

# ソフトウェア概論 A/B

-- 再帰 again --

数学科 栗野 俊一 / 渡辺 俊一

2017/06/30 ソフトウェア概

# 伝言

---

## 私語は慎むように !!

- 出席パスワード : 20170630
- 色々なお知らせについて
  - 栗野の Web Page に注意する事  
<http://edu-gw2.math.cst.nihon-u.ac.jp/~kurino>
- 廊下側の一行は遅刻者専用です(早く来た人は座らない)
- 講義開始前に済ませておく事
  - PC の電源を入れておく
  - ネットワークに接続しておく
  - 今日の資料に目を通しておく
- 講義前の注意
  - 講義前は、栗野は準備で忙しいので TA を捕まえてください
- やる気のある方へ
  - 今日の資料は、すでに上っています
    - ▶ どんどん、先に進んでかまいません

# お知らせ

---

## □ 本日(2017/06/30)の予定

### ○ 表現

- ▶ 再帰構造(再)
- ▶ インพุットループ
- ▶ 乱数

## □ 本日(2017/06/30)の目標

### ○ 講義

- ▶ 入力(出力)という「副作用」
- ▶ 再帰構造の更なる理解

### ○ 演習

- ▶ 課題の提出

# 前回(2017/06/23)の復習

---

## □ 前回(2017/06/23)の内容

### ○ 関数と制御構造

- ▶ 順接 : 命令を並べると、並べた順に命令を実行を行う(命令列を作る)
- ▶ 関数 : 命令列に名前を付ける事ができる
- ▶ 関数の引数 : 関数の命令内に「変数」を含める事が可能になる(値を後から決められる)
- ▶ 条件分岐 : if 文により、命令を選択できる
- ▶ 再帰呼出し : 関数定義で自分自身を呼び出す事により、命令を繰り返す事ができる

### ○ 値と型

- ▶ 値 : 実行時に決る (後から変えられる:柔軟/便利)
- ▶ 型 : コンパイル時に決る (適切な操作が選択できる:高速/安全)

### ○ s\_print.h/s\_input.h

- ▶ 基本的なデータ(char,int,文字列)の I/O
- ▶ 使い方 : エディタで include して、コンパイルで -I ~/include を付ける

# 先週 (2017/06/23) の課題

---

## □ 先週 (2017/06/23) の課題

### ○ 課題 20170623-01:

- ▶ ファイル名 : 20170623-01-QQQQ.c (QQQQ は学生番号)
- ▶ 内容 : 二つの整数の積を返す関数
- ▶ ファイル形式 : テキストファイル(C 言語プログラムファイル)

### ○ 課題 20170623-02:

- ▶ ファイル名 : 20170623-02-QQQQ.c (QQQQ は学生番号)
- ▶ 内容 : 自然数の階乗を返す関数
- ▶ ファイル形式 : テキストファイル(C 言語プログラムファイル)

### ○ 課題 20170623-03: (これは、今週[2017/06/30]に回す)

- ▶ ファイル名 : 20170623-03-QQQQ.c (QQQQ は学生番号)
- ▶ 内容 : 二つの非負の整数の最大公約数を返す(ユークリッドの互除法)
- ▶ ファイル形式 : テキストファイル(C 言語プログラムファイル)

# 今週 (2017/06/30) の課題

---

## □ 今週 (2017/06/30) の課題

### ○ 課題 20170630-01:

- ▶ ファイル名 : 20170630-01-QQQQ.c (QQQQ は学生番号)
- ▶ 内容 : 数当てをするプログラム
- ▶ ファイル形式 : テキストファイル(C 言語プログラムファイル)

### ○ 課題 20170630-02:

- ▶ ファイル名 : 20170630-02-QQQQ.c (QQQQ は学生番号)
- ▶ 内容 : 与えられた整数の素因数を表示するプログラム
- ▶ ファイル形式 : テキストファイル(C 言語プログラムファイル)

