Bisection Method Program

```
program bisection
   implicit none
   real(8)::a,b,c,fa,fb,tol,fc,cc
   integer::max_iter,iter
   write(*,*)'entre the value of tolerance='
   read(*,*)tol
   write(*,*)'entre the value of maximum iteration='
   read(*,*)max_iter
   10 write(*,*)'entre the value of intervel a and b='
   read(*,*)a,b
   if(f(a)*f(b)>0)then
      write(*,*)'root does not exit in this intrevel.....'
      write(*,*)'put another value for iteration='
      goto 10
      stop
   end if
   iter=0
   write(*,20)
   20 format('Iterztion',15x,'a values',15x,'b values',15x,'c values')
   WRITE(*,*)'-------------------------------------------------------------------------------'
   do while(abs(b-a)>tol.and.iter<max_iter)
      iter=iter+1
      c=(a+b)/2.0
      cc=abs(a-b)/2.0
      fa=f(a)
      fb=f(b)
      fc=f(c)
      write(*,2)iter,a,b,c
      2 format(I5,5X,F20.5,5X,F20.5,5X,F20.5)
      if(f(c)==0.0.or.cc<tol)then
```

```
      write(*,*)'The root is=',c
      write(*,*)'Number of iteration=',iter
         stop
      end if
      if(fa*fc<0.0)then
         b=c
         else
            a=c
      end if
      if(iter==max_iter)then
         write(*,*)'Method failed after number of iteration=',iter
         stop
      end if
   end do
   contains
   real(8)function f(x)
   real(8),intent(in)::x
   f=x**3.0-2.0*x-5.0
   end function f
end program
```

Fixed Point Iteration Program

```
 program fixpoint
   implicit none
   real(8)::x,xn,tol,er
   integer::max_iter,iter
   write(*,*)"put values xn,tolerance and maximum iteration:"
   read(*,*)xn,tol,max_iter
   if(f(xn)>1.0)then
      write(*,*)'the function is divergence'
      stop
```

```
end if
if(g(xn)==0.0)then
   write(*,*)"the root is=",xn
end if
iter=0
x=xn
write(*,8)
8 format('iteration',15x,'x values',15x,'xn values',15x,'errors')
print*,'--------------------------------------------------------------------------------'
do while(iter<max_iter)
   iter=iter+1
   xn=x
   x=g(xn)
   er=abs((g(x)-g(xn))/g(xn))*100.0
   write(*,4)iter,x,xn,er
   4 format(I5,5X,F20.5,5X,F20.5,5X,F20.5)
   if(g(x)==0.0.or.abs(x-xn)<tol)then
      write(*,*)'the root is=',x
      write(*,*)'the num of iteration=',iter
      stop
   end if
   if(iter==max_iter)then
      write(*,*)'method failed after iteration num=',iter
   end if
end do
contains
real(8)function g(x)
real(8),intent(in)::x
g=(COS(X)+1.0)/3.0
end function g
REAL(8)FUNCTION f(x)
```

```
 real(8),intent(in)::x
 f=-sin(x)/3.0
 end function f
end program
```

Newton Raphson Method Program

```
program NEWTONRAPHSAN
 implicit none
 real(8)::x,xn,tol,er
 integer::max_iter,iter
20 write(*,*)"put values xn,tolerance and maximum iteration:"
 read(*,*)xn,tol,max_iter
 if(f(xn)==0.0)then
    write(*,*)'the function is divergence and solution can not possible'
    write(*,*)'take another value of xn'
    goto 20
    stop
 end if
 if(g(xn)==0.0)then
    write(*,*)"the root is=",xn
 end if
 iter=0
 x=xn
 write(*,8)
 8 format('iteration',15x,'x values',15x,'xn values',15x,'errors')
 print*,'-------------------------------------------------------------------------------'
 do while(iter<max_iter)
    iter=iter+1
    xn=x
    x=xn-(g(xn)/f(xn))
```

```
      er=abs((x-xn)/xn)
      write(*,4)iter,x,xn,er
      4 format(I5,5X,F20.5,5X,F20.5,5X,F20.5)
      if(g(x)==0.0.or.abs(x-xn)<tol)then
         write(*,*)'the root is=',x
         write(*,*)'the num of iteration=',iter
         stop
      end if
      if(iter==max_iter)then
         write(*,*)'method failed after iteration num=',iter
      end if
   end do
   contains
   real(8)function g(x)
   real(8),intent(in)::x
   g=x**3-3.0
   end function g
   REAL(8)FUNCTION f(x)
   real(8),intent(in)::x
   f=3.0*x**2
   end function f
 end program
```

