(x) + length(y) - 2
  pValue <- pt(q=tValue, df=df)
  cat("My t-test:\nt:", tValue,"\ndf:",df,"\np-value:",pValue)
}

myTTest(x, y)

My t-test:
t: -3.10294
df: 41
p-value: 0.00173192
```

(d) Gör nu om samma test med funktionen `t.test()`. **0.5p**

Lösningsförslag:

```
t.test(x, y, var.equal = TRUE, alternative = "less")


Two Sample t-test

data:  x and y
t = -3.1029, df = 41, p-value = 0.001732
alternative hypothesis: true difference in means is less than 0
95 percent confidence interval:
      -Inf -0.449338
sample estimates:
mean of x mean of y
  10.1633   11.1452
```

Lycka till!