

言語研究のための プログラミング入門

Python を活用したテキスト処理

浅尾仁彦・李在鎬

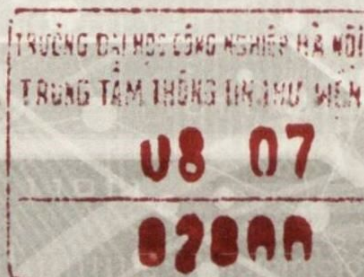
Python

開拓社

言語研究のための プログラミング入門

Pythonを活用したテキスト処理

浅尾仁彦・李在鎬



開拓社

Python

1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100

はじめに

言語研究にとって、コンピュータの重要性はますます増えています。本書を手にとられる方も、文書作成やメール、ウェブ閲覧などで日常的にコンピュータを利用しているのではないかと思います。また、大規模コーパスの整備やインターネットの発達によって、これまで直感に頼るしかなかったような事柄でもデータに基づいて統計的に調べられるようになってきました。

しかしながら、コンピュータを前にして、単調なコピー・アンド・ペーストの作業を繰り返すなど、人間の側が機械的な作業をしていることがないでしょうか。コンピュータは、人間の代わりに機械的な作業をするためにあります。

時にはソフトウェアが初めから用意されていて、メニューから機能を選ぶだけでコンピュータに機械的な作業をさせることができる場合もあります。しかし、研究の目的は常にユニークなものです。ソフトウェアは、多くの利用者が必要とするであろうという最大公約数的な機能しか用意してくれませんから、既存のソフトウェアでうまく作業を自動化できるとは限りません。このような時に新しい「機能を作る」作業がプログラミングです。

しかしながら、言語研究者がプログラミングを学ぼうとして書店で本を手にとってみても、ちょうど良い本を見つけるのはなかなか難しいのが実情です。プログラミングのすぐれた入門書は多数出版されていますが、理工系の読者を想定していたり、ウェブサービスの構築など特定の目的を念頭においていたり、多くの場合、内容が言語研究者の背景知識や関心に合っていないためです。

本書は、言語学研究者に向け、言語研究のための処理を解説しています。コンピュータの操作に慣れていない文系の大学院生や研究者でも実践できるように、丁寧な解説を心がけています。全体の流れにおいても、ごく簡単な内容から出発して、一步一步複雑なプログラムが書けるようになるよう、

工夫しています。また同時に、テキスト処理での実用例を多く盛り込み、各機能がどのように役立つのか伝わるように心がけました。その代わりに、一般の入門書であれば書いてある事柄であっても、本書の目的に直接関係しない点については大胆に省略しており、機能をくまなく紹介するということはありません。大事なことは、機能の1つ1つを覚えることより、どうやってコンピュータに作業の流れを教え込むのかについての考え方に親しむことだと思うからです。実際に手を動かしながら読み進めていただければ幸いです。本書を通じてプログラミング処理が持つ威力を実感していただき、ご自分の研究にもぜひ応用していただきたいと思います。

本書の執筆のきっかけの1つになった Python 勉強会に参加していただいた京都大学文学研究科、人間・環境学研究科の皆様には感謝いたします。また、本書の草稿に目を通して多くの有益なコメントを下された小松原哲太氏、遠藤智子氏、中川奈津子氏、小木曾智信氏に感謝いたします。そして、Python 自体をはじめとして、本書で紹介したソフトウェアや言語資源を公開していただいている開発者やボランティアの方々に感謝いたします。最後に、企画および編集を通じてさまざまな便宜をはかってくださった開拓社の川田賢氏にお礼を申し上げます。

2013年5月

浅尾仁彦・李在鎬

目次

はじめに

第1章 言語研究とプログラミング	1
1.1. 本章の概要	1
1.2. なぜプログラミングが必要か	1
1.2.1. 大規模データによる言語研究	1
1.2.2. 機能を作る	3
1.2.3. 留意点	5
1.3. プログラミングの前に	6
1.4. 本書の活用	7
1.4.1. 構成	7
1.4.2. 専用ウェブサイトの活用	7
1.5. まとめ	9

第I部 プログラミングによるテキスト処理

第2章 テキストデータに親しもう	12
2.1. 本章の概要	12
2.2. テキストファイルの利点	12
2.3. テキストファイルを使う	14
2.3.1. テキストエディタとは	15
2.3.2. テキストエディタのインストール	16
2.3.3. テキストエディタによる検索例1	18
2.3.4. テキストエディタによる検索例2	19
2.3.5. テキストエディタによる置換	21
2.4. 文字コードと改行コード	23
2.4.1. 文字コードとは	23
2.4.2. 文字コードを調べる	24

2.4.3. 文字コードを変更する	25
2.4.4. 改行コードとは	27
2.5. まとめ	28

第3章 正規表現	29
3.1. 本章の概要	29
3.2. 正規表現とは	29
3.3. 正規表現の書き方	30
3.3.1. 「?」について	31
3.3.2. 「.」について	31
3.3.3. 「+」について	33
3.3.4. 「*」(アスタリスク)について	33
3.3.5. 「[]」(ブラケット)について	34
3.3.6. 「 」(パイプ)について	36
3.3.7. 「^」(キャレット)と「\$」(ドル)について	37
3.3.8. 後方参照	38
3.3.9. 組み合わせ	39
3.4. 正規表現を用いた置換	41
3.5. まとめ	44
3.6. 章末問題	44

