ソフトウェア概論 A/B

-- 再帰呼び出し / 文字と文字列 / ミク プログラム --

数学科 栗野 俊一 / 渡辺 俊一

2013/05/17 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

前回(2013/05/10)の復習

- 前回(2013/05/10)の内容
  - 作成方法
    - 引数付き関数：引数によって関数の振舞を連続的に変える(関数の汎用化)
    - 条件分岐：条件判断を利用して、振舞を不連続的に変える
  - 表現方法
    - 引数付き関数の作り方と呼び出し方
    - if 文と strcmp による文字列の比較による条件判断の書き方
    - Turtle Graphics (亀プログラム) の作成方法
  - 操作方法
    - make で 独自の作成規則(makefile)を指定する方法
      - 「make」とすると「makefile」を作成規則として利用する
      - 「makefile」以外の作成規則を指定した場合 -f を利用する
      - 「make -f 独自の作成規則がかかれたファイルの名前」」
    - 独自の作成規則ファイル(makefile)の作り方
      - 「copy makefie 独自のファイル名」:新しい作成規則ファイルを作る
      - 「sakura 独自のファイル名」: 作成規則ファイルの内容を変更
      - 「make -f 独自のファイル名」:作成規則ファイルを利用して make

お知らせ

- 本日(2013/05/17)の予定
  - 復習
    - 引数付き関数を作ってみよう
    - 条件判定をしてみよう
    - 独自の作成規則ファイル(makefile) の作成して turtle graphics を行う
  - 新規
    - 再帰呼び出しをしてみよう
    - 文字列と文字の関係
- 本日(2013/05/17)の目標
  - プログラムの三つの基本構造を把握する
    - 順接 / 条件分岐 / 繰返し
  - 演習
    - makefile と make と分割コンパイル
    - 文字の出力
    - 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 (2013/05/10) の課題

□ 前回 (2013/05/10) の課題

- ○ 課題 1:
  - ▹ ファイル名 : 20130510-1-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 : 20130510-2-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 : 20130510-3-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
  - ▹ 注意 : これは今回[2013/05/17]の課題にとするが課題名は先週の日付のまま出す

本日 (2013/05/17) の課題

- 本日 (2013/05/17) の課題
  - 課題 1:
    - ファイル名 : 20130517-1-QQQQ.c (QQQQ は学生番号)
    - 内容 : 渦巻模様 (10 辺) の絵を Turtle Graphics で書なさい
    - 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

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

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

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 (2013/05/17) 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/2013/soft/2013/05/17/2013/05/17.html
  - ダウンロードするファイル
    - 2013/05/17-update.zip
  - 展開して出て来た、三つのフォルダを「c:\usr\c」に移動する事
- ファイルをダンロードしたら次の作業を行う
  - コマンドプロンプトを開く
  - 「cd c:\usr\c」とする
  - 「mingwenv」とする
  - 「cd 2013/05/17\miku」とする
  - 「make」でコンパイル、リンク
  - 「walk2」で実行して、ミクが歩けばよい

ミクプログラム

- おまじない
  - #include "s_walk.h" を冒頭にいれる
- 「ミク」の操り方
  - 「ミク」は、最初の状態では
    - 画面の真中にいます
    - 前を向いています
  - 「ミク」への命令は次の四つ
    - void s_walk_move(void); 少し、歩く
    - void s_walk_jump(void); 少し、走る
    - void s_walk_turn(void); 45 度向きを変える
    - void s_walk_wait(void); しばらく、体を揺らして待っている

引数付き関数の作り方 (復習)

- □ 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 (復習)

- 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」を憶えればあとは組み合わせを考えるだけ !!

文字列と文字

- C 言語での「文字」の扱い
  - 文字：文字をシングルクォート(「’」)ではさんだもの
    - cf. ’A’, ’a’, ’1’
    - 「一文字」と対応している「表現」
  - 文字の出力：putchar 関数を利用する
    - cf. putchar (’A’);
  - 文字列：文字列をダブルクォート(「"」)ではさんだもの
    - cf. "ABC", "123", "" (空文字列)
    - 「文字の並び(0個以上)」と対応している「表現」
    - <<注意>>：全角(日本語)は、一文字で、二文字分になる
  - 文字列の出力：printf 関数を利用する
    - cf. printf ("abc");

文字列の構造

- 文字列は、文字の並び + 文字の終りからなる
  - "ABC" == { 'A', 'B', 'C', '\0' }
    - 長さ 3 (n) の文字列は、4 (n+1) つの部分からなる
    - '\0' を EOS (End Of String) と呼ぶ
  - k 番目の文字の取り出し方 : [k] をつける (k は 0 から始まる事に注意)
    - cf. "ABC"[0] == 'A', "ABC"[3] == '\0', "ABC"[9] == ? (未定義)
  - 先頭の文字を取り出すには * を付けてもよい
    - cf. *"ABC" == 'A'
  - 先頭の文字を取り除いた残りを取り出すには 1 を加えればよい
    - cf. "ABC" + 1 == "BC"
- 文字列の判定 : strcmp を使うと、「二つの文字列が同じなら偽」になる
  - cf. strcmp ("ABC", "ABC") ==> 偽, strcmp ("ABC", "XYZ") ==> 真
  - 文字列が「空文字列(長さが 0 / 文字を含まない文字列)」は先頭が EOS かどうかで判定できる
    - (*"" == '\0') ==> 真