Aim : To interpolate the using the Newton Forward Interpolation method.

Theory:

Newton Forward Interpolation Method:

It is a finite difference identity giving an interpolated value between tabulated points in terms of the first value and the powers of the forward difference .

Formula :

$$f(x)=f(x_0)+\frac{p}{1!}\Delta f(x_0)+\frac{p(p-1)}{2!}\Delta^2 f(x_0)+\frac{p(p-1)(p-2)}{3!}\Delta^3 f(x_0)+\ \dots\ +\frac{p(p-1)\dots(p-n-1)}{n!}\Delta^n f(x_0)$$

here $p\ =\ \frac{x-x_0}{h}$ and h is spacing between the points

Algorithm:

Here we are interpolate the graph using the points by Newton Forward Interpolation Method and we can do this by taking a large number of points between the first and the last point that given and finding the value of *f(x)* at every point using the Newton Forward Interpolation Method.

This can be done by make first finding the Forward Difference for all given points and then running a loop for all the new points , at first we find the x_0 for every value of x , then using the formula at the function finding the value at that point and write the value in a file.

Program :

```
program main
   implicit none ! declaring the variables

   real , dimension(:) , allocatable :: x(:) , y(:) , diff(:)
   real :: result , p , d , val , fact
   integer :: i , j , k , n , index , row , num_points
   logical :: f1,f2

   print*, "Enter the number of the given points : " ! getting the number of points
   read*, n
   if (n <= 0) stop "Invalid entry" ! And validating them

   allocate(x(n),y(n),diff((n**n - n)/2)) ! allocating the memory for all arrays
! here we are using the linear array for the forward difference table to saving the memory

   print*, "Enter the value of x and y : " ! reading the given points and number of points
generated b/w them
   print*, "x      y"
   read*, (x(i),y(i),i=1,n)
   print *, "Enter the number of points taking between x0 and xn :: "
   read*, num_points
   if (num_points <= 0 ) stop "Invalid entry"

   inquire(file="givenpoints.dat",exist=f1) ! opening the file for storing the value of given and
calculated points
   if(f1) then
```

