

# 2025年度前期 情報処理演習 第9回

2025.6.18



名古屋大学減災連携研究センター  
Disaster Mitigation Research Center, NAGOYA UNIVERSITY

平山 修久

1

## Tips

### 前回の課題

#### ー DOループの制御変数は整数型：Integer

調べる除数Mの最大値MNの計算は、

```
MN = INT(SQRT(REAL(N)))
```

#### ー フラグの使い方

- ✓ 素数であると仮定：FP=0
- ✓ 割り切れる＝素数でない，FP=1とする
- ✓ 割る数字をすべて調べたのち（DOループ後）FPが0のままであれば，素数，FPが1であれば素数でない。

# 素数の計算プログラム例

```
program kadai0604
  implicit none
  integer:: n, m, p, mn, np, pmax, flag

  print *, "Max N?"
  read *, p

  pmax=3
  np=2
  Do n = 5, p, 2
    flag=0
    mn = int(sqrt(real(n)))
    Do m = 3, mn
      if (mod(n, m) == 0) flag=1
    end do
    if (flag == 0) then
      np=np+1
      pmax=n
    end if
  end do
  print *, "Number of Prime Number : ",np, " (" ,p,")"
  print *, "Maximun Prime Number : ", pmax
end program
```

3

## 本日の目標

2025.6.18

- ・ 配列を理解する
- ・ 一様乱数を扱える

4

# 配列

## 一次元配列

### 一 配列の宣言

- ✓ 型宣言文でDIMENSION属性を指定
- ✓ 変数名に直接、配列の形状を指定する。

```
型, DIMENSION ( [下限 :] 上限 ) :: 配列名のリスト
型 :: 配列名 ( [下限 :] 上限 ), ...
INTEGER, DIMENSION (1 : 100) :: number, mark
REAL :: x, a( -100 : 100 )
```

- ✓ 下限が省略された場合は1とみなされる。
- ✓ 要素の個数 = 上限値 - 下限値 + 1

# 配列要素

## 要素ひとつひとつは変数と同等

### 一 宣言された配列の1つ1つの要素は単純な変数と同等。

- ✓ 配列要素は、配列名に添え字の値を()でくくったものを付けて表す。

```
INTEGER, DIMENSION (1 : 100) :: number, mark
REAL :: x, a( -100 : 100 )
mark(1)
number(i)
a(-40)
```

### 一 1次元配列の初期値設定には配列定数やDO形反復。

```
(/ 定数, 定数, ... /) [ 定数, 定数, ...]
(/ (データのリスト, DO変数 = 開始値, 限界値) /)
INTEGER:: a( 0 : 9 ) = (/ 0,2,4,6,8,10,12,14,16,18 /)
INTEGER:: a( 0 : 9 ) = (/ (n, n=0,18,2) /)
INTEGER:: a( 0 : 9 ) = (/ (2*n, n = 0, 9 ) /)
```

# 配列の演算

## 部分配列

### 一 部分配列

- ✓ 実行文の中で、配列の一部を、宣言なしで配列として用いることができる。

```
INTEGER :: a( 1 : 5 ) = ( / 2, 4, 6, 8, 10 / )
a( 4 : ) は a( 4 : 5 )と同じで 8,10
a( : 3 ) は a( 1 : 3 )と同じで 2,4,6
a( : ) は a( 1 : 5 )と同じ
a( 1 : 5 : 2 ) は奇数番目の要素からなる配列 2,6,10
a( 5 : 1 : -1 ) は要素を逆順 10,8,6,4,2
a( 3 : 3 ) は要素 a(3)ではなく、要素数1の配列( / 6 / )
```

# 配列

## 添え字

### 一 配列名（添え字配列）

```
INTEGER :: a( -2 : 2 ) = ( / 1,2,3,4,5 / ), &
               b( 1 : 2 ) = ( / 0, 2 / )
a(b)
( / 3, 5 / )
a( ( / 0, 2 / ) )とも書くことができる。
```

# 配列

## 代入文

- 一 配列要素に、右辺に書かれた同じ形状の配列の配列要素を一括して代入することができる。

```
INTEGER :: a( 1 : 10 ), b( 1 : 5 ), c( 1 : 16 ), n
a = ( / 0,1,2,3,4,5,6,7,8,9 / )
a = ( / (n, n = 0, 9 ) / )
b = a( 1 : 5 )
c( 1 : 10 ) = a
c( 12 : 16 ) = a( 6 : 10 )
c( 11 ) = 1
b = 0
c( 1 : 16 : 2 ) = 1
c( 2 : 16 : 2 ) = 0
```

