Source Code:

//Secant (a)

#include <stdio.h.>

#include <math.h>

#include <stdlib.h>

float f (float x)

{

return (x\*x - 4\*x - 10);

}

float absolute(float n)

{

return(n < 0 ? -1.0 \* n : n);

}

float Secant(float x0, float x1, float e)

{

float x2, error;

int iteration = 0, loop = 1;

while (loop == 1)

{

iteration ++;

x2 = x1 - ( f(x1) \* ( (x1 - x0)/( f(x1) - f(x0) ) ) );

error = absolute((x2 - x1) / x2) ;

printf("\n\nx2 = %f with E = %f, in Iteration %d", x2, error, iteration);

x0 = x1;

x1 = x2;

if(error < e)

{

loop = 0;

}

else if(iteration > 200)

{

exit(1);

}

}

return x1;

}

int main()

{

float result, a, b, e, root, slope, fa, fb;

int i;

printf("\nEnter initial range : ");

scanf("%f%f", &a, &b);

printf("\nEnter acceptable error : ");

scanf("%f", &e);

/\*

Here in the while block below we make use of slope made by the initial

guesses to determine whether to increase b or decrease a.

\*/

while(f(a)\*f(b) > 0)

{

fa = f(a);

fb = f(b);

slope = (fb - fa) / (b - a);

if(slope > 0)

{

(fa > 0 && fb > 0)? a-- : b++ ;

}

else if(slope < 0)

{

(fa > 0 && fb > 0)? b++ : a-- ;

}

else{ // deals with infinite slope or zero slope cases.

b++;

}

}

printf("\nOperting with initial assumptions : x0 = %f, x1 = %f .", a, b);

if(f(a) == 0){

root = a;

}

else if(f(b) == 0){

root = b;

}

else {

root = Secant(a, b, e);

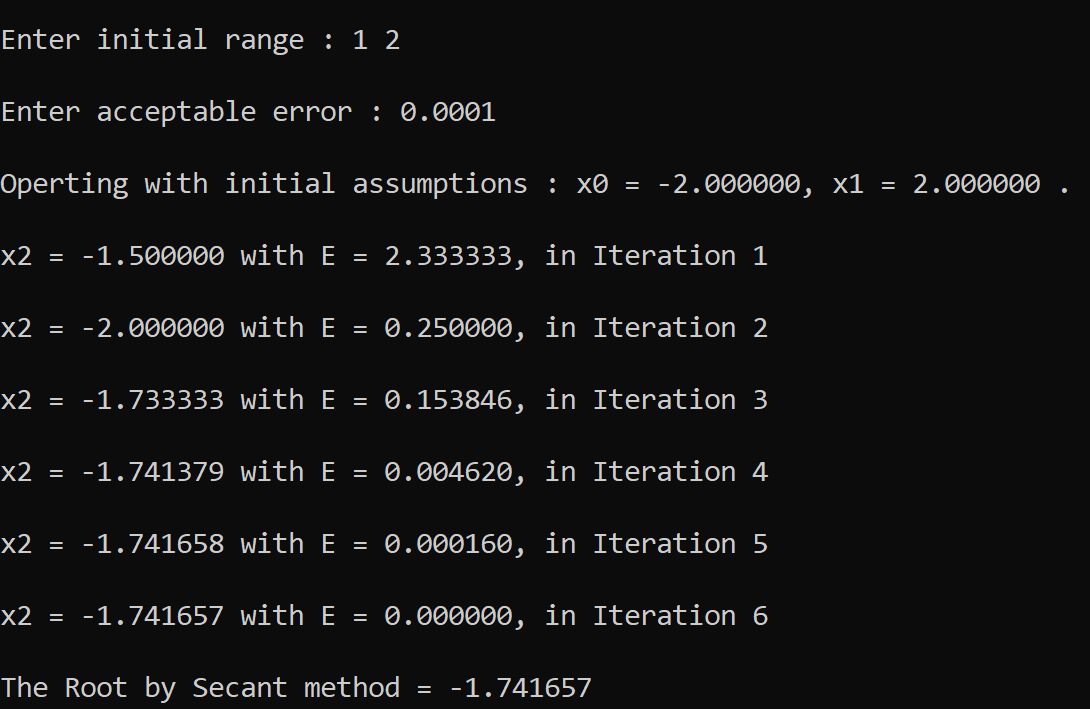
}

printf("\n\nThe Root by Secant method = %f", root);

return 0;

