情報とコンピュータ

静岡大学工学部
安藤和敏

2004.11.29

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

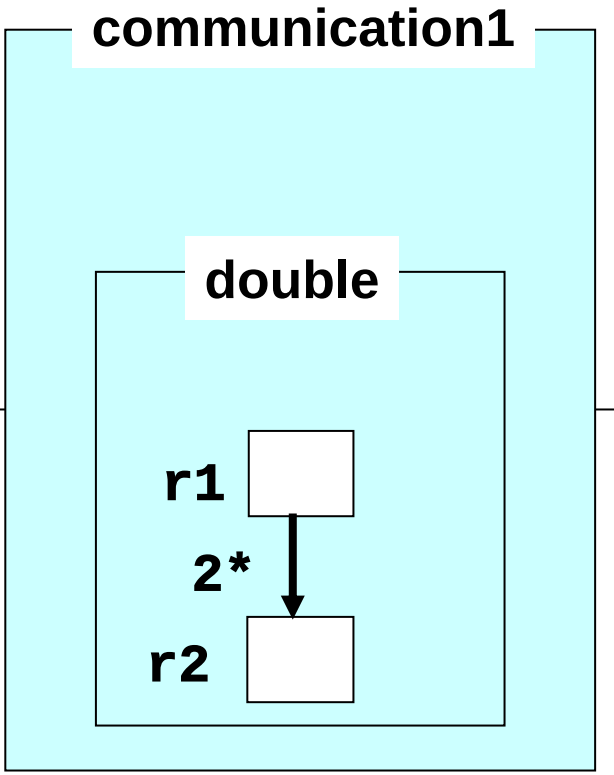
- サブルーチン間通信の例
- データベースのための事実の記憶と出力
- 質問の表示と答えの検索
- データベースプログラムの組み立てと注釈の付加

プログラムCommunication1

```
program communication1(input, output);
procedure double;
  var
    r1, r2: real;
  begin
  r2 := 2 * r1;
  end;
begin
double;
end.
```

プログラムCommunication1

```
program communication1(input, output);
procedure double;
  var
    r1, r2: real;
  begin
  r2 := 2 * r1;
  end;
begin
double;
end.
```

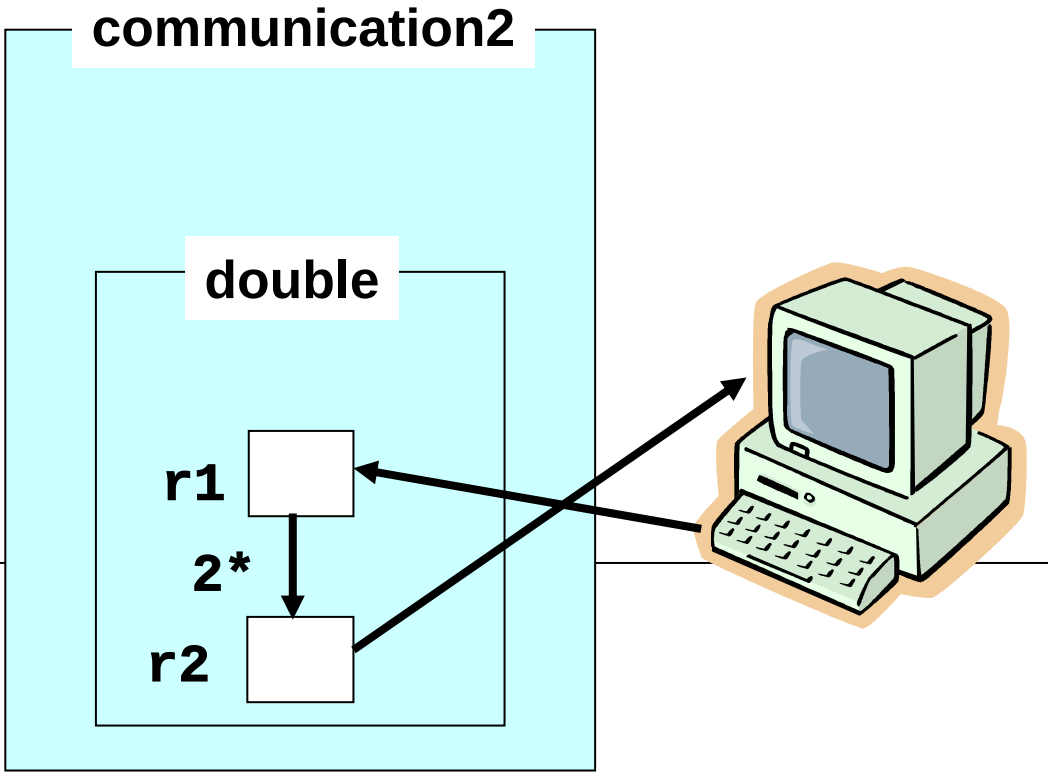


プログラムCommunication2

```
program communication2(input, output);
procedure double;
  var
    r1, r2: real;
  begin
  readLn(r1);
  r2 := 2 * r1;
  writeLn(r2);
  end;
begin
double;
end.
```

