

edice

začínáme s ...

Začínáme programovat v jazyku PYTHON

2. přepracované a rozšířené vydání

- » Nepředpokládá předchozí znalosti programování
- » Po přečtení prvních dvou částí můžete začít vyvíjet jednoduché programy
- » Získané znalosti si procvičíte na příkladu vývoje netriviální aplikace
- » Kromě kódování v Pythonu se naučíte programy navrhovat a postupně vyvíjet
- » Naučíte se programovat podle moderních zásad a metodik

← soubory ke stažení na www.grada.cz



edice
začínáme s ...

Začínáme programovat v jazyku

PYTHON

2. přepracované a rozšířené vydání

RUDOLF PECINOVSKÝ



GRADA Publishing

Upozornění pro čtenáře a uživatele této knihy

Všechna práva vyhrazena. Žádná část této tištěné či elektronické knihy nesmí být reprodukována a šířena v papírové, elektronické či jiné podobě bez předchozího písemného souhlasu nakladatele.

Neoprávněné užití této knihy bude **trestně stíháno**.

Rudolf Pecinovský

Začínáme programovat v jazyku Python

Druhé, přepracované a rozšířené vydání

Vydala Grada Publishing, a.s.

U Průhonu 22, Praha 7

obchod@grada.cz, www.grada.cz

tel.: +420 234 264 401

jako svou 8412. publikaci

Odpovědný redaktor: Petr Somogyi

Fotografie na obálce Depositphotos/novotnyfi

Grafická úprava a sazba Rudolf Pecinovský

Počet stran 320

Druhé, přepracované a rozšířené vydání, Praha 2022

Vytiskla tiskárna PBtisk a.s. Příbram

© Grada Publishing, a.s., 2022

Cover Design © Grada Publishing, a. s., 2022

Cover Photo © Depositphotos/novotnyfi

Názvy produktů, firem apod. použité v knize mohou být ochrannými známkami nebo registrovanými ochrannými známkami příslušných vlastníků.

ISBN 978-80-271-4752-6 (ePub)

ISBN 978-80-271-4751-9 (pdf)

ISBN 978-80-271-3609-4 (print)

*Mé ženě Jarušce a dětem
Štěpánce, Pavlínce, Ivance a Michalovi*

Stručný obsah

Poděkování	18
Úvod	19
Část A Superzáklady	25
1 Počítače a jejich programy	26
2 Vývojová prostředí	36
3 Zadávání hodnot a proměnné	44
4 Používání funkcí a objektů	58
5 Moduly a práce s nimi	69
Část B Začínáme programovat	81
6 Definice funkcí	82
7 Balíčky, knihovny, robot Karel a jeho svět	94
8 Rozhodování	104
9 Opakování kódu, cykly	118
10 Kontejnery	129
11 Práce s kontejnery	138
12 Ošetřování chyb	149
Část C Základy OOP	159
13 Úvod do OOP	160
14 Dědění	174
15 Co ještě můžete potřebovat	191
16 Vytváříme balíčky a aplikace	204

Část D	Vývoj aplikace	213
<hr/>		
17	Základy objektové architektury	214
18	Návrh základní architektury	225
19	Připravujeme test	235
20	Definice testu a start hry	245
21	Svět hry a první akce	258
22	Co ještě chybí	273
23	Spustitelná aplikace	286
24	Vylepšujeme aplikaci	292
25	Kudy dál	306
Literatura		310
<hr/>		
Rejstřík		312
<hr/>		
Část E	Přílohy a seznamy	321
<hr/>		