第II部 Pythonの基本

第4章 Pythonに触れてみよう	48
4.1. 本章の概要	48
4.2. なぜPythonか	48
4.3. Pythonのインストール	49
4.4. Pythonの起動	51
4.5. Pythonで計算する	52
4.5.1. Pythonを電卓として試してみる	52
4.5.2. 変数	53
4.6. Pythonで文字列を扱う	55
4.6.1. 文字列の1文字目が何か調べる：インデクシング	56
4.6.2. 文字数を調べる：len()関数	58

4.6.3.	数値と文字列の区別	59
4.6.4.	データを文字列に変換する：str() 関数	60
4.6.5.	データを数値に変換する：int() 関数	60
4.7.	Python を終了する	61
4.8.	まとめ	62
4.9.	章末問題	62
第5章	Python でファイルの内容を表示してみよう	64
5.1.	本章の概要	64
5.2.	Python のプログラムを保存・実行する	64
5.2.1.	エディタでプログラムを書く	65
5.2.2.	プログラムを実行するための準備	66
5.2.3.	プログラムの実行	68
5.2.4.	エラーが出てしまったら	69
5.3.	プログラムにコメントを書く	72
5.3.1.	Python スクリプトの中に日本語で書くには	72
5.4.	実行結果をファイルに保存する：リダイレクト	74
5.5.	各行の解説	75
5.6.	まとめ	77
5.7.	章末問題	77
第6章	Python で検索しよう：条件分岐	78
6.1.	本章の概要	78
6.2.	プログラムを条件分岐させる	78
6.3.	文字列に関する条件式	81
6.3.1.	文字列が含まれるかどうかを調べる：in 演算子	82
6.3.2.	startswith() と endswith() 関数	83
6.4.	and, or, not	83
6.4.1.	否定：not	84
6.4.2.	かつ：and	84
6.4.3.	または：or	85
6.5.	else と elif	86
6.6.	if の実例	88
6.6.1.	ある条件を満たす行だけ表示する	88
6.6.2.	大文字小文字を区別せず検索する	89

6.6.3.	何も書かれていない行を削除する	91
6.7.	まとめ	92
6.8.	章末問題	92
第7章	繰り返し処理を覚えよう：ループ	94
7.1.	本章の概要	94
7.2.	ループの基本	94
7.3.	ループの制御	97
7.3.1.	必要のない行をスキップする：continue	97
7.3.2.	ループを中断する：break	99
7.4.	ループの応用例	101
7.4.1.	ファイルに行番号を振る：カウンタ	101
7.4.2.	ファイルの最初の10行だけ表示する：カウンタ	103
7.4.3.	キーワードがあったことを覚えておく：フラグ	104
7.4.4.	どの行にも「ない」ことを確かめる	106
7.5.	まとめ	108
7.6.	章末問題	109
第8章	単語の一覧表を作ろう：リスト	111
8.1.	本章の概要	111
8.2.	リストの基本	111
8.2.1.	リストのインデクシングとスライシング	113
8.2.2.	リストへの追加	115
8.2.3.	リストの並べ替え	117
8.3.	リストとループ	119
8.4.	ファイルの行を並べ替える	121
8.5.	出現した単語のリストを作る	123
8.5.1.	文字列とリストの変換：split と join	123
8.5.2.	出現した単語のリストを作る	126
8.5.3.	句読点などの処理	129
8.6.	表形式のデータを処理する	130
8.6.1.	発音辞書を使って特定の発音をもつ単語のみ表示する	130
8.7.	まとめ	134
8.8.	章末問題	135

第9章 頻度表を作ろう：ディクショナリ	136
9.1. 本章の概要	136
9.2. ディクショナリの基本	136
9.2.1. ディクショナリの項目を1つ1つ処理する	139
9.3. ディクショナリで頻度表を作ろう	141
9.3.1. 頻度表をアルファベット順に並べ替える	144
9.3.2. 頻度表を頻度順に並べ替える	144
9.4. 発音辞書をディクショナリとして読み込んで利用する	145
9.5. まとめ	149
9.6. 章末問題	149
第10章 ファイル操作	151
10.1. 本章の概要	151
10.2. ファイル入出力の仕方のいろいろ	151
10.2.1. 入力ファイル名を実行時に指定する	152
10.2.2. ファイル名を指定して書き込む	155
10.2.3. 検索語をコマンドラインで指定できるようにする	156
10.3. 複数あるファイルを一気に処理する	158
10.3.1. Python でファイル一覧を表示する	158
10.3.2. フォルダ内の全ファイルの内容を表示する	160
10.3.3. ファイル名付きで全ファイルを表示する	161
10.3.4. フォルダ内の全ファイルから検索する	162
10.3.5. フォルダ内の全ファイルを書き換える	163
10.4. まとめ	166
10.5. 章末問題	167
第11章 Python で正規表現を使ってみよう	168
11.1. 本章の概要	168
11.2. Python での正規表現検索の基本	168
11.2.1. マッチしたかどうかで条件分岐する	169
11.2.2. マッチした行を一覧表示する	170
11.2.3. マッチした単語を集計する	171
11.2.4. マッチした部分をリストにする	174
11.3. 置換	175
11.3.1. 置換機能を用いて検索結果を強調する	176