Procvičování 9 - řešení

1. Importujte do **R** tabulku jmen s daty narození českých občanů *jmena.txt*. Nazvěte ji *jm.data*. Tabulka obsahuje v řádcích všechna jednoslovná křestní jména českých obyvatel, ve sloupcích pak jejich frekvenci v jenotlivých ročnících od r. 1896 do 2013. Tabulka je docela velká, nepokoušejte se ji proto celou zobrazit v **R**, nechejte si raději vypsat jen prvních pár řádků a sloupců. Najdete ji ve studijních materiálech, nebo ji můžete načíst přímo z `http://www.sci.muni.cz/~syrovat/jmena.txt`, to ale bude chvilku trvat.

```
# Mam problem s encodovanim v LyXu, nepodarilo se mi pres LyX nacist spravne jmena
# s diakritikou, coz zpusobuje, ze nektera jmena jsou zdanlive duplikovana.
# Nactu proto dataset bez jmen radku, odstranim duplikovana jmena, a pak teprve definuji
# jmena radku.
# Nastaveni pracovniho adresare:
setwd("D:/My Dropbox/predmety/uvod do R/2014/cv09")
# nacteni jmen bez jmen radku, zabranim vytvoreni faktoru ze jmen a kontrole jmen sloupcu:
jm.data<- read.delim('jmena.txt', sep= '\t', check.names= F, stringsAsFactors= F)
# odstraneni radku s duplikovanymi jmeny:
jm.data<- jm.data[!duplicated(jm.data[, 1]), ]
# definovani jmen radku:
jm.data<- data.frame(jm.data, row.names= 1)

## Error: row names contain missing values

# R hlasi, ze mezi jmeny jsou missing values (NA), odstranim i ty:
jm.data<- jm.data[!is.na(jm.data[, 1]), ]
# definovani jmen radku podruhe:
jm.data<- data.frame(jm.data, row.names= 1)
```

2. Do vektoru *jm.freq* vypočítejte frekvenci výskytu jednotlivých jmen v české populaci.

```
jm.freq<- rowSums(jm.data)
```

3. Do vektoru *jm* si vytáhněte křestní jména z dataframu *jm.data*.

```
jm<- rownames(jm.data)
```

4. Seřaďte křestní jména podle abecedy a nechejte si zobrazit prvních a posledních 20.

```
sort(jm)[1:20]

##  [1] ""            "\016ABER"    "\016AKELIN"  "\016AKELINA" "\016AKELMNA"
##  [6] "\016AKLE"    "\016AKLINA"  "\016ANA"     "\016ANARGUL" "\016ANDA"
## [11] "\016ANET"    "\016ANETA"   "\016ANETKA"  "\016ANETTA"  "\016ANNA"
## [16] "\016ANNETA"  "\016ARKO"    "\016AV"      "\016EANETA"  "\016ELAN"

# Vas vysledek by mel byt jiny (lepsi), me zlobi to kodovani v LyXu.
```

5. Zjistěte nejdelší křestní jméno. Postup: vytvořte si vektor délek jmen, seřaďte jména podle jejich délek od nejvyšších po nejnižší a vyberte první. Délkou rozumíme počet znaků (characters). Funkci, která zjistí délku textového řetězce musíte vyhledat.

```
# Hledame funkci, ktera vrati pocet znaku v nejakem retezci, anglicky "number of characters".
# Nejprve muzeme zkusit hledani funkce uvnitr nasi instalace R pomoci ??.
# ??"number of characters"
# mi vratilo funkci nchar z baliku base (psano base::nchar).
# Pripadne (nebo pokud ?? nefunguje) muzeme primo hledat na googlu. Jeden z prvnich
# odkazu urcite bude smerovat na stejnou funkci.
# Abychom zjistili, jak funkci pouzit, podivame se na jeji napovedu:
# ?nchar
# Vektor delek jmen (poctu znaku ve jmenech):
jm.nchar<- nchar(jm)
# Ted staci jmena seradit sestupne podle delek a vybrat prvni:
jm[order(-jm.nchar)[1]]

## [1] "ANDRIANANDRAINA"

# Tohle mame snad vsichni stejne.
```