プログラムCommunication2

```
program communication2(input, output);
procedure double;
  var
    r1, r2: real;
  begin
  readLn(r1);
  r2 := 2 * r1;
  writeLn(r2);
  end;
begin
double;
end.
```

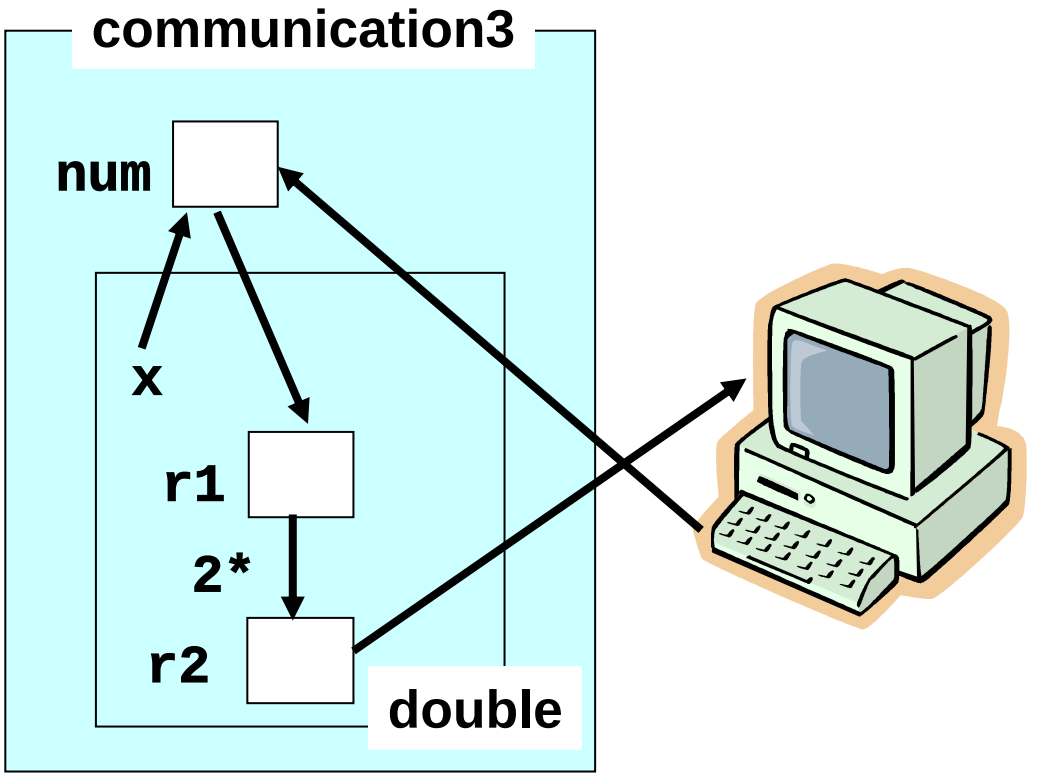


プログラムCommunication3

```
program communication3(input, output);
var
  num: real;
procedure double(var x: real);
  var
    r1, r2: real;
  begin
  r1 := x;
  r2 := 2 * r1;
  writeLn(r2);
  end;
begin
readLn(num);
double(num);
end.
```

プログラムCommunication3

```
program communication3(input, output);
var
  num: real;
procedure double(var x: real);
  var
    r1, r2: real;
  begin
  r1 := x;
  r2 := 2 * r1;
  writeLn(r2);
  end;
begin
readLn(num);
double(num);
end.
```

