**University of Calcutta**

**B.A/BSc Honours Semester VI Examination 2023 under CBCS**

**Session:2020-2023**

**Paper: CC14 (Practical)**

**Numerical Methods**

**Paper Code: MTMA-CC-6-14-P**

**CU Roll Number: 203034-11-0103**

**CU Registration Number:**

**034-1215-0354-20**

**Index**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl No** | **Date** | **Program** | **Page No** |
| 1 | 22/2/23 | Sorting an array in ascending order | 3-4 |
| 2 | 23/2/23 | Sum of Series 1 | 5 |
| 3 | 23/2/23 | Sum of Series 2 | 6 |
| 4 | 24/2/23 | Sum of Series 3 | 7 |
| 5 | 15/3/23 | Computation of real root using Bisection method | 8-9 |
| 6 | 16/3/23 | Computation of real root using Newton Raphson method | 10-11 |
| 7 | 16/3/23 | Computation of double root using Newton Raphson method | 12-13 |
| 8 | 17/3/23 | Computation of root using Secant method | 14-15 |
| 9 | 22/3/23 | Computation of root using Regula Falsi method | 16-17 |
| 10 | 23/3/23 | Lagrange’s Interpolation | 18-19 |
| 11 | 24/3/23 | Newton’s Forward Interpolation | 20-22 |
| 12 | 24/3/23 | Newton’s Backward Interpolation | 23-26 |
| 13 | 29/3/23 | Newton’s Divided Difference Interpolation | 27-28 |
| 14 | 31/3/23 | Computation of Integral by Trapezoidal Rule | 29-30 |
| 15 | 31/3/23 | Computation of Integral by Simpson’s 1/3 Rule | 31-32 |
| 16 | 5/4/23 | Computation of Integral by Weddle Rule | 33-34 |
| 17 | 6/4/23 | Computation of Integral by Gauss Quadrature Formula | 35-36 |
| 18 | 12/4/23 | Gauss Elimination Method | 37-39 |
| 19 | 13/4/23 | Gauss-Jacobi Method | 40-41 |
| 20 | 19/4/23 | Gauss-Seidel Method | 42-43 |
| 21 | 20/4/23 | LU Decomposition Method | 44-46 |
| 22 | 21/4/23 | Computation of Eigen value and Eigen vector by Power Method | 47-49 |
| 23 | 26/4/23 | Straight Line Fitting | 50-51 |
| 24 | 27/4/23 | Parabola Fitting | 52-55 |
| 25 | 28/4/23 | Curve Fitting | 56-57 |
| 26 | 4/5/23 | Euler’s Method | 58 |
| 27 | 10/5/23 | Modified Euler’s Method | 59-60 |
| 28 | 11/5/23 | 4th Order Runge-Kutta Method | 61-62 |
| 29 | 12/5/23 | Picard’s Method | 63-64 |

**Program 1:** Write a program in C to sort an array in ascending order.

**Source Code:**

#include<stdio.h>

int main()

{

int n,i,j,var,arr[100];

printf("\n Enter the number of arrays: ");

scanf("%d",&n);

printf("\n Enter the integers:\n ");

for(i=0;i<n;i++)

{

scanf("%d",&arr[i]);

}

for(i=0;i<n-1;i++)

{

for(j=0;j<n-1;j++)

{

if(arr[j]>arr[j+1])

{

var=arr[j];

arr[j]=arr[j+1];

arr[j+1]=var;

}

}

}

printf("\n The sorted array is:\n ");

for(i=0;i<n;i++)

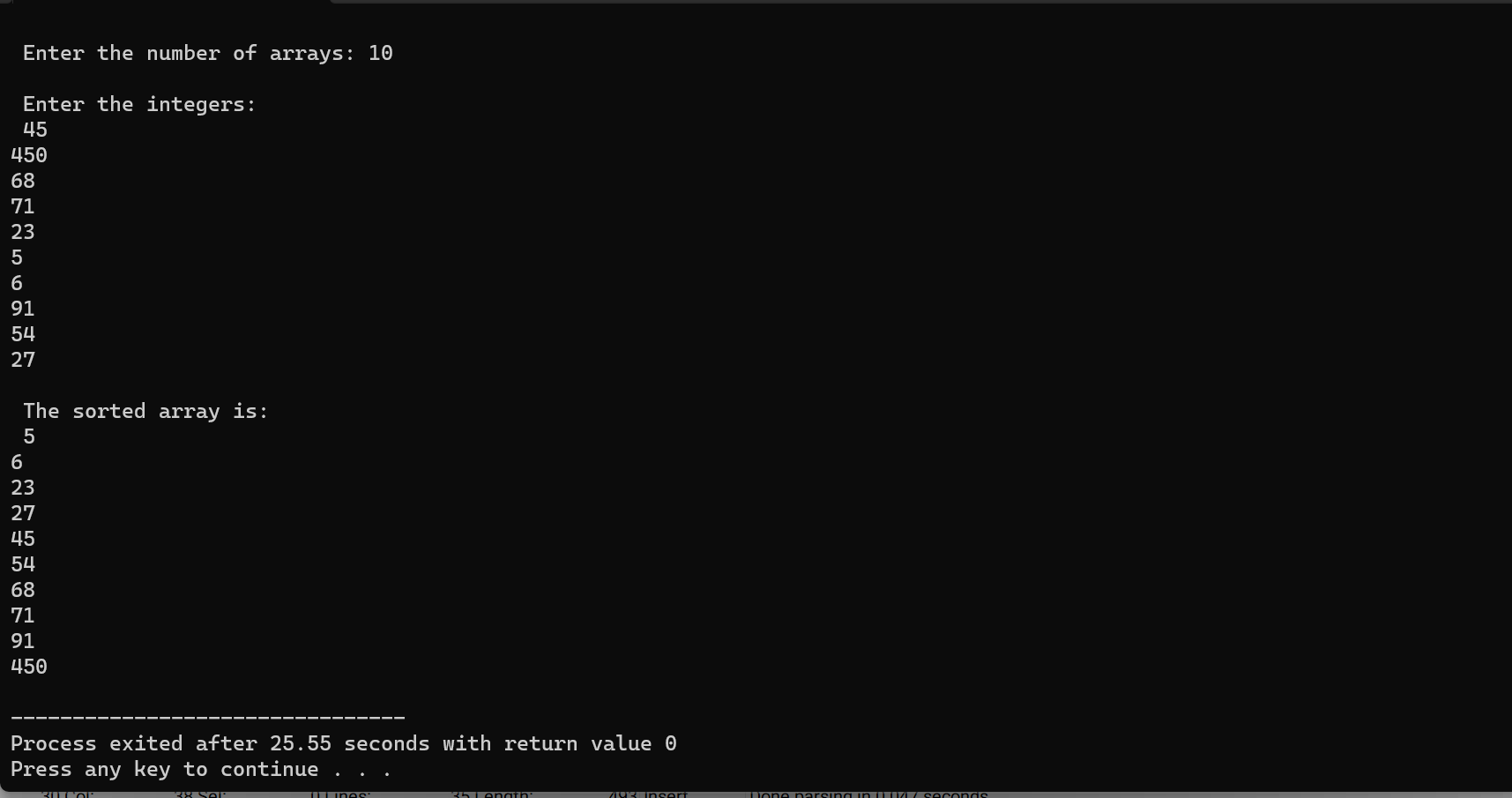
printf("%d\n",arr[i]);

return 0;

1

}

**Output:**



**Program 2:** Write a program in C to print the sum of the series: .

**Source Code:**

#include<stdio.h>

#include<math.h>

int main()

{

int i, n, sum=0;

printf("\n Enter the value of n: ");

scanf("%d",&n);

for(i=1;i<=n;i++)

{

sum=sum+pow(i,2);

}

printf("\n Required sum is: %d",sum);

return 0;

}

**Output:**



**Program 3:** Write a program in C to print the sum of the series: .

**Source Code:**

#include<stdio.h>

#include<math.h>

int main()

{

int i, n;

float sum=0.0;

printf("\n Enter the value of n: ");

scanf("%d",&n);

for(i=1;i<=n;i++)

{

sum=sum+pow(i,-2);

}

printf("\n Required sum is: %f",sum);

return 0;

}

**Output:**



**Program 4:** Write a program in C to print the sum of the series: .

**Source Code:**

#include<stdio.h>

#include<math.h>

float recursion(float a)

{

if (a>=1)

return a\*recursion(a-1);

else

return 1;

}

int main()

{

int i, n;

float sum=1.0;

printf("\n Enter the value : ");

scanf("%d",&n);

for(i=1;i<=n;i++)

{

sum+=1/recursion(i);

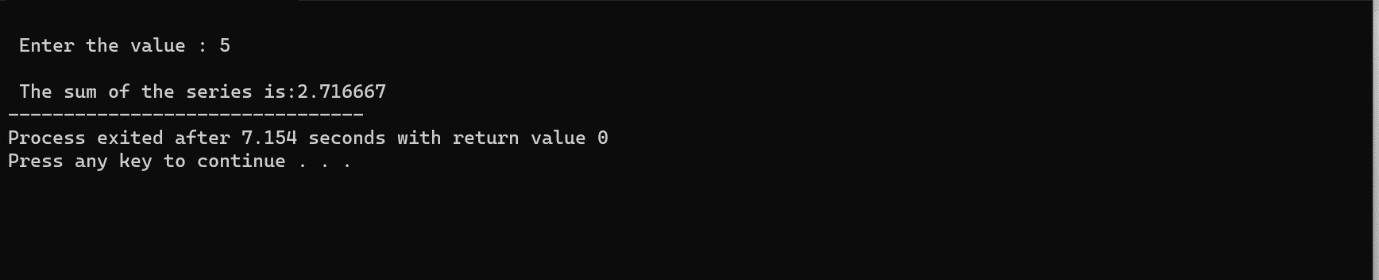
}

printf("\n The sum of the series is:%f",sum);

return 0;

}

**Output:**



**Program 5:** Write a program in C to compute real root of using Bisection method.

**Source Code:**

#include<stdio.h>

#include<math.h>

#define f(y) y+log10(y)-2

int main()

{

float a,b,x,e;

int i=0;

double fa,fb,fx,t;

/\*The double in C is a data type that is used to store high-precision

floating-point data or numbers (up to 15 to 17 digits).\*/

printf("\n Enter value of a0: ");

scanf("%f",&a);

/\*root will lie between two points a0 and b0 \*/

printf("\n Enter value of b0: ");

scanf("%f",&b);

printf("\n Enter tolerable error: ");

scanf("%f",&e);

printf("\n");

printf("\n n a \t b \t fa \t fb \t fx ");

printf("\n");

do{

x=(a+b)/2;

fa=f(a);

fb=f(b);

fx=f(x);

printf("\n %d \t %f \t %f \t %f \t %f \t %f",i,a,b,x,fa,fb,fx);

if(fa\*fx<0){

b=x;

}else{

a=x;

}

i++;

}while(fabs(fx)>e); //fabs:absolute value

printf("\n");

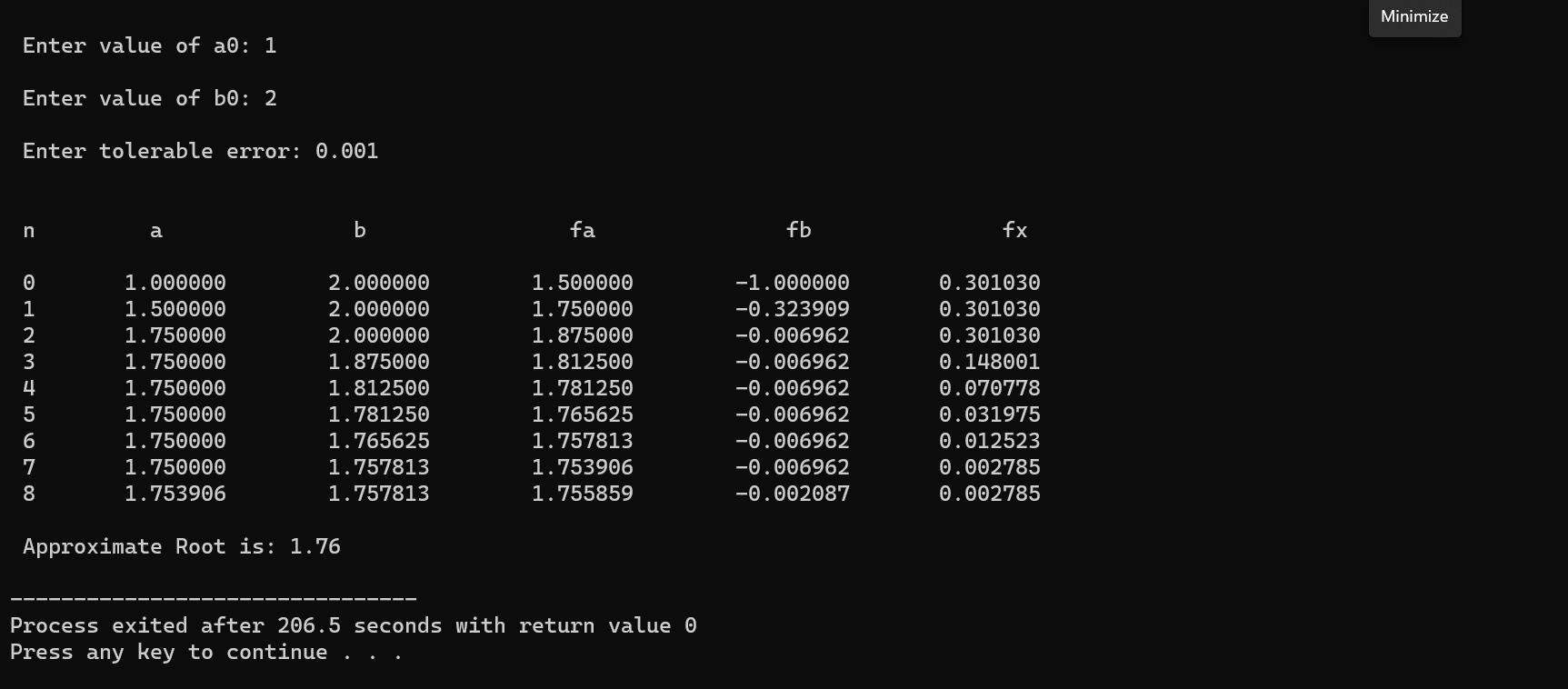
printf("\n Approximate Root is: %0.2f",x);

printf("\n");

return 0;

}

**Output:**



**Program 6:** Write a program in C to compute real root of between 3 and 4, by Newton Raphson Method.

**Source Code:**

#include <stdio.h>

#include <math.h>

#define f(x) pow(x,3)-(8\*x)-4

#define dof(x) (3\*pow(x,2))-8

int main()

{

float xn,hn,e,fxn,der;

int i=0;

printf("\n First Approximation, xn: ");

scanf("%f",&xn);

printf("\n Enter tolerable error: ");

scanf("%f",&e);

do

{

fxn=f(xn);

der=dof(xn);

hn=-1\*fxn/der;

xn=xn+hn;

i++;