6. Nechejte si vypsat 100 nejkratších jmen.

```
jm[order(jm.nchar)[1:100]]

##   [1] ""    "E"   "A"   "T"   "R"   "I"   "N"   "O"   "P"   "U"   "V"
##  [12] "KA"  "AI"  "AJ"  "AL"  "AN"  "IK"  "TI"  "TA"  "AR"  "ES"  "OM"
##  [23] "OT"  "BA"  "IR"  "BI"  "BO"  "KO"  "DA"  "JA"  "DE"  "DI"  "DO"
##  [34] "DU"  "ED"  "EL"  "AD"  "EK"  "GA"  "HA"  "IM"  "EM"  "HI"  "AM"
##  [45] "MK"  "JE"  "JO"  "KE"  "KI"  "AV"  "OF"  "MM"  "LA"  "LE"  "LY"
##  [56] "MA"  "MI"  "MO"  "MR"  "MY"  "OK"  "OR"  "OZ"  "PA"  "PE"  "PI"
##  [67] "IA"  "RA"  "MD"  "RI"  "RM"  "RU"  "ID"  "RY"  "R]"  "SA"  "SI"
##  [78] "SY"  "ON"  "TE"  "TL"  "TU"  "TZ"  "TY"  "IE"  "UR"  "AK"  "VI"
##  [89] "VM"  "VU"  "IN"  "VY"  "WU"  "YA"  "YE"  "YI"  "ZO"  "ABA" "ABE"
## [100] "ABI"

# opet, vas vysledek bude jiny.
```

7. Nechejte si vypsat 50 nejběžnějších jmen.

```
jm[order(-jm.freq)[1:50]]

##  [1] "JIXM"      "JAN"       "MARIE"     "PETR"      "JANA"
##  [6] "JOSEF"     "PAVEL"     "MARTIN"    "JAROSLAV"  "EVA"
## [11] "MIROSLAV"  "HANA"      "ANNA"      "ZDENLK"    "MICHAL"
## [16] "LENKA"     "KATEXINA"  "VLRA"      "MILAN"     "KAREL"
## [21] "LUCIE"     "ALENA"     "PETRA"     "JAKUB"     "DAVID"
## [26] "JAROSLAVA" "VLADIMMR"  "VERONIKA"  "MARTINA"   "JITKA"
## [31] "TEREZA"    "LUDMILA"   "HELENA"    "MICHAELA"  "LADISLAV"
```

```
## [36] "ZDERKA"    "ONDXEJ"    "IVANA"     "ROMAN"     "STANISLAV"
## [41] "JARMILA"   "MONIKA"    "MAREK"     "ZUZANA"    "JIXINA"
## [46] "MARKITA"   "RADEK"     "DANIEL"    "ANTONMN"   "MARCELA"

# tady taky vas vysledek bude jiny.
```

8. Zjistěte, zda se Vaše jméno nachází mezi 50 nejběžnějšími.

```
# 50 nejbeznejsich jmen uz umime zjistit, ted se jen zeptame, zda to nase je mezi nimi:
"VIT" %in% jm[order(-jm.freq)[1:50]]

## [1] FALSE

# Pokud bychom neznali %in% operator, muzeme se porovnat nase jmeno
# s kazdym z 50 nejbeznejsich a secist logicke hodnoty vzesle z tohoto porovnani.
# Pokud je jejich suma > 0, pak se moje jmeno vyskytuje mezi temi 50:
sum("VIT" == jm[order(-jm.freq)[1:50]]) > 0

## [1] FALSE
```

9. Zjistěte kolikáté nejběžnější je Vaše jméno.

```
# Muzeme vyuzit toho, ze jmena rozmeru (radku ci sloupcu) se pri vypoctech
# mezi objekty prenaseji. Staci zjistit poradi (rank) frekvenci jmen a vyhledat
# poradi daneho jmena. Zajima nas samozrejme poradi od nejvyssi po nejnizsi prekvenci:
rank(-jm.freq)["VIT"]

##  VIT
## 6486

# Stejneho dosahneme i tak, ze ke jmenum pripojime jejich ranky, a pak se podivame na radek
# daneho jmena. Zobrazit muzeme i frekvenci:
jm.dtf<- data.frame(jmena= jm, jm.freq= jm.freq, jm.rank= rank(-jm.freq), row.names= 1)
jm.dtf["VIT", ]

##     jm.freq jm.rank
## VIT       2    6486
```

