

配列

配列, 繰り返し文と配列

本日の内容

例題1. 月の日数

配列とは. 配列の宣言. 配列の添字.

例題2. ベクトルの内積

例題3. 棒グラフを描く

例題4. Horner 法による多項式の計算

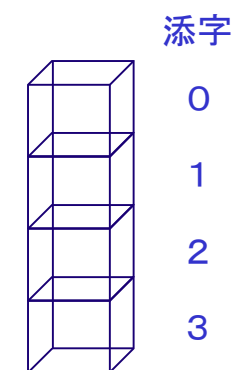
例題5. エラトステネスのふるい

配列と繰り返し計算の関係

今日の到達目標

- 配列とは何かを理解し, integer, real の配列を使ったプログラムを書けるようになる
- 配列と繰り返し文を組み合わせ, 多量のデータを扱えるようになる

配列



- データの並びで, 番号(添字)が付いている

例題1. 月の日数

- 年と月を読み込んで、日数を求めるプログラムを作る
 - うるう年の2月ならば29
 - 日数を求めるために、サイズ12(1から12まで)の integer の配列を使う

例) 2001 年 11 月 → 30

```
program sum;
{$APPTYPE CONSOLE}
uses SysUtils;
var num : array [1..12] of integer;
var y, m : integer;
begin
  num[1] := 31; num[2] := 28; num[3] := 31; num[4] := 30;
  num[5] := 31; num[6] := 30; num[7] := 31; num[8] := 31;
  num[9] := 30; num[10] := 31; num[11] := 30; num[12] := 31;
  write('Please Enter y(year): ');
  readln(y);
  write('Please Enter m(month): ');
  readln(m);
  if (m=2) and (((y mod 400)=0) or (((y mod 100)<>0) and ((y mod 4)=0))) then begin
    writeln('number of days(, y, ', m, ') is 29');
  end
  else begin
    writeln('number of days(, y, ', m, ') is ', num[m]);
  end;
  readln
end.
```

配列への書き込み

配列からの読み出し

月の日数

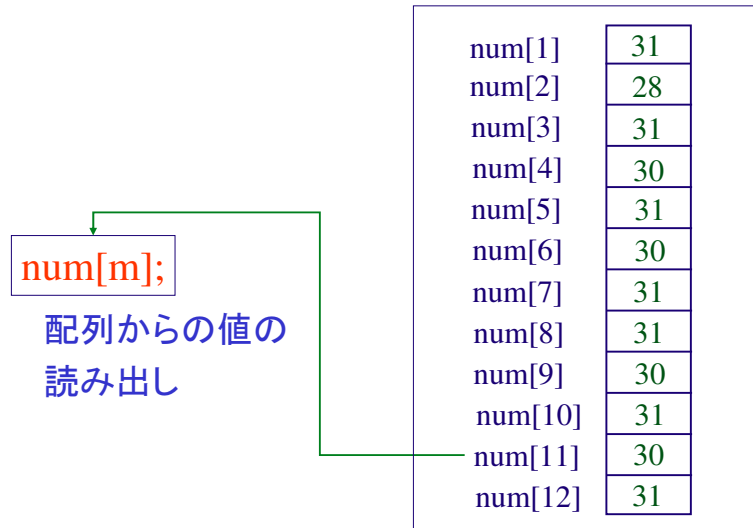
実行結果の例

```
Please Enter y(year): 2002
Please Enter m(month): 6
number of days(2002/6) is 30
```

```
Please Enter y(year): 2002
Please Enter m(month): 6
number of days(2002/6) is 30
```

プログラムとデータ

メモリ



配列の宣言

- 配列には、**名前**と**型**(データの種類のこと)と**サイズ**がある
 - 整数データ **integer**
 - 浮動小数データ **real**
- 配列を使うには、**配列の使用をコンピュータに伝えること(宣言)**が必要

```
var num : array [1..12] of integer;
```

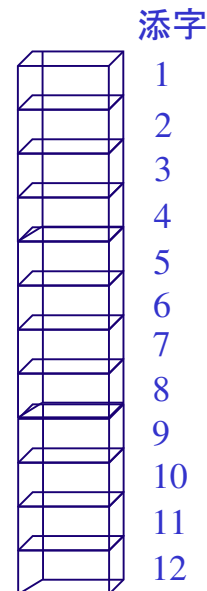
名前が num 配列の添字は1から12 整数データ

array [] of の意味

- [] の中に、添字の範囲を書く
 - おのずと、配列のサイズが決まる

例) 添字は、1から12まで
⇒ array [1..12] of ...

