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**

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

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

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

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

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

   (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**

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**

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

3. Strängar och datum (4.5p)

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

(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

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

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

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
```

(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
```

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**

   (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
```

   (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:

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

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

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

Lycka till!