```
input a, b, and (a pointer to) a function for f
calculate f(a)
calculate f(b)
if f(a)f(b) ≥ 0 then
    exit function because the root is not bracketed.
end if
if |f(a)| < |f(b)| then
    swap (a,b)
end if
c := a
set mflag
repeat until f(b or s) = 0 or |b − a| is small enough (convergence)
    if f(a) ≠ f(c) and f(b) ≠ f(c) then
        s := a f(b) f(c) / ((f(a)−f(b))(f(a)−f(c))) + b f(a) f(c) / ((f(b)−f(a))(f(b)−f(c))) + c f(a) f(b) / ((f(c)−f(a))(f(c)−f(b)))  (inverse quadratic interpolation)
    else
        s := b − f(b) (b−a)/(f(b)−f(a))  (secant method)
    end if
    if (condition 1) s is not between (3a + b)/4 and b or
       (condition 2) (mflag is set and |s−b| ≥ |b−c|/2) or
       (condition 3) (mflag is cleared and |s−b| ≥ |c−d|/2) or
       (condition 4) (mflag is set and |b−c| < |δ|) or
       (condition 5) (mflag is cleared and |c−d| < |δ|) then
        s := (a+b)/2  (bisection method)
        set mflag
    else
        clear mflag
    end if
    calculate f(s)
    d := c  (d is assigned for the first time here; it won't be used above on the first iteration because
mflag is set)
    c := b
    if f(a)f(s) < 0 then
        b := s
    else
        a := s
    end if
    if |f(a)| < |f(b)| then
        swap (a,b)
    end if
end repeat
output b or s (return the root)
```