

数理・データサイエンス・AIを学ぶための土台となるアルゴリズムの基礎を解説！  
**『数理・データサイエンス・AIのためのアルゴリズム入門』**  
発行

インプレスグループで理工学分野の専門書出版事業を手掛ける株式会社近代科学社は、2026年6月25日に、近代科学社 Digital レーベル(※)より、単行本版『数理・データサイエンス・AIのためのアルゴリズム入門』(著者:鄭 宇景)の発売を開始いたしました。



※近代科学社 Digital とは:近代科学社が著者とプロジェクト方式で協業する、デジタルを駆使したオンデマンド型の出版レーベルです。詳細はこちらもご覧ください。

<https://www.kindaikagaku.co.jp/kdd/scheme/>



●書誌情報

【書名】数理・データサイエンス・AIのためのアルゴリズム入門

【著者】鄭 宇景

【仕様】B5判・並製・印刷版モノクロ/電子モノクロ・本文224頁

【印刷版基準価格】3,200円(税抜)

【電子版基準価格】3,200円(税抜)

【ISBN】(カバー付き単行本)978-4-7649-0792-8 C3004

【ISBN】(POD)978-4-7649-6146-3 C3004

【商品URL】[https://www.kindaikagaku.co.jp/book\\_list/detail/9784764961463/](https://www.kindaikagaku.co.jp/book_list/detail/9784764961463/)

●内容紹介

本書は、プログラミング経験のない大学1年生を主な対象とした、アルゴリズムとデータ構造の教科書です。現代社会で必須の「問題を解くための手順を考える力」を養うことを目的とし、効率的な手順の設計力を身につけることができます。

全14章の構成は基礎的なフローチャートから始まり、探索・整列アルゴリズム、計算量の評価、スタックやキューなどのデータ構造までを段階的に解説。

各章では日常の具体例から理論を導入し、Pythonによる実装、演習問題へと進むサイクルを繰り返すため、初心者でも抽象的な概念を無理なく理解できる設計となっています。

2進数や画像データの仕組みといったコンピュータの基礎知識もバランスよく習得可能で、独学者はもちろん大学の講義用テキストとしても最適な構成となっています。

●著者紹介

鄭 宇景(ジョン・ウーギョン)

早稲田大学政治経済学部卒(2017)、早稲田大学経済学研究科修士課程修了(2019)、早稲田大学経済学研究科博士後期課程満期退学(2025)。現在、関東学園大学経済学部准教授。同大学のデータサイエンス教育プログラムにおいて「データリテラシー」「プログラミング実践」「アルゴリズム論」などを担当している。専門はマクロ計量経済学。

---

## ●目次

### 第1章 アルゴリズムの考え方

#### 1.1 アルゴリズムとは

##### 1.1.1 日常生活のアルゴリズム

##### 1.1.2 良いアルゴリズムとは

#### 1.2 論理的思考の基礎

##### 1.2.1 命題の真・偽

##### 1.2.2 否定

##### 1.2.3 「または」と「かつ」の使い分け

##### 1.2.4 否定と論理演算の組み合わせ

##### 1.2.5 条件文「ならば」

#### 1.3 フローチャートによる図式化

#### 1.4 アルゴリズムからプログラムへ

#### 1.5 演習問題

### 第2章 Python から学ぶプログラミング基礎

#### 2.1 四則演算

#### 2.2 変数

#### 2.3 データ型

#### 2.4 文字列

#### 2.5 リスト

#### 2.6 関数

#### 2.7 ブール型、比較演算子、そして論理演算子

#### 2.8 if 文

#### 2.9 while 文

#### 2.10 for 文

#### 2.11 演習問題

### 第3章 基礎的なアルゴリズム

#### 3.1 日常問題

##### 3.1.1 お掃除

#### 3.2 数字のアルゴリズム

##### 3.2.1 数列の合計

##### 3.2.2 コラッツ予想

##### 3.2.3 素数の判定

##### 3.2.4 最大公約数

#### 3.3 演習問題

### 第4章 サーチアルゴリズムの概要

#### 4.1 「探す」ことの抽象化

- 4.1.1 日常でものを探す場合
- 4.1.2 リストから値を探す
- 4.1.3 一般的な探し方を求めるには
- 4.2 一番簡単な探し方「リニアサーチ」
- 4.2.1 全ての要素との比較
- 4.2.2 リニアサーチの関数化
- 4.2.3 リニアサーチのフローチャート
- 4.2.4 リニアサーチでインデックスを返す
- 4.2.5 リニアサーチの特徴
- 4.3 演習問題