10. Naimportujte do **R** dataframy *spe* a *env* a vektor korelací abundancí jednotlivých druhů s Froudeho číslem *korelace*. Použijte funkce `read.delim()` a `scan()`. Vše najdete v *dat09.xls*.

```
spe<- read.delim('D:/My Dropbox/predmety/uvod do R/2013/cv09/spe.txt', row.names=1)
env<- read.delim('D:/My Dropbox/predmety/uvod do R/2013/cv09/env.txt', row.names=1)
```

```
kor <- scan()
```

11. Nechejte si vypsat zaokrouhlené hodnoty Froudeho čísla (na 2 desetinná místa) seřazené od nejmenší po největší, a pak od největší po nejmenší.

```
sort(round(env$froude, 2))

##  [1] 0.00 0.02 0.04 0.05 0.05 0.07 0.07 0.13 0.13 0.14 0.14 0.14 0.17 0.18
## [15] 0.18 0.22 0.24 0.30 0.30 0.33 0.41 0.43 0.48 0.51 0.52 0.54 0.56

sort(round(env$froude, 2), decreasing=T)

##  [1] 0.56 0.54 0.52 0.51 0.48 0.43 0.41 0.33 0.30 0.30 0.24 0.22 0.18 0.18
## [15] 0.17 0.14 0.14 0.14 0.13 0.13 0.07 0.07 0.05 0.05 0.04 0.02 0.00
```

12. Zjistěte 5 nejnižších a 5 nejvyšších hodnot Froudeho čísla.

```
sort(env$froude)[1:5]

## [1] 0.001813 0.020319 0.039662 0.046165 0.051925

sort(env$froude, decreasing= T)[1:5]

## [1] 0.5584 0.5395 0.5225 0.5071 0.4767

# nebo naraz treba takhle:
sort(env$froude)[c(1:5, (length(env$froude)-4) : length(env$froude))]

##  [1] 0.001813 0.020319 0.039662 0.046165 0.051925 0.476675 0.507082
##  [8] 0.522477 0.539509 0.558432
```

13. Nechcejte si vypsat jména 5 lokalit s nejvyšším Froudeho číslem.

```
rownames(env)[order(env$froude, decreasing=T)][1:5]

## [1] "s24" "s25" "s12" "s21" "s20"
```

14. Do dataframu *env* přidejte proměnné *ind* a *spec* obsahující celkové počty jedinců a druhů pakomárů na lokalitách.

```
env$ind<- rowSums(spe)
env$spec<- rowSums(spe>0)
```

15. Nechejte si vypsat počty druhů seřazené podle Froudeho čísla od nejmenšího po největší.

```
env$spec[order(env$froude)]

##  [1] 31 28 27 19 26 29 25 15 20 16 22 19 17 27 22 21 31 26 30 23 19 17 20
## [24] 28 24 19 31
```