Podrobný obsah

Poděkování	18
Úvod	19
Komu je kniha určena	19
Koncepte výkladu a jeho uspořádání	20
Potřebné vybavení	21
Použité typografické konvence	22
Odbočka – podšeděný blok	23
Zpětná vazba	24
Část A Superzáklady	25
1 Počítače a jejich programy	26
1.1 Hardware a software	26
První počítače	26
Co je to program	27
Syntaxe – sémantika – paradigma	27
Změny přístupu k tvorbě programů	28
Důležitost čitelnosti programu	29
1.2 Překladače, interprety, platformy	29
Operační systém	29
Platforma	30
Programovací jazyky	30
1.3 Platforma Python	32
Skripty	32
Dokumentace	33
1.4 Interpret spuštěný z příkazového řádku	33
Odsazování	35
1.5 Zdrojové kódy	35
2 Vývojová prostředí	36
2.1 Vývojové prostředí	36
2.2 Prostředí IDLE	37
Spuštění	37
Základní popis	37
Příkazové okno	39
Opětne zadání dříve zadaných příkazů	39
Restart interaktivního systému	39
Uložení záznamu seance	40
Editační okno	40
Umístění editovaných souborů	41
Barevné zvýraznění textu	41
Použité písmo	41
2.3 IDLE versus příkazový řádek	42
Zobrazování výpisů programů	42

2.4	Zdrojové kódy	43
3	Zadávání hodnot a proměnné	44
3.1	Počáteční mezery	44
	Komentáře	45
3.2	Celá čísla	45
3.3	Reálná čísla	46
3.4	Další možné zápisy čísel	46
3.5	Textové řetězce – stringy	47
	Znak # ve stringu	47
	Víceřádkové stringy	48
	Escape sekvence	49
	Bílé znaky	50
3.6	Proměnné a přiřazovací příkaz	50
	Identifikátor	50
	Konvence pro podobu identifikátorů	51
	Definice a použití proměnné, přiřazovací příkaz	52
	Zadání skupiny hodnot	53
	N-tice hodnot	54
3.7	Hodnota versus odkaz na hodnotu	54
	Halda a správa paměti	54
	Terminologie	55
	Nebezpečné změny hodnot	55
3.8	Literály	56
3.9	DRY – bez kopií	56
3.10	Zdrojové kódy	57
4	Používání funkcí a objektů	58
4.1	Volání funkcí	58
	Příklady funkcí	58
	Parametr versus argument	59
4.2	Hodnota None	60
	Podrobnosti o volání funkcí	60
4.3	Objekt výpustka –	61
4.4	Základní aritmetické operace	61
4.5	Formátovací stringy – f-stringy	62
4.6	Výrazy, příkazy, výrazové příkazy	62
	Proměnná	63
	Přiřazovací výraz	63
	Více příkazů na řádku versus více výrazů na řádku	63
4.7	Složený přiřazovací příkaz	64
4.8	Zadání údajů z klávesnice	65
4.9	Základy práce s objekty	66
	Vše je objekt	66
	Třída – instance – typ	66
	Vytváření objektů	67
	Atributy objektů a jejich kvalifikace	67
4.10	Zdrojové kódy	68
5	Moduly a práce s nimi	69
5.1	Moduly – základní informace	69
	Vše je součástí nějakého modulu	69
	Dva názvy objektů	70
	Zdrojový soubor	70
	Přeložený soubor	71
5.2	Příkaz import	71
	Import je jen jiný druh přiřazení	71

	Čistý import jiného modulu	71
5.3	Import modulu pod jiným názvem	73
	Přímý import vyjmenovaných objektů	74
5.4	Vytvoření vlastního modulu	75
	Název modulu	76
	Kódová stránka	76
	Dokumentační komentář	77
	Import ladícího modulu a kontrolní tisky	77
	Zadané příkazy	78
5.5	Práce s vytvořeným modulem	78
	Proměnná s odkazem na objekt modulu	78
5.6	Oprava načteného modulu	79
5.7	Opětovné načtení opraveného modulu	79
5.8	Zdrojové kódy	80

