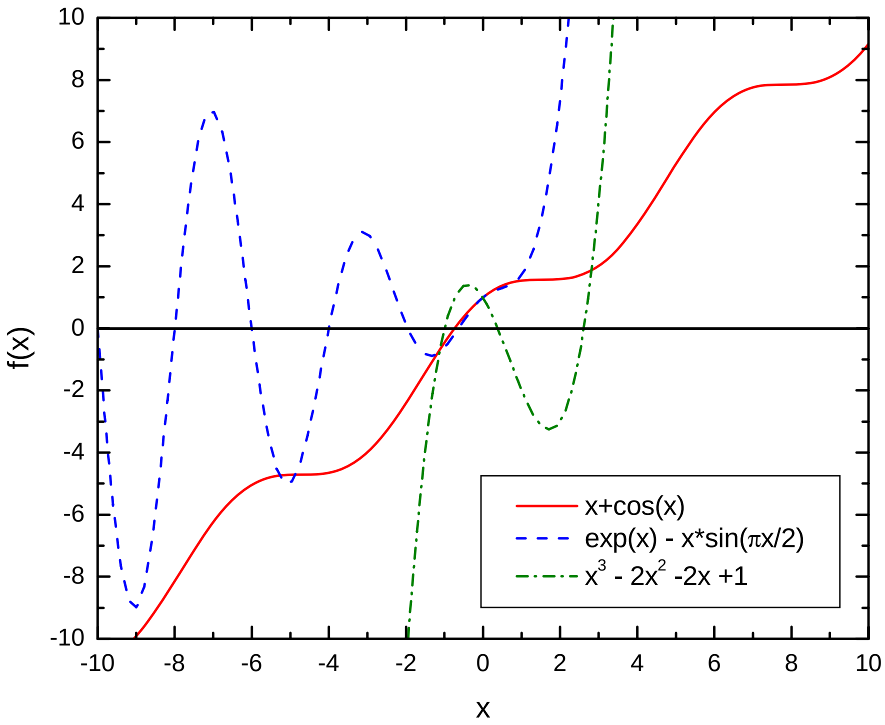
Homework 1: Nonlinear equations. (February 2, 2010)

by Alex



Question 1a: $f(x) = x + \cos(x) = 0$ interval [-10.0,10.0]

```
                 iterations     root
Bisection method         25     -0.73909
False position            7     -0.73909
Secant method             8     -0.73909
Newton method             5     -0.73909
```

Question 1b: $f(x) = \exp(x) - x\sin\left(\frac{\pi x}{2}\right) = 0$ intervals [-5.0,-2.5] and [-5.0,5.0]

```
                  iterations     root             iterations       root
Bisection method          22     -4.00290                 24       -4.00290
False position             6     -4.00290                 31       -4.00290
Secant method              5     -4.00290                  6       -0.63316
Newton method              4     -4.00290                  5       -0.63316
```

Question 1c): $f(x) = x^3 - 2x^2 - 2x + 1 = 0$ interval [0.0,2.0]

```
                  iterations     root
Bisection method          21       0.38197
False position             6       0.38197
Secant method              5       0.38197
Newton method              4       0.38197
```

Comment: for the secant and Newton's methods the initial point has been selected as (b+a)/2

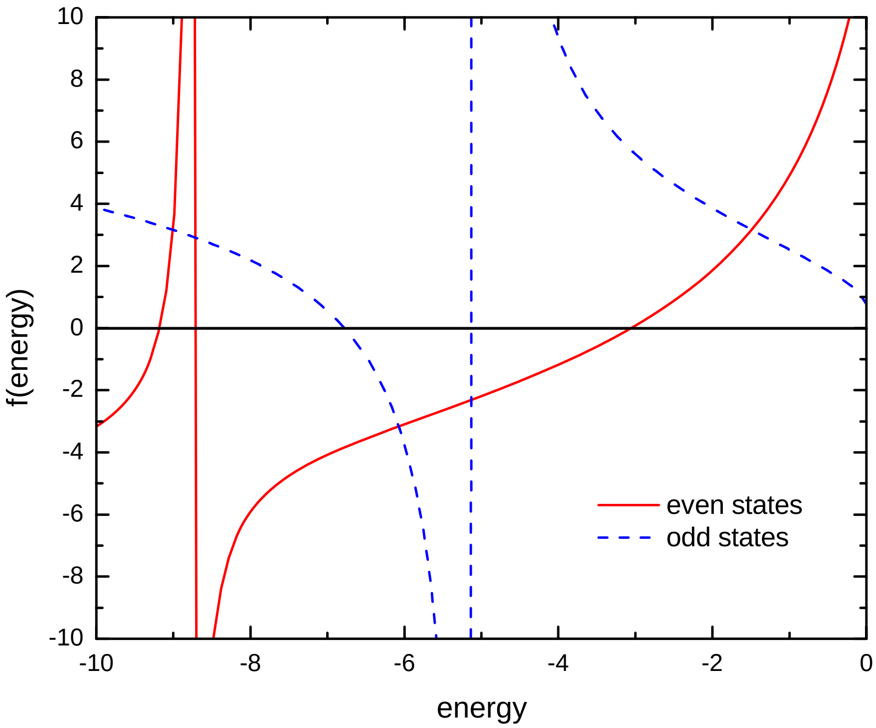
Question 2a) $x^4 - 6x^3 + 12x^2 - 36x - 18 = 0$ on [-10.0, +10.0]

```
  Brute force method
    root     value
     1     -0.42579
     2      5.15784
```