}while(fabs(hn)>e);

printf("Approximate Root: %f",xn);

return 0;

}

**Output:**



**Program 7**: Write a program in C to compute the double root of using Newton Raphson method choosing

**Source Code:**

#include <stdio.h>

#include <math.h>

#define f(x) pow(x,3)-(7\*pow(x,2))+(16\*x)-12

#define dof(x) (3\*pow(x,2))-(14\*x)+16

int main()

{

float xn,hn,e,fxn,der;

int i=0,m=2;

printf("\n First Approximation, xn: ");

scanf("%f",&xn);

printf("\n Enter tolerable error: ");

scanf("%f",&e);

do

{

fxn=f(xn);

der=dof(xn);

hn=-m\*fxn/der;

xn=xn+hn;

i++;

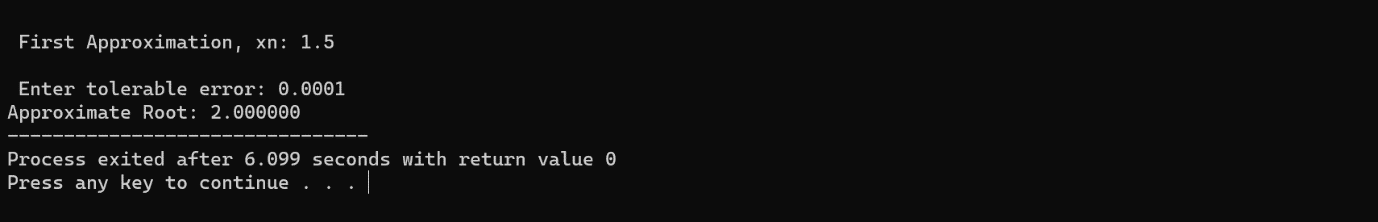
}while(fabs(hn)>e);

printf("Approximate Root: %f",xn);

return 0;

}

**Output:**



**Program 8:** Write a program in C to compute the root of using Secant Method lying between 4 and 5 upto 4DP.

**Source Code:**

#include<stdio.h>

#include<math.h>

#include<stdlib.h>

#define f(x) pow(x,2)-4\*x-10

int main()

{

float a,b,xn,fa,e,fb,fxn;

int i=0;

printf("\n Enter first initial guess: ");

scanf("%f",&a);

printf("\n Enter second initial guess: ");

scanf("%f",&b);

printf("\n Enter tolerable error: ");

scanf("%f",&e);

printf("\n");

printf("\n Iteration a \t b fa fb xn fxn");

printf("\n");

do

{

fa=f(a);

fb=f(b);

if(fa==fb)

{

printf("\n Mathematical error.");

exit (0);

}

xn=a-fa\*((b-a)/(fb-fa));

fxn=f(xn);

printf("\n %d \t %f \t %f \t %f \t %f \t %f \t %f",i,a,b,fa,fb,xn,fxn);

a=b;

b=xn;

i++;

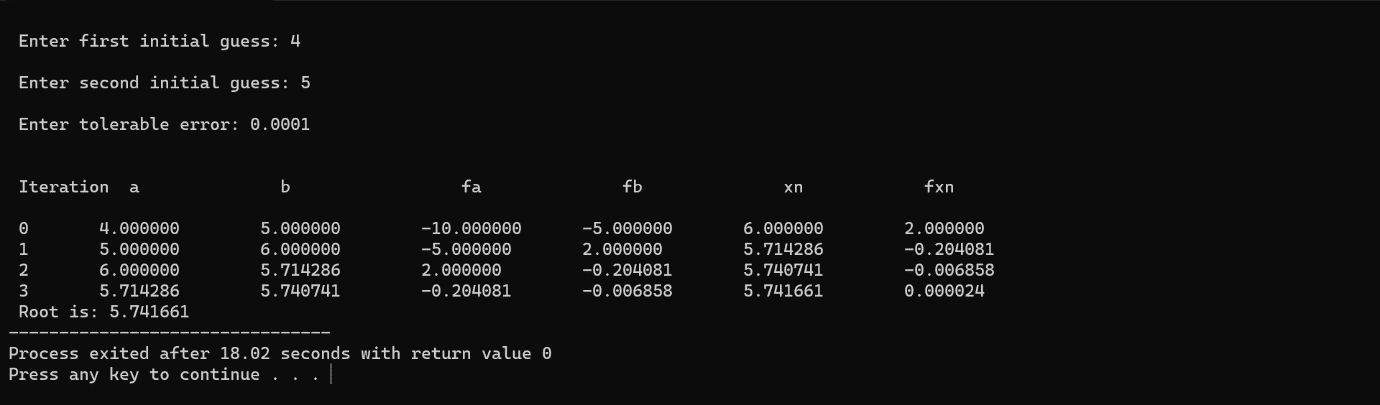
}while(fabs(fxn)>e);

printf("\n Root is: %f",xn);

return 0;

}

**Output:**



**Program 9:** Write a program in C to compute the root of by Regula Falsi with initial conditions 0,1 upto 4DP.

**Source Code:**

#include<stdio.h>

#include<math.h>

#define f(x) (3\*x)-cos(x)-1

int main()

{

float an,bn,e,fan,fbn,hn,xn,fxn;

int n=0;

up:

printf("\n Enter first initial guess, a0: ");

scanf("%f",&an);

printf("\n Enter second initial guess, b0: ");

scanf("%f",&bn);

printf("\n Enter tolerable error: ");

scanf("%f",&e);

if(an\*bn>0)

{

printf("\n Incorrect initial guess: Try again!");

goto up;

}

printf("\n n an bn f(an) f(bn) hn x(n+1) f(x(n+1)) ");

printf("\n");

do // abs is used to turn negative number into positive

{

fan=f(an);

fbn=f(bn);

hn=(abs(fan)\*(bn-an))/(abs(fan)+abs(fbn));

xn=an+hn;

fxn=f(xn);

printf("\n %d %f %f %f %f %f %f %f ",n,an,bn,fan,fbn,hn,xn,fxn);

if(fxn>0)

{

bn=xn;

}else{

an=xn;

}

n++;

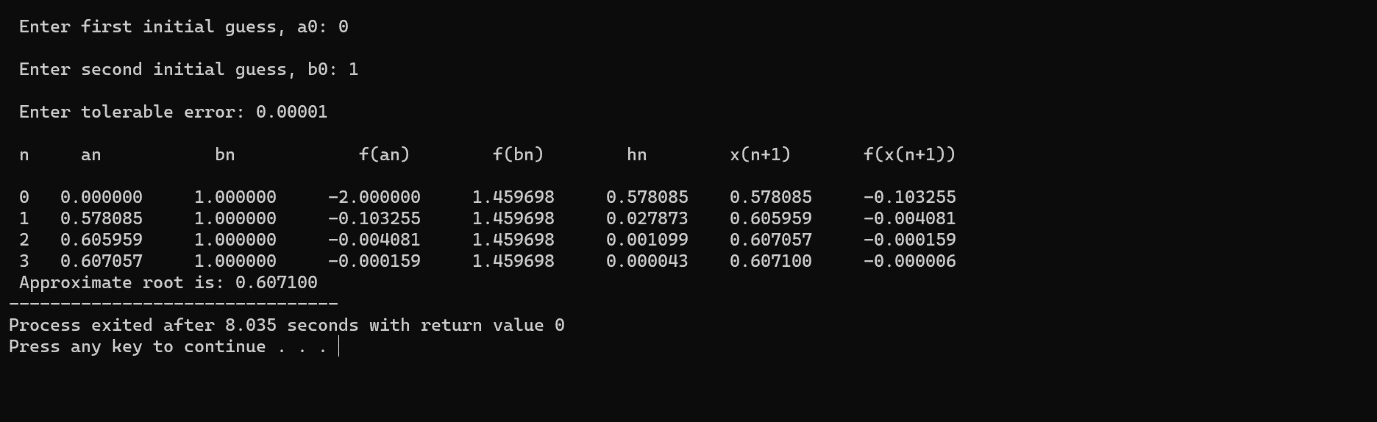
}while(fabs(fxn)>e);

printf("\n Approximate root is: %f",xn);

return 0;

}

**Output:**



**Program 10:** Write a program in C to calculate the value of f(2) using Lagrange’s Interpolation formula from the given table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X | 0 | 1 | 3 | 4 |
| f(x) | 5 | 6 | 50 | 105 |

**Source Code:**

#include<stdio.h>

int main()

{

float x[10],y[10],d[10],xx,w,sum;

int n, i, j;

printf("enter the number of interpolating points \n");

scanf("%d",&n);

n=n-1;

printf("enter the values of interpolating points \n ");

for(i=0;i<=n;i++)

scanf("%f", &x[i]);

printf("enter the values of y \n" );

for(i=0;i<=n;i++)

scanf("%f",& y[i]);

printf("enter the values for which interpolation required \n");

scanf("%f",&xx);

for(i=0;i<=n;i++)

{

d[i]=1;

for(j=0;j<=n;j++)

{

if(j==i)

d[i]=d[i]\*(xx-x[i]);

else

d[i]=d[i]\*(x[i]-x[j]);

}

}

w=1;

for(i=0;i<=n;i++)

w=w\*(xx-x[i]);

sum=0;

for(i=0;i<=n;i++)

sum=sum+(y[i]/d[i]);

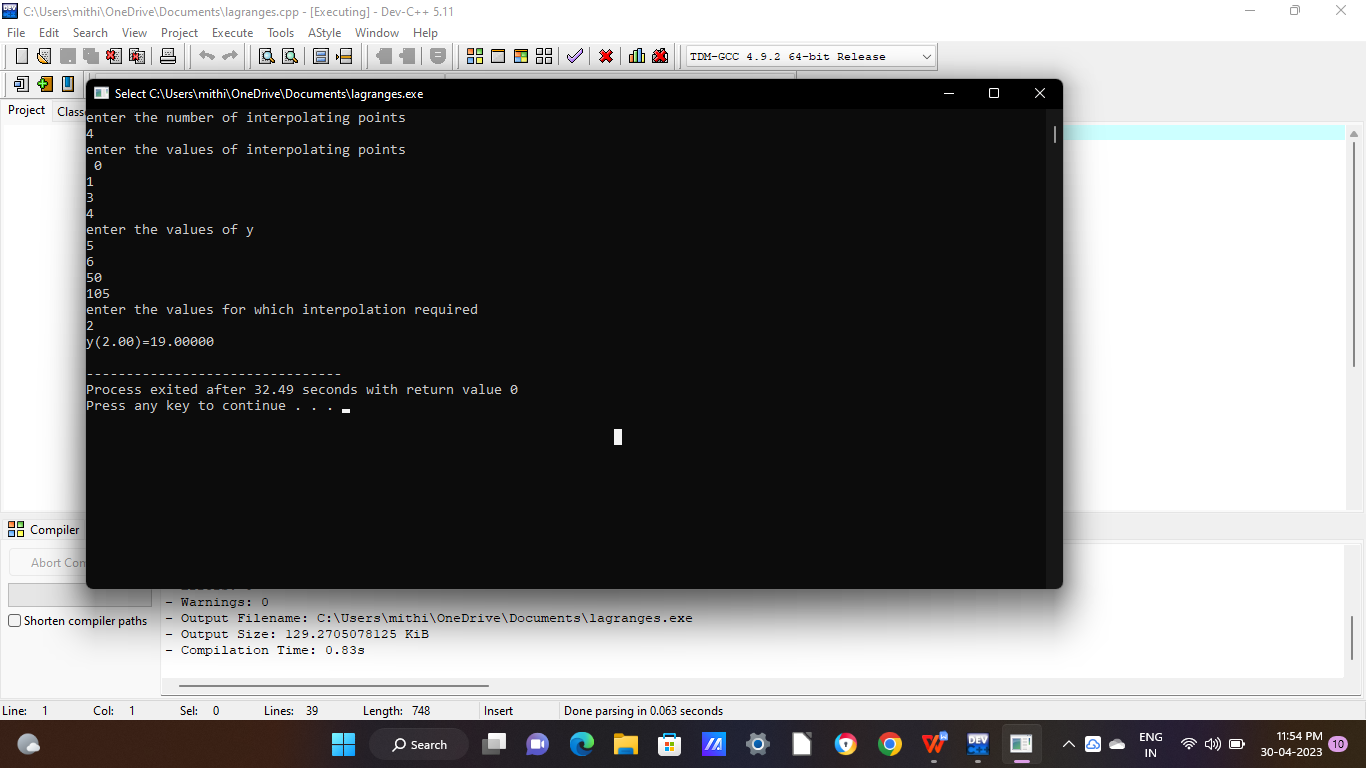
sum=w\*sum;

printf("y(%4.2f)=%7.5f \n",xx,sum);

return 0;

}

**Output:**



**Program 11:** Write a program in C to calculate the value of f(1.5) using Newton’s forward interpolation formula from the given table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x | 1 | 2 | 3 | 4 |
| f(x) | 1 | 8 | 27 | 64 |

**Source Code:**

#include<stdio.h>

#include<math.h>

int main()

{

float x[10],y[10][10],h,u1,u,fx,fy,fact ;

int i,j,n;

printf("How many terms you want to enter: ");

scanf("%d",&n);

for(i=0;i<n;i++)

{

printf("Enter the value of x%d:",i);

scanf("%f",&x[i]);

printf("Enter the value of f(x%d):",i);

scanf("%f", &y[i]);

}

printf("\nEnter the value of x for which you wants to find Y: ");

scanf("%f",&fx);

for(j=1;j<n;j++)

{

for(i=0;i<n-j;i++)

{

y[i][j]=y[i+1][j-1]-y[i][j-1];

}

}

printf("Difference Table\n");

printf("--------------------------------");

printf("\nx(i)\t y\t");

for(i=1;i<n;i++){

printf("y%d(i)\t",i);

}

printf("\n------------------------------\n");

for(i=0;i<n;i++)

{

printf("%.2f",x[i]);

j=0;

while(j<n-i)

{

printf("%.3f",y[i][j]);

j++;

}

printf("\n");

}

h=x[1]-x[0];

u=(fx-x[0])/h;

printf("u=%f",u);

fy=y[0][0];

u1=u;

fact=1;

for(i=1;i<n;i++)

{

fy=fy+(u1\*y[0][i])/fact;

u1=u1\*(u-i);

fact=fact\*(i+1);