16. Nechejte si vypsat dataframe *env* seřazený podle Froudeho čísla od nejvyššího po nejnižší.

```
env[order(-env$froude),]
```

```
##          gr depth   vel   froude ind spec
## s24  Er_VEG 0.300 0.958 0.558432 678   31
## s25      Er 0.139 0.630 0.539509  47   19
## s12  Er_VEG 0.245 0.810 0.522477 450   24
## s21  Er_VEG 0.138 0.590 0.507082 531   28
## s20      Er 0.290 0.804 0.476675 221   20
## s08      Er 0.213 0.618 0.427528 146   17
## s10  Er_VEG 0.353 0.758 0.407331 212   19
## s09      Er 0.243 0.508 0.329023 220   23
## s15  Er_VEG 0.279 0.496 0.299809 738   30
## s07      Er 0.278 0.490 0.296715 258   26
## s18  Er_VEG 0.200 0.340 0.242733 565   31
## s11  Er_VEG 0.236 0.334 0.219511 181   21
## s14      Ep 0.155 0.224 0.181655 150   22
## s02      Ep 0.422 0.358 0.175951 405   27
## s16      Ep 0.501 0.372 0.167799 122   17
## s03      Ep 0.496 0.310 0.140536 146   19
## s01      Ep 0.395 0.274 0.139193 139   22
## s17      Ep 0.454 0.286 0.135520 243   16
## s22      Ep 0.340 0.246 0.134698 122   20
## s23      Ep 0.344 0.234 0.127380 145   15
## s06 Ep_FPOM 0.328 0.126 0.070242 611   25
## s13 Ep_CPOM 0.184 0.088 0.065500 569   29
## s05 Ep_FPOM 0.320 0.092 0.051925 489   26
## s04 Ep_FPOM 0.291 0.078 0.046165 183   19
## s19      Ep 0.162 0.050 0.039662 198   27
## s27 Ep_CPOM 0.478 0.044 0.020319 416   28
## s26 Ep_CPOM 0.124 0.002 0.001813 893   31
```

17. Zjistěte, kolik druhů a kolik jedinců pakomárů bylo na 5 lokalitách s nejvyšším Froudeho číslem (vyberte jen příslušné elementy z proměnných *ind* a *spec* dataframu *env*).

```
env[order(env$froude, decreasing=T)[1:5], c('ind','spec')]
```

```
##     ind spec
## s24 678   31
## s25  47   19
## s12 450   24
## s21 531   28
## s20 221   20
```

18. Zjistěte, jaké Froudeho číslo bylo naměřeno na 3 lokalitách s nejvyšším počtem jedinců pakomárů.

```
env$froude[order(env$ind, decreasing=T)[1:3]]
```

```
## [1] 0.001813 0.299809 0.558432
```

19. Nechejte si vypsat dataframe *env* s lokalitami seřazenými podle Froudeho čísla.

```
env[order(env$froude),]
```

```
##          gr depth   vel   froude ind spec
## s26 Ep_CPOM 0.124 0.002 0.001813 893   31
## s27 Ep_CPOM 0.478 0.044 0.020319 416   28
## s19      Ep 0.162 0.050 0.039662 198   27
## s04 Ep_FPOM 0.291 0.078 0.046165 183   19
## s05 Ep_FPOM 0.320 0.092 0.051925 489   26
## s13 Ep_CPOM 0.184 0.088 0.065500 569   29
## s06 Ep_FPOM 0.328 0.126 0.070242 611   25
## s23      Ep 0.344 0.234 0.127380 145   15
## s22      Ep 0.340 0.246 0.134698 122   20
## s17      Ep 0.454 0.286 0.135520 243   16
## s01      Ep 0.395 0.274 0.139193 139   22
## s03      Ep 0.496 0.310 0.140536 146   19
## s16      Ep 0.501 0.372 0.167799 122   17
## s02      Ep 0.422 0.358 0.175951 405   27
## s14      Ep 0.155 0.224 0.181655 150   22
## s11  Er_VEG 0.236 0.334 0.219511 181   21
## s18  Er_VEG 0.200 0.340 0.242733 565   31
## s07      Er 0.278 0.490 0.296715 258   26
## s15  Er_VEG 0.279 0.496 0.299809 738   30
## s09      Er 0.243 0.508 0.329023 220   23
## s10  Er_VEG 0.353 0.758 0.407331 212   19
## s08      Er 0.213 0.618 0.427528 146   17
## s20      Er 0.290 0.804 0.476675 221   20
## s21  Er_VEG 0.138 0.590 0.507082 531   28
## s12  Er_VEG 0.245 0.810 0.522477 450   24
## s25      Er 0.139 0.630 0.539509  47   19
## s24  Er_VEG 0.300 0.958 0.558432 678   31
```

