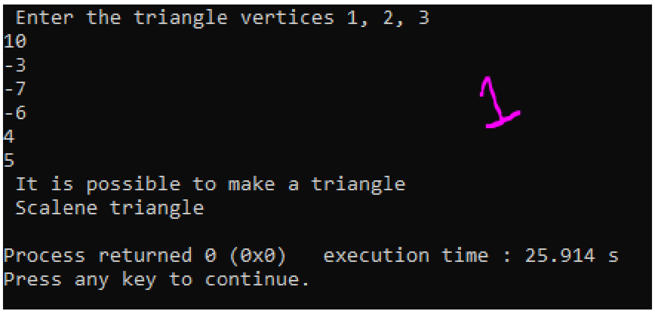
Problems Solved by JOY

▼ 1. Write a program to check whether a triangle is Equilateral, Isosceles or Scalene. (Vertices given)

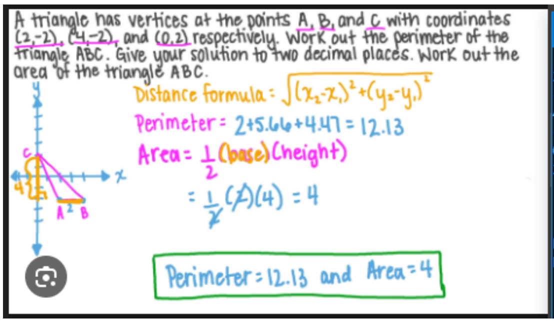
```
program number_1
    implicit none
    real:: a, b, c,  x1, y1, x2, y2, x3, y3
    print*, "Enter the triangle vertices 1, 2, 3"
    read*, x1, y1, x2, y2, x3, y3

    !it's for calculate the edge of triangles
    a=sqrt((x2 - x1)**2 + (y2 - y1)**2)
    b=sqrt((x3 - x1)**2 + (y3 - y1)**2)
    c=sqrt((x3 - x2)**2 + (y3 - y2)**2)

    if (a<b+c .and. b<a+c .and. c<a+b) then
        print*, "It is possible to make a triangle"
    ! Check the type of triangle
      if (a == b .and. b == c) then
         print*,"Equilateral triangle"
      else if (a == b .or. a == c .or. b == c) then
          print*,"Isosceles triangle"
      else
       print*,"Scalene triangle"
      end if
    else
        print*, "It is not a triangle"
    end if
end program
```



```
 Enter the triangle vertices 1, 2, 3
10
-3
-7
-6
4
5
 It is possible to make a triangle
 Scalene triangle

Process returned 0 (0x0)   execution time : 25.914 s
Press any key to continue.
```

▼ 2. Print the area and perimeter of a tringle if the vertices of the tringle are given.

```
program num_2
    implicit none
    real:: x1, y1, x2, y2, x3, y3, a, b, c, s,  area, perimeter
    print*, "Enter the triangle vertices 1, 2, 3"
    read*, x1, y1, x2, y2, x3, y3

    !it's for calculate the edge of triangles
    a=sqrt((x2 - x1)**2 + (y2 - y1)**2)
    b=sqrt((x3 - x1)**2 + (y3 - y1)**2)
    c=sqrt((x3 - x2)**2 + (y3 - y2)**2)
    s=(a+b+c)/2

    !check the validity of triangle
    if (a<b+c .and. b<c+a .and. c<a+b) then
        perimeter = (a+b+c)
        area = sqrt(s*(s-a)*(s-b)*(s-c))
        print*, "The perimeter of triangle is", perimeter
        print*, "The area of triangle is", area
    else
        print*, "It is not a triangle"
    end if
end program
```



A triangle has vertices at the points A, B, and C with coordinates (2,-2), (4,-2), and (0,2) respectively. Work out the perimeter of the triangle ABC. Give your solution to two decimal places. Work out the area of the triangle ABC.

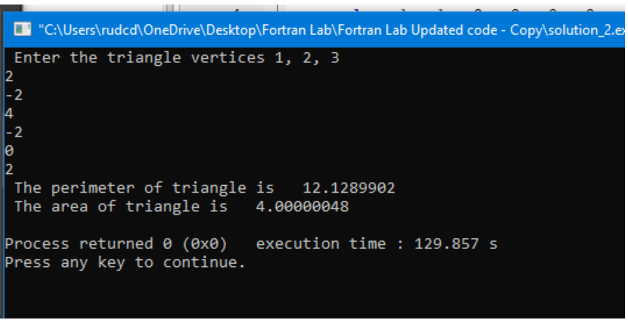
Distance formula: $\sqrt{(x_2-x_1)^2+(y_2-y_1)}^2$

Perimeter: $2+5.66+4.47=12.13$

Area: $\frac{1}{2}$(base)(height)

$= \frac{1}{\cancel{2}}(\cancel{2})(4) = 4$

Perimeter = 12.13 and Area = 4



"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_2.ex

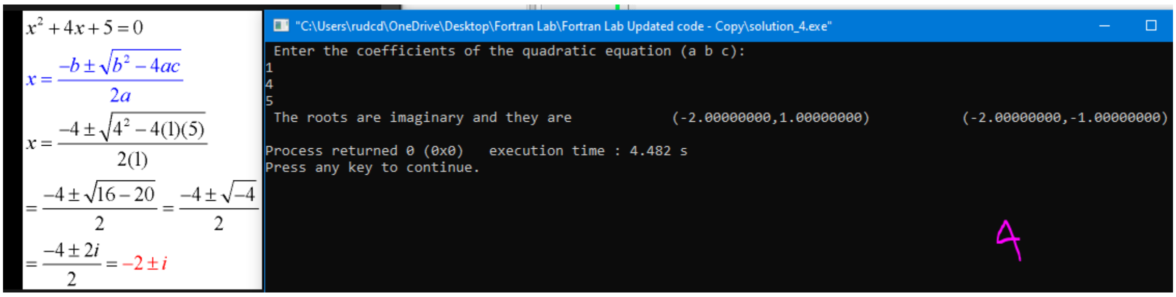
```
 Enter the triangle vertices 1, 2, 3
2
-2
4
-2
0
2
 The perimeter of triangle is   12.1289902
 The area of triangle is    4.00000048

Process returned 0 (0x0)   execution time : 129.857 s
Press any key to continue.
```

▼ 3. Find the center and radius of a tringle's circum circle if the tringle's vertices are given.

```
program num_3
  implicit none
  real :: x1, y1, x2, y2, x3, y3, a, b, c, a1, a2, b1, b2, c1, c2, x, y, r

  ! Input vertices of the triangle
  print *, "Enter the 1st vertices of the triangle (x1 y1):"
  read *, x1, y1
  print *, "Enter the 2nd vertices of the triangle (x2 y2):"
  read *, x2, y2
  print *, "Enter the 3rd vertices of the triangle (x3 y3):"
  read *, x3, y3

  ! Calculate the side lengths
  a = sqrt((x1 - x2)**2 + (y1 - y2)**2)
  b = sqrt((x2 - x3)**2 + (y2 - y3)**2)
  c = sqrt((x3 - x1)**2 + (y3 - y1)**2)

  ! Check if a triangle with the given vertices exists
  if (a + b > c .and. b + c > a .and. c + a > b) then
    a1 = 2.0 * (x2 - x1)
    a2 = 2.0 * (x2 - x3)
    b1 = 2.0 * (y2 - y1)
    b2 = 2.0 * (y2 - y3)
    c1 = x1**2 - x2**2 + y1**2 - y2**2
    c2 = x3**2 - x2**2 + y3**2 - y2**2

    ! Calculate the coordinates of the circumcenter
    x = (b1 * c2 - b2 * c1) / (a1 * b2 - a2 * b1)
    y = (c1 * a2 - c2 * a1) / (a1 * b2 - a2 * b1)

    ! Calculate the radius
    r = sqrt((x - x1)**2 + (y - y1)**2)

    ! Output the results
    print *, "Center of the circumcircle is", x,y
    print *, "Radius of the circumcircle is", r
  else
    print *, "Triangle does not exist for those points."
  end if
end program num_3
```

▼ 4. Solve a quadratic equation and print its roots (roots maybe real or complex).

```
program num_4
  implicit none
  real :: a, b, c, d, r1, r2, realPart, imagPart
  complex :: cr1, cr2

  ! Input coefficients
  print *, "Enter the coefficients of the quadratic equation (a b c):"
  read *, a, b, c

  ! Calculate the discriminant
  d = b**2 - 4.0*a*c

  ! Check the discriminant to determine the roots
  if (d > 0.0) then
    r1 = (-b + sqrt(d)) / (2.0*a)
    r2 = (-b - sqrt(d)) / (2.0*a)
    print *, "The roots are real and unequal. Hence, The roots are:", r1, r2
  else if (d == 0.0) then
    r1 = -b / (2.0*a)
    print *, "The roots are real and equal. Hence, The root is:", r1
  else
    realPart = -b / (2.0*a)
    imagPart = sqrt(-d) / (2.0*a)
    cr1 = cmplx(realPart, imagPart)
    cr2 = cmplx(realPart, -imagPart)
    print *, "The roots are imaginary and they are", cr1, cr2
  end if
end program num_4
```



$$x^2+4x+5=0$$

$$x=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

$$x=\frac{-4\pm\sqrt{4^2-4(1)(5)}}{2(1)}$$

$$=\frac{-4\pm\sqrt{16-20}}{2}=\frac{-4\pm\sqrt{-4}}{2}$$

$$=\frac{-4\pm 2i}{2}=-2\pm i$$

"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_4.exe"

