# A
# Presentation
# on
# The *Bisection* Method

**Presented by:**
**Dr. Deepshikha**
**Assistant Prof.**
**Department of Physics**
**HMV College, Jalandhar**

# Introduction

- Bisection Method:

> • Bisection Method = a numerical method in Mathematics to find a root of a given *function*

# Introduction (cont.)

- *Root* of a function:

  - Root of a function $f(x)$ = a value $a$ such that:

    - $f(a) = 0$

# Introduction (cont.)

- Example:

Function: $f(x) = x^2 - 4$

Roots: $x = -2, \; x = 2$

Because:

$$f(-2) = (-2)^2 - 4 = 4 - 4 = 0$$
$$f(2) = (2)^2 - 4 = 4 - 4 = 0$$

# A Mathematical Property

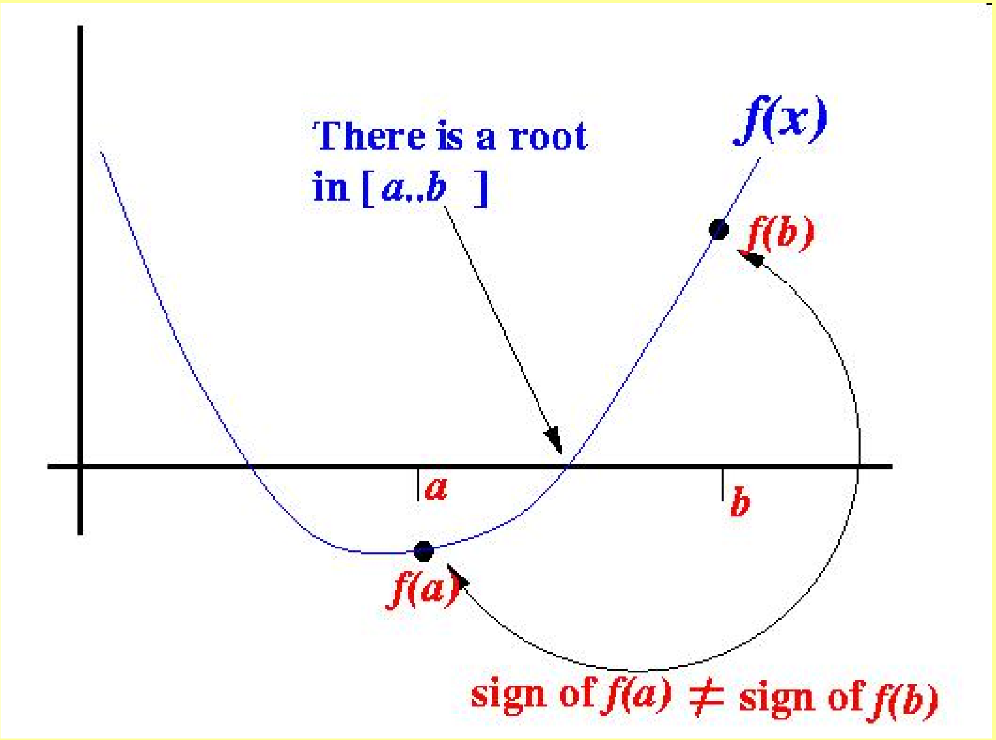
- Well-known Mathematical Property:

  - If a function $f(x)$ is continuous on the interval $[a..b]$ and sign of $f(a) \neq$ sign of $f(b)$, then

    - There is a value $c \in [a..b]$ such that: $f(c) = 0$ I.e., there is a root $c$ in the interval $[a..b]$

# A Mathematical Property (cont.)

- Example:



# The *Bisection* Method

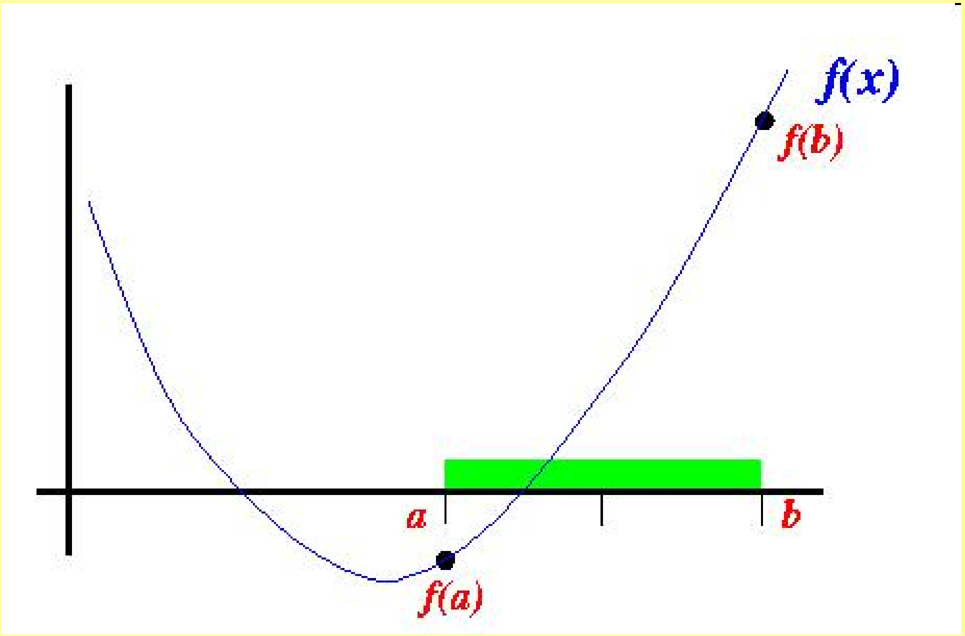
- The Bisection Method is a *successive* approximation method that narrows down an interval that contains a root of the function *f(x)*
- The Bisection Method is *given* an initial interval [*a*..*b*] that contains a root (We can use the property sign of *f(a)* ≠ sign of *f(b)* to find such an initial interval)
- The Bisection Method will *cut the interval* into 2 halves and check which half interval contains a root of the function
- The Bisection Method will keep *cut the interval* in halves until the resulting interval is extremely small

  The root is then *approximately equal* to *any value* in the final (very small) interval.

# The *Bisection* Method (cont.)

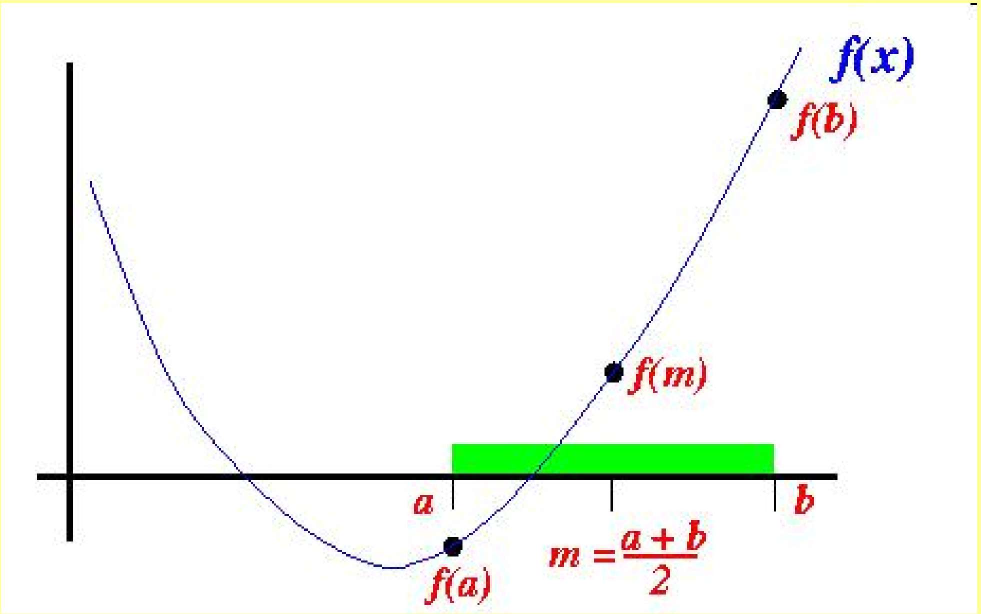
- Example:

• Suppose the interval $[a..b]$ is as follows:



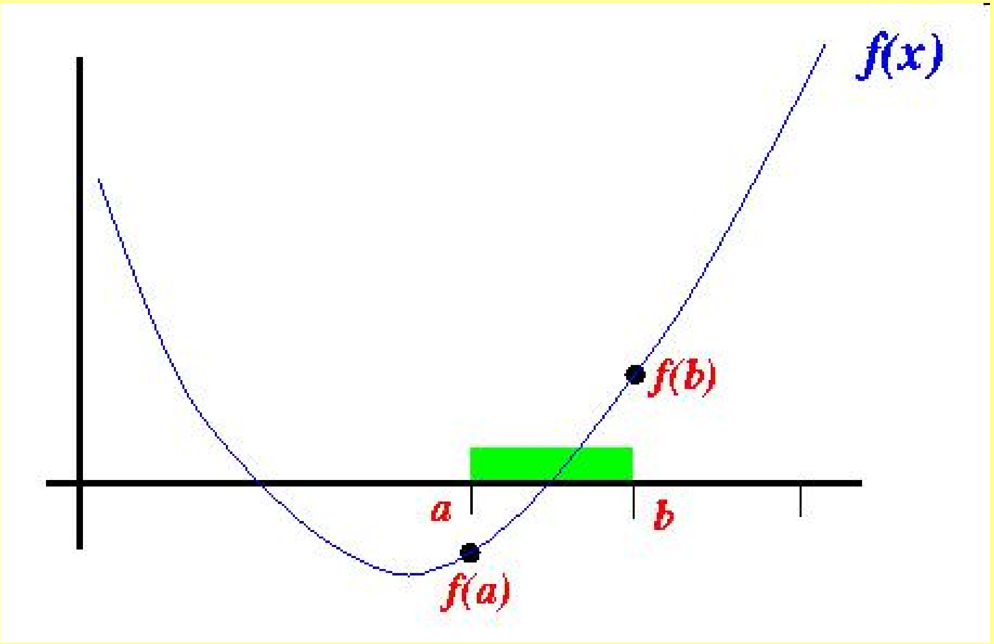
# The *Bisection* Method (cont.)

- We cut the interval [*a*..*b*] in the middle: $m = (a+b)/2$



# The *Bisection* Method (cont.)

- Because sign of *f(m)* ≠ sign of *f(a)* , we *proceed* with the search in the *new* interval [*a*..*b*]:



# The *Bisection* Method (cont.)

We can use this statement to change to the new interval:

```
b = m;
```

# The *Bisection* Method (cont.)

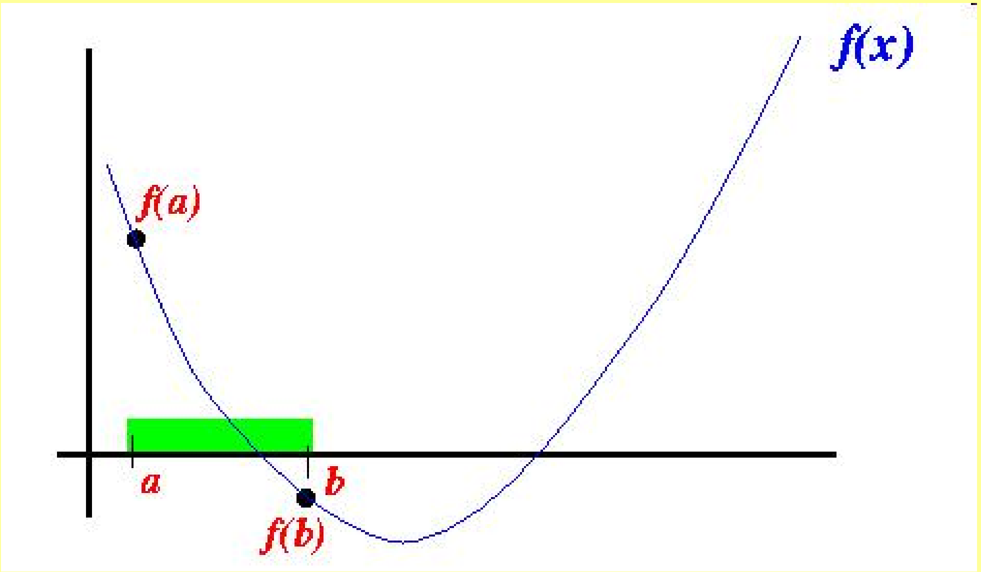
- In the above example, we have changed the end point $b$ to obtain a smaller interval that still contains a root