Část B Začínáme programovat 81

6	Definice funkcí	82
6.1	Nejprve trocha syntaxe	82
	Fyzické a logické řádky	82
	Složené příkazy a odsazování	83
6.2	Definice funkce	83
	Funkce je objekt, na nějž odkazuje proměnná	84
	Dokumentační komentář	85
	Získání nápovědy – dokumentace	86
	Definice funkce je obvyčejný složený příkaz	86
6.3	Definice vlastní funkce	86
	Lokální proměnné	87
6.4	Funkce s návratovou hodnotou	87
	Shoda názvu proměnných	87
6.5	Funkce s parametry	88
	Zadávání argumentů	88
	Implicitní hodnoty argumentů	89
	Povinně poziční a povinně pojmenované argumenty	89
6.6	Funkce print() a její parametry	90
6.7	Definice prázdné funkce	90
6.8	Datový typ	91
6.9	Anotace	92
6.10	Zdrojové kódy	93
7	Balíčky, knihovny, robot Karel a jeho svět	94
7.1	Balíčky	94
	Trocha teorie	94
	Název modulu – balíčku	95
7.2	Knihovny	95
7.3	Robot Karel a jeho svět	95
	Historie robota Karla	96
	Vytvoření světa	97
	Vytvoření robota	98
	Akce	99
	Testy	101
	Zrychlování	101
	Vnořené zrychlování činnosti	102
	Ukončení práce s daným světem robotů	103
7.4	Zdrojové kódy	103

8	Rozhodování	104
8.1	Logické hodnoty	104
8.2	Terminologie výrazů	105
8.3	Porovnávání hodnot	106
	Porovnání reálných čísel	106
	Zřetěžené porovnávání	107
	Porovnávání textů	107
	Porovnávání totožnosti objektů	107
8.4	Logické operátory a operace	108
8.5	Podmíněný výraz	110
8.6	Podmíněný příkaz	111
	Jednoduchý podmíněný příkaz	111
	Vnořování složených příkazů	113
	Větev <code>else</code> – úplný podmíněný příkaz	114
	Rozhodování s více větvemi: rozšířený podmíněný příkaz	115
8.7	Přepínač – příkaz <code>match ... case</code>	116
8.8	Zdrojové kódy	117
9	Opakování kódu, cykly	118
9.1	Předehra	118
9.2	Příkaz <code>while</code> – cyklus se vstupní podmínkou	118
	Zanořování cyklů	119
9.3	Nekonečný cyklus	120
9.4	Příkaz <code>break</code> – cyklus s podmínkou uprostřed	121
9.5	Cyklus s koncovou podmínkou	122
9.6	Zdroje hodnot	122
	Rozbalovací hvězdička	123
	Stringy	123
	Objekt typu <code>range</code>	123
	Objekt typu <code>enumerate</code>	124
9.7	Příkaz <code>for</code> – cyklus s parametrem	124
	Proměnná odkazující na funkci	125
	Definice testu	126
9.8	Rekurze	127
9.9	Zdrojové kódy	128
10	Kontejnery	129
	Zvláštnosti programových kontejnerů	129
10.1	Kontejnery	129
10.2	Proměnné, neměnné a hešovatelné objekty	130
	Hešovatelné objekty	130
10.3	Druhy kontejnerů	131
10.4	Vytváření kontejnerů	132
	Vytváření prostřednictvím literálů	132
	Vytváření prostřednictvím konstruktorů	133
	Vytváření prostřednictvím generátorové notace	135
	Generátory lze použít jen jednou	136
10.5	Zdrojové kódy	137
11	Práce s kontejnery	138
11.1	Funkce versus metoda	138
11.2	Získání prvku z posloupností a slovníků	138
11.3	Procházení kontejnerů – cyklus <code>for</code>	139
	Procházení posloupnostmi	140
	Procházení slovníků – pohledy	141
	Konvence pro názvy slovníků	142