}

Output:



Source Code:

//Secant (b)

#include <stdio.h.>

#include <math.h>

#include <stdlib.h>

float f (float x)

{

return (4 \* sin(x) - exp(x));

}

float absolute(float n)

{

return(n < 0 ? -1.0 \* n : n);

}

float Secant(float x0, float x1, float e)

{

float x2, error;

int iteration = 0, loop = 1;

while (loop == 1)

{

iteration ++;

x2 = x1 - ( f(x1) \* ( (x1 - x0)/( f(x1) - f(x0) ) ) );

error = absolute((x2 - x1) / x2) ;

printf("\n\nx2 = %f with E = %f, in Iteration %d", x2, error, iteration);

x0 = x1;

x1 = x2;

if(error < e)

{

loop = 0;

}

else if(iteration > 200)

{

exit(1);

}

}

return x1;

}

int main()

{

float result, a, b, e, root, slope, fa, fb;

int i;

printf("\nEnter initial range : ");

scanf("%f%f", &a, &b);

printf("\nEnter acceptable error : ");

scanf("%f", &e);

/\*

Here in the while block below we make use of slope made by the initial

guesses to determine whether to increase b or decrease a.

\*/

while(f(a)\*f(b) > 0)

{

fa = f(a);

fb = f(b);

slope = (fb - fa) / (b - a);

if(slope > 0)

{

(fa > 0 && fb > 0)? a-- : b++ ;

}

else if(slope < 0)

{

(fa > 0 && fb > 0)? b++ : a-- ;

}

else{ // deals with infinite slope or zero slope cases.

b++;

}

}

printf("\nOperting with initial assumptions : x0 = %f, x1 = %f .", a, b);

if(f(a) == 0){

root = a;

}

else if(f(b) == 0){

root = b;

}

else {

root = Secant(a, b, e);

