西安交通大学工程分析程序设计Fortran上机作业

Ganlv

June 2, 2016

Contents

1 Fortran程序设计基础

上机目的：掌握程序书写、字符集和标识符、数据类型、声明、算术表达式、表控输入输出语句等编程基本概念。

1.1 求表达式的值

若有实数$A = 1.0, B = 3.5, T = 10.0, X = 5.0$，整数$I = -5, J = 7, K = 3$，求下列表达式的值。

$$
\begin{aligned}
&-(A+T)\\
&(B+(X/T))/(4.0*A)\\
&(I*J)/K\\
&(I/K)*J+T/X\\
&-(K+1)/5+I*A-B\\
&SQRT(REAL(ABS(K)+1))\\
&MAX(J,MOD(J,K))\\
&J+INT(T/B)/2
\end{aligned}
$$

```
program EX0101
  implicit none
  real, parameter :: A = 1.0, B = 3.5, T = 10.0, X = 5.0
  integer, parameter :: I = -5, J = 7, K = 3
  print *, '-(A+T)               ', -(A + T)
  print *, '(B+(X/T))/(4.0*A)    ', (B + (X / T)) / (4.0 * A)
  print *, '(I*J)/K              ', (I * J) / K
  print *, '(I/K)*J+T/X          ', (I / K) * J + T / X
  print *, '-(K+1)/5+I*A-B       ', -(K + 1) / 5 + I * A - B
  print *, 'SQRT(REAL(ABS(K)+1)) ', sqrt(real(abs(K) + 1))
  print *, 'MAX(J,MOD(J,K))      ', max(J, mod(J, K))
  print *, 'J+INT(T/B)/2         ', J + int(T / B) / 2
end program
```

1.2 求数学表达式的值

若有$A = 1.0, B = 2.0, C = -1.0$，求下列数学表达式的值：

$$\frac{3A^2 + 4B^2}{A - B}$$

$$\frac{-B + \sqrt{B^2 - 4AC}}{2A}$$

$$\frac{6\ln(B + C)^2}{\frac{140}{3+A}}$$

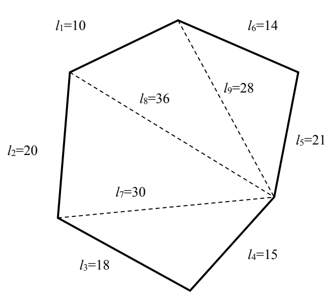
$$\cos\frac{B}{\sqrt{A^2 + B^2}}$$

$$\sin\left(\arctan\frac{\sqrt{A^2 + B^2}}{|C|}\right)$$

```
program EX0102
  implicit none
  real, parameter :: A = 1.0, B = 2.0, C = -1.0
  print *, '(3*A^2+4*B^2)/(A-B)            ', (3 * A ** 2 + 4* B ** 2) / (A - B)
  print *, '(-B+SQRT(B^2-4*A*C))/(2*A)     ', (-B + sqrt(B ** 2 - 4 * A * C)) / (2 * A)
  print *, '6*LN((B+C)^2)/(140/(3+A))      ', 6 * log((B + C) ** 2) / (140 / (3 + A))
  print *, 'COS(B/SQRT(A^2+B^2))           ', cos(B / sqrt(A ** 2 + B ** 2))
  print *, 'SIN(ATAN(SQRT(A^2+B^2)/ABS(C))) ', sin(atan(sqrt(A ** 2 + B ** 2) / abs(C)))
end program
```

1.3 求六边形面积

如图，求六边形面积。



提示：三角形面积$area = \sqrt{s(s-a)(s-b)(s-c)}$，其中$s = \frac{a+b+c}{2}$，$a, b, c$为三个边长。

```
program EX0103
  implicit none
  real, parameter :: L1 = 10, L2 = 20, L3 = 18, L4 = 15, L5 = 21, L6 = 14, L7 = 30, L8 = 36, L9 = 28
  real :: s, area = 0
  s = (L1 + L8 + L9) / 2
  area = area + sqrt(s * (s - L1) * (s - L8) * (s - L9))
  s = (L2 + L7 + L8) / 2
  area = area + sqrt(s * (s - L2) * (s - L7) * (s - L8))
  s = (L3 + L4 + L7) / 2
  area = area + sqrt(s * (s - L3) * (s - L4) * (s - L7))
  s = (L5 + L6 + L9) / 2
  area = area + sqrt(s * (s - L5) * (s - L6) * (s - L9))
  print *, 'the area of the hexagon is ', area
end program
```

1.4 计算人口增长

假设目前我国人口13.0亿人，每年增长率为1.5%，求n年以后的人口数。

要求：n从键盘输入。

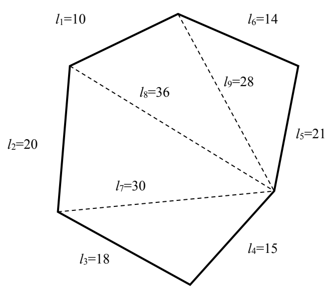
```
program EX0104
  implicit none
  real, parameter :: N0 = 13.0
  integer :: n
  print *, 'Please input years past:'
  read *, n
  print *, 'Now there are ', N0 * (1.015 ** n), '*10^8 people'
end program
```

2 模块化编程（1）

上机目的：掌握主程序、内部例程、外部例程、模块等功能的使用方法。

2.1 求六边形面积（函数）

如图，求六边形面积。



要求：用内部函数来实现

提示：三角形面积$area = \sqrt{s(s-a)(s-b)(s-c)}$，其中$s = \frac{a+b+c}{2}$，$a, b, c$为三个边长。

```
program EX0201
  implicit none
  real, parameter :: L1 = 10, L2 = 20, L3 = 18, L4 = 15, L5 = 21, L6 = 14, L7 = 30, L8 = 36, L9 = 28
  print *, 'the area of the hexagon is', &
    area(L1, L8, L9) + area(L2, L7, L8) + area(L3, L4, L7) + area(L5, L6, L9)
  contains
    function area(a, b, c)
      real a, b, c, area, s
      s = (a + b + c) / 2
      area = sqrt(s * (s - a) * (s - b) * (s - c))
    end function
end program
```