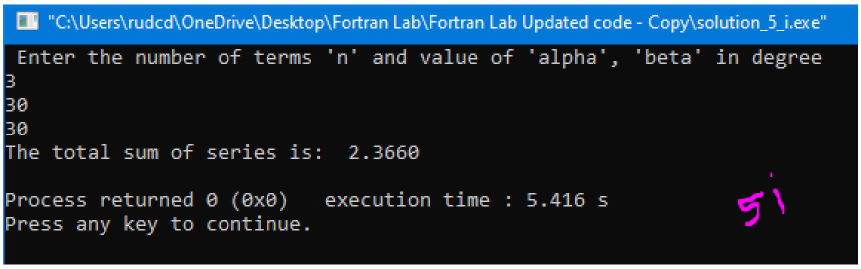
```
 Enter the coefficients of the quadratic equation (a b c):
1
4
5
 The roots are imaginary and they are            (-2.00000000,1.00000000)            (-2.00000000,-1.00000000)

Process returned 0 (0x0)   execution time : 4.482 s
Press any key to continue.
```

4

Print the sum or product up to n terms of the series:

▼ 5i. sinα + sin(α+β) + sin(α+2β) + … …

```
program num_5_i
    implicit none
    integer:: n, i
    real:: alpha, beta, sum
    real, parameter::pi=3.1416
    print*, "Enter the number of terms 'n' and value of 'alpha', 'beta'"
    read*, n, alpha, beta
    alpha = (pi/180)*alpha !Degree to Radian
    beta  = (pi/180)*beta  !Degree to Radian
    sum=0
    do i=1,n
        sum=sum+sin(alpha+(i-1)*beta)
    end do
    print'(A,F8.4)',"The total sum of series is:", sum
end program
```



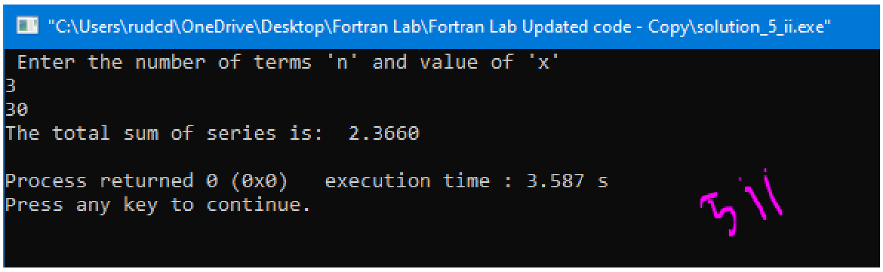
"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_5_i.exe"

```
 Enter the number of terms 'n' and value of 'alpha', 'beta' in degree
3
30
30
The total sum of series is:  2.3660

Process returned 0 (0x0)   execution time : 5.416 s
Press any key to continue.
```

5i

5ii. 1 + cos x + cos 2x + cos 3x + … …

```
program num_5_ii
    implicit none
    integer:: n, i
    real:: x, sum
    real, parameter::pi=3.1416
    print*, "Enter the number of terms 'n' and value of 'x'"
    read*, n, x
    x = (pi/180)*x !Degree to Radian
    sum=0
    do i=1,n
        sum=sum+cos((i-1)*x)
    end do
    print'(A,F8.4)',"The total sum of series is:", sum
end program
```



"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_5_ii.exe"

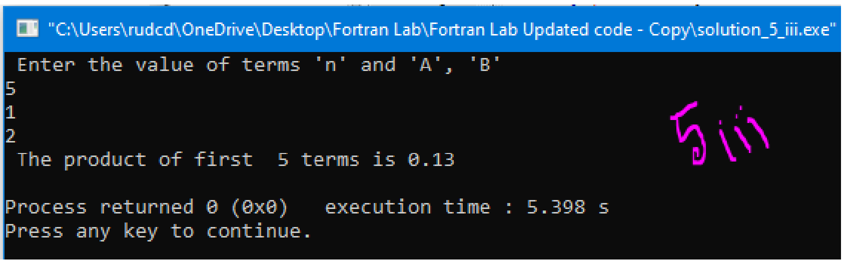
```
 Enter the number of terms 'n' and value of 'x'
3
30
The total sum of series is:  2.3660

Process returned 0 (0x0)   execution time : 3.587 s
Press any key to continue.
```

5ii

5iii.

$$\text{iii.}\quad \frac{1}{A}\cdot\frac{2}{A+B}\cdot\frac{3}{A+2B}\;\cdots\;\cdots\;\cdots$$

```
program num_5_iii
  implicit none
  integer :: n, i
  real :: a, b, proDuct
  print *, "Enter the value of terms 'n' and 'A', 'B'"
  read *, n, a, b
  proDuct = 1.0
  do i = 1, n
    proDuct = proDuct * (i / (a*1.0 + (i-1)*b*1.0))
  end do
  print 10, n, proDuct
  10 format(1x, "The product of first", I3, " terms is", f5.2)
  !print *, "The product of first", n, "terms is", proDuct
end program
```

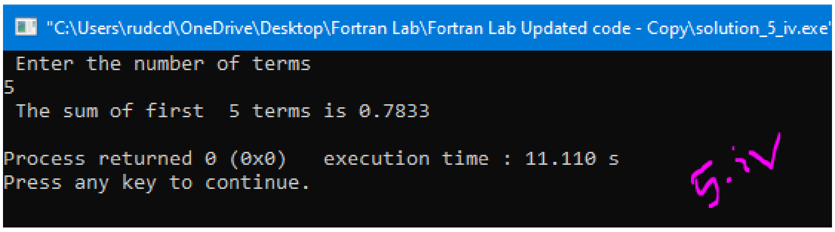

"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_5_iii.exe"
5 iii

```
 Enter the value of terms 'n' and 'A', 'B'
5
1
2
 The product of first  5 terms is 0.13

Process returned 0 (0x0)   execution time : 5.398 s
Press any key to continue.
```

▼ 5iv.

$$\text{iv.}\quad 1-\frac{1}{2}+\frac{1}{3}-\frac{1}{4}+\dots\ \dots\ \dots$$

```
program num_5_iv
    implicit none
    integer:: n, i
    real:: sum
    print*,"Enter the number of terms"
    read*, n
    sum = 0
    do i=1, n
        sum = sum+(((-1)**(i+1))/(i*1.0))
    end do
      print 10, n, sum
    10 format(1x, "The sum of first",I3, " terms is", f7.4)
  !print *, "The sum of first", n, "terms is", sum
end program
```

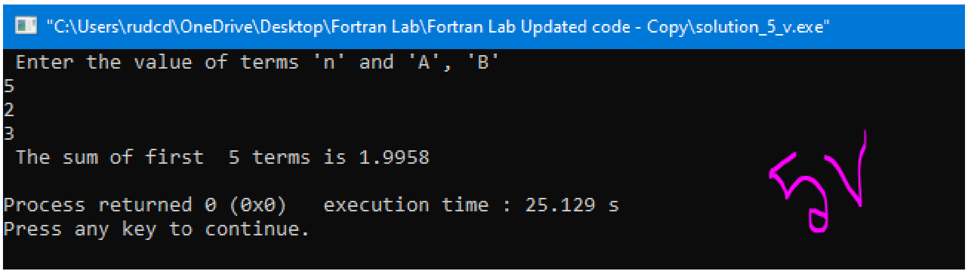

"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_5_iv.exe"
5.iv

```
 Enter the number of terms
5
 The sum of first  5 terms is 0.7833

Process returned 0 (0x0)   execution time : 11.110 s
Press any key to continue.
```

▼ 5v.

$$\text{v.}\quad \frac{1}{A}+\frac{2}{A+B}+\frac{3}{A+2B}+\dots\ \dots\ \dots$$

```
program num_5_v
  implicit none
  integer :: n, i
  real :: a, b, sum
  print *, "Enter the value of terms 'n' and 'A', 'B'"
  read *, n, a, b
  sum = 0
  do i = 1, n
    sum = sum + (i /(a+(i-1)*b))
  end do
  print 10, n, sum
  10 format(1x, "The sum of first", I3, " terms is", f7.4)
  !print *, "The sum of first", n, "terms is", sum
end program
```


"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_5_v.exe"

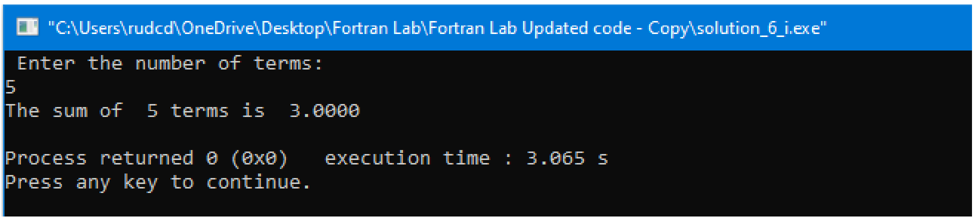
```
 Enter the value of terms 'n' and 'A', 'B'
5
2
3
 The sum of first  5 terms is 1.9958

Process returned 0 (0x0)   execution time : 25.129 s
Press any key to continue.
```

Evaluate the series $\sum_{n=1}^{N} a(n)$, where

▼ 6i.