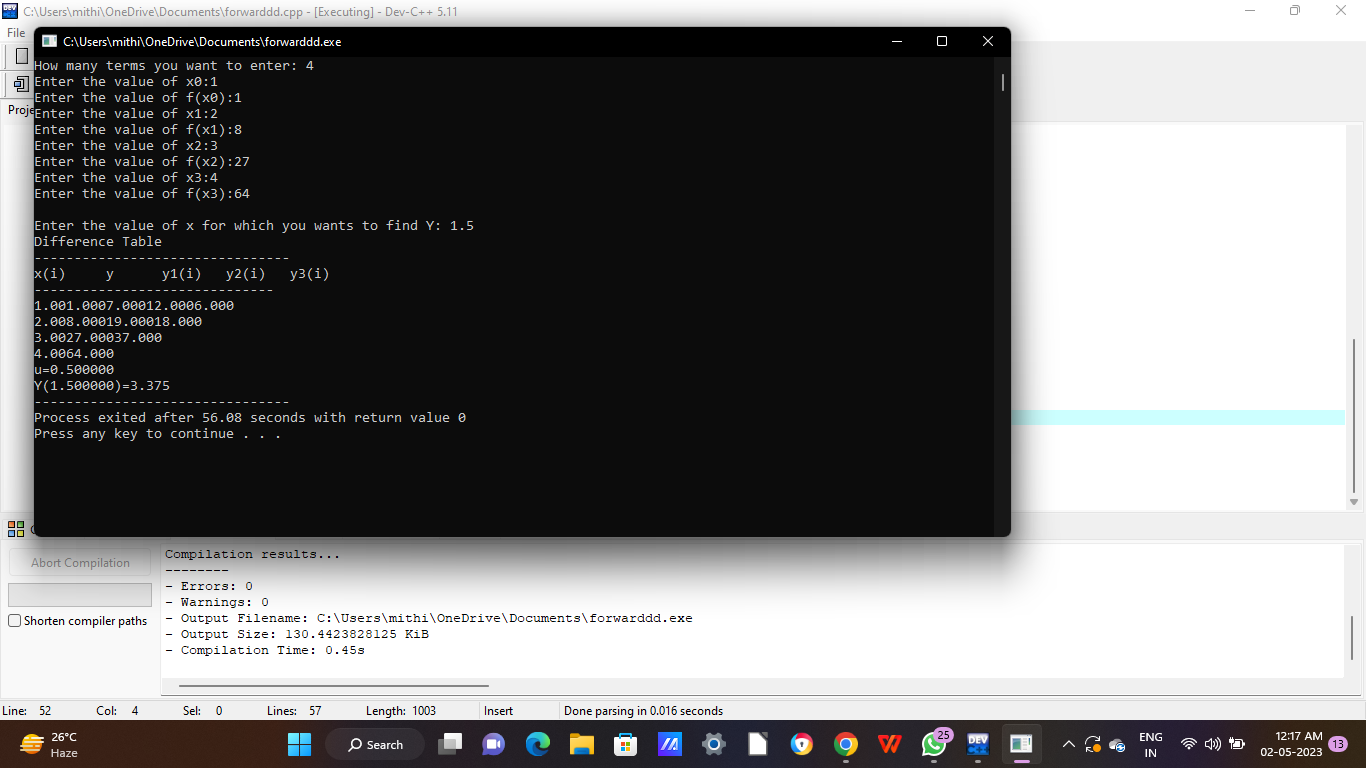
}

printf("\nY(%f)=%.3f",fx,fy);

return 0;

}

**Output:**



**Program 12:** Write a program in C to calculate the value of f(7.5) using Newton’s backward interpolation formula from the given table:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| x | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| f(x) | 1 | 8 | 27 | 64 | 125 | 216 | 343 | 512 |

**Source Code:**

#include<stdio.h>

#include<math.h>

int main()

{

float x[10],y[10][10],sum,p,u,temp;

int i,n,j,k=0,f,m;

float fact(int);

printf("\n How many records you will be enter?:");

scanf("%d",&n);

for(i=0;i<n;i++)

{

printf("\n enter the value of x%d:",i);

scanf("%f",&x[i]);

printf("\nenter the value of f(x%d):",i);

scanf("%f",&y[k][i]);

}

printf("\nenter X for finding f(x):");

scanf("%f",&p);

for(i=1;i<n;i++)

{

for(j=i;j<n;j++)

{

y[i][j]=y[i-1][j]-y[i-1][j-1];

}

}

printf("\n-----------------------------------------------\n");

printf("\n x(i)\t ");

for(i=1;i<n;i++){

printf("y%d(i)\t ",i);

}

printf("\n-----------------------------------------------\n");

for(i=0;i<n;i++)

{

printf("\n%.3f",x[i]);

for(j=0;j<=i;j++)

{

printf(" ");

printf("%.3f",y[j][i]);

}

printf("\n");

}

i=0;

do

{

if(x[i]<p && p<x[i+1])

k=1;

else

i++;

}while(k!=1);

f=i+1;

u=(p-x[f])/(x[f]-x[f-1]);

printf("\n\n u=%.3f",u);

sum=0;

for(i=0;i<n;i++)

{

temp=1;

for(j=0;j<i;j++)

{

temp=temp\*(u+j);

}

m=fact(i);

sum=sum+temp\*(y[i][f]/m);

}

printf("\n\n f(%.2f)=%f",p,sum);

return 0;

}

float fact(int a)

{

float fac=1;

if(a==0)

fac=1;

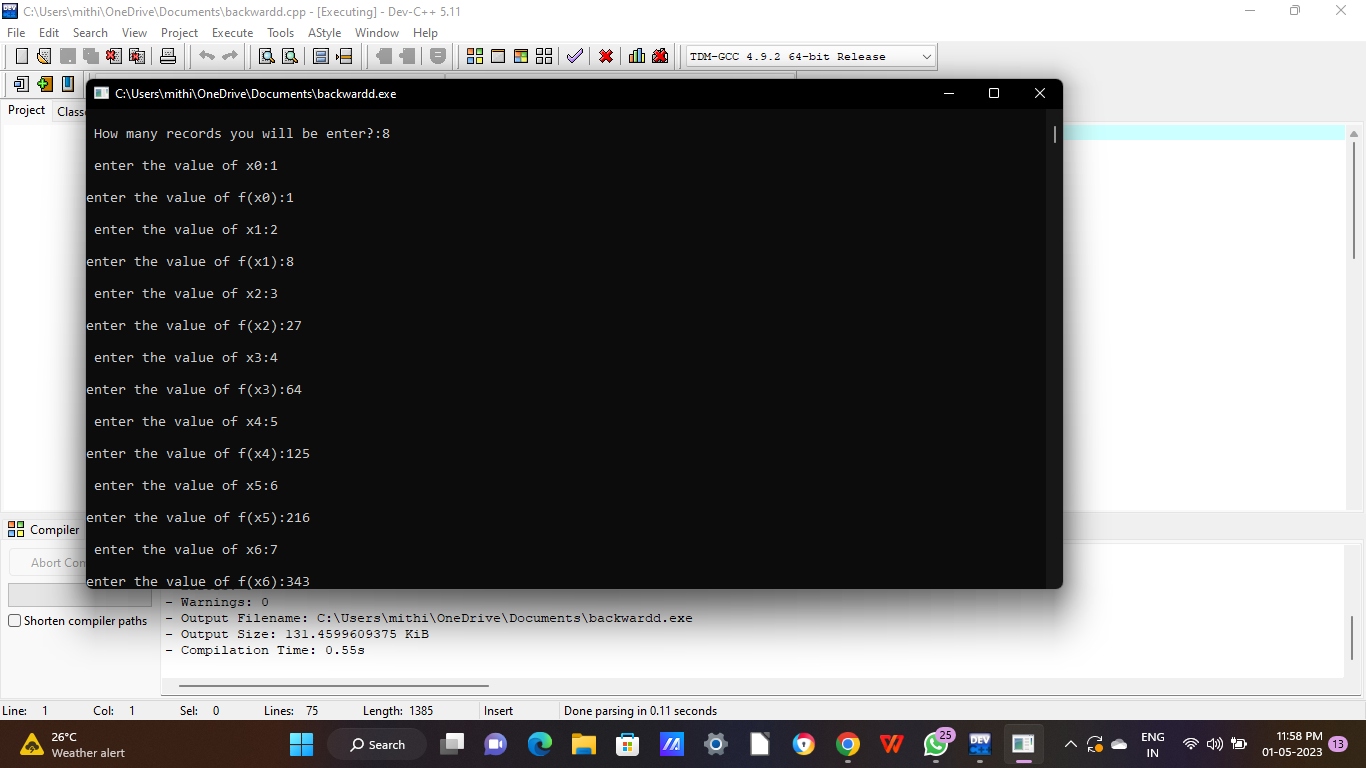
else

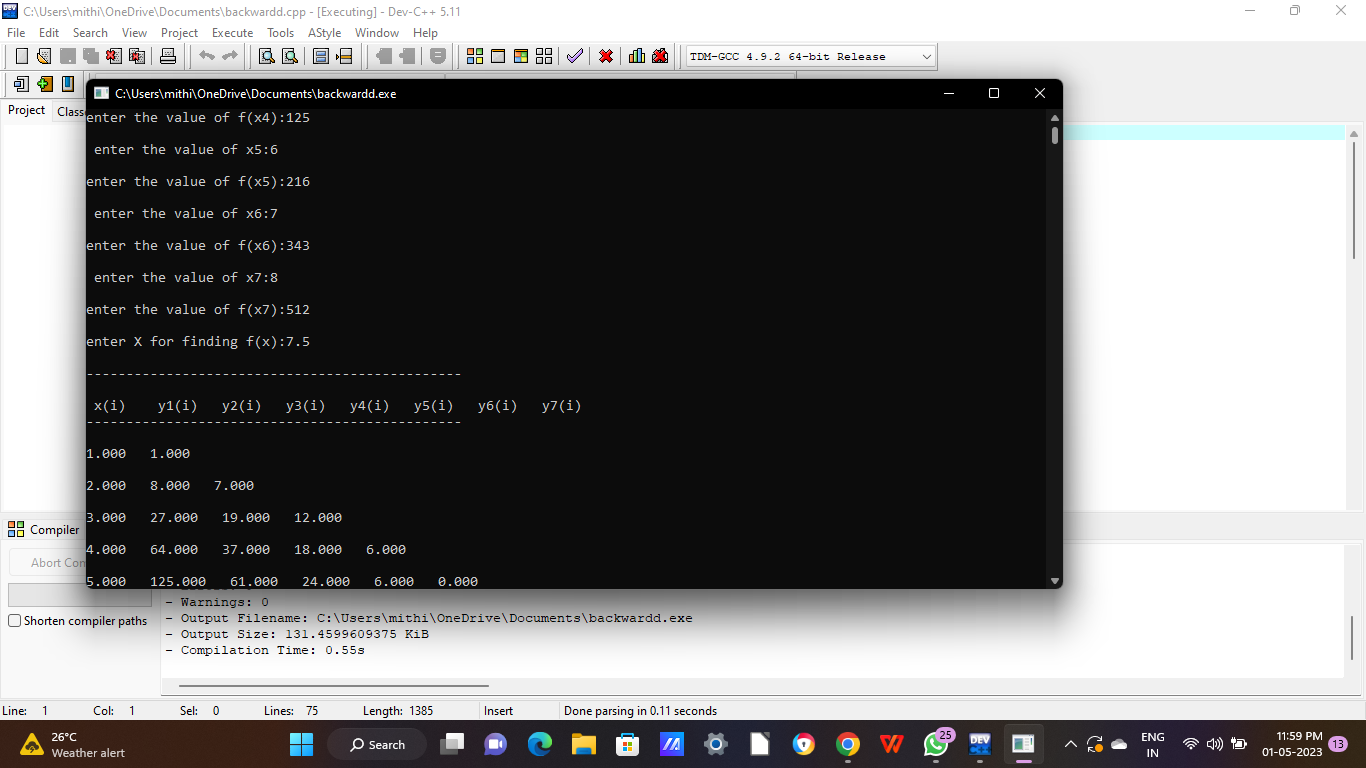
fac=a\*fact(a-1);

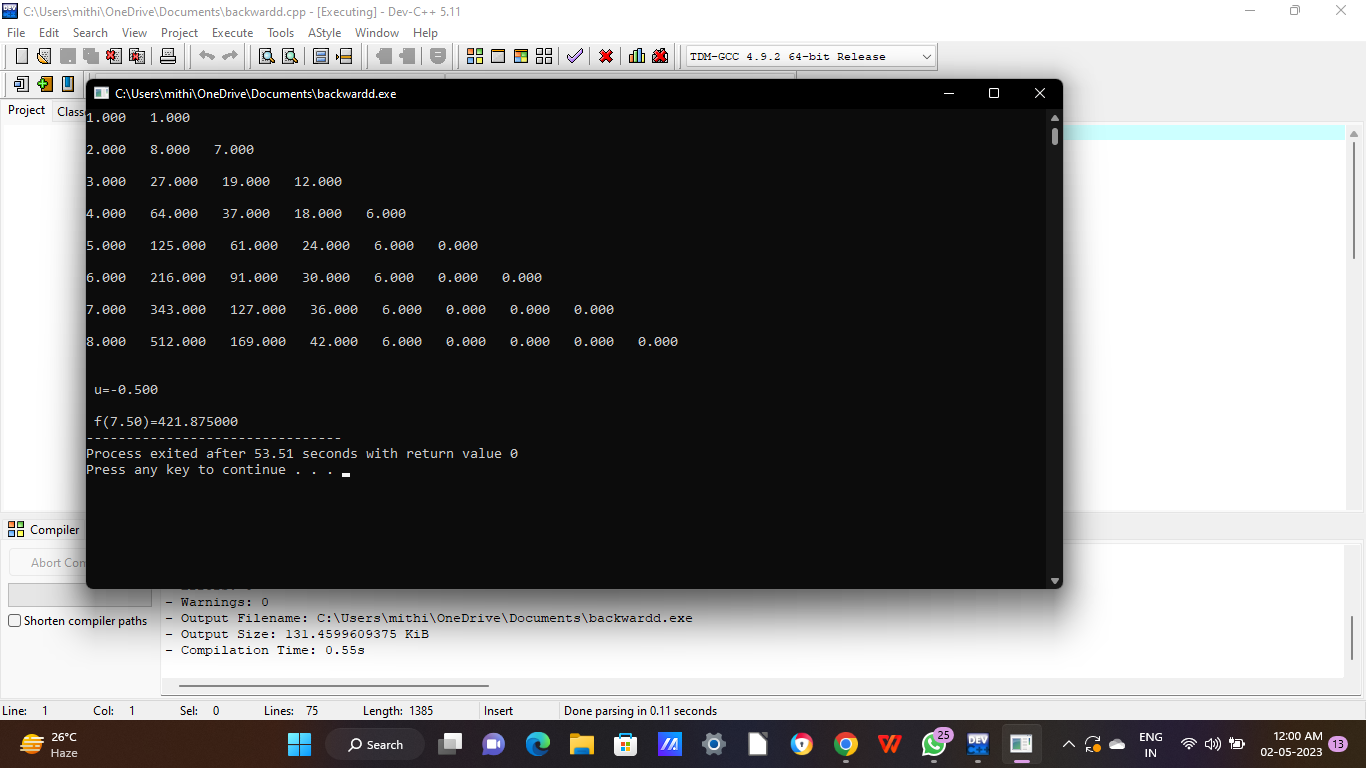
return (fac);