2.2 牛顿迭代法解方程

使用牛顿法求解以下方程的根：

$$x^2 + 4x + 1 = 0$$

$$7x^4 + 6x^3 - 5x^2 + 4x + 3 = 0$$

要求：用外部子程序

```
program EX0202
  implicit none
  integer, parameter :: MAX_N = 100
  real, parameter :: EPS = 1.0e-6
  integer :: n
  print *, 'To solve x^2+4*x+1=0 with Newton''s method'
  call newton(f1, EPS, MAX_N)
  print *, 'To solve 7*x^4+6*x^3-5*x^2+4*x+3=0 with Newton''s method'
```

```
  call newton(f2, EPS, MAX_N)
  contains
    function f1(x)
      real :: x, f1
      f1 = x - (x ** 2 + 4 * x + 1) / (2 * x + 4)
    end function
    function f2(x)
      real :: x, f2
      f2 = x - (7 * x ** 4 + 6 * x ** 3 - 5 * x ** 2 + 4 * x + 3) / (28 * x ** 3 + 18 * x ** 2 - 10 * x + 4)
    end function
end program
subroutine newton(f, eps, maxN)
  implicit none
  real :: f, eps, x, tmpx, dx
  integer :: n, maxN
  dx = 1.0
  n = 0
  print *, 'please input an initial number:'
  read *, x
  do while (abs(dx) > eps .and. n < maxN)
    tmpx = x
    x = f(x)
    dx = x - tmpx
    n = n + 1
  end do
  print *, 'The result is ', x
end subroutine
```

2.3 表达式求值（子程序）

编写一个子例程子程序$sum(s, t, n1, n2)$。把整型数$n1$到$n2$进行求和，并把求和的结果放置到s，把整型数$n1$到$n2$进行求积，并把求积的结果放置到t。并用这个子程序来计算：

$$y = (1 + 2 + 3 + 4) + (3 + 4 + 5 + 6 + 7 + 8) + (3 \times 4 \times 5 \times 6) - (1 \times 2 \times 3)$$

```
program EX0203
  implicit none
  integer :: s, t, y = 0
  call sum(s, t, 1, 4)
  y = y + s
  call sum(s, t, 3, 8)
  y = y + s
  call sum(s, t, 3, 6)
  y = y + t
  call sum(s, t, 1, 3)
  y = y - t
  print *, 'y = (1+2+3+4)+(3+4+5+6+7+8)+(3*4*5*6)-(1*2*3) = ', y
  contains
    subroutine sum(s, t, n1, n2)
      integer :: s, t, n1, n2, i
      s = 0; t = 1
      do i = n1, n2
        s = s + i
        t = t * i
      end do
    end subroutine
end program
```

2.4 表达式求值（模块）

编写一个模块程序，提供以下功能：定义出常量π, e；定义出例程，实现求和$\sum_{i=1}^{n} i^2$、求阶乘$n!$。并在主程序中执行：从键盘上输入整数n、实型数$a, r, r0$，求：

$$\frac{n!}{\sum_{i=1}^{n} i^2}$$

$$\frac{an}{2\pi r^2}\left(\frac{r}{r_0}\right)^n e^{-\left(\frac{r}{r_0}\right)^n}$$

```
module MyModule
  implicit none
  real*8, parameter :: PI = 3.1415926535897932_8, E = 2.7182818284590452_8
  contains
    function sumN2(n)
      integer :: n, sumN2, i
      sumN2 = 0
      do i = 1, n
        sumN2 = sumN2 + i ** 2
      end do
    end function
    function fact(n)
      integer :: n, fact, i
      fact = 1
      do i = 2, n
        fact = fact * i
      end do
    end function
end module
program EX0204
  use MyModule
  implicit none
  integer :: n
  real :: a, r, r0
  print *, 'please input n, a, r, r0'
  read *, n, a, r, r0
  print *, 'n!/sum(n^2,1,n) = ', real(fact(n)) / sumN2(n)
  print *, '(a*n/(2*PI*r^2))*((r/r0)^n)*(e^(-(r/r0)^n)) = ', &
    (a * n / (2 * PI * r ** 2)) * ((r / r0) ** n) * (E ** (-(r / r0) ** n))
end program
```

3 模块化编程（2）

上机目的：进一步掌握内部例程、外部例程、接口块、模块等功能的使用方法。练习例程重载、例程递归的使用方法。

3.1 求余数

利用例程重载编写一个求余数的函数$acr(a,b)$。

要求：对两个整型数和两个实型数都有效。

要求：不能用Fortran的内部函数$mod(x,y)$

提示：实型数相除，两个实型数相除后的商仍然取一个整数，但余数小于除数且大于零

```
program EX0301
  implicit none
  interface acr
    procedure acrInteger, acrReal
  end interface
  integer :: a, b
  real :: c, d
  print *, 'please input two integer numbers a, b'
  read *, a, b
  print *, 'acr(a,b) = ', acr(a, b)
  print *, 'please input two real numbers c, d'
  read *, c, d
  print *, 'acr(c,d) = ', acr(c, d)
  contains
    function acrInteger(a, b)
      integer :: a, b, acrInteger
      acrInteger = a - a / b * b
    end function
    function acrReal(a, b)
      real :: a, b, acrReal
      acrReal = a - int(a / b) * b
    end function
end program
```

3.2 求最大公约数

编写函数子程序$gdc(a,b)$求两个数的最大公约数。并调用此函数，求$1260, 198, 72$三个数的最大公约数。

提示：求最大公约数的算法：把两个数中大的那个数作为被除数，两数相除得到一个余数。用余数去除除数得到新一轮的余数。不断重复这一过程直到余数为0，这时另一个非零的数就是两个数的最大公约数。

```
program EX0302
  implicit none
  print *, 'gdc{1260,198,72} = ', gdc(gdc(1260, 198), 72)
  contains
    function gdc(a, b)
      integer :: a, b, gdc, a1, b1
      a1 = a
      b1 = b
      do while (a1 /= 0 .and. b1 /= 0)
        if (a1 > b1) then
          a1 = mod(a1, b1)
        else
          b1 = mod(b1, a1)
        end if
      end do
      gdc = a1 + b1
    end function
end program
```

3.3 求双曲正弦

编写一个函数子程序求$sinh(x)$

提示：以下分三个步骤完成：

（1）用递归算法，求出：

$$\frac{x^n}{n!} = \frac{x^{n-1}}{(n-1)!} \cdot \frac{x}{n}$$

（2）自行编程求出：

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \cdots = \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

要求：计算精度是$\frac{x^n}{n!} < 10^{-6}$

要求：不能用Fortran的内部函数exp(x)

（3）计算：

$$\sinh(x) = \frac{e^x - e^{-x}}{2}$$

```
program EX0303
  implicit none
  real :: x
  print *, 'Please input x to calculate sinh(x):'
  read *, x
  print *, 'sinh(x) = ', sinh1(x)
  contains
    recursive subroutine exp2(x, n, eps, term, ex)
      integer :: n, n1
      real :: x, eps, term, ex, term1
      term1 = term
      n1 = n
      if (n == 0) then
        term1 = 1.0
        ex = term1
        n1 = 1
      end if
      if (abs(term1) > eps) then
        if (n1 > 0) then
          term1 = term1 * x / n1
          ex = ex + term1
        end if
```

```
        call exp2(x, n1 + 1, eps, term1, ex)
      end if
    end subroutine
    function exp1(x, eps)
      real :: x, eps, exp1
      call exp2(x, 0, eps, 0.0, exp1)
    end function
    function sinh1(x)
      real, parameter :: EPS = 1e-6
      real :: x, sinh1
      sinh1 = (exp1(x, EPS) - exp1(-x, EPS)) / 2.0
    end function
end program
```