11.4	Vykrajování (slicing) posloupností	142
11.5	Další možnosti práce s jednotlivými prvky	143
	Test přítomnosti prvku	143
	Přidání a odebrání prvku	144
11.6	Funkce s proměnným počtem argumentů	145
	Hvězdičkový parametr	145
	Hvězdičkový argument	145
	Dvuhvězdičkový parametr	146
	Dvuhvězdičkový argument	146
11.7	Jmenné prostory	148
11.8	Zdrojové kódy	148
12	Ošetřování chyb	149
12.1	Tři druhy chyb	149
	Syntaktické chyby	149
	Běhové chyby	150
	Logické chyby	150
12.2	Reakce na vznik běhových chyb	150
12.3	Zachycení a ošetření výjimky	151
	Průchod programu bloky try ... except ... finally	151
12.4	Demonstrační příklad	152
12.5	Chování programu za běhu	153
12.6	Analýza chybové zprávy	154
12.7	Ladění a kontrolní tisky	155
	Služby modulu dbg	155
12.8	Zdrojové kódy	158

Část C Základy OOP 159

13	Úvod do OOP	160
13.1	Proč se učit objektové paradigma	160
	Kdy se OOP začíná vyplácet	161
13.2	Základní princip OOP	161
	Objekty a jejich atributy	162
	Specifika atributů Pythonu	163
	Práce s objekty – kvalifikace	163
	Zprávy x metody	164
	Metody versus funkce	164
13.3	Třídy a jejich instance	164
	Třída	165
	Instance	165
	Vytváření instancí – konstruktor, alokátor, initor	165
13.4	Definice třídy a jejich atributů	166
	Dokumentační komentář	167
	Výkonné a výrazové příkazy	168
	Datové atributy	168
	Instanční metody	168
	Statické metody	169
	Dekorátory	169
	Initor a instanční datové atributy	170
	Metody <code>__repr__()</code> a <code>__str__()</code>	170
	Speciální identifikátory – dunder	171
	Definice třídy je obyčejný příkaz	172
13.5	Práce s vytvořenou třídou a jejími instancemi	172
13.6	Definice prázdné třídy	173
13.7	Zdrojové kódy	173

14	Dědění	174
14.1	Rozhraní versus implementace	174
	Signatura versus kontrakt	175
	PINI	175
14.2	Základní terminologie dědění	175
	Dědění versus dědičnost	176
14.3	Tři druhy dědění	176
	Přirozené (nativní) dědění	176
	Dědění rozhraní	177
	Dědění implementace	177
	LSP – substituční princip Liskovové	177
14.4	Způsoby dědění	178
	Statické a dynamické typování	178
	Jmenovité dědění (nominal subtyping)	178
	Strukturální dědění a kachní typování	179
14.5	Polymorfismus	180
14.6	Virtuální metody a jejich přebíjení	180
14.7	Rodičovský podobjekt	181
14.8	Initory v procesu dědění	181
14.9	Definice rodičovské a dceřiné třídy	181
	Použití vytvořených tříd	183
14.10	Násobné dědění a diamantový problém	183
	Návrh třídy s více bezprostředními rodiči	184
	Použití definovaných tříd	185
14.11	Zobecňování	186
14.12	Abstraktní třídy	186
	Terminologie	186
	Shrnutí	187
14.13	Protokoly a statické kachní typování	188
	Třída Protocol	188
	Příklad	189
14.14	Zdrojové kódy	190
15	Co ještě můžete potřebovat	191
15.1	Ještě jednou funkce versus metody	191
	Převod funkce na metodu	191
	Účel návratové hodnoty s odkazem na instanci	193
	Převod metody na funkci	193
15.2	Použití nelokálních proměnných	194
	Příkaz global	194
	Příkaz nonLocal	196
15.3	Neveřejné atributy	196
	Atribut <code>_all_</code> modulů	197
15.4	Atributy x vlastnosti	197
	Zadávání a používání vlastností	198
	Vlastnost je instancí třídy	200
15.5	Pojmenované n-tice	200
15.6	Výčtové typy	201
15.7	Zdrojové kódy	203
16	Vytváříme balíčky a aplikace	204
16.1	Balíčky jako moduly	204
	Initor balíčku	204
	Šablona initoru balíčku	205
	Relativní import	206
16.2	Přímé spuštění zadaného skriptu	208

