Introduction to
the New Information World

第5章 計算とプログラム

本章の目的

- モデル化した問題を”計算”して解く
- 計算
  - モデル化した問題に対する操作
- 本章で説明すること
  - 計算の概観と記述法
  - 代表的な計算モデル
  - プログラムとプログラム言語

計算とその記述方法

- **計算の例：計数(counting)**
  - ある集合Aの要素数を求める
- **計算のやり方**
  - Aのデータモデルとして，どのような処理が用意されているか，に依存
- **計数のやり方の種類**
  - 取り出し型
    - 要素を1つ1つ，指折り数えていくやり方
  - 分割型
    - 自分の手に余る仕事を下請けに出していくやり方

計算の方法 – 取り出し型

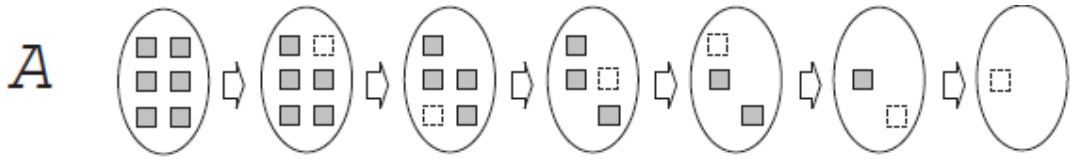
- 集合に対して用意されている処理
  - 空（要素が1つもない）かどうかを判定する
  - 要素を1つ取り出す（集合の要素数は1だけ減る）
- 計算
  - <答>をゼロにする
  - A が空でないあいだ，以下の処理を繰り返す
    - 要素を1つ取り出す
    - <答>を1増やす

計算の方法 – 分割型

- 集合に対して用意されている処理
  - 空か，要素が1つだけであるかを判定する
  - 空でない2つの集合に分割する
- 計算
  - Aが空なら〈答〉は0，要素数が1なら〈答〉は1
  - そうでなければ
    - AをBとCに分割（BもCも空集合ではない）
    - 〈答〉 ＝ Bの要素数 ＋ Cの要素数

