ソフトウェア概論 A/B

-- ガイダンスと復習 --

数学科 栗野 俊一

2011/09/30 ソフトウェア概論

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

前期の成績について

- 前期(ソフトウェア概論 A)の成績について
  - 成績処理が遅れてしまった
    - 再履の人は印刷が間に合わず、成績欄が空白に (ごめんなさい)
    - Web の方を参照してください
  - かなり甘く付けた
    - 後期は、辛くする予定..
    - レポート全提出+試験参加で、単位は保証する
- 評価内容に疑問がある場合
  - 講議終了後、栗野に申し出てください
    - ミスをしている可能性あり
  - メールでも構いません
    - kurino@math.cst.nihon-u.ac.jp

後期の方針

- 後期 (ソフトウェア概論 B) の方針
  - 基本は前期 (ソフトウェア概論 A) と同じ
  - ただし...
    - 前期の知識を仮定する : 身についていない所は復習する
    - 後期は前期を踏まえ、更に高度な内容になる予定
- 方針(ポイント)の復習
  - 私語厳禁：他人に迷惑をかけるな !!
    - 自分がやらないのは自分の問題(好きにすれば..)
    - 他人への迷惑は断固とした態度を取る
  - 実習重視：毎回 Note-PC /LAN を利用する
    - 習うより慣れろ / 普段から利用する
  - 評価：課題+試験(講議時間中に行う)
    - 前期より厳しく..
  - Web/Mail/Chat を「活用」する
    - 口を止めて、頭と手(目/耳)を動かせ

本日の予定

- 講義
  - ガイダンス
    - 前期と同じ (という事でほぼ終了)
  - 前期の復習
    - 前期の内容を概観 (これは、解っていると仮定される !!)
- 演習
  - 課題の提出

課題

□ 前回の課題

- ○ なし (最初の講議なので)

□ 今週 (2011/09/30) の課題

- ○ 課題 1:
  - ▹ ファイル名 : 20110930-1-QQQQ.c (QQQQ は学生番号)
  - ▹ 内容 : 成績処理を行う
  - ▹ 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)
  - ▹ 引数で自分の成績(S,A,B,C,D)をいれて、順位を表示させてみよ

前期の復習：プログラム

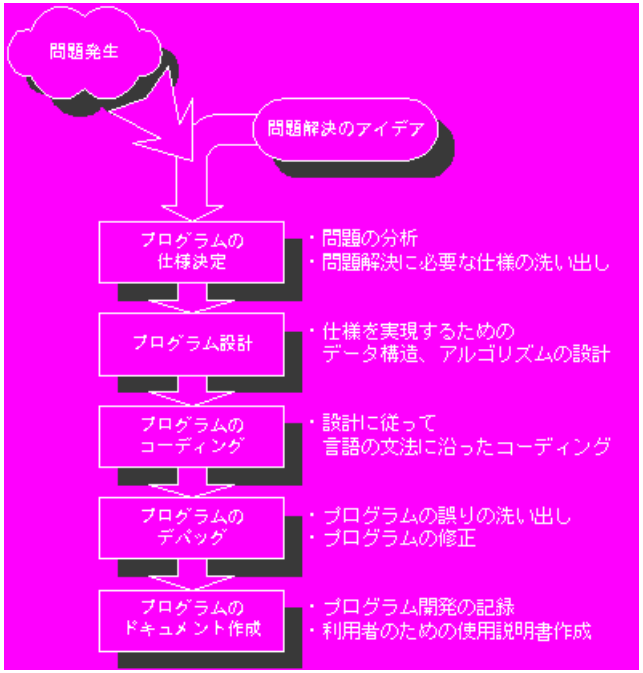
- プログラムとは
  - 命令を並べた物
    - 命令:計算機に何か(単純な事..)をさせる事ができる
    - 「どんな命令が利用できるか」は憶える必要がある
- 命令の組み合わせによって様々な機能(複雑な事)が実現できる
    - 「命令の組み合わせ方法」も考える必要がある
  - 計算機はプログラムによって動いている
    - プログラムに記述された命令を順番に実施しているだけ
    - 計算機の機能は即ちプログラムの機能
- プログラミングとは
  - 命令を自分の意図に従って並べる
    - 計算機に自分の意図通りの動作をさせる事ができる
  - 新しいプログラムを自分で作成する事ができる
- プログラミング言語とは
  - プログラムを記述するための言葉
    - 命令の書き方(単語)や組み合わせ方の規則(文法)
    - 使える命令や、組み合わせ方法は、言語によって異る