}

**Output:**







**Program 13:** Write a program in C to calculate the value of f(301) using Newton’s divided difference formula from the given table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x | 300 | 304 | 305 | 307 |
| f(x) | 2.477 | 2.483 | 2.484 | 2.487 |

**Source Code:**

#include<stdio.h>

int main()

{

float x[10],y[10],d[10][10],xx,term,sum;

int i,j,n;

printf("Enter The No. Of Interpolating Points \n");

scanf("%d",&n);

printf("Enter The Interpolating Points \n");

for(i=0;i<n;i++)

scanf("%f",&x[i]);

printf("Enter The Values Of y \n");

for(i=0;i<n;i++)

scanf("%f",&y[i]);

printf("Enter x For Which Interpolation Is Reqd. \n");

scanf("%f",&xx);

for(i=0;i<n;i++)

d[i][0]=y[i];

for(j=1;j<n;j++)

{

for(i=0;i<n-j;i++)

d[i][j]=(d[i][j-1]-d[i+1][j-1])/(x[i]-x[i+j]);

}

sum=y[0];

term=(xx-x[0]);

for(j=1;j<n;j++)

{

sum=sum+term\*(d[0][j]);

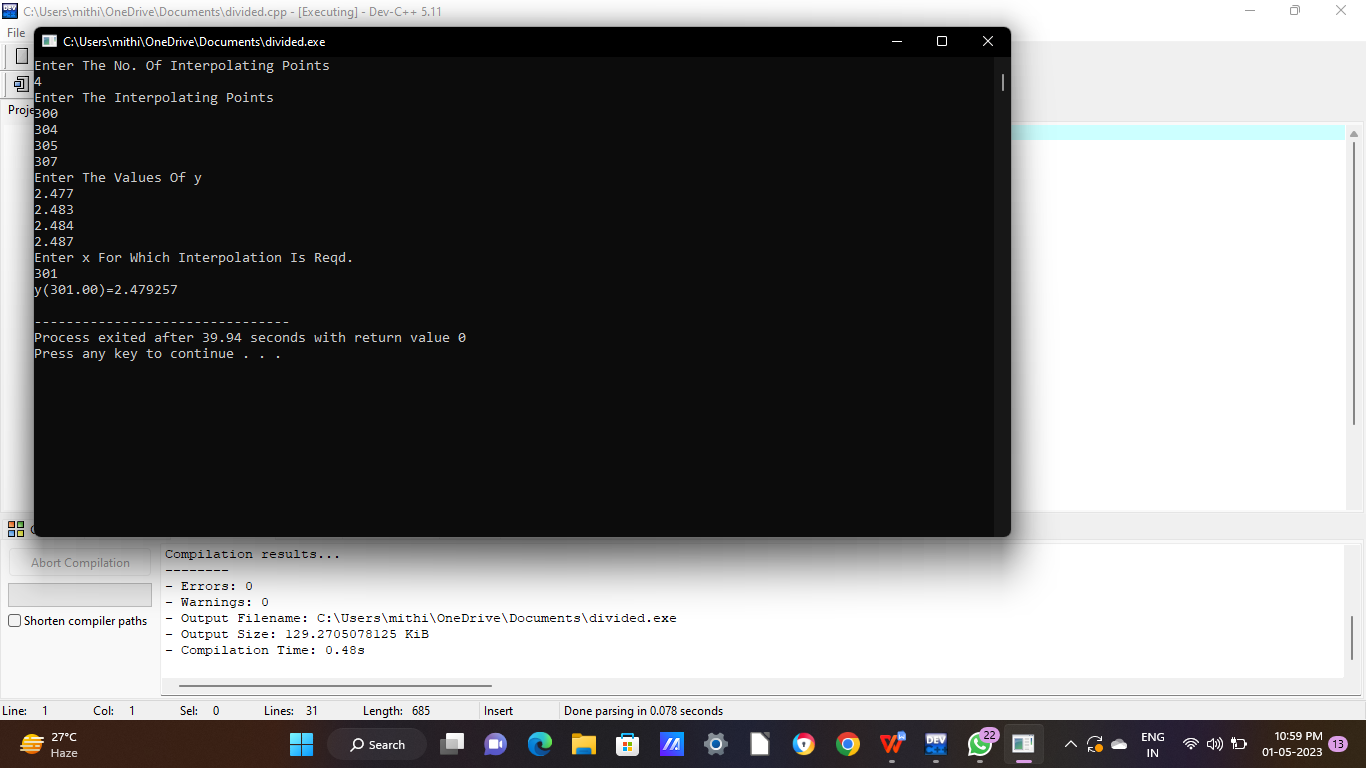
term=term\*(xx-x[j]);

}

printf("y(%4.2f)=%8.6f\n",xx,sum);

}

**Output:**



**Program 14:** Write a program in C to evaluate , by Trapezoidal Rule, taking 5 intervals upto 4DP.

**Source Code:**

#include<stdio.h>

#include<math.h>

#define f(x) 1/(x+1)

int main()

{

float upper, lower, integration=0.0, xn,stepsize;

int i, subinterval;

printf("\n Enter the lower limit: ");

scanf("%f",&lower);

printf("\n Enter the upper limit: ");

scanf("%f",&upper);

printf("\n Enter the number of sub-intervals: ");

scanf("%d",&subinterval);

stepsize=(upper-lower)/subinterval;

integration=f(lower)+f(upper);

for(i=1;i<=subinterval-1;i++)

{

xn=lower+i\*stepsize;

integration=integration+2\*f(xn);

}

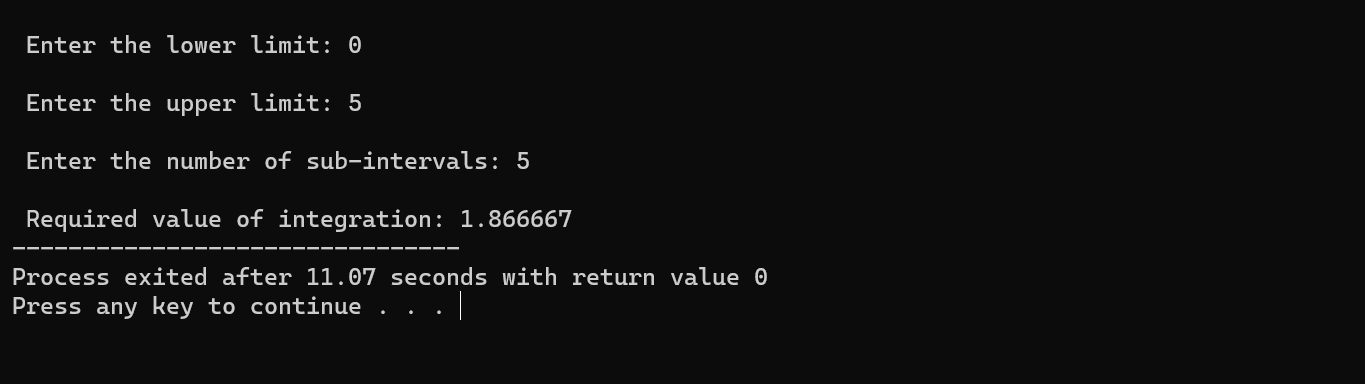
integration=integration\*(stepsize/2);

printf("\n Required value of integration: %f",integration);

return 0;

}

**Output:**



**Program 15:** Write a program in C to calculate the value of correct upto 3 decimal places, taking six intervals by Simpson’s One-third Rule.

**Source Code:**

#include<stdio.h>

#include<math.h>

#define f(x) x/(x+1)

int main()

{

float upper, lower, integration=0.0, xn,stepsize;

int i, subinterval;

printf("\n Enter the lower limit: ");

scanf("%f",&lower);

printf("\n Enter the upper limit: ");

scanf("%f",&upper);

printf("\n Enter the number of sub-intervals: ");

scanf("%d",&subinterval);

stepsize=(upper-lower)/subinterval;

integration=f(lower)+f(upper);

for(i=1;i<=subinterval-1;i++)

{

xn=lower+i\*stepsize;

if(i%2==0)

{

integration=integration+2\*f(xn);

}else{

integration=integration+4\*f(xn);

}

}

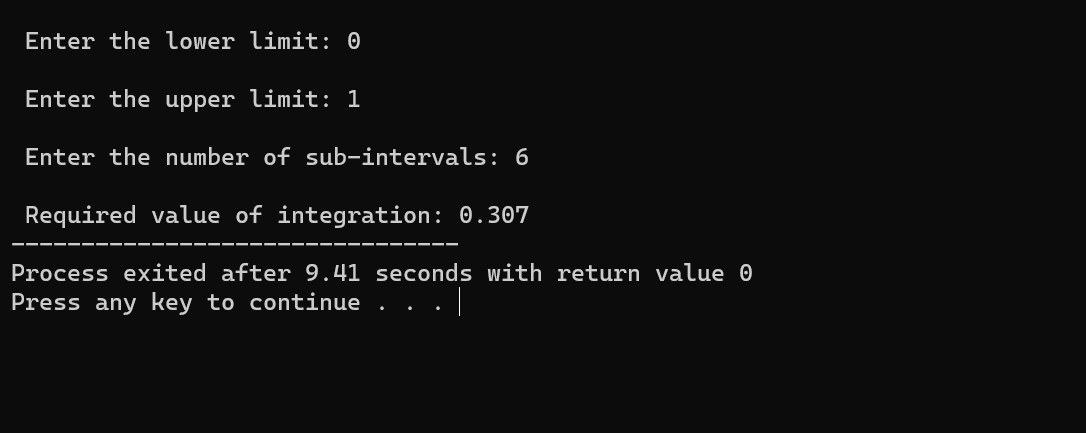
integration=integration\*(stepsize/3);

printf("\n Required value of integration: %.3f",integration);

return 0;

}

**Output:**



**Program 16:** Write a program in C to evaluate , by Weddle’s Rule taking 12 interavls correct upto 4 decimal places.

**Source Code:**

#include<stdio.h>

#include<math.h>

#define f(x) 1/(1+pow(x,2))

int main()

{

float b,a,integration=0.0,h;

int i, n,m;

printf("\n Enter lower limit: ");

scanf("%f",&a);

printf("\n Enter upper limit: ");

scanf("%f",&b);

printf("\n Enter the number of sub-intervals: ");

scanf("%d",&n);

h=(b-a)/n;

printf("\n The stepsize is: %f",h);

if(n%==0)

{

integration=(3\*h/10)\*(f(a)+f(b)+5\*(f(a+h)+f(a+5\*h))+f(a+2\*h)+f(a+4\*h)+6\*(f(a+3\*h)));

printf("\n The required Integration is: %.4f",integration);

}else{

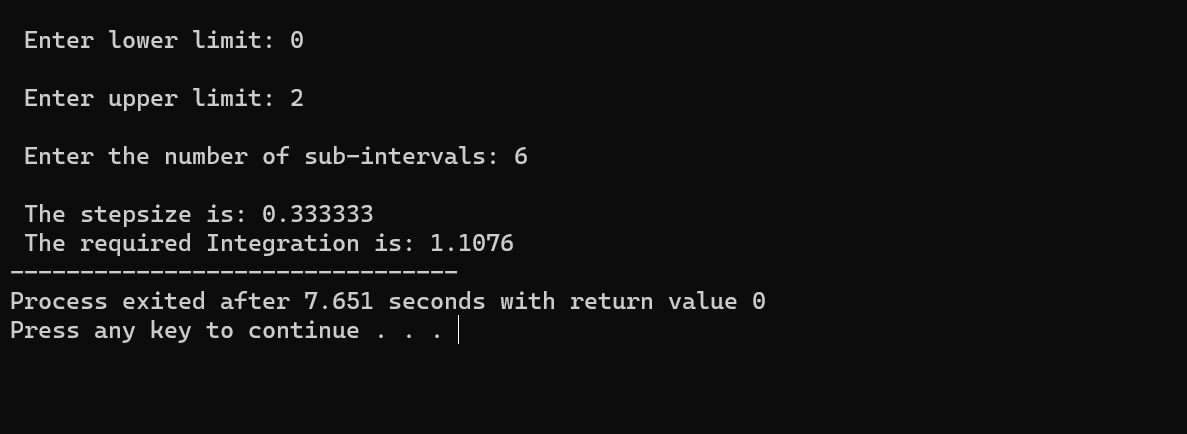
printf("\n Sorry! Weddle's Rule does not apply");

}

return 0;

}

**Output:**



**Program 17:** Write a program in C to calculate the value of dx using Gauss Quadrature formula.

**Source Code:**

#include<stdio.h>

#include<math.h>

float f(float a,float b,float t);

int main()

{

float a,b,w[11],t[11],sum=0.0;

int n,i;

printf("Enter the lower and upper limit respectively:\n");

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

printf("Enter the number of ordinates:\n");

scanf("%d",&n);

printf("Enter the ordinates:\n");

for(i=1;i<=n;i++)

{

scanf("%f",&t[i]);

}

printf("Enter the weights:\n");

for(i=1;i<=n;i++)

{

scanf("%f",&w[i]);

}

for(i=1;i<=n;i++)

sum=sum+w[i]\*f(a,b,t[i]);

sum=sum\*(b-a)/2;

printf("The value of the integral correct upto 4 decimal places=%0.4f",sum);

return 0;

}

float f(float a,float b,float t)

{

float x,y;

