情報とコンピュータ

静岡大学工学部
安藤和敏

2004.11.08

3章　数値計算と関数の学習

- 数値計算をしてみよう
- 単純な計算
- 関数
- ループの作成と関数の学習
- 最適値の探索
- 情報を配列に格納する
- 総和，極小，極大を求める
- プログラミングのパターン

前回に考えた問題

20歳の青年が60歳までに1億円貯めるには，毎月いくら貯金しなければならないか？

預金残高計算のアルゴリズム

1. **指定された毎月の積立額** (payment) **を読み込む.**
2. savings = 0, monthint = 0.01 (**年率**12%) **と設定する. .**
3. **480ヶ月の各月の残高を**

savings := savings + (savings*monthint) + payment;

で計算する.

4. 40年後の残高を出力する.

４０年後の預金計算のプログラム

```
program Savings40Years(input, output);
var
  payment, savings, monthint, month: real;
begin
writeLn('毎月いくらずつ積み立てますか？(単位：万円) ');
readLn(payment);
savings := 0; monthint := 0.001; month :=1;
while month <= 480 do
  begin
  savings := savings + (savings*monthint) + payment;
  month := month +1;
  end;
writeLn('40年後の預金残高は, ',savings:10:2,'万円です. ');
end.
```

配列

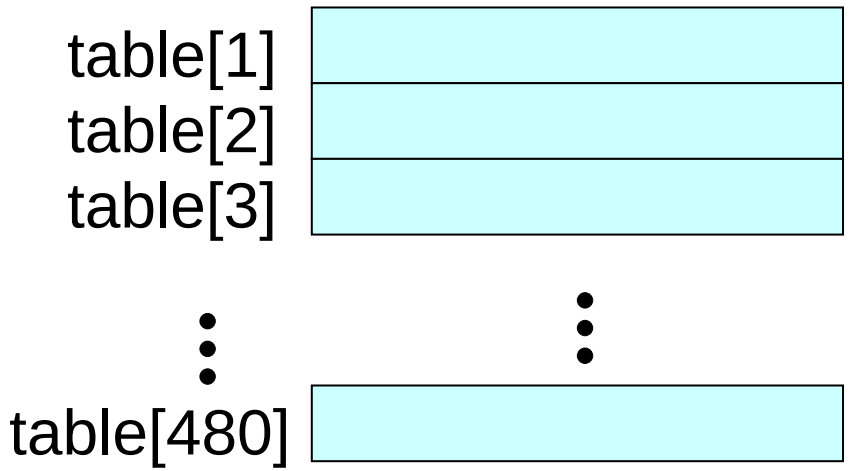
- 月 (1〜480) を指定したときに，その月末の預金残高が分かるようなプログラムを作りたい．
- ４８０ヶ月の全ての預金残高をコンピュータのメモリに記憶できればよい．
- ４８０個の変数を用意する便利な方法がある．

配列の宣言

例:

```
type
  realarray480 = array[1..480] of real;
var
  table: realarray480;
```

メモリ上に, table[1], table[2], …, table[480] **という名前の付いた記憶場所が確保される. この記憶場所には, 実数型のデータが記憶される.**



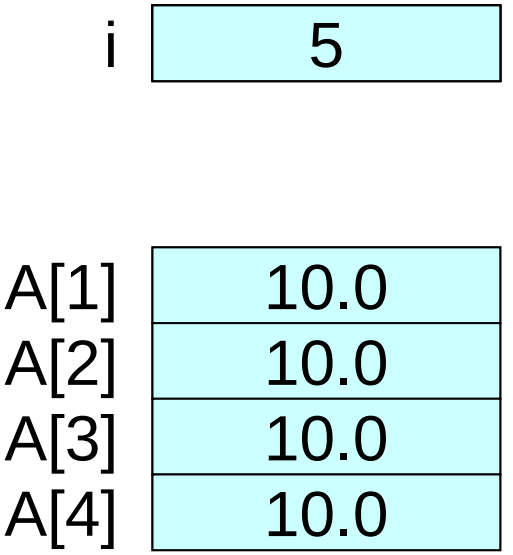
プログラム FirstArray

```
program FirstArray(input, output);
type
  realarray4  = array[1..4] of real;
var
  A: realarray4;
begin
A[1] := 10;
A[2] := 10;
A[3] := 10;
A[4] := 10;
end.
```