i. $a(n)=\begin{cases} \frac{1}{n}\ when\ n\ is\ odd \\ \frac{n}{n+1}\ when\ n\ is\ even \end{cases}$

```
program num_6_i
    implicit none
    integer:: n, i
    real:: sum
    print*, "Enter the number of terms:"
    read*,n
    sum=0
    do i=1,n
        if (mod(i,2)==1) then
           sum=sum+(1.0/i)
        else
            sum=sum+(i/(i+1.0))
        end if
    end do
    print'(A,I3,A,F8.4)',"The sum of" ,n, " terms is", sum
end program
```

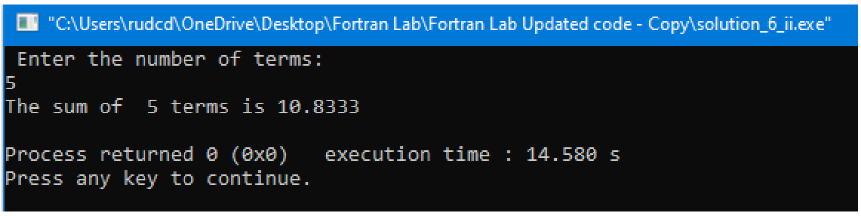

"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_6_i.exe"

```
 Enter the number of terms:
5
The sum of  5 terms is   3.0000

Process returned 0 (0x0)   execution time : 3.065 s
Press any key to continue.
```

▼ 6ii

ii. $$a(n)=\begin{cases} n \; when \; n \le 4 \\ \frac{n}{n+1} \; when \; n > 4 \end{cases}$$

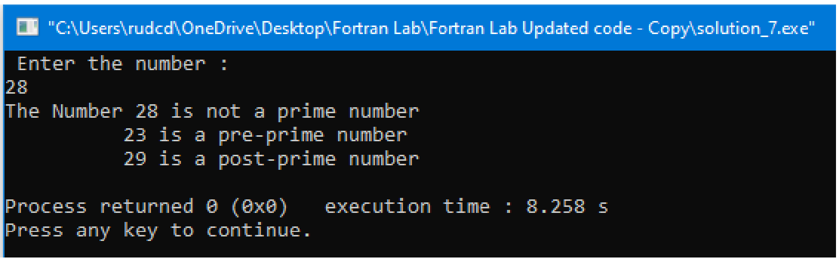
```
program num_6_ii
    implicit none
    integer::n, i
    real:: sum
    print*,"Enter the number of terms:"
    read*,n
    sum=0
    do i=1,n
        if(i<=4) then
         sum=sum+i
        else
         sum=sum+(i/(i+1.0))
        end if
    end do
     print'(A,I3,A,F8.4)',"The sum of" ,n, " terms is", sum
end program
```


"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_6_ii.exe"

```
 Enter the number of terms:
5
The sum of  5 terms is 10.8333

Process returned 0 (0x0)   execution time : 14.580 s
Press any key to continue.
```

▼ 7. Write a program to print a given number whether it is prime or not and print its pre and post prime numbers.

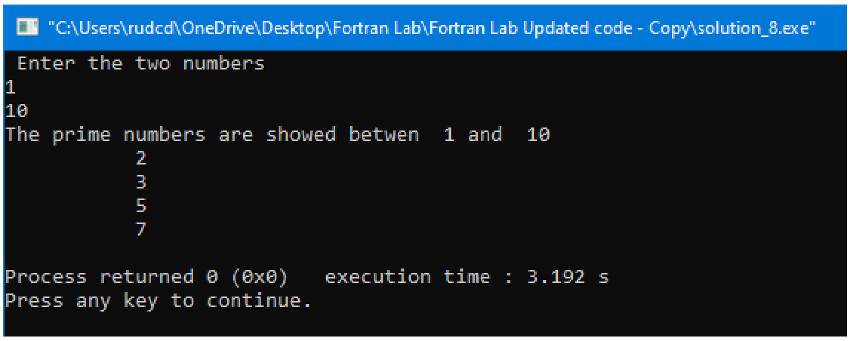
```
program num_7
    implicit none
    integer:: n,j, isprime
    print*,"Enter the number :"
    read*,n
    !--------prime---------------
    if(isprime(n)==1)then
        print'(A, I3, A)',"The Number", n," is a prime number"
        !print*,n,"is a prime number"
    else
        print'(A, I3, A)',"The Number", n," is not a prime number"
        !print*,x,"is not a prime number"
   end if
   !-----------pre prime-----------
   if(n<=2) then
        print*,"There is no pre-prime number of",n
    else
      do j=n-1,2,-1
        if(isprime(j)==1)then
        print*,j,"is a pre-prime number"
        exit
        end if
      end do
   end if
   !--------post prime------------
   do j=n+1, n+5000, 1
        if(isprime(j)==1)then
        print*,j,"is a post-prime number"
        exit
        end if
    end do
end program

!custom function to find prime
integer function isprime(x)
implicit none
integer::i,x,check
check=1
do i=2,x/2,1
    if(mod(x,i)==0)then
    check=0
    exit
    end if
end do
isprime=check
end function
```


"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_7.exe"

```
 Enter the number :
28
The Number 28 is not a prime number
          23 is a pre-prime number
          29 is a post-prime number

Process returned 0 (0x0)   execution time : 8.258 s
Press any key to continue.
```

▼ 8. Write a program to print the number of primes in a range of positive integers.

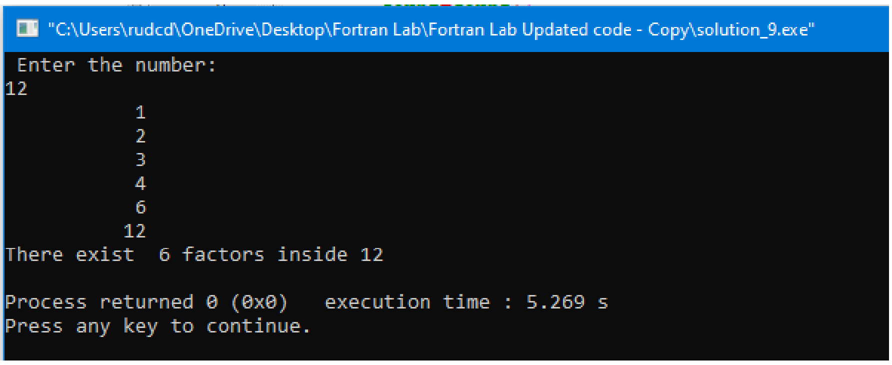
```
program num_8
    implicit none
    integer::a,b,lowest,highest,isprime,j
    print*,"Enter the two numbers"
    read*,a,b
    lowest=min(a,b)
    highest=max(a,b)
    print'(A,I3,A,I3)',"The prime numbers are showed betwen",lowest ," and" ,highest
    do j=lowest,highest,1
        if(isprime(j)==1)then
           if(j>=2)then
             print*,j
           end if
        end if
    end do
end program

!custom function to find prime
integer function isprime(x)
implicit none
integer::i,x,check
check=1
do i=2,x/2,1
    if(mod(x,i)==0)then
    check=0
    exit
    end if
end do
isprime=check
end function
```


"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_8.exe"

```
 Enter the two numbers
1
10
The prime numbers are showed betwen  1 and  10
           2
           3
           5
           7

Process returned 0 (0x0)   execution time : 3.192 s
Press any key to continue.
```

▼ 9. Write a program to read a positive integer n and print the factors of n and count them.

```
program num_9
implicit none
integer:: i, n, count
print*, "Enter the number:"
read*,n
count=0
do i=1,n
    if(mod(n,i)==0) then
        count=count+1
        print*,i
    end if
end do
print'(A,I3,A,I3)',"There exist", count," factors inside",n
end program
```

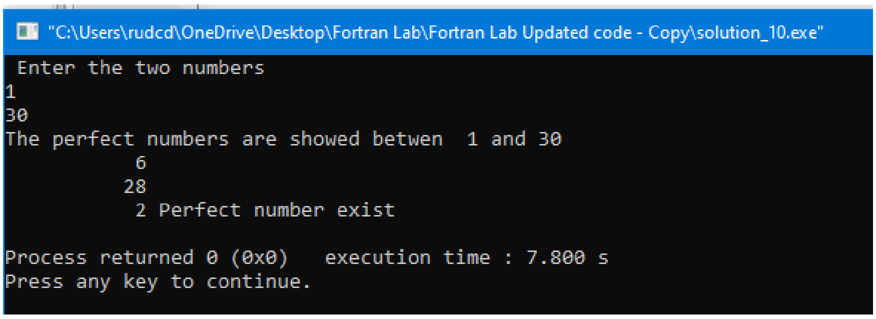

"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_9.exe"

```
 Enter the number:
12
           1
           2
           3
           4
           6
          12
There exist  6 factors inside 12

Process returned 0 (0x0)   execution time : 5.269 s
Press any key to continue.
```

▼ 10. Write a program to print the perfect numbers in a range of positive integers.

```
program num_10
    implicit none
    integer::a,b,lowest,highest,total,i,j,count
    print*,"Enter the two numbers"
    read*,a,b
    lowest=min(a,b)
    highest=max(a,b)
       count=0
     print'(A,I3,A,I3)',"The perfect numbers are showed betwen",lowest ," and" ,highest
    do j=lowest,highest,1
        total=0
        do i=1, j/2,1
            if(mod(j,i)==0) then
```

```
                total=total+i
            end if
        end do
        if(j==total) then
            print*,j
            count=count+1
        end if
    end do
    print*,count,"Perfect number exist"
end program
```

