

učební texty Univerzity Karlovy

PRACUJEME s programem R

Jiří Makovička

Pracujeme s programem R

Jiří Makovička

Recenzovali:

doc. RNDr. Karel Zvára, CSc.

Mgr. Alena Černíková, Ph.D.

Vydala Univerzita Karlova

Nakladatelství Karolinum

Praha 2025

Sazba DTP Nakladatelství Karolinum

Vydání první

© Univerzita Karlova, 2025

© Jiří Makovička, 2025

ISBN 978-80-246-6046-2

ISBN 978-80-246-6063-9 (pdf)



Univerzita Karlova
Nakladatelství Karolinum

www.karolinum.cz
ebooks@karolinum.cz

OBSAH

1. ÚVOD	9
2. SLOVO KE STUDIJNÍM PRAMENŮM	11
3. PROGRAM R, POROVNÁNÍ S PYTHONEM	12
4. INSTALACE R A RSTUDIA	13
5. INTERPRETAČNÍ A KOMPILAČNÍ PROGRAMOVACÍ JAZYKY	15
5.1 Interpretační jazyk	15
5.2 Kompilační jazyk	15
5.3 Kombinace obou přístupů	15
6. JAZYK PROGRAMU R	17
7. NÁPOVĚDA V RSTUDIU	19
8. PŘÍKAZY JAZYKA R, KONZOLE	20
8.1 Přiřazovací příkaz	20
9. BĚH PROGRAMU (SKRIPTU) V RSTUDIU	22
10. HISTORIE PŘÍKAZŮ, UKLÁDÁNÍ PRACOVNÍHO PROSTORU	23
11. FUNKCE A JEJICH ARGUMENTY	24
12. BĚŽNÉ MATEMATICKÉ FUNKCE V R – PŘÍKLADY	26
13. VÝRAZY, OPERANDY A OPERÁTORY V R	27
14. ATOMICKÉ DATOVÉ TYPY R	30
15. FUNKCE PRO PRÁCI S ATOMICKÝMI DATOVÝMI TYPY	31
15.1 Funkce typeof	31
15.2 Skupina is funkcí	31
15.3 Skupina as funkcí	32
15.4 Funkce cat	33
15.5 Funkce print	34
15.6 Funkce readline	36

16. DATOVÝ TYP VEKTOR – ZÁKLADNÍ DATOVÝ TYP	38
16.1 Funkce c	38
16.2 Funkce length	39
16.3 Další nástroje pro tvorbu vektorů	40
16.3.1 Operátor dvojtečka	40
16.3.2 Funkce seq	40
16.3.3 Indexové výrazy	41
16.3.4 Funkce vector	42
16.3.5 Vektorová aritmetika, vektorizace a recyklace	43
16.3.6 Subsetting vektoru	46
16.3.7 Změna počtu složek vektoru, přidání prvků do vektoru	47
16.3.8 Aplikace běžných funkcí na vektor	48
16.3.9 Vektor s pojmenovanými prvky – složkami	49
16.4 LETTERS	50
16.5 Třídění složek vektoru, funkce sort a order	51
16.6 Funkce which	52
16.7 Funkce which.min a which.max	53
16.8 Faktory	53
17. MATICE (DVOJROZMĚRNÁ POLE)	57
17.1 Parametr dimnames funkce matrix a funkce dimnames	59
17.2 Funkce dim	62
17.3 Matice z vektorů či matic – funkce cbind, rbind	64
17.4 Matice – subsetting	66
17.5 Funkce apply – aplikuje zvolenou funkci na řádky či sloupce matice	67
17.6 Matice – funkce t	68
17.7 Funkce summary	68
17.8 Diagonální matice	69
17.9 Jednotková matice	70
17.10 Násobení matic	71
17.11 Inverzní matice	72
17.12 Soustavy lineárních algebraických rovnic	73
17.13 Funkce solve	75
18. DATASET (DATA FRAME)	77
18.1 Dataset – důležité funkce	78
18.2 Funkce data	78
18.3 Funkce View	78
18.4 Dataset – subsetting	79
18.5 Odstranění řádku či sloupce v datasetu	82
18.6 Přidání řádku do datasetu	82
18.6.1 Přidání nového řádku funkcí rbind	82
18.6.2 Přidání nového řádku technikou indexace	83
18.7 Přidání sloupce do datasetu	83
18.7.1 Přidání nového sloupce referencí na jeho jméno	83
18.7.2 Přidání nového sloupce funkcí cbind	84
18.8 Funkce transform	84
18.9 Editace položky datasetu	85
18.10 Souhrnné výpočty se sloupci v datasetu	85
18.10.1 Funkce tapply	85
18.10.2 Funkce sapply	86
18.11 Jména řádků a sloupců datasetu	87
18.12 Třídění datasetu podle určitého sloupce	88
18.13 Odstranění duplicit	89
18.14 Funkce table a prop.table	90
18.15 Funkce sample	93

