ソフトウェア概論 A/B

-- 引数付き関数 / 条件判定 / 繰返し / Turtle Graphics --

数学科 栗野 俊一 / 渡辺 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
- 講義開始前に済ませておく事
  - PC の電源を入れる
  - ネットワークに接続しておく事
  - 今日の資料に目を通しておく事
- 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
- 本日の CST Portal の出席パスワード : 20150501
  - 出席は成績に影響しませんが、折角の機能なので、使いましょう

前回(2015/04/24)の復習

- □ 前回(2015/04/24)の内容
  - ○ 学習の目標 : ソフトウェアの作成 (三つの異るものを習得する必要がある)
    - ▹ [作] 作成方法 : どの様にプログラムを作成するか (思考法の習得)
    - ▹ [表] 表現方法 : どの様にプログラムを表現するか (C 言語の習得)
    - ▹ [利] 利用方法 : どの様にプログラムを利用するか (操作の習得)
  - ○ 作成方法
    - ▹ 順接 : したい事をしたい順に並べる (文字列を表示する / 音を鳴らす)
    - ▹ 関数 : 命令列に名前を付け、名前を指定するだけで何度も呼び出せる
  - ○ 表現方法
    - ▹ 頭の部分 (#include ～)と尾の部分(return 0; ～) は、「お呪い」
    - ▹ 取り敢えず、「{」から「return 0;」の間に命令を並べる
    - ▹ 関数の作り方と呼び出し方
    - ▹ printf (文字列を表示する) / s_midi_play (音の鳴らす)
  - ○ 操作方法
    - ▹ コンパイルの仕方 (cc -c qqqq.c)
    - ▹ リンクの仕方 (cc -o qqqq.exe qqqq.o)
    - ▹ 実行の仕方 (./qqqq.exe)
    - ▹ make と Makefile

お知らせ

- 本日(2015/05/01)の予定
  - PC で Turtle Graphics (亀プログラム) をしてみよう
  - 引数付き関数を作ってみよう
  - 条件判定をしてみよう
  - 再帰呼び出しをしてみよう
- 本日(2015/05/01)の目標
  - プログラムの基本ブロックである関数を学ぶ
  - 演習
    - makefile と make
    - 引数付き関数の使い方と作り方
    - 亀プログラム / 文字の出力
    - 条件判定をするプログラム : 状態によって振舞を変更する
    - 再帰呼び出しをするプログラム : 同じ事を必要なだけ繰り返す
    - 課題の提出

前回 (2015/04/24) の課題

- 前回 (2015/04/24) の課題
  - 課題 1:
    - ファイル名 : 20150424-01-QQQQ.c (QQQQ は学生番号)
    - 内容 : 童謡を演奏するプログラムを作成しなさい
    - 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 : 20150424-02-QQQQ.c (QQQQ は学生番号)
    - 内容 : 童謡の歌詞を出力する
    - 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)
    - 可能な限り引数付きの関数で..

本日 (2015/05/01) の課題

- 本日 (2015/05/01) の課題 (CST Portal のみ)
  - 課題 1:
    - ファイル名 : 20150501-01-QQQQ.c (QQQQ は学生番号)
    - 内容 : 漢字の「回」という文字(にみえる..) 絵を Turtle Graphics で書きなさい
    - 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 : 20150501-02-QQQQ.c (QQQQ は学生番号)
    - 内容 : 引数付き関数で、if 文で条件判断をするプログラムを作成しなさい
    - 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 : 20150501-03-QQQQ.c (QQQQ は学生番号)
    - 内容 : 底辺の長さが指定した文字列の二倍の長さ - 1 の横向のピラミッドを作成するプログラムを作成しなさい
    - 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)
    - 再帰呼び出しを利用する

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
  - 本日のフォルダ(c:\usr\c\20150501)を作成
  - 次の本日 (2015/05/01) 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/2015/soft/20150501/20150501.html
  - ダウンロードするファイル (c:\usr\c\20150501 に保存)
    - turtle.zip
- ファイルをダンロードしたら次の作業を行う
  - ubuntu 側で ~/c/20150501/turtle に移動する
    - cd ~/c/20150501/turtle
  - 「make test」で実行して、絵が出れば良い

Turtle Graphics (亀プログラム)

- お呪い
  - #include "s_turtle.h" を冒頭にいれる
- 「亀」の操り方
  - 「亀」は、最初の状態では
    - 画面の真中にいます
    - 上を向いています
  - 「亀」への命令は次の三つ
    - s_turtle_move(); : 現在の位置に足跡を残し、現在の方向に一歩進みます
    - s_turtle_jump(); : 現在の位置に足跡を残さず、現在の方向に一歩進みます
    - s_turtle_turn(); : 現在の方向を時計回りに 45 度変更します
- 「亀」プログラムの実行手順
  - サクラエディタで、C ファイルを作成(foobar.c とする)
    - c:\usr\c\20150501\turtle に保存する
  - ubuntu で次のコマンドを実行する
    - make BASE=foobar test

