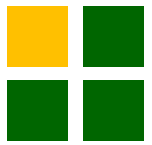


アルゴリズム講座

頭の迷路を書きのこそう！

Ver. 1.1



アジェンダ

- アルゴリズムとは
- 手順について
- サンプル
- アルゴリズムの残し方
- アルゴリズムとプログラム
- 良いフローチャート
- フローチャートについて
- 基本構造
 - 順次構造
 - 選択構造
 - 反復構造
- 演習①・②
- 総合演習1
- 代入
- 計算
- 大小比較
- 単一分岐
- 多方向分岐
- 反復構造
- ループ記号
- 総合演習2・3



アルゴリズムとは

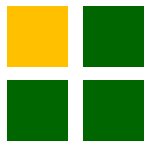
Wikipedeia様のご意見ー

アルゴリズム (英: algorithm [\[ˈælgəˌrɪðəm\]](#)) とは、数学、コンピューティング、言語学、あるいは関連する分野において、問題を解くための手順を定式化した形で表現したものを言う。算法と訳されることもある。

「問題」はその「解」を持っているが、アルゴリズムは正しくその解を得るための具体的手順および根拠を与える。さらに多くの場合において効率性が重要となる。

コンピュータにアルゴリズムをソフトウェア的に実装するものがコンピュータプログラムである。人間より速く大量に計算ができるのがコンピュータの強みであるが、その計算が正しく効率的であるためには、正しく効率的なアルゴリズムに基づいたものでなければならない。

「アルゴリズム」『フリー百科事典 ウィキペディア日本語版』(<http://ja.wikipedia.org>) 2017年8月11日 09:19 UTC



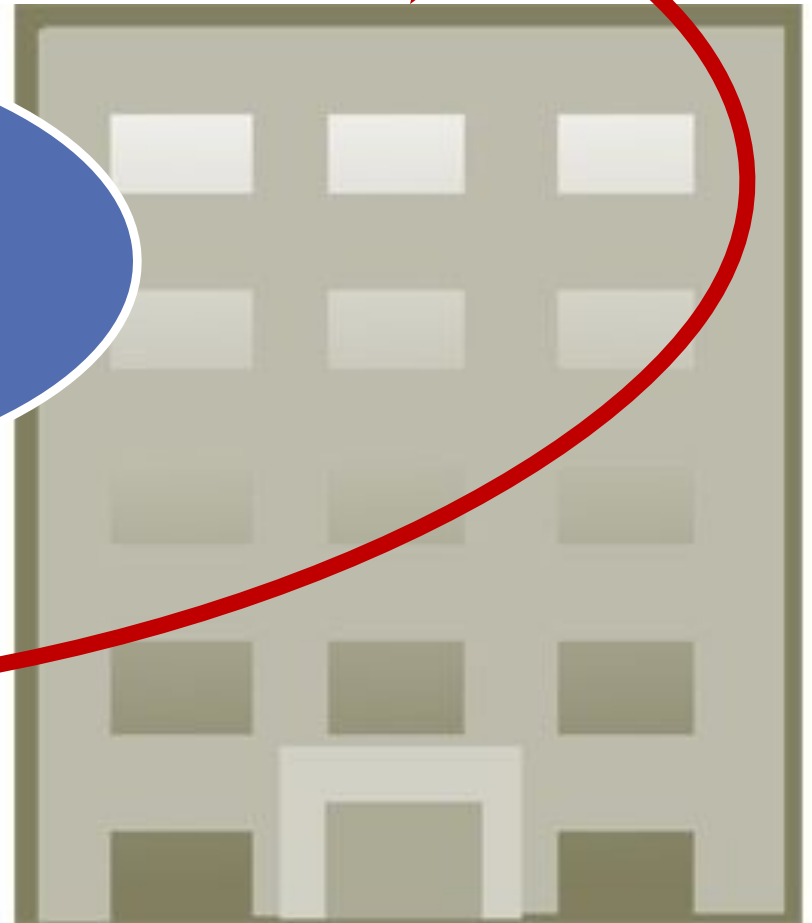
難しい(汗)



アルゴリズムとは

実は日常に溢れています！ 先ずはそれを体験して頂きます。

屋上に行くには？





アルゴリズムとは

オムライスを
作るには？





アルゴリズムとは

日常に溢れている！



アルゴリズムとは

つまり手順です。手順とはやり方とも言えますね。

手 → 手続き(やること)

順 → 順序



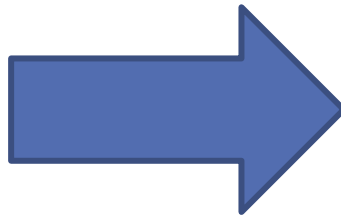
ここで特に重要視して欲しいのは・・・**順序**です。

例えばフライパンでハンバーグを焼く手順を考えてみましょう。

①火をかける

②油を敷く

③ハンバーグを入れる



①油を敷く

②ハンバーグを入れる

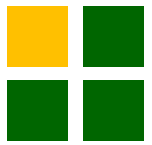
③火をかける

上記の例で味に変化が出そうというイメージつきますよね？



アルゴリズムとは

何を
どのような順番で
何に対して行うのか



手順について

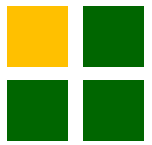
- 重要なこと

曖昧さを排除！

誰がやっても同じ結果になります。

いつでも正しい結果！

大抵うまくいくではダメです。

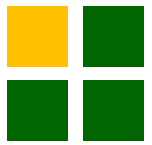


サンプル

この手順の問題点を考え、挙げてみましょう

- ①具材を炒める
- ②味付けに塩を少々
- ③盛り付ける





アルゴリズムの残し方

アルゴリズムは脳で考えているもの

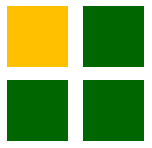
だから...

- ①他人と共有できない
- ②考えた本人も忘れちゃう

という問題点があります。



そのため、何らかの方法で、考えたアルゴリズムを残さないといけないわけです。



アルゴリズムの残し方

文字で書き残す

プログラミング言語で書き残す

図で書き残す

推奨



アルゴリズムの残し方

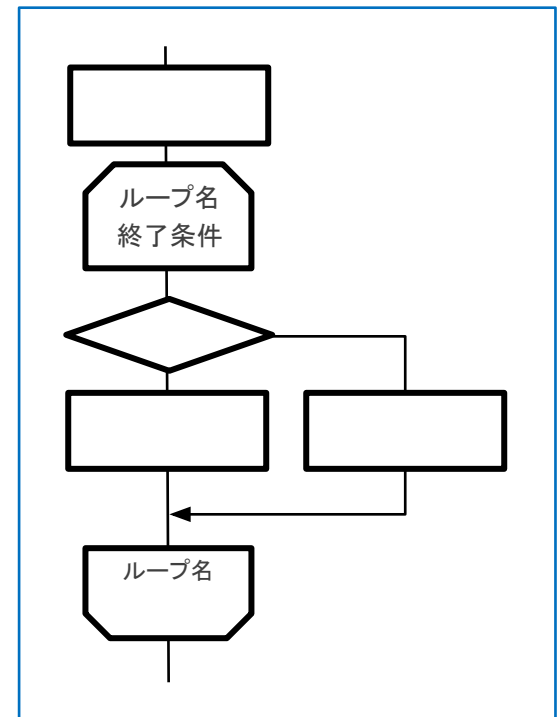
先ほどの「図で残す」場合、よく使用されるのが、

流れ図(フローチャート)

と呼ばれるものです。

JIS X 0121(日本工業規格の1つ)で規定されており、一般的に用いられています。

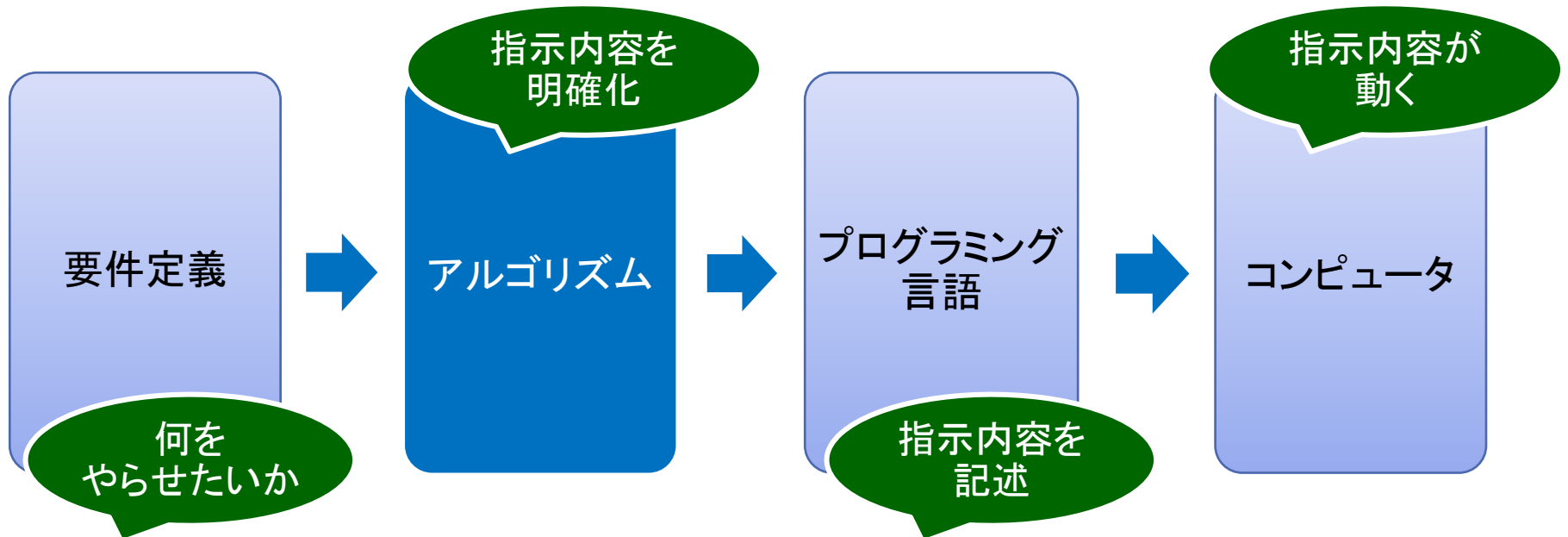
流れ図以外にも表現方法がありますが、使用頻度としては流れ図が一般的に用いられます。