19. DATOVÝ TYP LIST	96
19.1 Práce se jmény objektů listu	98
19.2 Přidání a odstranění položek listu	98
19.3 Funkce attach, detach	102
19.4 *Funkce with	106
19.5 Funkce aggregate	109
20. ZVLÁŠTNÍ OBJEKTY V R – INF, NULL, NA, NAN	111
20.1 Inf	111
20.2 NULL	111
20.3 NA	112
20.4 NaN	112
21. PRÁCE SE SOUBORY (ČTENÍ A UKLÁDÁNÍ DAT)	114
21.1 Textové soubory typu csv	114
21.1.1 Funkce write.csv	115
21.1.2 Funkce read.csv	115
21.2 Uložení a čtení objektu binárně	116
21.3 Funkce file.choose	117
21.4 Čtení excelovských dat	118
21.5 Jak RStudio pomáhá při načítání datových souborů	118
22. TVORBA JEDNODUCHÝCH GRAFŮ V R	120
22.1 Některé důležité parametry grafických funkcí	121
22.2 Nastavení a získání grafických parametrů funkcí par	124
22.3 Zadání datasetu jako parametru funkce plot	126
22.4 Legenda obrázků	127
22.5 Funkce title, vzdálenost titulku a popisů os od grafu	129
22.6 Udávání barev v grafických funkcích plot, points, line, funkce palette	131
22.7 Prázdné okno pro graf	133
23. ELEMENTY PROGRAMOVÁNÍ V R	134
23.1 Příkazové závorky – složený příkaz, blok příkazů	134
23.2 Podmíněné příkazy	135
23.2.1 Příkaz if	135
23.2.2 Příkaz if else	137
23.2.3 Příkaz if else if	138
23.2.4 Funkce ifelse	140
23.3 Cykly	141
23.3.1 Cyklus while	141
23.3.2 Cyklus repeat	142
23.3.3 Repeat versus while	144
23.3.4 Cyklus for	145
24. FUNKCE DEFINOVANÉ UŽIVATELEM	149
24.1 Poziční volání funkce	151
24.2 Volání s pojmenovanými argumenty	151
24.3 Počáteční hodnoty parametrů	153
24.4 *Anonymní funkce	155
24.5 *Lokální a globální data funkce	157
24.5.1 *Prostředí funkce with	160
24.5.2 *Má pracovat funkce s globálními proměnnými?	162
24.6 *Parametr tři tečky (dot-dot-dot)	163
24.7 *Funkce vyššího řádu (higher order functions)	165
24.8 *Uzávěr (closure)	166

25. ŘEŠENÍ NĚKTERÝCH ÚLOH NUMERICKÉ MATEMATIKY V PROGRAMU R	170
25.1 Analytické a numerické řešení	170
25.2 Základy práce s knihovnamí a balíčky programu R	170
25.3 Numerické derivování	174
25.3.1 Funkce grad	176
25.4 Numerické integrování	176
25.4.1 Obdélníková metoda	176
25.4.2 Funkce integrate	179
25.4.3 Integrace metodou Monte Carlo	180
25.4.4 Symbolické výpočty v R	181
25.5 Numerické řešení rovnice $f(x) = 0$	183
25.5.1 Obecný postup řešení	183
25.5.2 Newtonova metoda tečen k řešení rovnice $f(x) = 0$	183
25.5.3 Řešení rovnic typu $f(x) = 0$ v programu R	187
25.6 Numerické řešení obyčejných diferenciálních rovnic 1. řádu	190
25.6.1 Eulerova metoda	191
25.6.2 Exponenciální růst	192
25.6.3 Exponenciální pokles	193
25.6.4 Jak na diferenciální rovnice v R	194
25.6.5 Funkce ode45 a integrateODE	194
26. LINEÁRNÍ REGRESE METODOU NEJMENŠÍCH ČTVERCŮ	203
26.1 Regresní funkce – příklady	204
26.2 Funkce lm	204
26.3 Základní typy lineárních regresních modelů pro funkci lm	205
26.4 Formule jako objekt R	206
26.5 Funkce summary	207
26.5.1 Výpis funkce summary	211
26.6 Predikce	214
27. NELINEÁRNÍ REGRESE METODOU NEJMENŠÍCH ČTVERCŮ	217
27.1 Funkce nls	217
27.2 Důležité funkce pracující s objektem navráceným funkcí nls	218
27.3 Příklady nelineární regrese	218
28. SEZNAM OBRÁZKŮ	226
29. BIBLIOGRAFIE	227
30. REJSTŘÍK	228