3.4 Euler法求解微分方程

编写一程序用Euler法求解微分方程$\frac{dy}{dx} = y^2 - x^2$，当$x = 0$时，$y = 1.0$。试求出$x = 0.1, 0.2, 0.3, 0.4, \cdots, 1.0$时的$y$值。

提示：算法如下：

Euler法求解$y' = f(x, y(x))$，定解条件：$\begin{cases} x = x_0 \\ y = y_0 \end{cases}$。取向前差分，

令

$$y' = \frac{1}{h}(y_{n+1} - y_n), h = x_{n+1} - x_n$$

得

$$y_{n+1} = y_n + hf(x_n, y_n) + O(h^2)$$

所以

$$\begin{cases} x = x_0 \\ y = y_0 \end{cases} \Rightarrow f(x_0, y_0) \Rightarrow y1, f(x_1, y_1) \Rightarrow y2 \Rightarrow \cdots \Rightarrow y_n$$

```
program EX0304
  implicit none
  real, parameter :: DX = 0.1
  real :: x0 = 0.0, y0 = 1.0, x1
  x1 = x0 + DX
  print *, 'To solve "dy/dx=y^2-x^2" with Euler''s method,'
  print *, 'when x=', x0, 'y=', y0, ', this is what we know.'
  do while (x0 < 1.0)
    y0 = eular(f, x0, y0, x1)
    print *, 'when x=', x1, 'y=', y0
    x0 = x1
    x1 = x1 + DX
  end do
  contains
    function eular(f, x0, y0, x1)
      real :: f, x0, y0, x1, eular
      eular = y0 + (x1 - x0) * f(x0, y0)
    end function
    function f(x, y)
      real :: x, y, f
      f = y ** 2 - x ** 2
    end function
end program
```

4 结构化编程

上机目的：练习使用选择结构和循环结构编制程序。

4.1 分段函数求值

由键盘输入x，求y值。

$$y = \begin{cases} x & ,0 \leqslant x < 10 \\ x^2+1 & ,10 \leqslant x < 20 \\ x^3+x^2+1 & ,20 \leqslant x < 30 \end{cases}$$

```
program EX0401
  implicit none
  real :: x
  print *, '       { x         ,  0 <= x < 10'
  print *, 'f(x) = { x^2+1     , 10 <= x < 20'
  print *, '       { x^3+x^2+1 , 20 <= x < 30'
  print *, 'Please input x to calculate the piecewise function f(x):'
  read *, x
  if (x < 0) then
    print *, 'ERROR: x < 0'
  else if (x < 10) then
    print *, x
  else if (x < 20) then
    print *, x ** 2 + 1
  else if (x < 30) then
    print *, x ** 3 + x ** 2 + 1
  else
    print *, 'ERROR: x >= 30'
  end if
end program
```

4.2 简单排序

输入4个数a, b, c, d，按由大到小的顺序打印出来。

```
program EX0402
  implicit none
  real :: a, b, c, d
  print *, 'Please input 4 numbers to sort:'
  read *, a, b, c, d
  if (a > b .and. b > c .and. c > d) print *, a, b, c, d
  if (a > b .and. b > d .and. d > c) print *, a, b, d, c
  if (a > c .and. c > b .and. b > d) print *, a, c, b, d
  if (a > c .and. c > d .and. d > b) print *, a, c, d, b
  if (a > d .and. d > b .and. b > c) print *, a, d, b, c
  if (a > d .and. d > c .and. c > b) print *, a, d, c, b
  if (b > a .and. a > c .and. c > d) print *, b, a, c, d
  if (b > a .and. a > d .and. d > c) print *, b, a, d, c
  if (b > c .and. c > a .and. a > d) print *, b, c, a, d
  if (b > c .and. c > d .and. d > a) print *, b, c, d, a
  if (b > d .and. d > a .and. a > c) print *, b, d, a, c
  if (b > d .and. d > c .and. c > a) print *, b, d, c, a
  if (c > a .and. a > b .and. b > d) print *, c, a, b, d
  if (c > a .and. a > d .and. d > b) print *, c, a, d, b
  if (c > b .and. b > a .and. a > d) print *, c, b, a, d
  if (c > b .and. b > d .and. d > a) print *, c, b, d, a
  if (c > d .and. d > a .and. a > b) print *, c, d, a, b
  if (c > d .and. d > b .and. b > a) print *, c, d, b, a
  if (d > a .and. a > b .and. b > c) print *, d, a, b, c
  if (d > a .and. a > c .and. c > b) print *, d, a, c, b
  if (d > b .and. b > a .and. a > c) print *, d, b, a, c
  if (d > b .and. b > c .and. c > a) print *, d, b, c, a
  if (d > c .and. c > a .and. a > b) print *, d, c, a, b
  if (d > c .and. c > b .and. b > a) print *, d, c, b, a
end program
```

4.3 判断素数

编写一个子程序，判断一个整数是否素数

```
program EX0403
  implicit none
  integer :: x
  print *, 'Please input a number to determine whether it is a prime number:'
  read *, x
  print *, is_prime(x)
  contains
    function is_prime(x)
      integer :: x, i
      logical :: is_prime
      if (x <= 1) then
        is_prime = .false.
      else
        is_prime = .true.
        do i = 2, (x - 1)
          is_prime = is_prime .and. (mod(x, i) > 0)
          if (.not. is_prime) exit
        end do
      end if
    end function
end program
```

4.4 验证哥德巴赫猜想

利用上一编程结果，对$6 \sim 1000$的所有偶数验证哥德巴赫猜想，列出每个数的所有分解式，并统计每个数的分解式个数（两个素数互换位置的只算其中一个）。研究随着偶数增大，分解的等式个数的变化规律。

提示：对于大于等于于6的任一偶数，先分解为两个奇数，然后验证它们是否为素数，如果都是素数，则输出结果。

```
program EX0404
  implicit none
  integer :: i, j, k, n
  print *, 'Verifying the Goldbach Conjecture within 1000'
  do i = 6, 1000, 2
    n = 0
    do j = 3, (i / 2), 2
      k = i - j
      if (is_prime(j) .and. is_prime(k)) then
        print *, i, '=', j, '+', k
        n = n + 1
      end if
    end do
    print *, i, ' have ', n, ' decomposition'
  end do
  contains
    function is_prime(x)
      integer :: x, i
      logical :: is_prime
      if (x <= 1) then
        is_prime = .false.
      else
        is_prime = .true.
        do i = 2, (x - 1)
          is_prime = is_prime .and. (mod(x, i) > 0)
          if (.not. is_prime) exit
        end do
      end if
    end function
end program
```