アルゴリズムとプログラム

なぜアルゴリズムを勉強しないといけないのかというと・・・





良いフローチャート

これからフローチャートの書き方を説明しますが、以下のことを意識して取り組みましょう。

曖昧さを排除！




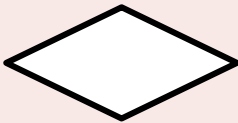
いつでも正しい結果！

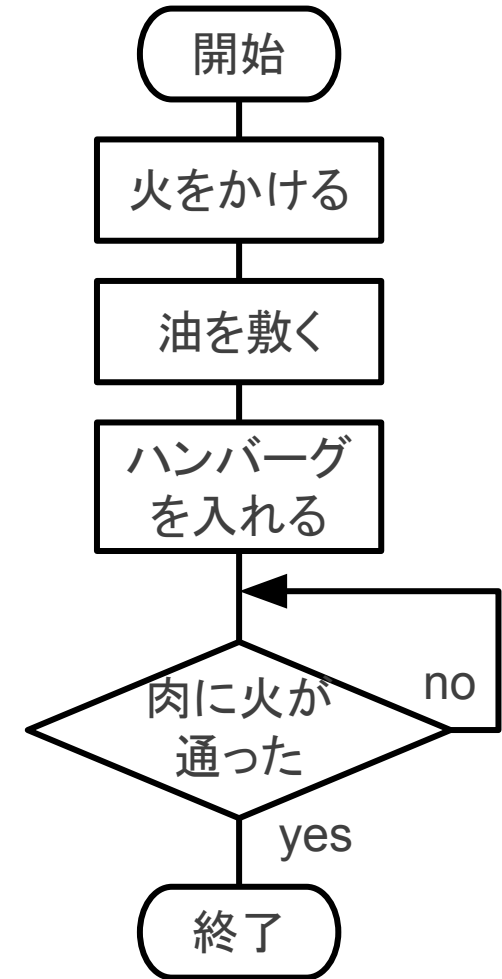
入口1つ、出口1つ！

必ず終わる！



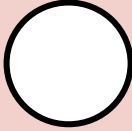
フローチャートについて

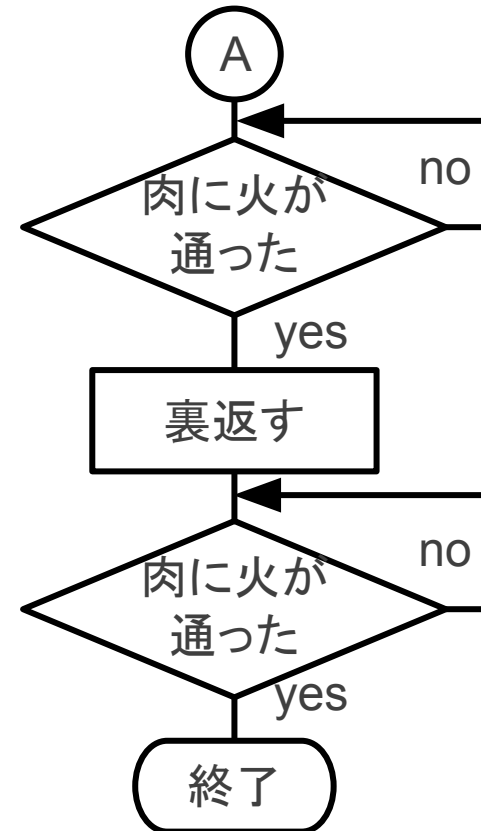
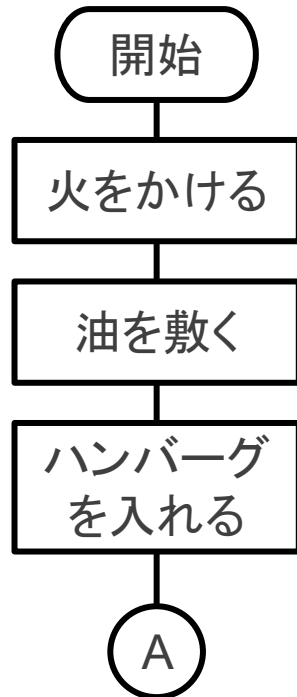
記号名	記号	概要
端子		入 口（開 始）・出 口（終 了）を表します。 基本的に開始・終了は1つずつとなります。
線		データ・制 御 の 流 れを表します。 流 れの 向 きを 提 示する 必 要が あるときは 矢 先を 付 けなけれ ば いけません。
処理		任意の処理機能を書きます。 中 に 複 数 の 処 理を 書くと きの 順 序に 注 意して くだ さい。
判断		記号中に定義された条件の評価に従って流れの向きを選びます。 評価結果は経路を表す線に近接して書きます。 主に2つ以上の流れがある場合に使用します。





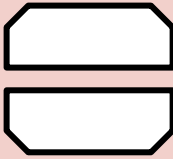
フローチャートについて

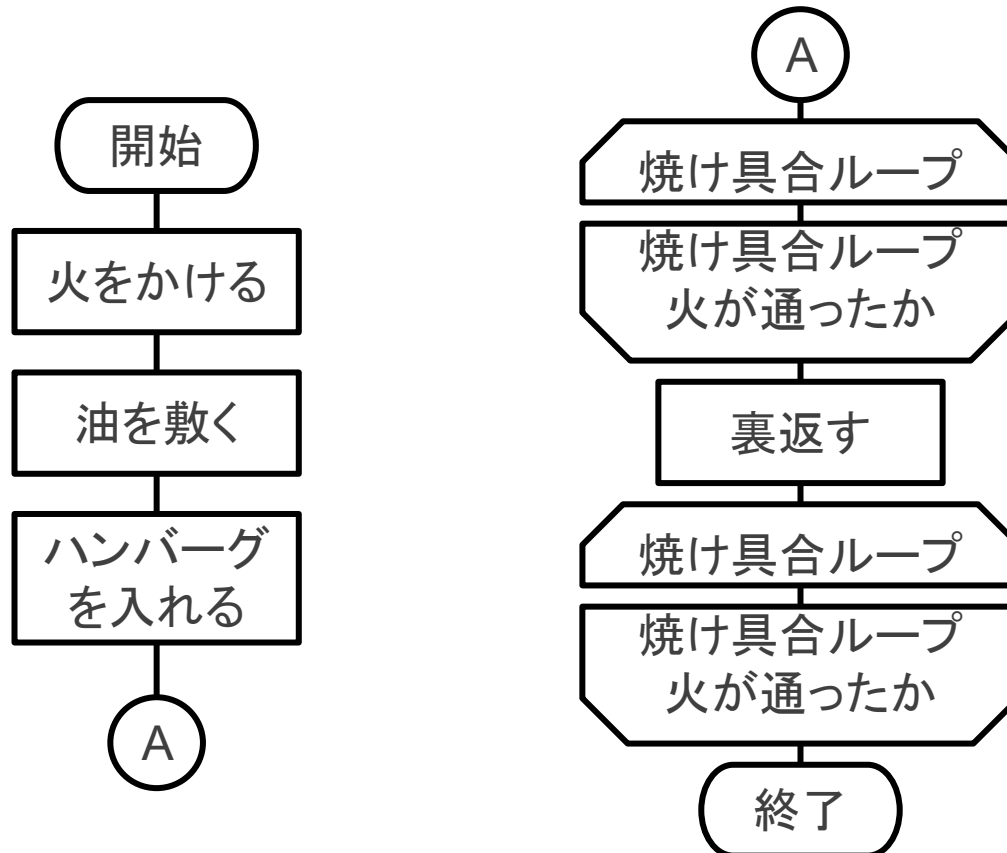
記号名	記号	概要
結合子		線が交差したり長くなりすぎたりするのを避けます。 または図が2ページ以上に及ぶ場合に使用します。






フローチャートについて

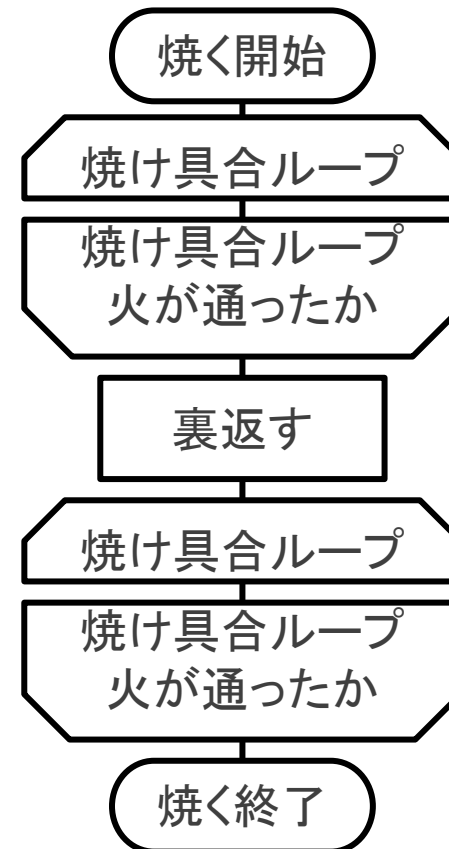
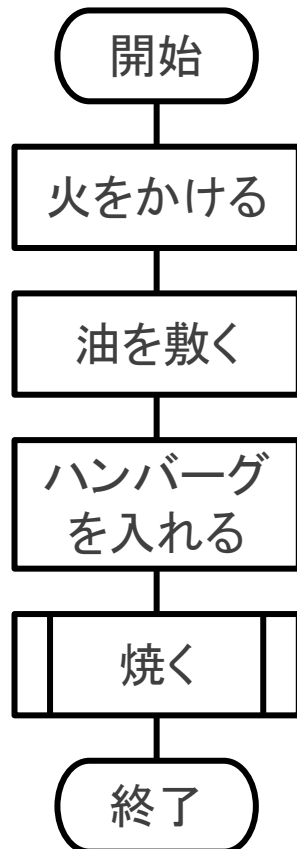
記号名	記号	概要
ループ端		ループの始まりと終わりを表します。 同じ名前を持ち、始端または終端の記号中に終了条件を表記します。(前判定、後判定)