1 ÚVOD

Předkládaný učební text je určen pro výuku předmětu Výpočetní technika, Výpočetní technika pro chemiky a Výpočetní technika (pro geologické obory) na Přírodovědecké fakultě Univerzity Karlovy. Popisuje program R a práci s ním s pomocí RStudio.

Jedním z důvodů, proč tato skripta o R vznikla, bylo seznámit studenty před absolvováním přednášek ze statistiky s programem R, který se při její výuce používá. Aplikace R ve statistice není proto obsahem těchto skript. Studenti přírodovědných oborů by ale měli být také seznámeni se základy programování a numerické matematiky, což není obsaženo v běžných učebních plánech. Proto byly do skript zahrnuty i tyto kapitoly.

Velké poděkování patří oběma recenzentům, doc. RNDr. Karlu Zvárovi, CSc. a Mgr. Aleně Černíkové, Ph.D., kteří pozorně pročetli předběžný text skript, upozornili na jeho nedostatky a předali mi náměty k jeho zlepšení. Docent Zvára je na slovo vzatý odborník na regresi, zkušený autor a R používá v praxi. Doktorka Černíková vyučuje statistiku s pomocí programu R a své zkušenosti z výuky vtělila do mnou respektovaných připomínek ke skriptům.

Výklad lze rozdělit do čtyř oblastí:

- 1) Seznámení s programem R a jeho datové struktury:
Popíšeme základní vlastnosti a funkce R, datové struktury vektor, matice, dataset a seznam. Dále se budeme věnovat ukládání dat do souboru a čtení ze souborů do paměti.
- 2) Programování v jazyku R:
Uvedeme základní programové konstrukce jazyka, tj. podmíněné příkazy, cykly a funkce definované uživatelem.
- 3) Základní metody numerické matematiky s použitím programu R:
Programování v R budeme aplikovat na řešení jednoduchých úloh z numerické matematiky. Souběžně si uvedeme funkce R, které lze přímo na řešení úlohy použít.
- 4) Lineární a nelineární regrese v R:
Přírodovědec přijde s touto tematikou často do styku. Proto mu dáváme k dispozici nástroje pro řešení regresních úloh, které má program R k dispozici.

Ve výkladu budeme předpokládat, že program R běží v prostředí operačního systému Windows.

Program R obsahuje integrované prostředí pro práci s ním (IDE, *integrated development environment*). Je však výhodnější používat místo toho nějakou pokročilejší nadstavbu. V tomto učebním textu se orientujeme na program RStudio.

Funkce jazyka R mají často velké množství parametrů. Kdybychom uváděli všechny, stal by se výklad značně nepřehledný. Proto budeme diskutovat jen parametry často používané, které jsou třeba při řešení běžných úloh. Syntax funkcí, v nichž uvedeme pouze tyto důležité parametry, budeme avizovat nadpisem „Zjednodušená syntax“. Samotná syntax bude formátovaná tučně.

Příklady v textu budou uvozeny slovem *Příklad*. Konec příkladu označíme čtyřmi pomlčkami.

Poznámky budou uvozeny slovem *Poznámka* a odlišeny od ostatního textu kurzívou.

Názvy odstavců obsahujících pokročilejší látku, nebo látku jejíž znalost není bezprostředně nutná pro další studium těchto skript, začínají znakem hvězdička uvedeným v horním indexu.

2 SLOVO KE STUDIJNÍM PRAMENŮM

Jakou literaturu doporučujeme?

Systematický výklad práce s programem R najdeme v knize *R For Dummies* [1]. Kniha je zaměřena, kromě obecných kapitol, na zpracování dat v R, najdeme zde však i kapitoly o tvorbě skriptů. Je určena začátečníkům a pokročilejším uživatelům.

Programování v R se věnuje knížka *THE ART OF R PROGRAMMING* [2]. Seznamuje čtenáře s datovými typy R, dále s programovými strukturami jazyka R, včetně objektově orientovaného programování, prací s grafikou, a dotkne se i paralelního programování.