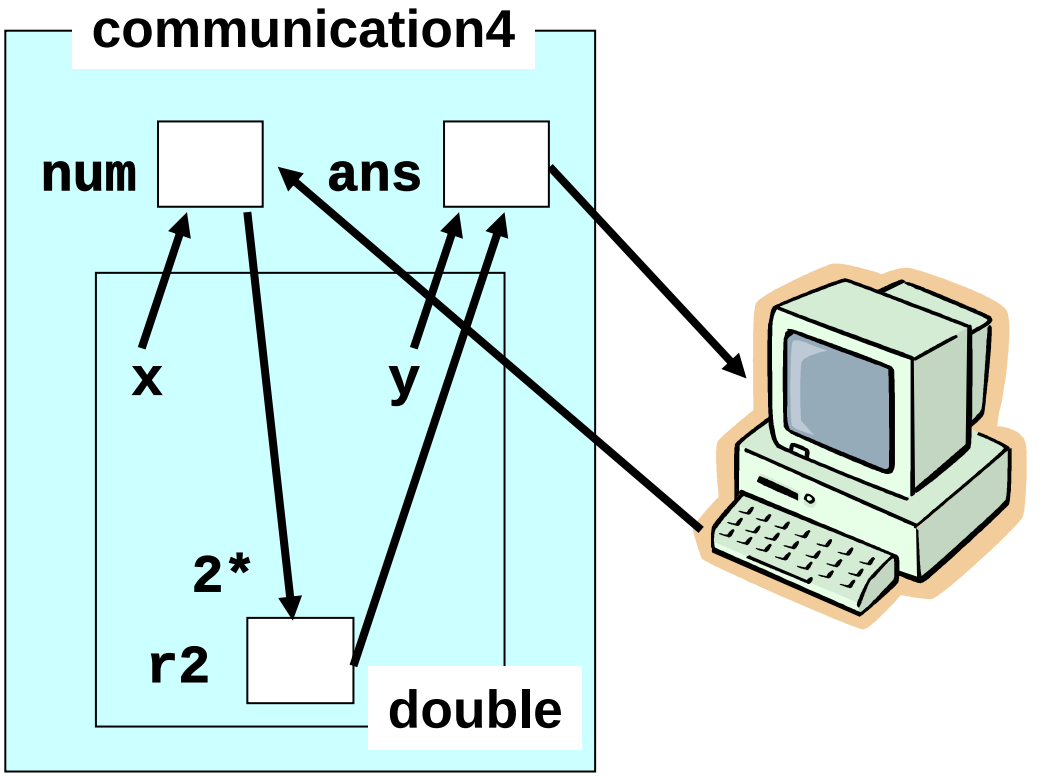


プログラムCommunication4

```
program communication4(input, output);
var
  num, ans: real;
procedure double(var x, y: real);
  var
    r2: real;
  begin
  r2 := 2 * x;
  y  := r2;
  end;
begin
readLn(num);
double(num,ans);
writeLn(ans);
end.
```

プログラムCommunication4

```
program communication4(input, output);
var
  num, ans: real;
procedure double(var x, y: real);
  var
    r2: real;
  begin
  r2 := 2 * x;
  y  := r2;
  end;
begin
readLn(num);
double(num,ans);
writeLn(ans);
end.
```