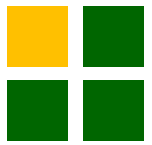




フローチャートについて

記号名	記号	概要
定義済み処理		別の場所で定義された処理を表します。 例: メソッド

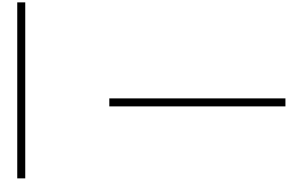




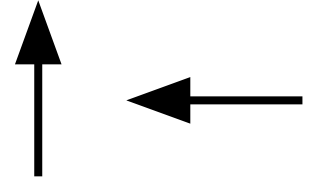
フローチャートについて

- 書き方の作法

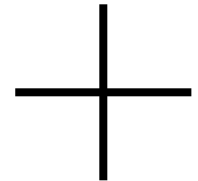
上→下 または 左→右 が基本



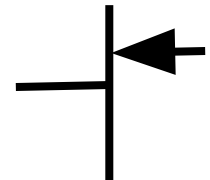
下→上 または 右→左 は矢印を使用



線の交差は極力書かない



線の合流はずらす





基本構造

アルゴリズムの3つの基本構造は以下の通りです。

順次

処理が上から下に
順番に並んでいる
基本的な構造

選択

条件によって
処理が分かれる構造

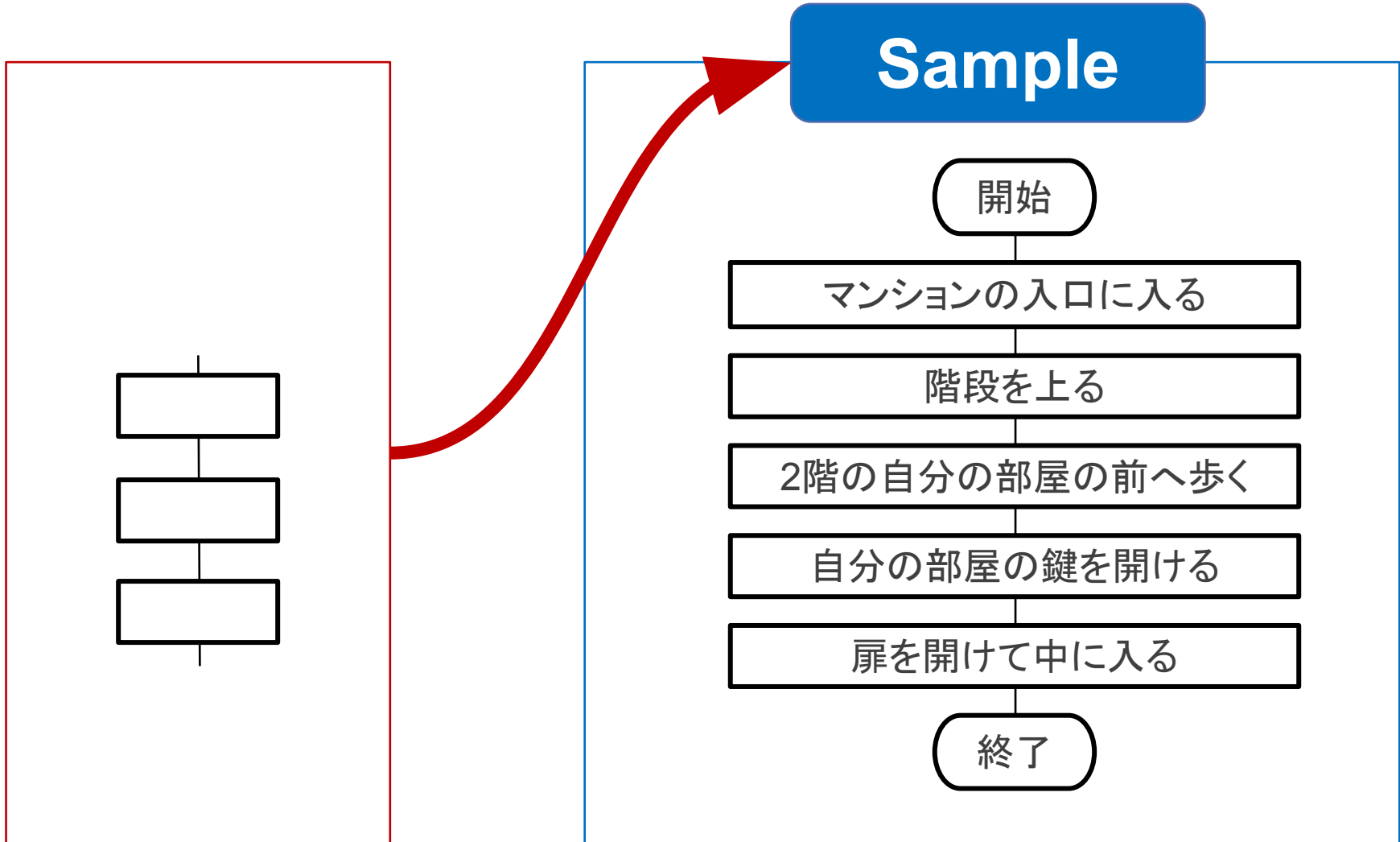
反復

条件を
満たしている
(満たしていない)あい
だは処理を実行する
構造



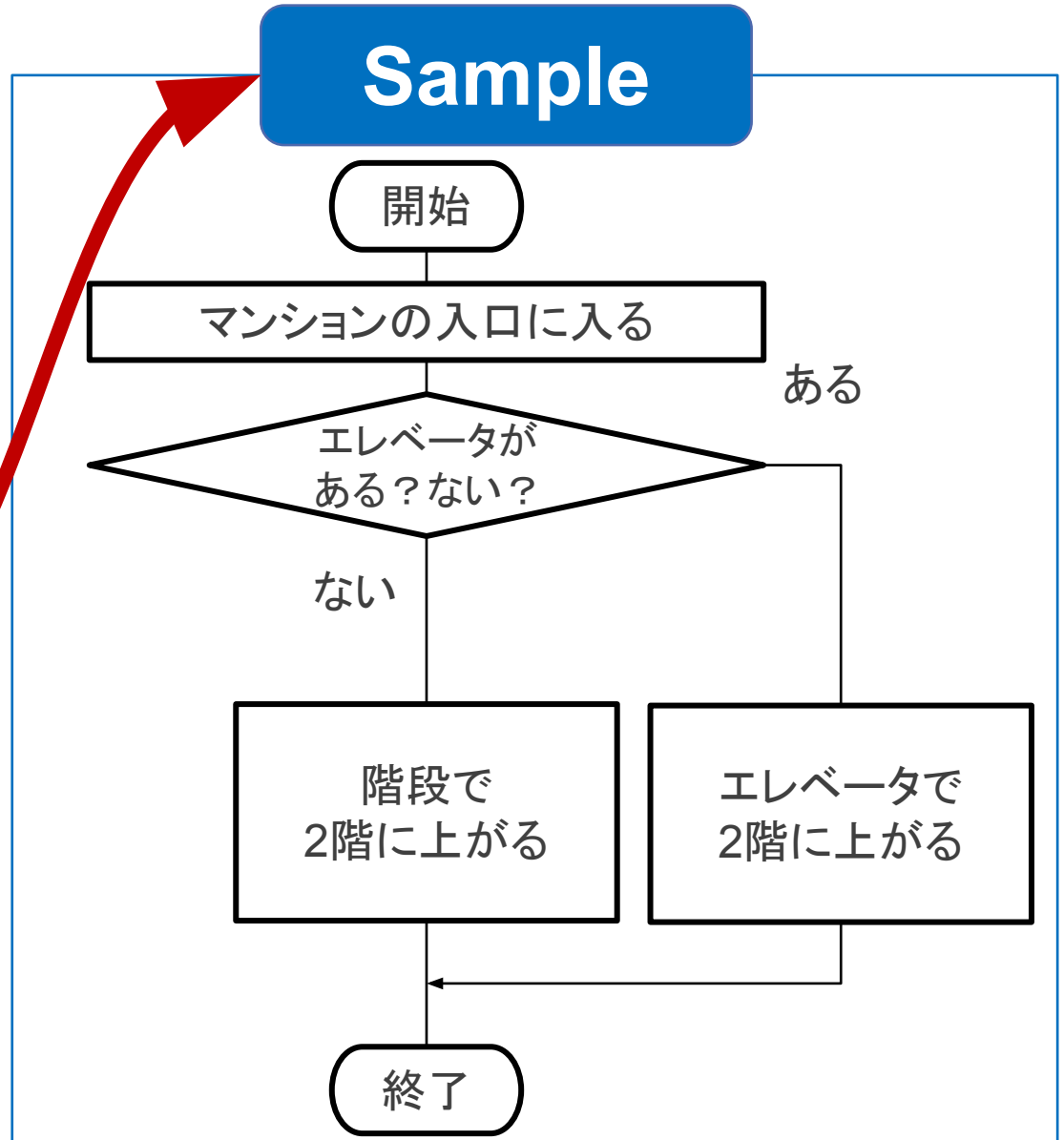
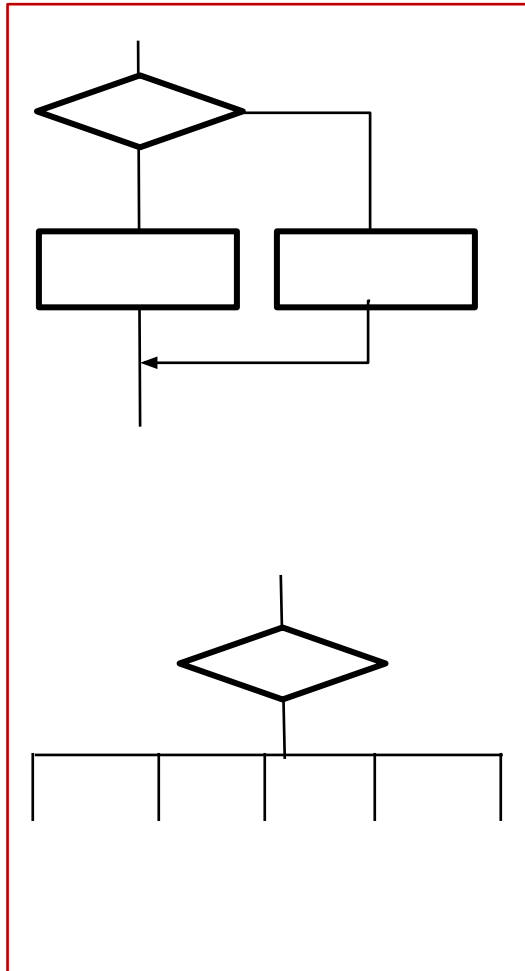
順次構造