Question 2b) $f(x) = \exp(x) - x\sin\left(\frac{\pi x}{2}\right) = 0$ on [-10.0, +10.0].

```
  Brute force method
    root     value
     1    -10.00000
     2     -8.00003
     3     -5.99974
     4     -4.00290
     5     -1.95378
     6     -0.63316
```

Question 3:



```
  even       root        f(x)
     1     -9.18026    5.605E-06
     2     -8.76630   -1.532E+08    (not a root)
     3     -3.05423   -7.150E-07
   odd       root        f(x)
     1     -6.77906   -1.435E-06
     2     -5.06520   -2.136E+07    (not a root)
```

Programs:

Multiple roots of f(x) using one of the closed domain methods

Program: BForce.f90 (Fortran 90)

```
program main
!===================================================================
! Multiple roots of an equation f(x)=0 in [x1,x2] interval
! Method: Brute force + Closed Domain (Bisectional or False position)
!===================================================================
implicit none
integer, parameter:: n=1000
double precision f,x1,x2,eps
double precision Roots(n)
integer key, nroots, i
external f

key = 1
x1 = -10.0
x2 =  10.0
eps = 1.0e-7

write (*,*) ' Roots of f(x) by brute force method'
if(key == 1) write (*,*) ' Subintervals - Bisectional method'
if(key == 2) write (*,*) ' Subintervals - False position method'
write (*,100) x1, x2
write (*,101) n
write (*,102) eps

  call BForce(f,x1,x2,eps,Roots,key,n,nroots)

! print solutions
if(nroots == 0) then
  write(*,*)' no roots found'
  stop
end if
write(*,*) ' number     root       f(root)'
do i=1,nroots
  write(*,103) i, Roots(i), f(Roots(i))
end do

100 format(' interval [',f7.3,',',f7.3,']')
101 format(' subintervals= 'i12)
102 format(' tolerance   = ',1pe12.3,/)
103 format(i5,3x,f11.6,2(1pe12.3))
end program main
```

```
   Subroutine BForce(f,x1,x2,eps,Roots,key,n,nroots)
!============================================================
! Multiple roots of equation f(x)=0 on [x1,x2] interval
! Method: Brute force with one of closed domain methods
! Close domain methods: bisectional or false position
! Alex G. January 2010
!------------------------------------------------------------
! input ...
! f   - function - evaluates f(x) for any x in [x1,x2]
! x1  - left endpoint of initial interval
! x2  - right endpoint of initial interval
! eps - desired uncertainity of the root as |b-a|<eps
! key - select a method
!       1 - bisectional method
!       2 - false position method
! n   - number of subintervals for [x1,x2]
! output ...
! Root(n) - roots of the equation f(x)=0 on [x1,x2]
! nroots  - number of roots (nroots<=n)
!
! Comments:
! The program divide [x1,x2] into n subintervals
! Max number of iterations for every subinterval - 200
!====================================================================
implicit none
integer n, nroots
double precision f, x1, x2, eps, Roots(n)
double precision a, b, c, dx, root
integer i, j, key
integer, parameter:: iter=200

! initializayion
dx = (x2-x1)/real(n)
nroots = 0

! loop over subintervals
do j=1,n
  a = x1  + real(j-1)*dx
  b = a + dx
! check the closed domain condition f(a)*f(b)<0
  if(f(a)*f(b)>0) cycle

! Iterative refining the solution
  do i=1,iter
    if(key == 1) then
      c=(b+a)/2.0
      else
      c = b - f(b)*(b-a)/(f(b)-f(a))
    end if
    if(f(a)*f(c).le.0.0) then
      b = c
    else
      a=c
    end if
!   condition(s) to stop iterations)
    if(abs(b-a)<= eps) exit
  end do
! check if it is a root or singularity
  root = (b+a)/2.0
  if (abs(f(root)) < 1.0) then
    nroots = nroots+1
    Roots(nroots)=root
  end if
end do
end subroutine BForce

!-----------------------------------------------------
  Function f(x)
implicit none
double precision f, x
double precision, parameter::pi=3.141592654
  f = exp(x)-x*sin(pi*x/2.0)
! physics 1
!double precision U, a
!U =10.0
!a = 1.0
!f = sqrt(U-abs(x))*tan(sqrt(2.0*(U-abs(x))*a*a))-sqrt(abs(x))
end function f
```

Sample output:

```
 Roots of f(x) by brute force method
 Subintervals - Bisectional method
 interval [-10.000, 10.000]
 subintervals=         1000
 tolerance   =    1.000E-07

  number     root       f(root)
    1     -9.999997   4.797E-07
    2     -8.000027   3.605E-07
    3     -5.999737  -1.354E-07
    4     -4.002904  -1.030E-08
    5     -1.953775  -5.007E-08
    6     -0.633156   4.906E-08
```