WRITE 文

ソースファイル名：ex1.for，実行ファイル名：ex1.out

```
$ vi   ex1.for
$ gfortran   -o   ex1.out   ex1.for
$ ./ex1.out
```

```
      write(*,*)'Hello, world.'
      End
```

```
#include<stdio.h>
main()
{
   printf("Hello, world.");
   return 0;
}
```

*READ 文

ソースファイル名：ex2.for，実行ファイル名：ex2.out

```
$ vi   ex2.for
$ gfortran   -o   ex2.out   ex2.for
$ ./ex2.out
```

```
      integer a, b, c, d, e, f
      write(*,*)'a = ?'
      read(*,*)a

      write(*,*)'b = ?'
      read(*,*)b

      c = a + b
      d = a - b
      e = a * b
      f = a / b

      write(*,*)a,'+', b, '=',c
      write(*,*)a,'-', b, '=',d
      write(*,*)
      write(*,*)

      end
```

```
#include<stdio.h>
main()
{
   int a, b, c, d, e, f;
   printf("a = ?¥n");
   scanf("%d", &a);
   printf("b = ?¥n");
   scanf("%d", &b);
   c = a + b;
   d = a - b;
   e = a * b;
   f = a / b;
   printf("%d + %d = %d¥n", a, b, c);
   printf("%d - %d = %d¥n", a, b, d);
   printf("%d * %d = %d¥n", a, b, e);
   printf("%d / %d = %d¥n", a, b, f);
   return 0;
}
```

DO 文

ソースファイル名：ex3.for，実行ファイル名：ex3.out

```
$ vi    ex3.for
$ gfortran    -o    ex3.out    ex3.for
$ ./ex3.out
```

```
      integer i, n, s

      write(*,*)
      read(*,*)

      s = 0
      do i = 1, n
        
      end do

      write(*,'(a,i2,a,i4)')'sum form 1 to',n,'=',s

      end
```

```
#include <stdio.h>
main()
{
   int i, n, s;
   printf("n = ?¥n");
   scanf("%d", &n);
   s = 0;
   for(i = 1;i <= n;i++){
      s = s + i;
   }
   printf("sum from 1 to %d = %d¥n", n, s);
   return 0;
}
```

*IF 文，配列

ソースファイル名：ex4.for，実行ファイル名：ex4.out

```
$ vi   ex4.for
$ gfortran   -o   ex4.out   ex4.for
$ ./ex4.out
```

```
      integer a(5), temp, i, j

      do i = 1, 5
        write(*,100) 'a[',i,'] ='
        read(*,*) a(i)
      end do
100   format(a,i1,a)

      write(*,*) 'Numbers you set;'
      do i = 1, 5
        write(*,200) 'a[',i,'] =',a(i)
      end do
200   format(a,i1,a,i2)

      do i = 1, 4
        [          ]
        if(a(i) .lt. a(j)) then
          [          ]
          [          ]
          [          ]
        end if
        [      ]
      end do

      write(*,*) 'Arranged numbers are'
      write(*,*)(a(i), i = 1,5)

      end
```

```
#include <stdio.h>
main()
{
   int a[5], temp, i, j;
   for(i = 0;i <= 4;i++){
      printf("a[%d]= ?", i+1);
   scanf("%d", &a[i]);
   }
   printf("Numbers you set; ¥n");
   for(i = 0;i <= 4;i++){
      printf("a[%d] = %d¥n", i+1, a[i]);
   }
   for(i = 0;i <= 3;i++){
      for(j = i+1;j <= 4;j++){
         if(a[i] < a[j]){
         temp = a[j];
         a[j] = a[i];
         a[i] = temp;
         }
      }
   }
   printf("Arranged numbers are¥n");
   for(i = 0;i <= 4;i++){
      printf("%d ¥t", a[i]);
   }
   printf("¥n");
   return 0;
}
```