	Rozpoznání režimu, v němž byl modul spuštěn.....	208
	Ukázka	208
16.3	Vytvoření spustitelné aplikace	209
	Soubor typu pyz	212
16.4	Zdrojové kódy	212

Část D Vývoj aplikace 213

17	Základy objektové architektury	214
17.1	Předmluva	214
17.2	Architektura.....	215
17.3	Hlavní zásady návrhu	215
	Připravenost na změny	216
	CRIDP – maximální přehlednost	216
	KISS – maximální jednoduchost	216
	YAGNI – žádné zbytečnosti	216
	SoC – jediný zodpovědný	217
	SRP – jediná zodpovědnost	217
17.4	Návrhové vzory	217
17.5	Antivzory	219
17.6	Návrh programu.....	219
17.7	Druhy vytvářených objektů.....	220
17.8	Dva způsoby návrhu.....	221
	Návrh shora dolů	221
	Návrh zdola nahoru	221
	Porovnání	222
17.9	UML – diagram tříd	223
17.10	Zdrojové kódy	224
18	Návrh základní architektury	225
18.1	Proč právě textová konverzační hra	225
18.2	Záznamy průběhu vývoje v této učebnici.....	226
18.3.	Koncepce vyvíjené aplikace	226
	Co to je h-objekt	228
18.4	Zadání	228
18.5	Účastníci – objekty vystupující ve hře	229
18.6	Správci skupin objektů	232
	Správci v naší aplikaci	233
18.7	Vytvoření zárodka budoucí aplikace	233
18.8	Zdrojové kódy a UML diagram	234
19	Připravujeme test	235
19.1	Jak testovat	235
	Programování řízené testy	235
	Jednotkové, integrační a regresní testy.....	236
	Možnosti testování naší hry	237
19.2	Scénáře.....	237
	Modul scenarios	238
19.3	Kroky definující stav hry	238
19.4	Definice třídy ScenarioStep.....	239
19.5	Definice šťastného scénáře.....	241
19.6	Simulace běhu hry	242
	Jednoduchá simulace.....	242
	Podrobnější simulace	243
19.7	Zdrojové kódy a UML diagram	244

20	Definice testu a start hry	245
20.1	Jak budeme testovat	245
	Zadání příkazu hře	245
	Odpověď a pozice	246
	Sousedé	246
	H-objekty v prostoru	246
	H-objekty v batohu	246
	Oznámení o navštíveném prostoru	247
20.2	Vlastní test hry	247
	Úvodní testy	248
	Funkce ERROR()	249
	Funkce compare_containers()	249
	Proč na malá písmena	250
20.3	Spouštíme test	250
20.4	Další postup	251
20.5	Tři druhy objektů	251
20.6	Delegování zodpovědnosti	252
20.7	Informace, zda hra běží	252
20.8	Funkce execute_command() v modulu actions	253
	Definice má být krátká	254
20.9	Funkce execute_empty_command()	254
20.10	Funkce _execute_standard_command()	255
20.11	Spuštění testu	255
20.12	Zdrojové kódy a UML diagram	256
21	Svět hry a první akce	258
21.1	Přípravné akce – inicializace	258
21.2	Pojmenované objekty	259
21.3	Úprava definice třídy Item	259
21.4	Úprava definice třídy Place	260
	Inicializace prostorů	260
21.5	Inicializace modulu a test	262
21.6	Mezivýsledek a jeho UML diagram	262
21.7	Obecná akce	263
	Řešení v minulém vydání	263
	Alternativní řešení	264
21.8	Společný rodič batohu a prostorů	264
	Initor	264
	Inicializace	265
	Přidání položky	266
	Odebrání položky	266
21.9	Nebezpečí degenerovaných objektů	266
21.10	Úprava initoru třídy ANamed	267
21.11	Upravená definice prostorů a batohu	267
21.12	Upravená definice batohu	268
21.13	Definice akce Vezmi	269
	Test	269
21.14	Definice akce Polož	270
21.15	Definice akce Jdi	271
21.16	Akce Konec	271
21.17	Zdrojové kódy a UML diagram	272
22	Co ještě chybí	273
22.1	Nesplněné body zadání	273
	Nový scénář	274