Velice široký záběr má dílo *R Cookbook* [3], které nezapře svůj název. Kniha je plná receptů, jak řešit konkrétní problémy v R. Najde v ní odpovědi na své otázky jak začátečník, tak i pokročilejší uživatel.

Zájemci o hlubší studium R, a pochopení jazyka, najdou četné poklady v knížce Hadley Wickhama, *Advanced R* [4]. Autor je odborníkem v R na slovo vzatým, tvůrcem pozoruhodných a široce používaných balíčků a též napsal několik snadno pochopitelných učebnic.

V knize *R for Data Science* [5] najde čtenář vše, co potřebuje pro zpracování dat pomocí programu R. Kniha je napsána velice srozumitelným způsobem zkušenými autory, Jedním z nich je opět Hadley Wickham. Její webová verze je na adrese <https://r4ds.hadley.nz/>.

Kniha *ZÁKLADY STATISTIKY V PROSTŘEDÍ R* [6] od Karla Zváry používá R na poli statistiky, což je vědní obor, pro který tento jazyk primárně vznikl. Kniha je zdařilou ukázkou kompozice výkladu statistiky a práce s jazykem R při řešení příkladů ilustrujících probíranou látku.

Hlubokou studnicí vědění plnou článků, rad a řešených příkladů je samozřejmě internet. Stačí do adresního řádku prohlížeče napsat R a potom název funkce, či jiné klíčové slovo, a téměř vždy jste spokojeni. Hledané heslo „R tutorial“ odkryje velké množství seminářů, z nichž si vybere snad každý. Začátečníkům doporučuji seminář na adrese <https://www.w3schools.com/r/>.

Nejpřesnější informace najdete v dokumentaci k R na adrese <https://www.rdocumentation.org/>. Hledáme-li přímo informace k určité funkci doporučuji heslo „R Documentation“ následované názvem funkce. Je však nutné si zvyknout na velice stručnou formu výkladu a často výklad doplnit hledáním dalších pramenů.

3 PROGRAM R, POROVNÁNÍ S PYTHONEM

R je program obsahující interpretační (viz odstavec 5) programovací jazyk, který také nazýváme R. Tento nástroj patří mezi vyšší programovací jazyky. Program R byl původně vyvinut pro statistické výpočty a tvorbu grafů, ale byla sestavena celá řada knihoven obsahujících kvalitní algoritmy z lineární algebry, numerické matematiky, zásobu programů pro práci s daty a samozřejmě i algoritmy z umělé inteligence. V R jsou také balíčky funkcí `rmarkdown` a `shiny` sloužící pro webovou prezentaci výsledků.

Program R se hodí spíše pro zpracování dat a může se stát se svým velkým množstvím knihoven příručním programem nejen pro první vyhodnocení dat, ale i pro jejich následné zpracování. Výhodou R ve srovnání s Pythonem je, že můžeme, s pomocí `RStudio`, účinněji skloubit interaktivní práci na konzoli s tvorbou a spouštěním skriptů. Balíčky R poskytují, v některých ohledech, více nástrojů než Python. Funkcionální podstata R též umožní získat výsledky i složitých procedur snadněji. Pro statistické vyhodnocování dat má R nesporné přednosti před Pythonem.

Python je obecný interpretační jazyk. Proto může být použit k řešení širšího rozsahu úloh než R, jako je tvorba webových aplikací, práce v počítačové síti, programování her apod. Tvorba programů v Pythonu je snadnější než v R, díky jeho moderní syntaxi. Často se říká, že programovat v Pythonu je snadné. To je ale pravdou jen pro tvorbu jednoduchých programů. Pokud chceme napsat složitější program, alespoň na poloprofesionální úrovni, je programování v Pythonu srovnatelné s ostatními používanými programovacími jazyky.

Porovnání těchto dvou jazyků je velice dobře zpracováno v citaci <https://www.datacamp.com/blog/python-vs-r-for-data-science-whats-the-difference>.

4 INSTALACE R A RSTUDIA

Oba dva programy jsou snadno dostupné na internetu a jejich instalace je velice jednoduchá.