配列

ソースファイル名：ex5.for，実行ファイル名：ex5.out

```
      real*8 a(10,10), b(10,10), c(10,10)
      [redacted] i, j, k, n

      write(*,*) 'dimension n =? ( <= 10)'
      read(*,*) n

      write(*,*) 'Input the entries of A'
      do i = 1, n
        do j = 1, n
          write(*,100)'a[',i,'][',j,']=?'
          read(*,*) a(i,j)
        end do
      end do

      write(*,*) 'Input the entries of B'
      [redacted]
        [redacted]
          [redacted]
          [redacted]
        [redacted]
      [redacted]

      do i = 1, n
        do j = 1, n
          c(i,j) = 0.
          do k = 1, n
            c(i,j) = [redacted]
          end do
        end do
      end do

      write(*,*) [redacted]
      [redacted]
        [redacted]
      [redacted]

100   format(a,i1,a,i1,a)
      end
```

行列の掛け算　C = A B

```
#include <stdio.h>
main()
{
  float a[10][10], b[10][10], c[10][10];
  int i, j, k, n;
  printf("dimension n=? (<=10)");
  scanf("%d", &n);
  printf("Input the entries of A¥n");
  for(i = 1;i <= n;i++){
    for(j = 1;j <= n;j++){
      printf("a[%d][%d] = ?", i, j);
      scanf("%f", &a[i-1][j-1]);
    }
  }
  printf("Matrix A = ¥n");
  for(i = 1;i <= n;i++){
    for(j = 1;j <= n;j++){
      printf("%f¥t", a[i-1][j-1]);
    }
    printf("¥n");
  }
  printf("Input the entries of B¥n");
  for(i = 1;i <= n;i++){
    for(j = 1;j <= n;j++){
      printf("b[%d][%d] = ?", i, j);
      scanf("%f", &b[i-1][j-1]);
    }
  }
  printf("Matrix B = ¥n");
  for(i = 1;i <= n;i++){
    for(j = 1;j <= n;j++){
      printf("%f¥t", b[i-1][j-1]);
    }
    printf("¥n");
  }
  for(i = 1;i <= n;i++){
    for(j = 1;j <= n;j++){
      c[i-1][j-1] = 0;
      for(k = 1;k <= n;k++){
        c[i-1][j-1] = c[i-1][j-1] +a[i-1][k-1] * b[k-1][j-1];
      }
    }
  }
  printf("Matrix C = A * B =¥n");
  for(i = 1;i <= n;i++){
    for(j = 1;j <= n;j++){
      printf("%f¥t", c[i-1][j-1]);
    }
    printf("¥n");
  }
  return 0;
}
```

サブルーチン

ソースファイル名：ex6.for，実行ファイル名：ex6.out

```
$ vi    ex6.for
$ gfortran    -o    ex6.out    ex6.for
$ ./ex6.out
```

```
      integer a, b;

      a = 5
      b = 10

      write(*,*)'a=',a,'b =',b
      call replace(a,b)
      write(*,*)'a=',a,'b =',b

      end

      subroutine replace(a,b)
      integer a, b, tmp

      tmp = a
      a = b
      b = tmp

      return
      end
```

```
#include <stdio.h>

void replace(int *x, int *y);

main()
{
   int a, b;
   a = 5;
   b = 10;

   printf("a = %d, b= %d¥n",a,b);

   replace(&a,&b);

   printf("a = %d, b= %d¥n",a,b);
   return 0;
}

void replace(int *x, int *y)
{
   int tmp;
   tmp = *x;
   *x = *y;
   *y = tmp;
}
```

Do 文の復習

ソースファイル名：ex7.for，実行ファイル名：ex7.out

```
$ vi    ex7.for
$ gfortran    -o    ex7.out    ex7.for
$ ./ex7.out
```

```
do i = 1,10,1
  x = 2097 * 1201          (←学籍番号上 4 桁×学籍番号下 4 桁)
  write(*,*)x, i
end do
end
```