分割コンパイル

- C 言語で記述されたプログラムの構造
  - main 関数が必ず必要
    - 他の関数は main 関数から呼び出される
  - 関数の定義
    - ソースファイル (*.c) の中に記述する
    - 同じファイル内である必要はない
- 分割コンパイル
  - 関数を別のファイルで定義し、個々にコンパイルする事
    - 後でリンクにより一つの実行ファイルにまとめる

make と makefile

- 分割コンパイルは複数のファイルを処理
  - 作業も面倒だし、間違いも起きやすい
    - コンパイルの手順を記述してコンピュータにやらせちゃおう
- makefile
  - コンパイルの手順などを記述したファイル
- make
  - makefile を読んで、コンパイルを自動的に行ってくれる

関数の作り方 (その 1)

- 関数の作り方(引数のない場合)
  - 名前を決める
    - cf. subfunc
  - どの部分を関数にするかを決める
  - 関数にする部分を取り出し、外に出し、ブロックにする
    - ブロックにするには '{' と '} で囲めばよい
    - 名前を付ける (cf. void subfuc())
  - もともと部分があった所に関数呼び出しを書込む
    - cf. subfunc();

関数呼び出しの挙動

- 関数呼び出しは次のように振舞う
  - 関数呼び出しのある場所から関数の先頭にゆく
  - 関数の中身を実行する
  - 関数呼び出しのある場所の次に戻る
- 関数の引数とは
  - 関数の振舞いを変更するための情報 (パラメータ)
    - 同じ関数でも引数が異れば異なる振舞いをす
- 引数付きの関数の呼び出し
  - 関数の中の変数に、引数の値が入っている

引数付き関数の作り方

- □ 引数とは
  - ○ 関数に与える事により、関数にその引数に対応した挙動をさせるもの
    - ▷ 引数付き関数の定義：引数の値によって挙動が変る
    - ▷ 引数付き関数の利用：指定したい挙動をさせるための値を指定する
    - ▷ cf. 三角関数：引数の角度によって異なる値を返す
- □ 引数付き関数の作り方
  - ○ 似ている二つ関数を一つの引数付き関数にまとめる
    - ▷ 関数の本体の部分を、同じ部分と違う部分に分ける
    - ▷ 違う部分は「変数」に置き換えて、一つの関数定義にまとめる
    - ▷ 関数の仮引数の所に、「変数」を追加する
    - ▷ 呼出す側は、実引数に、「違っていた部分の内容」を指定する

条件分岐

- 引数の内容によって振舞いを「大幅」に変更したい
  - if 文と strcmp 関数を利用して対応できる
    - strcmp 関数：二つの文字列を比較する
  - if (!strcmp (A, B)) { X } else { Y }
    - A と B が同じなら X を、そうでなければ Y を行う
  - 「else if」を使うと更に複数の命令が選べる
    - if (C1) { P1 } else if (C2) { P2 } .. else { Pn }
    - C1 の時 P1、そうでなく C2 の時は P2 .. いずれでもないと Pn
  - おまじない
    - #include <string.h>
  - strncmp (A, B, N);
    - A と B の先頭の N 文字だけを比較する
    - !strncmp ("abc", "abz", 3); : 等しくない
    - !strncmp ("abc", "abz", 2); : 等しい

再帰呼び出し

- 文字列を順番にみてゆく
  - 「"abc" + 1」は「"bc"」と同じ振舞いをする
    - どうゆう仕組かは今回は説明しない
  - 次々と 1 を加えれば、どんどん短かくなる
    - 最も短かくなったかは、空文字(「""」)と比較すれば判定できる
- 再帰呼び出し
  - 普通の関数は、別の関数を呼出す事ができた
    - 「自分の中」で「自分自身」を呼出す事ができる !! : 再帰呼び出し
- 再帰呼び出しと帰納法
  - 再帰呼び出しは、帰納法の考え方で問題を解く場合に利用できる
    - 再帰呼び出しが上手く行く事は、帰納法で証明できる (数学との関係)
  - 再帰呼び出しをする場合は次の二点が重要 (帰納法と同じ)
    - 最も小さい場合 (ここでは、文字列が "" の場合) には終了する
    - そうでない時は、再帰呼び出しするが、その時には文字列を短くする

再帰呼び出しの考え方

- 目標
  - 「全部」をやりたい
    - でも一挙にはできない
- 対策
  - そこで問題を二つに分ける
    - 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
- 注意点
  - 「全部」が空っぽの時を忘れずに処理する

三つの基本制御構造と万能性

- 三つの基本制御構造
  - f を関数, A,B を命令、p(x) を条件とする時、次の三つの基本構造がある
  - [順接] f() { A B }
    - f は A をしてから B をする
  - [分岐] f(x) { if (p(x)) { A } else { B } }
    - f は p(x) が成立すれば A そうでなければ B をする
  - [繰返] f(x) { if (p(x)) { A f(x') } else {} }
    - f は p(x) が成立する限り A を行う
    - x' は x から計算される
- 万能性
  - 任意のプログラムこの三つの基本制御構造で構成可能
    - 「三つの基本制御構造」を憶えれば、後は「組み合わせ」を考えるだけ !!