配列の添字



- 配列の中身を読み書きするときには、配列の**名前**と**添字**を書く
例) num[m]
これが添字

配列の読み書き

- 配列の名前と添字を書く

例)

```
num[1] := 31;  
writeln('number of days(' y, '/', m, ') is', num[m]);
```

練習1. 例題1を実行せよ

- ① 例題1のプログラム
- ② Borland Delphi 6 の起動
「スタート」→「プログラム」→「Borland Delphi 6」→「Delphi 6」
- ③ コンソールアプリケーションの新規作成
「ファイル」→「新規作成」→「その他」⇒ ウィンドウが現れる
「コンソールアプリケーション」→「OK」⇒ ウィンドウが現れる
- ④ ①のプログラムコードを「コピー」して「貼り付け」
- ⑤ コンパイル
「プロジェクト」→「〇〇をコンパイル」
- ⑥ 実行
「実行」→「実行」 ☆余裕がある人は練習2に進んでください

例題2. ベクトルの内積

- ベクトル(1.9, 2.8, 3.7)と、ベクトル(4.6, 5.5, 6.4)の内積を表示するプログラムを作る
 - 2つのベクトルの内積の計算のために、**サイズ3の配列を2つ使う**

ベクトルの内積

- 成分による内積

$$\vec{a} = (a_0, a_1, a_2) \quad \vec{b} = (b_0, b_1, b_2) \quad \text{のとき}$$

$$\vec{a} \cdot \vec{b} = a_0 b_0 + a_1 b_1 + a_2 b_2$$

```

program sum;
{$APPTYPE CONSOLE}
uses SysUtils;
var u, v : array [0..2] of real;
var i : integer;
var ip : real;
begin
  u[0] := 1.9; u[1] := 2.8; u[2] := 3.7;
  v[0] := 4.6; v[1] := 5.5; v[2] := 6.4;
  ip := 0;
  for i := 0 to 2 do begin
    ip := ip + u[i] * v[i];
  end;
  writeln('product =', ip:8:3 );
  readln
end.

