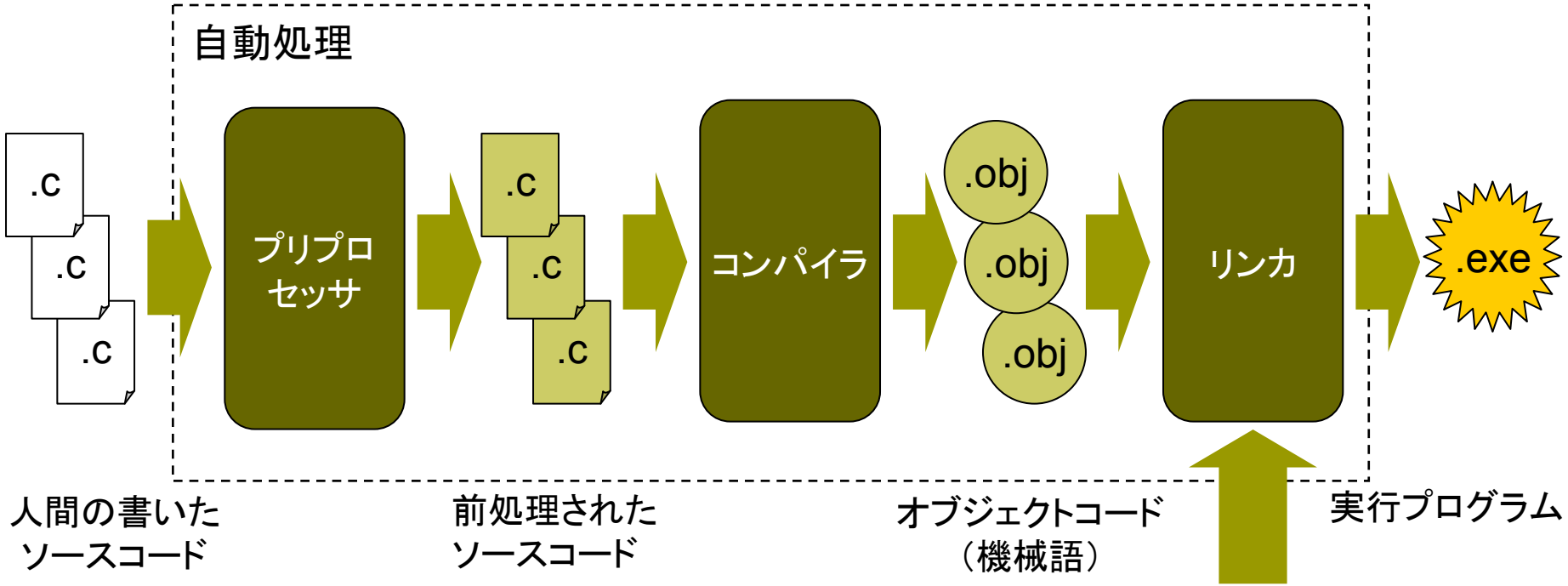
Programming I 0x0f

演習(2010.07.12)

塩澤秀和 http://vilab.org

コンパイラの働き（復習）



コンパイラの役割（p.2）
人間がプログラム言語で書いたソースコードを
コンピュータが分かる機械語に翻訳する

マクロ定数(復習)

- プリプロセッサとは(p.93)
  - 前処理(コンパイルする前の処理)をするソフトウェア
  - 「#」で始まるプリプロセッサ指令に従って、ソースコードを“加工”する
- #define 指令(p.95)
  - 書式:「#define　マクロ名　文字列」
  - ソースコード内のマクロ名を、すべて文字列で置き換える
    例: #define PI 3.14　← 以後、「PI」と書くと「3.14」と同じ意味になる
  - さらに高度な #define ⇒ 引数付きマクロ(p.97)
- マクロの利点
  - 定数に名前をつけると、プログラムの意味が分かりやすくなる
  - プログラムを修正するとき、1ヶ所の定義を変更すればよい
  - 単なる文字列の置き換えなので、変数のようにメモリを使わない

ヘッダファイル（復習）

- #include 指令（p.94）
  - 書式：「#include <ファイル名>」または「#include "ファイル名"」
  - 指定したファイルの内容を、その場所に丸ごと取り込む
  - <ファイル名> なら、コンパイラ設定のフォルダからファイルを探す
  - "ファイル名" なら、ソースコードと同じフォルダからファイルを探す