A[1]	10.0
A[2]	10.0
A[3]	10.0
A[4]	10.0

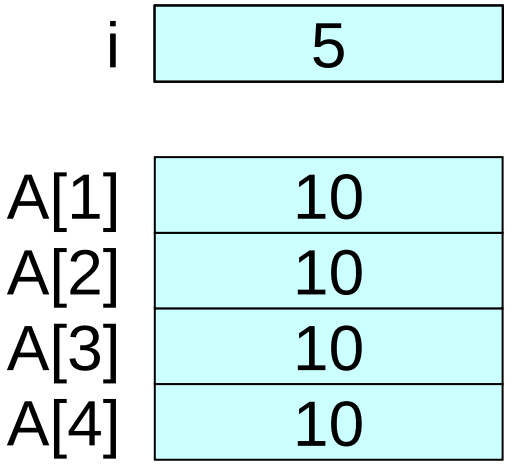
プログラム FirstArray’

```
program FirstArray(input, output);
type
  realarray4  = array[1..4] of real;
var
  A: realarray4;
  i : integer;
begin
i := 1;
while i <= 4 do
  begin
  A[i] := 10;
  i := i + 1;
  end;
end.
```



配列要素の画面出力

```
i := 1;
while i <= 4 do
  begin
  writeLn(A[i]);
  i := i + 1;
  end;
end.
```




```
10.0
10.0
10.0
10.0
```

プログラム SecondArray

```
program SecondArray(input, output);
type
  realarray4  = array[1..4] of real;
var
  A: realarray4;
  i : integer;
begin
i := 1;
while i <= 4 do
  begin
  A[i] := i + 10;
  i := i + 1;
  end;
```

以下省略

i	5

A[1]	11.0
A[2]	12.0
A[3]	13.0
A[4]	14.0

プログラム FillSavingTable

```
program FillSavingsTable(input, output);
type
  realarray480: array[1..480] of real;
var
  table: realarray480;
  payment, savings, monthint: real;
  month: integer;
```

（次のページへ続く）

プログラム FillSavingTable（続き）

```
（前ページからの続き）
begin
writeLn('毎月いくらずつ積み立てますか？(単位：万円) ');
readLn(payment);
savings := 0; monthint := 0.001; month :=1;
while month <= 480 do
  begin
  savings := savings + (savings*monthint) + payment;
  table[month] := savings;
  month := month +1;
  end;
end.
```

次のような動作をするプログラムを作りたい

コンピュータ	毎月いくらずつ積み立てますか？(単位:万円)
ユーザ	0.5
コンピュータ	40年分の貯金計画が準備できました.
コンピュータ	何ヶ月目の積立額を知りたいですか？
ユーザ	60
コンピュータ	40.83
コンピュータ	何ヶ月目の積立額を知りたいですか？
ユーザ	100
コンピュータ	40.83
コンピュータ	何ヶ月目の積立額を知りたいですか？
ユーザ	0
コンピュータ	これでプログラムは終了です.

質問ルーチンのアルゴリズム

1. 「何ヶ月目の積立額を知りたいですか？」と質問する.
2. 知りたい月 (month) を受け取る.
3. その月が0より大きい限りは次の動作を繰り返す.

- その月の預金残高を出力する.
- 「何ヶ月目の積立額を知りたいですか？」と質問する.
- 知りたい月 (month) を受け取る.

プログラム SavingsTable (1)

```
program SavingsTable(input, output);
type
  realarray480 = array[1..480] of real;
var
  table: realarray480;
  payment, savings, monthint: real;
  month: integer;
```

プログラム SavingsTable (2)

```
begin
writeLn('毎月いくらずつ積み立てますか？(単位：万円) ');
readLn(payment);
savings := 0;
monthint := 0.01;
month :=1;
while month <= 480 do
  begin
  savings := savings + (savings*monthint) + payment;
  table[month] := savings;
  month := month +1;
  end;
writeLn('40年分の貯金計画が準備できました. ');
```

プログラム SavingsTable (3)

```
writeLn('何ヶ月目の積立額を知りたいですか？');
readLn(month);
while month > 0 do
  begin
    writeLn(table[month]:10:2);
    writeLn('何ヶ月目の積立額を知りたいですか？');
    readLn(month);
  end;

writeLn('これでプログラムは終了です．');
end.
```

