

---

# 論理思考とプログラミング

## 第5回－2時限目

---

アルゴリズム概念の理解と構築

Logical Thinking



# 2時限目の目標

Logical Thinking

- 手作業で理解した(並び替え)アルゴリズムをプログラムとして書けるようになる

Logical Thinking

# まとめ: 最小値選択法の効率計算

## ○ 並び替えるカードの枚数と比較回数

- カードが1枚の場合→0回
- カードが2枚の場合→1回
- カードが3枚の場合→2+1回
- カードが4枚の場合→3+2+1回
- カードが5枚の場合→4+3+2+1回

## ○ カードの枚数がN枚の場合の比較回数

- $(N-1)+(N-2)+(N-3)+\dots+1=N \times (N-1)/2$ 
  - 比較回数はNの2乗に比例して増加する
  - 計算結果は $(N^2-N)/2$ であるが, Nが十分に大きければ,  $N/2$ は無視できる
  - カードの枚数が2倍になると, 時間は4倍かかる

# 講師によるデモ

Logical Thinking

## 第5回 > Project10 > 10.3~10.4

The screenshot shows a programming demo interface with a yellow background. At the top left is a red button labeled "リセット" (Reset). At the top right is a purple box labeled "最小値候補" (Minimum Candidate) containing the number "30". Below this are three horizontal bars representing arrays:

- ソート済束** (Sorted): A yellow bar containing the numbers 12 and 14.
- 検索済束** (Searched): A green bar containing the numbers 99, 83, 79, 60, 67, 58, 98, 55, and 47.
- 未処理束** (Unprocessed): A blue bar containing the numbers 19, 90, 71, 39, 52, 17, 75, and 80.

第5回 > Project10 > 10.3~10.4

# 2時限目の演習範囲

- Project10 並び替えを試してみよう<後半 10.3 ~10.4>
  - テキストの範囲
    - P.123~P.130
  - 指定問題
    - やってみよう No.10-2(P.127)
    - やってみよう No.10-3(P.130)
  - 発展問題
    - やってみよう No.10-4(P.130)



# やってみよう No.10-2

## 概要

- 1限目に手作業で実行した最小値選択法をコンピュータで再現します
- 2人組は解散し、1人で作業をします

## 注意

- テキスト P.123～P.130をよく読む
- 入れ物の使い方に関しては、テキストのP.163「付録 A.17」を参照すること

# 指定問題の取り組み方

- P.123～P.127までを読んで、カードの準備をし、「やってみよう No.10-2」(P.127)に取り組む
- P.128～P.130までを読んで、途中まで実装をして、残りを「やってみよう No.10-3」(P.130)として取り組む

# 実装時に気をつけるポイント

- ① タイルをよく読み, 命令に間違いがないか丁寧に確認する
  - ② 例: 追加先の入れ物を間違える
    - ③ 「●●に××に追加する」という命令のつもりで,  
「●●を△△に追加する」という命令を入れてしまう
- ① フローチャートを確認しながら実装する
  - ② フローチャートを無視して, タイルをいじくりまわすだけでは無理です
- ① 1回だけ実行ボタン **実行** を有効活用する
  - ② 1回実行してうまく行かないものは, チクタク(繰り返し)をしても, 絶対にうまくいかない
  - ③ 1回だけ実行して, 命令がうまく動いていることを確認しながら, 少しずつ作っていくことが大切



# 実装方法について

## ○ 良い方法

- 1. 少し作ったら、テストをする
- 2. テストをしてうまくいかなかったら、一番最近追加した部分が間違っていることが分かる
- 3. 間違いを修正し、意図通りに動いたら1.に戻る(完成まで繰り返し)

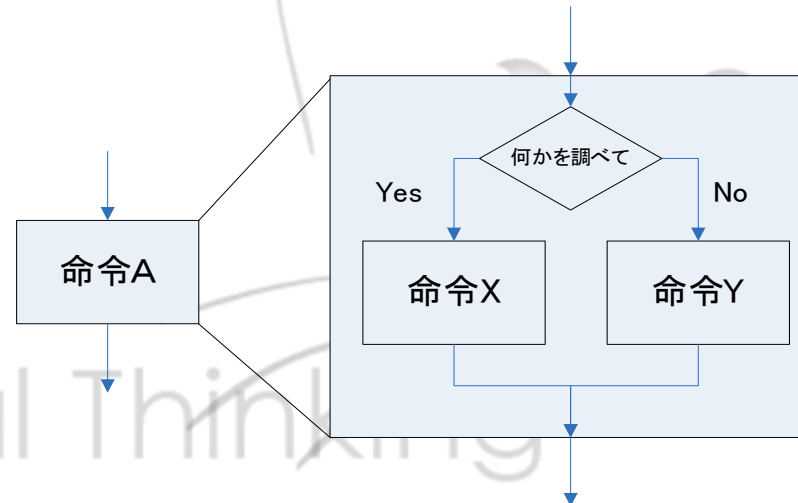
## ○ 悪い方法

- 1. 一度も実行(テスト)をしないで、一気に完成させる
- 2. 実行すると、意図通りに動かない
- 3. どこが間違っているか、正しいか分からない
- 4. 手当たり次第に変更し、混乱してしまう

