```c
/* Mwanje Isma sserunkuma  20/U/5700/EVE 2000705700
/* Kakande Abdullatiff Musa  20/U/13543/EVE  2100713543
/* Adoch Teopista - 21/U/12898/PS - 2100712898
 */

//Cprogram to illustrate the bisection method of root finding
//square of 2 root is 1.414 after 20 iterations, initial guesses 1, 2
//7th root of 126 is 1.996 after 20 iterations initial guesses 1, 2
//cube root of 43 is 3.504 after 20 iterations initial guesses 3, 4
//5th root of 729 is 3.737 after 20 iterations initial guesses 3, 4

#include<stdio.h>
#include<math.h>

float f( float x); //polinomial function
float bisect(float z, float l, float r, int ctr, int iter);//bisection function to return root

int main()
{
    int ctr = 1, iter; //iter is the number of iterations needed for the root approximation
    float z, l, r, t; //l and r are two initial guesses and z is their mid-point/average t is the root

    printf("C program illustrates the bisection method:\n\n");
    printf(" change the retun statement in function func for a polinomial whose root is required\n\n");

    printf("\nEnter the first initial guess to the root :  ");
    scanf("%f",&l);

    printf("\n\nEnter the second initial guess to the root :   ");
    scanf("%f",&r);

    printf("\n\nEnter the number of iterations you want to perform :   ");
    scanf("%d",&iter);



    //We check if the initial approximations are the root or not
    if(f(l) == 0)// it is a root
        z = l;
    else if(f(r) == 0)
        z = r;
    else //If the above two values are not the roots of the given function
        t = bisect(z, l, r, ctr, iter);

    printf("\n\nThe approximation to the root is %.3f\n",t);
    return 0;
}

float f( float x) //polinomial function definition
{
    return (pow(x, 3) - 4); //returns the value of the function given x value
}

float bisect(float z, float l, float r, int ctr, int iter)
{
    while(ctr <= iter) //Since, ctr is initialized to 1
        {
            z = (l + r) / 2.0;

            if(f(z) == 0)
            {
                z = f(z);
                break; //stop and Return value of z as the root approximation
            }
            printf("The root after %d iteration is %f \n\n",ctr, z);

            if(f(l) * f(z) < 0)//Both are of opposite sign
                r = z;
            else if(f(r) * f(z) <0)
                l=z;
            ctr++;
        }

        return z;
}
```