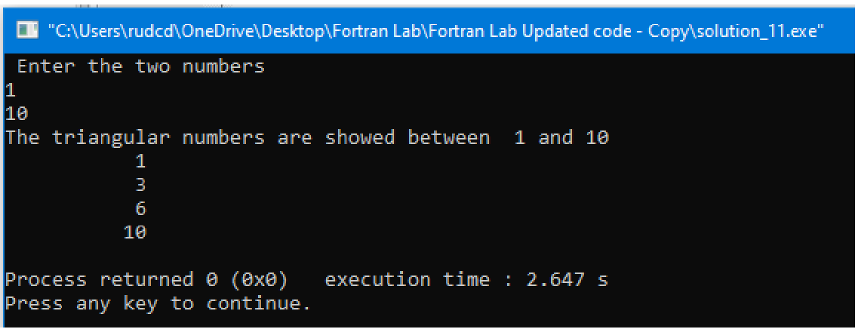


"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_10.exe"

```
 Enter the two numbers
1
30
The perfect numbers are showed betwen  1 and 30
           6
          28
           2 Perfect number exist

Process returned 0 (0x0)   execution time : 7.800 s
Press any key to continue.
```

▼ 11. Write a program to print the triangular numbers in a range.

```
program num_11
    implicit none
    integer::a,b,lowest,highest,sum,i,j
    print*,"Enter the two numbers"
    read*,a,b
    lowest=min(a,b)
    highest=max(a,b)
    print'(A,I3,A,I3)',"The triangular numbers are showed between",lowest ," and" ,highest
    do j=lowest,highest,1
        sum=0
        do i=1,j
          sum=sum+i
          if(sum==j) then
            print*,j
            exit
          end if
        end do
    end do
end program
```

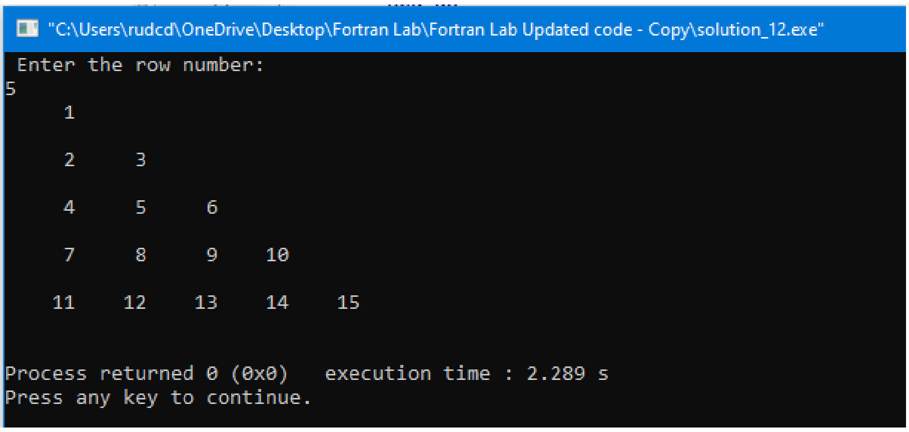


"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_11.exe"

```
 Enter the two numbers
1
10
The triangular numbers are showed between  1 and 10
           1
           3
           6
          10

Process returned 0 (0x0)   execution time : 2.647 s
Press any key to continue.
```

▼ 12. Write a program to print the Floyd's triangle.

```
program num_12
    implicit none
    integer::i,n,j,sum
    print*,"Enter the row number:"
    read*,n
    sum=0
    do j=1,n
        do i=1,j
         sum=sum+1  !next row te koyta element hobe te jante
        write(*,"(I6)", advance="no")sum
        end do
        write(*,"(/)")
        !print*," "
    end do
end program
```

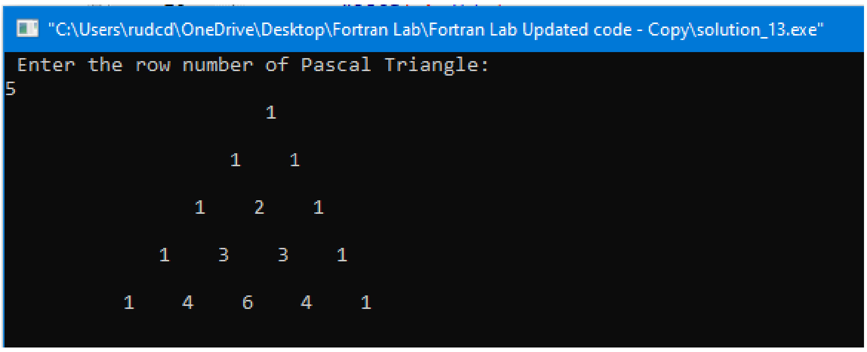


"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_12.exe"

```
 Enter the row number:
5
     1

     2     3

     4     5     6

     7     8     9    10

    11    12    13    14    15


Process returned 0 (0x0)   execution time : 2.289 s
Press any key to continue.
```

▼ 13. Write a program to print the Pascal’s triangle.

```
program num_12
    implicit none
    integer::i,n,j,sum
    print*,"Enter the row number:"
    read*,n
    sum=0
    do j=1,n
        do i=1,j
         sum=sum+1  !next row te koyta element hobe te jante
        write(*,"(I6)", advance="no")sum
        end do
        write(*,"(/)")
        !print*," "
    end do
end program
```



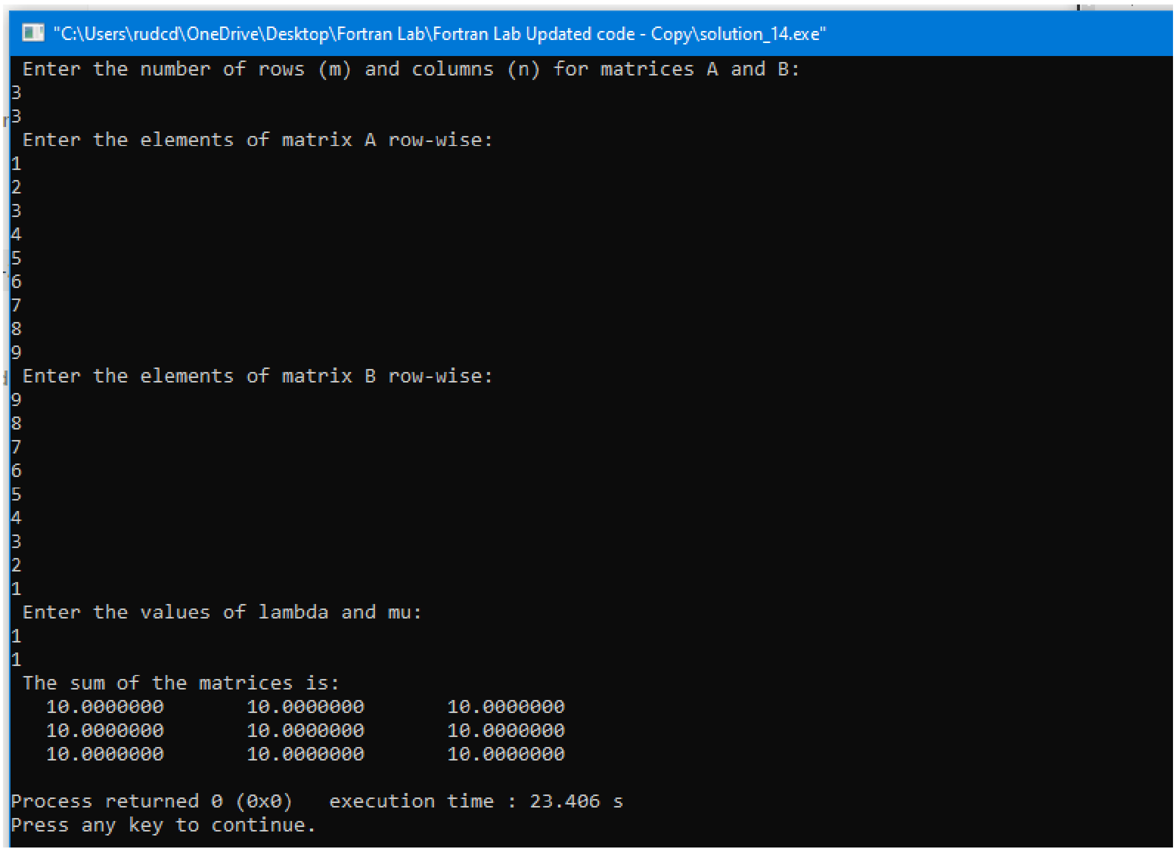
"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_13.exe"

```
 Enter the row number of Pascal Triangle:
5
                     1

                  1     1

               1     2     1

            1     3     3     1

         1     4     6     4     1
```

▼ 14.

Read matrices A_{mxn} and B_{mxn} and print the matrix $\lambda A+\mu B$.