4.5 分解质因数

输入一个自然数，进行因子分解并输出结果。

例如：$24 = 1 \times 2 \times 2 \times 2 \times 3$（输出格式不限）。

```
program EX0405
```

```
  implicit none
  integer :: i, x, n
  print *, 'Please input a number to do Prime Factorization'
  read *, x
  do i = 2, int(sqrt(real(x)))
    n = 0
    do while (mod(x, i) == 0)
      x = x / i
      n = n + 1
    end do
    if (n > 0) then
      print *, i, '^', n
    end if
    if (x <= 1) then
      exit
    end if
  end do
end program
```

5 数组（1）

上机目的：练习数组的声明、存储、操作，以及数组参数、动态数组、数组函数的使用。

5.1 杨辉三角形

打印杨辉三角形（格式不限）。

```
program EX0501
  implicit none
  integer i, n
  integer, allocatable :: a(:)
  print *, 'Please input calculate how many rows of Pascal''s Triangle:'
  read *, n
  allocate(a(n))
  a = 0
  a(1) = 1
  print *, a(1)
  do i = 2, n
    a = (/0, a(1 : n - 1)/) + a
    print *, a(1 : i)
  end do
  deallocate(a)
end program
```

5.2 矩阵乘法

输入两个矩阵，并用矩阵作为子例程的参数，用子例程完成任意两个矩阵的乘法。

要求：如果可能，用数组函数子程序来完成这一功能

```
program EX0502
  implicit none
  integer :: i, j, m, n, s
  real, allocatable :: a(:, :), b(:, :), c(:, :)
  print *, 'Please input m,n,s for the Matrix a(m*n),b(n*s):'
  read *, m, n, s
  allocate(a(n, m), b(s, n), c(s, m))
  print *, 'Please input each element in the Matrix in a and b:'
  read *, a, b
  c = reshape((/((sum(a(:, i) * b(j, :)), j = 1, s), i = 1, m)/), (/s, m/))
  do i = 1, m
    print *, c(:, i)
  end do
  deallocate(a, b, c)
end program
```

5.3 冒泡排序

用“冒泡算法”对一个数列$\{a_n\}$进行排序。

提示：冒泡排序的步骤如下：

（1）若$a_2 < a_1$，则将a_2与a_1对换位置。

（2）若$a_3 < a_2$，则将a_3与a_2对换位置；然后再重复步骤（1）。

（3）对a_4, a_5等数列中的所有数，重复以上算法，直到整个数组中的元素从小到大排列。

（因为这种算法的特点是，每个元素总是和比它大的数交换位置，小的元素不断“上浮”，象水中的气泡不断上浮一样，所以称之为“冒泡算法”）

```
program EX0503
  implicit none
  integer :: n, i, j
  integer, allocatable :: a(:)
  print *, 'Please input how many integer numbers to do bubble sort:'
  read *, n
  allocate(a(n))
  print *, 'Please input each number:'
  read *, a
  do i = 1, n
    do j = 1, n - i
      if (a(j + 1) < a(j)) then
        a((/j, j + 1/)) = a((/j + 1, j/))
      end if
    end do
  end do
  print *, a
  deallocate(a)
end program
```

5.4 差集

从$\{a_n\}, \{b_n\}$两个数列中，把同时出现在两个数列中的数据删去。

例如：

$$a = [2\ 5\ 5\ 8\ 9\ 12\ 18]$$
$$b = [5\ 8\ 12\ 12\ 14]$$

操作完成后：

$$a = [2\ 9\ 18]$$
$$b = [14]$$

```
program EX0504
  implicit none
  integer :: m, n, i, j
  integer, allocatable :: a(:), b(:), a1(:), b1(:)
  logical, allocatable :: aflag(:), bflag(:)
  print *, 'Please input how many integer numbers in these two sets:'
  read *, m, n
  allocate(a(m), b(n), aflag(m), bflag(n))
  print *, 'Please input each number in these two sets:'
  read *, a, b
  aflag = (/(.not. any(a(i) == b), i = 1, m)/)
  bflag = (/(.not. any(a == b(i)), i = 1, n)/)
  allocate(a1(count(aflag)), b1(count(bflag)))
  j = 1
  do i = 1, m
    if (aflag(i)) then
      a1(j) = a(i)
      j = j + 1
    end if
  end do
  j = 1
  do i = 1, n
    if (bflag(i)) then
      b1(j) = b(i)
      j = j + 1
```

```
    end if
  end do
  print *, a1
  print *, b1
  deallocate(a, b, aflag, bflag, a1, b1)
end program
```

6 数组（2）

上机目的：练习数组的声明、存储、操作，以及数组参数、动态数组、数组函数的使用。

6.1 数组逆序

从键盘上输入10个整数，然后逆序打印出来。

```
program EX0601
  implicit none
  integer :: a(10)
  print *, 'Please input 10 numbers to reverse:'
  read *, a
  print *, a(10 : 1 : -1)
end program
```

6.2 数组轮换

输入任意n个数存放在数组中（如5个数$1, 2, 8, 2, 10$），请在屏幕上打印如下方阵

$$\begin{matrix} 1 & 2 & 8 & 2 & 10 \\ 10 & 1 & 2 & 8 & 2 \\ 2 & 10 & 1 & 2 & 8 \\ 8 & 2 & 10 & 1 & 2 \\ 2 & 8 & 2 & 10 & 1 \end{matrix}$$

```
program EX0602
  implicit none
  integer :: n, i
  integer, allocatable :: a(:)
  print *, 'Please input how many numbers to cyclic shift:'
  read *, n
  allocate(a(n))
  print *, 'Please input each number:'
  read *, a
  do i = 0, n - 1
    print *, a(n - i + 1 : n), a(1 : n - i)
  end do
  deallocate(a)
end program
```

6.3 选择排序

用选择法进行排序。

提示：假定数组$\{a_n\}$中有n个数，声明两个变量p, j，j每一趟都依次指向下一个，p则指向j所指数据右侧剩余的数据中最小的数。然后把j指向的元素和p指向的元素进行比较大小，若p指向的元素更小，则把二者对换。如此把j从1号位向后移动至数组末尾，则完成了排序。例如：