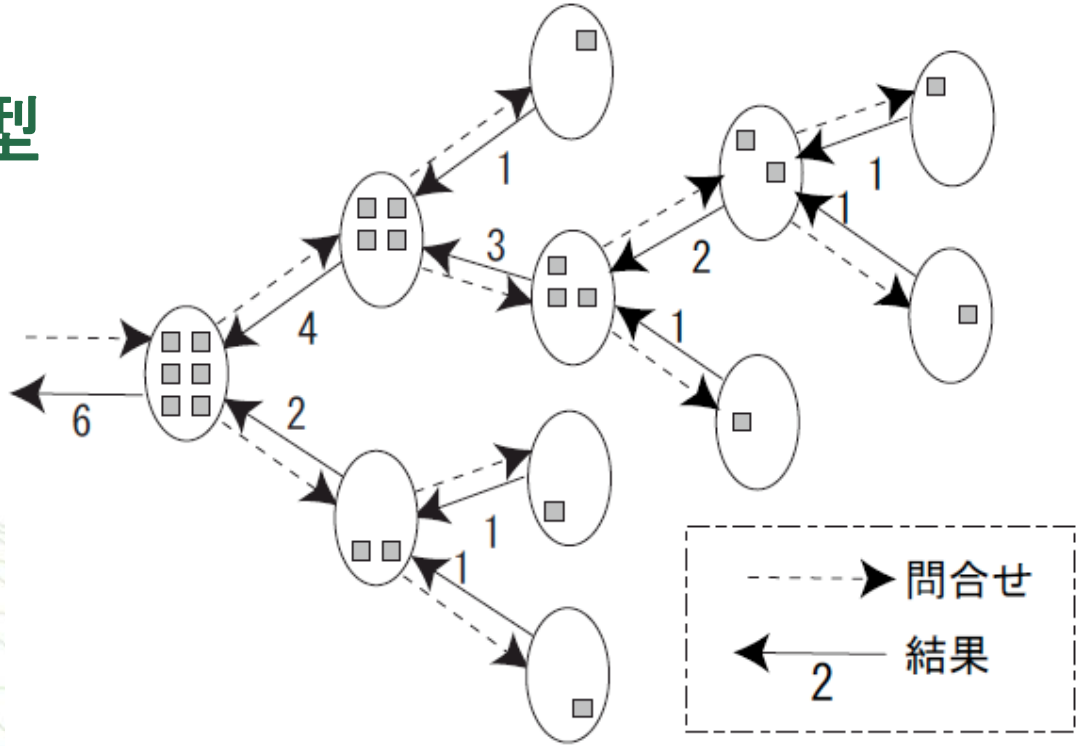
計算の方法の図

↘ 取り出し型



答　0 → 1 → 2 → 3 → 4 → 5 → 6

↘ 分割型



計算の記述

問題

- 今年の八十八夜(立春から数えて88日目)は何月何日か. ただし今年の立春は2月4日であり, 今年は平年である.

考え方

- 2月4日の88日後は"2月をはみ出す" → 2月の残り日数(28 − 4 = 24日)を引く(88 − 24 = 64日)
- 3月64日は3月を越す → 31日を引く(64 − 31 = 33日)
- 4月33日は4月を越す → 30日を引く(33 − 30 = 3日)
- 5月3日は5月に収まる！→ 最終的な答えは5月3日

計算の記述 – より明確に

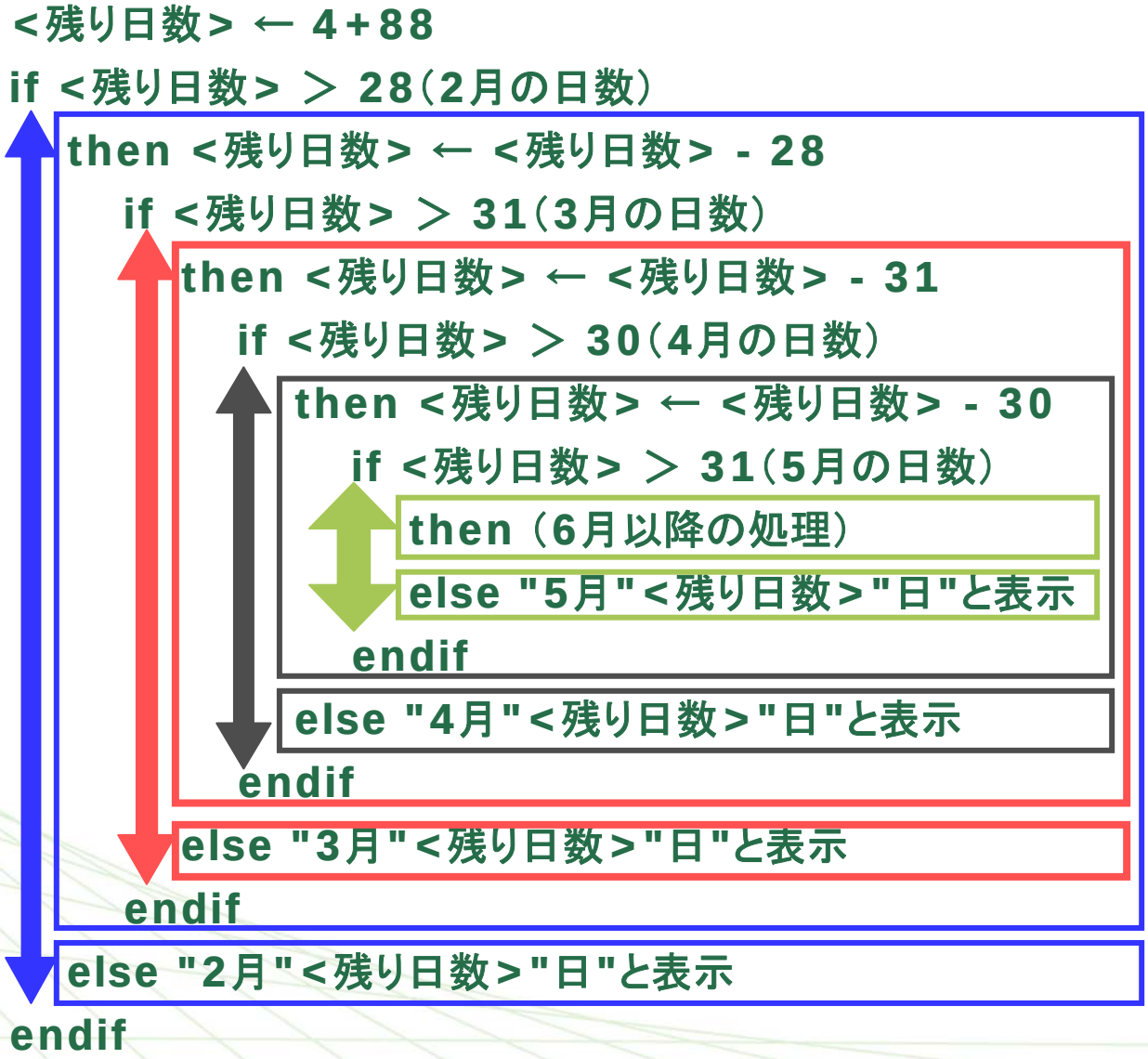
- 2月4日の88日後
  - 〈残り日数〉= 4 + 88 → 2月92日
- 92 > 28(2月の日数)
  - 〈残り日数〉= 92 − 2月の日数 → 3月64日
- 64 > 31(3月の日数)
  - 〈残り日数〉= 64 − 3月の日数 → 4月33日
- 33 > 30(4月の日数)
  - 〈残り日数〉= 33 − 4月の日数 → 5月3日
- 3 < 31(5月の日数)なので計算終了