```

配列への
書き込み

配列からの
読み出し

ベクトルの内積

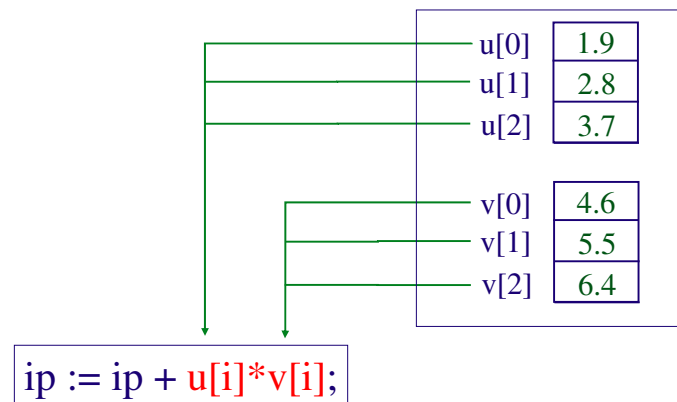
実行結果の例

product = 47.820

product = 47.820

プログラムとデータ

メモリ



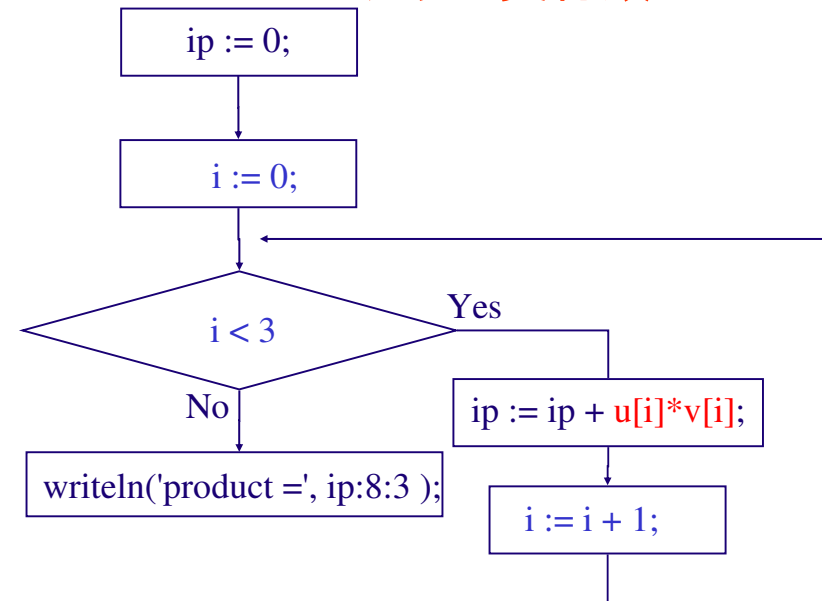
配列からの
読み出し

配列の宣言

```
var u, v : array [0..2] of real;
```

名前は uとv 添字は 0から2まで 浮動小数データ

プログラム実行順



ベクトルの内積

	i の値	繰り返し条件式 が成り立つか	ip の値
繰り返し 1回目	i = 0	i < 3 が成り立つ	ip := ip + u[0] * v[0]; つまり ip の値は u[0]*v[0]
繰り返し 2回目	i = 1	i < 3 が成り立つ	ip := ip + u[1] * v[1]; つまり ip の値は u[0]*v[0] + u[1]*v[1]
繰り返し 3回目	i = 2	i < 3 が成り立つ	ip := ip + u[2] * v[2]; つまり ip の値は u[0]*v[0] + u[1]*v[1] + u[2]*v[2]
繰り返し 4回目	i = 3	i < 3 が成り立たない	

練習2. 例題2を実行せよ

- ① 例題2のプログラム
- ② Borland Delphi 6 の起動
「スタート」→「プログラム」→「Borland Delphi 6」→「Delphi 6」
- ③ コンソールアプリケーションの新規作成
「ファイル」→「新規作成」→「その他」⇒ ウィンドウが現れる
「コンソールアプリケーション」→「OK」⇒ ウィンドウが現れる
- ④ ①のプログラムコードを「コピー」して「貼り付け」
- ⑤ コンパイル
「プロジェクト」→「〇〇をコンパイル」
- ⑥ 実行
「実行」→「実行」 ☆余裕がある人は練習3に進んでください

練習3. 例題2をステップ実行せよ

- ① すでに「練習2」を終えていること
正確には, 練習2の「⑤コンパイル」を終えた状態であること
- ② 「実行」する代わりに「ステップ実行」の「F8」を押す
F8はステップ実行の意味
- ③ F8を押し続けながら, 実行の様子を画面で確認する
- ④ ステップ実行に飽きたら, 「F9」を押すとプログラムの最後の行まで1度に実行される

☆余裕がある人は練習4に進んでください

例題3. 棒グラフを描く

- 整数の配列から, その棒グラフを表示するプログラムを作る.
 - ループの入れ子で, 棒グラフの表示を行う

```
program sum;
{$APPTYPE CONSOLE}
uses SysUtils;
var a : array [1..7] of integer;
var i, j : integer;
begin
  a[1] := 6; a[2] := 4; a[3] := 7; a[4] := 1;
  a[5] := 5; a[6] := 3; a[7] := 2;
  for i := 1 to 7 do begin
    for j := 1 to a[i] do begin
      write('*');
    end;
    writeln;
  end;
  readln
end.
```