第一步：

$$\begin{matrix} 15 & 8 & 4 & 10 & 1 \\ j & & & & p \end{matrix}$$

j指向1号位，p指向$2 \sim 5$号位的最小值5号位的1，二者对换。

第二步：

$$\begin{array}{ccccc} 1 & 8 & 4 & 10 & 15 \\ & j & p & & \end{array}$$

j指向2号位，p指向$3\sim5$号位的最小值3号位的4，二者对换。

第三步：

$$\begin{array}{ccccc} 1 & 4 & 8 & 10 & 15 \\ & & j & p & \end{array}$$

j指向3号位，p指向$4\sim5$号位的最小值4号位的10，二者不对换。

第四步：

$$\begin{array}{ccccc} 1 & 4 & 8 & 10 & 15 \\ & & & j & p \end{array}$$

j指向4号位，p指向5号位的最小值号位的15，二者不对换。

根据以上四步，整个数组完成排序。要求在屏幕上显示排序法每步进行的情况。有兴趣的同学还可以比较选择法排序和“冒泡算法”进行排序的速度。

```
program EX0603
  implicit none
  integer :: n, j, p, amin, pmin
  integer, allocatable :: a(:)
  print *, 'Please input how many numbers to do selection sort:'
  read *, n
  allocate(a(n))
  print *, 'Please input each number:'
  read *, a
  do j = 1, n - 1
    pmin = j
    amin = a(pmin)
    do p = j + 1, n
      if (a(p) < amin) then
        amin = a(p)
        pmin = p
      end if
    end do
    a(pmin) = a(j)
    a(j) = amin
    print *, a
  end do
  deallocate(a)
end program
```

6.4 高斯消去法

利用高斯消去法求解线性代数方程组：

$$\begin{cases} x_1 + 2x_2 + 3x_3 = 1 \\ 5x_1 + x_2 - 3x_3 = -4 \\ 7x_1 + 8x_2 + 11x_3 = -3 \end{cases}$$

$$\begin{cases} 10x_1 + 2x_2 = 1 \\ 2x_1 + 3x_2 = 2 \end{cases}$$

提示：对于一般的n阶方程组

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \cdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n \end{cases}$$

高斯消去法步骤如下：

第一步：若$a_{11}\neq 0$，令$l_{i1}=\frac{a_{i1}}{a_{11}},i=2,3,\dots,n$，用$(-l_{i1})$乘第1个方程加到第$i$个方程上$(i=2,3,\dots,n)$，得同解方程组

$$\begin{cases} a_{11}^{(1)}x_1+a_{12}^{(1)}x_2+\cdots+a_{1n}^{(1)}x_n=b_1^{(1)} \\ \qquad a_{22}^{(2)}x_2+\cdots+a_{2n}^{(2)}x_n=b_2^{(2)} \\ \qquad\qquad \cdots \\ \qquad a_{n2}^{(2)}x_2+\cdots+a_{nn}^{(2)}x_n=b_n^{(2)} \end{cases}$$

第二步：若$a_{22}^{(2)}\neq 0$，令$l_{i2}=\frac{a_{i2}^{(2)}}{a_{22}^{(2)}},i=3,4,\dots,n$，用$(-l_{i2})$乘第2个方程加到第$i$个方程上$(i=3,4,\dots,n)$，将$a_{i2}^{(2)}(i=3,4,\dots,n)$消去。一般，设第$k-1$步后方程组化为如下的同解方程组

$$\begin{cases} a_{11}^{(1)}x_1+a_{12}^{(1)}x_2+\cdots+a_{1n}^{(1)}x_n=b_1^{(1)} \\ \qquad a_{22}^{(2)}x_2+\cdots+a_{2n}^{(2)}x_n=b_2^{(2)} \\ \qquad\qquad \cdots \\ \qquad a_{kk}^{(k)}x_k+\cdots+a_{kn}^{(k)}x_n=b_k^{(k)} \\ \qquad\qquad \cdots \\ \qquad a_{nk}^{(k)}x_k+\cdots+a_{nn}^{(k)}x_n=b_n^{(k)} \end{cases}$$

则**第k步：**若$a_{kk}^{(k)}\neq 0$，令$l_{ik}=\frac{a_{ik}^{(k)}}{a_{kk}^{(k)}},i=k+1,k+2,\dots,n$，用$(-l_{ik})$乘第$k$个方程加到第$i$个方程上$(i=k+1,k+2,\dots,n)$，得到如下的同解方程组

$$\begin{cases} a_{11}^{(1)}x_1+a_{12}^{(1)}x_2+\cdots+a_{1n}^{(1)}x_n=b_1^{(1)} \\ \qquad a_{22}^{(2)}x_2+\cdots+a_{2n}^{(2)}x_n=b_2^{(2)} \\ \qquad\qquad \cdots \\ \qquad a_{kk}^{(k)}x_k+\cdots+a_{kn}^{(k)}x_n=b_k^{(k)} \\ a_{k+1,k+1}^{(k+1)}x_{k+1}+\cdots+a_{k+1,n}^{(k+1)}x_n=b_{k+1}^{(k+1)} \\ \qquad\qquad \cdots \\ \quad a_{n,k+1}^{(k+1)}x_{k+1}+\cdots+a_{nn}^{(k+1)}x_n=b_n^{(k+1)} \end{cases}$$

按上述做法，做完$n-1$步，原方程组化为同解的上三角形方程组

$$\begin{cases} a_{11}^{(1)}x_1+a_{12}^{(1)}x_2+\cdots+a_{1n}^{(1)}x_n=b_1^{(1)} \\ \qquad a_{22}^{(2)}x_2+\cdots+a_{2n}^{(2)}x_n=b_2^{(2)} \\ \qquad\qquad \cdots \\ \qquad a_{kk}^{(k)}x_k+\cdots+a_{kn}^{(k)}x_n=b_k^{(k)} \\ \qquad\qquad \cdots \\ \qquad\qquad a_{nn}^{(n)}x_n=b_n^{(n)} \end{cases}$$

最后：设$a_{nn}^{(n)}\neq 0$，逐步代回得原方程组的解

$$\begin{cases} x_n=\dfrac{b_n^{(n)}}{a_{nn}^{(n)}} \\ x_k=\dfrac{b_k^{(k)}-\sum_{j=k+1}^{n}a_{kj}^{(k)}x_j}{a_{kk}^{(k)}}(k=n-1,n-2,\dots,2,1) \end{cases}$$