計算を記述するために必要な要素

- **変数(variable)**
  - 値(例：残り日数)を覚えておく”もの”
  - 変数の値は”代入”により変化させることができる
  - 代入(assignment)
    - 変数に値を設定する操作，”変数名” ← ”式” で表す
- **条件付き処理の操作**

```
if 条件
    then ″条件が成立した場合に行なう処理″
    else ″条件が成立しない場合に行なう処理″
endif
```

八十八夜問題の解法手順



解法手順の改良

- **6月以降については"(6月以降の処理)"としか書いていない**
  - 実際に6月以降の処理が必要になったときに正しい答えが求まる保証がない
  - もっとすっきりと，12月まで対処できる方法は？
- **”m(n月の日数)”という表現が何度も現れている**
  - 例：28(2月の日数)
  - もっとすっきりと記述する方法は？

反復処理と配列

反復処理(repetitive processing)

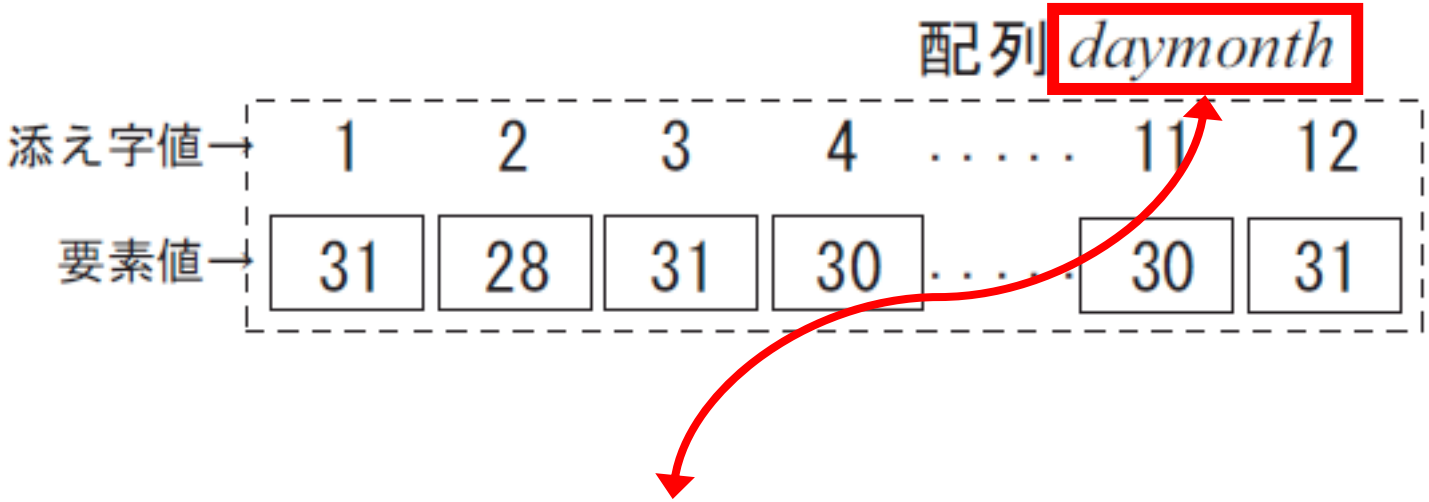
- 条件が成立している限り”処理”を繰り返し実行する

```
while 条件 do
    ”処理”
done
```