順次構造は上から下に処理が並ぶ構造です。





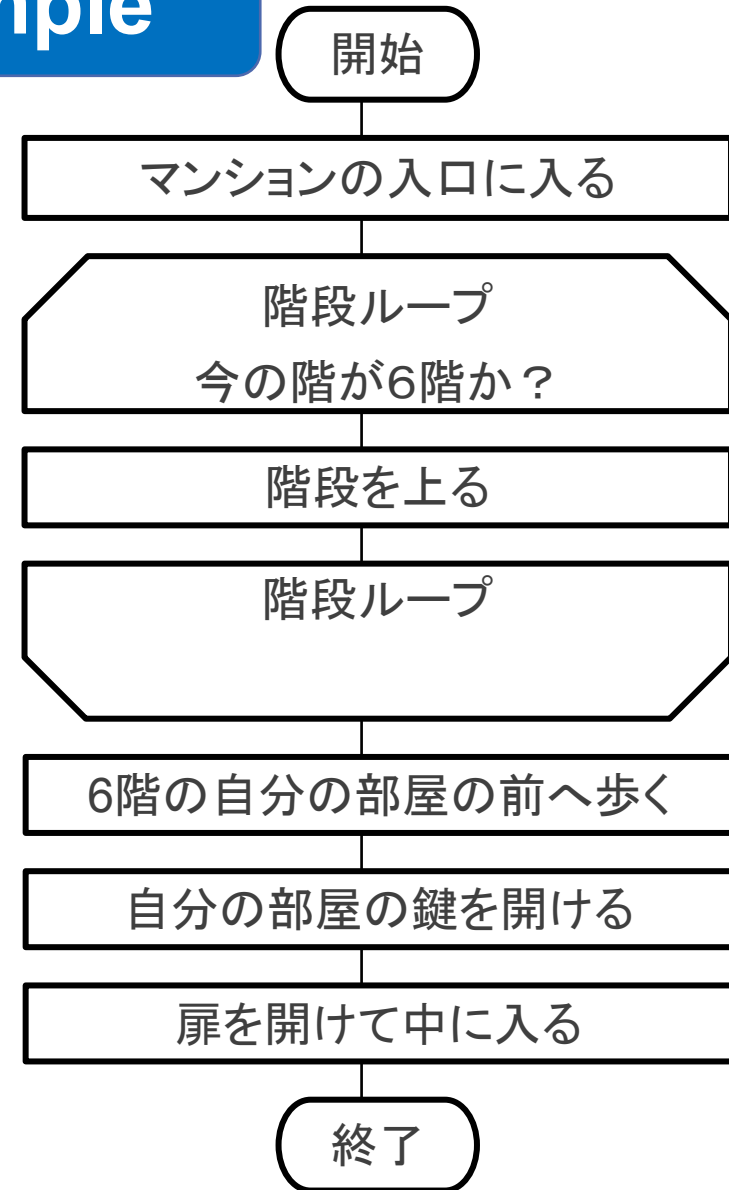
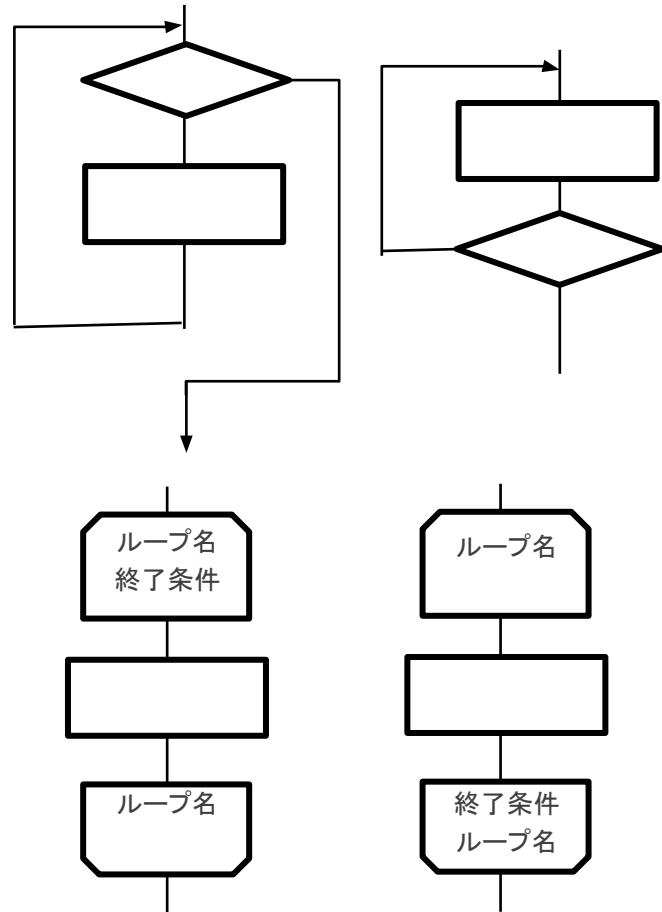
選択構造





反復構造

Sample





演習①

お湯を入れて3分で出来上がるカップラーメンがあります。

このラーメン調理方法の説明を、自分が書くとしたらどのように書くか考えて発表してください。





演習② 例題

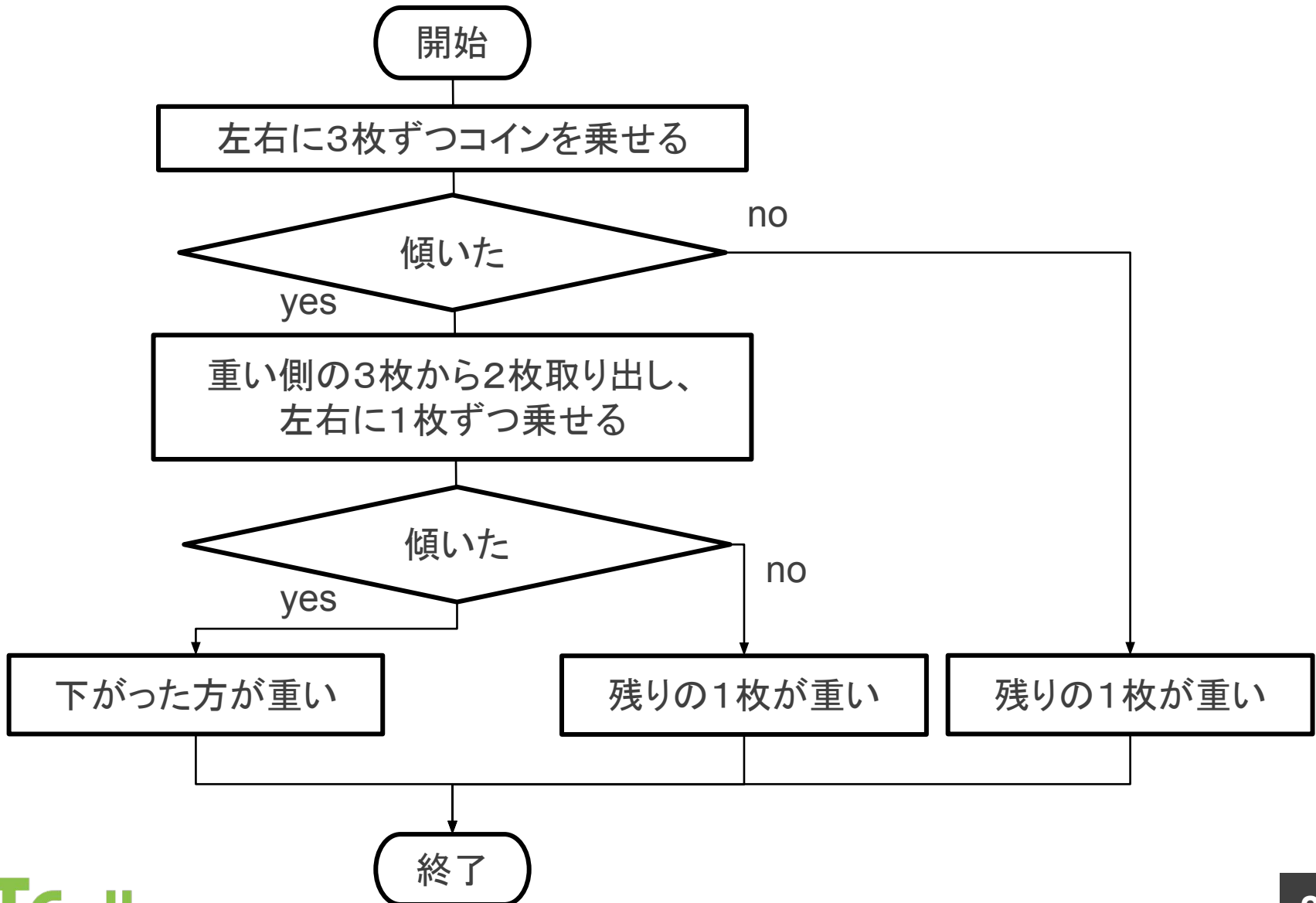
7枚のコインがあり、1枚だけ他のコインより重いコインが混じっています。

天秤を2回だけ使って重さの違うコインを特定するにはどうしたら良いか、流れ図を使って表現しましょう。





演習② 例題解答





演習②

9枚のコインがあり、1枚だけ他のコインより重いコインが混じっています。

天秤を2回だけ使って重さの違うコインを特定するにはどうしたら良いか、流れ図を使って表現しましょう。





総合演習1

オリジナルの自動販売機を考えて流れ図で表現しましょう。

流れ図を書く前に以下のことを行ってください。

・機能列挙

【例】

商品選択機能:購入する商品を選択する機能

入金機能:購入時にお金を入れる機能

など

・オリジナル機能の検討

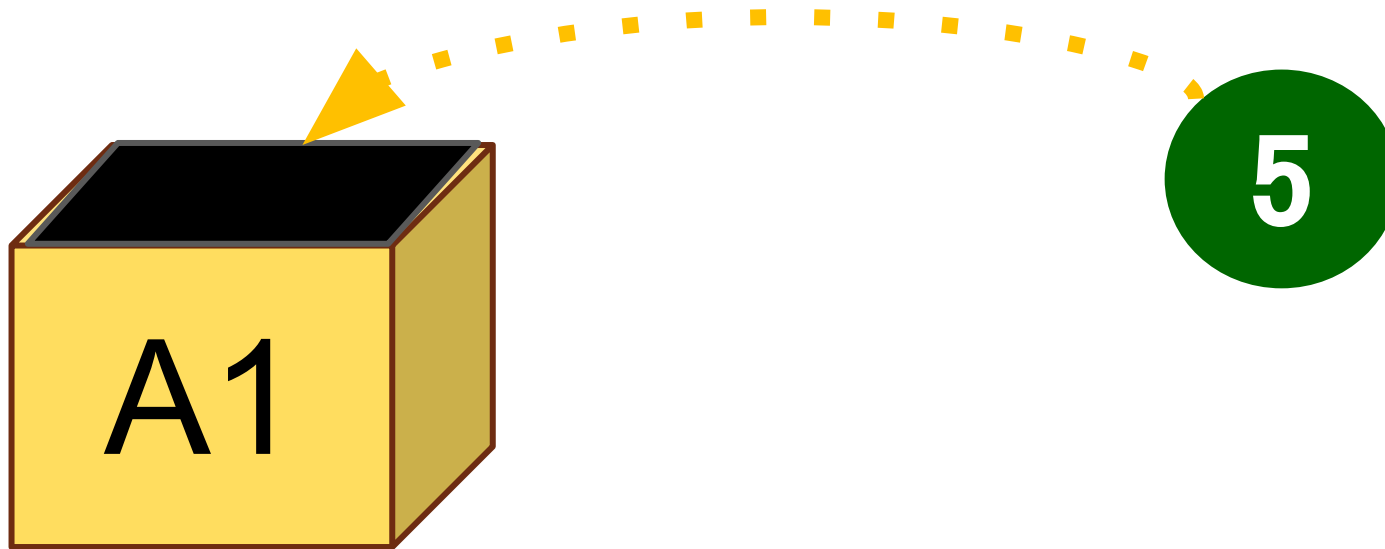
あとで流れ図を説明してもらいますので、オリジナル機能は特にアピールしてください。