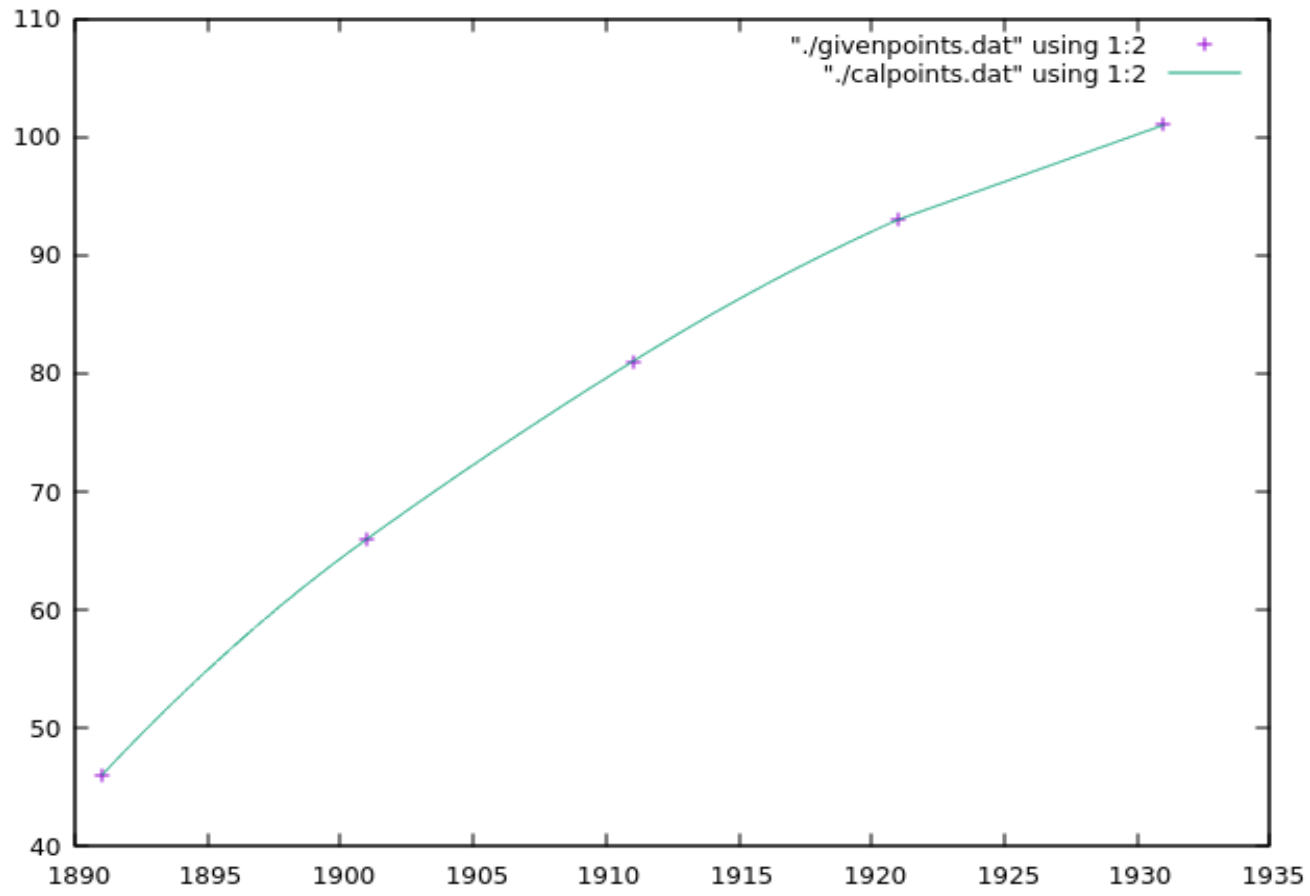
```
   open(1,file="givenpoints.dat",status="replace")
else
   open(1,file="givenpoints.dat",status="new",action="write")
endif
inquire(file="calpoints.dat",exist=f2)
if(f2) then
   open(2,file="calpoints.dat",status="replace")
else
   open(2,file="calpoints.dat",status="new",action="write")
endif

do i=1,n
   write(1,*)x(i),y(i) ! writing the given points in the file
enddo

row = 0
do i = 1,n-1  ! here calculating the forward difference and storing in linear array diff
   do j = 1,n-i
      if(i==1) then
         diff(j) = y(j+1) - y(j)
      else
         diff(row+j) = diff((row-n+i-1)+j+1) - diff((row-n+i-1)+j)
      endif
   enddo
   row = row + n - i ! this variable row can give ability to change the column of table
enddo

d = (x(n) - x(1))/real(num_points)    ! getting the difference between the points we calculating

do k = 1,num_points-1       ! taking a loop for calculating the points
   val = x(1) + k*d
   index = cal_x0_index(val)     ! here we find the index of x0 for a given point val
   p = (val - x(index))/(x(2) - x(1))
   result = y(index)          ! result is the value of f(x) at x
   fact = 1           ! for getting the insuit factorial in the formulae
   row = 0
   do i = 1, n-index
      fact = fact*i
      result = result + p*diff(row+index)/fact
      p = p*(p-i)
      row = row + n-i
   enddo
   write(2,*)val,result
enddo

close(1)

print*, "Simulation is complete"

stop

contains
```

```
    integer function cal_x0_index(value)    ! this is function for calculating the index of
       real , intent(in) :: value           ! x0 for a given value of x
       integer :: k
       do k = 1,n
          if (x(k) > value) then
             cal_x0_index = k-1
             return
          endif
       enddo
       stop "Value of x  is over the upper limit"
    end function cal_x0_index
end program
```

