数学実習 2011 年度第 6 回
担当：中口悦史

今週の講義メモ：文字列データ，再帰呼出し

文字列

- 文字列データ：引用符で囲まれた部分 '...'
  例： 'A', 'ZZZ', '3.14159', ' こんにちは。', 'I can''t speak Japanese.' など
- 文字列型のデータ型名：FreePascal では string という型名が使える。
- 文字列の変数と配列：string 型で宣言
  `var s: string`　　文字列型変数の宣言
  `var a: array [1..<m>] of string`　　文字列を<m>個保管できる配列

データの入力（復習）

- キーボードからの入力
  read(q) 入力された文字列を変数 q へ保存
  ただし，変数 q が文字列型なら文字列のまま，数値型なら適当な数値に変換される

再帰呼出し (recursion)　手続きや関数の中で，自分自身を呼び出すこと。特に引数の値を変えて呼び出すことで，繰返し処理や処理の分割が可能になる。

例 1. 整数 k の階乗を返す関数

```
function fact( k: integer ): real;
begin
  if (k<0) then       fact := 0
  else if (k=0) then  fact := 1
  else                fact := fact(k-1) * k
end;
```

プログラム本体の直前（本体が始まる begin の前）にこの定義を書いておけば，本体中で変数のように fact(n) と書くだけで，n! の値になる。

例 2. i 番目のフィボナッチ数を返す関数

```
function fibo( i: integer ): integer;
begin
  if (i<=0) then      fibo := 0
  else if (i=1) then  fibo := 1
  else                fibo := fibo(i-1) + fibo(i-2)
end;
```

i が 2 以上のときは，fibo(i-1) と fibo(i-2) を呼び出して，漸化式を計算する。

例 3.

```
procedure multiwrite( n: integer );
begin
  if (n>1) then  multiwrite( n-1 );
  write( n, ' ' );
  if (n>1) then  multiwrite( n-1 )
end;
```

プログラム本体で multiwrite(3); と書いておくと，1 2 1 3 1 2 1 と表示される，はず。

今週のサンプルプログラム

(6a) [Renshu6a.pas] 文字列の入出力

```
program Renshu6a( input, output );
var s: string;
begin
  writeln( ’コンニチハ。’ );
  write( ’オナマエヲイレテクダサイ：’ );
  read( s );
  writeln( ’コンニチハ, ’, s, ’サン。’ )
end.
```

(6b) [Renshu6b.pas] 文字列配列：n 月は英語で？

```
program Renshu6b( input, output );
var emonth: array [1..12] of string;
    i: integer;
begin
  emonth[ 1] := ’January’;   emonth[ 2] := ’February’;
  emonth[ 3] := ’March’;     emonth[ 4] := ’April’;
  emonth[ 5] := ’May’;       emonth[ 6] := ’June’;
  emonth[ 7] := ’July’;      emonth[ 8] := ’August’;
  emonth[ 9] := ’September’; emonth[10] := ’October’;
  emonth[11] := ’November’;  emonth[12] := ’December’;

  writeln( ’The 4-th month is ’, emonth[4], ’ in English.’ );
  i := 8;
  writeln( ’What month is ’, emonth[i], ’ in Japanese ?’ )
end.
```

(6c) [Renshu6c.pas] 再帰呼出しの例：フラクタル出力

```
program Renshu6c( input, output );
var k: integer;

  procedure multiwrite( n: integer );
  begin
    if (n>1) then  multiwrite( n-1 );
    write( n, ’ ’ );
    if (n>1) then  multiwrite( n-1 )
  end;  {end of definition of multiwrite}

begin
  read( k );
  multiwrite( k );
  writeln
end.
```