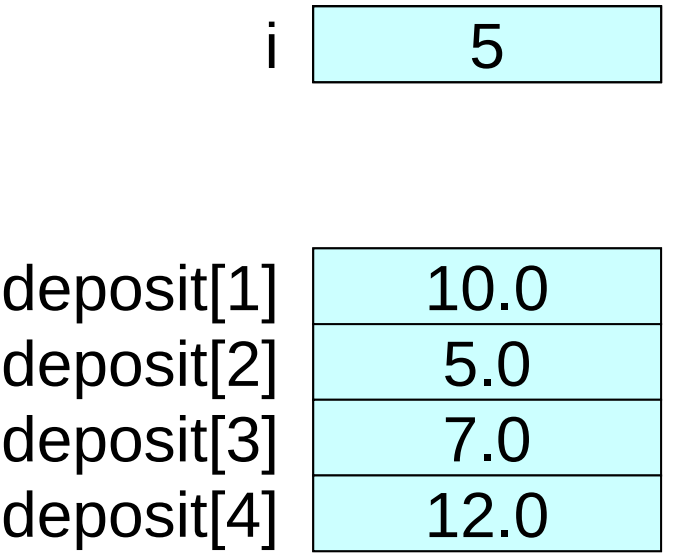
総和，極小，極大を求める

ある人が毎週1回ずつ，1ヶ月に4回預金するとして，利息分を無視する場合，総預金残高はいくらになるか？また，預金時の最小金額と最大金額はいくらか？

この質問に答えるために，一般的な累算方法（この場合は総和）と極値的要素（最大値と最小値）を求める方法を調べる．

プログラム Deposit

```
program Deposit(input, output);
type
  realarray4  = array[1..4] of real;
var
  deposit: realarray4;
  i : integer;
begin
i := 1;
while i <= 4 do
  begin
  writeLn('預金額はいくらですか？');
  readLn(deposit[i]);
  i := i + 1;
  end;
end.
```



deposit[1]〜deposit[4]の総和を求める方法

sum := 0;
deposit[1] を sum に加える.
deposit[2] を sum に加える.
deposit[3] を sum に加える.
deposit[4] を sum に加える.

deposit[1]	10.0
deposit[2]	5.0
deposit[3]	7.0
deposit[4]	12.0

deposit[1]を sum に加えるには,

sum := sum + deposit[1];

とする.

deposit[1]〜deposit[4]の総和を求める方法

```
sum := 0;
sum := sum + deposit[1];
sum := sum + deposit[2];
sum := sum + deposit[3];
sum := sum + deposit[4];
```

deposit[1]	10.0
deposit[2]	5.0
deposit[3]	7.0
deposit[4]	12.0

sum	34.0

deposit[1]〜deposit[4]の総和を求める方法

```
sum := 0;
i := 1;
while i <= 4 do
  begin
  sum := sum + deposit[i];
  i := i + 1;
  end;
```

deposit[1]	10.0
deposit[2]	5.0
deposit[3]	7.0
deposit[4]	12.0

sum	34.0

i	5

累算の一般的なアルゴリズム

1. アキュムレータに初期値を入れる.
2. インデックスに初期値を与える.
3. 操作対象が他にもある場合は以下を繰り返す.
- アキュムレータ := アキュムレータ　操作　操作対象
- インデックスの値を増やす.

プログラム SumN

```
program SumN(input, output);
var
  i, n, sum: real;
begin
readln(n);
sum := 0;
i :=1;
while i <= n do
  begin
  sum := sum + i;
  i := i + 1;
  end;
writeLn('合計は, ',sum:8:2,'です. ');
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.
```

プログラム Asequence

```
program Asequence(input, output);
var
  i, n: integer;
  Asequence: string;
begin
readln(n);
Asequence := '';
i := 1;
while i <= n do
  begin
  Asequence := Asequence & 'A';
  i := i + 1;
  end;
writeLn('Aの列は, ',Asequence,'です. ');
end.
```

deposit[1]〜deposit[4]の最大値を求めるアルゴリズム

1. **現在までの最大値** := **最初の要素の値.**
2. i := 2;.
3. i <= 4 **である限り以下を繰り返す.**

- **もし**deposit[i] > **現在までの最大値ならば,**
  現在までの極値 := deposit[i]

とする.

- i := i + 1.

deposit[1]〜deposit[4]の最大値を求める方法

```
largestsofar := deposit[1];
i := 2;
while i <= 4 do
  begin
  if deposit[i] > largestsofar do
    begin
      largestsofar := deposit[i];
    end;
  i := i + 1;
  end;
```

deposit[1]	10.0
deposit[2]	5.0
deposit[3]	7.0
deposit[4]	12.0

largestsofar	12.0
i	5

極値を求める一般的なアルゴリズム

1. 現在までの極値 := 最初の要素の値.
2. インデックスに初期値を与える.
3. 他にも調べる要素がある場合は以下を繰り返す.
   - その要素の値が, 現在までの極値を越えていたら,
     現在までの極値 := その要素の値

   とする.
   - インデックスの値を増やす.

宿題

- p. 121**の練習問題** 1.
- p. 126**の練習問題** 2, 3, 4.
- **提出しなくても良い.**