[B2] 分割コンパイル

大崎 博之

関西学院大学 理工学部 情報科学科

ohsaki@kwansei.ac.jp

[B2] 分割コンパイル

1. **関数**
2. **分割コンパイル**
3. **構造体**
4. **乱数**

これらを学ぶことに何の意味があるのか?

1. 関数

- C 言語は手続き型言語 (procedual programming language) である
  - 手続き型言語 = 手続き (procedure) の呼び出しによって処理が実行される言語
- C 言語における手続き = **関数 (function)**
  - 手続き (procedure) のうち、値を返すものは関数 (function) と呼ばれる

→ つまり、C 言語では **関数を次々と呼び出す** ことがプログラミングの本質である

2. 分割コンパイル

- C 言語のプログラム = (通常数多くの) 関数の集まり
- 関数を複数のファイルに分けて記述するメリットは?
  1. **管理が容易** になる (目的ごとに別のファイルに記述できる)
  2. **コンパイルが高速** になる (変更のあったファイルだけ再コンパイルすればいい)
  3. よく使うものを **ライブラリ化** できる (実は、printf、scanf なども別コンパイルされた関数)

→ 数百行程度のプログラムならわざわざ分割する必要はない

3. 構造体

- プログラミングの本質は アルゴリズム と データ構造
  - プログラミングの成功/失敗は……
    - 良い アルゴリズム を考えられるか?
    - 良い データ構造 を考えられるか?
  - プログラミングテクニックは実は瑣末 (さまつ) な問題
  - だから大学の講義はとても重要!

→ 設計した データ構造 を実現するために 構造体 が不可欠

4. 乱数

- **乱数** は重要な概念だが、上の 3 つほどではない
- 乱数について知っておくべきこと
  - コンピュータでは数学的に本当にランダムな乱数は **生成できない**
  - 標準ライブラリの rand() 関数はかなりいいかげんな (= 規則性のある) 擬似乱数

POSIX.1-2001 で示されている rand() 関数の実装例

```
static unsigned long next = 1;

/* RAND_MAX assumed to be 32767 */
int myrand(void) {
    next = next * 1103515245 + 12345;
    return((unsigned)(next/65536) % 32768);
}

void mysrand(unsigned seed) {
    next = seed;
}
```