配列への
書き込み

配列からの
読み出し

棒グラフを描く

実行結果の例

```
*****
****
*****
*
*****
***
**
```

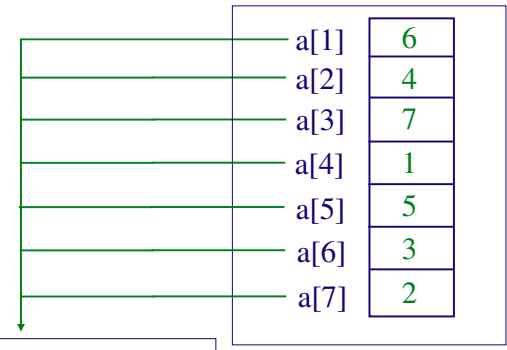
```

*****
****
*****
*
****
**
**

```

プログラムとデータ

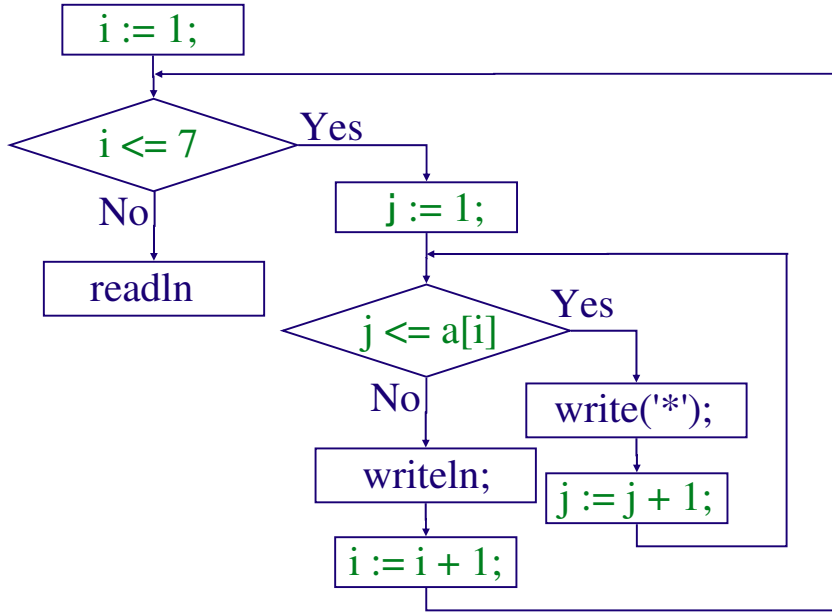
メモリ



```
for j := 1 to a[i] do begin
```

配列からの
読み出し

プログラム実行順



練習4. 例題3を実行せよ

- ① 例題3のプログラム
- ② Borland Delphi 6 の起動
「スタート」→「プログラム」→「Borland Delphi 6」→「Delphi 6」
- ③ コンソールアプリケーションの新規作成
「ファイル」→「新規作成」→「その他」⇒ ウィンドウが現れる
「コンソールアプリケーション」→「OK」⇒ ウィンドウが現れる
- ④ ①のプログラムコードを「コピー」して「貼り付け」
- ⑤ コンパイル
「プロジェクト」→「〇〇をコンパイル」
- ⑥ 実行
「実行」→「実行」 ☆余裕がある人は練習5に進んでください

例題4. Horner 法による多項式の計算

• n次の多項式

$$f(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + \dots + a_n \cdot x^n$$

について、次数 n と、係数 a_0 から a_n を読み込んで、 $f(x)$ を計算するプログラムを作る

- まず n を読み込んで、その後に a_0 から a_n を読み込む。最後に x を読み込む。
- 次ページで説明する Horner法を使う
- 読み込んだ係数は、いったん配列に格納する。 n は高々20とする。

Horner法

$$\begin{aligned} f(x) &= a_0 + a_1 \cdot x + a_2 \cdot x^2 + \dots + a_n \cdot x^n \\ &= a_0 + (a_1 + (a_2 + \dots + (a_{n-1} + a_n \cdot x) \cdot x \dots) \cdot x) \cdot x \end{aligned}$$

例えば, $5 + 6x + 3x^2$
 $= 5 + (6 + 3x) \cdot x$

計算手順

- ① a_n
- ② $a_{n-1} + a_n \cdot x$
- ③ $a_{n-2} + (a_{n-1} + a_n \cdot x) \cdot x$
- ⋯ (a_0 まで続ける)

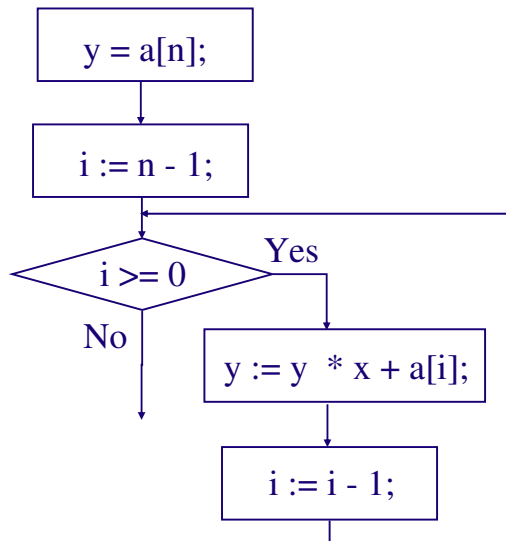
```
program sum;
{$APPTYPE CONSOLE}
uses SysUtils;
var a : array [0..20] of real;
var i, n : integer;
var x, y : real;
begin
  write('Please Enter n: ');
  readln(n);
  for i := 0 to n do begin
    write('Please Enter a[', i, ']: ');
    readln(a[i]);
  end;
  write('Please Enter x: ');
  readln(x);
  y := a[n];
  i := n - 1;
  while i >= 0 do begin
    y := y * x + a[i];
    i := i - 1;
  end;
  writeln('y =', y:8:3);
  readln
end.
```

配列への
書き込み

配列からの
読み出し

```
Please Enter n: 2
Please Enter a[0] : 5
Please Enter a[1] : 6
Please Enter a[2] : 3
Please Enter x: 3
y = 50.000
```