## 第5章 再帰関数とバイナリサーチ

- 5.1 フィボナッチ数列と階乗
- 5.2 再帰関数の概念
- 5.2.1 再帰関数とは
- 5.2.2 フィボナッチ数列の再帰関数
- 5.2.3 階乗の再帰関数
- 5.2.4 再帰関数の注意点
- 5.3 より速い探し方「バイナリサーチ」
- 5.3.1 バイナリサーチの考え方
- 5.3.2 バイナリサーチの手順
- 5.3.3 バイナリサーチと再帰
- 5.3.4 値が見つからない場合
- 5.3.5 バイナリサーチのフローチャート
- 5.4 演習問題

## 第6章 ビッグオー技法による時間計算量の測り方

- 6.1 時間計算量の概念
- 6.2 ビッグオー技法
- 6.2.1 代表的な時間計算量
- 6.2.2 演算回数の比較表
- 6.3 リニアサーチとバイナリサーチの比較
- 6.3.1 リニアサーチの時間計算量
- 6.3.2 バイナリサーチの時間計算量
- 6.3.3 具体的な比較
- 6.3.4 比較のまとめ
- 6.4 演習問題

## 第7章 空間計算量とハッシュ探索法

- 7.1 空間計算量とは
- 7.2 ハッシュ関数の概念
- 7.2.1 簡単なハッシュ関数を作ってみよう
- 7.3 ハッシュ探索法
- 7.3.1 Python の辞書型
- 7.3.2 ハッシュ探索法の実装