配列(array)

- 要素の集合から，”添え字”を使って値を取り出し，変数として扱うことができる
- 要素全体をまとめて扱うことができる

配列の例:配列名



例: $daymonth_2 = 28$

配列名 = daymonth

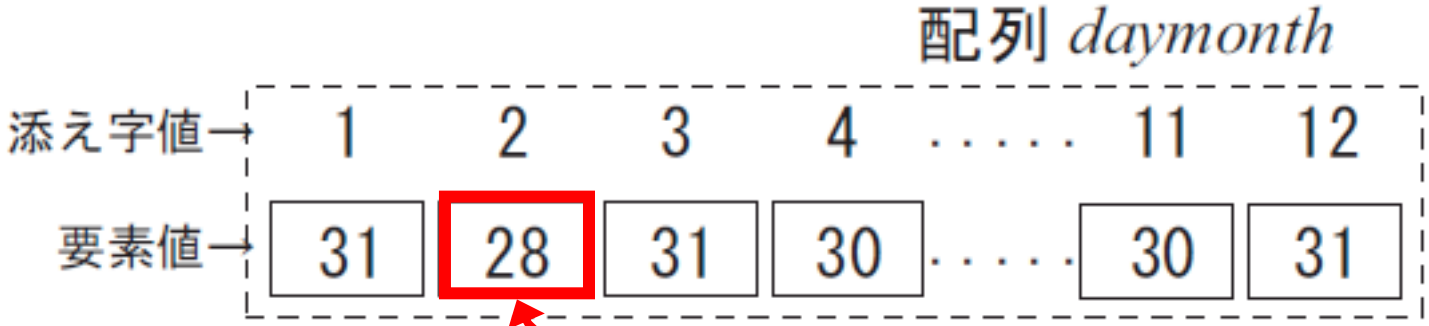
配列の例：添え字

例：daymonth[2] = 28

配列名 = daymonth

添え字 = 2

配列の例



例: $daymonth_2$ = 28

配列名 = daymonth
添え字 = 2
値 = 28

八十八夜問題の解法手順 – 改良版

<残り日数> ← 4+88

m ← 2

while <残り日数> ＞ daymonth$_m$ do

<残り日数> ← <残り日数> - daymonth$_m$

m ← m + 1

done

- ”<残り日数>と月の日数の比較を繰り返し行う”手順を，”反復処理”と”配列”を使うことですっきりと記述することができた

計算モデル

計算モデル

- 計算の記述に従って，実際にどのように計算が進んでいくか，を分類したもの

代表的な計算モデル

- 手続き型(procedural)
  - 計算を前から順番に処理していくことで計算が進行
- 関数型(functional)
  - さまざまな関数を定義し，関数が他の関数を次々に呼び出していくことで計算が進行

