Aim : To interpolating and extrapolating by using Lagrangian Interpolation Method

Theory :

Lagrangian Interpolation Method:

It’s a numerical method which use the Lagrangian function and given value of function to find the value of the function at arbitrary point.

$$f(x)=\sum_i [\mathrm{L}(i, x_i) f(x_i)]$$

How to program :

First we need to define two array to store the value of the x and f(x) and we need to define a subroutine for finding Lagrangian at every x and value of the x. And at the end we need a loop from 1 to total points and sum all the value of multiplication of function and the Lagrangian at that point . And this gives the value of the function at arbitrary x. And for continue interpolating and extrapolate we doing the same for a number of value of the slight higher and lower value of x than the given extreme points.

Program in FORTRAN 95:

```
program main

    implicit none

    ! declaring the variables

    real , dimension(:) , allocatable :: x ,y , nx , ny
    integer :: n,i,m , j
    real :: larg , x_rand , result , upper , lower ,h
    logical :: f1,f2

    ! getting the number of the set of points given and validate the number of points

    print *, "Enter the number of the entry :: "
    read *, n

    if (n <= 0) then
```

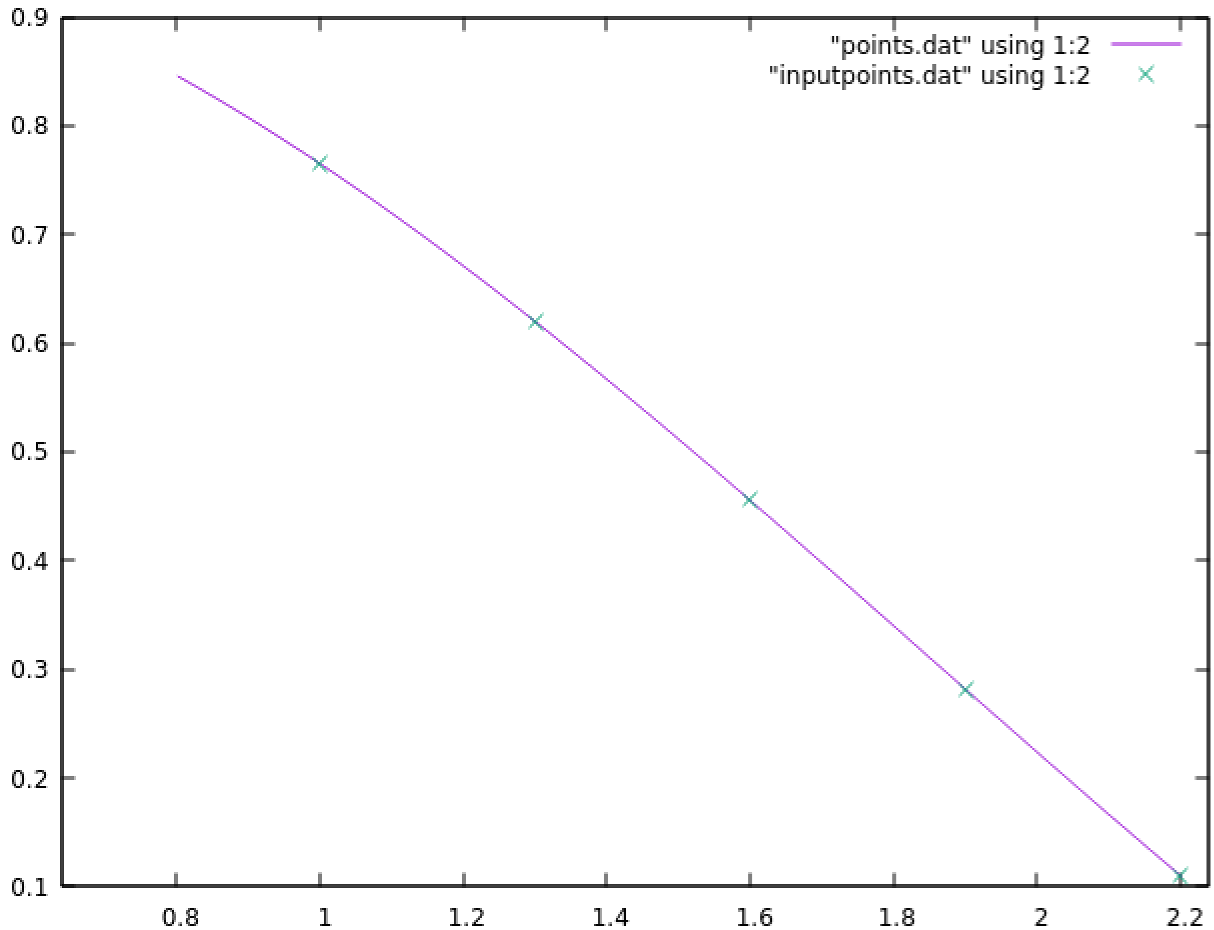
```
    stop "Invalid number of the entry"
endif

! open the file store the input and calculated points

inquire(file="points.dat",exist=f1)
inquire(file="inputpoints.dat",exist=f2)
if (f1) then
    open(1,file="points.dat",status="replace")
else
    open(1,file="points.dat",status="new",action="write")
endif
if (f2) then
    open(2,file="inputpoints.dat",status="replace")
else
    open(2,file="inputpoints.dat",status="new",action="write")
endif

! allocate the x and y array for storing the points

allocate(x(n),y(n))

! getting the points from the user

print *, "Enter the entry :: "
print *, "x     y"
read *,(x(i),y(i),i = 1,n)
do i = 1,n
    write(2,*)x(i),y(i)
enddo

! getting the upper and lower limit of the graph with points number

print *, "Enter the upper limit of the extrapolation ::"
read *, upper
print *, "Enter the lower limit of the extrapolation ::"
read *, lower
print *, "Enter the number of x we find between the limits :: "
read *, m

if (m <=0) stop "Invalid number of x taken "

! Making the step size

h = (upper - lower)/real(m)
```

```fortran
    if (h<=0) stop "upper and lower limit is not good"

    ! find the value of f(x) at every point we getting

    do j = 1,m
        x_rand = lower + j*h
        result = 0
        do i = 1,n
            call lagrange(i,x_rand)
            result = result + y(i)*larg
        enddo
        write(1,*)x_rand,result ! writing into file
    enddo

    print*, "interpolation and extrapolate is completed"

    ! deallocate the x and y array to free the memory

    deallocate(x,y)

    close(1) ! closing the file
    close(2)

    stop

    contains
        ! subroutine for calculating the lagrange at the point
        subroutine lagrange(index,val)
            integer , intent(in) :: index
            real , intent(in) :: val
            integer :: j
            larg = 1.0
            do j = 1,n
                if (j.ne.index) then
                    larg = larg*(val-x(j))/(x(index)-x(j))
                endif
            enddo
            return
        end subroutine

end program
```