Horner 法



練習5. 例題4を実行せよ

- ① 例題4のプログラム
- ② Borland Delphi 6 の起動
「スタート」→「プログラム」→「Borland Delphi 6」→「Delphi 6」
- ③ コンソールアプリケーションの新規作成
「ファイル」→「新規作成」→「その他」⇒ ウィンドウが現れる
「コンソールアプリケーション」→「OK」⇒ ウィンドウが現れる
- ④ ①のプログラムコードを「コピー」して「貼り付け」
- ⑤ コンパイル
「プロジェクト」→「〇〇をコンパイル」
- ⑥ 実行
「実行」→「実行」 ☆余裕がある人は練習6に進んでください

例題5. エラトステネスのふるい

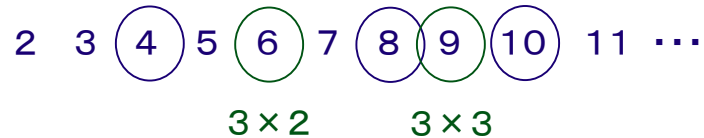
- 「エラトステネスのふるい」の原理に基づいて100以下の素数を求め、結果を表示するプログラムを作成する
 - 100以下の素数を求めるために、2から100までの配列を使う
 - 配列には、添字が素数なら1、そうでなければ0をセットする

エラトステネスのふるい (1/4)

2 3 4 5 6 7 8 9 10 11 ...
 2×2 2×3 2×4 2×5

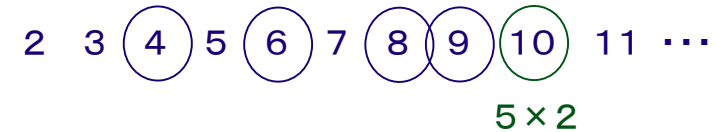
まず、2の倍数を消す

エラトステネスのふるい (2/4)



次に、3の倍数を消す

エラトステネスのふるい (3/4)



次に、5の倍数を消す
(「4の倍数」は考えない。
それは、「4」がすでに消えているから)

エラトステネスのふるい (4/4)



以上のように、2, 3, 5, 7の倍数を消す。

10(これは100の平方根)を超えたら、この操作を止める
(100以下で、11, 13...の倍数はすでに消えている)

```
program sum;
{$APPTYPE CONSOLE}
uses SysUtils;
var p : array [2..100] of integer;
var n, i : integer;
begin
  for i := 2 to 100 do begin
    p[i] := 1;
  end;
  n := 2;
  while n <= sqrt(100) do begin
    i := 2;
    while (n * i) <= 100 do begin
      p[n * i] := 0;
      i := i + 1;
    end;
    n := n + 1;
    while (p[n] = 0) and (n <= sqrt(100)) do begin
      n := n + 1;
    end;
  end;
  for i := 2 to 100 do begin
    if p[i] = 1 then begin
      write(i, ' ');
    end;
  end;
  readln;
end.
```

a[2] から a[100] までを
「1」にセット

まず「2」の倍数から
n が 10 を超えたら終わり

n の倍数を「消す」
(n が 100 以上になったら終わり)

n の「次」の素数を
探す

求めた素数の表示

2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97,

練習6. 例題5を実行せよ

- ① 例題5のプログラム
- ② Borland Delphi 6 の起動
「スタート」→「プログラム」→「Borland Delphi 6」→「Delphi 6」
- ③ コンソールアプリケーションの新規作成
「ファイル」→「新規作成」→「その他」⇒ ウィンドウが現れる
「コンソールアプリケーション」→「OK」 ⇒ ウィンドウが現れる
- ④ ①のプログラムコードを「コピー」して「貼り付け」
- ⑤ コンパイル
「プロジェクト」→「〇〇をコンパイル」
- ⑥ 実行
「実行」→「実行」 ☆余裕がある人は「課題」に進んでください

課題1. 1000 までの素数

- エラトステネスのふるいを使って1000までの数の素数を求めるプログラムを作りなさい
 - 例題5のプログラムを参考にしなさい
 - 1000よりも小さくて、1000に最も近い3つの素数の値を報告せよ
 - 必ず、動作確認まで行うこと

課題2

- 2つのベクトル(u_0, u_1, u_2)と、ベクトル(v_0, v_1, v_2)を読み込んで、内積を表示するプログラムを作りなさい
 - 2つのベクトルの内積の計算のために、サイズ3の配列を2つ使うこと