前期の復習：C 言語

- C 言語とは
  - プログラミング言語の一つ
    - プログラミング言語には色々あり得手不得手がある
  - 手続型: 命令を並べると、その命令がその順に実行される
    - 「手続(C 言語では関数)を定義する事」が「プログラミング」
  - コンパイラ型: C 言語で記述されたプログラムは翻訳される
    - C 言語で作られたプログラムは直接は実行できない(ソース)
    - C 言語から実行可能な形式への翻訳する(オブジェクト)
    - コンパイラ(ソースからオブジェクトへ翻訳するプログラム)が必要
    - オブジェクトコードはリンクされて実行形式になる
- Borland C++
  - Borland 社が提供するコンパイラ&リンカー
    - コンパイラは他にも色々ある (cf. Visual C++)
  - ソフトウェア概論ではこのコンパイラを使う
    - 実は C++ という C 言語を拡張した言語のコンパイラ

前期の復習：プログラム作成手順

- プログラム作成手順
  - 仕様: どんな機能にするか
    - プログラムの機能を考える
    - cf. 課題が出た
  - 設計: どうやって実現するか
    - 機能をどうやって実現するかを考える
    - cf. 課題を解く方法を考える
  - コード化: C 言語で記述する
    - プログラミング言語を利用して実現方法を記述
    - cf. C のプログラムを作成
  - デバッグ: 誤りがないか確認
    - プログラムが要求する機能をもっているか確認
    - cf. コンパイル/実行して機能を確認
  - 納入: 他の人に渡す
    - プログラムの使い方など資料を作る
    - cf. 課題を提出



(c) http://wwws.kobe-c.ac.jp/deguchi/c/step.html

前期の復習：プログラムファイルとツールの関係

□ プログラム作成の流れ

○ ソースファイル(*.c)

▹ C 言語で記述

▹ サクラエディタで作成

C> sakura hello.c

○ オブジェクトファイル(*.obj)

▹ ソースファイルからコンパイラで作成

C> cc -c hello.c

○ ライブラリファイル(*.lib)

▹ 予めコンパイラと一緒に配布される

▹ 自分で作ったり他から入手する場合もある

○ 実行ファイル(*.exe)