# 制御構造の組み合わせ(復習)

- 第2回の授業で、制御構造について整理した
  - 複雑な処理を記述する場合は3つの制御構造を入れ子状に組み合わせる
  - どの制御構造も処理の入りが1つで、出口も1つなので可能

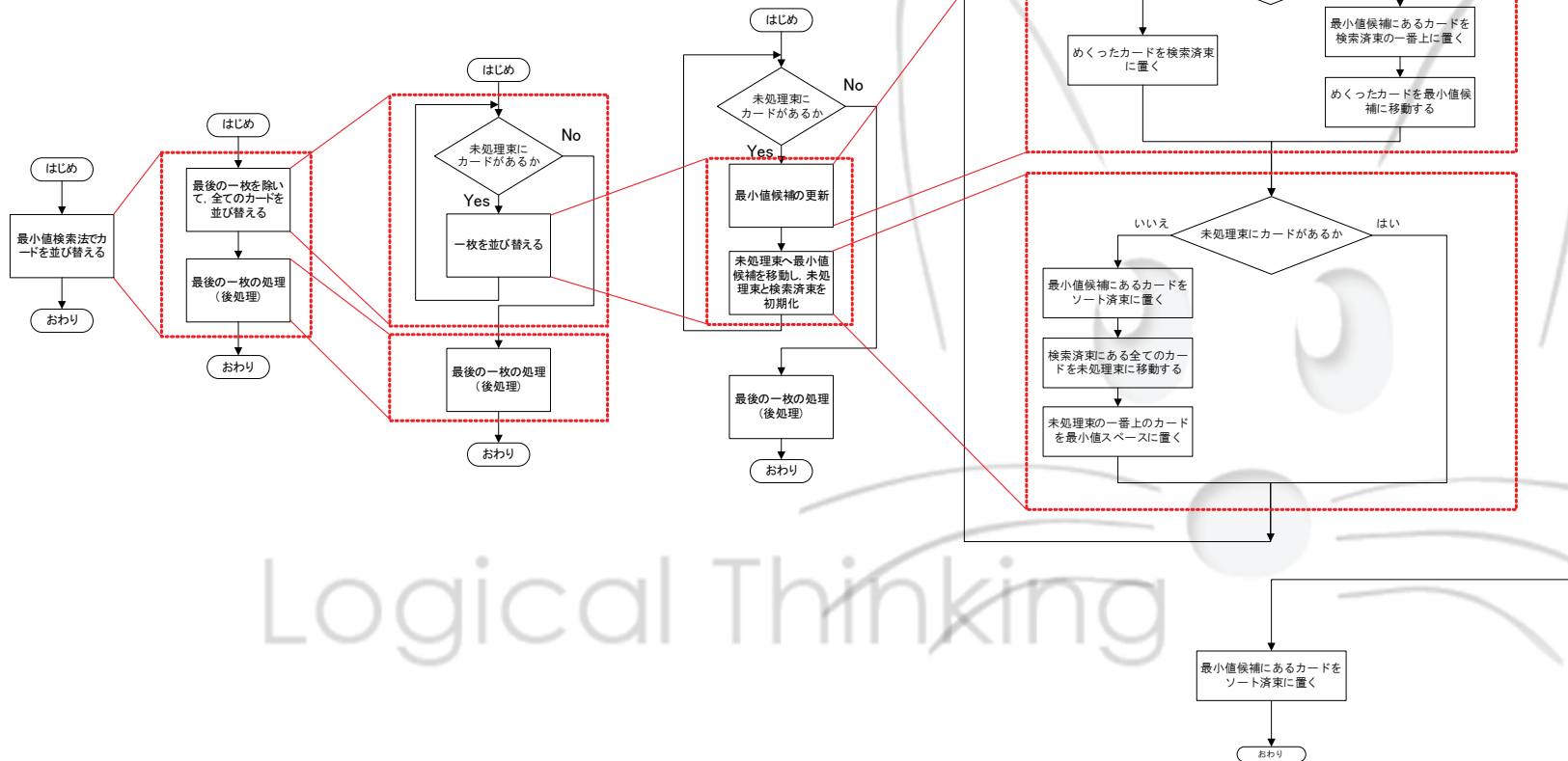
命令Aを入り口1つ、出口1つの箱として、入り口1つ、出口1つの分岐構造に置き換えることが可能



# 最小値選択法の例

Logical Thinking

制御構造を組み合わせれば  
(入れ子にする)実現できる



Logical Thinking