情報とコンピュータ

静岡大学工学部
安藤和敏

2004.11.22

4章
トップダウンプログラミング，サブルーチン，データベースの応用

- サブルーチン
- 内部変数を持つサブルーチン
- 配列を引数とするサブルーチン
- サブルーチン間通信の例

Pascalにおけるサブルーチンの定義

```
procedure Byline;
  begin
  writeLn('-----------------------------');
  writeLn(' This program is written by  ');
  writeLn('     Lady Emily Dunsmore     ');
  writeLn('-----------------------------');
  end;
```

プログラムの中に
`Byline;`
という一文を入れる
だけで右の4行が
画面に表示される.

```
-----------------------------
 This program is written by
     Lady Emily Dunsmore
-----------------------------
```

サブルーチン呼び出し

```
program FirstSubrutine(input, output);
var ;{変数宣言}
procedure Byline;
  begin
  writeLn('-----------------------------');
  writeLn(' This program is written by  ');
  writeLn('     Lady Emily Dunsmore     ');
  writeLn('-----------------------------');
  end;
begin
Byline;
…
Byline;
end.
```

主プログラムの中に入れられた
`Byline;`
という文は，**サブルーチン呼び出し**と呼ばれる．

引数を持つサブルーチン

```
procedure Byline2(var name: string);
  begin
  writeLn('------------------------------');
  writeLn(' This program is written by  ');
  writeLn('      ',name);
  writeLn('------------------------------');
  end;
```

name という新しい変数は，出力したい名前（が格納されているメモリ中の記憶領域を）指し示す．

このような変数は，サブルーチンの仮引数と呼ばれる．

仮引数の宣言は，サブルーチン名の後に括弧に入れておく．

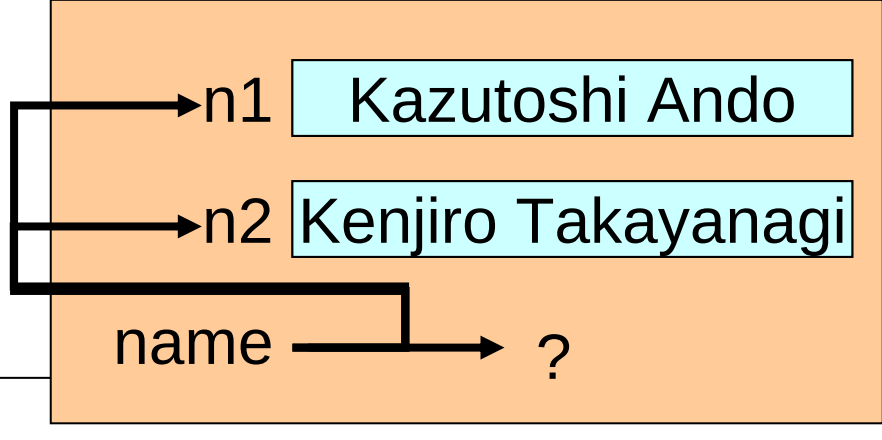
プログラムSecondSubrutine

```
program SecondSubrutine(input, output);
var n1,n2,n3: string;
procedure Byline2(var name: string);
  begin
  writeLn('-----------------------------');
  writeLn(' This program is written by  ');
  writeLn('     ',name);
  writeLn('-----------------------------');
  end;
begin
n1 := 'Kazutoshi Ando';
n2 := 'Kenjiro Takayanagi';
Byline2(n2);
Byline2(n1);
end.
```

Byline2の呼び出しにおいて，サブルーチンに渡されているn1とn2は，**実引数**と呼ばれる．

SecondSubrutine実行の状況

```
program SecondSubrutine(input, output);
var n1,n2: string;
procedure Byline2(var name: string);
  begin
  writeLn('-----------------------------');
  writeLn(' This program is written by  ');
  writeLn('     ',name);
  writeLn('-----------------------------');
  end;
begin
n1 := 'Kazutoshi Ando';
n2 := 'Kenjiro Takayanagi';
Byline2(n2);
Byline2(n1);
end.
```