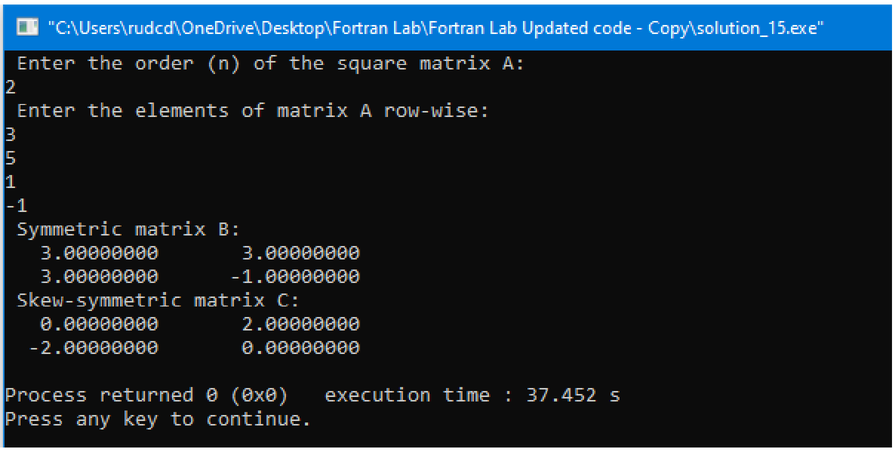
```
program num_14
  implicit none
  integer :: i, j, m, n
  real :: lambda, mu
  real, allocatable :: a(:,:), b(:,:), c(:,:)

  ! Step 1: Input the dimensions of matrices A and B
  print *, "Enter the number of rows (m) and columns (n) for matrices A and B:"
  read *, m, n

  ! Step 2: Allocate memory for matrices A, B, and C
  allocate(a(m, n), b(m, n), c(m, n))

  ! Step 3: Input elements for matrix A
  print *, "Enter the elements of matrix A row-wise:"
  do i = 1, m
    read *, (a(i, j), j = 1, n)
  end do
    !read *, ((a(i, j), j = 1, n), i=1, m)

  ! Step 3: Input elements for matrix B
  print *, "Enter the elements of matrix B row-wise:"
  do i = 1, m
    read *, (b(i, j), j = 1, n)
  end do
  !read *, ((b(i, j), j = 1, n), i=1, m)

  ! Step 4: Input values for lambda and mu
  print *, "Enter the values of lambda and mu:"
  read *, lambda, mu

  ! Step 5: Calculate the sum of matrices A and B scaled by lambda and mu
  do i = 1, m
    do j = 1, n
      c(i, j) = lambda * a(i, j) + mu * b(i, j)
    end do
  end do

  ! Step 6: Print the resulting matrix C
  print *, "The sum of the matrices is:"
  do i = 1, m
    print *, (c(i, j), j = 1, n)
  end do
end program num_14
```


"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_14.exe"

```
 Enter the number of rows (m) and columns (n) for matrices A and B:
3
3
 Enter the elements of matrix A row-wise:
1
2
3
4
5
6
7
8
9
 Enter the elements of matrix B row-wise:
9
8
7
6
5
4
3
2
1
 Enter the values of lambda and mu:
1
1
 The sum of the matrices is:
   10.0000000       10.0000000       10.0000000
   10.0000000       10.0000000       10.0000000
   10.0000000       10.0000000       10.0000000

Process returned 0 (0x0)   execution time : 23.406 s
Press any key to continue.
```

▼ 15. Read a square matrix A of order n and print a symmetric matrix B and skew-symmetric matrix C such that A=B+C.

```
program num_15
  implicit none
  integer :: n, i, j
  real, allocatable :: A(:,:), B(:,:), C(:,:)

  ! Step 1: Input the order of the square matrix A
  print *, "Enter the order (n) of the square matrix A:"
  read *, n

  ! Step 2: Allocate memory for matrices A, B, and C
  allocate(A(n, n), B(n, n), C(n, n))

  ! Step 3: Input elements for matrix A
  print *, "Enter the elements of matrix A row-wise:"
  do i = 1, n
    read *, (A(i, j), j = 1, n)
  end do
  !read *, ((A(i, j), j = 1, n),i=1,n)

  ! Step 4: Calculate matrices B (symmetric) and C (skew-symmetric)
  do i = 1, n
    do j = 1, n
      B(i, j) = 0.5 * (A(i, j) + A(j, i))  ! Symmetric part
      C(i, j) = 0.5 * (A(i, j) - A(j, i))  ! Skew-symmetric part
    end do
  end do

  ! Step 5: Print matrices B and C
  print *, "Symmetric matrix B:"
  do i = 1, n
    print *, (B(i, j), j = 1, n)
  end do
  print *, "Skew-symmetric matrix C:"
  do i = 1, n
    print*, (C(i, j), j = 1, n)
  end do

end program num_15
```

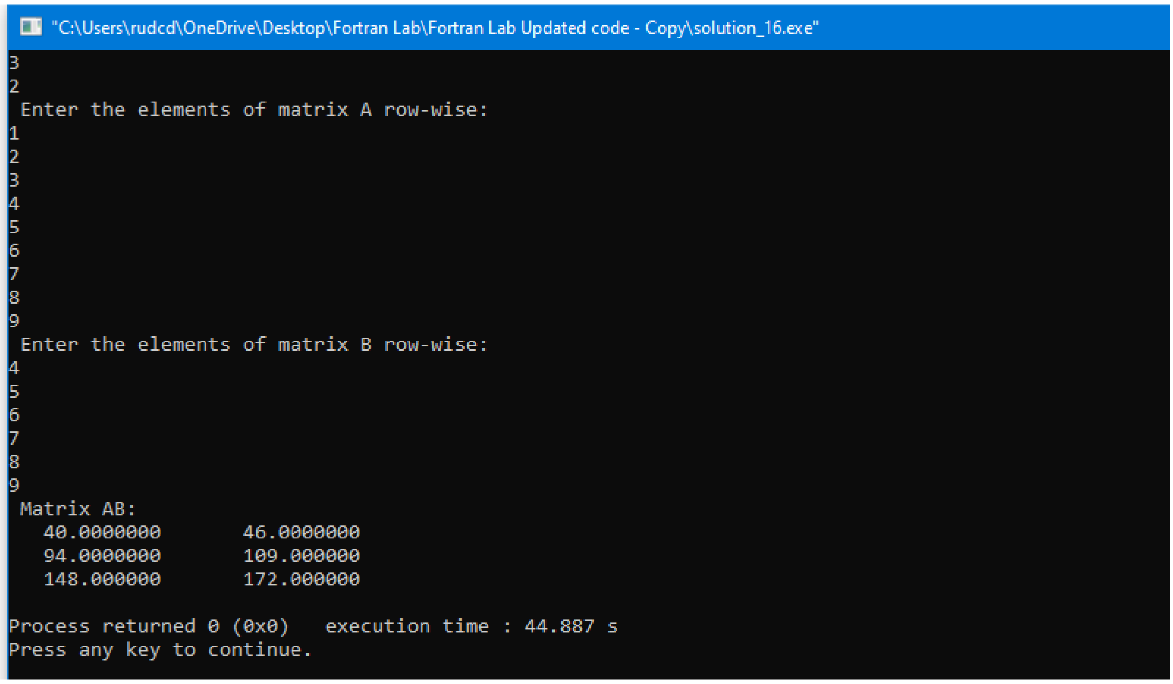


"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_15.exe"

```
 Enter the order (n) of the square matrix A:
2
 Enter the elements of matrix A row-wise:
3
5
1
-1
 Symmetric matrix B:
   3.00000000       3.00000000
   3.00000000      -1.00000000
 Skew-symmetric matrix C:
   0.00000000       2.00000000
  -2.00000000       0.00000000

Process returned 0 (0x0)   execution time : 37.452 s
Press any key to continue.
```

▼ 16. Read two matrices A and B and test whether AB defined if AB exists print the product AB.

```
program num_16
  implicit none
  integer :: i, j, k, m, n, p, q
  real, allocatable :: a(:,:), b(:,:), c(:,:)

  ! Step 1: Input dimensions for matrix A
  print *, "Enter the number of rows (m) and columns (n) for matrix A:"
  read *, m,n

  ! Step 2: Input dimensions for matrix B
   print *, "Enter the number of rows (p) and columns (q) for matrix B:"
  read *, p,q

  ! Step 3: Check if matrix multiplication is possible
  if (n == p) then
    ! Step 4: Allocate memory for matrices A, B, and C
    allocate(a(m, n), b(n, q), c(m, q))

    ! Step 5: Input elements for matrix A
    print *, "Enter the elements of matrix A row-wise:"
    do i = 1, m
      read *, (a(i, j), j = 1, n)
    end do
    ! read *, ((a(i, j), j = 1, n),i=1,m)

    ! Step 6: Input elements for matrix B
    print *, "Enter the elements of matrix B row-wise:"
    do i = 1, p
      read *, (b(i, j), j = 1, q)
    end do
    !read *, (a(i, j), j = 1, q)),i=1,p)

    ! Step 7: Calculate the product C
    do i = 1, m     !iterates over the rows
      do j = 1, q   !iterates over the columns
        c(i, j) = 0.0
        do k = 1, n     ! iterates over the columns of matrix A and the rows of matrix B
          c(i, j) = c(i, j) + a(i, k) * b(k, j)
        end do
      end do
    end do
    ! Step 8: Print the product matrix C
    print *, "Matrix AB:"
    do i = 1, m
      print *, (c(i, j), j = 1, q)
    end do

    ! Step 9: Deallocate memory to free up resources (optional error)
    deallocate(a, b, c)

  else
    ! Step 10: Notify if multiplication is not possible
    print *, "Product of A and B is not possible. Number of columns in A should be equal to the number of rows in B."
  end if
end program num_16
```