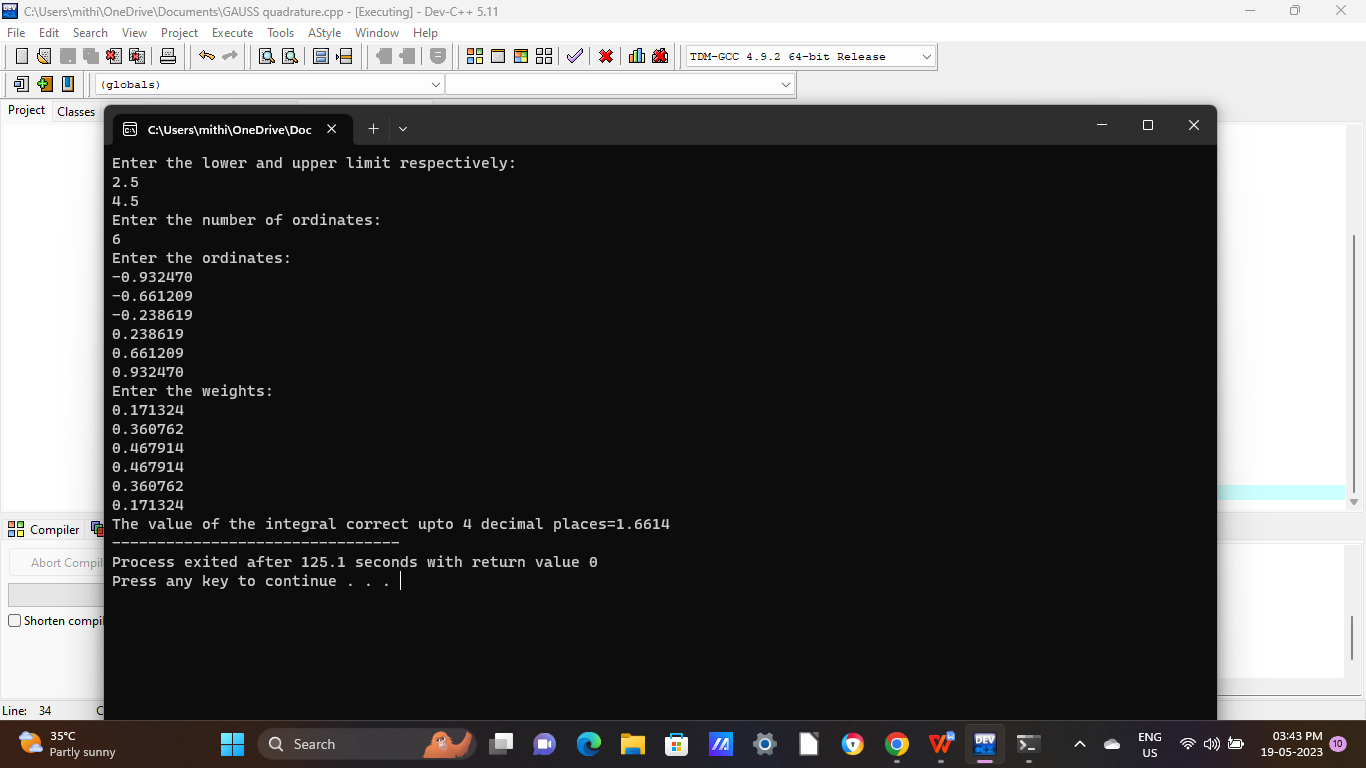
x=(t\*(b-a)+a+b)/2;

y=exp(x)/(x\*x\*x-1);

return (y);

}

**OUTPUT:**



**Program 18:** Write a program in C to solve the system of equations, by Gauss Elimination Method correct upto 4 DP.

**Source Code:**

#include<stdio.h>

#include<math.h>

int main()

{

int i,j,k,n;

float a[20][20],c,x[10],sum=0.0;

printf("\n Enter the order of matrix:");

scanf("%d",&n);

printf("\n Enter the elements of augmented matrix row-wise:\n\n");

for(i=1;i<=n;i++)

{

for(j=1;j<=(n+1);j++)

{

printf("a[%d][%d]: ",i,j);

scanf("%f",&a[i][j]);

}

}

for(j=1;j<=n;j++)

{

for(i=1;i<=n;i++)

{

if(i>j)

{

c=a[i][j]/a[j][j];

for(k=1;k<=n+1;k++)

{

a[i][k]=a[i][k]-c\*a[j][k];

}

}

}

}

x[n]=a[n][n+1]/a[n][n];

// loop for backward substitution

for(i=n-1;i>=1;i--)

{

sum=0;

for(j=i+1;j<=n;j++)

{

sum=sum+a[i][j]\*x[j];

}

x[i]=(a[i][n+1]-sum)/a[i][i];

}

printf("\n the solution is:");

for(i=1;i<=n;i++)

{

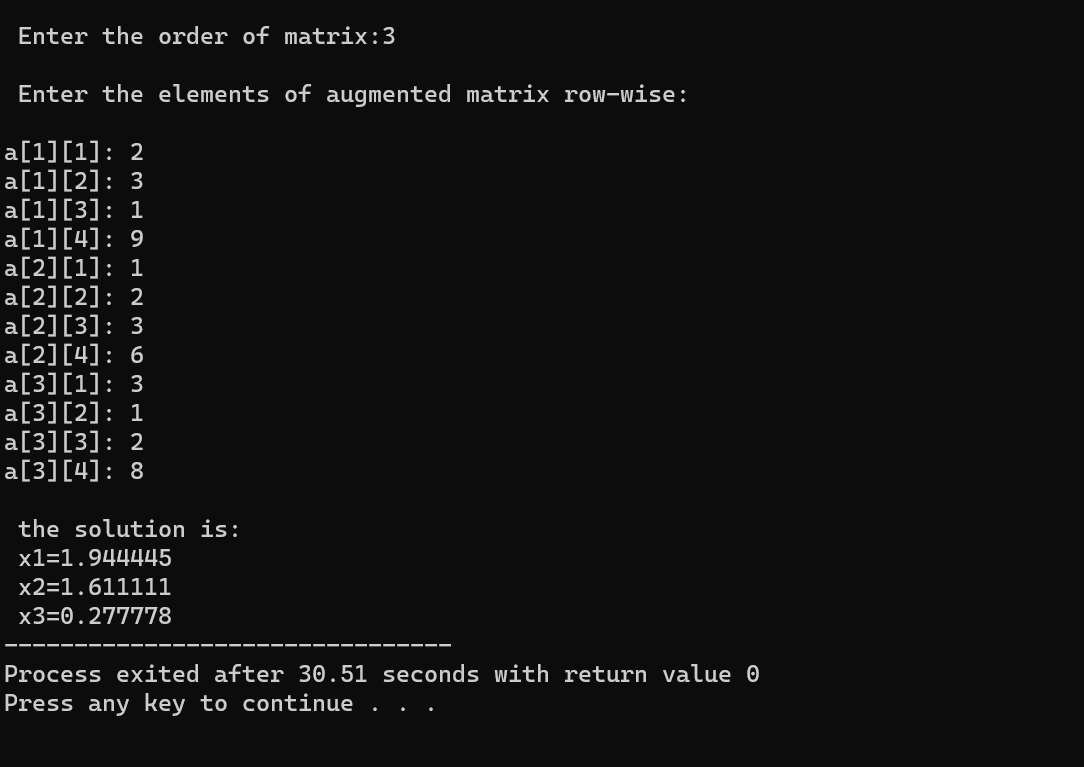
printf("\n x%d=%f\t",i,x[i]);

}

return 0;

}

**Output:**



**Program 19:** Write a program in C to solve the system by Gauss-Jacobi Method (Iteration Method).

**Source Code:**

#include<stdio.h>

#include<math.h>

int main()

{

int i,j,k,n;

float A[100][100],c,x[10];

printf("\n Enter the size of matrix: ");

scanf("%d",&n);

printf("\n Enter the elements of augumented matrix row-wise: ");

for(i=1;i<=n;i++)

{

for(j=1;j<=(n+1);j++)

{

printf("\n A[%d][%d]: ",i,j);

scanf("%f",&A[i][j]);

}

}

// findind elements of diagonal matrix

for(j=1;j<=n;j++)

{

for(i=1;i<=n;i++)

{

if(i!=j)

{

c=A[i][j]/A[j][j];

for(k=1;k<=n+1;k++)

{

A[i][k]=A[i][k]-c\*A[j][k];

}

}

}

}

printf("\n The solution is: \n");

for(i=1;i<=n;i++)

{

x[i]=A[i][n+1]/A[i][i];

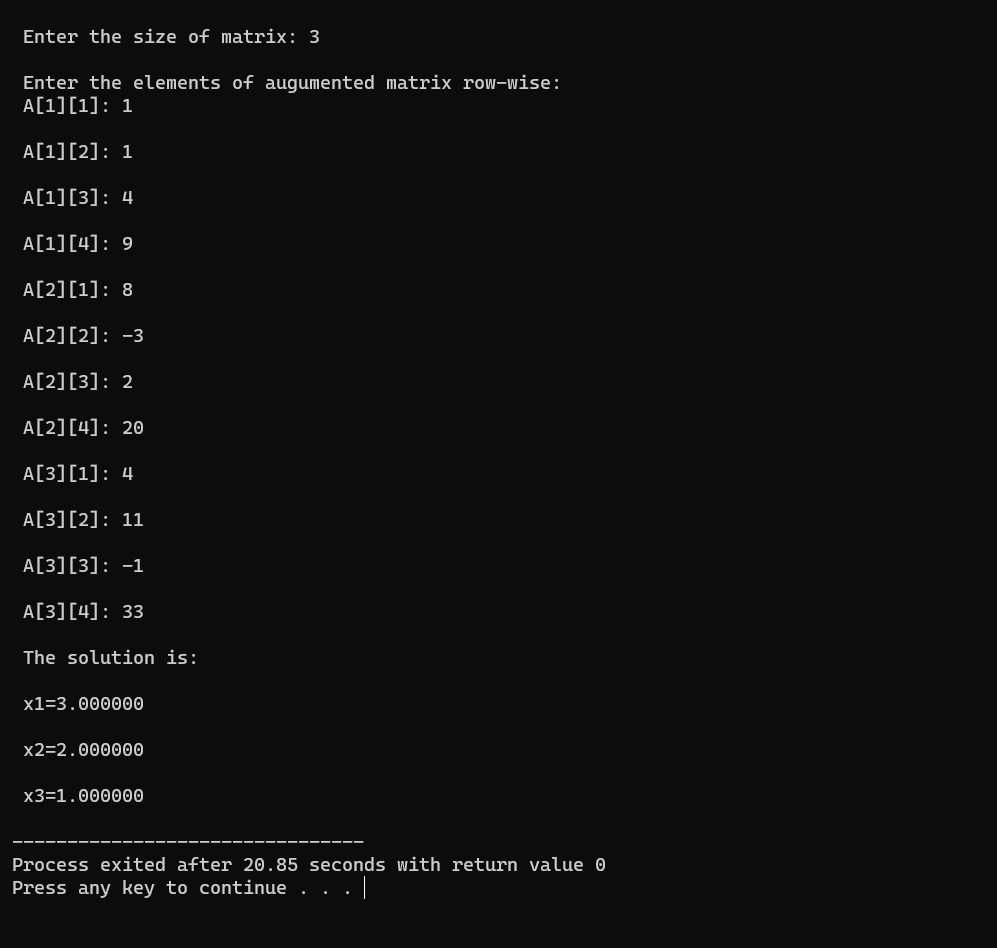
printf("\n x%d=%f \n",i,x[i]);

}

return 0;

}

**Output:**



**Program 20:** Write a program in C to solve the follogram wing system of equations by Gauss Seidel method .

3x+20y-z=-18

2x+3y+20z =25

20x+y-2z=17

**Source Code:**

#include<stdio.h>

#include<math.h>

float f1(float x,float y,float z)

{

float result1;

result1=(17-y+2\*z)/20;

return result1;

}

float f2(float x,float y,float z)

{

float result2;

result2=(-18-3\*x+z)/20;

return result2;

}

float f3(float x,float y, float z)

{

float result3;

result3=(25-2\*x+3\*y)/20;

return result3;

}

int main()

{

float x0=0,y0=0,z0=0,x1,y1,z1,e1,e2,e3,e;

int count=1;

printf("Enter tolerable error:\n");

scanf("%f",&e);

printf("\nCount\tx\ty\tz\n");

do

{

x1=f1(x0,y0,z0);

y1=f2(x1,y0,z0);

z1=f3(x0,y1,z0);

printf("%d\t%0.4f\t%0.4f\t%0.4f\n",count,x1,y1,z1);

e1=fabs(x0-x1);

e2=fabs(y0-y1);

e3=fabs(z0-z1);

count++;

x0=x1;

y0=y1;

z0=z1;

}

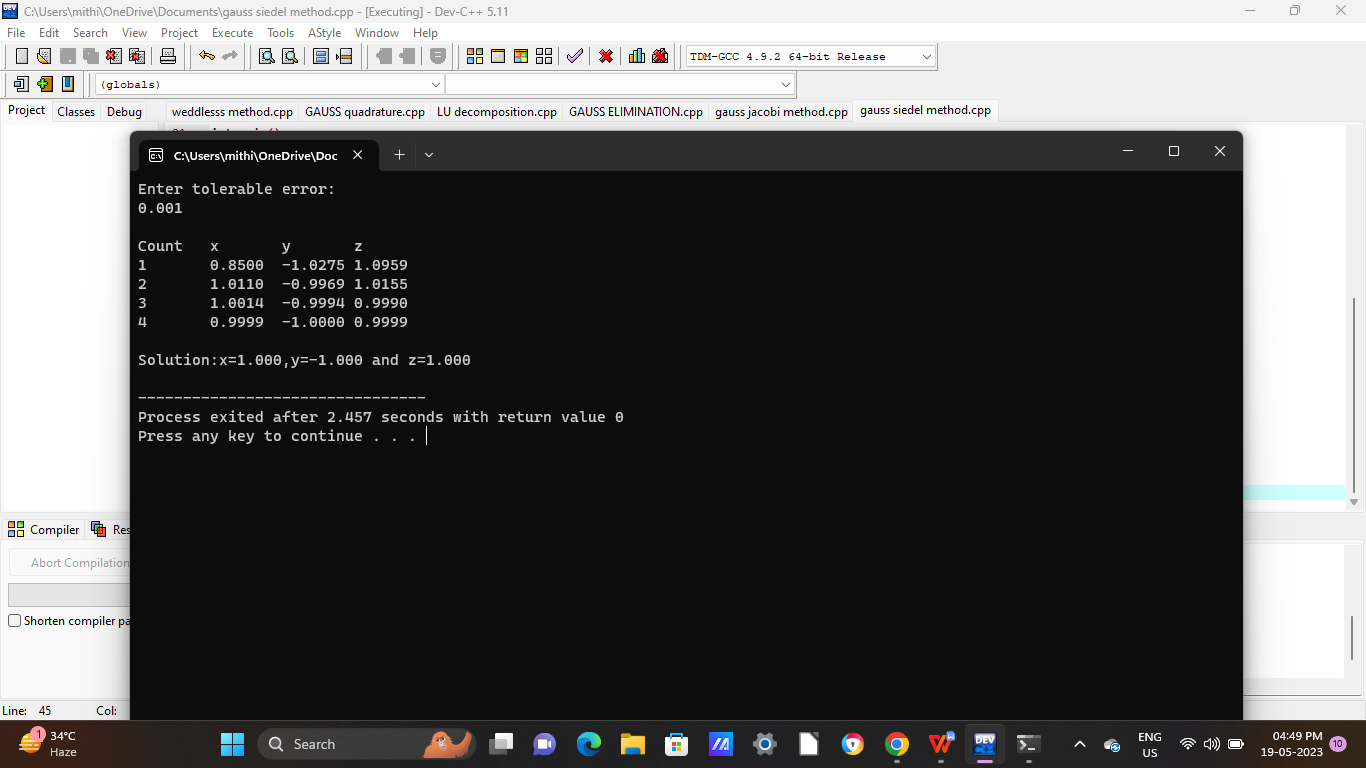
while(e1>e&& e2>e&& e3>e);

printf("\nSolution:x=%0.3f,y=%0.3f and z=%0.3f\n",x1,y1,z1);

return 0;