# 配列

## 演算式

- 一 1つ1つの配列要素は、各種の演算において普通の変数と同等に扱ってよい。配列名のままで要素間の演算を一括して書くこともできる。要素の対応は添え字ではなく各配列内での相対位置である。

```
INTEGER, DIMENSION( 1 : 50 ) :: wa, sa, seki, sho, &
    even = ( / (2*i, i = 0, 49 ) / ), &
    odd = ( / (2*i+1, i = 0, 49 ) / )
wa = even + odd
sa = even - odd
seki = even * odd
syo = even / odd
wa( : ) = even( : ) + odd( : )
```

# ベクトル演算

## 配列を用いた

```
INTEGER :: a( 0 : 9 ) = ( / 0,1,2,3,4,5,6,7,8,9 / )
```

のとき

```
a( 1 : 9 ) = a( 0 : 8 )
```

と

```
DO i = 1, 9
  a(i) = a(i-1)
END DO
```

は異なる。

DOループで実現するには、別の**一時的な待避配列b**を用意

```
INTEGER :: a( 0 : 9 ) = ( / 0,1,2,3,4,5,6,7,8,9 / ), &
  b( 0 : 9 )
b = a
DO i = 1, 9
  a(i) = b(i-1)
END DO
```

# 配列による計算例

```
[MBPro13-2016:0617 nhirayama$ gfortran sample.f90
[MBPro13-2016:0617 nhirayama$ ./a.out
```

	0	1	2	3	4	5
6	7	8	9			
0	0					
1	0					
2	1					
3	2					
4	3					
5	4					
6	5					
7	6					
8	7					
9	8					

```
program sample6_1
  implicit none
  integer :: a(0:9) = (/0,1,2,3,4,5,6,7,8,9/), i
  print *,a
  a(1:9)=a(0:8)
  do i=1, 10
    print *,i-1,a(i-1)
  end do
end program sample6_1
```

```
[MBPro13-2016:0617 nhirayama$ gfortran sample6_2.f90
[MBPro13-2016:0617 nhirayama$ ./a.out
```

	0	1	2	3	4	5
6	7	8	9			
0	0					
1	0					
2	0					
3	0					
4	0					
5	0					
6	0					
7	0					
8	0					
9	0					

```
program sample6_2
  implicit none
  integer :: a(0:9) = (/0,1,2,3,4,5,6,7,8,9/), i
  print *,a
  do i=1,9
    a(i)=a(i-1)
  end do
  do i=1, 10
    print *,i-1,a(i-1)
  end do
end program sample6_2
```

MBPro13-2016:0617 nhirayama\$ █

# 配列の入出力

## READ文, PRINT文

```
INTEGER :: a( 10 )
PRINT `( 10I5 )`, ( a(i), i = 1, 10 )
PRINT `( 10I5 )`, a
PRINT `( 5I5 )`, a( 1 : 5 )
PRINT `( 10I5 )`, ( a(i), i = 10, 1, -1 )

DO i = 1, 10
    PRINT `( 10I5 )`, a(i)
END DO

READ *, a
READ *, ( a(i), i = 1, 10 )

DO i = 1, 10
    READ *, a(i)
END DO
```

13

# 出力時

## READ文での書式の指定

PRINT 書式識別子, 出力データのリスト

> 書式は、コンマ「,」または斜線「/」で区切って並べたリスト全体を、'()'でくくって構成する。斜線の位置で改行が行われる。 $w, d, r, n$ は整数値。

Iw 整数データを文字幅wに右詰め

Fw.d 実数データを文字幅wに右詰め、小数点以下d桁、 $w > d + 1$

Ew.d 実数データを10の指数表式、 $w > d + 5$

Dw.d 倍精度実数データを10の指数表式、 $w > d + 6$

Aw 文字データをw文字幅に右詰め

r.. これらの項目の前に反復数を書くことができる。

nX n桁スキップ。 $n = 1$ も省略できない。

```
PRINT '( 1X, I3, 2X, 3F10.5)', k, s, c, p
```

```
[MacPro2017:0607 nhirayama$ ./a.out
```

```
112312123456789012345678901234567890
      0      0.00000      1.00000      1.00000
     10      0.17365      0.98481      1.00000
     20      0.34202      0.93969      1.00000
```

14

```

program sample6_3
  integer :: a(10), i

  do i=1, 10
    a(i)=i*i
  end do

  print '( 10I5 )', (a(i), i=1,10)
  print '( 10I5 )', a
  print '( 5I5 )', a(1:5)
  print '( 10I5 )', (a(i), i=10,1,-1)

  do i=1,10
    print '( 10I5 )',a(i)
  end do
end program sample6_3

