プログラミング基礎'94 ＃5

久野　靖*

1994.7.26

いよいよ最終回です。残っている内容から、ファイルの読み書きと前回積み残した関数の話をやりますが、併せて再帰呼び出しも取り上げます。「各種の型」をやろうかとも思っていたのですが、それよりもこれまでに学んだことを活用してイメージしやすい課題をやる方が有益だと思ったので、時間の許す範囲でいろいろな問題を解いてみましょう。

13　ファイル入出力

これまでのプログラムでは、入力はすべてキーボードから、出力はすべて画面へだった。しかしやりたいことによってはそれでは不便なので、ファイル入出力の説明をしておこう。

これまで、プログラムのはじまりは「おまじない」として

```
program プログラム名 (input, output);
```

と書いてもらっていたが、実はこの input と output は標準入出力つまり C でいえばファイルディスクリプタ 0 番と 1 番に当っていた。そして、これら以外のファイルを使いたい場合にはここに別の名前をつけ加える。ここでは datafile という名前を使おう。

```
program table(input, output, datafile);
type str = array[1..20] of char;
var  datafile: text;
     name: array[1..100] of str;
     ...
```

このように、datafile は text 型の変数として宣言する。text 型というのは、要は文字を読み書きするためのファイルを意味している。実は input や output も text 型なのだが、これは毎回宣言しなくても自動的に用意されるというだけである。

次に、この datafile 変数を使って読み書きするには、C などと同様ファイルを OPEN しなければならない。Pascal ではそのために次の手続きを使う。

```
reset(datafile, 入力ファイル名);  --- 入力オープン
reset(datafile, 出力ファイル名);  --- 出力オープン
```

ここでファイル名は文字列を直接書いてもいいし、str 型など文字の配列を内容とする変数であってもよい。

そして、読み書き自体は

*筑波大学大学院経営システム科学専攻

```
read(datafile, ...);
write(datafile, ...);
```

のように read、write(readln、writeln でもよい) の第 1 実引数として text 型変数を渡すことでオープンしたファイルが読み書きできる。(実は text 型変数を省略すると input と output がそれぞれ使われる。)

これらの機能を利用して、例の個人データプログラムを改良して内容をファイルに保存できるようにしたものを示しておく。

```
program table3(input, output, datafile);
type str = array[1..20] of char;
var  datafile: text;
     name: array[1..100] of str;
     age:  array[1..100] of integer;
     weight: array[1..100] of real;
     n: integer;
     c: char;

procedure swapitems(i:integer; j:integer);
var n:str;
    w:real;
    a:integer;
begin
  n := name[i]; name[i] := name[j]; name[j] := n;
  a := age[i]; age[i] := age[j]; age[j] := a;
  w := weight[i]; weight[i] := weight[j]; weight[j] := w
end;

procedure listallitems;
var i: integer;
begin
  for i := 1 to n do begin
    writeln(name[i]:12, age[i]:2, weight[i]:8:2)
  end
end;

procedure insert1item;
begin
  if n >= 100 then begin
    writeln('table overflow.')
  end
  else begin
    n := n + 1;
    write(' name> '); readln(name[n]);
    write(' age>  '); readln(age[n]);
    write(' weight> '); readln(weight[n])
  end
```

```
 end;

procedure lookupname;
var sname:str;
    i: integer;
begin
  write(' name to lookup> '); readln(sname);
  for i := 1 to n do begin
    if name[i] = sname then begin writeln('no = ', i:1) end
  end
end;

procedure swaptwo;
var x, y:integer;
begin
  write(' exchange> '); readln(x);
  write('     with> '); readln(y);
  if (x >= 1) and (x <= n) and
     (y >= 1) and (y <= n) then begin swapitems(x, y) end
  else begin writeln('specified item no. out of range') end
end;

procedure sorttable;
var flag: integer;
    i: integer;
begin
  flag := 1;
  while flag = 1 do begin
    flag := 0;
    for i := 1 to n-1 do begin
      if name[i] > name[i+1] then begin
        swapitems(i, i+1); flag := 1
      end
    end
  end
end;

procedure readfromfile;
var fname: str;
begin
  write('input file> '); readln(fname); reset(datafile, fname);
  while not eof(datafile) and (n < 100) do begin
    n := n + 1;
    readln(datafile, name[n]); readln(datafile, age[n]);
    readln(datafile, weight[n])
  end
```

```
end;

procedure writetofile;
var fname: str;
    i: integer;
begin
  write('output file> '); readln(fname); rewrite(datafile, fname);
  for i := 1 to n do begin
    writeln(datafile, name[i]); writeln(datafile, age[i]:0);
    writeln(datafile, weight[i]:5:2)
  end
end;

begin
  n := 0;
  writeln('A simple table manager; type "?" for help.');
  write('Command> '); readln(c);
  while c <> 'q' do begin
    if c = 'i' then begin insert1item end
    else if c = 'l' then begin listallitems end
    else if c = 's' then begin lookupname end
    else if c = 'x' then begin swaptwo end
    else if c = 'o' then begin sorttable end
    else if c = 'r' then begin readfromfile end
    else if c = 'w' then begin writetofile end
    else if c = '?' then begin
      writeln(' list of commands are:');
      writeln('  i : insert new item');
      writeln('  l : list all items');
      writeln('  s : search item no. from name');
      writeln('  x : exchange two items');
      writeln('  o : order items by name');
      writeln('  w : write to file');
      writeln('  r : read from file')
    end
    else begin
      writeln('unknown command: ', c);
    end;
    write('Command> '); readln(c)
  end
end.
```