}

**Output:**



**Program 21:** Write a program in C to solve the following system of equations by LU decomposition method .

2x-3y+4z=8

x+y+4z =15

3x+4y-z=8

**Source Code:**

#include<stdio.h>

int main()

{

int n,i,j,k;

float a[11][11],l[11][11], u[11][11],b[11],x[11],y[11],sum;

printf("Enter the number of unknown:\n");

scanf("%d",&n);

printf("Enter the coefficient matrix: \n");

for(i=1;i<=n;i++)

{

for(j=1;j<=n;j++)

scanf("%f",&a[i][j]);

}

printf("Enter the constant matrix:\n");

for(i=1;i<=n;i++)

scanf("%f",&b[i]);

for(j=1;j<=n;j++)

{

for(i=1;i<=n;i++){

if(i<=j){

u[i][j]=a[i][j];

for(k=1;k<=i-1;k++)

u[i][j]=u[i][j]-l[i][k]\*u[k][j];

if(i==j)

l[i][j]=1;

else

l[i][j]=0;

}else{

u[i][j]=0;

l[i][j]=a[i][j];

for(k=1;k<=j-1;k++)

l[i][j]=l[i][j]-l[i][k]\*u[k][j];

l[i][j]=l[i][j]/u[j][j];

}

}

}

printf("The upper triangular matrix is: \n");

for(i=1;i<=n;i++){

for(j=1;j<=n;j++)

printf("%7.5f",u[i][j]);

printf("\n");

}

printf("The lower triangular matrix is: \n");

for(i=1;i<=n;i++){

for(j=1;j<=n;j++)

printf("%7.5f",l[i][j]);

printf("\n");

}

for(i=1;i<=n;i++){

y[i]=b[i];

for(j=1;j<i;j++)

y[i]=y[i]-l[i][j]\*y[j];

}

for(i=n;i>=1;i--){

x[i]=y[i];

for(j=i+1;j<=n;j++)

x[i]=x[i]-u[i][j]\*x[j];

x[i]=x[i]/u[i][i];

}

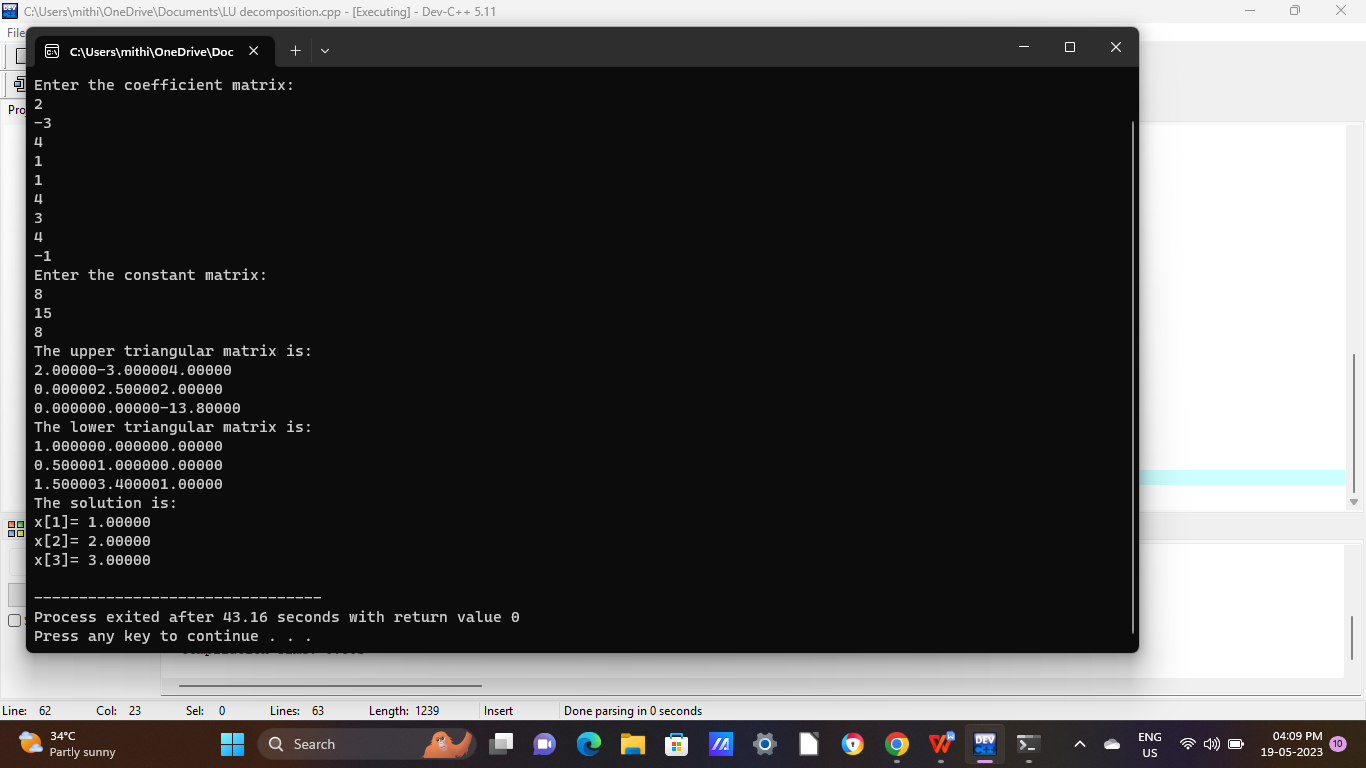
printf("The solution is:\n");

for(i=1;i<=n;i++)

printf("x[%d]=%8.5f\n",i,x[i]);

}

**Output:**



**Program 22:** Write a program in C to find the largest eigenvalue and the corresponding eigenvector of the matrix by power method .

**Source Code:**

#include<stdio.h>

#include<math.h>

int main()

{

int i,j,n;

float A[40][40],x[40],z[40],e[40],zmax,emax;

printf("\nEnter the order of matrix:");

scanf("%d",&n);

printf("\nEnter matrix elements row-wise\n");

for(i=1;i<=n;i++)

{

for(j=1;j<=n;j++)

{

printf("A[%d][%d]=", i,j);

scanf("%f",&A[i][j]);

}

}

printf("\nEnter the column vector\n");

for(i=1;i<=n;i++)

{

printf("X[%d]=",i);

scanf("%f",&x[i]);

}

do{

for(i=1;i<=n;i++)

{

z[i]=0;

for(j=1;j<=n;j++)

{

z[i]=z[i]+A[i][j]\*x[j];

}

}

zmax=fabs(z[1]);

for(i=2;i<=n;i++)

{

if((fabs(z[i]))>zmax)

zmax=fabs(z[i]);

}

for(i=1;i<=n;i++)

{

z[i]=z[i]/zmax;

}

for(i=1;i<=n;i++)

{

e[i]=0;

e[i]=fabs((fabs(z[i]))-(fabs(x[i])));

}

emax=e[1];

for(i=2;i<=n;i++)

{

if(e[i]>emax)

emax=e[i];

}

for(i=1;i<=n;i++)

{

x[i]=z[i];

}

}

while(emax>0.001);

printf("\n The required eigen value is %f",zmax);

printf("\n\nThe required eigen vector is :\n");

for(i=1;i<=n;i++)

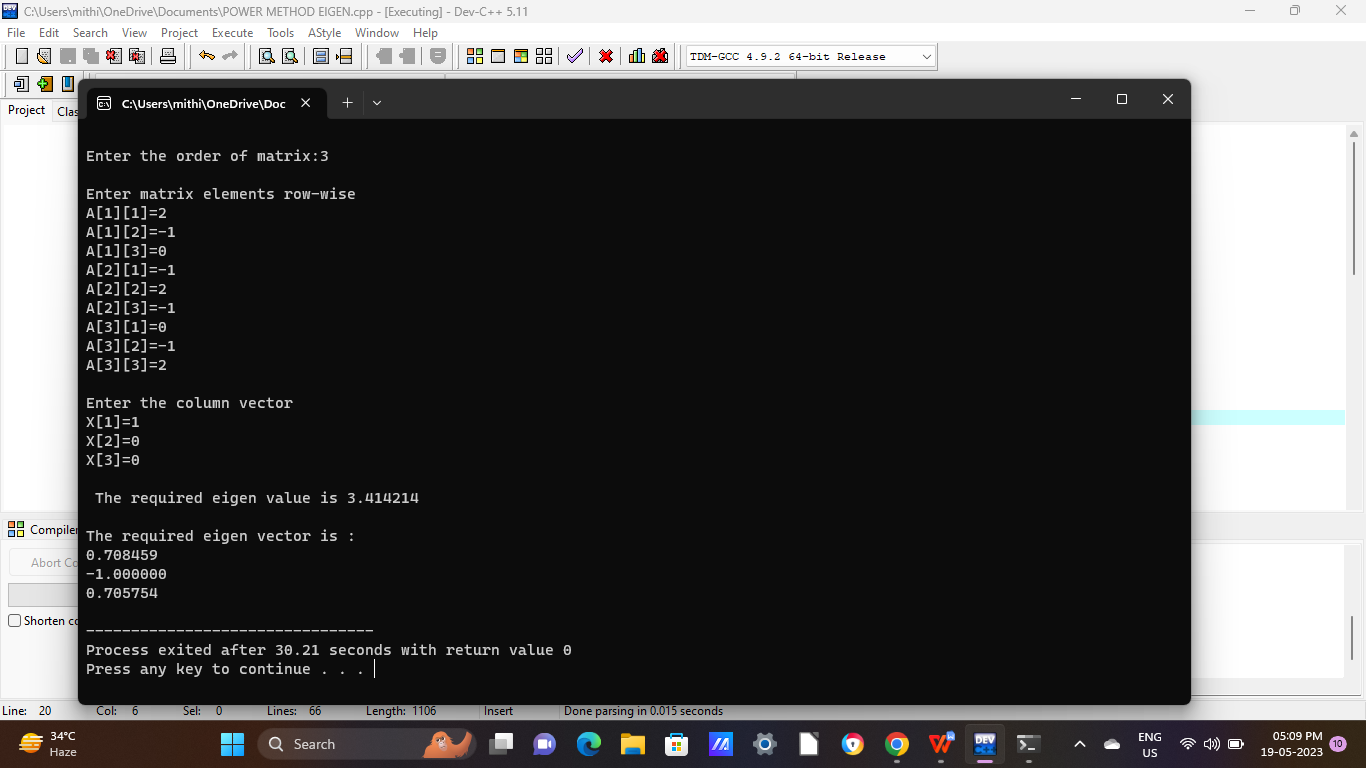
{

printf("%f\n",z[i]);

}

}

**Output:**



**Program 23:** Write a program in C to fit a straight line y=bx+a to the following

data correct to 2DP.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| X | -1 | 0 | 1 | 2 | 3 |
| Y | -3 | 2 | 7 | 12 | 17 |

**Source Code:**

#include<stdio.h>

int main()

{

int n,i,j;

float sum1=0,sum2=0,sum3=0,sum4=0,a,b;

printf("Enter the number of observation: ");

scanf("%d",&n);

float x[n],y[n],aug\_matrix[2][3];

printf("\nEnter the values of x\n");

for(int i=0;i<n;i++)

{

scanf("%f",&x[i]);

}

printf("\nEnter the values of y\n");

for(int i=0;i<n;i++)

{

scanf("%f",&y[i]);

}

for(int i=0;i<n;i++)

{

sum1=sum1+x[i];

sum2=sum2+y[i];

sum3=sum3+x[i]\*y[i];

sum4=sum4+x[i]\*x[i];

}

aug\_matrix[0][0]=n;

aug\_matrix[0][1]=sum1;

aug\_matrix[0][2]=sum2;

aug\_matrix[1][0]=sum1;

aug\_matrix[1][1]=sum4;

aug\_matrix[1][2]=sum3;

int ratio=aug\_matrix[1][0]/aug\_matrix[0][0];

for(int i=0;i<3;i++)

{

aug\_matrix[1][i]=aug\_matrix[1][i]-ratio\*aug\_matrix[0][i];

}

for(i=0;i<2;i++)

{

for(int j=0;j<3;j++)

{

printf("\n%.2f",aug\_matrix[1][j]);

}

printf("\n\n");

}

b=aug\_matrix[1][2]/aug\_matrix[1][1];

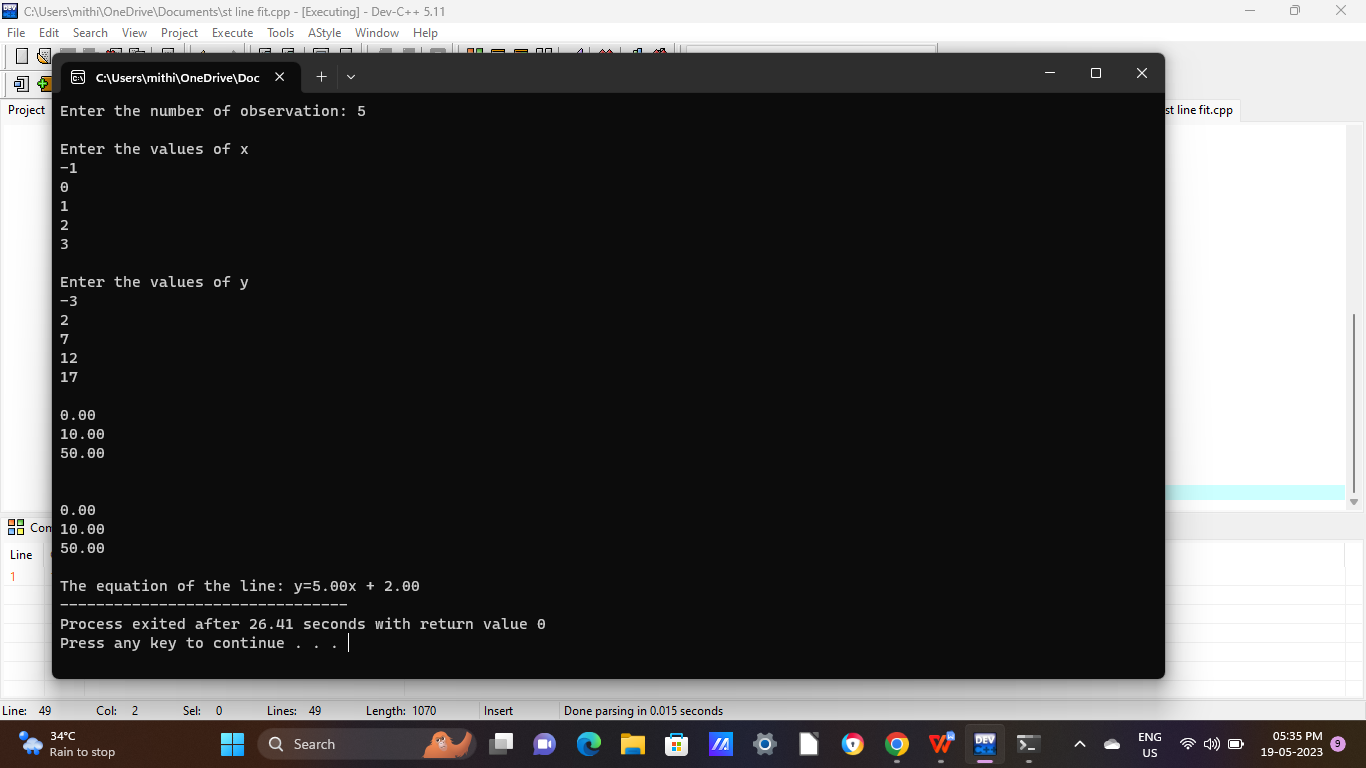
a=(aug\_matrix[0][2]-aug\_matrix[0][1]\*b)/aug\_matrix[0][0];

printf("The equation of the line: y=%0.2fx + %0.2f",b,a);

return 0;