22.2	Třída <code>Scenario</code>	274
22.3	Chybový scénář	275
	Společný startovní krok	275
	Co vše se má zkontrolovat	275
	Nekorektní spuštění	276
22.4	Dodatečné definice testů	276
	Odstranění automatického spuštění testu při importu	278
22.5	Opakované spuštění	278
22.6	Nekorektní spuštění	279
	Změna indexace	279
	Test ukončení hry	279
22.7	Nezadané argumenty	280
22.8	Nezvednutelné h-objekty	281
	Předpona může mít širší význam	282
	Oprava definice prostorů	282
	Úprava metody <code>_Take.execute()</code>	283
22.9	Konečná kapacita batohu	283
	Metoda <code>try_add()</code>	283
	Konečná verze metody <code>_Take.execute()</code>	284
22.10	Nápověda	284
22.11	Úprava testovací funkce	285
22.12	Zdrojové kódy a UML diagram	285
23	Spustitelná aplikace	286
23.1	Jednoduché textové uživatelské rozhraní	286
23.2	Odstranění kontrolních tisků	287
23.3	Možnost opakovaného spuštění	288
23.4	Rozšíření výstupu	289
23.5	Vytvoření spustitelné aplikace	289
23.6	Argumenty příkazového řádku	290
	Doplnění modulu <code>__init__</code>	291
23.7	Zdrojové kódy a UML diagram	291
24	Vylepšujeme aplikaci	292
24.1	Primitivní GUI	292
	Změna architektury	292
	Třída <code>Console</code>	292
	Atribut <code>io</code>	294
	Úprava funkcí <code>run()</code> a <code>multirun()</code>	294
	Knihovna <code>tkinter</code>	295
	Modalita dialogových oken	295
	Návrhový vzor Fasáda	296
	Primitivní dialogová okna	296
	Modul <code>tkinter.messagebox</code>	296
	Modul <code>tkinter.simpledialog</code>	297
	Rodičovské okno	298
	Schování okna	299
	Modul dialogových oken	299
	Přímé spuštění aplikace	300
	Mezivýsledek – balíček <code>v21_gui</code>	301
24.2	Převod literálů na konstanty	301
	Magické hodnoty	301
	Modul textových konstant	302
	Konstanty související s prostory	302
	Konstanty související s h-objekty	302
	Definice světa hry	304

	Další úpravy.....	304
24.3	Zdrojové kódy	305
25	Kudy dál	306
25.1	Další vylepšování.....	306
25.2	Přehled námětů	306
	Převod pod kvalitní grafické uživatelské rozhraní	307
	Změna světa hry.....	307
	Zdokonalení h-objektů.....	307
	H-objekty – prostory	307
	Rozšiřování sady příkazů.....	308
	Rozhovor	308
	Přehled námětů pro inspiraci	308
25.3	Tipy pro učitele	308
25.4	Další zdroje.....	309

Literatura	310
-------------------	------------

Rejstřík	312
-----------------	------------

Část E Přílohy a seznamy	321
---------------------------------	------------

Poděkování

Vím, že se v českých knížkách většinou neděkuje, ale tvorba knih v současném tempu je spojena s takovými oběťmi řady lidí z mého blízkého i vzdálenějšího okolí, že bych měl velkou újmu na duši, kdybych tak neučinil.