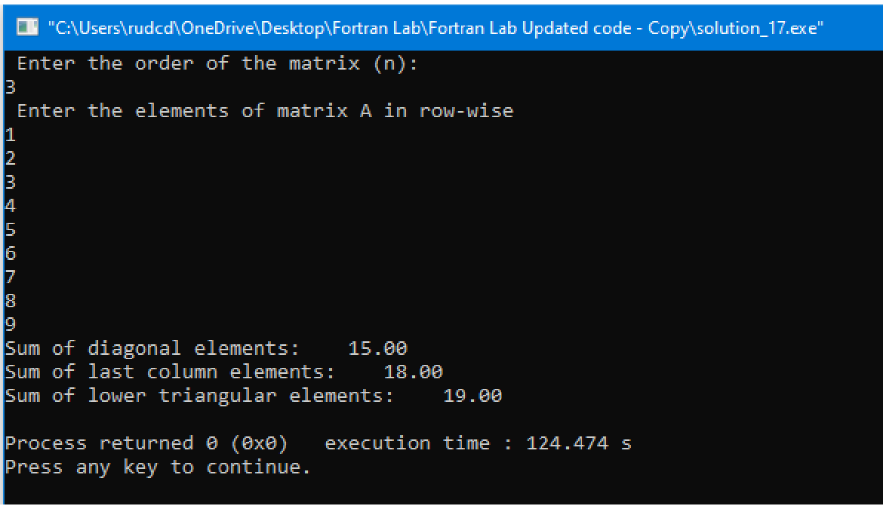


"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_16.exe"

```
3
2
 Enter the elements of matrix A row-wise:
1
2
3
4
5
6
7
8
9
 Enter the elements of matrix B row-wise:
4
5
6
7
8
9
 Matrix AB:
   40.0000000       46.0000000
   94.0000000       109.000000
   148.000000       172.000000

Process returned 0 (0x0)   execution time : 44.887 s
Press any key to continue.
```

▼ 17. Read a matrix A of order n and print the sum of:
i. diagonal elements
ii. the last column elements
iii. the lower triangular elements

```
program num_17
  implicit none
  integer :: n, i, j
  real :: diagonal_sum=0, last_column_sum=0, lower_triangular_sum=0
  real, allocatable :: a(:,:)

  ! Read the order of the matrix
  print *, 'Enter the order of the matrix (n): '
  read*, n

  ! Allocate memory for matrix A
  allocate(a(n, n))

  ! Input: Enter the elements of matrix A in row-wise
  print *, "Enter the elements of matrix A in row-wise"
  do i=1,n
    read *, (a(i, j), j = 1, n)
  end do
  ! Calculate the sums
  do i = 1, n
     do j = 1, n
        ! Diagonal elements
        if (i == j) then
           diagonal_sum = diagonal_sum + a(i, j)
        end if

        ! Last column elements
        if (j == n) then
           last_column_sum = last_column_sum + a(i, j)
        end if

        ! Lower triangular elements
        if (i > j) then
           lower_triangular_sum = lower_triangular_sum + a(i, j)
        end if
     end do
  end do

  ! Print the results
  print '(A,F8.2)', 'Sum of diagonal elements: ', diagonal_sum
  print '(A,F8.2)', 'Sum of last column elements: ', last_column_sum
  print '(A,F8.2)', 'Sum of lower triangular elements: ', lower_triangular_sum

end program num_17
```

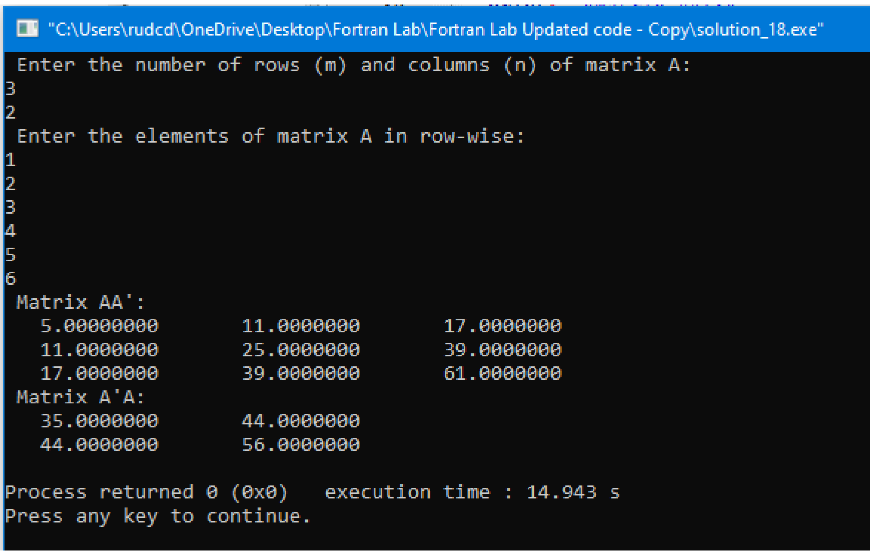


"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_17.exe"

```
 Enter the order of the matrix (n):
3
 Enter the elements of matrix A in row-wise
1
2
3
4
5
6
7
8
9
Sum of diagonal elements:    15.00
Sum of last column elements:    18.00
Sum of lower triangular elements:    19.00

Process returned 0 (0x0)   execution time : 124.474 s
Press any key to continue.
```

▼ 18. Read a matrix A and print AA’ and A’A.

```
program num_18
  implicit none
  integer :: i, j, k, m, n
  real, allocatable :: a(:,:), a_transpose(:,:), aa(:,:), a_transpose_a(:,:)

  ! Input: Enter the order [m*n] of matrix A
  print*, "Enter the number of rows (m) and columns (n) of matrix A:"
  read *, m, n

  ! Allocate memory for matrices
  allocate(a(m, n), a_transpose(n, m), aa(m, m), a_transpose_a(n, n))

  ! Input: Enter the elements of matrix A in row-wise
  print*, "Enter the elements of matrix A in row-wise:"
  do i = 1, m
    do j = 1, n
      read*, a(i, j)
    end do
  end do
  ! Calculate A_transpose (transpose of A)
  do i = 1, n
    do j = 1, m
      a_transpose(i, j) = a(j, i)
    end do
  end do

  ! Calculate AA' (matrix multiplication)
  do i = 1, m
    do j = 1, m
      aa(i, j) = 0.0
      do k = 1, n
        aa(i, j) = aa(i, j) + a(i, k) * a_transpose(k, j)
      end do
    end do
  end do

  ! Output: Matrix AA'
  print*, "Matrix AA':"
  do i = 1, m
    print*,(aa(i, j), j = 1, m)
  end do

  ! Calculate A'A (matrix multiplication)
  do i = 1, n
    do j = 1, n
      a_transpose_a(i, j) = 0.0
      do k = 1, m
        a_transpose_a(i, j) = a_transpose_a(i, j) + a_transpose(i, k) * a(k, j)
      end do
    end do
  end do

  ! Output: Matrix A'A
  print*, "Matrix A'A:"
  do i = 1, n
    print*, (a_transpose_a(i, j), j = 1, n)
  end do

end program num_18
```



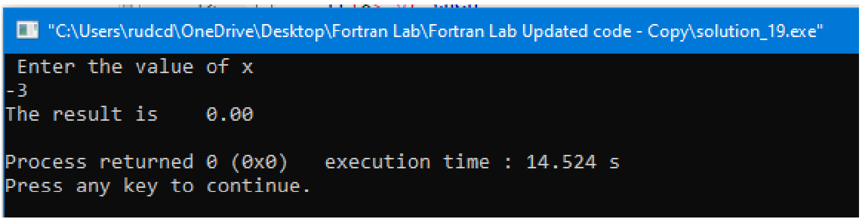
"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_18.exe"

```
 Enter the number of rows (m) and columns (n) of matrix A:
3
2
 Enter the elements of matrix A in row-wise:
1
2
3
4
5
6
 Matrix AA':
   5.00000000       11.0000000       17.0000000
   11.0000000       25.0000000       39.0000000
   17.0000000       39.0000000       61.0000000
 Matrix A'A:
   35.0000000       44.0000000
   44.0000000       56.0000000

Process returned 0 (0x0)   execution time : 14.943 s
Press any key to continue.
```

▼ 19. Write a function subprogram for the function f(x) defined as follows:

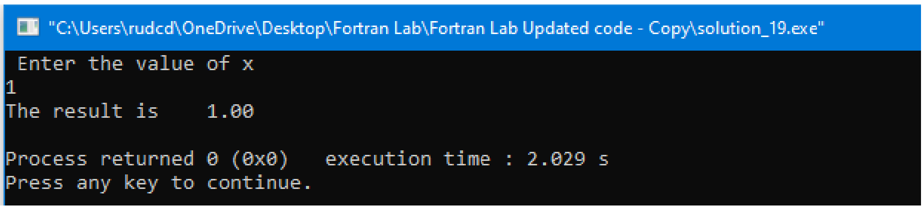
$$f(x) = \begin{cases} 0 & if\ x \leq 0 \\ x & if\ 0 < x \leq 2 \\ 2x - 2 & if\ x > 2 \end{cases}$$

```
program num_19
    implicit none
    real:: value , f
    print*,"Enter the value of x"
    read*, value
    print'(A, F8.2)',"The result is",f(value)
end program

real function f(x)
    implicit none
    real::x
    if(x<=0) then
        f=0
    elseif(x>0 .and. x<=2) then
        f=x
    else
        f=2*x-2
    end if
end function
```



