Základy programu R

Pojďme se nejprve rychle seznámit s prostředím R. Po instalaci a spuštění programu asi budete trochu zklamáni, nezapomeňte ale že tento program jsme nevybrali kvůli jeho vzhledu, ale pro efektivitu práce. Všechny operace budeme provádět v podokně pojmenovaném R Console, tamtéž se budou také vypisovat výsledky nebo případné chyby skriptu.

Nápovědu vyvoláme buď prostřednictvím menu programu nebo příkazem help.start() – tímto se otevře webová stránka (dostupná i bez připojení k internetu, kde asi nejužitečnější sekcí je „Search Engine & Keywords“ umožňující vyhledávání podle klíčového slova; o programu dále existuje nepřeberné množství knih i online zdrojů, zejména v angličtině.

V této kapitole jsou vedle úplných základů R užívány také některé statistické pojmy a složitější příkazy, které nemusí být plně srozumitelné, což ovšem není v daném kontextu nezbytně nutné. Nebudou zde tudíž podrobněji vysvětlovány ježto slouží pouze pro rychlou demonstraci možností programu a budou jednotlivě představeny v následujících kapitolách.

Příklad 1: zkusme spočítat počet hodin v jednom kalendářním roce.

```
> 365*24
  [1] 8760
```

Pro R je oddělovačem příkazů vždy nový řádek. Výsledek se zobrazí hned pod příkazem který jej vyvolal; protože výsledky mohou být více řádkové, jsou řádky výsledku vzestupně číslovány počínaje jedničkou.

Jakoukoli hodnotu si můžeme uložit do objetu. Přiřazení do objektů probíhá buď pomocí obvyklého rovnítka, nebo zápisem podobným šipce, který je vhodnější, neboť jasně sděluje směr přiřazování.

Příklad 2: tentýž výpočet za využití pojmenovaných objektů.

```
> dny=365
>       #text který napíšeme za křížek(hash) je komentář
> hodiny<-24
```

Počet mezer mezi symboly R nezajímá; při přiřazování se vypočítaná hodnota nezobrazuje a tak chceme-li zobrazit hodnotu objektu, zadáme jeho jméno

```
> dny * hodiny->hodinZaRok
> hodinZaRok
  [1] 8760
```

Jméno objektů je vcelku libovolná kombinace alfanumerických znaků; dodejme, že R rozlišuje velká a malá písmena.

```
> hodinZaRok
  [1] 8760
> hodinzarok
   Error: object 'hodinzarok' not found
```

Všechny dostupné objekty lze vypsat příkazem *ls()*, již nepotřebnou proměnnou zahodíme příkazem *rm(dny)*. Hlubší pohled na adresování proměnných a jejich podsložek viz kapitola xxx.

```
> ls()
  [1] "dny"        "hodiny"     "hodinZaRok"
> rm(dny
> ls()
  [1] "hodiny"     "hodinZaRok"
```

Naprostá většina objektů která chceme analyzovat je nějakým způsobem „víceprvková“, lépe řečeno jedná se o vektory popř. faktory[4]. Jasným příkladem jsou řady naměřených dat. Tyto do programu načteme dvěma různými cestami - buď přímým vložením do skriptu nebo z externího souboru. Zatím si ukažme jen první cestu, s druhou se seznámíme rovněž později.

[4] Vektor je řada numerických dat, faktor je řada ordinálních či nominálních dat

Příklad 3: definice vektoru

```
> trida8A.vek<-c(12,13,10,12,11,12,13,12,13,12,11,12)
> #zjištěný věk dětí ve třídě
> trida8A.vek
 [1] 12 13 12 10 12 11 12 13 10 12 13 12 11 12
```

Příkaz c() sestavuje z dílčích hodnot daný vektor (řadu). Obzvláště při spolupráci více autorů nebo používáteli skript R na více počítačích se vyplatí mít data přímo vložená ve skriptu právě tímto způsobem. Určitě ale nezapisujte výsledky měření přímo do R – doporučení k správě primárních data shrnuje kapitola xxx

Základní informace o datech nám poskytují funkce *lenght(data)*, *mean(data)*, *median(data)*, *max(data)*, *min(data)*, *sd(data)* - většinu z nich vypíše i sdružený zápis *summary(data)*.

```
> summary(trida8A.vek)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   10.00   11.25   12.00   11.79   12.00   13.00
```

Nyní tedy známe věk nejmladšího a nejstaršího žáka, medián, aritmetický průměr i polohu kvartilů - viz dále v textu.