Regula Falsi Method Program

```
program regularfalsimethod
   implicit none
   real(8)::a,b,c,fa,fb,tol,fc,cc
   integer::max_iter,iter
   write(*,*)'entre the value of tolerance='
   read(*,*)tol
   write(*,*)'entre the value of maximum iteration='
   read(*,*)max_iter
```

```
10 write(*,*)'entre the value of intervel a and b='
read(*,*)a,b
if((f(b)-f(a))==0.0)then
   write(*,*)'this method will failed'
   end if
if(f(a)*f(b)>0.0)then
   write(*,*)'root does not exit in this intrevel.....'
   write(*,*)'put another value for iteration.....'
   goto 10
   stop
end if
iter=0
write(*,20)
20 format('Iterztion',15x,'a values',15x,'b values',15x,'c values')
WRITE(*,*)'-------------------------------------------------------------------------------'
do while(iter<max_iter)
   iter=iter+1
   c=(a*f(b)-b*f(a))/(f(b)-f(a))
   fa=f(a)
   fb=f(b)
   fc=f(c)
   cc=abs(f(c))
   write(*,2)iter,a,b,c
   2 format(I5,5X,F20.5,5X,F20.5,5X,F20.5)
   if(f(c)==0.0.or.cc<tol)then
      write(*,*)'the approximate root=',c
      write(*,*)'number of iteration=',iter
   stop
   end if
   if(fa*fc<0.0)then
      b=c
```

```
            else
               a=c
        end if
        if(iter==max_iter)then
           write(*,*)'Method failed after number of iteration=',iter
           stop
        end if
     end do
     contains
     real(8)function f(x)
     real(8),intent(in)::x
     f=2.0*exp(x)*sin(x)-3.0
     end function f
end program
```

Lagrange Program:

```
dimension x(30),y(30)
write(*,*)'how many numbers='
read*,n
do i=1,n
   write(*,*)'entre values of x and y='
   read(*,*)x(i),y(i)
end do
write(*,*)'Values for interpulation'
write(*,*)'-------------------------------------------------'
do i=1,n
   write(*,20)x(i),y(i)
```

```
   20 format('x=',f20.5,5x,'y=',f20.5)
end do
write(*,*)'-------------------------------------------------'
10 write(*,*)'entre the expected point='
read(*,*)t
s=0.0
do i=1,n
   prod=1.0
   do j=1,n
      if(i/=j)prod=prod*(t-x(j))/(x(i)-x(j))
   end do
   s=s+y(i)*prod
end do
write(*,30)s
30 format('interpoleted value=',f20.5)
goto 10
stop
end
```