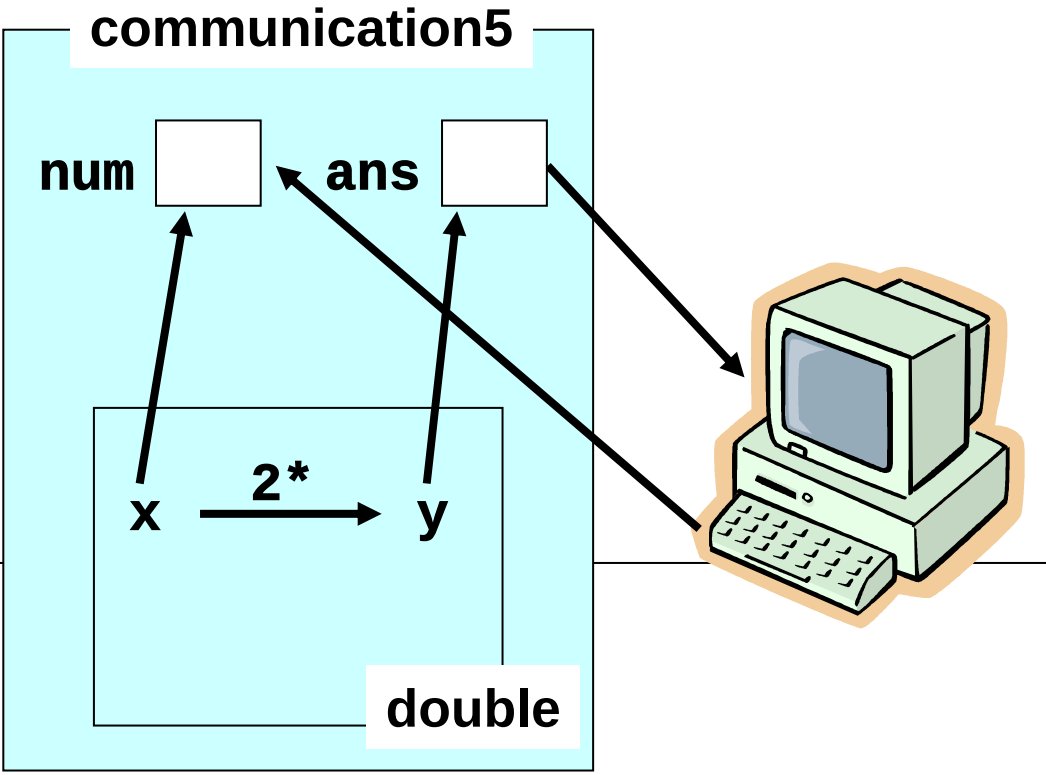


プログラムCommunication5

```
program communication5(input, output);
var
  num, ans: real;
procedure double(var x, y: real);
  begin
  y  := 2 * x;
  end;
begin
readLn(num);
double(num,ans);
writeLn(ans);
end.
```

プログラムCommunication5

```
program communication5(input, output);
var
  num, ans: real;
procedure double(var x, y: real);
  begin
  y  := 2 * x;
  end;
begin
readLn(num);
double(num,ans);
writeLn(ans);
end.
```