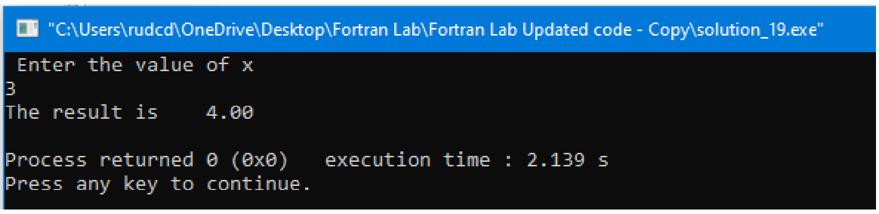
"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_19.exe"

```
 Enter the value of x
-3
The result is    0.00

Process returned 0 (0x0)   execution time : 14.524 s
Press any key to continue.
```



"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_19.exe"

```
 Enter the value of x
1
The result is    1.00

Process returned 0 (0x0)   execution time : 2.029 s
Press any key to continue.
```

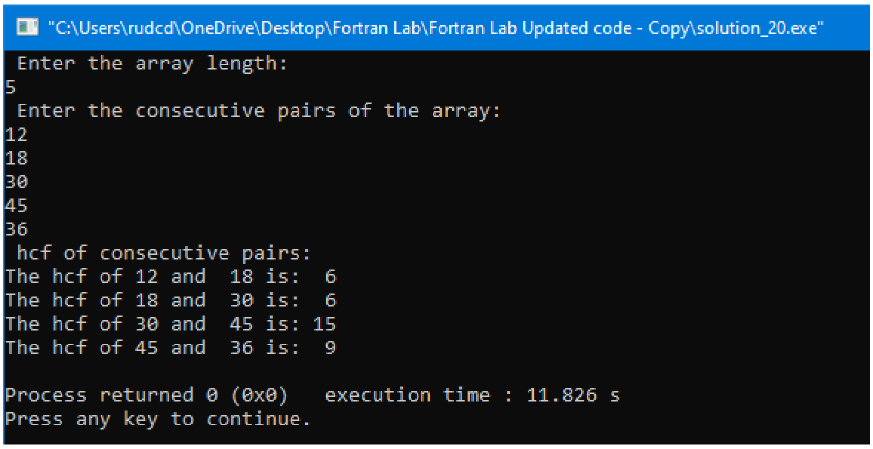


"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_19.exe"

```
 Enter the value of x
3
The result is    4.00

Process returned 0 (0x0)   execution time : 2.139 s
Press any key to continue.
```

▼ 20. Define a function to determine h.c.f. of a pair of positive integers and use it to print the h.c.f. of consecutive pairs of an array of positive integers.

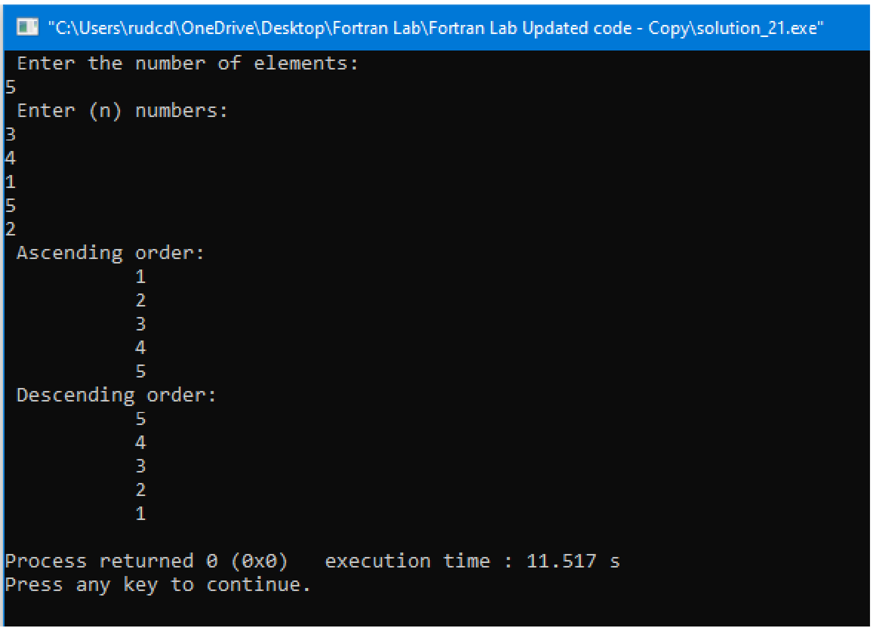
```
program num_20
  implicit none
  integer :: i, m, thcf, hcf
  integer, allocatable :: a(:)

  print *, "Enter the array length:"
  read *, m
  allocate(a(m))

  print *, "Enter the consecutive pairs of the array:" !consecutive pairs = por por jora number
  read *, (a(i), i = 1, m)

  print *, "hcf of consecutive pairs:" !HCF = highest common factor
  do i = 1, m - 1
    thcf = hcf(a(i), a(i+1)) !THCF = Temporary highest common factor
    print'(A,I3,A,I3,A,I3)', "The hcf of", a(i),  " and ", a(i+1),  " is:",  thcf
  end do

end program num_20

integer function hcf(m, n)
  implicit none
  integer :: m, n, minimum, maximum, i
  maximum = max(m, n)
  minimum = min(m, n)

  do i = 1, minimum
    if (mod(minimum, i) == 0 .and. mod(maximum, i) == 0) then
      hcf = i
    end if
  end do
end function hcf
```


"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_20.exe"

```
 Enter the array length:
5
 Enter the consecutive pairs of the array:
12
18
30
45
36
 hcf of consecutive pairs:
The hcf of 12 and  18 is:  6
The hcf of 18 and  30 is:  6
The hcf of 30 and  45 is: 15
The hcf of 45 and  36 is:  9

Process returned 0 (0x0)   execution time : 11.826 s
Press any key to continue.
```

▼ 21. Read n numbers and print them and their sorted list (ascending and descending).

```
program num_21
  implicit none
  integer :: n, i, j
  integer, allocatable :: a(:)
  real :: tempo

  ! Prompt user for the number of elements
  print *, "Enter the number of elements:"
  read *, n

  ! Allocate memory for the array
  allocate(a(n))

  ! Prompt user to enter n numbers
  print *, "Enter (n) numbers:"
  read *, a  !5, 3 , 2, 4, 1
  do i=1,5
    do j=1, 5-i
        if(a(3)<a(4)) then
          tempo = a(j)
          a(j)= a(j+1)
          a(j+1) = tempo
        end if
    end do
  end do

  ! Display the sorted list in ascending order
  print *, "Ascending order:"
  do i = 1, n
    print *, a(n+1-i)
  end do

  ! Display the sorted list in descending order
  print *, "Descending order:"
  do i = 1, n
    print *, a(i)
  end do
end program num_21
```

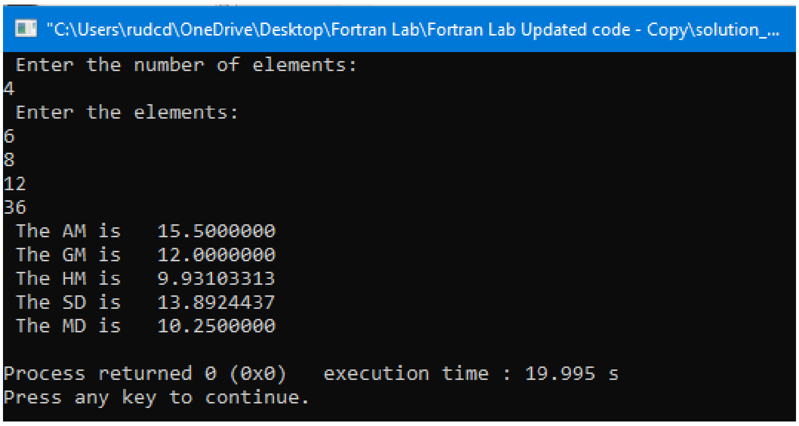


"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_21.exe"

```
 Enter the number of elements:
5
 Enter (n) numbers:
3
4
1
5
2
 Ascending order:
           1
           2
           3
           4
           5
 Descending order:
           5
           4
           3
           2
           1

Process returned 0 (0x0)   execution time : 11.517 s
Press any key to continue.
```

▼ 22. Print the A.M, G.M, H.M, SD, MD of a list of numbers using subroutine subprogram and function subprogram.

```
program num_22
  implicit none
  integer :: m, i
  real :: AM, GM, HM, SD, MD
  real, allocatable :: a(:)

  print *, "Enter the number of elements:"
  read *, m
  allocate(a(m))
  print *, "Enter the elements:"
  read *, (a(i), i = 1, m)

  print *, "The AM is", AM(a, m)
  print *, "The GM is", GM(a, m)
  print *, "The HM is", HM(a, m)
  print *, "The SD is", SD(a, m)
  print *, "The MD is", MD(a, m)

end program num_22

!Function to calculate the Arithmetic Mean (AM)
real function AM(a, m)
  implicit none
  integer :: i, m
  real :: a(m), sum
  sum = 0.0
  do i = 1, m
    sum = sum + a(i)
  end do
  AM = sum / m
end function

!Function to calculate the Geometric Mean(GM)
real function GM(a, m)
  implicit none
  integer :: i, m
  real :: a(m), pro
  pro = 1.0
  do i = 1, m
    pro = pro * a(i)
  end do
  GM = pro ** (1.0 / m)
end function

!Function to calculate the Harmonic Mean (HM)
real function HM(a, m)
  implicit none
  integer :: i, m
  real :: a(m), sum
  sum = 0.0
  do i = 1, m
    sum = sum + (1.0 / a(i))
  end do
  HM = m / sum
end function

!Function to calculate the Standard Deviation (SD)
real function SD(a, m)
  implicit none
  integer :: i, m
  real :: a(m), sum, AM, Avg
  Avg = AM(a, m)  !function called
  sum = 0.0
  do i = 1, m
    sum = sum + ((a(i) - Avg) ** 2)
  end do
  SD = sqrt(sum / (m - 1))
end function

!Function to calculate the Median (MD)
real function MD(a, m)
  implicit none
  integer :: i, m
  real :: a(m), sum, AM, Avg
  Avg = AM(a, m)
  sum = 0.0
  do i = 1, m
    sum = sum + abs(a(i) - Avg)
  end do
  MD = sum / m
end function
```