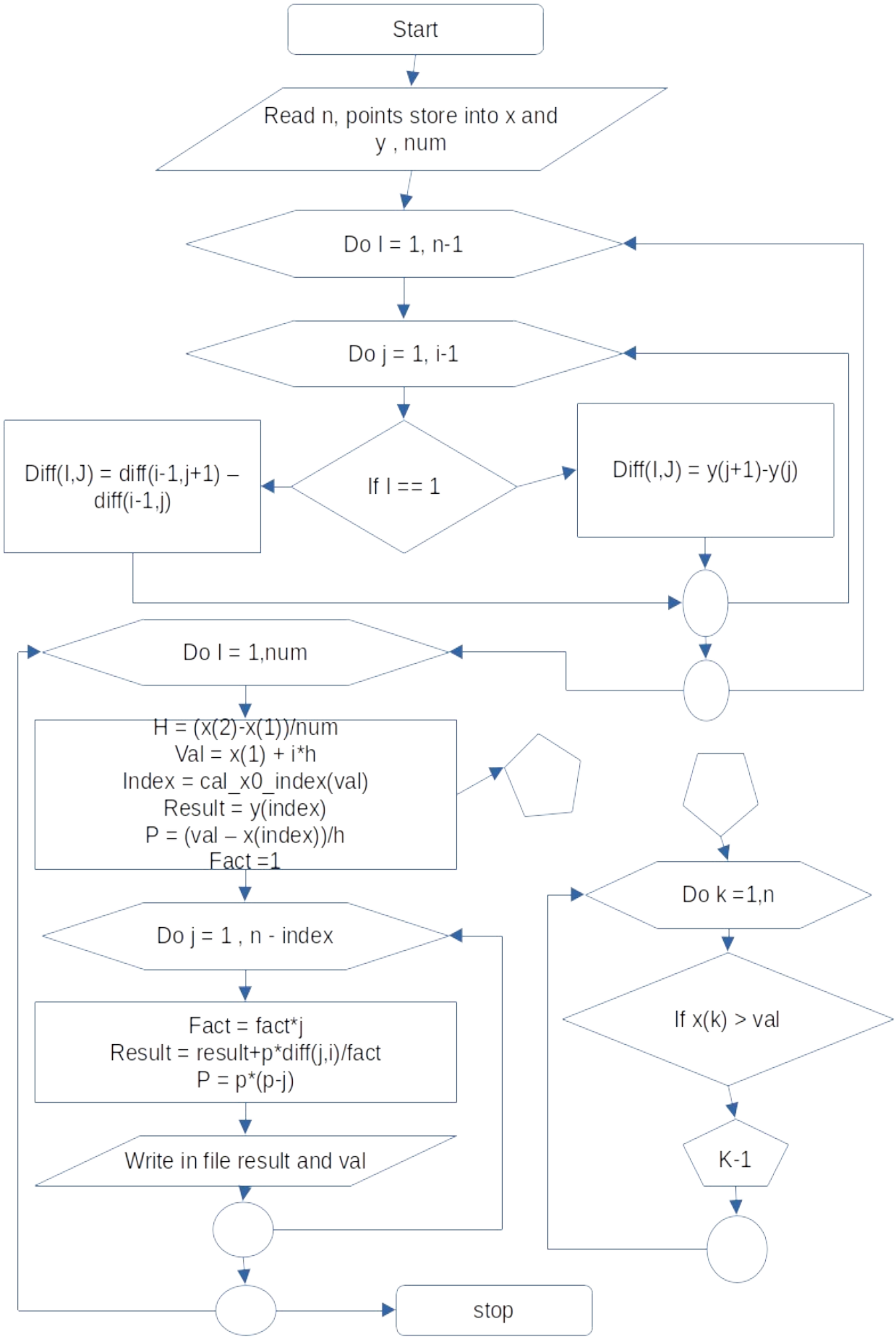
Output:

```
 Enter the number of the given points :
5
 Enter the value of x and y :
 x      y
1891    46
1901    66
1911    81
1921    93
1931    101
 Enter the number of points taking between x0 and xn ::
10000
 Simulation is complete
```

Result :



Flow Chart:


Start
Read n, points store into x and y , num
Do I = 1, n-1
Do j = 1, i-1
If I == 1
Diff(I,J) = diff(i-1,j+1) – diff(i-1,j)
Diff(I,J) = y(j+1)-y(j)
Do I = 1,num
H = (x(2)-x(1))/num
Val = x(1) + i*h
Index = cal_x0_index(val)
Result = y(index)
P = (val – x(index))/h
Fact =1
Do j = 1 , n - index
Fact = fact*j
Result = result+p*diff(j,i)/fact
P = p*(p-j)
Write in file result and val
Do k =1,n
If x(k) > val
K-1
stop