}

**Output:**



**Program 24:** Write a program in C to fit a parabola y=ax2+bx+c to the following

data correct to 2DP.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| X | -2 | -1 | 0 | 1 | 2 |
| Y | 4 | 1 | 0 | 1 | 4 |

**Source Code:**

#include<stdio.h>

void convertToUpperTriangular (float a[3][4],int n){

int i,j,x,y,k;

float ratio;

for(i=0;i<n;i++){

for(j=0;j<n;j++){

if(j>i)

{

ratio=a[j][i]/a[i][i];

printf("Ratio = %f\n",ratio);

for(k=0;k<n+1;k++)

a[j][k]=a[j][k]-(ratio\*a[i][k]);

printf("Intermediate forms: \n");

for(x=0;x<n;x++){

for(y=0;y<n+1;y++)

printf("%.2f",a[x][y]);

printf("\n");

}

printf("\n");

}

}

}

}

void ApplyBackSubstitution(float a[3][4],float value[],int n){

int i,j;

float sum;

value[n-1]=a[n-1][n]/a[n-1][n-1];

for(i=n-2;i>=0;i--){

sum=0;

for(j=i+1;j<n;j++)

sum=sum+a[i][j]\*value[j];

value[i]=(a[i][n]-sum)/a[i][i];

}

}

int main(){

int n,i;

float sx=0,sy=0,sxy=0,sx2y=0,sx2=0,sx3=0,sx4=0,a,b,c;

float value[3];

printf("Enter the number of observations: ");

scanf("%d",&n);

float x[n],y[n],augmatrix[3][4];

printf("Enter the values of x\n");

for(i=0;i<n;i++)

scanf("%f",&x[i]);

printf("Enter the values of y\n");

for(i=0;i<n;i++)

scanf("%f",&y[i]);

for(i=0;i<n;i++)

{

sx=sx+x[i];

sy=sy+y[i];

sxy=sxy+x[i]\*y[i];

sx2y=sx2y+x[i]\*x[i]\*y[i];

sx2=sx2+x[i]\*x[i];

sx3=sx3+x[i]\*x[i]\*x[i];

sx4=sx4+x[i]\*x[i]\*x[i]\*x[i];

}

augmatrix[0][0]=sx2;

augmatrix[0][1]=sx;

augmatrix[0][2]=n;

augmatrix[0][3]=sy;

augmatrix[1][0]=sx3;

augmatrix[1][1]=sx2;

augmatrix[1][2]=sx;

augmatrix[1][3]=sxy;

augmatrix[2][0]=sx4;

augmatrix[2][1]=sx3;

augmatrix[2][2]=sx2;

augmatrix[2][3]=sx2y;

convertToUpperTriangular(augmatrix,3);

ApplyBackSubstitution(augmatrix,value,3);

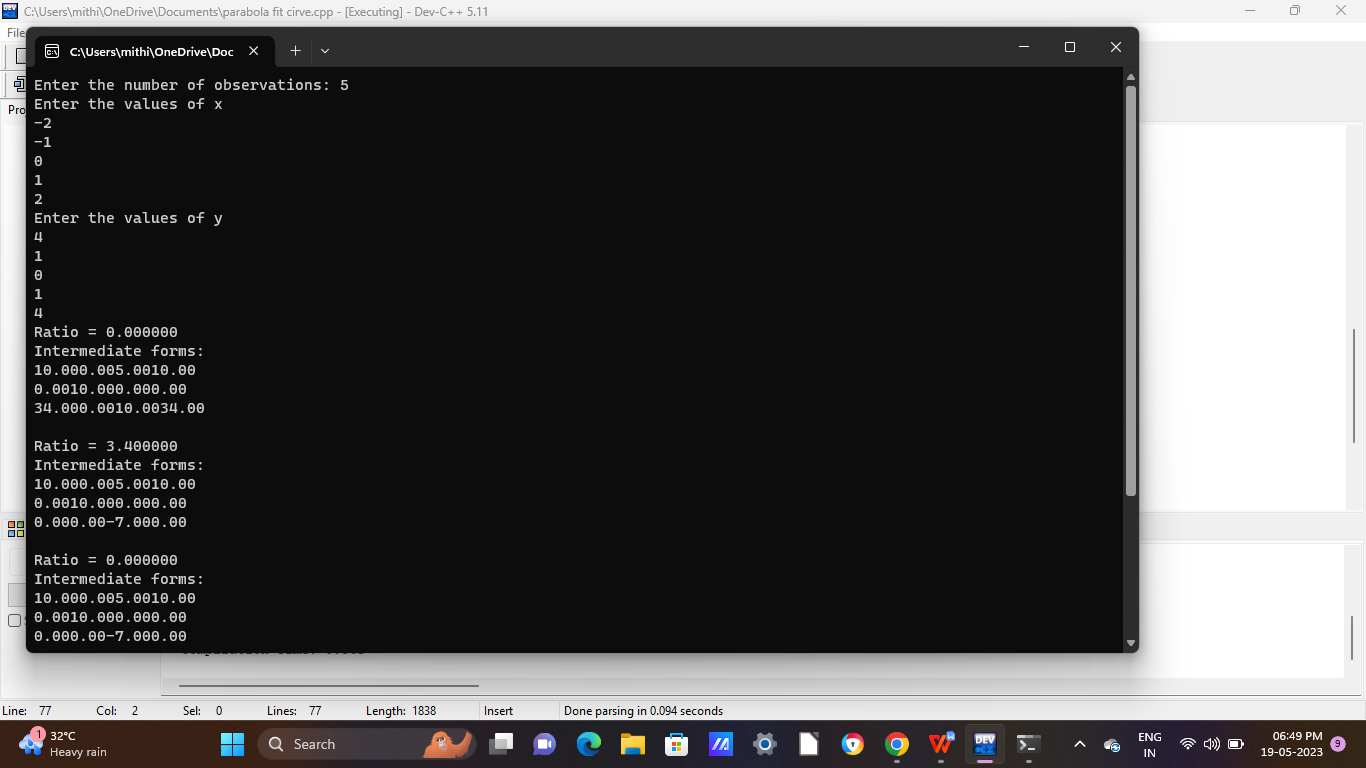
printf("\nValue of a=%0.2f \nValue of b=%0.2f \nValue of c=%0.2f\n\n",value[0],value[1],value[2]);

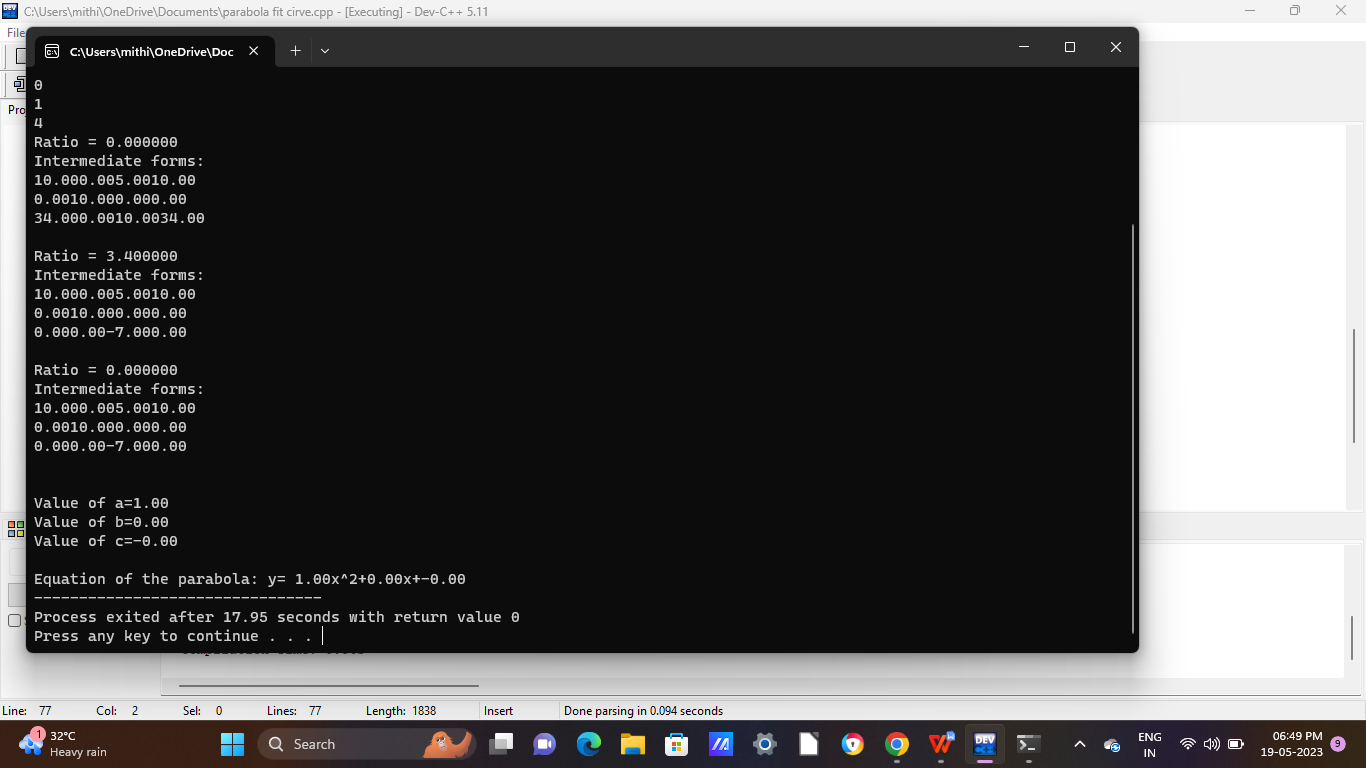
printf("Equation of the parabola: y= %0.2fx^2+%0.2fx+%0.2f",value[0],value[1],value[2]);

return 0;

}

**Output:**





**Program 25:** Write a program in C to fit a curve y=axb to the following data correct to 2DP.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X | 61 | 26 | 7 | 2.6 |
| Y | 350 | 400 | 500 | 600 |

**Source Code:**

#include<stdio.h>

#include<math.h>

#define s 50

int main()

{

int n,i;

float x[s],y[s], sumx=0,sumx2=0,sumy=0,sumxy=0,a,b,A;

printf("How many data points?\n");

scanf("%d",&n);

for(i=1;i<=n;i++)

{

printf("x[%d]=",i);

scanf("%f",&x[i]);

printf("y[%d]=",i);

scanf("%f",&y[i]);

}

for(i=1;i<=n;i++)

{

sumx=sumx+log(x[i]);

sumx2=sumx2+log(x[i])\*log(x[i]);

sumy=sumy+log(y[i]);

sumxy=sumxy+log(x[i])\*log(y[i]);

}

b=(n\*sumxy-sumx\*sumy)/(n\*sumx2-sumx\*sumx);

A=(sumy-b\*sumx)/n;

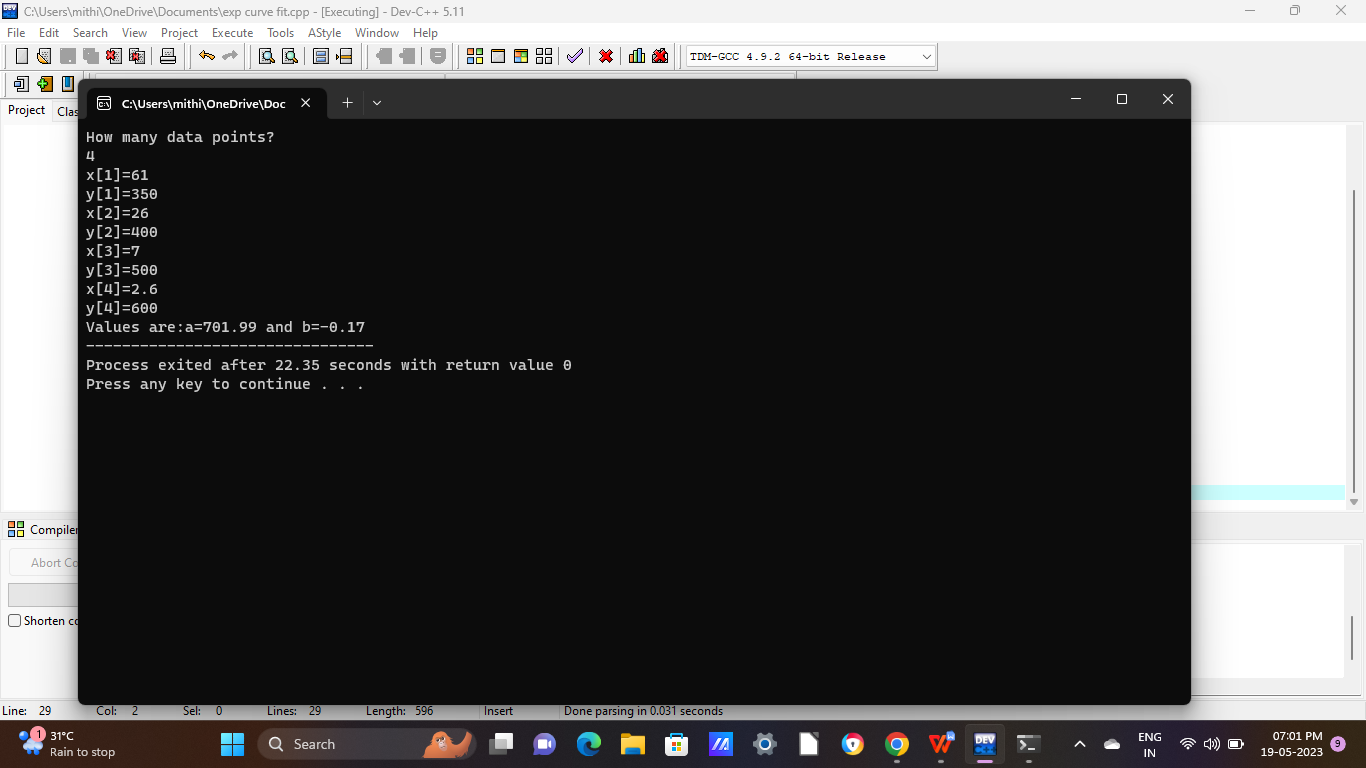
a=exp(A);

printf("Values are:a=%0.2f and b=%0.2f",a,b);

return(0);

}

**Output:**



**Program 26:** Write a program in C to determine y(1) by Euler’s Method with step length 0.1 correct upto 4 significant figures where

= x+y with y(0)=1.