Chtěl bych především nesmírně poděkovat své ženě Jarušce, která byla po celou dobu mojí největší oporou a jejíž nekonečná trpělivost a vstřícnost mi pomohla dokončit knihu v termínu, který se příliš nelišil od toho, jež jsme původně s nakladatelem dohodli, a ne až někdy za rok po něm. Původně jsem se domníval, že s druhým vydáním nebude moc práce. Šeredně jsem se zmýlil, protože veškeré úpravy, které jsem se rozhodl do knihy zanést (a že jich bylo požehnaně), musely zapadnout do předchozího textu.

Na vylepšování textu nového vydání se ale podílela řada dalších lidí. Mezi nimi musím poděkovat především těm, kteří si dali tu práci a při objevení chyby v minulém vydání mi o ní napsali. Mezi nimi pak především Luděkovi Šťastnému, který po celou přípravu rukopis pročítal a odhaloval v něm pasáže, které by si zasloužily vylepšit. Neméně pak děkuji Mirkovi Viriusovi, který celý finální rukopis pečlivě přečetl a objevil v něm několik přehlédnutí, za něž bych se po vydání styděl. Velkou zásluhu na současné podobě má i Jirka Kofránek, který mne upozornil na některé problémy s výukou podle minulého vydání a průběžně pak se mnou konzultoval problémy, na něž narazili jeho studenti. Svými poznámkami přispěl ke srozumitelnosti textu i Josef Froněk.

Svůj dík si zaslouží i Karina Mukhambetova, Iuliia Shestopalova, Paulina Znamenaková a Valeriia Zvezdina, které pročetly rukopis jako laické lektorky a upozorňovaly na pasáže, které byly hůře pochopitelné.

Velký dík patří i redaktoru Petrovi Somogyimu, který musel opakovaně procházet některé již zredigované pasáže, protože jsem je znovu a znovu upravoval. Nemalý dík patří i šéfredaktoru Radku Matulíkovi, který mne k napsání jednotlivých knih z posledních let vyhecoval, a byl pak ochoten týden či dva počkat, když se mi nepodařilo přesně dodržet původně dohodnutý termín.

Úvod

Python je moderní programovací jazyk, který umožňuje velmi jednoduše navrhovat jednoduché programy, ale na druhou stranu nabízí dostatečně mocné prostředky k tomu, abyste mohli s přiměřeným úsilím navrhovat i programy poměrně rozsáhlé. Je pro něj vyvinuto obrovské množství knihoven a frameworků, které uživatelům umožňují soustředit se na řešení úkol a nerozptylovat se vývojem nejrůznějších pomocných podprogramů.

Python je v současné době nejlepším jazykem pro ty, kteří se nechtějí živit jako programátoři, ale jejich profese či zájem je nutí jednou za čas něco naprogramovat. Potřebují proto jazyk, který se mohou rychle naučit a v němž budou moci rychle vytvářet jednoduché programy řešící (nebo pomáhající řešit) jejich problém. Na druhou stranu ale sílí i jeho využití profesionálními vývojáři pro rozsáhlé podnikové a webové aplikace.

Komu je kniha určena

Tato kniha je určena především těm, kteří ještě nikdy neprogramovali, anebo se je to sice někdo snažil naučit, ale oni už většinu látky zapomněli. Nepředpokládá žádné předběžné znalosti a dovednosti kromě základů práce s počítačem. Jejím cílem je předat čtenáři základní znalosti a naučit ho dovednosti potřebné k vytváření jednoduchých aplikací. Osvojené základy jim pak umožní, aby v případě hlubšího zájmu o programování v jazyku *Python* pokračovali některou z učebnic určených pro mírně pokročilé programátory – nejlépe samozřejmě mojí učebnicí [13] a doplňkovou příručkou [14].