Newton Forward Program:

```
dimension x(30),y(30,30)
write(*,*)'how many values='
```

```
read*,n
do i=1,n
   write(*,*)'values of x and y='
   read(*,*)x(i),y(i,1)
end do
write(*,*)'Values for interpolation'
write(*,*)'================================================================='
do i=1,n
   write(*,20)x(i),y(i,1)
   20 format('x=',f20.5,5x,'y=',f20.5)
end do
write(*,*)'================================================================='
write(*,*)'entre require value='
read*,t
h=x(2)-x(1)
u=(t-x(1))/h
do j=2,n
  do i=2,n
   y(i,j)=y(i,j-1)-y(i-1,j-1)
end do
end do

write(*,*)'The interpulation table'
write(*,*)'-------------------------------------------'
```

```
do i=1,n
   write(*,'(f10.5,3x)',advance="no")x(i)
   do j=1,i
         write(*,'(f10.5,3x)',advance="no")y(i,j)
   end do
   write(*,*)
end do
write(*,*)'-------------------------------------------------'

sum=y(1,1)
do i=2,n
   prod=1.0
   do j=2,i
      prod=prod*(u-(j-2.0))
   end do
   sum=sum+prod/IFACT(i-1)*y(i,i)
end do
write(*,*)'Approximate value=',sum
stop
end
function IFACT(i)
   f=1
   do k=1,i
```

```
      f=f*k
   end do
   return
end function
```

Newton's Backward Program:

```
dimension x(30),y(30,30)
write(*,*)'how many values='
read*,n
do k=1,n
   i=n-(k-1)
   write(*,*)'values of x and y='
   read(*,*)x(i),y(i,1)
end do
write(*,*)'Values for interpulation'
write(*,*)'============================================================='
do i=1,n
   write(*,20)x(i),y(i,1)
   20 format('x=',f20.5,5x,'y=',f20.5)
end do
write(*,*)'============================================================='
write(*,*)'entre require value='
read*,t
h=x(1)-x(2)
u=(t-x(1))/h
do j=2,n
  do i=2,n
   y(i,j)=y(i-1,j-1)-y(i,j-1)
end do
end do
```

```
write(*,*)'The interpulation table'
write(*,*)'-----------------------------------------------'

do i=1,n
   write(*,'(f10.5,3x)',advance="no")x(i)
   do j=1,i
        write(*,'(f10.5,3x)',advance="no")y(i,j)
   end do
   write(*,*)
end do
write(*,*)'-------------------------------------------------'

sum=y(1,1)
do i=2,n
   prod=1.0
   do j=2,i
      prod=prod*(u+(j-2.0))
   end do
   sum=sum+prod/IFACT(i-1)*y(i,i)
end do
write(*,*)'Approximate value=',sum
stop
end
function IFACT(i)
   f=1
   do k=1,i
      f=f*k
   end do
   return
end function
```

Divided Difference Program:

```
program divided_difference
   integer::i,j,n,fact,k
   real::x(20),y(20),d(20,20),xr(3),yr(3),p,s,h
   write(*,*)'number of values='
   read(*,*)n
   write(*,*)'tergated values xr='
   do k = 1, 3
      read(*, *) xr(k)
   end do
   do i=1,n
      write(*,*)'values of x and y='
      read(*,*)x(i),y(i)
   end do
   do i=1,n-1
      d(i,1)=y(i+1)-y(i)
   end do
   do j=2,n-1
      do i=1,n-j
         d(i,j)=d(i+1,j-1)-d(i,j-1)
      end do
   end do
   write(*,*)'difference table::-'
   write(*,*)'-----------------------------------------------'
   do i=1,n
      write(*,30)x(i),y(i),(d(i,j),j=1,n-i)
   end do
   30 format(/,2x,f12.3,2x,f12.3,10(2x,f12.3))
   do k=1,3
      yr(k)=y(1)
      h=x(2)-x(1)
      p=(xr(k)-x(1))/h
```

```
      s=p
      do i=1,n-1
         yr(k)=yr(k)+s*d(1,i)/fact(i)
         s=s*(p-i)
      end do
   end do
   write(*,*)'-----------------------------------------------'
   do k=1,3
      write(*,20)xr(k),yr(k)
   end do
   20 format(2x,'value of y(x) at x=',f12.3,1x,'is',1x,f12.3)

end program
Integer function fact(l)
fact=1
do i=1,l
   fact=fact*i
end do
return
end function
```