### 7.3.3 衝突の対処法

## 7.4 演習問題

## 第8章 ソートアルゴリズムの概要

### 8.1 「並べ替える」ことの抽象化

#### 8.1.1 日常における並べ替えの例

#### 8.1.2 昇順と降順

### 8.2 リストの並べ替え

### 8.3 ソートアルゴリズム

#### 8.3.1 ソートアルゴリズムの考え方

### 8.4 リスト要素の入れ替え

#### 8.4.1 Python の便利な入れ替え方法

#### 8.4.2 入れ替えを関数にする

### 8.5 演習問題

## 第9章 バブルソート

### 9.1 隣り合うもの同士の比較

### 9.2 コードが簡潔な整列方法「バブルソート」

#### 9.2.1 バブルソートの仕組み

#### 9.2.2 バブルソートの図式

#### 9.2.3 フローチャート

#### 9.2.4 Python での実装

#### 9.2.5 バブルソートの改良

### 9.3 計算量の解析

#### 9.3.1 比較回数の計算

#### 9.3.2 最良・最悪・平均の場合

#### 9.3.3 バブルソートの特徴のまとめ

### 9.4 演習問題

## 第10章 選択ソート

### 10.1 最大値・最小値の探し方

#### 10.1.1 日常での最小値探し

#### 10.1.2 最小値を探す Python コード

#### 10.1.3 最小値のインデックスを探す

### 10.2 仕組みが理解しやすい整列方法「選択ソート」

#### 10.2.1 選択ソートの考え方

#### 10.2.2 選択ソートの動作例

#### 10.2.3 選択ソートの Python 実装

#### 10.2.4 選択ソートのフローチャート

### 10.3 計算量の解析

#### 10.3.1 時間計算量

#### 10.3.2 バブルソートとの比較

#### 10.3.3 最良・最悪ケース

### 10.4 演習問題

## 第11章 クイックソート

### 11.1 分割統治法とは

#### 11.1.1 日常例で考える分割統治法

### 11.2 再帰を用いる整列方法「クイックソート」

#### 11.2.1 クイックソートの考え方

#### 11.2.2 ステップごとの動作

#### 11.2.3 Python での実装

#### 11.2.4 再帰呼び出しの追跡

### 11.3 計算量の解析

#### 11.3.1 平均的な場合

#### 11.3.2 最悪の場合

#### 11.3.3 計算量のまとめ

### 11.4 演習問題

## 第12章 マージソート

### 12.1 二つのリストの結合

#### 12.1.1 日常での「結合」

#### 12.1.2 結合操作のイメージ

#### 12.1.3 結合操作の Python 実装

#### 12.1.4 結合操作のフローチャート

### 12.2 空間計算量を使う整列方法「マージソート」

#### 12.2.1 マージソートの基本的な考え方

#### 12.2.2 マージソートの動作例

#### 12.2.3 マージソートの Python 実装

#### 12.2.4 マージソートと空間計算量

### 12.3 計算量の解析

#### 12.3.1 時間計算量の解析

#### 12.3.2 マージソートの安定性

#### 12.3.3 他のソートアルゴリズムとの比較

### 12.4 演習問題

## 第13章 データ構造

### 13.1 データ構造とは

### 13.2 スタック

#### 13.2.1 スタックの概念

#### 13.2.2 Python リストによるスタックの実現

#### 13.2.3 スタックの応用:括弧の対応チェック

### 13.3 キュー

#### 13.3.1 キューの概念

#### 13.3.2 Python リストによるキューの実現

#### 13.3.3 キューの状態を追ってみよう

### 13.4 連結リスト

#### 13.4.1 連結リストの概念

#### 13.4.2 辞書によるノードの表現

#### 13.4.3 連結リストの操作

## 13.5 演習問題

### 第14章 様々なデータ

#### 14.1 2進数と16進数

##### 14.1.1 なぜ2進数なのか

##### 14.1.2 2進数の仕組み

##### 14.1.3 10進数から2進数への変換

##### 14.1.4 16進数

#### 14.2 データ量の単位

#### 14.3 ASCII テーブルと文字列

##### 14.3.1 文字コードとASCII

##### 14.3.2 Python での文字コード操作

##### 14.3.3 ASCII コードを使ったアルゴリズム

##### 14.3.4 Unicode

#### 14.4 RGB 体系と画像

##### 14.4.1 色の表現

##### 14.4.2 Python での RGB 操作

##### 14.4.3 ピクセルと画像

#### 14.5 演習問題

### 付録A Python 設定の仕方

#### A.1 Anaconda によるインストール

##### A.1.1 Anaconda とは

##### A.1.2 Windows へのインストール

##### A.1.3 Mac へのインストール

#### A.2 IPython の操作方法

##### A.2.1 IPython とは

##### A.2.2 IPython の起動と終了

##### A.2.3 基本的な使い方

##### A.2.4 マジックコマンド

#### A.3 Jupyter Notebook と Google Colab

##### A.3.1 Jupyter Notebook とは

##### A.3.2 Google Colab とは

### 付録B 演習問題の解答

---

**【近代科学社 Digital】** <https://www.kindaikagaku.co.jp/kdd/index.htm>

近代科学社 Digital は、株式会社近代科学社が推進する 21 世紀型の理工系出版レーベルです。デジタルパワーを積極活用することで、オンデマンド型のスピーディで持続可能な出版モデルを提案します。

**【株式会社 近代科学社】** <https://www.kindaikagaku.co.jp/>

株式会社近代科学社（本社：東京都千代田区、代表取締役社長：大塚浩昭）は、1959 年創立。

数学・数理学・情報科学・情報工学を基軸とする学術専門書や、理工学系の大学向け教科書等、理工学専門分野を広くカバーする出版事業を展開しています。自然科学の基礎的な知識に留まらず、その高度な活用が要求される現代のニーズに応えるべく、古典から最新の学際分野まで幅広く扱っています。また、主要学会・協会や著

名研究機関と連携し、世界標準となる学問レベルを追求しています。

**【インプレスグループ】 <https://www.impressholdings.com/>**

株式会社インプレスホールディングス（本社：東京都千代田区、代表取締役：塚本由紀）を持株会社とするメディアグループ。「IT」「音楽」「デザイン」「山岳・自然」「航空・鉄道」「モバイルサービス」「学術・理工学」を主要テーマに専門性の高いメディア&サービスおよびソリューション事業を展開しています。さらに、コンテンツビジネスのプラットフォーム開発・運営も手がけています。

**【お問い合わせ先】**

株式会社近代科学社

TEL:03-6837-4828

電子メール: [kdd-qa@kindaikagaku.co.jp](mailto:kdd-qa@kindaikagaku.co.jp)