Zkušenost však ukázala, že v této učebnici najdou řadu cenných informací i programátoři, kteří již mají jisté zkušenosti, ale kurzy, jimiž doposud prošli, se soustředily především na odpověď na otázku *jak program zapsat*, a oni by se nyní rádi dozvěděli také odpověď na otázku *jak složitější program navrhnout*.

Kniha je učebnicí programování. Učí své čtenáře navrhovat programy a dále je vylepšovat. Není učebnicí jazyka *Python* (tou je učebnice [13]), a proto se nesnaží probrat všechny jeho konstrukce, ale omezuje se při výkladu pouze na ty, jejichž zvládnutí je pro návrh jednoduché aplikace nezbytné.

Vedle konstrukcí jazyka ale učí čtenáře také řadu zásad moderního programování, jejichž zvládnutí je nutnou podmínkou pro všechny, kdo nehodlají zůstat u malých žákovských programů, ale chtějí se naučit efektivně vyvíjet robustní středně rozsáhlé aplikace, jejichž údržba nebude vést uživatele k chrlení nepublikovatelných výroků na adresu autora.



Dopředu se omlouvám, že se kniha částečně překrývá se zmíněnou příručkou [13]. Některé věci je prostě třeba vysvětlit jak naprostým začátečníkům v programování, pro něž je určena tato učebnice, tak těm, kteří jsou sice zkušení programátoři, ale potřebují na *Python* přejít z jiného jazyka nebo si prostě prohloubit své znalosti (pro ně jsem napsal knihu [13]).

Koncepce výkladu a jeho uspořádání

Kniha je koncipována tak, aby mohla sloužit jako středoškolská učebnice programování i jako učebnice pro samouky zajímající se o programování. Probírá vše potřebné od naprostých základů až po některé rysy, které se v běžných příručkách často neprobírají, ale jejichž znalost považuji za velmi užitečnou, protože pomáhá rychleji odhalit příčiny mnohých chyb.

Kniha je rozdělena do čtyř částí.

První část

První část probírá naprosté základy, bez jejichž znalosti nelze vytvořit ani velice jednoduchou aplikaci. Naučíte se v ní pracovat s čísly a texty a dozvíte se, jak používat funkce. Poté stručně probere naprosté základy používání objektů, bez nichž se v *Pythonu* nedá programovat, a na závěr vysvětlí, jak vytvářet moduly, abyste mohli své programy také ukládat a opakovaně používat.

Doprovodné programy v první části se nesnaží nic řešit. Jsou to vesměs AHA-příklady, tj. příklady, jejichž jediným cílem je, aby si čtenář řekl: „Aha, takto to funguje.“

Druhá část

V druhé části začnete vytvářet jednoduché programy. Naučí vás definovat vlastní funkce a postupně probere základní algoritmické konstrukce a datové struktury, mezi něž patří především různé druhy kontejnerů. Na závěr vysvětlí koncepci ošetřování chyb a předvede používání k tomu určených programových konstrukcí.

Na konci druhé části budete umět samostatně navrhnout a odladit jednoduchý program. Kdo bude chtít, může v tuto chvíli příručku na chvíli odložit a pustit se do vývoje svých programů. Ke studiu dalších částí se vrátí, až získá v programování *Pythonu* jisté zkušenosti a bude umět docenit témata probíraná v těchto částech.

Třetí část

Třetí část vás seznámí se základy objektivě orientovaného programování. Vysvětlí, kdy začne být výhodný přechod na objektivě orientované paradigma, probere základní vlastnosti objektů a tříd a naučí vás pravidla pro jejich vytváření a používání. Zvláštní kapitola je věnována dědění, jeho různým podobám a pravidlům a především jeho nástrahám. Poté pokračuje výkladem některých specifických konstrukcí a datových typů.

Na konci druhé části budete nadstandardně obeznámeni s objektivě orientovaným paradigmatickým a jeho implementací v *Pythonu*. Budete připraveni vytvářet rozsáhlejší