  In other cases, we may need to changed the end point $b$ to obtain a smaller interval that still contains a root

# The *Bisection* Method (cont.)

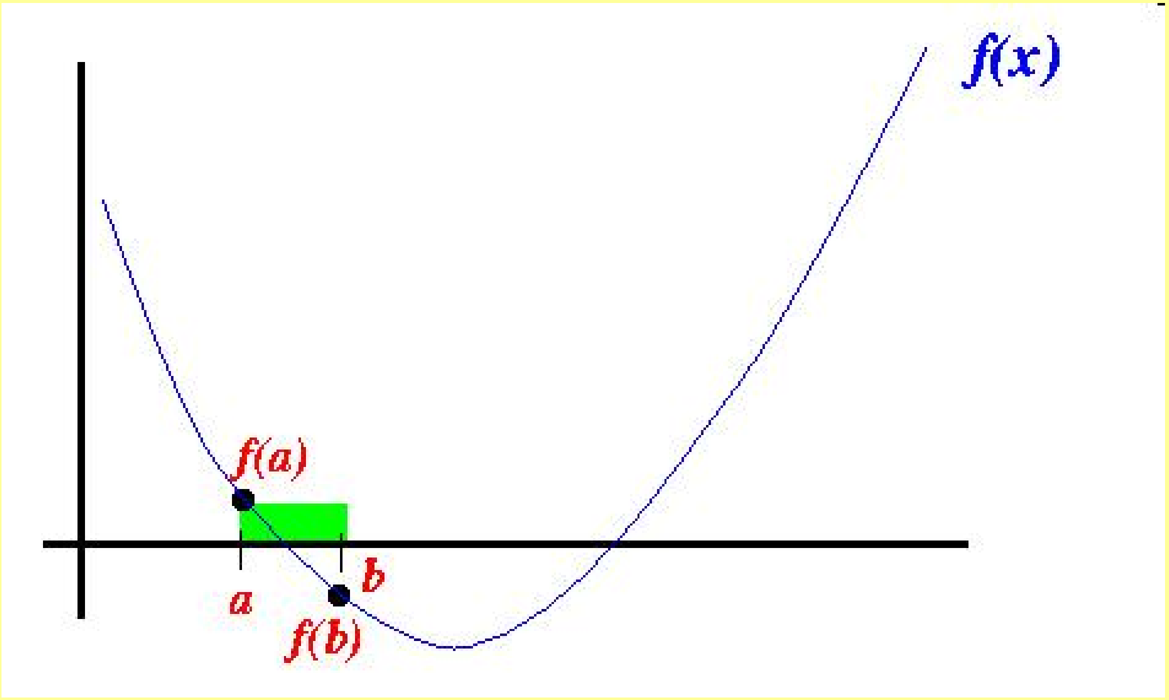
- Here is an example where you have to change the end point $a$:

• Initial interval $[a..b]$:



# The *Bisection* Method (cont.)

- After cutting the interval in half, the root is contained in the right-half, so we have to change the end point *a*:



# The *Bisection* Method (cont.)

- Rough description (pseudo code) of the Bisection Method:

```
Given: interval [a..b] such that: sign of f(a) ≠ sign of f(b)



repeat (until the interval [a..b] is "very small")
{
       a+b
   m = -----;      // m = midpoint of interval [a..b]
        2

   if ( sign of f(m) ≠ sign of f(b) )
   {
      use interval [m..b]  in the next iteration
```

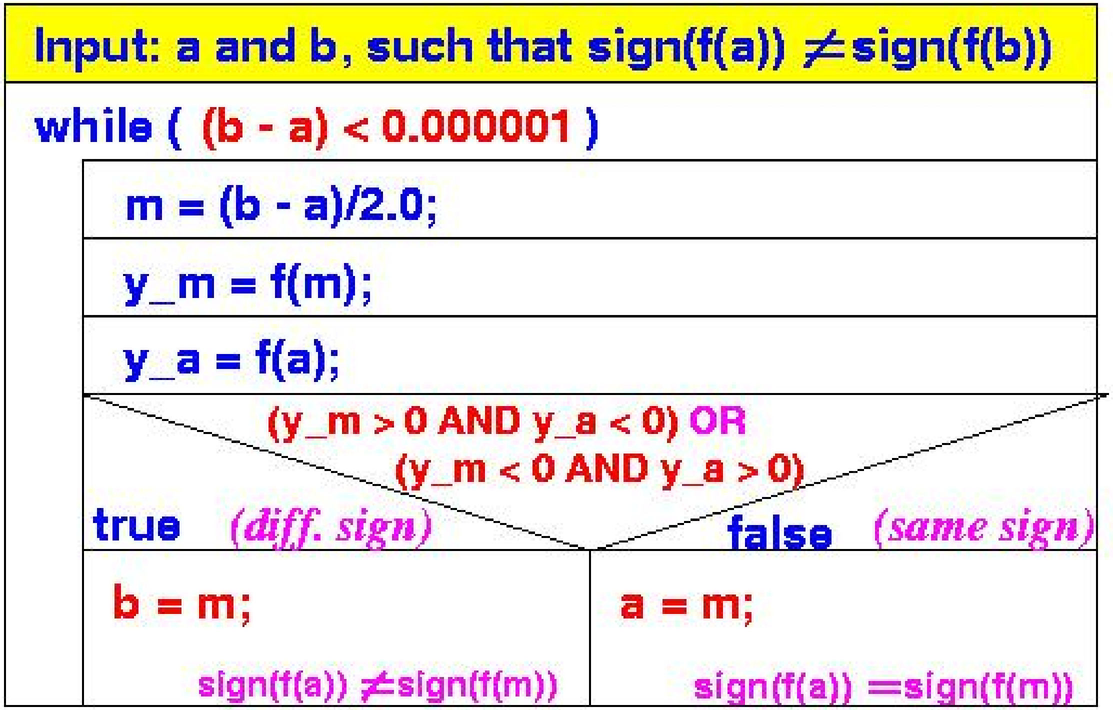
# The *Bisection* Method (cont.)

```
(i.e.: replace a with m)
     }
     else
     {
        use interval [a..m]  in the next iteration
 (i.e.: replace b with m)
     }
  }

  Approximate root = (a+b)/2;    (any point between [a..b] will do
                                  because the interval [a..b] is very small)
```

# The *Bisection* Method (cont.)

- Structure Diagram of the Bisection Algorithm:



# The *Bisection* Method (cont.)

- Example execution:

• We will use a simple function to illustrate the *execution* of the Bisection Method

• Function used:

$$f(x) = x^2 - 3$$

Roots: $\sqrt{3} = 1.7320508...$ and $-\sqrt{3} = -1.7320508...$

# The *Bisection* Method (cont.)

- We will use the starting interval [0..4] since:

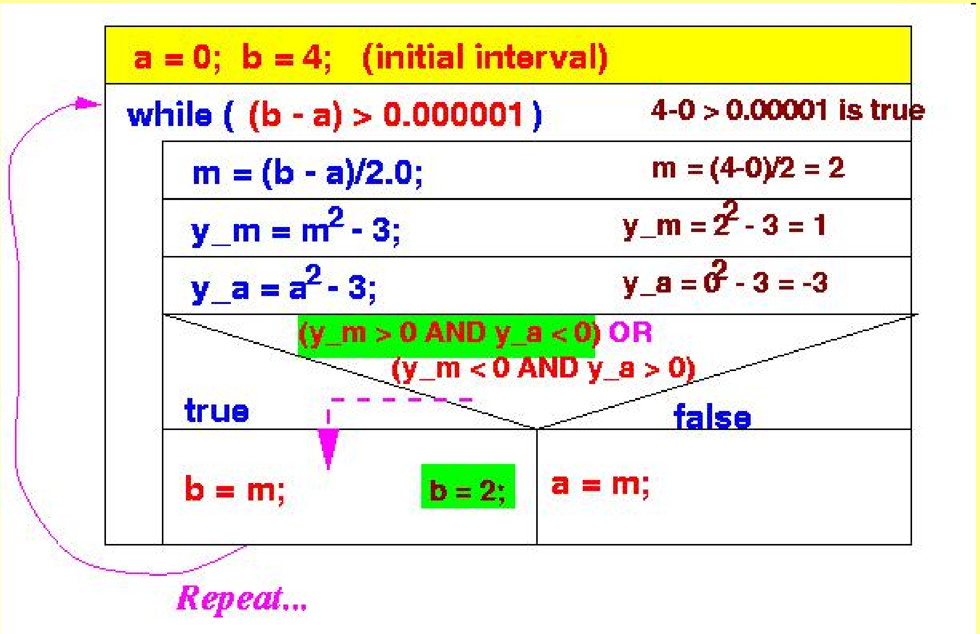
  - $f(0) = 0^2 - 3 = -3$
  - $f(4) = 4^2 - 3 = 13$

The interval [0..4] contains a root because: sign of $f(0) \neq$ sign of $f(4)$

# The *Bisection* Method (cont.)

- Steps taken by the *Bisection Method*:

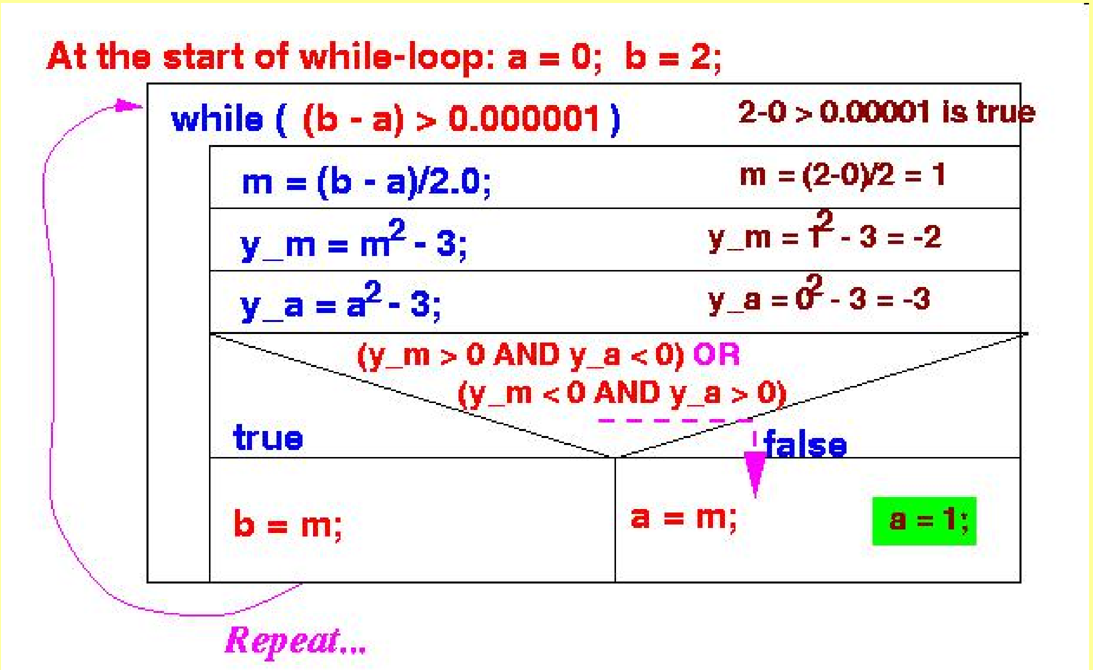
• Iteration 1:




*New* interval: [0..2] (it contains $\sqrt{3} = 1.7320508..$ !!!)

# The *Bisection* Method (cont.)

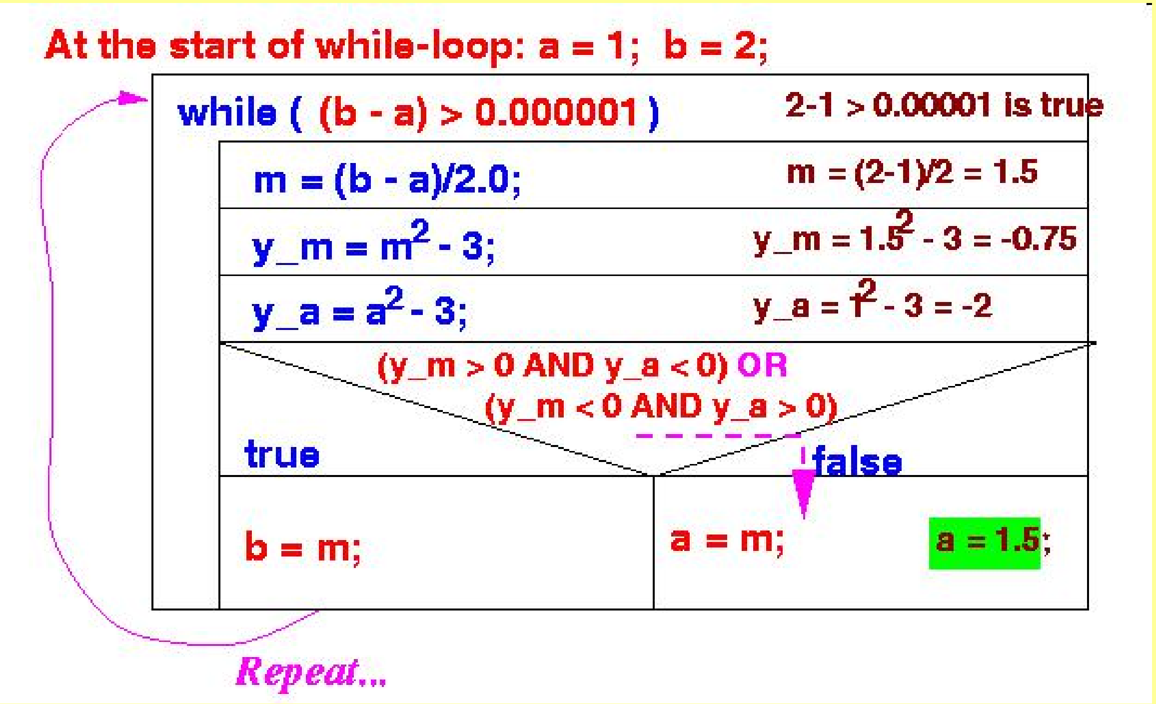
- Iteration 2:




*New* interval: [1..2] (it contains $\sqrt{3} = 1.7320508..$ !!!)

# The *Bisection* Method (cont.)

- Iteration 3:




*New* interval: [1.5 .. 2] (it contains √3 = 1.7320508.. !!!)

# The *Bisection* Method (cont.)

- And so on

Result:

- The interval gets smaller and smaller
- But it will *always* contain the root √3
- When the interval is smaller than 0.000001, the while-loop will exit

At that moment, the end points of the interval will be *very* close to root √3