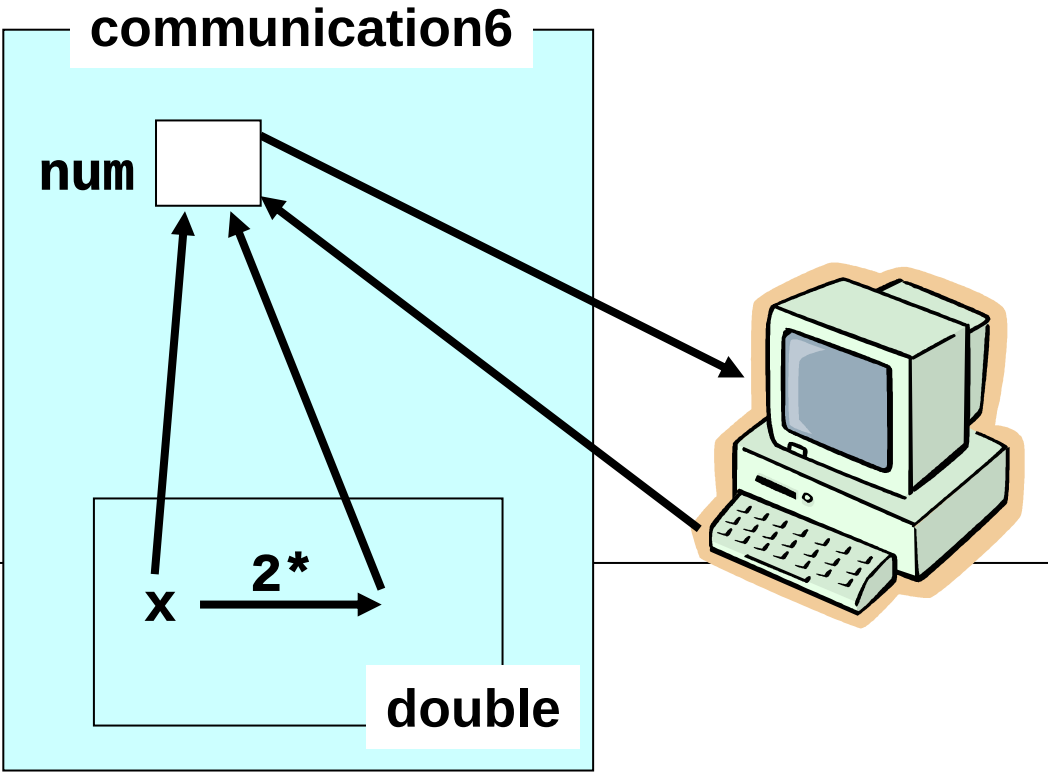


プログラムCommunication6

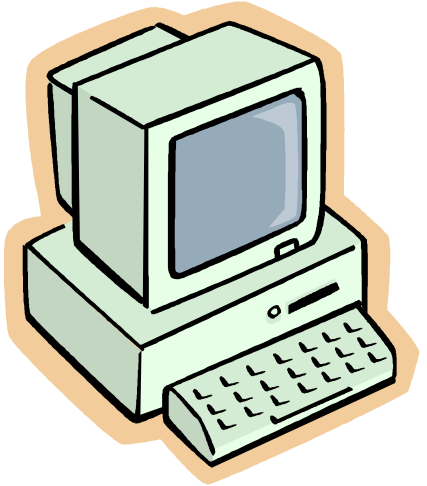
```
program communication6(input, output);
var
  num: real;
procedure double(var x: real);
  begin
  x  := 2 * x;
  end;
begin
readLn(num);
double(num);
writeLn(num);
end.
```

プログラムCommunication6

```
program communication6(input, output);
var
   num: real;
procedure double(var x: real);
   begin
   x  := 2 * x;
   end;
begin
readLn(num);
double(num);
writeLn(num);
end.
```



データベースプログラム



ダンスモア家のコンピュータの中にあったような情報の蓄積は，**データベース**と呼ばれる．

データベースに情報を記憶させ，データベースを検索して，質問に答えるプログラムは，**データベースプログラム**と呼ばれる．

我々は，データベースプログラムを作ろうとしている．

問題を扱いやすくできるように表現する(1)

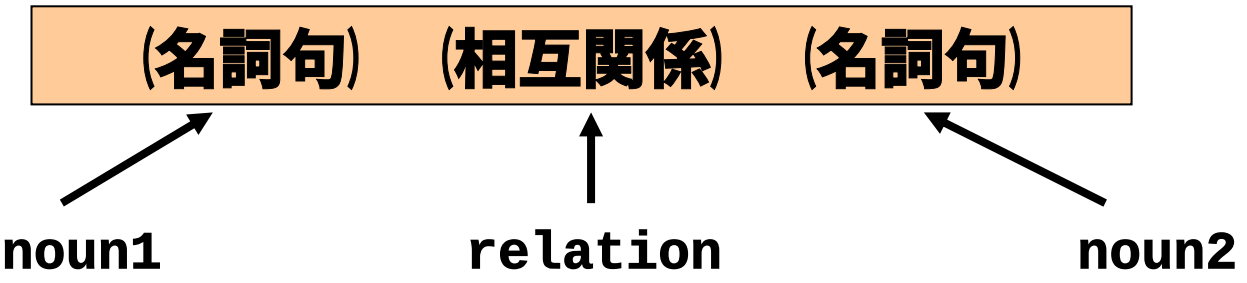
家族の情報は，一連の事実として平叙文で記憶されているとする．

メイソン氏は3時に来た．
(Mr. Maison visited at 3:00 P.M.)

メイソン氏は薬剤師である．
(Mr. Maison is a chemist.)

事実の構造化

(英文の) 事実の大部分は，次の形である．

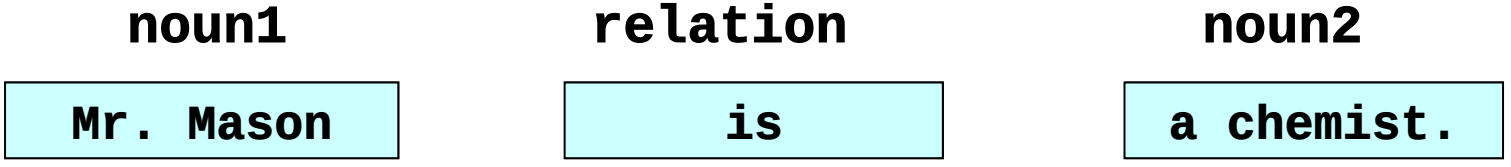


(Mr. Maison)　(visited at)　(3:00 P.M.)

(Mr. Maison)　(is)　(a chemist.)