注意：上述公式中的$a_{ij}^{(k)},b_i^{(k)}$上标(k)，是用来区别消去过程中第k步利用的量。在用编程求解时，可把$a_{ij}^{(k)}$存在位置a_{ij}，$b_i^{(k)}$存在位置b_i。

```
program EX0604
  implicit none
  integer :: n, i, j, row
  real, allocatable :: a(:, :)
  print *, 'Please input how many equations:'
  read *, n
  allocate(a(n + 1, n))
  print *, 'Please input each number in the augmented matrix:'
  read *, a
  do i = 1, n
    row = 0
    do j = i, n
      if (a(i, j) /= 0) then
        row = j
        exit
      end if
    end do
    if (row > i) then
      a(:, (/row, i/)) = a(:, (/i, row/))
    else if (row < i) then
      print *, 'ERROR: indetermined equation!'
    end if
    do j = i + 1, n
      a(i : n + 1, j) = a(i : n + 1, j) - a(i : n + 1, i) * a(i, j) / a(i, i)
    end do
  end do
  do i = n, 1, -1
    a(i : n + 1, i) = a(i : n + 1, i) / a(i, i)
    a(n + 1, 1 : i - 1) = a(n + 1, 1 : i - 1) - a(n + 1, i) * a(i, 1 : i - 1)
  end do
  print *, a(n + 1, :)
  deallocate(a)
end program
```

7 派生类型

上机目的：练习派生类型的定义、构造、初始化，成员的调用和操作。操作符的重载。

7.1 构建派生类型

用派生类型构造一个班的学生的数据库（10个人），包括学号、姓名、3门课的分数，并进行如下操作：

（1）按学号顺序显示每个人的信息

（2）计算每个人的平均分，按从高到低的顺序从屏幕上显示每个人的信息及其平均分。

```
program EX0701
  implicit none
  integer, parameter :: STU_COUNT = 10
  integer, parameter :: SCORE_COUNT = 3
  type Student
    integer*8 :: stuId
    character(10) :: stuName
    real :: score(SCORE_COUNT)
  end type
  type(Student) :: students(STU_COUNT)
  integer :: i, j, tmpIndex
  integer*8 :: minId
  real :: average(STU_COUNT), maxAverage
  print *, 'Please input the student number, name, three subject''s score of the 10 students:'
  read *, students
  print *, 'Sort according to the student number:'
  do i = 1, STU_COUNT
    tmpIndex = i
    minId = students(tmpIndex) % stuId
    do j = i + 1, STU_COUNT
      if (students(j) % stuId < minId) then
        tmpIndex = j
        minId = students(tmpIndex) % stuId
```

```
      end if
    end do
    students((/i, tmpIndex/)) = students((/tmpIndex, i/))
    print *, students(i)
  end do
  average =(/(sum(students(i) % score) / SCORE_COUNT, i = 1, STU_COUNT)/)
  print *, 'Sort according to the average score:'
  do i = 1, STU_COUNT
    tmpIndex = i
    maxAverage = average(tmpIndex)
    do j = i + 1, STU_COUNT
      if (average(j) > maxAverage) then
        tmpIndex = j
        maxAverage = average(tmpIndex)
      end if
    end do
    average((/i, tmpIndex/)) = average((/tmpIndex, i/))
    students((/i, tmpIndex/)) = students((/tmpIndex, i/))
    print *, students(i), average(i)
  end do
end program
```

7.2 操作符重载

仿照教材的例7-4：

（1）设计一个“+”操作符的重载，实现两个集合的合集

（2）设计一个“-”操作符的重载，把集合A中那些同时又出现在集合B中的元素去掉。

```
module SetModule
  implicit none
  integer, parameter :: MAX_COUNT = 50
  type :: Set
    integer :: counts = 0, elements(MAX_COUNT)
  end type
  interface operator(.in.)
    module procedure inSet
  end interface
  interface operator(+)
    module procedure add
  end interface
  interface operator(-)
    module procedure minus
  end interface
  contains
    function inSet(x, s)
      integer, intent(in) :: x
      type(Set), intent(in) :: s
      logical :: inSet
      inSet = any(s % elements(1 : s % counts) == x)
    end function
    function add(s1, s2)
      type(Set), intent(in) :: s1, s2
      type(Set) :: add
      integer :: i
      add % counts = 0
      do i = 1, s1 % counts
        if (.not.(s1 % elements(i) .in. add)) then
          add % counts = add % counts + 1
          add % elements(add % counts) = s1 % elements(i)
        end if
      end do
      do i = 1, s2 % counts
        if (.not.(s2 % elements(i) .in. add)) then
          add % counts = add % counts + 1
          add % elements(add % counts) = s2 % elements(i)
        end if
      end do
    end function
    function minus(s1, s2)
```

```
      type(Set), intent(in) :: s1, s2
      type(Set) :: minus
      integer :: i
      minus % counts = 0
      do i = 1, s1 % counts
        if (.not.((s1 % elements(i) .in. s2) .or. (s1 % elements(i) .in. minus))) then
          minus % counts = minus % counts + 1
          minus % elements(minus % counts) = s1 % elements(i)
        end if
      end do
    end function
    subroutine append(s, x)
      type(Set), intent(inout) :: s
      integer, intent(in) :: x
      if (.not.(x .in. s)) then
        s % counts = s % counts + 1
        s % elements(s % counts) = x
      end if
    end subroutine
    subroutine printSet(s)
      type(Set), intent(in) :: s
      integer :: i
      print *, (s % elements(i), i = 1, s % counts)
    end subroutine
end module
program EX0702
  use SetModule
  implicit none
  integer :: m, n, i
  integer, allocatable :: a1(:), b1(:)
  type(Set) :: a, b
  print *, 'Please input how many integer numbers(MAXIMUM: 50) in the sets a, b:'
  read *, m, n
  allocate(a1(m), b1(n))
  print *, 'Please input each number in the sets a, b:'
  read *, a1, b1
  do i = 1, m
    call append(a, a1(i))
  end do
  do i = 1, n
    call append(b, b1(i))
  end do
  deallocate(a1, b1)
  call printSet(a + b)
  call printSet(a - b)
end program
```

8 指针、格式化输入/输出、文件操作

上机目的：练习指针的使用，格式化的输入/输出，文件的操作。

8.1 格式化输入输出

8.1.1 整数

用自由格式,I2,I4,I4.2的格式操作符从键盘上读入整型数1234，然后再用自由格式,I2,I4,I4.2的格式操作符从屏幕上输出。

```
program EX080101
  implicit none
  integer :: x
  read *, x
  call output(x)
  read '(I2)', x
  call output(x)
  read '(I4)', x
  call output(x)
  read '(I4.2)', x
```

```
  call output(x)
  contains
    subroutine output(x)
      integer :: x
      print *, x
      print '(I2)', x
      print '(I4)', x
      print '(I4.2)', x
    end subroutine
end program
```

