ソフトウェア概論 A/B

-- 浮動小数点数型, 代入 --

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

今後の予定

- 終わりからの予定
  - 2012/07/13 (講議最終日)
    - 試験を行う
  - 2011/07/06
    - 1 限は、総まとめ / 2 限は模擬試験
  - 2011/06/29 : 本日
    - 本日までの内容が、試験範囲

前回の復習

- 入力
  - プログラムの実行中に、情報をプログラムに与える仕組み
  - s_input_QQQQ : 何かをキーボードから入力する
    - プログラムへの入力
    - キーボードから「文字列」を与えると、「数値」に変換される
- 浮動小数点数型 (double 型)
  - C 言語内での表現 : 小数点付きの数 (cf. 123.456)

お知らせ

- 本日の予定
  - 浮動小数点型
  - 代入
- 本日の目標
  - 演習
    - 課題の提出

前回の課題 (2012/06/22)

- 前回 (2012/06/22) の課題
  - 課題 1:(前々回の課題 2 : ファイル名の日時が先週となっている)
    - ファイル名 : 20120615-2-QQQQ.c (QQQQ は学生番号)
    - 内容 : 余りを計算するプログラム考えよ
    - 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: (課題番号が 2 から始まっている)
    - ファイル名 : 20120622-2-QQQQ.c (QQQQ は学生番号)
    - 内容 : 文字列の長さを計算する my_string_length
    - 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)

本日の課題 (2012/06/29)

- 本日 (2012/06/29) の課題
  - 課題 1:
    - ファイル名 : 20120629-1-QQQQ.c (QQQQ は学生番号)
    - 内容 : 実数値を二つキーボードから読込み、その四則を計算する
    - 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:
    - ファイル名 : 20120629-2-QQQQ.c (QQQQ は学生番号)
    - 内容 : 方程式「cos(x)=x」の答を求めよ
    - 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: (前回の課題なので、日付が 0622 である事に注意)
    - ファイル名 : 20120622-3-QQQQ.c (QQQQ は学生番号)
    - 内容 : cos のグラフ
    - 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 4: (前回の課題なので、日付が 0622 である事に注意)
    - ファイル名 : 20120622-4-QQQQ.c (QQQQ は学生番号)
    - 内容 : 素数の判定プログラム
    - 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)

ファイルの入手とインストール

- ファイルのダウンロード
  - 次の本日 (2012/06/29) のページからファイルをダウンロードする

    http://edu-gw2.math.cst.nihon-u.ac.jp/~kurino/2012/soft/20120629/20120629.html
  - ダウンロードするファイル
    - s_input.h :「c:\usr\c\include」に保存する事
    - s_print.h :「c:\usr\c\include」に保存する事

浮動小数点数型(再)

- □ double 型
  - ○ 小数点付きの数を表現する
    - ▹ C 言語内での表現：小数点付きの数 (cf. 123.456)
    - ▹ 出力：s_print_double (s_print.h 内)
    - ▹ 入力：s_input_double (s_input.h 内)
  - ○ 様々な数学的な関数が利用できる
    - ▹ sin/cos/exp/log/etc.. cf #include <math.h>
  - ○ (当然) 四則の計算ができる：3.0/2.0 → 1.5 (cf. 3/2 → 1)
- □ 型の昇格と型変換
  - ○ 型の昇格
    - ▹ 浮動小数点数型と整数型が混在する計算では、自動的に浮動小数点数になる
  - ○ 型変換(キャスト)
    - ▹ 値の前に「(型名)」とすると、その型の数に変換される
    - ▹ 浮動小数点数を整数型にするには、キャストを利用する

浮動小数点数型の性質

- 倍精度浮動小数点数型(double)のコーディング
  - [注] C 言語には、「float 型(単精度浮動小数点数型)」もあるが、これは後期
  - IEEE754 形式
    - 64bit(8byte) を 符号(1 bit)+指数(11 bit)+仮数(52bit) にわけて表現
  - 表現出来る範囲は(当然)有限
    - b x 10^e の形 : b は 16 桁の有効数値 / e は -308 ～ +308
    - [絶対値の大小] 2.225074 x 10^{-308} ～ 1.797693 x 10^{308}
    - この値は float.h で DBL_MIN および DBL_MAX で与えらえている
  - ポイント
    - 実数型には「誤差」が伴う (「==」は使わない)
- 浮動小数点関数
  - 一般的な関数 sin/cos/exp .. が利用できる
    - ソースファイル :「#include <math.h>」が必要
    - コンパイル :「-lm」オプションの追加が必要
    - cf.「cc -o foobar.exe foobar.c -lm」

代入

- 代入とは？
  - 変数の値を「変更」する事
    - 言い替えると..？「変数は代入が行われると変数の値が変化する」
  - 変数は常に値を持つが..
    - 代入された前と後では「値が変化」する
- 変数に値を代入するには？
  - 代入文 (「変数名 = 式」) を使う (sample-016)
    - 式の値が計算された上で、変数に代入される
    - 「=(等号)」を使っているが、「等しい」という意味ではない
    - C 言語では「等しい」という意味には「==」を使う
  - 変数の値は何度でも参照可能 (sample-017)
    - 計算結果を何度も利用する時には変数に入れておく
  - 代入の右の式の中で自分自身の現在の値が利用できる (sample-018)
    - やっぱり、「等号」じゃない

代入と関数引数

- 関数引数は「代入」か？(sample-019)
  - 代入は、「変更」がおきる
    - 元々変数がもっていた値が「失われて」しまう
  - 関数引数は、変数の値を決めるが、「変更」するわけではない
    - 変数の値が保存されている (sample-020)
    - 同じ「名前」を利用していても「異なる変数 (アドレスが異なる)」

局所変数宣言

- 引数でない変数
  - ブロックの先頭で変数宣言すれば、新しい変数が利用できる
    - メモリの一部がその変数名に割り当てられる
  - 引数との違い
    - 宣言直後は値が定まっていない(何が入っているかは不明)
    - 必ず、値を代入してから利用すること(可能なら初期代入すること)
    - 「初期代入」'=,「変数宣言」+「代入」(実は微妙に違うが..)
- 名前の有効範囲
  - 「変数名」の有効範囲は宣言の後からブロックの終りまで
    - ブロックの外からはその名前が利用できない
    - 同じ名前で宣言しても異なるメモリになる(可能性が高い)
  - 名前とメモリは「独立」な事に注意
    - 名前は利用できなくてもメモリは利用できる(ポインターの利用)

入力・処理・出力

- 変数を利用したプログラムパターン
  - 入力・処理・出力の三つの部分からなる
    - 入力：変数を初期化する
    - 処理：(計算を利用しつつ..) 変数の値を変更する
    - 出力：変数の値を「結果」として出力する
  - 関数定義もこの形
    - 引数の値の設定、関数の本体、値の返却

ステートマシンモデル (状態計算モデル)

- 計算とは変数の値を変更する事である
  - ステート：変数の値の集合が、
  - 計算：ステートを変更する事
  - 計算の終了：ステートが望みの結果になっているという事
    - 求める結果が、どこかの変数の値になっていればよい
    - 画面への出力も「出力画面」という変数への「蓄積結果」と考える