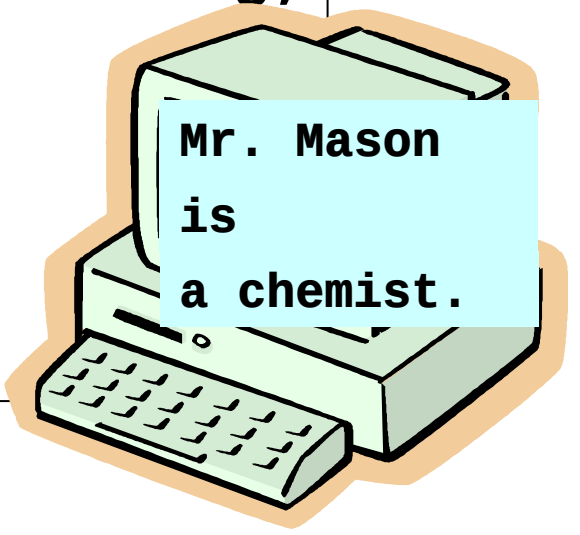
事実の記憶

```
var
  noun1, relation, noun2: string;
begin
noun1    := 'Mr. Mason';
relation := 'is';
noun2    := 'a chemist.';
end.
```



事実の読み込み

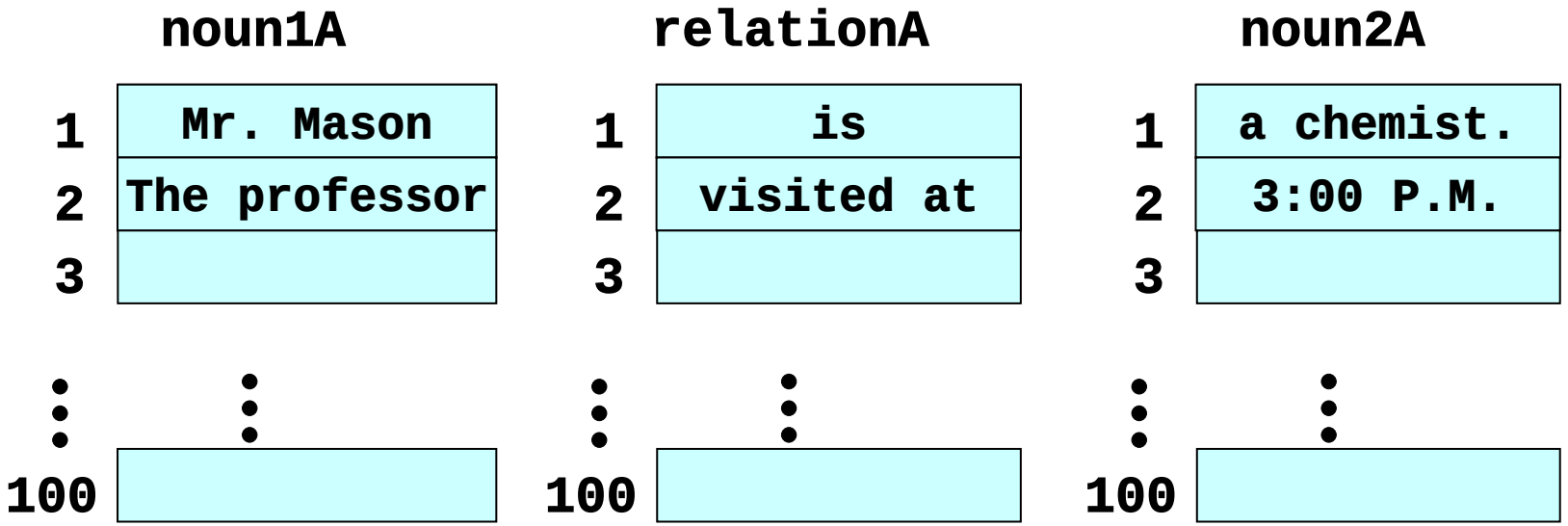
```
var
  noun1, relation, noun2: string;
begin
readLn(noun1);
readLn(relation);
readLn(noun2);
end.
```



noun1	relation	noun2
Mr. Mason	is	a chemist.