代入

箱に値を入りたい場合は、「**A1←5**」のように記述します。

イメージは以下の通りです。



この箱のことを**変数**と言います。

また、箱についている名前を**変数名**と言います。



計算

プログラム言語で計算を行う時は、以下のような記号を使うのが一般的です。

加算

+

乗算

*

減算

-

除算

/

$a + b \rightarrow c$

aという箱とbという箱に入っている値を足してcという箱に代入する

$a / b \rightarrow c$

aという箱とbという箱に入っている値を割ってcという箱に代入する

ディスプレイに入力や出力をしたい場合は、記号で厳密に区別する必要はなく、～を入力のように記述すれば良いです。



演習

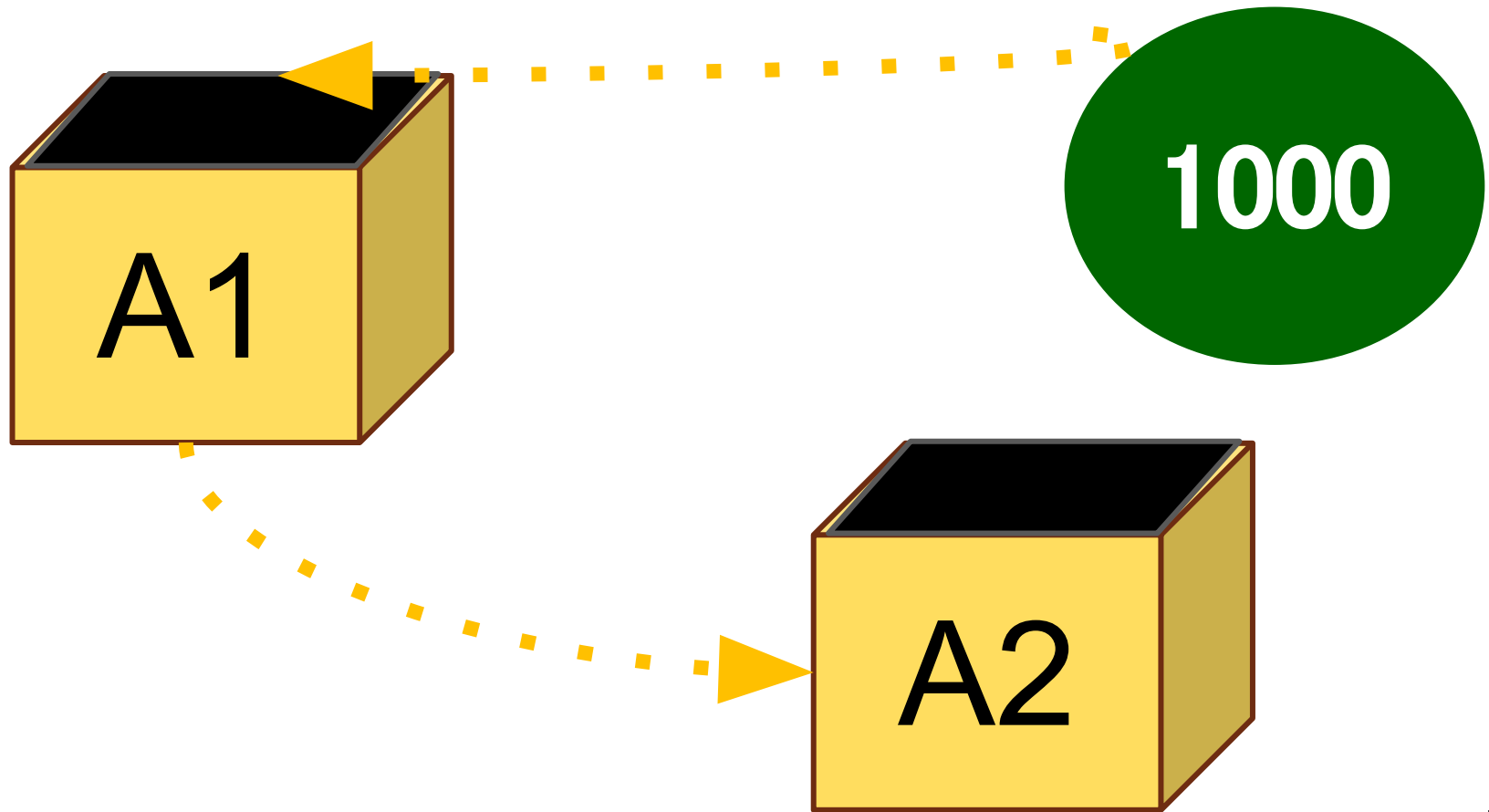
A1に値1000を入れるアルゴリズムを記述しましょう。





演習

A1に値1000を入れて、その後A1の値をA2の箱に代入するアルゴリズムを記述しましょう。

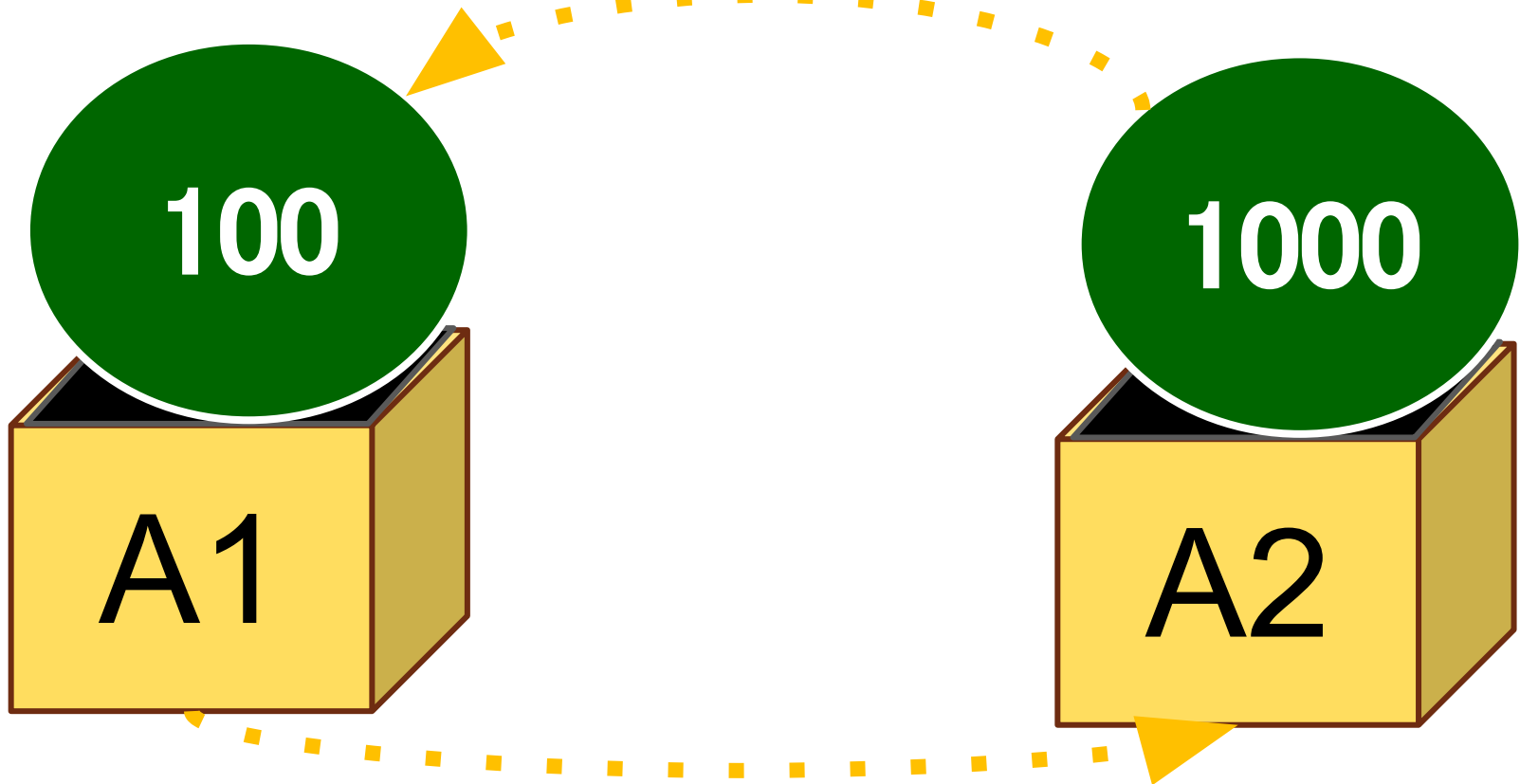




演習

A1に入っている値100とA2に入っている値1000を入れ替えるアルゴリズムを記述しましょう。

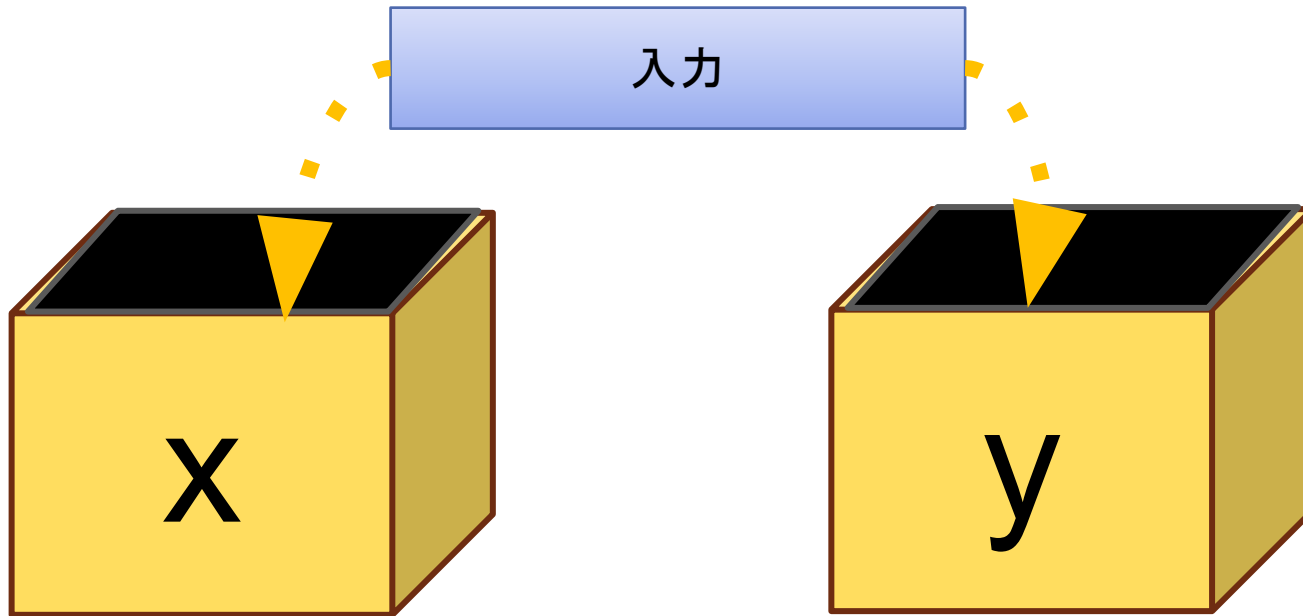
ヒント:A1をA2に、A2をA1に代入するだけでは、A2をA1に代入する前に、A2の値が壊れてしまうことに注意