ソースファイル名：ex8.for，実行ファイル名：ex8.out

```
$ vi   ex8.for
$ gfortran   -o   ex8.out   ex8.for
$ ./ex8.out
```

```
      real*8 t, x, u, dx, dt
      character a(60,10)

      t = 0.
      x = 0.
      u = 2.0
      dt = 1.0

      do k = 1,20
       dx = u * dt
       x = x + dx
       t = t + dt

       write(*,*) t, x
c.. graphics
        do j = 1,3
         do i = 1, 50
          a(i,j)= '*'
         end do;
        end do

        a(int(x+1),2)= 'o'

        do j = 1,3
         write(*,*) (a(i,j),i = 1,50)
        end do
        write(*,*) '*--------------------------------------------------*'
      end do

      end
```

サブルーチン化（ソースファイル名：ex8.for，実行ファイル名：ex8.out）

```
      real*8 t, x, u, dx, dt
      character a(60,10)

      t = 0.
      x = 0.
      u = 2.0
      dt = 1.0

      do k = 1,20
       dx = u * dt
       x = x + dx
       t = t + dt
       call graphics
      end do
      end

      subroutine graphics(t,x,a)


       write(*,*) t, x
c.. graphics
        do j = 1,3
         do i = 1, 50;
          a(i,j)= '*'
         end do;
          end do

        a(int(x+1),2)= 'o'

        do j = 1,3
         write(*,*) (a(i,j),i = 1,50)
        end do
        write(*,*) '*--------------------------------------------------*'

      return
      end
```

ソースファイル名：ex9.for，実行ファイル名：ex9.out

```
$ vi   ex9.for
$ gfortran   -o   ex9.out   ex9.for
$ ./ex9.out
```

```
real*8 t, y, u, dy, du, dt, a

t = 0.
y = 0.
u = 0.
dt = 0.25
a = 9.8

write(*,*)'time                positon                velocity'
do i = 1, 20
  dy = u * dt
    y = y + dy
  du = a * dt
    u = u + du
    t = t + dt
  write(*,*)t, y, u
end do

end
```

ファイルへの書き込み（**open, close**）
ソースファイル名：ex10.for，実行ファイル名：ex10.out

```
$ cp    ex9.for    ex10.for
$ vi    ex10.for
$ gfortran    -o    ex10.out    ex10.for
$ ./ex10.out
$ cat    yamada_data (または    vi    yamada_data)
```

```
      real*8 t, y, u, dy, du, dt, a

      t = 0.
      y = 0.
      u = 0.
      dt = 0.25
      a = 9.8

      open(10,file=' yamada_data',status='unknown')
      write(10,*)'time                positon                velocity'
      do i = 1, 20
        dy = u * dt
          y = y + dy
        du = a * dt
          u = u + du
          t = t + dt
        write(10,*)t, y, u
      end do

      close(10)
      end
```

(yamada_data のファイルの内容は次ページ↓)

time	positon	velocity
0.250000000000000	0.00000000000000	2.45000004768372
0.500000000000000	0.612500011920929	4.90000009536743
0.750000000000000	1.83750003576279	7.35000014305115
1.00000000000000	3.67500007152557	9.80000019073486
1.25000000000000	6.12500011920929	12.2500002384186
1.50000000000000	9.18750017881393	14.7000002861023
1.75000000000000	12.8625002503395	17.1500003337860
2.00000000000000	17.1500003337860	19.6000003814697
2.25000000000000	22.0500004291534	22.0500004291534
2.50000000000000	27.5625005364418	24.5000004768372
2.75000000000000	33.6875006556511	26.9500005245209
3.00000000000000	40.4250007867813	29.4000005722046
3.25000000000000	47.7750009298325	31.8500006198883
3.50000000000000	55.7375010848045	34.3000006675720
3.75000000000000	64.3125012516975	36.7500007152557
4.00000000000000	73.5000014305115	39.2000007629395
4.25000000000000	83.3000016212463	41.6500008106232
4.50000000000000	93.7125018239021	44.1000008583069
4.75000000000000	104.737502038479	46.5500009059906
5.00000000000000	116.375002264977	49.0000009536743