14 関数 (改定版)

既に見てきたように、平方根を求めるなどの計算はハードウェアの命令としてはなく、それアルゴリズムによって実現している。しかしプログラムの上では

```
sd := sqrt(v);
```

などのように短くそれらしく書けると嬉しい。そのためには、ちょうど手続きと同じように一連の計算を実行し、ただし最後に 1 つの値を結果として返すような機能があればよい。これを「関数」という。

Pascal では関数はほとんど手続きと同様の構文で定義する。ただし違うのは次の点である。

- `procedure` の代りに `function` と書く。
- 頭書きの末尾でその関数が返す型を指定する。
- 本体の中で、返したい値を関数名に代入する。

例えば整数の絶対値を計算する関数は次のようになる。

```
function iabs(x:integer): integer;
begin
  if x >= 0 then begin iabs := x end else iabs := -x end
end;
```

練習 5-1 上の関数を利用して整数の絶対値を計算するプログラムを設計し、Pascal に直して動かせ。

練習 5-2 次のような関数を作れ (まず PAD を書き、次に Pascal で)。

(a) 2 つの整数のうち大きい方 (等しければその値) を返す関数 `imax2`。
(b) 3 つの整数のうち最大のものを返す関数 `imax3`。
(c) 4 つの整数のうち最大のものを返す関数 `imax4`。
(d) 1 つの正整数を受け取りその階乗を返す関数 `fact`。
(e) 2 つの正整数を受け取りその最小公倍数を返す関数 `gcd`。
(f) n、r を受け取り、n 個のものから r 個選ぶ組合せを計算する関数 `comb`。

いずれも前問と同様の主プログラムを用意する必要がある。この手の主プログラムを (関数を drive するのが目的だから) ドライバなどと呼ぶ。

15 再帰呼び出し

プログラミングの世界では再帰という概念が多く使われるし、実際とても有効である。例えば図 1 の PAD を見て頂こう。

主プログラムはどうということはないが、関数 `fact` が変わっている。つまり、階乗なのにループがない。その変わり、`fact` の計算の中で `fact` を使っている。このようなのを再帰呼び出しという。これでは堂々めぐりと思いますか? よく考えるとそうではない。というのは、図 2 のようにだんだん問題が簡単になっていって、最後は $fact(0) = 1$ に帰着するからそこから計算しながら戻ってくればよいのである。