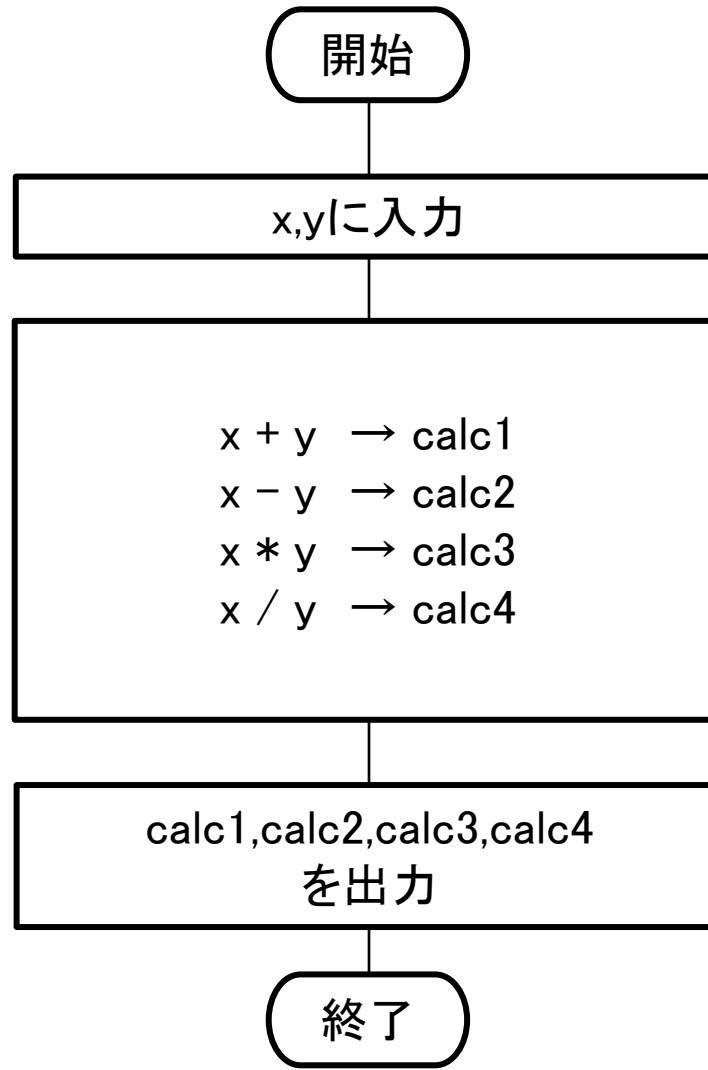
演習

変数 x, y に数字を入力して加算、減算、乗算、除算を行い、結果を表示するアルゴリズムを記述しましょう。





演習サンプル





大小比較

選択構造では左辺と右辺の値を大小比較して、選択方向を決めることが多いです。

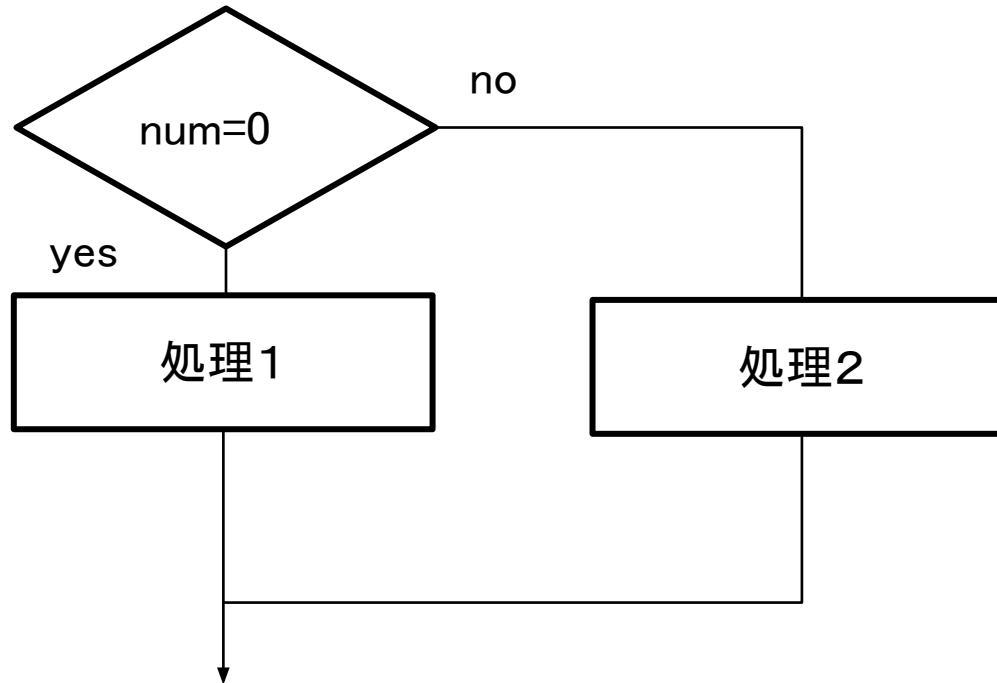
そのため以下のような記号を用い、大小比較を行います。

<	小さい(未満、以上でない)
≤	以下(大きくない)
=	等しい
≥	以上(小さくない)
>	大きい(超過、以下でない)
≠	等しくない
<>	等しくない

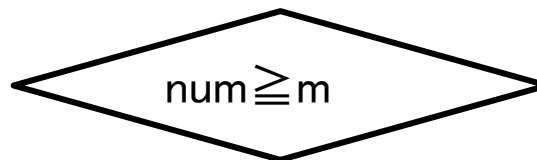


単一分岐

「変数numが0なら処理1をおこない、それ以外なら処理2を行います」という分岐処理を作成したいなら以下のように書きます。



numがm以上ならという条件にしたい場合は以下のように記述します。



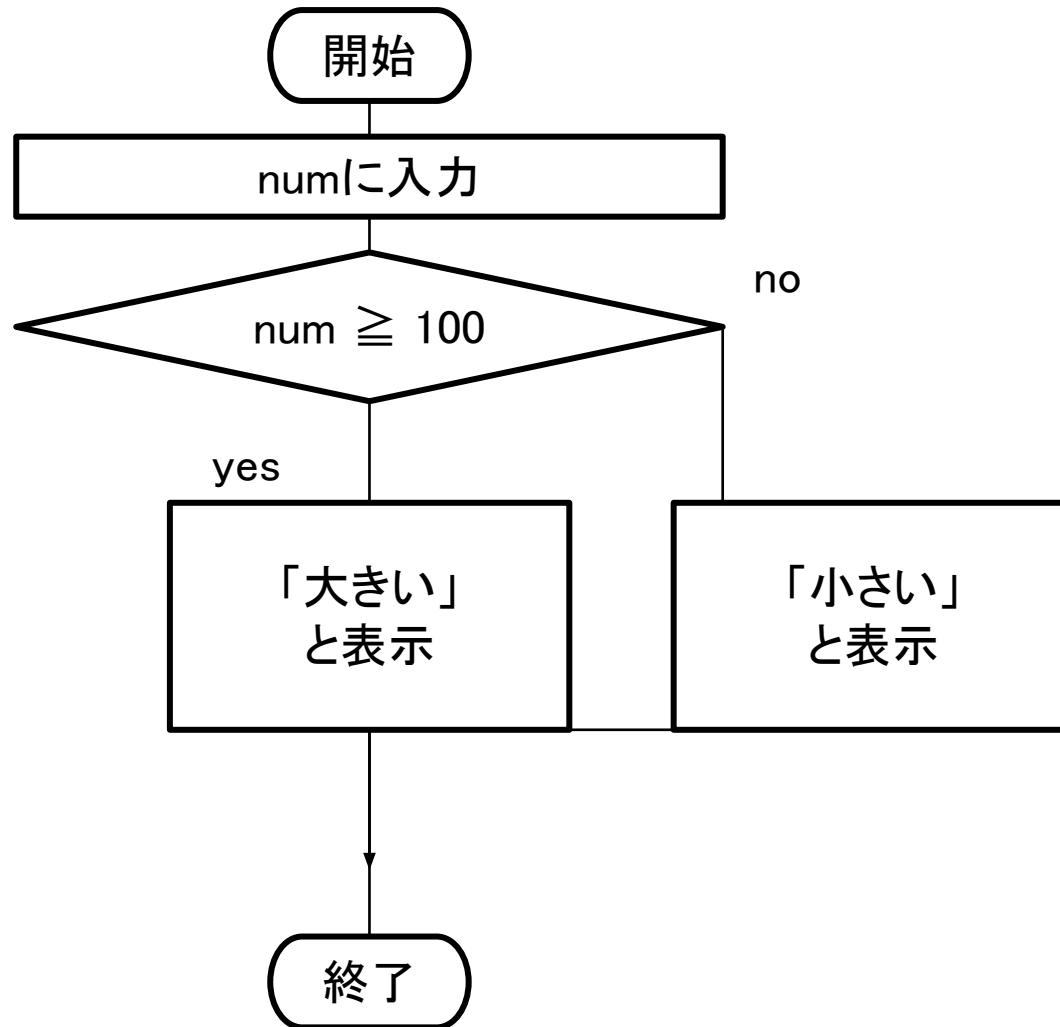


演習

