LINKÖPINGS UNIVERSITET
Institutionen för datavetenskap
Avdelningen för statistik
Måns Magnusson

2015-03-26
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 $\sqrt{(1 + \cos(\pi/3))}$ **1p**

   Lösningsförslag:

```
sqrt(1+cos(pi/3))
```

```
[1] 1.22474
```

   (b) Läs in det interna datamaterialet `mtcars` och räkna ut det genomsnittliga miles per galleon (mpg). **1p**

   Lösningsförslag:

```
data(mtcars)
mpg_mean <- mean(mtcars$mpg)
mpg_mean
```

```
[1] 20.0906
```

(c) Beräkna den genomsnittliga mpg efter antalet cylindrar. Ta dock först bort manuellt växlade bilar ur materialet (d.v.s. `am = 1`). Använd funktionen `aggregate()`. **1p**

Lösningsförslag:

```
new_mtcars <- mtcars[mtcars$am==0,]
mpg_by_cyl <- aggregate(new_mtcars$mpg, by=list(new_mtcars$cyl), FUN=mean)
mpg_by_cyl

  Group.1      x
1       4 22.900
2       6 19.125
3       8 15.050
```

(d) Skapa en lista som innehåller dels listelementet med namnet `mpg_mean` som innehåller resultatet från (b) ovan och ett listelement du kallar `mpg_by_cyl` som innehåller en vektor med resultatet från (c). **1p**

Lösningsförslag:

```
list(mpg_mean = mpg_mean, mpg_by_cyl = mpg_by_cyl)

$mpg_mean
[1] 20.0906

$mpg_by_cyl
  Group.1      x
1       4 22.900
2       6 19.125
3       8 15.050
```

2. Kontrollstrukturer (4p)

(a) Skapa en tom matris av storlek 2×3 och fyll den sedan med elementen $a_{ij} = i \cdot j$ där i är radnumret och j är kolumnnumret. Du ska använda en nästlad `for`-loop. **2p**

$$\mathbf{A} = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 4 & 6 \end{pmatrix}$$

Lösningsförslag:

```
A <- matrix(0,2,3)
for(i in 1:2){
  for(j in 1:3){
    A[i,j] <- i * j
  }
}
A
```

```
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    2    4    6
```

(b) Skapa en `while`-loop som skriver ut tal jämt delbara med 7 mellan 100 och 110 med `print()`. **2p**

Lösningsförslag:

```
i <- 100
while(i <= 110){
  if(i %% 7 == 0) print(i)
  i <- i + 1
}
```

```
[1] 105
```

---

3. Strängar och datum (4p)

(a) Läs in paketen `lubridate` och `stringr` i R. I paketet lubridate finns ett dataset kallat lakers. Läs in detta dataset. **0.5p**

Lösningsförslag:

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

(b) Läs in det interna datasetet `lakers`. I detta datmaterial finns variabeln `date`. Beräkna i genomsnitt hur många veckor dessa datum är från datumet 2009-12-31. **1.5p**

Lösningsförslag:

```
data(lakers)
mean(interval(ymd(lakers$date), ymd("2009-12-31")) / weeks(1))

[1] 48.4757
```

(c) Läs in wikipediaartikeln `wiki_robot.txt` i R som `wiki`. **0.5p**

Lösningsförslag:

```
wiki <- readLines("wiki_robot.txt")
```

(d) Räkna ut med `stringr` vilka rader som innehåller siffror av något slag. **1.5p**

Lösningsförslag:

```
str_detect(wiki, "[0-9]")

[1] FALSE  TRUE FALSE  TRUE  TRUE  TRUE
```

---

4. Funktioner (4p)

Vi ska skapa en funktion för Eratosthenes såll - en metod för att hitta primtal. Primtal är de tal som är jämt delbara med sig själv och . Exempel på primtal är 2, 3, 5, 7, ..., 31 o.s.v.

För att undersöka om ett givet tal p är ett primtal behöver vi därför gå igenom samtliga heltal från 2 till $\sqrt{p}$ och undersöka om något tal kan dela talet p jämt. Funktionen ska returnera `TRUE` om talet är ett primtal, annars ska `FALSE` returneras. **4p**

```
prime(99)

[1] FALSE

prime(97)
```

```
[1] TRUE

prime(9)

[1] FALSE
```

Lösningsförslag:

```
function(p){
  sq <- floor(sqrt(p))
  res <- TRUE
  for(i in 2:sq){
    if(p %% i == 0) res <- FALSE
  }
  return(res)
}
```

---

5. Statistik och grafik (4p)

   (a) Skapa en matris som ser ut som nedan. Det är antalet vuxna överlevande från Titanic efter kön (kolumn) och klass (rad). Gör ett χ^2-test på detta material. **1p**

```
    Male Female
1st   57    140
2nd   14     80
3rd   75     76
```

   .

   Lösningsförslag:

```
mat <- matrix(c(57,140,14,80,75,76), ncol=2, byrow = TRUE, dimnames=list(c("1st", "2nd"

chisq.test(mat)


Pearson's Chi-squared test

data:  mat
X-squared = 34.37, df = 2, p-value = 0.00000003441
```

(b) Vi ska nu göra ett eget χ^2-test. Beräkna χ^2-statistikan för matrisen du skapat ovan.

$$\chi^2 = \sum_{i=1}^{r}\sum_{j=1}^{c}\frac{(O_{i,j} - E_{i,j})^2}{E_{i,j}}.$$

där $O_{i,j}$ är det observerade antalet i rad i och kolumn j och $E_{i,j}$ är det förväntade värdet för rad i och kolumn j. $E_{i,j}$ beräknas på följande sätt.

$$E_{i,j} = n \cdot p_{r,j} \cdot p_{i,c}$$

där n är det totala antalet observationer, $p_{r,j}$ är radproportionen för kolumn j och $p_{i,c}$ är kolumnproportionen för rad$i$. **2p**
Tips! `rowSums()` och `colSums()`.

Lösningsförslag:

```
n <- sum(mat)
pc <- rowSums(mat)/n
pr <- colSums(mat)/n
chi_sum <- 0
for(i in 1:3){
  for(j in 1:2){
    E <- n * pr[j] * pc[i]
    O <- mat[i,j]
    chi_sum <- chi_sum + (O-E)^2 / E
  }
}
chi_sum
```

```
 Male
34.37
```

(c) Beräkna p-värdet för denna statistika i en χ^2-fördelning med två frihetsgrader. **1p**

Lösningsförslag:

```
1 - pchisq(chi_sum, df = 2)
```

```
           Male
0.0000000344073
```

Lycka till!