このような再帰関数は漸化式を書いてみるとよく対応がわかる。

$$fact(n) = \begin{cases} 0 & (n = 0) \\ n \times fact(n-1) & (n > 0) \end{cases}$$

練習 5-3 次のような関数を、再帰を用いて作れ (まず PAD を書き、次に Pascal で)。

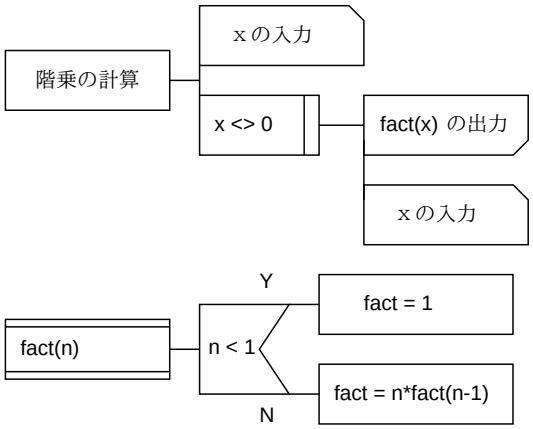

図 1: 階乗の計算 (再帰関数版)

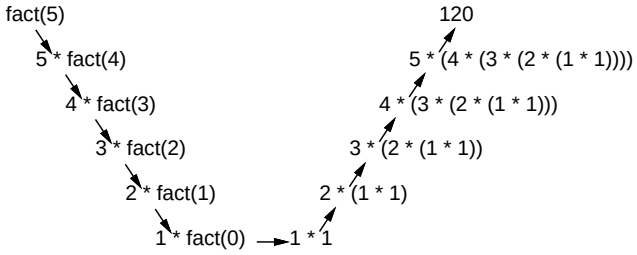

図 2: 階乗の再帰計算の流れ

(a) 正の整数 n を受け取り、1 から n までの自乗和を返す関数 `sqsum`。

(b) 2 つの正整数を受け取りその最小公倍数を返す関数 `gcd`。

(c) n、r を受け取り、n 個のものから r 個選ぶ組合せを計算する関数 `comb`。

16 さまざまな練習問題

さて、最後にお楽しみ (?!) として、手頃な練習問題をいくつかやってみよう。これまに学んだ様々な概念をうまく組み合わせるのは楽しいものである (と思うけど?)。

練習 5-4 1〜365 の整数を入れると、今年の始めからその日数たった日は何月何日かを教えてくれるプログラムを作れ。

練習 5-5 月と日を入れると、それが今年の始めから数えて何日目かを教えてくれるプログラムを作れ。

練習 5-6 今年の月と日を入れると、それが何曜日かを教えてくれるプログラムを作れ。

練習 5-7 A さんと B さんがそれぞれのジャンケンの手を入力すると、どちらが勝ったかまたは引き分けかを判定するプログラムを作れ。

練習 5-8 1〜1000 までの間の素数を出力するプログラムを作れ。

練習 5-9 x、y、z が 1〜100 までの整数だとして、$x^2 + y^2 = z^2$ を満たすような場合すべてを打ち出すプログラムを作れ。

17　課題

この講義は最初はアルゴリズムとは何かという話から始まったが、各種概念と併せて Pascal の説明もしてきたので、ここまでで各種の型とポインタとファイルの一部を除いては一通り説明したことになる。「ザ・Pascal」でいえば 4 章まで終ったわけである。わずか 5 回なのだからこれで上出来と思う。

それで、自主ゼミ科目なので厳しい課題を出すつもりは全然なく、次のようにさせて頂く。

- 明後日 (28 日) 一杯までに、今回の講義で学んだことの「まとめ」と「感想、コメント」を `gssm.lectures` にポストする (出席点がわり)。1 画面 (24 行) 程度でよい。
- 8 月末日までに、「ザ・Pascal」の本に載っている演習問題から任意に 3 課題選んでプログラムを作って動かし、レポートにして提出する。各課題ごとに課題の番号、プログラムの説明、リスティング、実行例、考察をつけ、最後に全体のまとめ (感想、反省、全体的な考察など) をつけること。