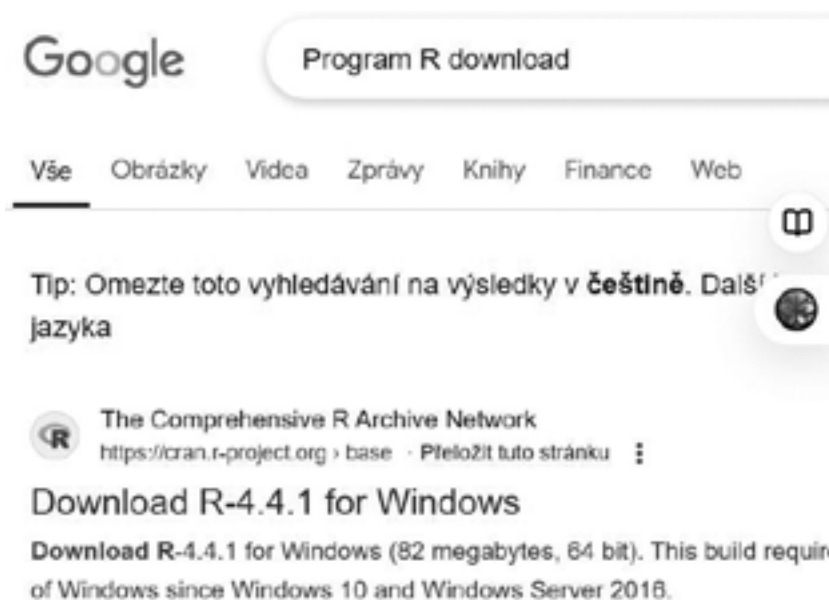
Nejdříve instalujeme program R a potom RStudio.

Do adresního řádku vyhledávače (browseru) napíšeme **Program R download** a bereme první nabídnutou možnost (obr. 1). Z adresy

<https://cran.r-project.org/bin/windows/base/>

stáhneme a nainstalujeme poslední verzi programu R, např. **R-4.4.1-win.exe**. Tento soubor spustíme a řídíme se jednoduchými pokyny pro instalaci R.

Při instalaci RStudio postupujeme obdobně (obr. 2). Do adresního řádku vyhledávače (browseru) napíšeme **RStudio download** a bereme první nabídnutou možnost. Z adresy <https://posit.co/download/rstudio-desktop/> stáhneme a nainstalujeme poslední verzi RStudio pro Windows (obr. 3).



Obr. 1: Stažení programu R

Google

RStudio download

Vše Video Obrázky Zprávy Knihy Web Finance



Posit

<https://posit.co> > download > rstu... · Přeložit tuto stránku

RStudio Desktop

Get started with **RStudio** on Posit Cloud for free . If you're a professional to **download RStudio** and also need common enterprise features, ...

Obr. 2: Stažení programu RStudio



<https://posit.co/download/rstudio-desktop/>



Home



Google Chrome The...



WEB



JavaScript



Nodejs express



2: Install RStudio

DOWNLOAD RSTUDIO DESKTOP FOR WINDOWS

Obr. 3: RStudio pro Windows

5 INTERPRETAČNÍ A KOMPILAČNÍ PROGRAMOVACÍ JAZYKY

Z hlediska překladu zdrojového kódu rozeznáváme dva základní typy jazyků, a to interpretační a kompilačních jazyky.

5.1 INTERPRETAČNÍ JAZYK

Při každém běhu programu, napsaného v klasickém interpretačním jazyku, načítá překladač – interpret – příkazy¹ zdrojového kódu programu jeden po druhém, převádí je na soubor instrukcí strojového kódu² a tyto instrukce procesor počítače ihned provádí. Překlad je tedy u těchto jazyků součástí běhu programu, děje se při každém běhu, tedy opakovaně. Z toho vyplývá, že běh programu v interpretačním jazyku je pomalý. Mezi interpretační jazyky patří např. rané verze Basicu nebo Python. Program napsaný v interpretačním jazyku často nazýváme **skript**. Doporučujeme přečíst si definici „skriptovací jazyk“ na Wikipedii.

5.2 KOMPILAČNÍ JAZYK

Překlad programu se děje jako samostatná operace před jeho během. Vzniklý přeložený modul (soubor zpravidla s příponou exe), obsahující strojové instrukce, pak opakovaně spouštíme. Ve srovnání s interpretačními jazyky mají kompilační jazyky vysokou rychlost programu ale horší ladění. Patří sem např. jazyky FORTRAN, C, C++.

5.3 KOMBINACE OBOU PŘÍSTUPŮ

Překlad interpretačních jazyků se však neustále zdokonaloval. Zásadní obrat znamenal překlad zdrojového kódu do tzv. byte kódu, který se poté interpretuje virtuálním strojem (virtual machine) každého jazyka. Takovýto způsob překladu nacházíme např. u jazyků Java, Visual

1 **Příkaz** – nejmenší jednotka programu kódující nějakou činnost. Program se vykonává po příkazech.