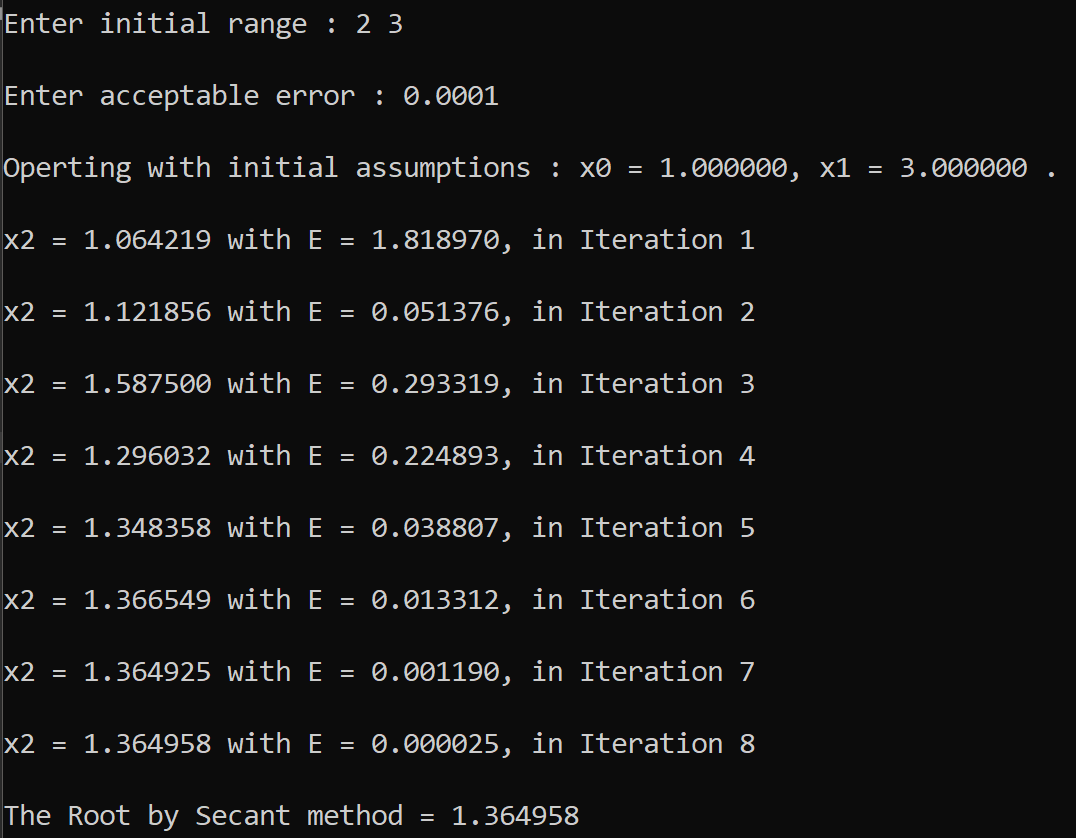
}

printf("\n\nThe Root by Secant method = %f", root);

return 0;

}

Output:



Source Code:

//Newton Raphson (a)

#include <stdio.h.>

#include <math.h>

#include <stdlib.h>

float f (float x)

{

return (x \* tan(x) - 1);

}

float f1 (float x)

{

return (x \* pow((1/cos(x)), 2) + tan(x)) ;

}

float absolute(float n)

{

return(n < 0 ? -1.0 \* n : n);

}

float NewtonRaphson(float x0, float e)

{

float x1, error;

int iteration = 0, loop = 1;

while (loop == 1)

{

iteration ++;

x1 = x0 - f(x0)/f1(x0);

error = absolute((x1 - x0) / x1) ;

printf("\n\nx1 = %f with E = %f, in Iteration %d", x1, error, iteration);

x0 = x1;

if(error < e)

{

loop = 0;

}

else if(iteration > 200){

exit(1);

}

}

return x1;

}

int main()

{

float root, a, e;

int i;

printf("\nEnter x0 : ");

scanf("%f", &a);

printf("\nEnter acceptable error : ");

scanf("%f", &e);

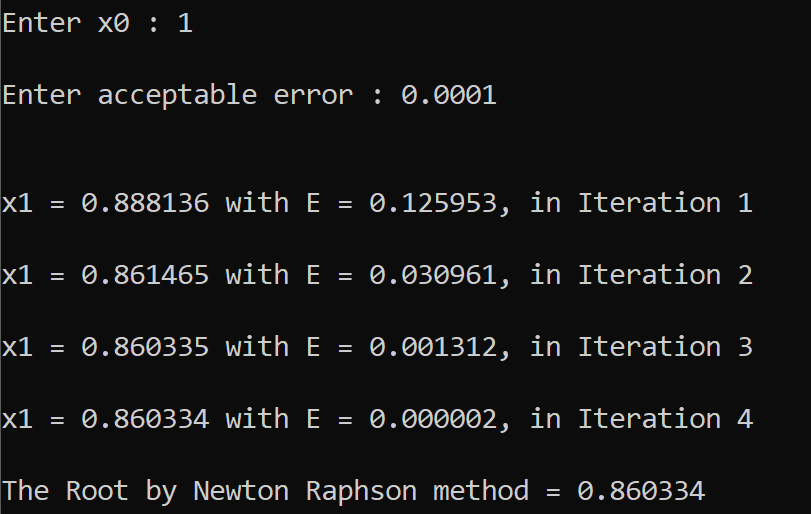
root = NewtonRaphson(a, e);

printf("\n\nThe Root by Newton Raphson method = %f", root);

return 0;

