```
/*Usage: For finding the real roots of any given Transcendental equation.

Specification: The program takes coeffients of the equation and upper and lower values of t
he interval as the input and computes the root for the given equation using Muller's method
.

In this method, f(x) is approximated by a second degree curve in the vicinity of a root. Th
e roots of the quadratic are then assumed to be the approximations to the roots of the equa
tion f(x) = 0.
The method is iterative, converges almost quadratically, and can be used
to obtain complex roots.
*/

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

//Function prototypes
float muller1(float a, float b, float c);
float f(float x);

int main(int argc, char **argv)
{
   float a, b, c, root;

   if (argc != 4)  //Verification of arguments
   {
      fprintf(stderr, "Usage: %s <approximates to the root>\n", argv[0]);
      exit(1);
   }

   //Getting the values of coefficients
   a = atof(argv[1]);
   b = atof(argv[2]);
   c = atof(argv[3]);

   root = muller1(a, b, c); //Calling Function
   printf("Root of the given equation is: %f\n", root);

   exit(0);
}

float muller1(float a, float b, float c)
{
   float h_b, h_c, lamk, sigk, gk, ck, lamda, lamda1;
   float val, val1, xk;

   h_c = (c - b), h_b = (b - a), lamk = h_c/h_b, sigk = lamk + 1;

   while (1)
   {
      gk = ((lamk*lamk*f(a))-(sigk*sigk*f(b))+((lamk+sigk)*f(c)));  //Value of gk
      ck = lamk*(((lamk*f(a))-(sigk*f(b))+f(c)));  //Value of ck

      val = (gk*gk - 4*sigk*ck*f(c));
      val1 = sqrtf(val);

      lamda = -2*sigk*f(c) / (gk-val1);  //Value of lamda1
      lamda1 = -2*sigk*f(c) / (gk+val1);  //Value of lamda2

      //Checking the convergence of the equation
      if (floor(gk*10000) == floor(val1*10000))
      {
         return c;
      }

      if (lamda1 < lamda)
      {
         lamda = lamda1;
      }

      xk = c + lamda*(c-b);

      a = b;
      b = c;
      c = xk;

      if (floor(c*10000) == floor(b*10000)) //Comparing the roots
      {
         return c;
      }
   }
}

float f(float x)
{
   float ans;

   ans = x*log10f(x) - 1.2;  // Function Equation

   if(ans != ans)
   {
       printf("Cannnot proceed further..Try changing the values\n");
       exit (2);
   }
   return ans;
}
```