**Source Code:**

#include<stdio.h>

float f(float x, float y)

{

return x+y;

}

int main()

{

float x0=0.0,y0=1.0,x,y,h=0.1,xn=1.0;

x=x0;

y=y0;

printf("\nOUTPUT");

printf("\n x\t y\n");

while(x<xn){

y=y+h\*f(x,y);

x=x+h;

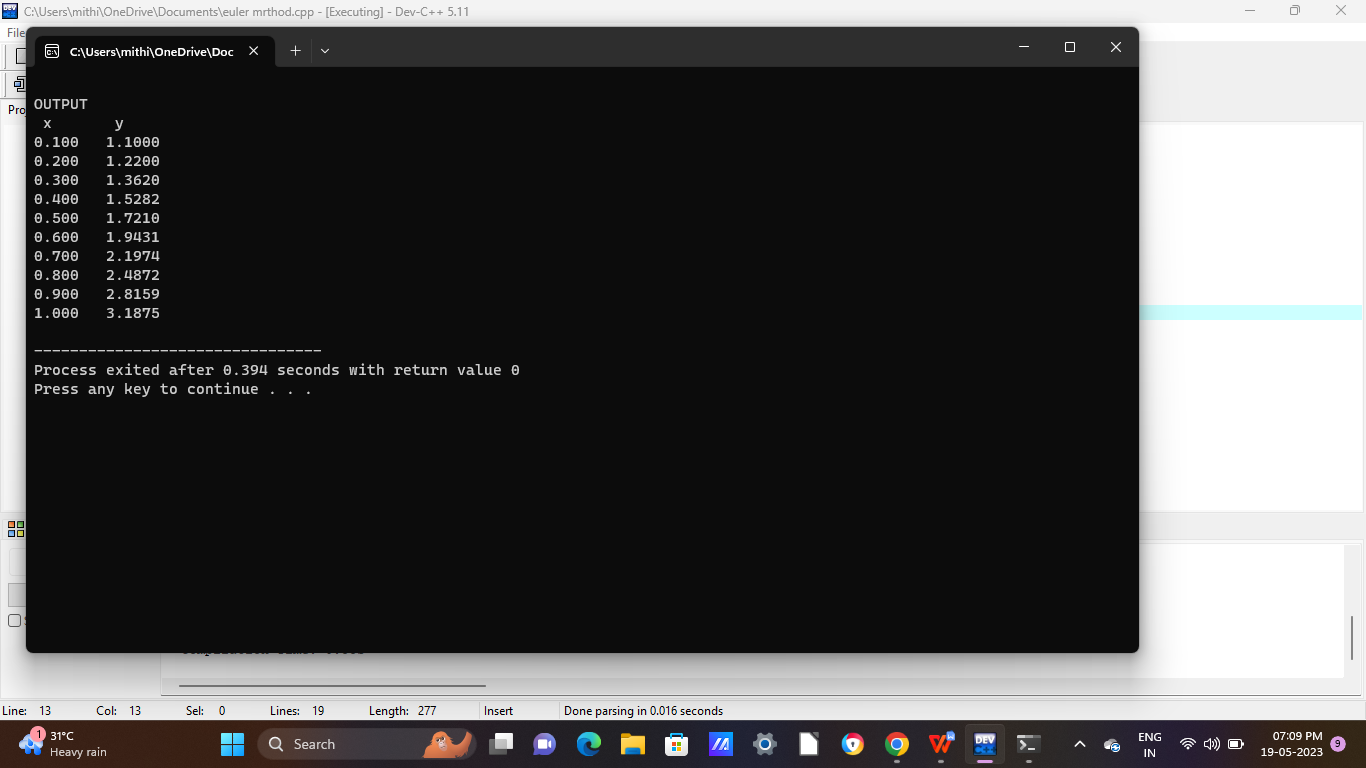
printf("%0.3f\t%0.4f\n",x,y);

}

return 0;

}

**Output:**



**Program 27:** Write a program in C to determine y(2.1) by Modified Euler’s Method with step length 0.05 correct upto 4 significant figures where

= 1 - with y(2)=2.

**Source Code:**

#include<stdio.h>

#include<math.h>

float fun(float,float);

int main(){

int i,j,c;

float x[100],y[100],h,n[100],m1,m2,a,s[100],w;

printf("\n C PROGRAM FOR MODIFIED EULER METHOD\n");

printf("Enter the initial value of x:");

scanf("%f",&x[0]);

printf("\n Enter the value of increment h:");

scanf("%f",&h);

printf("\n Enter the final value of x:");

scanf("%f",&a);

printf("Enter the initial value of y:");

scanf("%f",&y[0]);

s[0]=y[0];

for(i=1;x[i-1]<a;i++)

{

w=100.0;

x[i]=x[i-1]+h;

n[i]=fun(x[i-1],y[i-1]);

c=0;

while(w>0.0001){

m1=fun(x[i],s[c]);

m2=(n[i]+m1)/2;

s[c+1]=y[i-1]+m2\*h;

w=s[c]-s[c+1];

w=fabs(w);

c=c+1;

}

y[i]=s[c];

}

printf("\n\n The respective values of x and y are \n x\t y\n\n");

for(j=0;j<i;j++){

printf("%0.5f %0.5f",x[j],y[j]);

printf("\n");

}

}

float fun(float a,float b){

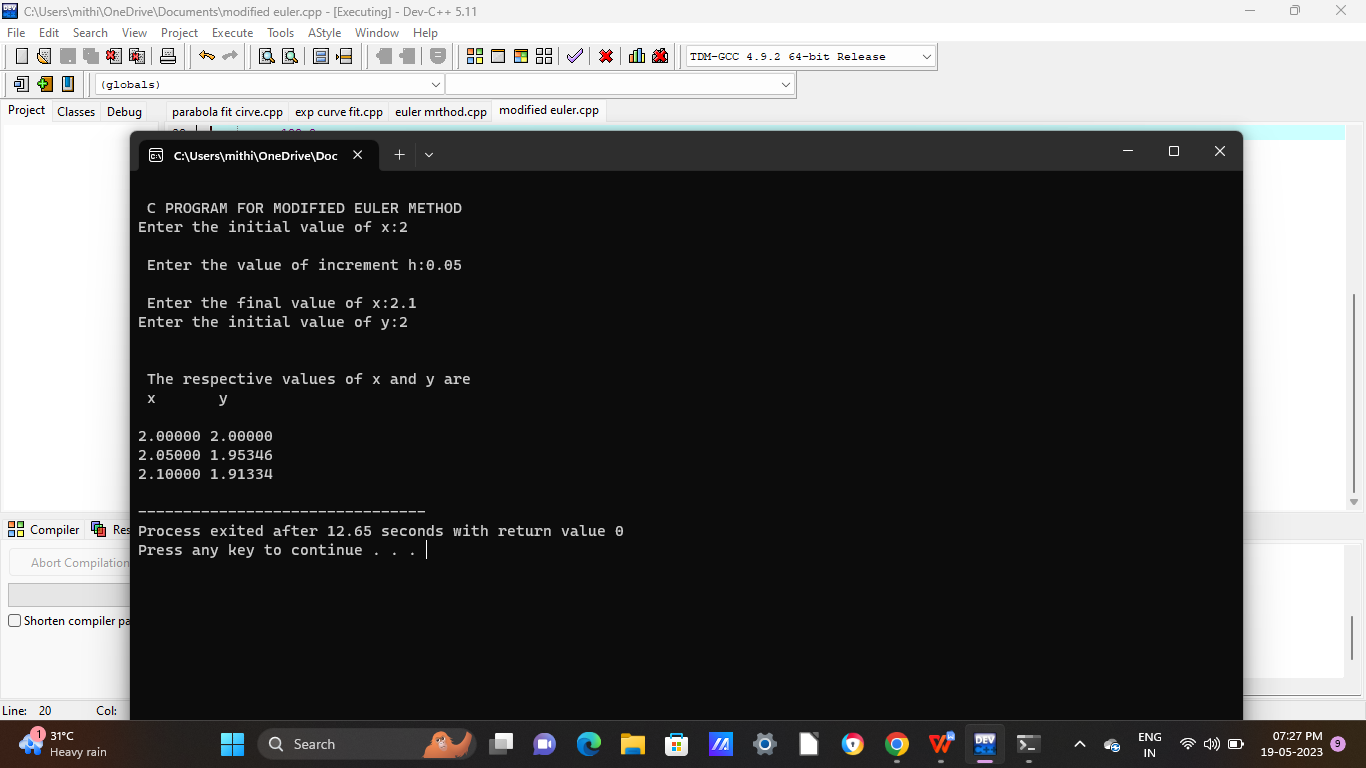
float c;

c=1-((b\*b)/a);

return(c);

}

**Output:**



**Program 28:** Write a program in C to determine y(0.4) using Runge Kutta Method of 4th order with h=0.2 ,given = , y(0)=0.8.

**Source Code:**

#include<stdio.h>

#include<math.h>

double f(double x, double y){

return sqrt((x\*x)+y);

}

int main()

{

int i;

double x,y,x0,y0,h,k1,k2,k3,k4;

printf("Enter the initial condition for y: ");

scanf("%lf",&y0);

printf("Enter the initial condition for x: ");

scanf("%lf",&x0);

printf("Enter the value of x for which y is required: ");

scanf("%lf",&x);

printf("Enter the step-width h: ");

scanf("%lf",&h);

printf("x\t\ty\t\tk1\t\tk2\t\tk3\t\tk4\t\tk\_avg\n");

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

while((x-x0)>0.0000000001)

{

k1=h\*f(x0,y0);

k2=h\*f(x0+h/2.0,y0+k1/2.0);

k3=h\*f(x0+h/2.0,y0+k2/2.0);

k4=h\*f(x0+h,y0+k3);

printf("%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%f\n",x0,y0,k1,k2,k3,k4,1/6.0\*(k1+2\*k2+2\*k3+k4));

y=y0+1/6.0\*(k1+2\*k2+2\*k3+k4);

y0=y;

x0=x0+h;

}

printf("%lf\t%lf\n",x0,y0);

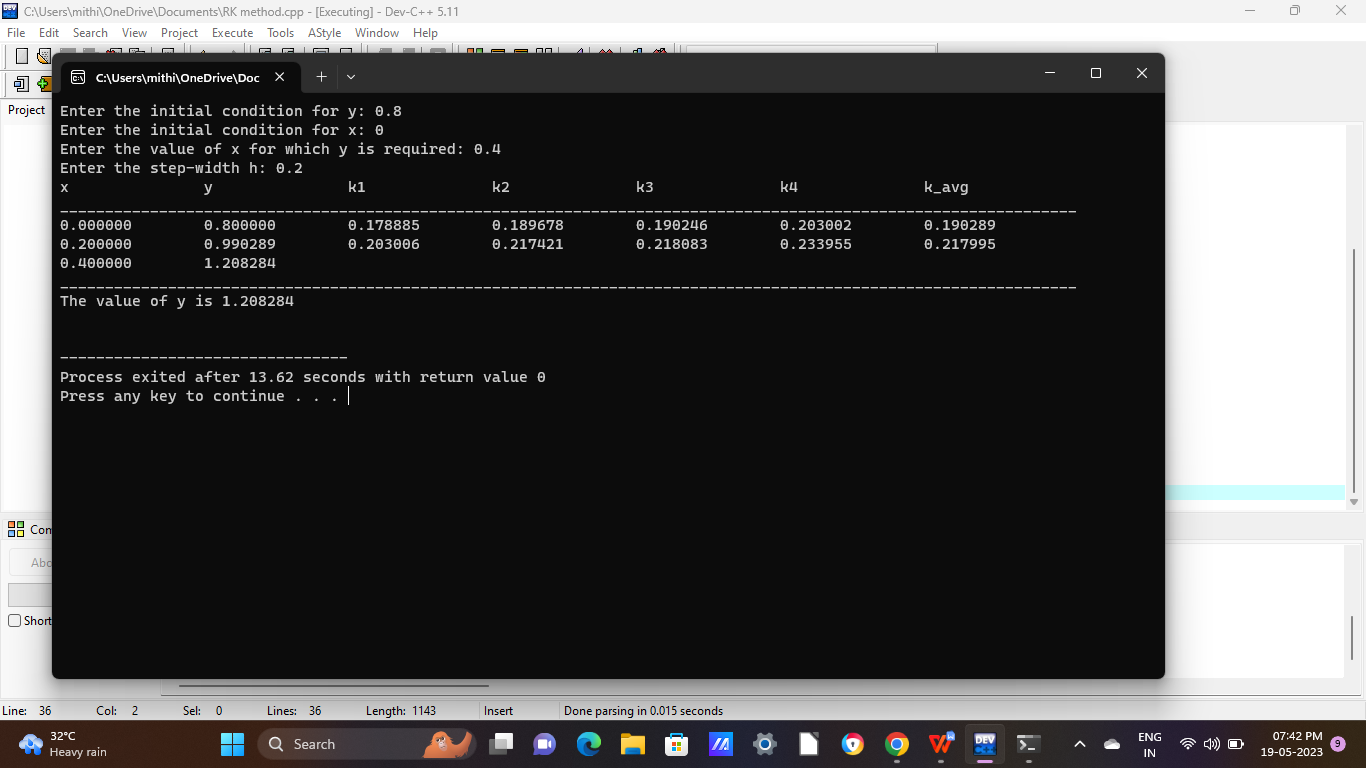
printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

printf("The value of y is %lf\n\n",y);

return 0;

}

**Output:**



**Program 29:** Write a program in C to determine y(0.1) using Picard’s Method upto 4 DP ,given

= 1+xy, y(0)=1.

**Source Code:**

#include<stdio.h>

#include<math.h>

float f1(float x)

{

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

}

float f2(float x)

{

return ((x)+pow(x,2)/2+pow(x,3)/3+pow(x,4)/8);

}

float f3(float x)

{

return ((x)+pow(x,2)/2+pow(x,3)/3+pow(x,4)/8+pow(x,5)/15+pow(x,6)/48);

}

int main()

{

float x0,xn,y0,a1,a2,a3;

printf("\n Enter the initial value of x: ");

scanf("%f",&x0);

printf("\n Enter the final value of x: ");

scanf("%f",&xn);

printf("\n Enter the initial value of y: ");

scanf("%f",&y0);

a1=y0+f1(xn);

printf("First approximation is: %0.4f\n",a1);

a2=y0+f2(xn);

printf("Second approximation is: %0.4f\n",a2);

a3=y0+f3(xn);

printf("Third approximation is: %0.4f\n",a3);

return 0;

}

**Output:**