Získání hodnoty modus, tedy nejčastěji naměřené hodnoty, je trochu méně přímočaré:

```
> as.numeric(names(which.max(table(trida8A.vek))))
  [1] 12
```

Upřímně řečeno nevím, proč není už v základní knihovně obsažen jednoduchý způsob zjištění modu, ale alespoň můžeme demonstrovat další výhodu R, kterou je možnost definování vlastních funkcí - od chvíle definování funkce *modus(data)* ji můžeme dále v tomto skriptu používat.

Příklad 4: definice funkce

Znaménka „+“ konzole ukazují že jsme uvnitř špičatých závorek; R bude čekat s vyhodnocením příkazů do uzavírající závorky.

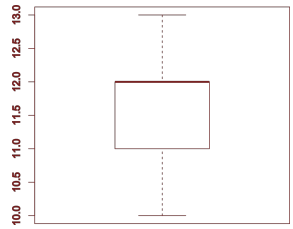
```
>#definujeme funkci
> modus <- function(data) {
+ as.numeric(names(which.max(table(data))))
+ }
>#používáme funkci
> modus(trida8A.vek)
  [1] 12
>#vypíšeme si graf pro vizuální potvrzení nejčastější hodnoty
> stem(trida8A.vek)
  The decimal point is at the |
   10 | 0
  11 | 00
  12 | 000000
  13 | 000
```

Poslední volaná funkce stem(data) vytvořila zajímavý typ „grafu“, který nám ukazuje rozložení dat (jasně teď vidíme že nejvíce máme dvanáctiletých, počet nul odpovídá počtu dětí daného věku).

GRAF JE ZÁKLADNÍ ZPŮSOB PREZENTACE NAŠICH DAT, proto si již nyní ukažme, jak snadné je vytvořit v jazyce R graf. Problematice tvorby grafů, přizpůsobování jejich nastavení, ale také významu v prezentaci dat je věnována samostatná kapitola, v této chvíli nám bude stačit zadat příkaz

```
> boxplot(trida8A.vek)
```

V novém okně určeném pro grafický výstup je vytvořen boxplot (krabicový graf) který rovněž ukazuje rozložení dat v rámci celého souboru – viz obrázek .



Obrázek 1: Silná čára udává medián, vnější zarážky určují 95% dat souboru, v boxu leží druhý a třetí kvartil dat.

Uložení skriptu

Jedním z velmi podstatných rysů práce v R je možnost uložit si veškerou práci a kdykoli se k ní vrátit či ji konzultovat s kolegou/vedoucím práce. První přístup spočívá v psaní skriptu v textovém editoru[5], stačí pak přes schránku zkopírovat do konzole(nebo načíst prostřednictvím menu) a snadno se k němu vrací pro znovupoužití potřebných částí skriptu. Je potom vhodné dodržovat určitou konvenci v uspořádání celého adresáře, například

. . . /projekt – zde je uložen skript

. . . /projekt/data – zde ukládáme primární data

. . . /projekt/output – sem generujeme grafy

Můžete také využít funkce „Save workspace“. Vytvořené soubor můžete zrovna tak poslat emailem či si uložit, obsahuje v podstatě otisk všech objektů z paměti R v okamžiku ukládání - jeho otevřením se můžete kdykoli vrátit k práci (v textovém editoru už jej ale neupravíte).

Oba postupy mají své výhody a nevýhody a je na vás který postup zvolíte, ze začátku ale prosím využívejte samostatné textové soubory – uložená workspace sice obsahuje všechny objekty, ale již nevidíte jasně jakou cestou vznikly, což není vhodné v okamžiku kdy se se systémem teprve seznamujete. Značnou část naší práce navíc bude spočívat v tvorbě grafů a příkazy které je vytvářejí budeme několikrát měnit, k čemuž je zase ukládání do workspace nevhodné.

[5] Tím nemyslím Word&spol, ale editor prostého textu jako je třeba Poznámkový blok, doporučuji Notepad++, budete-li dodržovat příponu souborů *.r, automaticky i rozpozná a zvýrazní syntaxi jazyka R.