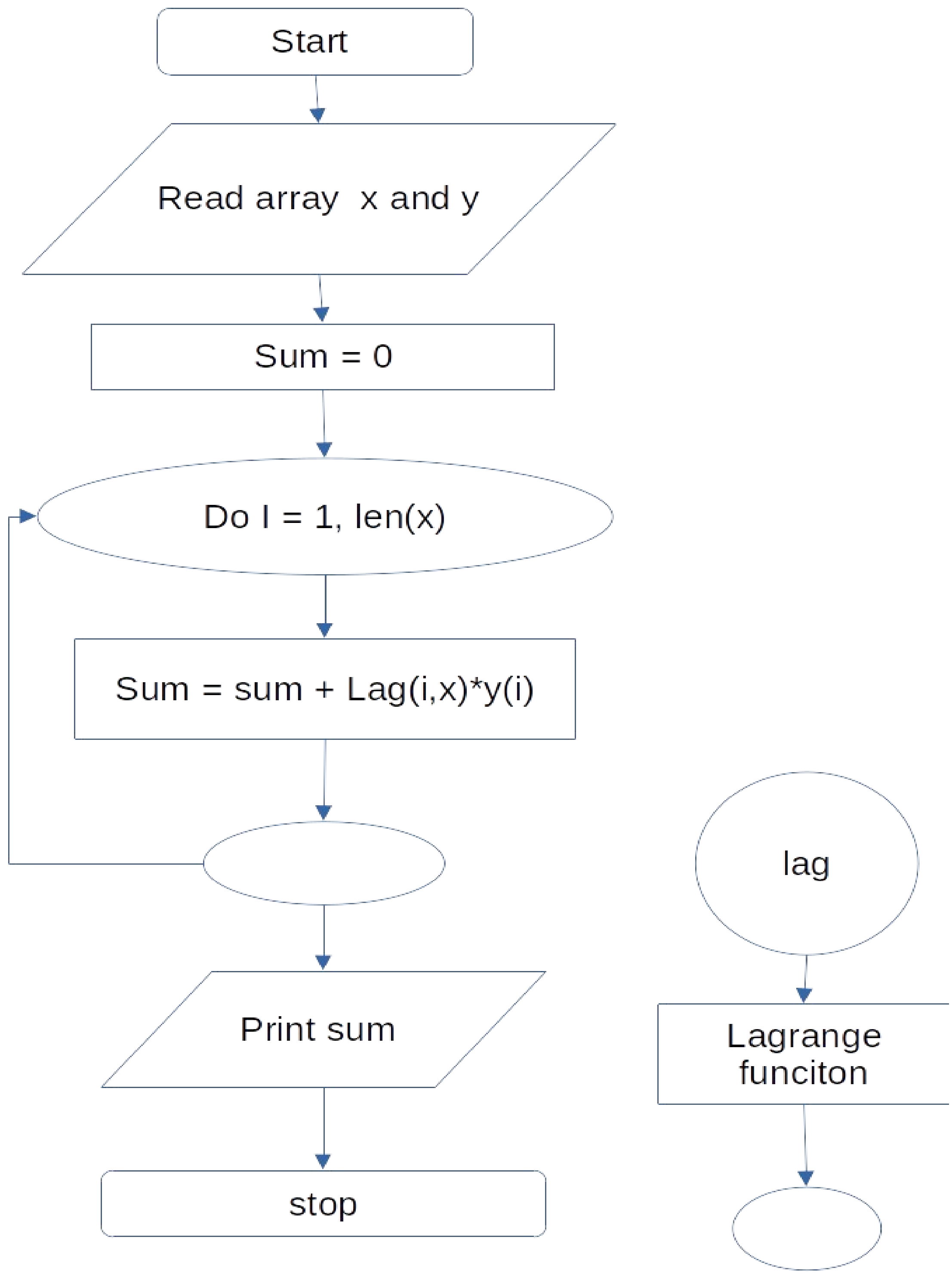
Output :

Enter the number of the entry ::
5
Enter the entry ::
x y
1 0.7651
1.3 0.6200
1.6 0.4554
1.9 0.2818
2.2 0.1103
Enter the upper limit of the extrapolation ::
2.2
Enter the lower limit of the extrapolation ::
0.8
Enter the number of x we find between the limits ::
1000
interpolation and extrapolation is completed

Result :



Flow Chart :


Start
Read array x and y
Sum = 0
Do I = 1, len(x)
Sum = sum + Lag(i,x)*y(i)
Print sum
stop
lag
Lagrange
funciton