20. Nechejte si vypsat dataframe *env* s lokalitami seřazenými podle typu habitatu (*gr*) a uvnitř typů podle hloubky vody.

```
env[order(env$gr, env$depth),]
```

```
##          gr depth   vel   froude ind spec
## s14      Ep 0.155 0.224 0.181655 150   22
## s19      Ep 0.162 0.050 0.039662 198   27
## s22      Ep 0.340 0.246 0.134698 122   20
## s23      Ep 0.344 0.234 0.127380 145   15
## s01      Ep 0.395 0.274 0.139193 139   22
## s02      Ep 0.422 0.358 0.175951 405   27
## s17      Ep 0.454 0.286 0.135520 243   16
## s03      Ep 0.496 0.310 0.140536 146   19
## s16      Ep 0.501 0.372 0.167799 122   17
## s26 Ep_CPOM 0.124 0.002 0.001813 893   31
## s13 Ep_CPOM 0.184 0.088 0.065500 569   29
```

```
## s27 Ep_CPOM 0.478 0.044 0.020319 416   28
## s04 Ep_FPOM 0.291 0.078 0.046165 183   19
## s05 Ep_FPOM 0.320 0.092 0.051925 489   26
## s06 Ep_FPOM 0.328 0.126 0.070242 611   25
## s25      Er 0.139 0.630 0.539509  47   19
## s08      Er 0.213 0.618 0.427528 146   17
## s09      Er 0.243 0.508 0.329023 220   23
## s07      Er 0.278 0.490 0.296715 258   26
## s20      Er 0.290 0.804 0.476675 221   20
## s21  Er_VEG 0.138 0.590 0.507082 531   28
## s18  Er_VEG 0.200 0.340 0.242733 565   31
## s11  Er_VEG 0.236 0.334 0.219511 181   21
## s12  Er_VEG 0.245 0.810 0.522477 450   24
## s15  Er_VEG 0.279 0.496 0.299809 738   30
## s24  Er_VEG 0.300 0.958 0.558432 678   31
## s10  Er_VEG 0.353 0.758 0.407331 212   19
```

21. Nechejte si vypsat jména pěti nejfrekventovanějších druhů (tedy těch s výskytem na nejvyšším počtu lokalit).

```
names(spe)[order(-colSums(spe>0))[1:5]]

## [1] "orthobum" "synosemi" "corysp."  "mictrasp" "thiegrge"
```

22. Nechejte si vypsat jména druhů seřazená podle jejich korelací s Froudeho číslem.

```
names(spe)[order(kor)]

##  [1] "micrchgr" "thiegrge" "cromussp" "cladotsp" "apsetrif" "natasp."
##  [7] "prodoliv" "polyscgr" "paraalgr" "corysp."  "tanytasp" "tanybrun"
## [13] "ablabesp" "phaepssp" "synosemi" "stembrgr" "cricbici" "thellasp"
## [19] "brilmode" "hetemarc" "nilodubi" "paratasp" "polyconv" "demisp."
## [25] "mictrasp" "orthobum" "pararufi" "brilflav" "diplcult" "eukisimi"
## [31] "rheofusc" "paracrsp" "pottgaed" "eukigrgr" "eukibrev" "orthfrig"
## [37] "nanoreag" "eukicoer" "eukimino" "parastyl" "cricannu" "critriia"
## [43] "polylagr" "orthrubi" "crictrgr" "rheotasp" "pottlong" "orthrigr"
## [49] "eukideil" "tvetbaca" "eukilobi" "cricbigr" "orththie" "tvetdive"
```

23. Zjistěte, kterých 10 druhů nejlépe koreluje s Froudeho číslem. Pozor na to, že korelace může být kladná nebo záporná, zajímají nás druhy, které nejlépe na hydraulické podmínky reagují, tedy ty s nejnižší a nejvyšší korelací (při řazení tedy použijeme absolutní hodnoty korelací).

```
names(spe)[order(-abs(kor))][1:10]

##  [1] "micrchgr" "thiegrge" "tvetdive" "cromussp" "orththie" "cricbigr"
##  [7] "eukilobi" "cladotsp" "tvetbaca" "apsetrif"
```