事実を記憶するための配列

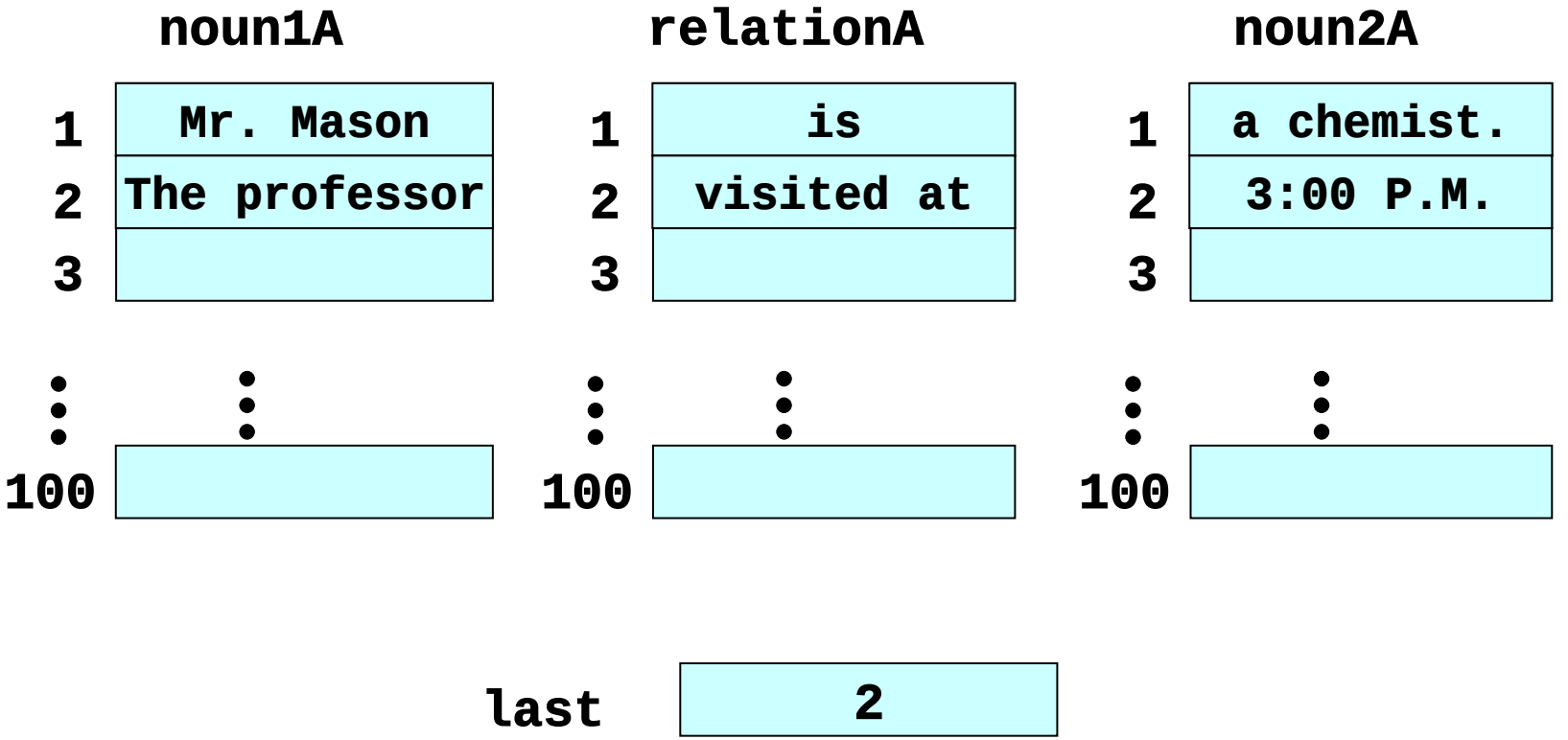
```
type
  stringarray100 = array[1..100] of string;
var
  noun1A, relationA, noun2A: stringarray100;
```



stringarray100 の型定義がテキストと異なることに注意！

last（最後の事実の場所を表す）

```
var
  last: integer;
```



プログラム Database1

```
program Database1(input, output);
type
  stringarray100 = array[1..100] of string;
var
  command: string;
  noun1A, relationA, noun2A: stringarray100;
  last: integer;

procedure help;
 (省略)
procedure InputFact;
(省略)
procedure PrintFacts;
 (省略)
procedure Find;
 (省略)
```

プログラム Database1

```
begin
noun1A[1] := 'Lord Dunsmore'; relationA[1] := 'is married to';
    noun2A[1]    := 'Lady Dunsmore.';
noun1A[2] := 'The gardener'; relationA[2] := 'is married to';
    noun2A[2]    := 'the maid.';
last := 2;
command := 'start';
while command <> 'q' do
  begin
  write('Command:');
  readLn(command);
  if command = 'f'    then Find;
  if command = 'help' then help;
  if command = 'i'    then InputFact;
  if command = 'p'    then PrintFacts;
  end;
writeLn('終了しました. ');
end.
```

入力ルーチン (InputFact)

入力

入力ルーチン:
メモリ内に新しい記憶場所を設定する.
新しい事実を読み込んで, そこに記憶する.