8.1.2 实数

用自由格式,F6.2,E8.2,E12.2E3,G6.2,EN10.2,ES10.2的格式操作符从键盘上读入实型数$-1.234, 0.0034567, 3.14159E01, 98.76E-2$，然后再用自由格式,F6.2,E8.2,E12.2E3,G6.2,EN10.2,ES10.2的格式操作符从屏幕上输出。

```
program EX080102
  implicit none
  real :: x
  read *, x
  call output(x)
  read '(F6.2)', x
  call output(x)
  read '(E8.2)', x
  call output(x)
  read '(E12.2E3)', x
  call output(x)
  read '(G6.2)', x
  call output(x)
  read '(EN10.2)', x
  call output(x)
  read '(ES10.2)', x
  call output(x)
  contains
    subroutine output(x)
      real :: x
      print *, x
      print '(F6.2)', x
      print '(E8.2)', x
      print '(E12.2E3)', x
      print '(G6.2)', x
      print '(EN10.2)', x
      print '(ES10.2)', x
    end subroutine
end program
```

8.1.3 复数

从屏幕上读入一个复形数$(1.23, -8.9E-02)$，然后从屏幕上用自由格式,F6.2,E8.2,实部+虚部i的形式输出。

```
program EX080103
  implicit none
  complex :: z
  read *, z
  print *, z
  print '(F6.2)', z
  print '(E8.2)', z
  print '(G14.7"+"G14.7"i")', z
end program
```

8.1.4 逻辑型

用自由格式,L,L4从键盘上读入$.TRUE., .T., .FALSE., .F.$，然后自由格式,L,L4从屏幕上输出。

```
program EX080104
  implicit none
  logical :: x
```

```
  read *, x
  call output(x)
  read '(L)', x
  call output(x)
  read '(L4)', x
  call output(x)
  contains
    subroutine output(x)
      logical :: x
      print *, x
      print '(L)', x
      print '(L4)', x
    end subroutine
end program
```

8.1.5 字符串

用自由格式,A,A3,A5从键盘上读入字符串"A","big","china","microsoft"，并用自由格式,A,A3,A5编辑符从屏幕上输出。——注意观察并理解格式输入/输出的结果。

```
program EX080105
  implicit none
  character(63) :: s
  read *, s
  call output(s)
  read '(A)', s
  call output(s)
  read '(A3)', s
  call output(s)
  read '(A5)', s
  call output(s)
  contains
    subroutine output(s)
      character(63) :: s
      print *, s
      print '(A)', s
      print '(A3)', s
      print '(A5)', s
    end subroutine
end program
```

8.2 输出金字塔形状

在屏幕上输出如下任意阶的金字塔形状。

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

```
program EX0802
  implicit none
  integer :: i, j, n
  print *, 'Please input how many rows of the pyramid:'
  read *, n
  do i = 1, n
    do j = 1, n - i
      print '(A,$)', ' '
    end do
    do j = 1, i * 2 - 1
      print '(A,$)', '*'
    end do
    print '(/,$)'
  end do
end program
```

8.3 整齐的杨辉三角形

使用整型、不带指数的实型（例如1.0）、带指数的实型（例如1.0E+00）的格式，输出杨辉三角形，要求排列成整齐的金字塔形。

$$\begin{array}{ccccccc} & & & 1 & & & \\ & & 1 & & 1 & & \\ & 1 & & 2 & & 1 & \\ 1 & & 3 & & 3 & & 1 \end{array}$$

$$\begin{array}{ccccccc} & & & 1.0 & & & \\ & & 1.0 & & 1.0 & & \\ & 1.0 & & 2.0 & & 1.0 & \\ 1.0 & & 3.0 & & 3.0 & & 1.0 \end{array}$$

$$\begin{array}{ccccccc} & & & 1.0E+00 & & & \\ & & 1.0E+00 & & 1.0E+00 & & \\ & 1.0E+00 & & 2.0E+00 & & 1.0E+00 & \\ 1.0E+00 & & 3.0E+00 & & 3.0E+00 & & 1.0E+00 \end{array}$$

```
program EX0803
  implicit none
  integer :: i, n
  integer, allocatable :: x(:, :)
  character(14) :: f
  print *, 'Please input calculate how many rows of Pascal''s Triangle:'
  read *, n
  allocate(x(n, n))
  x = 0
  x(1, 1) = 1
  do i = 2, n
    x(:, i) = (/0, x(1 : n - 1, i - 1)/) + x(:, i - 1)
  end do
  do i = 1, n
    f = "(??X,99I6)"
    write (f(2 : 3), '(I2)'), (n - i) * 3 + 1
    print f, x(1 : i, i)
  end do
  do i = 1, n
    f = "(??X,99F8.1)"
    write (f(2 : 3), '(I2)'), (n - i) * 4 + 1
    print f, real(x(1 : i, i))
  end do
  do i = 1, n
    f = "(??X,99ES10.1)"
    write (f(2 : 3), '(I2)'), (n - i) * 5 + 1
    print f, real(x(1 : i, i))
  end do
  deallocate(x)
end program
```

8.4 龙格-库塔法求解微分方程

编写一程序用四阶龙格-库塔法求解微分方程，当$x = 0.0$时，$y = 1.0$。试求出$x = 0.1, 0.2, 0.3, 0.4, \ldots, 1.0, \ldots, 100$时的y值。

提示：算法如下：

求解$y' = f(x, y(x))$，定解条件：$\begin{cases} x = x_0 \\ y = y_0 \end{cases}$。

$$\begin{cases} h = & x_{n+1} - x_n \\ k_1 = & hf(x_n, y_n) \\ k_2 = & hf(x_n + \frac{h}{2}, y_n + \frac{k_1}{2}) \\ k_3 = & hf(x_n + \frac{h}{2}, y_n + \frac{k_2}{2}) \\ k_4 = & hf(x_n + h, y_n + k_3) \\ y_{n+1} = & y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \end{cases}$$

所以

$$\begin{cases} x = x_0 \\ y = y_0 \end{cases} \Rightarrow f(x_0, y_0) \Rightarrow y1, f(x_1, y_1) \Rightarrow y2 \Rightarrow \cdots \Rightarrow y_n$$

