ソフトウェア概論 A/B

-- データ構造 (配列) --

数学科 栗野 俊一

伝言

私語は慎むように !!

- 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
- 廊下側の一列は遅刻者専用です(早く来た人は座らない)
- 講義開始前にすませておくこと
  - PC の電源を入れる
  - ネットワークに接続しておくこと
  - 今日の資料に目を通しておくこと
- 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

前回の復習：コーディング

- 前回の復習 (コーディング)
  - 情報を巡る、三つの「形態」: 相互に「同型変換」が行われる
    - 情報 :「現実」世界での「何か」
    - データ :「コンピュータ」内での「情報」
    - 数値 :「コンピュータ」が直接扱える「数値」
    - [注意]「情報/データ」は、今回限りの用語(一般的には違う意味で使う)
  - 同型変換のための仕組み
    - 情報/データ間 : ハードウェア (Input/Output)
    - データ/数値間 : プログラム (Encode/Decode)
  - コーディングの仕組み
    - 「直積」(既存の情報の組み合わせ) : 空間を拡大する
    - 「制約」(例外や同値の導入) : 空間を狭くする
  - 同型構造：情報, データ, 数値の三つは同型に *して* ある
    - 「可換」:「数値の計算」により、「情報の処理」ができる

前回の復習：構造体

- 前回の復習 (構造体)
  - C 言語でのコーディング
    - 直積 : 構造体
    - 制約 : 正規化のプログラムを作成する
  - 構造体
    - 既存の型の直積となる新しい型を作る仕組み
    - struct { 型1 : tag 名1; .. 型n : tag 名n } : 型1 .. 型n の直積
  - typedef
    - 新しい型に名前を付ける仕組み
    - 構文 : typedef 型 新しい型名;
  - 新しい型ですぐに実現される機能
    - 変数の宣言/代入(関数への引数/関数の値)はできる
    - 他の機能(を実現するための計算をする)は関数で実現する

お知らせ

- 本日の予定
  - データ構造と型定義 (配列)
- 本日の目標
  - 演習
    - 課題の提出

前回 (2011/11/11) の課題

- 前回 (2011/11/11) の課題
  - 課題 1:
    - ファイル名 : 20111111-1-QQQQ.c (QQQQ は学生番号)
    - 内容 :
      - ニ次元のベクトル型(Vector2D) を定義し、以下の機能を実現する
      - 座標は整数でよい
    - 機能 :
      - 入力,出力,和,定数倍
    - 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 (2011/11/18) の課題

- 今週 (2011/11/18) の課題
  - 課題 1: [参考: sample-006.c]
    - ファイル名 : 20111118-1-QQQQ.c (QQQQ は学生番号)
    - 内容 : 5 つの整数を読み込んで、その逆順に出力するプログラム
    - 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 2: [参考: sample-011.c]
    - ファイル名 : 20111118-2-QQQQ.c (QQQQ は学生番号)
    - 内容 : 5 つの整数を読み込んで、その全部と真ん中の三つの総和を計算するプログラム
    - 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 3:
    - ファイル名 : 20111118-3-QQQQ.c (QQQQ は学生番号)
    - 内容 : 3 次元行列の積を計算するプログラム
    - 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)

メモリモデルとポインター値 (復習)

- メモリモデル
  - 計算機の情報はメモリに記録される
    - メモリ: 1 byte の Cell の集まりで、個々の Cell にアドレスが付く
  - 変数
    - メモリの 1 個以上の Cell の集まりで、演算情報をもつ
    - cf. 型 = サイズ(何 byte か ?) + 演算情報
- ポインター値：アドレス値 + 型情報
  - 変数名に「&」を付けると「ポインター値」が得られる
  - ポインター値に「*」を付けると「変数名」と同じ振舞をする
    - int v; の時、「*(&v) == v」と考えてよい
    - 「変数」は関数に渡せないが、「ポインター値」は関数に渡せる
  - 添字
    - pa[i] == *(pa + i)

配列

- 配列
  - 同じ物の直積(並び)を作る
    - メモリモデルの抽象化
  - 配列の宣言 [001-003]
    - 型 配列名[サイズ]
- 配列宣言のポイント
  - 同じ型の変数を n 個同時に作る事ができる [004,005]
  - 添字に式がかける (これが「配列」の骨頂) [006]
  - 「配列名」は「ポインター定数」となる
    - 関数の引数として「配列名」を指定すると、ポインター値が渡る
  - 配列のサイズは、配列を宣言した所でしかわからない [007-010]
    - 関数の引数として配列を利用する場合はサイズをどう渡すか考える必要がある
    - 固定にする (定数[推奨しない]か型宣言をする) / 引数に渡す[強く推奨]
  - 配列を処理する関数はサイズが可変にできるので便利 [011]
- 多次元配列
  - 配列の配列も作れる[012]
    - 関数の引数に渡す場合は、最初の要素だけが省略可能[013]

□ 構造体と配列

- 共に既存の型から新しい型を作る仕組み
  - ▷ 構造体:異なる物の直積を作る
  - ▷ 配列:同じ物の直積(並び)を作る