数字を入力してその値が100以下なら小さい、100以上なら大きいと表示するアルゴリズムを記述しましょう。



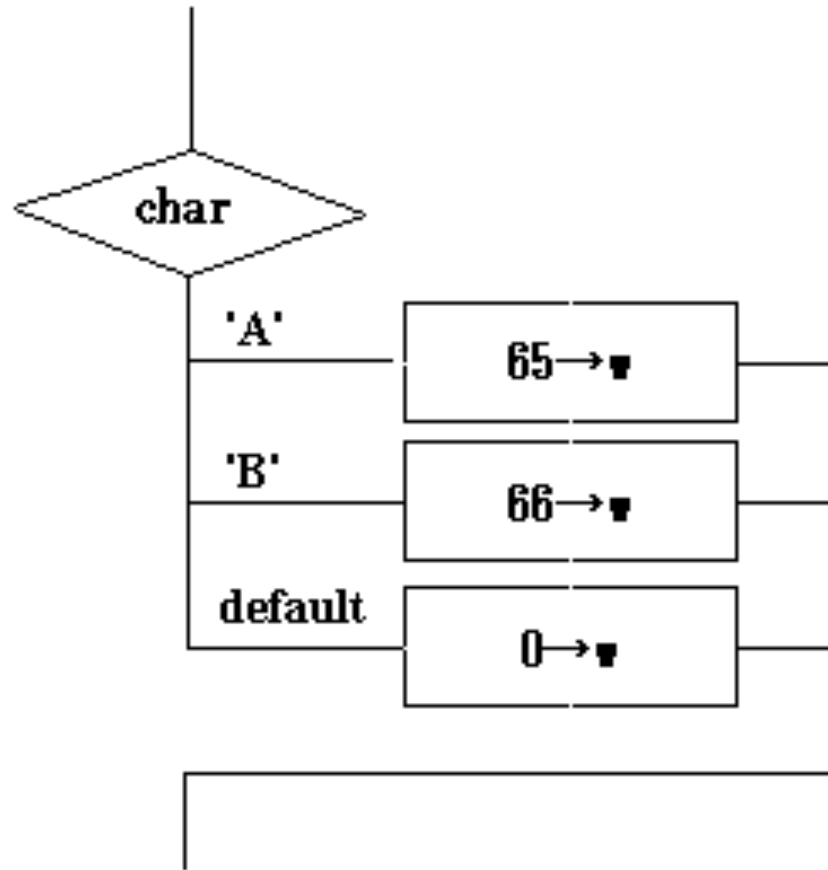
演習サンプル





多方向分岐

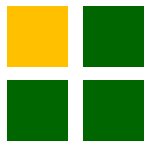
charという箱に入っている値がAの場合、wという変数に65を代入し、Bの場合は66を代入、それ以外の場合は0を代入するという記述方法です。



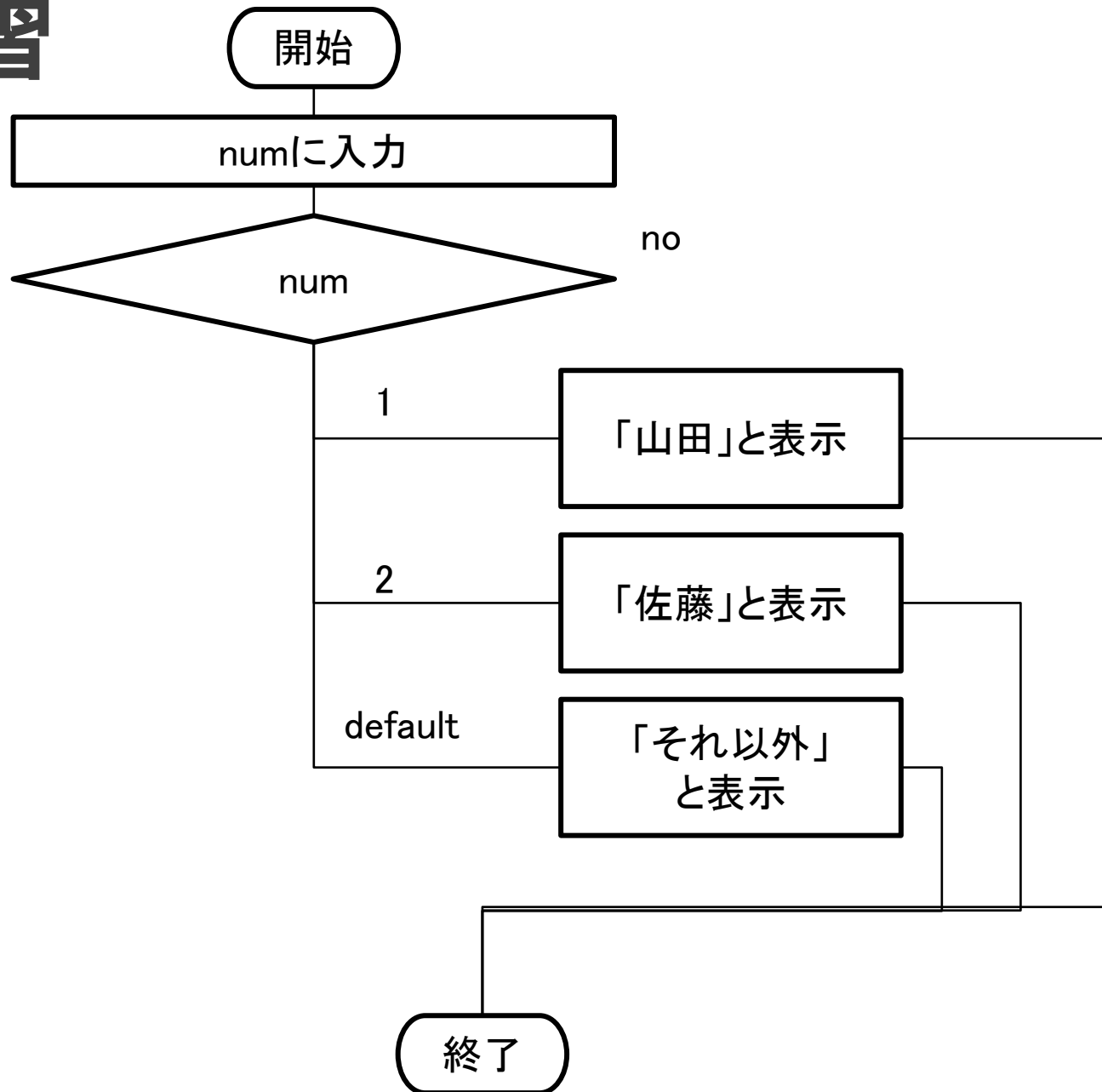


演習

1を入力したら「山田」、2を入力したら「佐藤」、それ以外なら「それ以外」と表示するアルゴリズムを記述しましょう。



演習

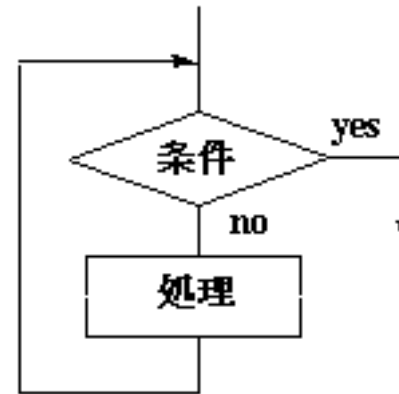
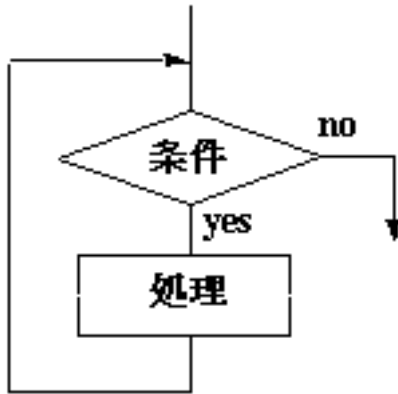




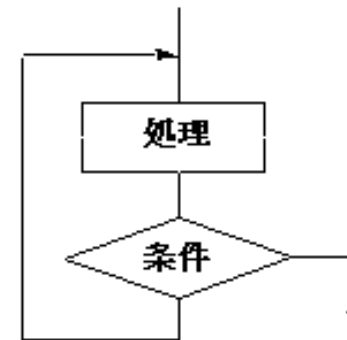
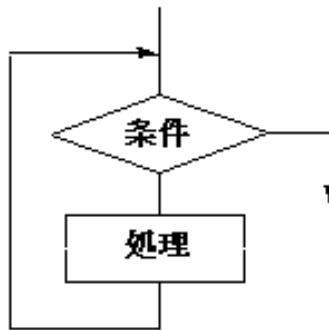
反復構造