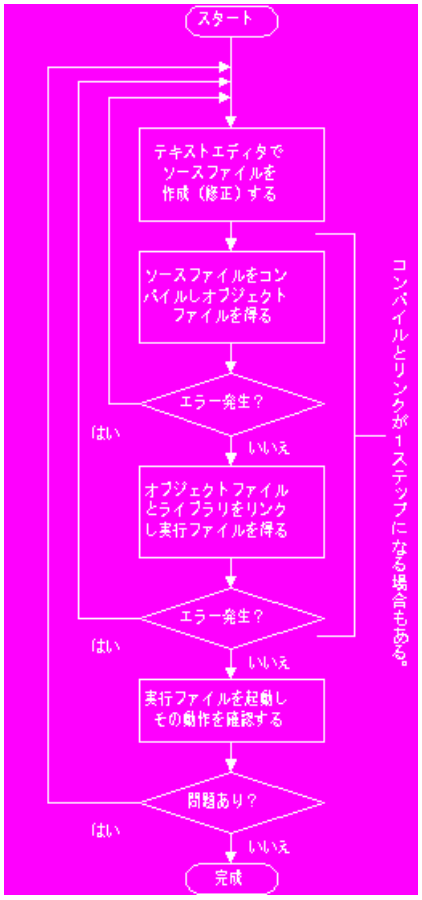
▹ オブジェクトファイルからリンカーで作成

C> cc hello.obj

○ 実行

▹ 実行ファイルを指定して実行する

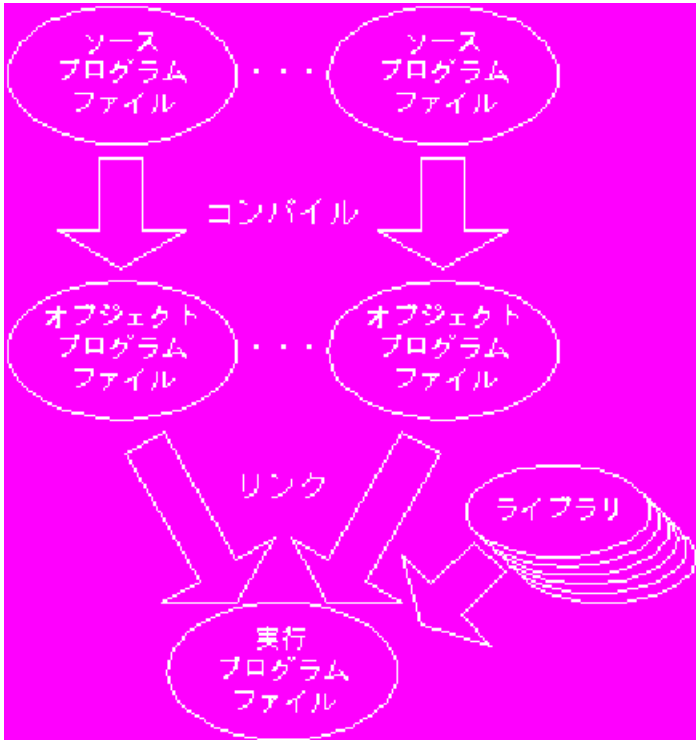
C> hello



(c) http://wwws.kobe-c.ac.jp/deguchi/c/step.html

前期の復習：様々なファイルの関係

- 様々なファイルの関係
  - ソースコード (*.c)
    - エディタで C プログラムを作成
  - オブジェクト(*.obj)
    - コンパイルがソースコードから作成
  - ライブラリ(*.lib)
    - コンパイラと一緒に配布される
  - 実行ファイル(*.exe)
    - リンカーがオブジェクトなどから作成



(c) http://wwws.kobe-c.ac.jp/deguchi/c/step.html

前期の復習：C 言語で Hello, World

- Hello, World プログラム (sample-001.c)
  - 「Hello, World[改行]」
  - 短いながら完全なプログラムで、意味がある
- プログラミング学習
  - 「動く」事が目標 (後期は「理解」も重視)
  - 差分プログラミング
    - 結果をすこしずつ作って行く
    - すでに動くプログラムの一部を変更する
- Hello, World の要素
  - main 関数
    - どのプログラムにも一つあり、最初に呼び出される
  - printf 関数
    - 「printf (引数文字列);」の形で呼出す
    - 引数文字列が画面に表示されるという「複作用」がある

前期の復習：C 言語の関数

- C 言語の関数
  - 定義: 関数は定義する事ができる
    - 「名前(型)」と「本体(手続)」をもち、それを結び付ける
    - 命令列を「{」と「}」でかこって、それに関数名をつける
    - cf. sample-001.c では「main 関数」が定義されている
  - 呼び出し: 関数は呼び出す事ができる
    - 「名前(引数並び)」で、関数を呼び出す事ができる
    - 呼び出すと本体に記述した命令列が実行される
    - cf. sample-001.c では「printf 関数」が呼び出されている
    - 関数呼び出しの結果として、関数値を得る事ができる
  - 関数呼び出しは、基本的な「命令」
    - 関数呼び出しによって、様々な命令が可能となる
    - cf.「printf 関数呼び出し」は「メッセージの表示」命令
  - 様々な標準関数 (ライブラリに定義されている)
    - printf - 文字列の出力機能を持つ
    - strlen - 文字列の長さを求める
    - strcmp - 文字列を比較する
    - 基本的な標準関数の名前と機能は、憶える(調べる知識)

前期の復習：命令の組み合わせ

- 順接 (sample-002.c)
  - 命令を並べる
    - 並べた順に実行される
- 条件分岐 (sample-003.c)
  - if 文 (switch 文) で、条件と選択肢(複数の命令)を記述する
    - 条件によって複数の命令のどれか一つが実行される
- 再帰呼び出し (sample-004.c)
  - 関数の定義の中で自分自身を呼び出す
    - 関数の本体が何度も呼び出される
    - 同じ命令を何度も繰り返す事ができる
  - 再帰呼び出しの考え方
    - 「全部」を「扱いやすい一部分」と「残り(全部)」の二つに分ける
    - 扱いやすい一部分：これは、そのまま対処してしまう
    - 残り全部：(残り)「全部」なので、再帰呼び出しする