サブルーチンInputFact(1)

```
procedure InputFact;
  begin
  writeLn('Input a fact. Type three fields
on sequential lines.');
  last := last + 1;
  readLn(noun1A[last]);
  readLn(relationA[last]);
  readLn(noun2A[last]);
  end;
```

サブルーチンInputFact(2)

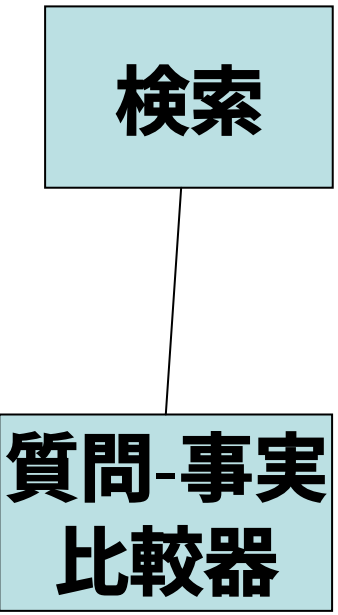
```
procedure InputFact(var n1,r,n2:stringarray100;
                    var last:integer);
  begin
  writeLn('Input a fact. Type three fields on
sequential lines.');
  last := last + 1;
  readLn(n1[last]);
  readLn(r[last]);
  readLn(n2[last]);
  end;
```

出力ルーチン (PrintFacts)

出力

出力ルーチン:
事実が入っている各記憶場所ごとに,
その内容を出力する.

検索ルーチン (Find)



検索ルーチン:

ユーザの質問を読む.

事実が入っている各記憶場所調べて,

質問-事実比較器を呼び出す.

比較器が「answer」と報告すれば,

その事実を出力する.

サブルーチンFind

```
procedure Find(var n1, r, n2: stringarray100;
               var last: integer);
 var
  noun1, relation, noun2, result: string;
  i: integer;
 begin
 writeLn('Give the query. ‘);
 writeLn('Type three fields on sequential lines.');
 readLn(noun1);readLn(relation);readLn(noun2);
 writeLn('THE RELATED FACTS:');
 i := 1;
 while i<=last do
  begin
  QFCompare(noun1,relation,noun2,n1[i],r[i],n2[i],result);
  if result = 'answer' then writeLn(n1[i],r[i],n2[i]);
  i := i+1;
  end;
 end;
```

サブルーチンQFCompare(1)

```
procedure QFCompare(var Qnoun1, Qrel, Qnoun2,
                        Fnoun1, Frel, Fnoun2,
                                          an: string);
  begin
  if (Qnoun1 = Fnoun1) and (Qrel = Frel)
    and (Qnoun2 = Fnoun2) then
    begin
    an := 'answer';
    end
  else
    begin
    an := 'no';
    end;
  end;
```

サブルーチンQFCompare(2)

```
procedure QFCompare(var Qnoun1, Qrel, Qnoun2,
                        Fnoun1, Frel, Fnoun2,
                                          an: string);
  begin
  if ((Qnoun1 = '?') or (Qnoun1 = Fnoun1))
     and (Qrel = Frel)
     and (Qnoun2 = Fnoun2) then
    begin
    an := 'answer';
    end
  else
    begin
    an := 'no';
    end;
  end;
```

注釈 (コメント) の付加

コメント：{　} 囲まれた部分は，コメント文 (注釈文) と呼ばれ，コンパイラはコメントを完全に無視する．

コメントの目的：後でプログラムを読んだり変更する人のため．

コメントには以下の3種類がある．

1. プログラムヘッダ．
2. コードブロックヘッダ．
3. ラインコメント．

関係データベース

この章で作ったデータベースは**関係データベース**と呼ばれるものの一種である.

我々のデータベースにおける各事実は, 3つのフィールドを持つが, 一般のデータベースはもっと多くの (数十の) フィールドを持つ.

我々のデータベースには**推論**の機能はない.

(Jill) (is a sister of) (Nancy.)

(Nancy) (is a sister of) (Barbara.)

データベースの推論機能

我々のデータベースには推論の機能はない.

(Jill) (is a sister of) (Nancy.)

(Nancy) (is a sister of) (Barbara.)

という2つの事実がデータベースに入っているならば,

(?) (is a sister of) (Barbara.)

という質問には, Jill **と** Nancy **と言う答えを期待したいところであるが, そう答えてくれない.**

商用のデータベースシステムではこのような推論機能が実現されている.

今日はおしまい

p. 167, 174の練習問題あるいは，この章のその他の練習問題を自力でやってみることは，この章の話を理解できたかどうかの確認になるであろう．