2 **Strojový kód** – Jediný programovací jazyk, kterému procesor počítače rozumí – jehož příkazy umí vykonat. Proto se programy napsané v ostatních (tzv. vyšších) programovacích jazycích musí před jejich spuštěním přeložit do strojového kódu, který je binární.

Basic .Net, C# , Python. Dochází tedy ke kombinaci kompilační způsobu – vznik byte kódu – a poté k interpretaci byte kódu virtuálním strojem jazyka. Tyto jazyky bychom mohli nazvat kompilačně interpretační.

Do této skupiny patří i náš programovací jazyk R. Všechny jeho základní funkce jsou již přeloženy do byte kódu. Některé funkce, nebo jejich části, byly navíc programovány přímo v kompilačních jazycích, jako C a FORTRAN. Ve Windows jsou běžně instalované balíčky R z úložiště CRAN již v binární formě. Tento přístup obrovsky zrychluje běh kódu jazyka R.

Překlad do byte kódu i použití funkcí napsaných v kompilačních jazycích (např. C či C++) jsou prostřednictvím knihoven R přístupné i běžnému uživateli – programátorovi v R – při tvorbě vlastních programů.

Napíšeme-li však v R skript a spustíme jej, je interpretován, ale volané knihovní funkce jsou již v byte kódu či byly napsány celé, nebo jejich části ve zmíněných kompilačních jazycích (a jsou tedy přeloženy do binární formy).

6 JAZYK PROGRAMU R

R je ve své nekonvenčnosti krásný.

Vysvětlíme nejdříve zjednodušeně podstatu imperativního a funkcionálního programování.

Imperativní programování – program je souhrn příkazů přesně popisujících, jak se dostaneme k výsledku. Takto se programuje v klasických jazycích, jako FORTRAN, C apod.

Funkcionální programování – víme, jaký chceme výsledek a za tím účelem voláme vhodnou funkci jazyka, které zadáme žádané argumenty. Tato funkce navrátí objekt obsahující řešení. Jazyk programu R patří mezi funkcionální programovací jazyky.

V jazyce R pracujeme s **objekty**, v nichž jsou ukládána naše data. Tyto objekty zadáváme jako argumenty funkcím jazyka a funkce nám navrací opět objekty s výsledky. Objekty musíme ale v paměti nějak identifikovat. K tomu používáme jméno neboli identifikátor objektu. Místo v paměti obsahující objekt a označené jménem, budeme nazývat **proměnná**. Proměnná má tedy dvě součásti. Jméno a místo v paměti určené pro ukládání našich dat ve formě objektů.

Pro tvorbu platných jmen však platí v R, jako v každém programovacím jazyku, striktní pravidla. Jak tedy tvoříme platná jména proměnných (symboly, identifikátory) v R?

Jméno proměnné může obsahovat písmena, číslice, tečky a podtržítka. Podtržítkem a číslicí však nesmí jméno začínat. Ve jménech proměnných rozlišujeme malá a velká písmena (case sensitivity).

Kuriozitou R je tečka na začátku jména proměnné. Takováto jména nejsou zobrazována kartou s výpisem prostředí (Environment) RStudio. Jsou tedy (do jisté míry) skrytá, nezobrazuje je ani funkce `ls`. Teprve nastavíme-li argument této funkce nazvaný `all.names`, na hodnotu `TRUE`, jsou skrytá jména zobrazena. Začíná-li jméno tečkou, nesmí potom následovat číslice (.3 není platné jméno). Jména proměnných začínajících tečkou se používají např. k určení konfigurace prostředí programu R. My taková jména dělat nebudeme.

Klíčová slova R (`if`, `else`, `for` apod.) též nesmíme použít jako jména proměnných. Seznam klíčových slov získáme vykonáním konzolového příkazu `?reserved`.

Poznámka:

Pomocí klíčových slov děláme platné příkazy programu R.

Další kuriozita: Uzavřeme-li jakoukoli posloupnost znaků do zpětných apostrofů (back-quotes), je to platné jméno. Tedy ``-`` a také ``_a`` jsou platná jména proměnných.

Doporučení pro tvorbu jmen proměnných: Vytvářejme platná jména tak, že začínají písmenem anglické abecedy a potom následují písmena nebo číslice. Vyplatí se používat popisná jména (koncMg pro koncentraci hořčíku, rychlostReakce pro reakční rychlost apod.).