まずは2つのことを意識しましょう。

まず1つ目は、「条件を満たす間繰り返す」のか「条件を満たすまで繰り返す」のかです。



次に前判定か後判定かを決めましょう。



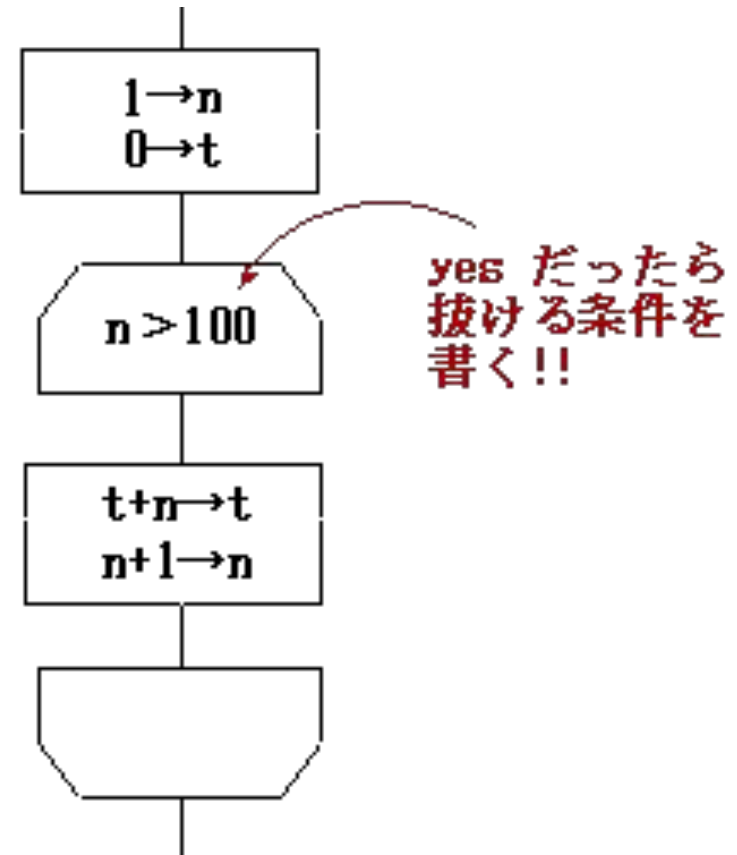
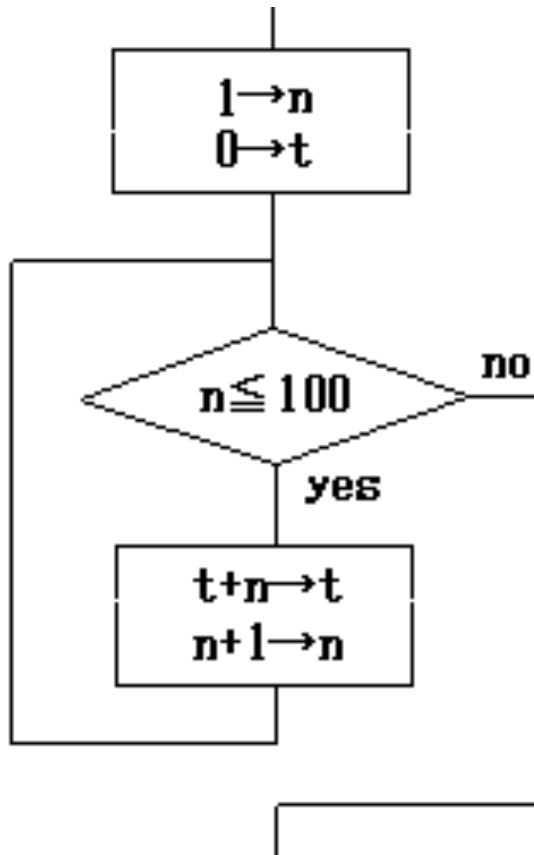
前判定型

後判定型



ループ記号

以下はループ記号を用いた書き方であるが、1から100まで値の合計を求めるフローチャートとなります。どちらでもやっていることは同じです。





演習

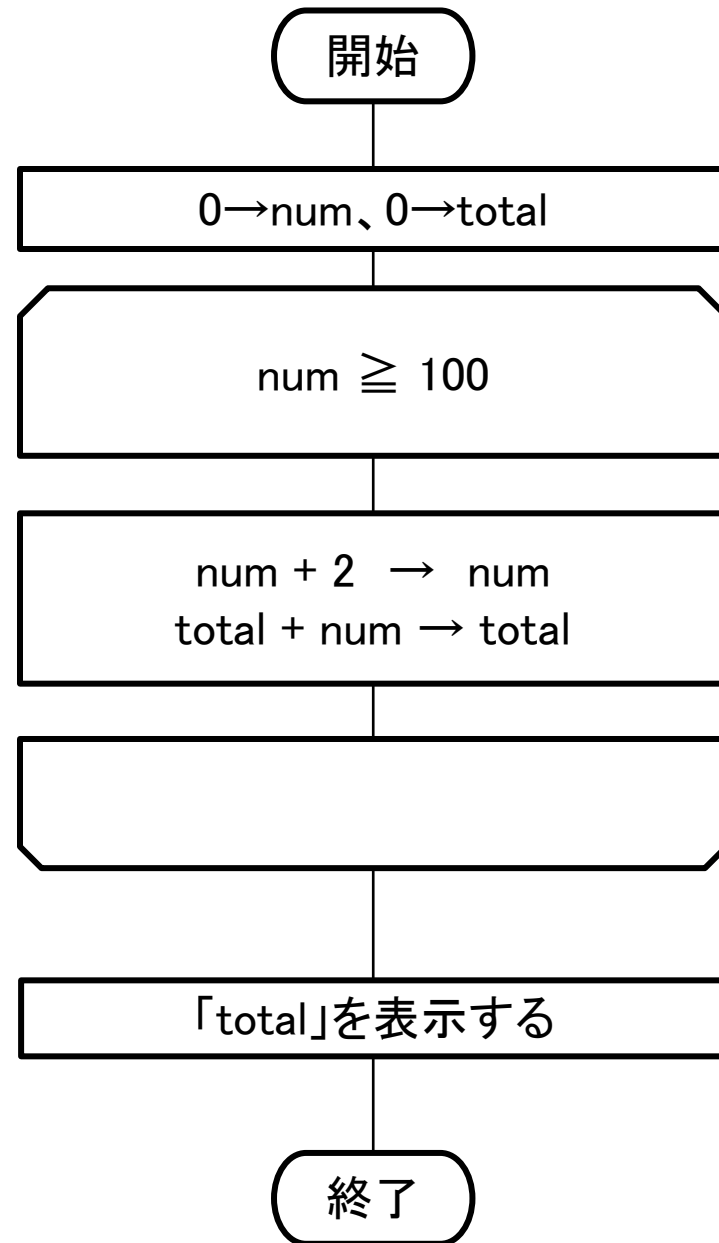
1から100までの偶数の合計を出すアルゴリズムを記述しましょう。



演習サンプル

余裕があれば

他のパターンも作ってみましょう。





演習

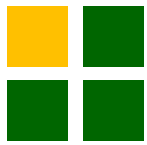
2つの数字a,bをキーボードから入力しaからbまでの合計を求めるアルゴリズムを記述しましょう。

ヒント:

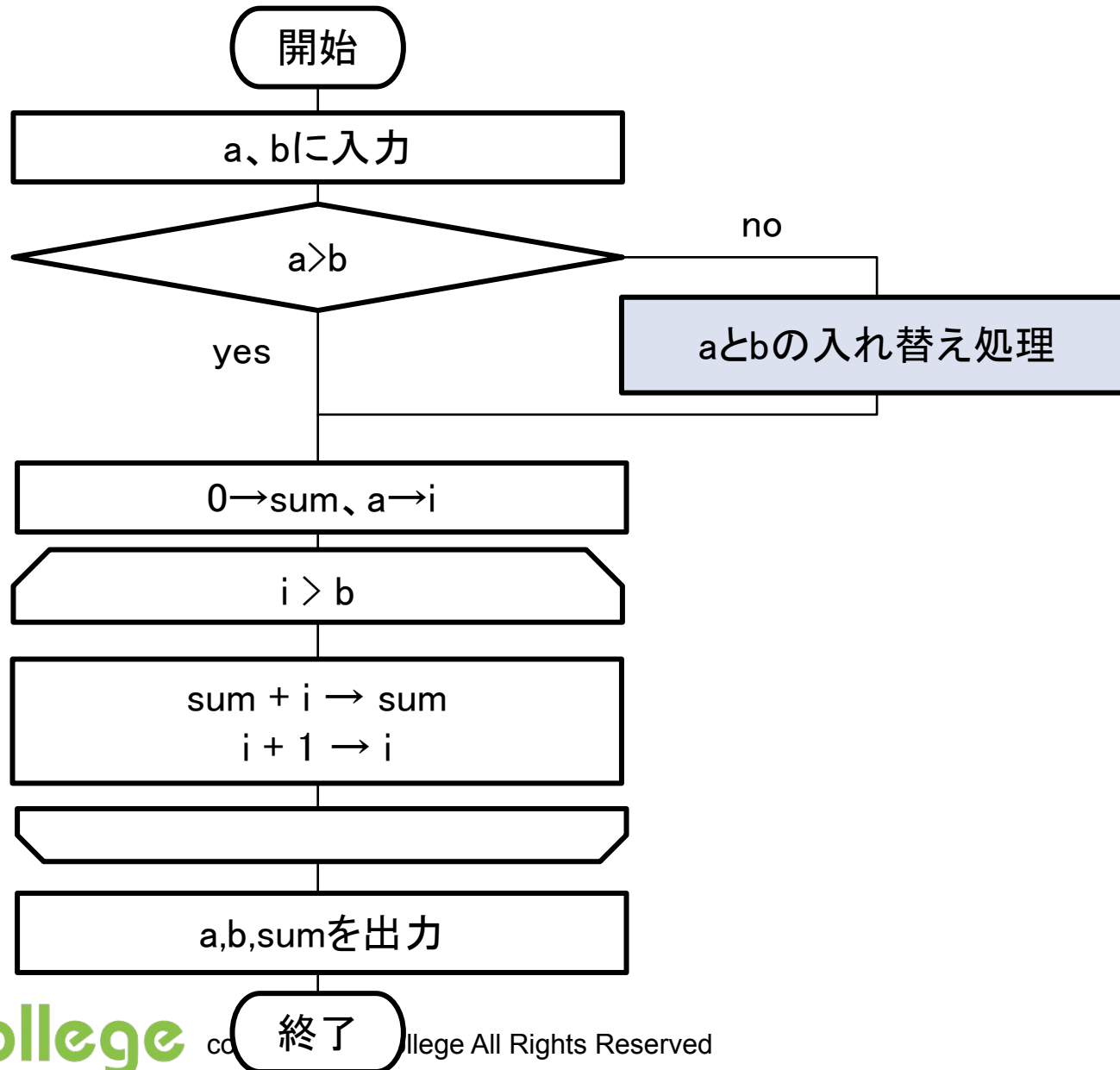
つねに、 $a < b$ とは限りません。

人が行う動作には間違えもありますが、プログラムはどんな条件でも暴走したりしてはいけません。

つまりそれを作るプログラマーは **どんな場合があるかを常に気にとめる必要**があり、今回のように、ただ単に、aからbまでを考えるのではなく、aからbまでという場合も考慮する必要があります。



演習サンプル





総合演習2

前回の振り返り点を含め、個人で再度オリジナルの自動販売機を考えて流れ図で表現しましょう。

流れ図を書く前に以下のことを行ってください。

・機能列挙

【例】

商品選択機能:購入する商品を選択する機能

入金機能:購入時にお金を入れる機能

など

・オリジナル機能の検討



総合演習3

Scratchによるプログラミング体験を行なってみましょう。

https://scratch.mit.edu/projects/editor/?tip_bar=home

ブロックを置いて全体をクリックすると動作します。