- ヘッダファイル（インクルードファイル）とは
  - #include で読み込むために用意されたファイル（拡張子「.h」）
  - 関数プロトタイプ宣言、マクロ定数の定義などを書いておく
  - ライブラリ関数を使うには、それぞれ指定のヘッダファイルが必要
    - C:¥Program Files¥Microsoft Visual Studio 9.0¥VC¥include¥

- その他のプリプロセッサ指令
  - #if, #ifdef, #endif, #line など ⇒ 教科書 p.98～100 を読んでおく

演習問題

15a. 0以上の整数を次々にキーボードから読み込んで合計を計算し、負数が入力された時点で、その前までの合計結果を表示して終了するプログラムを作成しなさい。（最後の負数を足さないように注意！）

15b. 実数x, min, maxを引数に取り、下記の数式で表される値を返す関数constrainを作成しなさい。適当なmain関数をつけて動作を確認しなさい。

$$constrain(x, min, max) = \begin{cases} min & x < min \text{ のとき} \\ x & min \le x \le max \text{ のとき} \\ max & x > max \text{ のとき} \end{cases}$$

15c. 下記の数列を a_1 から a_{10} まで表示するプログラムを作成しなさい。

$a_1 = 1,\ a_n = 2\,a_{n-1} + 3$（n≧2のとき）

15d. 4つの整数を引数に取り、先頭の数よりも大きい数の個数を返す関数を作成しなさい。適当なmain関数をつけて動作を確認しなさい。

応用問題

15e. 月利x%の利子がつく銀行預金に、毎月y円ずつ積み立てる。たとえば、x=1, y=1000なら、1ヵ月後には2010円（1000×1.01＋1000）、2ヵ月後には3030円になる（1円未満は切り捨て）。標準入力からx, yを読み込み、何年何ヶ月後に100万円を超えるか表示するプログラムを作成しなさい。

15f. 下記の数列を a_1 からa_{10} まで表示するプログラムを作成しなさい。ただし配列や再帰は使わず、ループで計算すること。

$$a_1 = 1, \quad a_2 = 1, \quad a_n = a_{n-1} + a_{n-2} \quad (n \geqq 3のとき)$$

15g. double型の引数 a, b, c を3辺の長さとする図形が、正三角形か（3）、二等辺三角形か（2）、不等辺三角形か（1）、三角形にならないか（0）を判定し、結果を0～3の整数で返す関数を作成しなさい。

15h. 次のように定義される再帰関数 gcd によって、整数xとyの最大公約数を求めることができる。プログラムを作成してみなさい。

$$gcd(x, y) = \begin{cases} x & y = 0のとき \\ gcd(y, r) & rはx \div yのあまり \end{cases}$$

プログラミングの考えかた(復習)

- 問題文を読んで大まかな“手順”を考える
  - いきなりプログラムを書く前に、まず問題文をよ～くよ～く読む
  - 入力は何か？　出力は何か？　何のためのプログラムなのか？
  - 公式で計算できるか？　どういう順番で計算すればいいか？
- どんな“パターン”が使えるか考える
  - いままで習ったプログラムで、似ているものはあったか？
  - 知っているパターンを、いくつか組み合わせると解けそうか？
  - パターンを思い出して、大まかな構造と必要そうな変数を考えよう
- 頭の中で“実行”するつもりでプログラムを作る
  - コンピュータがどう動くか考えながら、1行1行のプログラムを書く
  - 追加の変数が必要になったら、先頭に戻って付け加えればよい

統一テスト

- 7月21日（水）17:00～19:00　421教室
  - 範囲：主に、ループ（while, for）と関数の自作（～p.100）
  - 形式：中間テストと同じような形式
  - 対策：後半の演習問題がすぐに解けるように復習する
    ドリルのプログラムも、見てすぐ意味が分かるようにする

ソースコード	意味（復習）
#include <stdio.h>	標準入出力ライブラリのヘッダファイル（stdio.h）を読み込む。 #includeは「プリプロセッサ指令」と呼ばれるものの一種であり、 printfやscanfをプログラムで使用するときにはこの行が必要。
int main(void) {	main関数の定義の始まり。プログラムを起動すると、main関数 の定義の { } で囲まれた部分を順番に実行する。
return 0; }	main関数を正常に終了させる（0は正常終了の意味）。 最後の } はmainの定義の範囲の終わりを示す。