ソフトウェア概論 B

数学科 吉開範章, 渡辺俊一 (栗野 俊一)

おしらせ

□本日は先週に引続き栗野が担当します。

□資料

http://edu-gw2.math.cst.nihon-u.ac.jp/~kurino/2010/soft/20100702/20100702.html

本日の概要：関数

- 関数 (6 章 : p.114　p.145) の復習と落ち穂拾い
- 課題
  - 前回未提出の人は前回分も
    - 6-7 (p.133) : int 型の配列 vc の要素の最小値を求める
    - 6-9 (p.133) : int 型の配列 vc2 の要素の逆順を vc1 に入れる
    - 6-10 (p.139) : 行列の積を計算する
  - 6-11 (p.139) : 5 人の学生の 3 教科の点数を処理するプログラム
    - 幾つかバリエーションがあるので、それは学生番号に従って指示する

前回までの復習(1) : 関数とは

□ 関数 (6 章 : p.114　p.145)

○ 定義:「幾つかの命令文の集まり」に名前(関数名)を付けた物

▹ 関数定義 : 「幾つかの命令文の集まり」に関数名を付けて関数が作成できる

▹ 関数呼び出し : 関数名を使って「幾つかの命令文の集まり」が利用できる

○ 関数を作るには ..

▹ 関数にする命令の集まりを切り出して、ファイル一番最後に移動する

▹ それ全体を { } で囲み、関数名を付ける

▹ 元の位置に関数呼び出しを置く

▹ 元のプログラムと関数の情報のやり取りの為に引数や関数値を調整する

○ 情報のやり取り

▹ 引数:関数呼び出しの実引数の「値」が関数定義の仮引数(の変数)に代入される

値渡し:プログラムと関数の間は「値」しかやり取りできない

▹ 関数値: 関数定義の本体で return 文を使って、「関数の値」が指定できる

□ 関数利用の利点

○ 問題が分割される(分割統治)

▹ スコープ: 名前の有効範囲 / 自動変数の有効範囲は、ブロック内(複合文内)のみ

▹ スコープがあると、狭い範囲で物を考える(他の部分を忘れる)事ができる

○ 再利用ができる (cf. ライブラリ関数)

▹ 実引数を変えるだけで、異なる「命令の集まり」として利用できる

▹ 「機能」だけでなく「正しさ」も利用できる (cf. 単なるコピー との違い)

前回までの復習(2)：関数に拘る概念

- □ 関数原型宣言 (プロトタイプ宣言：p.127)
  - ○ 関数定義の「関数頭部」を取り出した物
    - ▹ 関数原型宣言の場合は仮引数の変数名が省略可能
  - ○ 関数の前方参照のために必要
- □ インクルード (#include)
  - ○ 一つのファイルに他のファイルの内容を埋め込む仕組
    - ▹ ヘッダーファイル：定数定義や関数原型宣言をまとめたファイル (stdio.h)
- □ 配列の共有
  - ○ 関数の呼び出し元と関数で、「配列の要素」を「共有」できる
    - ▹ 共有: 配列の要素の値を取り出したり、配列の要素を変更できる
    - ▹ cf. 「変数」は「共有」できない
  - ○ 「共有」の仕方 (p.130：取り敢えずの理解)
    - ▹ 実引数に「配列名」を記述する
    - ▹ 仮引数は「サイズの無指定の配列」として宣言する
    - ▹ 関数内では、仮引数が呼び出し元の配列の「別名」になる
  - ○ 関数では「値」しか渡せないのは「原則」
    - ▹ 「配列名」で「特殊な値」が表現されている (cf. 10 章 (p.202) のポインター)

const 型修飾子

- □ 関数の引数
  - ○ 値渡し
    - ▷ 呼び出し元と関数では、情報だけが共有される
    - ▷ 呼び出し元の「変数の値」が「関数呼び出し」で変らない
    - ▷ 関数を安心して利用できる (cf. ライブラリ関数)
  - ○ 配列名の指定
    - ▷ 呼び出し元の「配列の要素」の値が変えられる
    - ▷ 善し(変えられるので、実現できる機能がある)悪し(値が変ってしまう不安)
    - ▷ 「配列の要素の値」は渡したいが「変えられたくない」: const 宣言する
- □ const 型修飾子 (p.133)
  - ○ 仮引数宣言で、配列を宣言する時に const を付ける (List 6-13, p.133)
    - ▷ 関数内で、その配列の要素の内容を変更する文があるとエラーになる

逐次探査と番兵法

- 逐次探査 (p.134)
  - 配列の要素を順に調べる配列の基本アルゴリズム
    - 必要ならば、配列の要素全てを調べる必要がある
    - 配列の添字が、配列サイズを越えないようにする必要がある
    - cf. List 6-14 (p.134)
- 番兵法 (p.135)
  - 逐次探査は、終了条件が二つある
    - 配列の全てを調べた：探査は失敗
    - 途中で、目的のものを見付けた：探索は成功した
  - 繰り返しの終了条件が複数あるのは、わかりにくい
    - 探査が必ず終了するようにすれば、失敗の場合を考える必要がなくなる
    - 配列の最後に探査が成功する値(番兵)をおけば、この条件が成立する
    - cf. List 6-15 (p.136)

変数のスコープと記憶期間

- スコープと記憶期間
  - 変数:名前と領域(値を保持する)からなる
    - スコープ(空間/静的/コンパイル時): 名前が有効(名前と領域の対応が一致している)な範囲
    - 記憶期間(時間/動的/実行時): 領域がその値を保持している期間
  - 名前を持たない領域がある事に注意 (cf. 10 章)
- スコープの種類
  - 全域(グローバル) : どからでも
  - ファイル内 : ファイル内
  - ブロック内 : ブロックの内 (関数ブロックを含む)
- 記憶期間
  - 意味は領域の有効な時期だが、プログラマからは値の保存期間となる
  - 静的 (変数)
    - 何時でも : プログラムの実行開始から、終了まで
    - 関数内 : 関数が開始されてから、終了するまで
  - 動的
    - 動的 : 作られてから、破棄されるまで (cf. 10 章)

変数宣言

- 変数の宣言とそのスコープ記憶期間
  - static でない関数ブロックの外
    - スコープ:全域 / 記憶期間:何時でも
    - 他のファイルからは extern 宣言が必要
  - static 変数
    - スコープ:その位置で決定 / 記憶期間:何時でも
  - auto 変数
    - スコープ:その位置で決定 / 記憶期間:関数内
- 変数によるスコープの遮蔽
  - 内側のスコープの同名の宣言があると外側のスコープ無効になる
    - 内側のスコープが終了すると外のスコープが復活
    - 変数の初期化式のスコープは外の変数のスコープ
    - cf. 「int x = x;」 はちゃんと意味がある

課題

- 課題は、次の Web Page の内容を参照してください。

  http://edu-gw2.math.cst.nihon-u.ac.jp/~kurino/2010/soft/20100702/20100702.html
- 課題
  - 前回未提出の人は前回分も
    - 6-7 (p.133) : int 型の配列 vc の要素の最小値を求める
    - 6-9 (p.133) : int 型の配列 vc2 の要素の逆順を vc1 に入れる
    - 6-10 (p.139) : 行列の積を計算する
  - 6-11 (p.139) : 5 人の学生の 3 教科の点数を処理するプログラム
    - 幾つかバリエーションがあるので、それは学生番号に従って指示する

おわり

終了