第5章

例題：2つの数の大きい方を表示する

アイデアを設計図へ：アルゴリズムを表現する3つのステップ

5-1 アルゴリズム（文章）

第一歩：誰もが理解できる「言葉」によるアイデアの定義

1. 数値Aを入力する
2. 数値Bを入力する
3. AがBより大きいか調べる
4. 大きければAを表示する
5. そうでなければBを表示する

なぜ重要か？

文章による表現は、専門知識を問わず関係者間で「何をしたいのか」という初期の合意を形成するための重要なツールです。
しかし、言葉の解釈は人によって揺らぐ可能性があり、そのままではコンピュータが実行できるほど厳密ではありません。
ここから思考の解像度を上げていく必要があります。

5-2 擬似コード

構造化への一歩：「実装」を意識した思考の翻訳

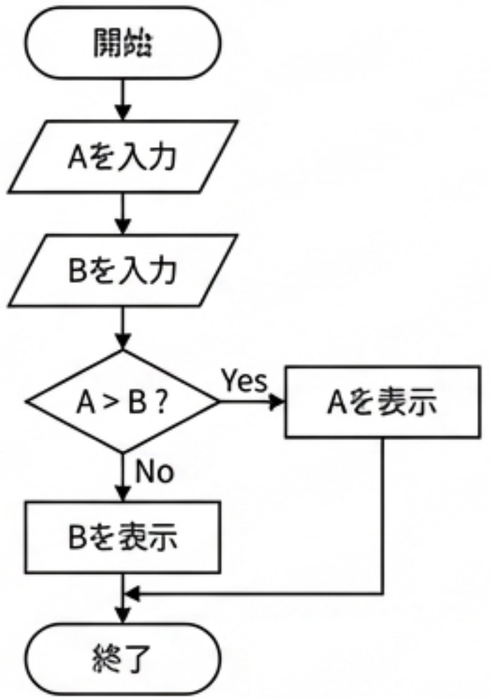
```
Aを入力
Bを入力
もし A > B なら
    Aを表示
そうでなければ
    Bを表示
終了
```

なぜ重要か？

擬似コードは、「もし〜なら」といった制御構造を導入することで、ロジックを厳密に表現します。特定のプログラミング言語の文法に縛られないため、エンジニアは実装の詳細ではなく、アルゴリズムの本質的な正しさに集中して議論することができます。

5-3 フローチャートの流れ

ロジックの可視化：処理の流れを示す最終的な「設計図」



なぜ重要か？

フローチャートは、処理全体の流れと分岐を一目で把握できるため、ロジックの誤りや考慮漏れを発見しやすくなります。コーディング前のレビューや、非エンジニアを含むチームメンバへの仕様説明において、絶大な効果を発揮するコミュニケーションツールです。

思考の解像度を高める3つのツール

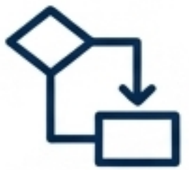
適切な場面で、適切な表現方法を使い分ける

文章

- **役割:** アイデア共有、初期の合意形成
- **特徴:** 直感的、平易、非専門家にも理解可能
- **課題:** 曖昧さが残りやすい

擬似コード

- **役割:** ロジックの構造化、実装への橋渡し
- **特徴:** 構造的、言語非依存、厳密なロジック記述
- **適した場面:** 実装前のロジックレビュー



フローチャート

- **役割:** プロセスの可視化、仕様のドキュメント化
- **特徴:** 視覚的、網羅的、分岐やループが明快
- **適した場面:** 複雑なロジックの設計、仕様説明

第5章のまとめと次章への繋がり

本章の学び

- **表現の多様性:** 1つのアルゴリズムが、目的に応じて複数の形式（文章、擬似コード、フローチャート）で表現されることを理解した。
- **思考のプロセス:** 日常的な「言葉」から、コンピュータが実行可能な「設計図」へと、思考を段階的に具体化・構造化するプロセスを学んだ。
- **ツールの役割:** 各表現方法が開発プロセスの異なる場面で果たす役割と価値を理解した。

次章（第6章 よく使われる処理の種類）へ

アルゴリズムの「表現方法」という強力なツールを手に入れた今、次章では、それを使って表現される具体的な処理の構成要素である「順次処理」「分岐処理」「繰り返し処理」を学びます。
今回学んだフローチャートなどを使い、より複雑な問題解決に挑戦していきましょう。