```

```

MBPro13-2016:0617 nhirayama$ gfortran sample6_3.f90
MBPro13-2016:0617 nhirayama$ ./a.out
  1   4   9  16  25  36  49  64  81 100
  1   4   9  16  25  36  49  64  81 100
  1   4   9  16  25
100  81  64  49  36  25  16   9   4   1
  1
  4
  9
 16
 25
 36
 49
 64
 81
100
MBPro13-2016:0617 nhirayama$

```

**PRINT '( 1X, I3, 2X, 3F10.5)', k,s,c,p**

スペース1桁, 3桁の整数が1つ, スペース2桁, 10桁の実数(小数点以下5桁)が3つ

```

MacPro2017:0607 nhirayama$ ./a.out
112312123456789012345678901234567890
  0      0.00000  1.00000  1.00000
 10     0.17365  0.98481  1.00000
 20     0.34202  0.93969  1.00000

```

15

# 本日の課題

## 2025.6.18

1. ベクトル  $u = (1.2, 3.5)$  と  $v = (4.1, 2.6)$  の和, 差, 積, 商, 内積を求めるプログラムを作成する。
2. 1から100までの整数の一樣乱数を配列を用いて20個生成し, その最大値と最小値を求めよ。また, 最大値と最小値が何番目の乱数であるのか示せ。なお, 最大値, 最小値が複数ある場合は最も最初のものでよい。



# 課題1 (2025.6.18)

## ベクトル演算

1. ベクトル  $u = (1.2, 3.5)$  と  $v = (4.1, 2.6)$  の和, 差, 積, 商, 内積を求めるプログラムを作成する。

提出物: 実行結果のスクリーン画像 kadai09\_1.png

$$\text{和: } u + v = (u(1) + v(1), u(2) + v(2))$$

$$\text{差: } u - v = (u(1) - v(1), u(2) - v(2))$$

$$\text{積: } u \times v = (u(1) \times v(1), u(2) \times v(2))$$

$$\text{商: } u \div v = (u(1) \div v(1), u(2) \div v(2))$$

$$\text{内積: } u \cdot v = u(1)v(1) + u(2)v(2)$$

# 乱数

## 一様乱数を生成する

- 一 計算機で用いることができる乱数は擬似乱数

```
call random_number(x)
```

- 一 変数  $x$  に区間  $[0, 1)$  の一様乱数が代入される。  $x$  は実数型であれば配列でもよい。配列の場合は全ての要素に一様乱数が代入される。
- 一 擬似乱数: 初期値により決定論的な数列である。初期値を指定する。

```
implicit none
integer:: nseed, clock
integer, allocatable:: seed(:)
call system_clock(clock)
call random_seed(size=nseed)
allocate(seed(nseed))
seed=clock
call random_seed(put=seed)
```

# 課題2 (2025.6.18)

## 一様乱数を用いて配列の最大, 最小

1. 1から100までの整数の一様乱数を配列を用いて20個生成し, その最大値と最小値を求めよ。また, 最大値と最小値が何番目の乱数であるのか示せ。なお, 最大値, 最小値が複数ある場合は最も最初のものでよい。

ただし, 組み込み関数 MAX( ), MIN( ) は使わないこと。

提出物: プログラム kadai09\_2.f90

実行画面 kadai09\_2.png

# 課題2(1)

## 20個の乱数を発生させる

1. 乱数 (実数) を発生させる。配列を用いる。

```
implicit none
integer :: nseed, clock
integer, allocatable :: seed(:)
real :: dum(20)
call system_clock(clock)
call random_seed(size=nseed)
allocate(seed(nseed))
seed=clock
call random_seed(put=seed)
```

2. 1から100の乱数 (整数) を発生させる。配列を用いる。

```
call random_number(dum)
int(100.0*dum)+1
```

# 課題2(2)

## アルゴリズム：最大値，最小値を探索

3.  $i$ 番目の配列が最大値より大きいのか，最小値より小さいか調べる。
4. 最大値 $nmax$ ，最小値 $nmin$ を配列の1番目と仮定する。最大値の順番 $pmax$ ，最小値の順番 $pmin$ を1とする。

```
nmax=d(1)
pmax=1
nmin=d(1)
pmin=1
```

5. 配列の2番目以降から， $d(i)$ と $nmax$ ， $nmin$ と比較する。
6.  $nmax$ より大きい場合には， $nmax$ と $pmax$ を更新する。 $nmin$ より小さい場合には， $nmin$ と $pmin$ を更新する。
7. すべて比較し終わったら，最大値，最小値を表示する。