前期の復習：コーディング

- □ 計算機で何ができるか？
  - ○ 狭い意味：計算(情報処理)
    - ▹ 数の処理
  - ○ 広い意味：何でも(計算可能ものであればなんでも)
    - ▹ 原理的に現実でできることは何でも ??
  - ○ 「狭い」と「広い」の差は？
    - ▹ 「コーディング」で結び付けられる
- □ コーディングとは
  - ○ (扱いたい)対象を別の物(計算機の場合は数値)で表現する事
    - ▹ 座標：「位置」を「数値の組」で表現
    - ▹ MIDI：「音」を「音階を表す数値」と、「音の長さを表す数値」で表現
    - ▹ URL：「Web Page の位置」を「文字列 (http://～)」で表現
    - ▹ 文字列：「文字列」を「文字の並び」で表現
    - ▹ ASCII Code：「文字」を「数値」で表現
    - ▹ 2の補数表現：「数値」を「ビット列」で表現
  - ○ 「対象」を「数値」に対応つけるルール(コーディング)を考える
    - ▹ 「数値」を操作する事が、「対象」を操作する事になる：万能性

前期の復習：分割コンパイル

- プログラム
  - 関数の集まり
    - main とその他の関数定義を、ファイル内に記述する
- プログラムとファイルの関係
  - 一つのファイルに全ての関数を記述するか？
    - No : printf などは記述されていない
- 複数のファイルに跨がる関数をどう利用する
  - 分割コンパイルする (sample-005.c, sample-006.c)
    - cc -c でコンパイルし、obj ファイルを作成
    - cc でリンクし、exe ファイルを作成する
- プロトタイプ宣言と include
  - ファイルを分割すると関数の引数が解らない
    - プロトタインプ宣言で、それを報せる
    - include でプロトタインプ宣言を共有する

前期の復習：データ型

- 値には型がある
  - 様々な型
    - 文字型(char), 整数型(int), 浮動小数点型(double), ポインタ(*), etc..
  - 型によって、計算の方法が違う
    - cf. 3/2 は 1 に 3.0/2.0 は 1.5 になる
  - 型によって、表示の方法も違う
    - s_print_*****
- 引数(変数)は、値を保持する
  - 変数には、型宣言が必要
- 関数の返り値も、型宣言が必要
- 型変換
  - 値の間で意味を保持したまま型が変更できる
    - 例：1.0(double) <-> 1(int)
    - 同じ「1」という数値だが、型が変る
  - 型変換の方法
    - 明示的に行う(キャスト)：(int)1.0 -> 1, (double)1 -> 1.0
    - 暗黙に行われる：演算の時に「型の昇格」がおきる場合がある
    - cf. char -> int, int -> double

前期の復習：メモリモデルとポインター値

- メモリ
  - 複数のセルの並び
    - 個々のセルにはアドレス(番地)がつき、区別される
    - 個々のセルは 1 byte のサイズを持つ
    - セルの並びで一つの情報を記憶する (cf. sizeof)
- 変数とは？
  - 複数の連続したセルに「型」をつけたもの
    - 型によって、振舞が異なる事に注意
- ポインター値
    - 型と「アドレス値」をもつ
    - 「アドレス値」は、メモリの番地と同じ
    - 変数名の前に '&' を付けるとポインター値が得られる
    - ポインター値の前に '*' を付けると元の変数と同じ振舞をする
    - 「 v == *(&v) 」が恒等的に成立する

前期の復習：ポインター値の計算

- ポインター値
  - 二つの情報をもつ
    - 型情報：何型の情報が入っているものか？
    - アドレス値：どこに入っているか？
- ポインター値の計算
  - 整数値 n を加える事ができる
    - 型情報は変らず、アドレス値だけが変化
    - アドレス値は n × sizeof (型) だけ変化 (n は負の数でもよい)
  - 同じ型のポインター同士なら引き算もできる
    - 結果は整数値で、(アドレス値の差) / sizeof (型) となる
    - p, q が同じポインター型なら「p + (q - p) == q」が恒等的に成立
  - キャストを利用して、型を変更できる
- ポインター値と添字
  - 恒等的に「 p[n] == *(p + n) 」が成立する

次回以後予告

- 後期の講議内容
  - いよいよ三周目が始まる
    - 一応、C 言語に関する内容はこの三周目で一通りやる
    - 項目: 代入/複雑なデータ構造/標準ライブラリ
  - C 言語を用いたアプリケーション開発
    - C 言語で色々なプログラムを作成する
    - cf. スクリプティング / 役立つ道具 / ゲーム
  - 色々なライブラリの利用法
    - 3D やります、他にも色々と...
- 次回の目玉
  - 代入：変数の値を変更する方法
  - while：もう一つの繰返し方法
  - 状態機械：変数の値の変化によって計算が進むというモデル