Neplatná jména proměnných: “ako” (uvozovky nelze použít), 5Korun (číslice na začátku), _Pokus (podtržítka na začátku).

Tento odstavec uzavřeme citátem Johna Chamberse (vyvinul jazyk S – předchůdce R – a je jednou z předních osobností projektu **R programming language**).

“To understand computations in R, two slogans are helpful:

Everything that exists is an **object**.

Everything that happens is a **function call**.”

Příklad: Operátory v R jsou také funkce.

Sčítáme-li v R dvě proměnné **a** a **b**, realizujeme to jako příkaz **a+b**. Na první pohled není v tomto příkaze žádná funkce. Jak je to tedy s funkcionální podstatou R? Tento příkaz se totiž interně provádí jako volání funkce pro sčítání, která má jméno ``+``. V tomto zápisu pozor, znaménko `+` je uzavřeno ve zpětných apostrofech, což z něho činí platné jméno v R. Toto si lze ověřit i na konzoli RStudio. Vykonejme příkazy (viz níže):

```
> a <- 5
> b <- 6
> `+`(a, b)
[1] 11
```

Proměnná **a** má hodnotu 5 a proměnná **b** hodnotu 6, což jsme zajistili pomocí operátoru `<-` (šipka). Viz dále.

Naši funkci se zvláštním jménem ``+`` jsme volali se dvěma argumenty **a** a **b**. Podobně se v R realizuje práce s dalšími operátory.

7 NÁPOVĚDA V RSTUDIU

V okně RStudia, které leží implicitně v pravém dolním rohu, je několik karet. Mezi nimi i karta nazvaná Help. V horním panelu má okno, do něhož lze zapsat heslo a odeslat k získání nápovědy klávesou Enter.

Další možnost je funkce `help`, které dáme jako argument heslo, k němuž chceme získat nápovědu. Analogicky lze použít operátor `?` působící na hledané heslo. Např. `help(matrix)` či `?list`. Nápověda se v RStudiosu opět objeví na kartě Help. Úplné možnosti nápovědy R lze nalézt na stránce <https://www.r-project.org/help.html>.

Příkazem `Alt + Shift + K` získáme okno s klávesovými zkratkami RStudia.

Často se vyplatí, jak jsme již uvedli, vykročit za meze RStudia a hledat nápovědu na internetu jako zadaný text „R heslo“. Takto získáme nejen dokumentaci R k zadanému heslu, ale i spoustu odkazů na stránky, kde se o této problematice píše.

8 PŘÍKAZY JAZYKA R, KONZOLE

Při interaktivní práci v RStudiosu používáme konzoli. Je to okno, obvykle vlevo dole, s nápovědou `>` (znak „větší“) v příkazovém řádku, na který píšeme příkazy a pak je odesíláme klávesou Enter k vykonání.

Poznámka:

Zapíšeme-li na konzoli příkaz a odešleme jej klávesou Enter, může se na dalším řádku konzole objevit znaménko plus. Je známkou toho, že jsme příkaz nedokončili (něco v něm ještě chybí). Můžeme jej na řádku uvozeném znaménkem + dokončit, nebo se znaménka zbavit klávesou Esc a napsat příkaz znovu.

Programy psané v R a jejich vstupy a výstupy budou v tomto textu psány neproporcionálním fontem Courier New. Neproporcionální font má pevnou šířku znaků, což je zejména důležité pro výstupy v tabulkové formě. Texty jednotlivých sloupců tabulek jsou pak pod sebou.

Dávku příkazů jazyka R lze zapisovat do textového souboru jako skript R a ten potom vykonat. Tento soubor má, po uložení, implicitně příponu R. Na jeho provedení však nestačí klávesa Enter v průzkumníku Windows (jako u souboru s příponou exe), neboť skripty R nejsou spustitelné soubory ve Windows. Skript musíme spouštět v nějakém mateřském prostředí s interpretem R, v samotném programu R nebo v RStudiosu, čemuž dáme přednost.

8.1 PŘÍŘAZOVACÍ PŘÍKAZ

Přiřazení dat proměnné děláme pomocí symbolů šipek (`<-`, `->`), rovnítko (`=`) a funkce `assign`. Doporučení: užívejme `<-`. Klávesová zkratka pro tento symbol je **Alt** – (klávesa levý Alt a znak mínus, platí v RStudiosu, ne v R). V odstavci 24.2 je uveden příklad, který podporuje naše doporučení.