# 条件分岐(再)

---

## □ 条件分岐

- 条件によって、「異なる二つの振舞の一方を実行(分岐)」する事

## □ if 構文

- 条件分岐を記述する「構文(記述形式)」の一つ

- ▶ 構文 : 「if (条件) { 命令1 } else { 命令2 }」

- ▶ 意味 : 「条件」が成立すれば、「命令1」が、そうでない時には、「命令2」が実行される

- 「単独」の if 構文は、「二択」だが、「if 構文」の組み合わせで「多択」が可能

## □ if 構文の亜種

- else 節の省略

- ▶ 「if(条件) { 命令 } else {}」の「else{}」は省略可能

- else if 句 : 次の二つは同じ意味

- ▶ 「if (条件1) { 命令1 } else {if (条件2) { 命令2 } else {命令3}}」

- ▶ 「if (条件1) { 命令1 } else if (条件2) { 命令2 } else {命令3}」

# 再帰呼出し(再)

---

## □ 再帰呼出し

○ ある関数の本体に、その関数の呼出しが含まれている事

▶ 「その関数の一度の呼出し」で「その関数の呼出しが」が何度も呼び出される可能性が生じる

○ 注意：安易に「再帰呼出し」を記述すると、「無限ループ」になる

▶ 「再帰呼出しを終了させる」仕組みを「意図的に導入」する必要がある

▶ 「条件分岐」が必須

○ 再帰句：P は、停止条件、A は繰り返す命令、X' は X から計算され、いつか P(X) が成立する

▶ 記述形式：`f(X) { if ( P(X) ) {} else { A(X); f(X'); }`

▶ 意味：f(X) を呼ぶと、P(X'..') が成立するまで A(X) を繰り返す

▶ ポイント：「A(X) を繰り返す」ために「f() を再帰で定義」する

## □ 再帰関数

○ 再帰呼出しを行う形で定義された関数

## □ 繰り返し：同じ命令(記述)を何度も呼び出す仕組み

○ 再帰呼出しを利用する事により、「繰り返し」が実現できる

○ 再帰関数は、「何らかの繰り返し」を実現している



# 入力(出力)という「副作用」

---

## □ 数学の関数

### ○ 値を与えると値を返す

▶ 何回利用しても同じ結果にしかない

▶ 例 :  $\sin(x) + \sin(x) == 2 * \sin(x)$

### ○ 「純粹関数」

▶ 「副作用」を持たない関数 ( cf. 「数学」の「関数」は「純粹」)

▶ 数学的な操作 (置き換えや、交換等)が可能 : 「(『論理的』に..)扱い易い」

▶ 「実行順序」に意味がない(素直)

## □ C 言語の関数

### ○ 「副作用」を持つ場合がある

▶ 「副作用」の例 : 入出力 (他にも色々ある)

### ○ 「実行順序」に意味がある (だから、「制御構造」に『意味』がある)

▶ 「扱いが不便」だが「やり甲斐」がある

# インプット・ループ：入力による繰り返し

---

## □ 一般的な再帰関数

- 引数の値によって、挙動(繰り返し回数)が変化する

- ▶ 引数が確定すれば、繰り返し回数も確定する

## □ インプット・ループ

- 入力(インプット)の値によって繰り返し回数を制御したい

- ▶ 典型的な応用: 「終わり」まで「繰り返す」

## □ インプット・ループを作るパターン

- 入力関数を引数とする再帰関数を作ればよい

- インプット・ループ句

- ▶ 記述形式: `f(X) { if (P(X)) {} else { A(X); f(input()); } }`

- ▶ 意味: 入力 X が P を満たすまで A(X) を繰り返す

- ▶ ポイント: f を最初に呼び出す場合も f(input()) の形にする

# 乱数

---

## □「乱数列」とは

○定義： $\{r_n\} (n = 1, 2, \dots)$  が「乱数列」であるとは

▷  $r_{i+1}$  が、それ迄に与えられた  $r_1, r_2, \dots, r_i$  から予想できない数列になっている場合

▷ cf. (不正のない)サイコロを(無作意に)投げて、出た目を並べた数列は「乱数列」になる

▷ cf.  $n$  番目の数  $r_n$  を  $n$  から求める「規則(例えば  $r_n = n^2$ )」がある場合は乱数でない

○[注意]「予想できない」を「数学的に表現する」のは難しい(西川先生を困らせてみよう)

▷ 現実的には「予想をしない(数学の立場)」か「乱数のようにみえる数列(疑似乱数: 計算機の立場)」を取る

## □「乱数」とは

○「乱数列  $\{r_n\}$ 」の  $i$  番目の要素  $r_i$  を取った物の事(なので、その値を事前に予想できない)

# 「乱数の分布」と「疑似乱数」

---

## □「乱数」の分布

- 「自然現象に現れる乱数: $\{r_n\}$ 」は、「個々の要素( $r_i$ )の振舞」は解らない
  - ▶「集合としての振舞(の一部)」である「分布」は解っている(仮定してもよい..)事が多い
- 一様分布： $r_i$ はある範囲の数で、どれかが現れる確率がどれも同じ
  - ▶cf. さいころの出目は、 $\{1 \sim 6\}$ で、どれも等確率(1/6)
- 正規分布： $\{r_i\}$ の分布は、正規分布  $N(\mu, \sigma^2)$  に従う
  - ▶cf. 自然現象における雑音等

## □乱数の応用

- 個々の現象が「起きる規則が知られていない/予想できない」場合に「乱数」として扱う
  - ▶ただし、予め「対象となる現象」の分布(統計を用いる)を調べる
  - ▶その乱数が、その分布に従うと「仮定」する

## □「疑似乱数列」とは

- 「乱数列」のように\*見える\*ように「規則的に作成された」数列