**Program:-1**

**AIM-** To Find the Roots of Non-Linear Equations Using Bisection Method

The simplest root-finding algorithm is the bisection method. Let f be a continuous function, for which one knows an interval [a, b] such that f(a) and f(b) have opposite signs. Let c = (a +b)/2 be the middle of the interval (the midpoint or the point that bisects the interval).Then either f(a) and f(c), or f(c) and f(b*)* have opposite signs, and one has divided by two the size of the interval.

**Features of Bisection Method:**

* Type – closed bracket
* No. of initial guesses – 2
* Convergence – linear
* Rate of convergence – slow but steady
* Accuracy – good
* Programming effort – easy
* Approach – middle point

**Algorithm:**

Start

1. Decide initial values for x1 and x2 and stopping criterion, E.
2. Compute **f1 = f(x1)** and **f2 = f(x2)**.
3. If **f1 \* f2>0**, x1 and x2 do not bracket any root and go to step 7  
   Otherwise continue.
4. Compute **x0 = (x1+x2)/2** and compute **f0 = f(x0)**
5. If **f1\*f0 < 0** then  
   **set x2 = x0**  
   else  
   **set x1 = x0**  
   **set f1 = f0**
6. If absolute value of **(x2 – x1)/x2** is less than error E, then  
   **root = (x1 + x2)/2**  
   write the value of root  
   go to step 7  
   else  
   go to step 4
7. Stop.

**Flow Chart:**

**a=xl b=xg**

**b = (a+c)/2**

**f(a) \* f(b) < 0 ?**

**No**

**Yes**

**a = b**

**c = b**

**|a – c| < e ?**

**No**

**Output: Root= b**

**FIG:** Flowchart representing Bisection method

**CODE**

#include<stdio.h>

#include<math.h>

float fun (float x)

{

    return (x\*x\*x - 4\*x - 9);

}

void bisection (float \*x, float a, float b, int \*itr)

/\* this function performs and prints the result of one iteration \*/

{

    \*x=(a+b)/2;

    ++(\*itr);

    printf("Iteration no. %3d X = %7.5f\n", \*itr, \*x);

}

void main ()

{

    int itr = 0, maxmitr;

    float x, a, b, allerr, x1;

    printf("\nEnter the values of a, b, allowed error and maximum iterations:\n");

    scanf("%f %f %f %d", &a, &b, &allerr, &maxmitr);

    bisection (&x, a, b, &itr);

    do

    {

        if (fun(a)\*fun(x) < 0)

            b=x;

        else

            a=x;

        bisection (&x1, a, b, &itr);

        if (fabs(x1-x) < allerr)

        {

            printf("After %d iterations, root = %6.4f\n", itr, x1);

            return 0;

        }

        x=x1;

    }

    while (itr < maxmitr);

    printf("The solution does not converge or iterations are not sufficient");

    return 1;

}

**Output**

A sample run of the program was carried out and the results were found as:-

This program illustrates the bisection method in C

**x^3 + 3\*x - 5 = 0**

Enter the first approximation to the root

1

Enter the second approximation to the root

2

Enter the number of iterations you want to perform

9

The root after 1 iteration is 1.500000

The root after 2 iteration is 1.250000

The root after 3 iteration is 1.125000

The root after 4 iteration is 1.187500

The root after 5 iteration is 1.156250

The root after 6 iteration is 1.146025

The root after 7 iteration is 1.148438

The root after 8 iteration is 1.152344

The root after 9 iteration is 1.154297

The root is **1.154297**