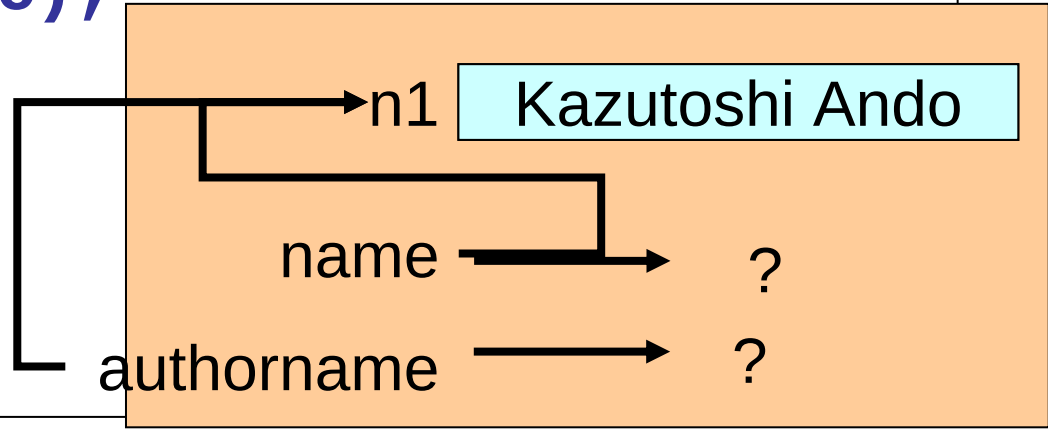


プログラムThirdSubrutine

```
program ThirdSubrutine(input, output);
var n1: string;
procedure Byline2(var name: string);
  {省略}
procedure Getname(var authorname: string);
  begin
  writeLn('Type your name.');
  readLn(authorname);
  end;
begin
GetName(n1);
Byline2(n1);
end.
```

ThirdSubrutine実行の状況

```
program ThirdSubrutine(input, output);
var n1: string;
procedure Byline2(var name: string);
  {省略}
procedure Getname(var authorname: string);
  begin
  writeLn('Type your name.');
  readLn(authorname);
  end;
begin
Getname(n1);
Byline2(n1);
end.
```

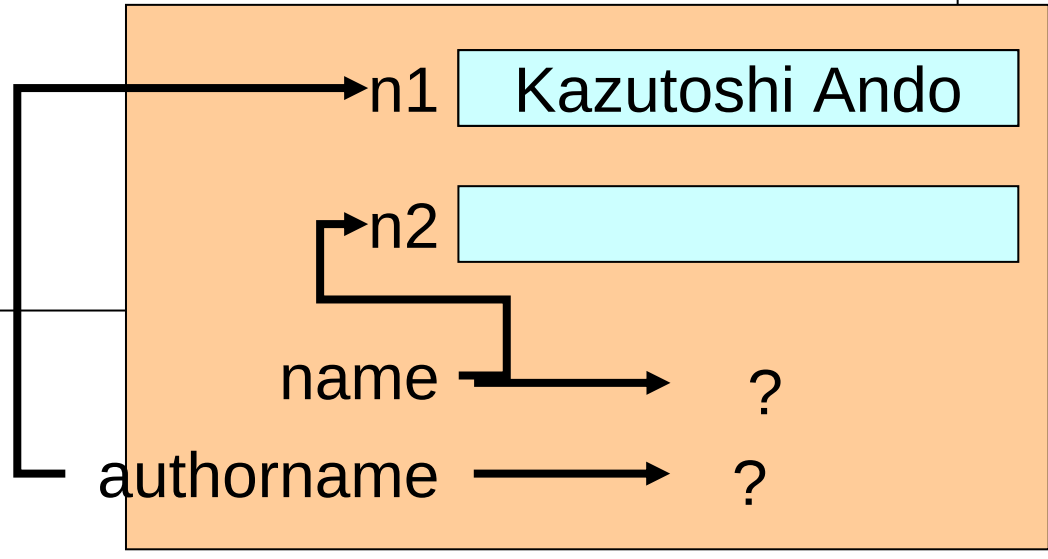


プログラムThirdSubrutine1

```
program ThirdSubrutine1(input, output);
var n1, n2: string;
procedure Byline2(var name: string);
  {省略}
procedure Getname(var authorname: string);
  begin
  writeLn('Type your name.');
  readLn(authorname);
  end;
begin
GetName(n1);
Byline2(n2);
end.
```

ThirdSubrutine1実行の状況

```
program ThirdSubrutine1(input, output);
var n1, n2: string;
procedure Byline2(var name: string);
  {省略}
procedure Getname(var authorname: string);
  {省略}
begin
Getname(n1);
Byline2(n2);
end.
```



内部変数を持つサブルーチン

サブルーチンの仮引数は，記憶場所をもたないが，サブルーチンの内部で記憶場所を必要とする場合がある．

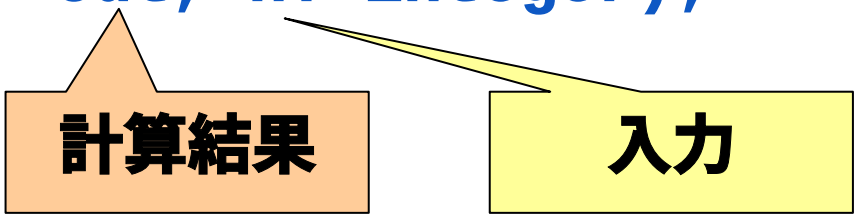
サブルーチンの内部で，記憶場所を持つ変数を使う方法を以下に示す．

内部変数を持つサブルーチン

```
var x, m, y, z3, num1: integer;
begin
...
z3 := xの階乗
...
y := mの階乗
...
num1 := yの階乗
...
end.
```