}

Output:



Source Code:

//Newton Raphson (b)

#include <stdio.h.>

#include <math.h>

#include <stdlib.h>

float f (float x)

{

return ( 3\*x + exp(x) );

}

float f1 (float x)

{

return (3 + exp(x)) ;

}

float absolute(float n)

{

return(n < 0 ? -1.0 \* n : n);

}

float NewtonRaphson(float x0, float e)

{

float x1, error;

int iteration = 0, loop = 1;

while (loop == 1)

{

iteration ++;

x1 = x0 - f(x0)/f1(x0);

error = absolute((x1 - x0) / x1) ;

printf("\n\nx1 = %f with E = %f, in Iteration %d", x1, error, iteration);

x0 = x1;

if(error < e)

{

loop = 0;

}

else if(iteration > 200){

exit(1);

}

}

return x1;

}

int main()

{

float root, a, e;

int i;

printf("\nEnter x0 : ");

scanf("%f", &a);

printf("\nEnter acceptable error : ");

scanf("%f", &e);

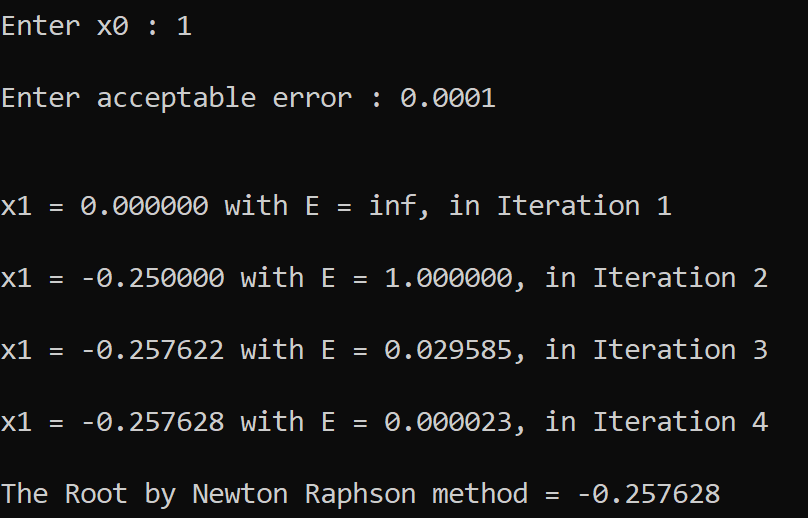
root = NewtonRaphson(a, e);

printf("\n\nThe Root by Newton Raphson method = %f", root);

return 0;

}

Output:



Source Code:

//Fixed Point Iteration (a)

#include <stdio.h.>

#include <math.h>

#include <stdlib.h>

float f (float x)

{

return ( sin(x) + 3\*x - 2 );

}

float g (float x)

{

return ( (2 - sin(x)) / 3 ) ;

}

float absolute(float n)

{

return(n < 0 ? -1.0 \* n : n);

}

float FixedPI(float x0, float e, int n)

{

float x1, error;

int iteration = 0, loop = 1;

while (loop == 1)

{

iteration ++;

x1 = g(x0);

error = absolute(f(x1)) ;

printf("\n\nx1 = %f with E = %f, in Iteration %d", x1, error, iteration);

x0 = x1;

if(error < e){

loop = 0;

}

else if(iteration > n){

printf("\nDid Not Converge.");

exit(1);

}

}

return x1;

}

int main()

{

float root, a, e;

int i, n;

printf("\nEnter x0 : ");

scanf("%f", &a);

printf("\nEnter acceptable error : ");

scanf("%f", &e);

printf("\nEnter how many iterations to check till : ");

scanf("%d", &n);