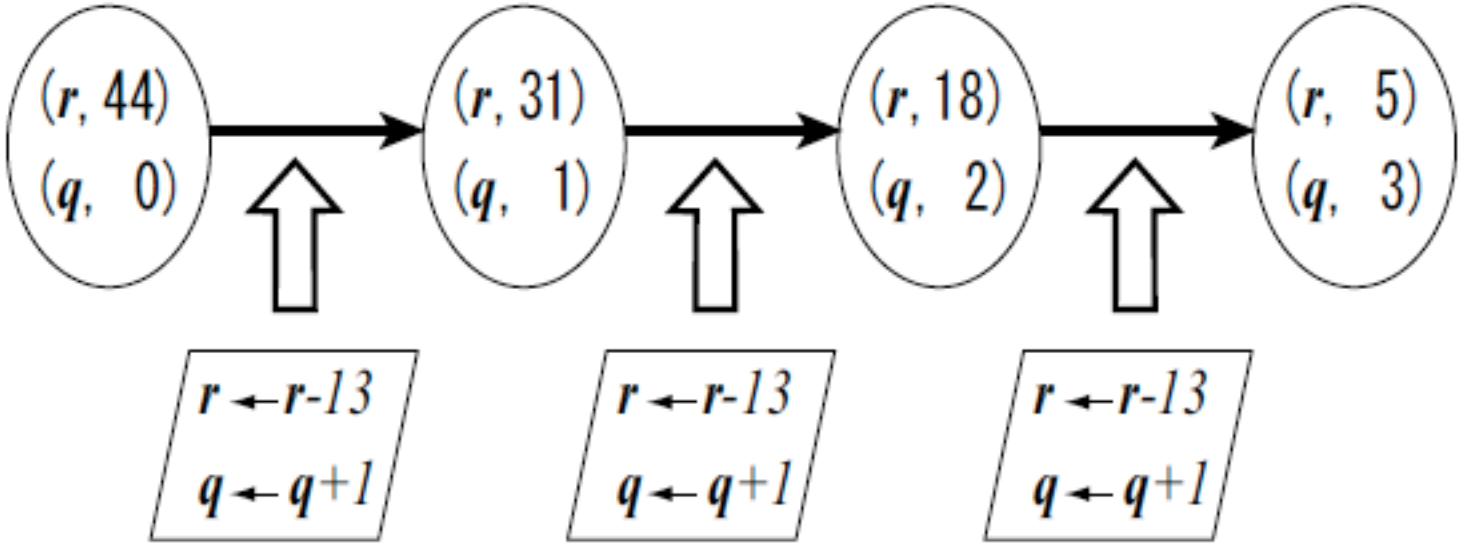
手続き型の例

- 例: 正の整数a, b, a / b の商q, 余りr を求める
  - 計算の手順
    - 被除数(a)から除数(b)を引けるだけ引く
    - 引いた回数を商，残りを余りとする
  - 計算の記述

```
r ← a
q ← 0
while r ≧ b do
    r ← r − b
    q ← q + 1
done
```

手続き型の例

- 手続き型の計算の進行



- 変数値の変化を示すことをトレース(trace)と呼ぶ

関数型の例

- 例: 正の整数a, b, a / b の商q, 余りr を求める
- 計算の記述

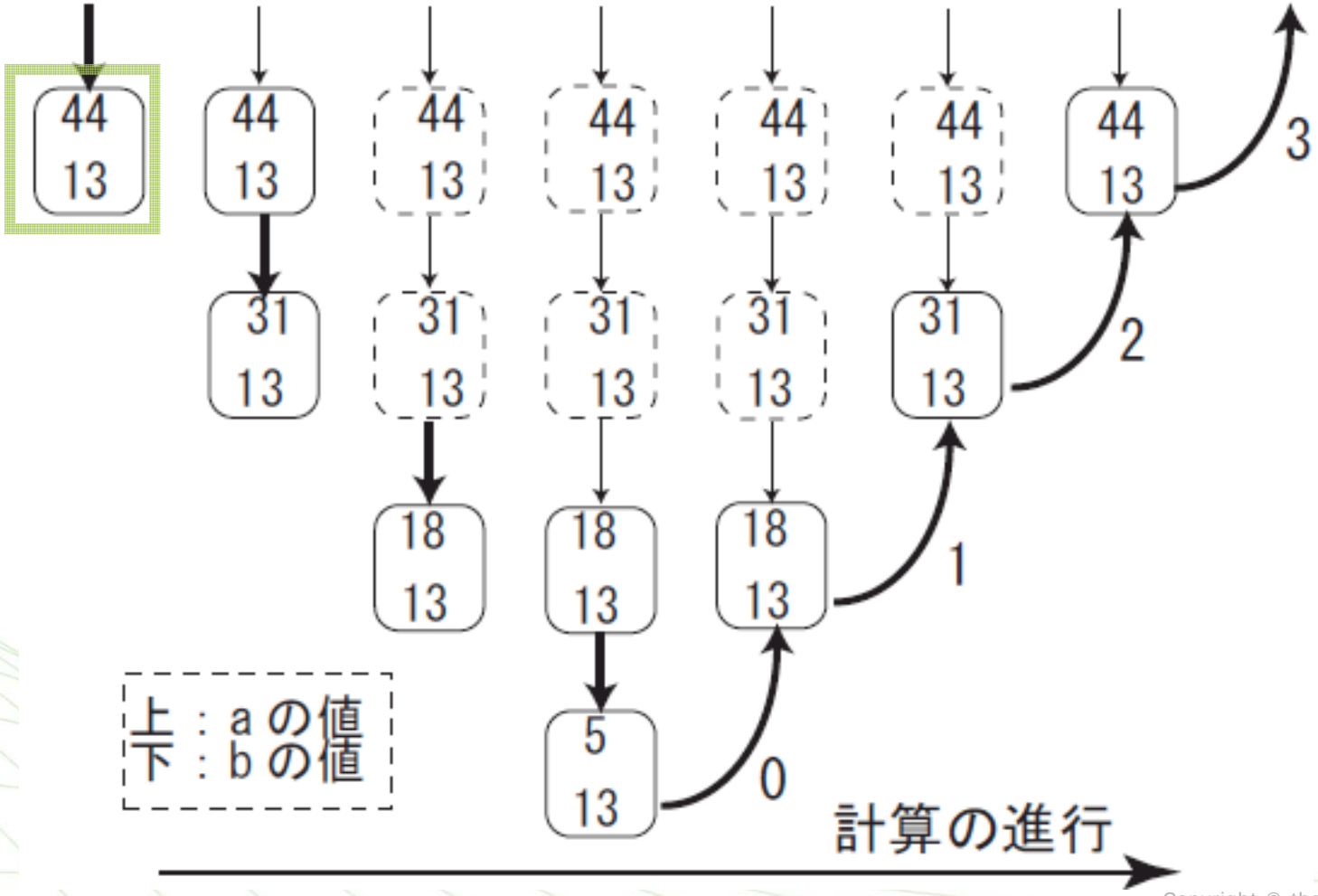
$$q(a, b) = \begin{cases} 0 & a < b \text{ なら} \\ q(a - b, b) + 1 & a \geqq b \text{ なら} \end{cases}$$

$$r(a, b) = \begin{cases} a & a < b \text{ なら} \\ r(a - b, b) & a \geqq b \text{ なら} \end{cases}$$

  - 関数として”自分自身を呼び出す”ことも許される
    - これを”再帰”と呼ぶ

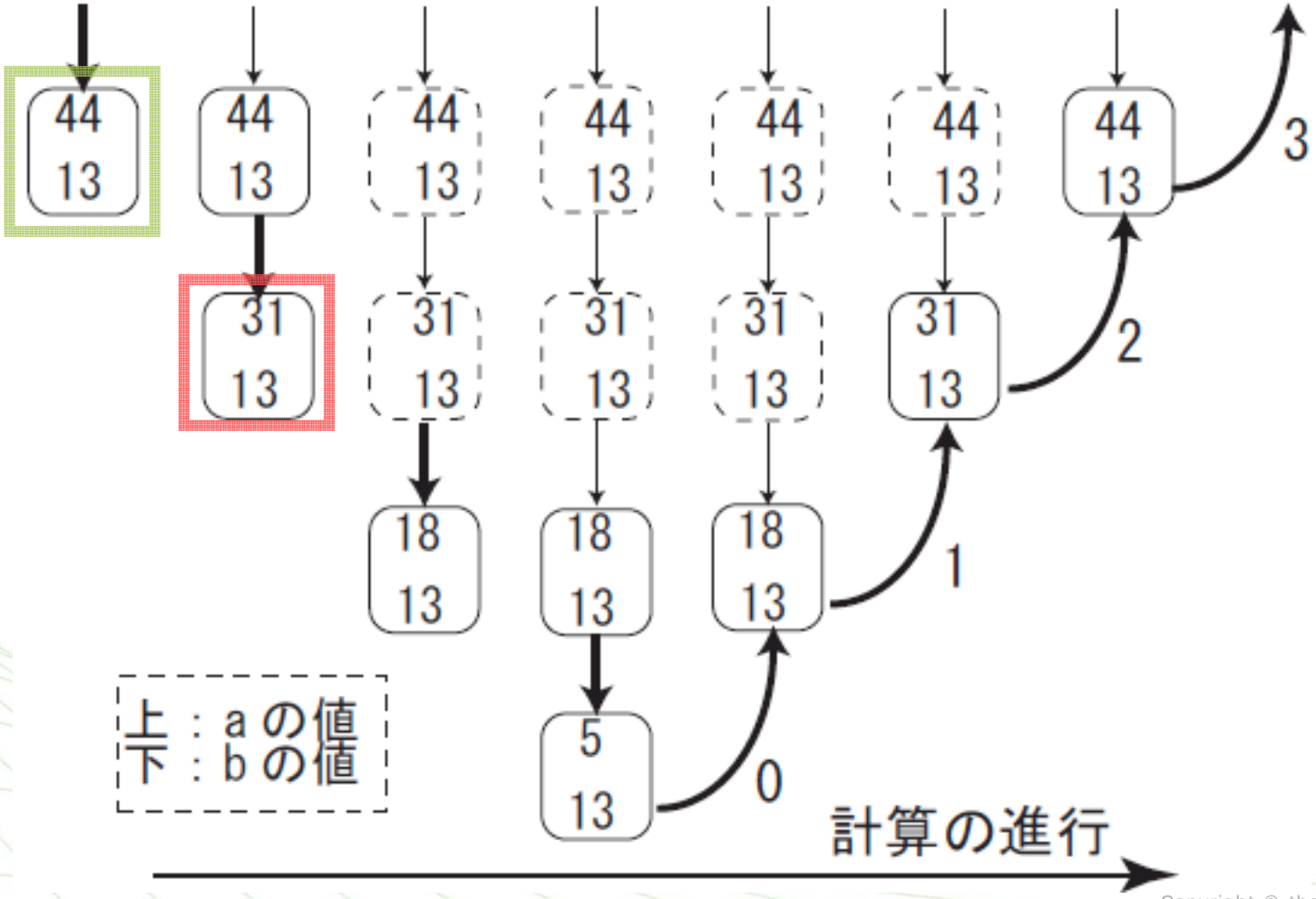
関数型の例：44割る13の商を求める

↘ 商を求める関数型の計算の進行



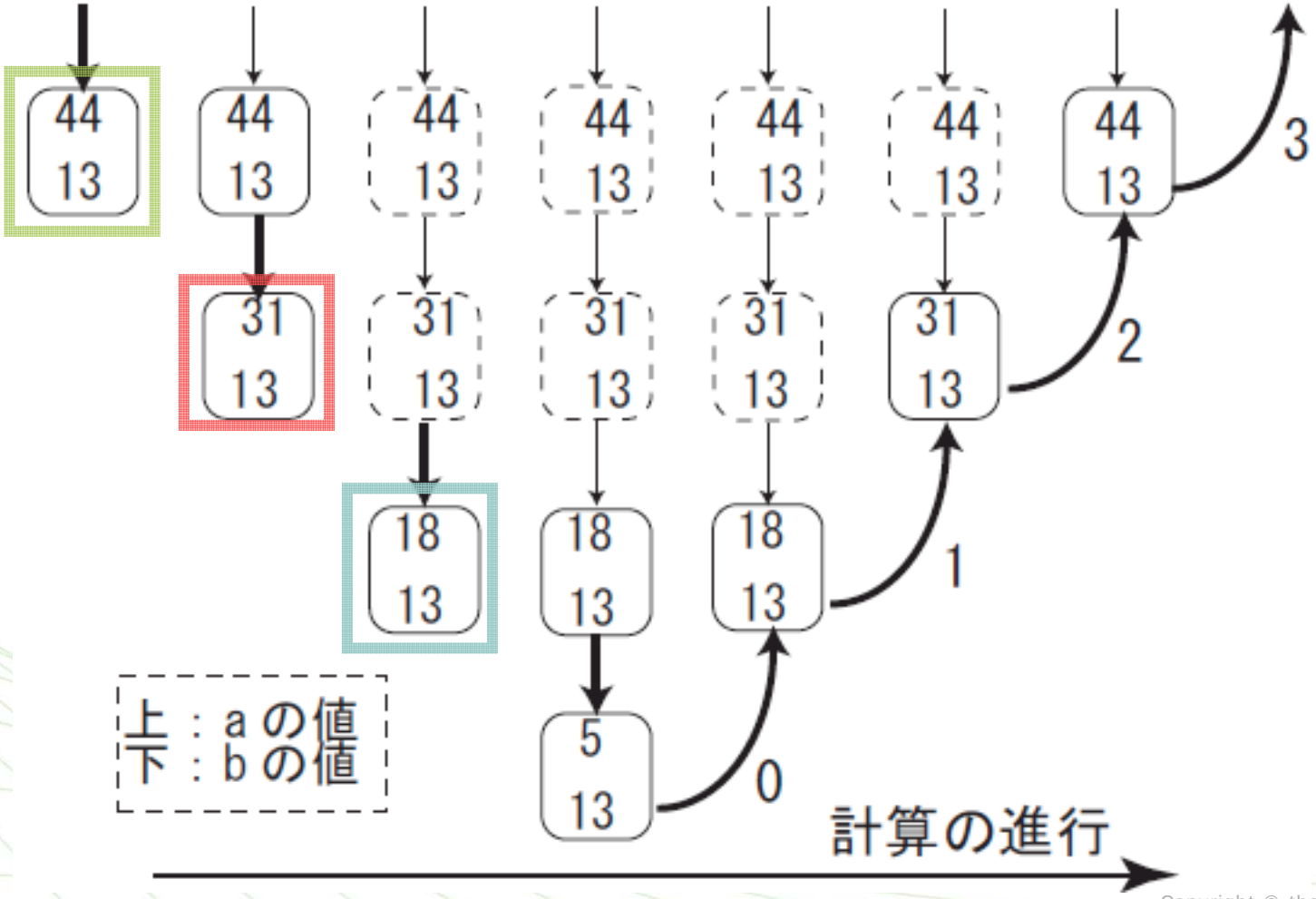
関数型の例:そのために，31割る13の商を. .

- 商を求める関数型の計算の進行



関数型の例:そのために，18割る13の商を…

- 商を求める関数型の計算の進行



プログラムとプログラム言語

- **プログラム(program)**
  - 計算を記述したもの
  - コンピュータを使った問題解決における活動単位
  - 一般に，人間には読み書き不可能
- **プログラム言語(programming language)**
  - プログラムを記述するための約束事をまとめたもの
  - 人工的に作られた言語(人工言語)
  - 人間にも読み書き可能

プログラム言語処理系と機械語

- **言語処理系**
  - ”プログラム言語で書かれたプログラム”を，”機械語で書かれたプログラム”に変換(翻訳)するシステム
- **機械語(machine language)**
  - コンピュータ(ハードウェア)が理解できる言語
- **機械語で書かれたプログラム**
  - すべて2進数か，決まった長さのビットパターン
  - バイナリ(binary) プログラムと呼ぶ
    - ”0と1の集まり”という意味
  - 各種のプログラムをソフトウェアと呼ぶ

アセンブリ言語とアセンブラ

- **アセンブリ言語**
  - 機械語の機能部分や，データの場所に人間が読めるような名前(文字列)をつける言語
  - ”データとしての”プログラムを作る
- **アセンブラ(assembler)**
  - アセンブリ言語でつけた名前を数値や”0と1の集まり”に変換するもの
  - ”データとしての”バイナリプログラムを作る

高水準言語

高水準言語

- 人間が理解しやすい形でプログラムを読み書きできる言語
- ”コンパイラ”を使うことで機械後に変換する
- 高水準言語の例：C言語，C++言語，Java，Fortran...

コンパイラ(compiler)

- プログラム言語で書かれたプログラム(ソース)を機械語(オブジェクト)に変換（翻訳）するソフトウェア

次週

- 次週は大演習室
- C言語のプログラムを入力して，コンパイルを実際にやってみる
  - アセンブラ言語とは，どんなもの？
  - コンピュータに計算をさせるとは，どういうこと？
- 日本語プログラミング言語Dolittleを使った演習
  - 日本語でプログラムを作って，プログラミングによってコンピュータへの指令体験をする.