Syntax přiřazovacího příkazu, kterým do proměnné vkládáme hodnotu (data) je potom

jmenoprom `<-` **hodnota**

kde **jmenoprom** je platné jméno proměnné, **hodnota** je platný **výraz** jazyka R (o výrazech viz dále).

Vložíme-li přiřazovací příkaz do kulatých závorek, je navíc zobrazena přiřazená hodnota.

Příklad: Konzole RStudio jako kalkulačka.

```
> x <- 10 # příkaz
> y <- 20
> x+y
[1] 30 # výstup výsledku předchozího příkazu
> x/y
[1] 0.5
> log(x) # přirozený logaritmus deseti
[1] 2.302585
> log10(x) # dekadický logaritmus deseti
[1] 1
> (a <- 89 + 11) # zobrazí se přiřazená hodnota
[1] 100
```

V hranatých závorkách výstupu je číslo první položky na řádku. Tuto vlastnost oceníme při výpisu hodnot složitějších datových typů, jako vektor, matice apod. Za znakem # je komentář, viz dále.

9 BĚH PROGRAMU (SKRIPTU) V RSTUDIU

Skript vytváříme příkazem **File/New File/R Script**. Na kartu, která je potom k dispozici, napíšeme příkazy jazyka R. Zakódovaný skript lze potom v prostředí RStudio vykonat. Při běhu skriptu se může zobrazovat i echo, tj. text realizovaného příkazu, který předchází jeho výstupu.

Tlačítkem **Run** vykonáme příkazy po řádcích či předem vyznačených blocích s echem. Rozbalovacím tlačítkem **Source** lze vykonat celý skript bez echa či s echem. Vyzkoušejte. Skript s echem lze též vykonat klávesovou zkratkou Ctrl+Alt+R, bez echa zkratkou Ctrl+Shift+S. Po vyznačených blocích jej s echem též vykonáme za použití zkratky Ctrl+Enter.

Jaké vlastnosti jazyka R jsou důležité při tvorbě skriptu: Do příkazového řádku lze zapsat více příkazů oddělených středníkem. Znaky zapsané za symbolem # slouží jako komentář. Komentáře často použijeme u příkazů skriptu a u konzolových příkazů pro jejich vysvětlení. Programátor píše komentáře i u programů, které udělal sám, resp. pro sebe. Důvod je jednoduchý, zapomnětlivost.

Skript lze příkazem **File/Save**, či tlačítkem **Save** uložit do souboru, přípona R. Příkazem **File/Open File** lze opět uložený skript otevřít v prostředí RStudio. Užitečný je příkaz **File/Recent Files**, který ve své nabídce zobrazí jména nedávno otevřených souborů se skripty.

Příklad skriptu:

```
a <- 5 ; b <- 10 ; c <- a + b
# definujeme proměnné a = 5, b = 10, c = a + b
ls() # voláním funkce ls vypisujeme seznam proměnných
# prostředí, prázdné závorky značí, že funkce ls nemá
# při tomto volání žádné argumenty. Blíže viz nápověda.
```

Skript má na prvním řádku tři příkazy. Na druhém řádku je komentář. Na třetím řádku je příkaz a komentář. Na dalších řádcích je komentář. Napište výše uvedený příklad jednoduchého skriptu na kartu pro skript a vyzkoušejte ho. Po běhu skriptu uvidíte na kartě Environment proměnné a, b, c s jejich hodnotami.

10 HISTORIE PŘÍKAZŮ, UKLÁDÁNÍ PRACOVNÍHO PROSTORU

V paměti konzole R je uchovávána historie příkazů, podobně jako u příkazového řádku Windows. V historii vydaných příkazů lze listovat šípkami (nahoru a dolů). Historie příkazů je, obvykle automaticky při končení programu, ukládána do souboru `.Rhistory`.

Při končení RStudio se objeví dotaz na uložení pracovního prostoru Workspace (obsahuje data programu R uložená v paměti a zobrazená na kartě Environment RStudio) do souboru `.Rdata`. Oba soubory `.Rhistory` a `.Rdata` se nacházejí v pracovním adresáři, obvykle `C:/Users/<user>/Documents`, kde `<user>` je jméno konkrétního uživatele.

Vypnout/zapnout ukládání položek do souborů `.Rhistory` a `.Rdata` lze v dialogovém okně příkazu **Tools/Global Options** RStudio. Implicitně jsou obě volby zapnuté.