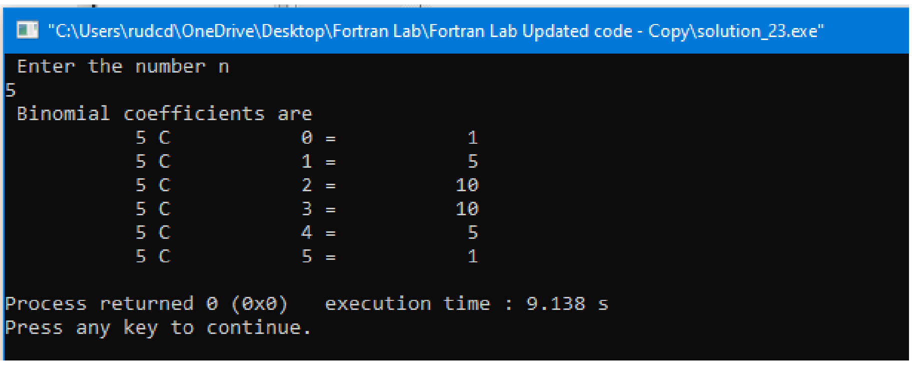
"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_...

```
 Enter the number of elements:
4
 Enter the elements:
6
8
12
36
 The AM is   15.5000000
 The GM is   12.0000000
 The HM is   9.93103313
 The SD is   13.8924437
 The MD is   10.2500000

Process returned 0 (0x0)   execution time : 19.995 s
Press any key to continue.
```

▼ 23. Print the binomial coefficients nCr for 0≤r≤n (n<9) using function sub program.

```
program num_23
  implicit none
  integer :: n, fac, nCr, i, q(20)
  integer, allocatable :: p(:)

  ! Input loop label
  100 continue

  ! Prompt user to enter the number n
  print*, "Enter the number n"
  read*, n

  ! Check if n is less than 9
  if (n < 9) then
    ! Allocate memory for array p
    allocate(p(n))

    ! Print binomial coefficients
    print*, "Binomial coefficients are"
    do i = 0, n
      nCr = fac(n) / (fac(i) * fac(n - i))
      p(i) = nCr
      print*, n, "C", i, "=", p(i)
    end do
  else
    ! If n is not less than 9, prompt user to enter a valid number
    print*, "Enter a number between 0 to 9"
    go to 100
  end if

end program num_23

! Function to calculate factorial
integer function fac(n)
  implicit none
  integer :: n, f, i

  f = 1

  ! Calculate factorial if n is non-negative
  if (n >= 0) then
    do i = 1, n
      f = f * i
    end do
  end if

  ! Return the factorial value
  fac = f
```

```
end function
```

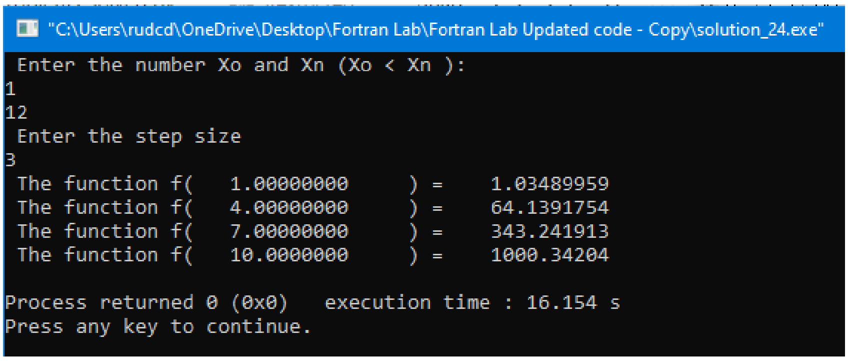


```
"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_23.exe"
 Enter the number n
5
 Binomial coefficients are
           5 C           0 =           1
           5 C           1 =           5
           5 C           2 =          10
           5 C           3 =          10
           5 C           4 =           5
           5 C           5 =           1

Process returned 0 (0x0)   execution time : 9.138 s
Press any key to continue.
```

▼ 24.

Print the values of x and y(x) for $x=x_0$ to $x=x_n$ with step h, where $y(x)=x^3+\sin 2x$.

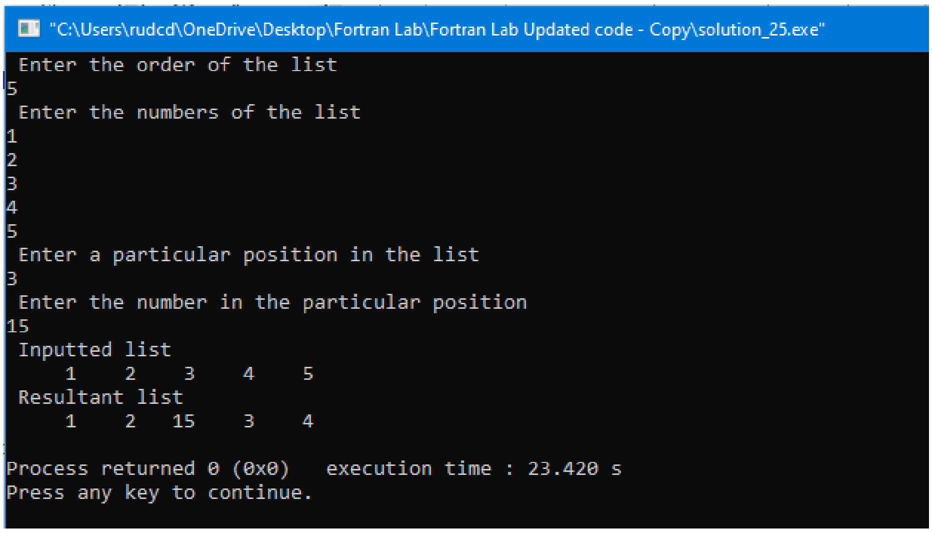
```
program num_24
    implicit none
    real:: x0,xn,f, i , h
    print*,"Enter the number Xo and Xn (Xo < Xn ): "
    read*, x0,xn
    print*, "Enter the step size"
    read*, h

    do i = x0, xn, h
     print*, "The function f(", i , ") = " ,f(i)
     x0 = x0+h
     if (x0>xn) exit
    end do
end program

   real function f(x)
   implicit none
   real:: x, pi
   pi = 3.1416
   f = x**3 + sin(2*x*pi/180)
   end function
```



```
"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_24.exe"
 Enter the number Xo and Xn (Xo < Xn ):
1
12
 Enter the step size
3
 The function f(   1.00000000     ) =    1.03489959
 The function f(   4.00000000     ) =    64.1391754
 The function f(   7.00000000     ) =    343.241913
 The function f(   10.0000000     ) =    1000.34204

Process returned 0 (0x0)   execution time : 16.154 s
Press any key to continue.
```

▼ 25. Read n numbers and insert a number in a particular position and print the inputted list and the resultant list.

```
program num_25
    implicit none
    integer :: i, n, p, m
    integer, allocatable :: x(:)

    ! Prompt user to enter the order of the list
    print*, "Enter the order of the list"
    read*, n

    ! Allocate memory for the array x
    allocate(x(n))

    ! Prompt user to enter the numbers of the list
    print*, "Enter the numbers of the list"
    read*, (x(i), i=1, n)

    ! Prompt user to enter a particular position in the list
    print*, "Enter a particular position in the list"
    read*, p

    ! Prompt user to enter the number to insert at the particular position
    print*, "Enter the number in the particular position"
    read*, m

    ! Print the inputted list
    print*, "Inputted list"
    print 10, (x(i), i=1, n)

    ! Shift elements to make space for the new number
    do i = p, n
        x(n + p + 1 - i) = x(n + p - i)
    end do

    ! Insert the new number at the particular position
    x(p) = m

    ! Print the resultant list
    print*, "Resultant list"
    print 10, (x(i), i=1, n ) !n+1 --if we want to show last element which is skipped

10  format(1X, 100i5, 1X)
end program num_25
```



"C:\Users\rudcd\OneDrive\Desktop\Fortran Lab\Fortran Lab Updated code - Copy\solution_25.exe"

```
 Enter the order of the list
5
 Enter the numbers of the list
1
2
3
4
5
 Enter a particular position in the list
3
 Enter the number in the particular position
15
 Inputted list
     1    2    3    4    5
 Resultant list
     1    2   15    3    4

Process returned 0 (0x0)   execution time : 23.420 s
Press any key to continue.
```