内部変数を持つサブルーチン

```
Program Factorial2(input, output);
var x, m, y, z3, y, num1: integer;
Factorial(var out, n: integer);
  {後で}
begin
...
Factorial(z3, x); {z3 := xの階乗}
...
Factorial(y, m);  {y := mの階乗}
...
Factorial(num1, y); {num1 := yの階乗}
...
end.
```



プログラム Factorial

```
program Factorial(input, output);
var
  i, n, product: real;
begin
readln(n);
product := 1;
i :=1;
while i <= n do
  begin
  product := product * i;
  i := i + 1;
  end;
writeLn(n:8:2,'の階乗は,',product:8:2,'
  です.');
end.
```

サブルーチンFactorial

```
procedure Factorial(var out, n: integer);
  var
    i: integer;
  begin
  out := 1;
  i := 1;
  while i <= n do
    begin
    out := i * out;
    i := i + 1;
    end;
  end;
```

内部変数の宣言

サブルーチンPower

```
procedure Power(var out, a, n: integer);
  var
    i: integer;
  begin
  out := 1;
  i := 1;
  while i <= n do
    begin
    out := out * a;
    i := i + 1;
    end;
  end;
```

内部変数の宣言

プログラム Factorial2

```
program Factorial2(input, output);
var n0, product: integer;
procedure Factorial(var out, n: integer);
  {省略}
begin
n0 := 10;
Factorial(product, n0);
writeLn(n0,'の階乗は,',product,'です.');
end.
```

サブルーチン内に現れる変数について，この本で用いられる規則

サブルーチン内に現れる全ての変数は，以下の2種類に限る
- そのサブルーチンの仮引数
- そのサブルーチンの内部変数

プログラム Factorial3

```
program Factorial3(input, output);
var i, n0, product: integer;
procedure Factorial(var out, n: integer);
{  var i: integer; 内部変数を使わない}
  begin
  {前と同じなので省略}
  end;
begin
i:=100;
n0 := 10;
Factorial(product, n0);
writeLn(n0,'の階乗は,',product,'です.');
writeLn(' i = ', i);
end.
```

配列を引数とするサブルーチン

サブルーチンは，配列を仮引数として扱うこともできる．

ここでは，(配列に記憶されている) 数列の総和を求めるプログラムを例にとって説明する．

プログラムは，整数を読み込む ReadArray と読み込んだ整数を全て足し合わせる AddArray の2つの手続きを用いる．

integerarray100という型が，

```
type
  integerarray100 = array[1..100] of integer;
```

と宣言されていると仮定する．

サブルーチンReadArray

```
procedure ReadArray(var n: integer;
                    var B: integerarray100);
  var i: integer;
  begin
  writeLn('How many entries?');
  readLn(n);
  i := 1;
  while i<=n do
    begin
    writeLn('Input entry ',i);
    readln(B[i]);
    i := i+1;
    end;
  end;
```

サブルーチンAddArray

```
procedure AddArray(var n: integer;
                   var C: integerarray100;
                   var answer: integer);
  var
    i: integer;
  begin
  answer := 0;
  i := 1;
  while i <= n do
    begin
    answer := C[i] + answer;
    i := i+1;
    end;
  end;
```

プログラム ReadAddArray

```
program ReadAddArray(input, output);
type
  integerarray100 = array[1..100] of integer;
var
  n, answer: integer;
  A: integerarray100;
procedure ReadArray(...);
procedure AddArray(...);
begin
writeLn('Read:');
ReadArray(n,A);
writeLn('Add up.');
AddArray(n,A,answer);
writeLn('The answer: ',answer);
end.
```

レポートについて

- **課題：次の (1) か (2) のどちらかを選択しなさい．**

  (1) 前回のレポート「あなたが興味のある事柄に関して決定木を作り，それをプログラムに書け．」のプログラムを，サブルーチンを用いて単純化せよ．(ヒントは，p. 153の練習問題4にある．)　ただし，前回のレポートで提出した決定木の内容があまりにも陳腐である場合は，プログラムだけではなく，決定木も新たに考え直して欲しい．

  (2) これまで学んだことを使って，何か意味のあるプログラムを作れ．
- **提出方法 (1)：決定木を記述した** Microsoft Word **ファイル，及び，** Pascal **のプログラム (ソースファイル) をEメールに添付して送信しなさい．**
- **提出方法 (2)：プログラムの説明が書かれた文書** (MS Word) **，及び，** Pascal **のプログラム (ソースファイル) をEメールに添付して送信しなさい．**
- **Eメールの宛先は，** ic@coconut.sys.eng.shizuoka.ac.jp
- **Eメールの送信アドレスは，大学から配布されたものを用いること．**
- **Eメールの件名には，学籍番号と氏名，及び，選んだ課題 ((1) または (2)) を記載すること．**
- **提出期限は，** 11**月**30**日 (火)** 17:00 **までである．**