24. Vytvořte kopii dataframu *spe*, v níž bude 7 druhů které nejlépe reagovaly na hydraulické podmínky.

```
spe.red<- spe[, order(-abs(kor))[1:7]]
spe.red
```

```
##     micrchgr thiegrge tvetdive cromussp orththie cricbigr eukilobi
## s01       25        5        4        0        0        2        3
## s02       31       15        7        0        0        7        1
## s03       31        8        5        0        0        0        0
## s04       14       17        1        0        0        0        0
## s05        3       59        0        1        0        0        0
## s06       31       61        0        4        0        1        1
## s07        0        5        9        0        0        2        2
## s08        0        0        4        0        1        1        1
## s09        0        1        5        0        2        7        2
## s10        0        0       17        0        0        6        4
## s11        0        1       12        0        0        7        2
## s12        0        1       32        0        1       13        4
## s13      213       72        0        2        0        0        0
## s14        0        3        2        1        0        2        1
## s15        9       17       64        0        1        3       14
## s16        0        1        1        0        0        2        1
## s17        1        4        0        0        1        0        1
## s18        0        3       53        0        1        9        6
## s19       25       14        2        1        0        0        8
## s20        0       13       41        0        0        0        3
## s21        0        1       87        0        3        2        7
## s22        8       13        5        6        0        0        0
## s23        5        6        0        7        0        0        0
## s24        0        6      116        0        2        3        7
## s25        0        1        1        0        1        1        1
## s26      287      127        1        5        0        0        0
## s27      126       32        2        2        0        0        0
```

25. Seřaďte sloupce tohoto dataframu podle typu odezvy (kladná nebo záporná korelace), a uvnitř každého typu podle síly korelace (první budou ty s nejvyšší odezvou). Řádky pak seřaďte podle Froudeho čísla.

```
# V predchozim bode jsme vybrali 7 druhu s nejsilnejsi odezvou.
# Techto 7 druhu ted potrebujeme dal seradit podle jejich korelaci, a na to potrebujeme mit
# vytazene korelace jen techto 7 druhu. Vytahneme si je stejne, jako jsme vybrali ty druhy:
kor.red<- kor[order(-abs(kor))[1:7]]
# Nyni mame dataframe s abundancemi 7 vybranych druhu (spe.red)
# a vektor jejich korelaci (kor.red).
# Staci tedy seradit radky a sloupce dataframu spe.red a jsme hotovi:
spe.red.sort<- spe.red[order(env$froude), order(kor.red>0, -abs(kor.red))]
spe.red.sort
```

```
##     micrchgr thiegrge cromussp tvetdive orththie cricbigr eukilobi
```

```
## s26      287      127         5         1         0         0         0
## s27      126       32         2         2         0         0         0
## s19       25       14         1         2         0         0         8
## s04       14       17         0         1         0         0         0
## s05        3       59         1         0         0         0         0
## s13      213       72         2         0         0         0         0
## s06       31       61         4         0         0         1         1
## s23        5        6         7         0         0         0         0
## s22        8       13         6         5         0         0         0
## s17        1        4         0         0         1         0         1
## s01       25        5         0         4         0         2         3
## s03       31        8         0         5         0         0         0
## s16        0        1         0         1         0         2         1
## s02       31       15         0         7         0         7         1
## s14        0        3         1         2         0         2         1
## s11        0        1         0        12         0         7         2
## s18        0        3         0        53         1         9         6
## s07        0        5         0         9         0         2         2
## s15        9       17         0        64         1         3        14
## s09        0        1         0         5         2         7         2
## s10        0        0         0        17         0         6         4
## s08        0        0         0         4         1         1         1
## s20        0       13         0        41         0         0         3
## s21        0        1         0        87         3         2         7
## s12        0        1         0        32         1        13         4
## s25        0        1         0         1         1         1         1
## s24        0        6         0       116         2         3         7
```

26. Exportujte do excelu výsledný dataframe. Použijte funkci `write.table()`.

```
write.table(spe.red.sort, file= 'D:/My Dropbox/predmety/uvod do R/2014/cv09/spe.red.txt',
sep= '\t', col.names = NA, row.names = TRUE)
```