要求：

（1）把结果按

$$x_1, y_1$$
$$x_2, y_2$$
$$x_3, y_3$$
$$\cdots$$

的格式输出到一个无格式文档data1.txt中和一个有格式文档data2.txt。注意比较两个文档占用存储空间的大小。

（2）编写另一个程序，从上述文档中提取出数据，然后找出这个函数的最大值ymax，和最大值所在的xmax，以及函数的最小值ymin，和最小值所在的xmin，从屏幕上输出。寻找最大值和最小值的算法要求用指针完成：设定一个最大值指针ymax和xmax，开始指向第一个y值和x值。然后提取下一个y值与指针指向的数值进行比较，如果更大，则令指针指向新的y值和对应的x值，如此扫描所有数据，指针指向的最后的数值，就是所求的数值。

```
program EX080401
  implicit none
  real, parameter :: DX = 0.1
  real :: x0 = 0.0, y0 = 1.0, x1
  x1 = x0 + DX
  print *, 'To solve "dy/dx=y^2-x^2" with Runge-Kutta''s method,'
  print *, 'when x=', x0, 'y=', y0, ', this is what we know.'
  print *, 'The following data are output to:'
  print *, 'the formatted file data1.txt and the unformatted file data2.txt.'
  open(11, file = 'data1.txt')
  open(12, file = 'data2.txt', form = 'unformatted')
  do while (x0 < 100.0)
    y0 = runge(f, x0, y0, x1)
    print *, 'when x=', x1, 'y=', y0
    write (11, *) x1, y0
    write (12) x1, y0
    x0 = x1
    x1 = x1 + DX
  end do
  close(11)
  close(12)
  contains
    function runge(f, x0, y0, x1)
      implicit none
      real :: f, x0, y0, x1, runge, h, k1, k2, k3, k4
      h = x1 - x0
      k1 = h * f(x0, y0)
      k2 = h * f(x0 + h / 2.0, y0 + k1 / 2.0)
      k3 = h * f(x0 + h / 2.0, y0 + k2 / 2.0)
      k4 = h * f(x0 + h, y0 + k3)
      runge = y0 + (k1 + 2.0 * k2 + 2.0 * k3 + k4) / 6.0
    end function
    function f(x, y)
      implicit none
```

```
      real :: x, y, f
      f = y ** 2 - x ** 2
    end function
end program
```

```
program EX080402
  implicit none
  real :: tmp(2), tmp2
  integer :: lineCount, i
  real, target, allocatable :: datafile(:, :)
  real, pointer :: xmax, ymax
  lineCount = 0
  open(11, file='data1.txt')
  do while (.true.)
    read (11, *, end = 1001) tmp
    lineCount = lineCount + 1
  end do
  1001 continue
  close(11)
  allocate(datafile(2, lineCount))
  open(11, file='data1.txt')
  read (11, *) datafile
  close(11)
  xmax => datafile(1, 1)
  ymax => datafile(2, 1)
  do i = 2, lineCount
    if (ymax < datafile(2, i)) then
      xmax => datafile(1, i)
      ymax => datafile(2, i)
    end if
  end do
  print *, xmax, ymax
  deallocate(datafile)
  lineCount = 0
  open(11, file='data2.txt', form='unformatted')
  do while (.true.)
    read (11, end = 1002) tmp
    lineCount = lineCount + 1
  end do
  1002 continue
  close(11)
  allocate(datafile(2, lineCount))
  open(11, file='data2.txt', form='unformatted')
  do i = 2, lineCount
    read (11) datafile(:, i)
  end do
  close(11)
  xmax => datafile(1, 1)
  ymax => datafile(2, 1)
  do i = 2, lineCount
    if (ymax < datafile(2, i)) then
      xmax => datafile(1, i)
      ymax => datafile(2, i)
    end if
  end do
  print *, xmax, ymax
  deallocate(datafile)
end program
```

8.5 Shell排序

【选做题，有剩余时间的同学可自己完成】Shell法排序。

要求：用指针完成。

提示：Shell法排序由发明者D.L. Shell的名字命名，这种方法有较快的排序速度。

假定要将数组$\{a_n\}$的数据由小到大排序，步骤如下：

（1）首先任意选定进行比较的两个元素的距离h，把a_i与a_{i+h}比较，若$a_i > a_{i+h}$则把这两个元素中数据进行对换，把小的放在前面，把大的放在后面。为了叙述方便，把这一操作步骤简称为一次“比较对调”，h值称为比较对调的“间距”。

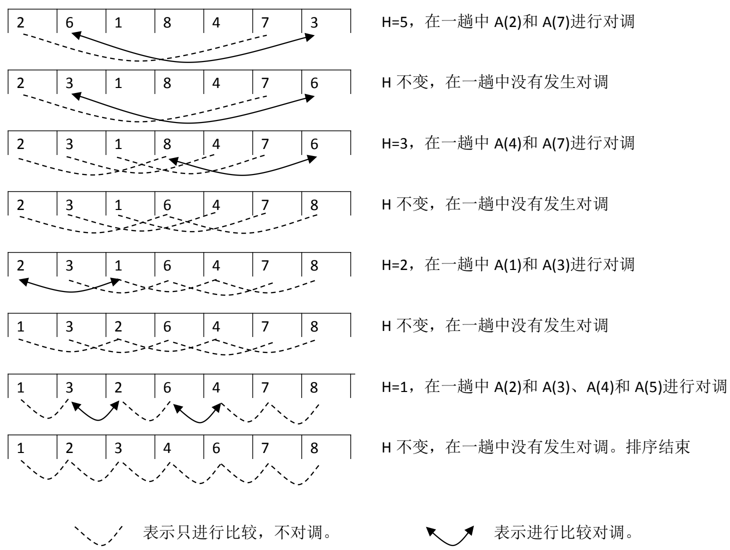
（2）如果数字共有n个元素，则需要把i值从1变化到$n-h$，对每个i值进行一次“比较对调”。而这一过程，简称为“一趟比较对调”。

（3）如果在一趟比较对调中有对调发生（哪怕只有一次），则保持间距h值不变，重复进行一趟比较对调，直到没有任何对调发生。此时才能改变h的值。

（4）把间距h的值减小，重复过程（1）～（3）。

（5）继续减小间距h的值，直到$h=1$且在此间距下进行一趟比较对调的时候没有对调发生。至此，排序完成。

图示如下：



通常，h的初始值取为$\left[\frac{n}{2}\right]$，而新一轮的h值取上一轮的$\left[\frac{h}{2}\right]$是最快的排序方法。

```
program EX0805
  implicit none
  integer :: n, h, i, c, tmp
  integer, pointer :: a1, a2
  integer, target, allocatable :: a(:)
  print *, 'Please input how many integer numbers to do shell sort:'
  read *, n
  allocate(a(n))
  print *, 'Please input each number:'
  read *, a
  h = n / 2
  do while (h >= 1)
    1001 c = 0
    do i = 1, n - h
      a1 => a(i)
      a2 => a(i + h)
      if (a1 > a2) then
        tmp = a1
        a1 = a2
        a2 = tmp
        c = c + 1
      end if
    end do
    if (c > 0) goto 1001
    h = h / 2
  end do
  print *, a
  deallocate(a)
end program
```

9 参考文献

[1]工程分析程序设计上机作业Fortran部分.doc

[2]魏进家,陈斌,周屈兰,刘小民,等.工程分析程序设计[M].西安：西安交通大学出版社,2015.1

[3]丁泽军,李会民,等.Fortran77和90/95编程入门(http://micro.ustc.edu.cn/Fortran/ZJDing/),2001.10

10 LICENSE