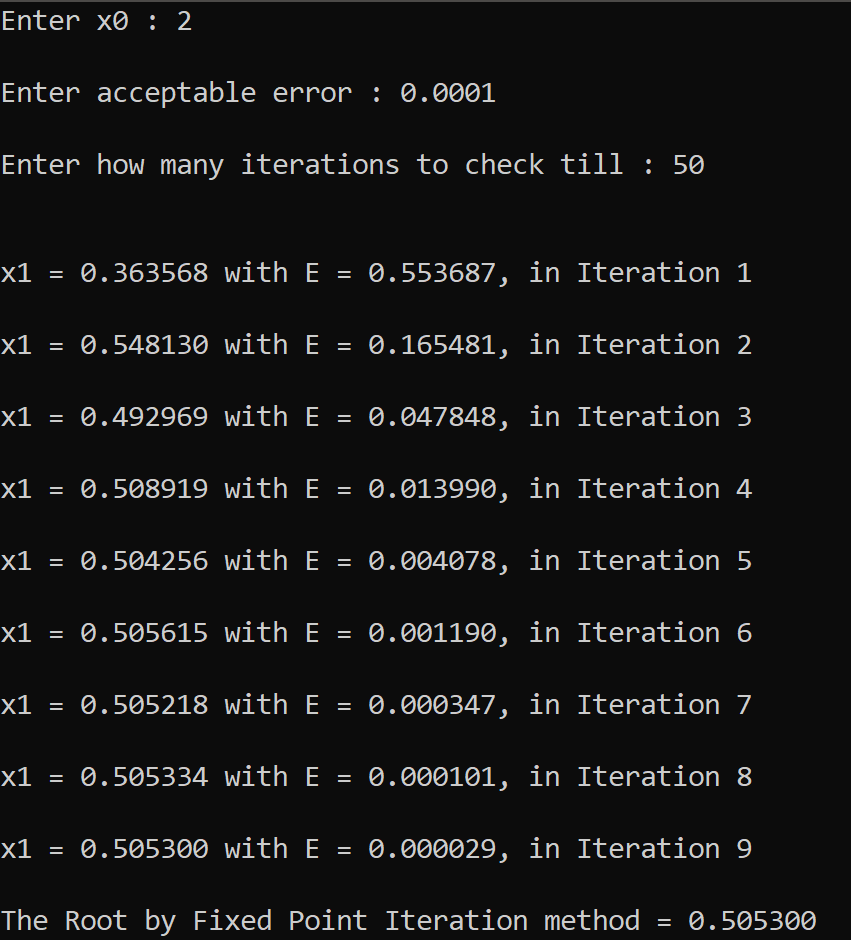
root = FixedPI(a, e, n);

printf("\n\nThe Root by Fixed Point Iteration method = %f", root);

return 0;

}

Output:



Source Code:

//Fixed Point Iteration (b)

#include <stdio.h.>

#include <math.h>

#include <stdlib.h>

float f (float x)

{

return (pow(x, 3) + pow(x, 2) - 1);

}

float g (float x)

{

//return ( 1 / (pow(x, 2) + x) );

return ( 1 / sqrt(x + 1) );

}

float absolute(float n)

{

return(n < 0 ? -1.0 \* n : n);

}

float FixedPI(float x0, float e, int n)

{

float x1, error;

int iteration = 0, loop = 1;

while (loop == 1)

{

iteration ++;

x1 = g(x0);

error = absolute(f(x1)) ;

printf("\n\nx1 = %f with E = %f, in Iteration %d", x1, error, iteration);

x0 = x1;

if(error < e){

loop = 0;

}

else if(iteration > n){

printf("\nDid Not Converge.");

exit(1);

}

}

return x1;

}

int main()

{

float root, a, e;

int i, n;

printf("\nEnter x0 : ");

scanf("%f", &a);

printf("\nEnter acceptable error : ");

scanf("%f", &e);

printf("\nEnter how many iterations to check till : ");

scanf("%d", &n);

root = FixedPI(a, e, n);

printf("\n\nThe Root by Fixed Point Iteration method = %f", root);

return 0;

}

Output:

