**GAUSS BACKWARD INTERPOLATION**

# include <stdio.h>

# include <conio.h>

# include <math.h>

# include <conio.h>

int main()

{

int n;

int i,j; float ax[10];

float ay[10];

float x;

float y=0;

float h;

float p;

float diff[20][20];

float y1,y2,y3,y4;

printf("\n Enter the number of terms - ");

scanf("%d",&n);

printf("\n Enter the value in the form of x - ");

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

{

printf("\n\n Enter the value of x%d - ",i+1);

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

}

printf("\n\n Enter the value in the form of y - ");

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

{

printf("\n Enter the value of y%d - ",i+1);

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

}

printf("\nEnter the value of x for - ");

printf("\nwhich you want the value of y - ");

scanf("%f",&x);

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

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

{

diff[i][1]=ay[i+1]-ay[i];

}

for(j=2;j<=4;j++)

{

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

{

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

}

}

i=n-1;

do {

i--;

}

while (ax[i]>x);

i++;

p=(x-ax[i])/h;

y1=p\*diff[i-1][1];

y2=p\*(p+1)\*diff[i-1][2]/2;

y3=(p+1)\*p\*(p-1)\*diff[i-2][3]/6;

y4=(p+2)\*(p+1)\*p\*(p-1)\*diff[i-3][4]/24;

y=ay[i]+y1+y2+y3+y4;

printf("\nwhen x=%6.4f,y=%6.8f ",x,y);

getch();

}

**GAUSS FORWARD INTERPOLATION**

#include<stdio.h>

#include<conio.h>

#include<math.h>

#include<conio.h>

int main()

{

int n; // no. of terms.

int i,j; // Loop variables

float ax[10]; // 'X' array limit 9

float ay[10]; // 'Y' array limit 9

float x; // User Query for what value of X

float y=0; // Calculated value for coressponding X.

float h; // Calc. Section

float p; // Calc. Section

float diff[20][20]; // to store Y

float y1,y2,y3,y4; // Formulae variables.

printf("\t\t !! GAUSS' FORWARD INTERPOLATION FORMULA !! ");

// Input section.

printf("\n\n Enter the no. of terms -> ");

scanf("%d",&n);

// Input Sequel for array X

printf("\n\n Enter the value in the form of x -> ");

// Input loop for X.

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

{

printf("\n Enter the value of x%d -> ",i+1);

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

}

// Input sequel for array Y.

printf("\n\n Enter the value in the form of y -> ");

// Input loop for Y.

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

{

printf("\n Enter the value of y%d -> ",i+1);

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

}

// Inputting the required value quarry

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

printf("\n which u want the value of y -> ");

scanf("%f",&x);

// Calculation and processing section.

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

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

{

diff[i][1]=ay[i+1]-ay[i];

}

for(j=2;j<=4;j++)

{

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

{

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

}}

i=0;

do

{

i++;

}

while(ax[i]<x);

i--;

p=(x-ax[i])/h;

y1=p\*diff[i][1];

y2=p\*(p-1)\*diff[i-1][2]/2;

y3=(p+1)\*p\*(p-1)\*diff[i-2][3]/6;

y4=(p+1)\*p\*(p-1)\*(p-2)\*diff[i-3][4]/24;

// Taking sum

y=ay[i]+y1+y2+y3+y4;

// Outut Section

printf("\n When x = %6.4f , y = %6.8f",x,y);

// Invoke user watch halt function

printf("\n\n\n\t\t\t !! PRESS ENTER TO EXIT !! ");

return 0;}

**ITERATION METHOD**

#include<stdio.h>

#include<math.h>

#include<conio.h>

double F(double x)

{

return (((x\*x\*x)-2\*x)+1);

}

double G(double x)

{

return (((x\*x\*x)+1)/2);

}

int main()

{

int i,n;

printf("enter no. of iterations\n");

scanf("%d",&n);

double x0,x1;

printf("enter the initial guess\n");

scanf("%f",&x0);

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

{

x1=G(x0);

printf("\n value after x %d iteration is %f",i,x1);

x0=x1;

}

printf("\n value of root after %d iterations=%1f,value of function %1f",n,x0,F(x0));

}

**NEWTON BACKWARD INTERPOLATION**

#include<stdio.h>

#include<conio.h>

int main()

{

float x[20], y[20][20];

int i,j, n;

//Input Section

printf("Enter number of data?\n");

scanf("%d", &n);

printf("Enter data:\n");

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

{

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

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

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

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

}

float a ; // interpolation point

printf("\n enter the value u want to find\n");

scanf("%f", &a);

float h, u, sum, p;

//Generating Backward Difference Table

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

{

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

{

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

}

}

//Displaying Backward Difference Table

printf("\nBACKWARD DIFFERENCE TABLE\n\n");

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

{

printf("%0.2f", x[i]);

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

{

printf("\t%0.2f", y[i][j]);

}

printf("\n");

}

p = 1.0;

sum = y[n - 1][0];

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

u = (a - x[n - 1]) / h;

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

p = p \* (u + j - 1) / j;

sum = sum + p \* y[n - 1][j];

}

printf("\nThe value of y at x=%0.1f is %0.3f", a, sum);

return 0;

}

**NEWTON FORWARD INTERPOLATION**

#include<stdio.h>

#include<conio.h>

int main()

{

float x[20], y[20][20];

int i,j, n;

// Input Section

printf("Enter number of data?\n");

scanf("%d", &n);

printf("Enter data:\n");

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

{

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

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

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

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

}

float a ; // interpolation point

printf("\n enter the value u want to find\n");

scanf("%f", &a);

float h, u, sum, p;

// Generating Forward Difference Table

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

{

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

{

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

}

}

// Displaying Forward Difference Table

printf("\nFORWARD DIFFERENCE TABLE\n\n");

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

{

printf("%0.2f", x[i]);

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

{

printf("\t%0.2f", y[i][j]);

}

printf("\n");

}

p = 1.0;

sum = y[0][0];

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

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

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

p = p \* (u - j + 1) / j;

sum = sum + p \* y[0][j];

}

printf("\nThe value of y at x=%0.1f is %0.3f", a, sum);

return 0;

}

**NEWTON RAPHSON**

#include <stdio.h>

#include <conio.h>

#include <math.h>

#include <process.h>

#include <string.h>

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

#define df(x) 3+sin(x)

void New\_Rap();

int main()

{

printf("\n solution by newton raphson method\n");

printf("\n equation is ");

printf("\n\t\t\t 3\*X-cos X-1=0\n\n");

New\_Rap();

getch();

}

void New\_Rap()

{

float x1,x0;

float f0,f1;

float df0;

int i=1;

int itr;

float EPS;

float error;

for(x1=0;x1+=0.01;)

{

f1=f(x1);

if(f1>0)

{

break;

}

}

x0=x1-0.01;

f0=f(x0);

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

scanf("%d",&itr);

printf("Enter the maximum possible error: \n");

scanf("%f",&EPS);

if(fabs(f0)>f1)

{

printf("\n\t\t The root is near to %.4f\n",x1);

}

if(f1>fabs(f(x0)))

{

printf("\n\t\t the root is near to %.4f\n",x0);

}

x0=(x0+x1)/2;

for(;i<=itr;i++)

{

f0=f(x0);

df0=df(x0);

x1=x0-(f0/df0);

printf("\n\t\t the %d approximation to the root is %f",i,x1);

error=fabs(x1-x0);

if(error<EPS)

{

break;

}

x0=x1;

}

if(error>EPS)

{

printf("\n\n\t NOTE:-");

printf("The number of iterations are not sufficient");

}

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

printf("\n\t\t\t The root is %.4f",x1);

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

}

**REGULA FALSI**

#include<stdio.h>

#include<math.h>

#include<conio.h>

#include<string.h>

#include<process.h>

#define EPS 0.00005

#define f(x) (x)\*log10(x)-1.2

void REG\_FAL();

int main()

{

printf("\n Solution of Regula-Falsi method\n");

printf("\n Equation is ");

printf("\n\t\t\t (x)\*log10(x)-1.2=0 \n\n ");

REG\_FAL();

}

void REG\_FAL()

{

float f0,f1,f2;

float x0,x1,x2;

int itr;

int i;

printf("enter the number of iterations:");

scanf("%d",&itr);

for(x1=0.0;;)

{

f1=f(x1);

if(f1>0)

{

break;

}

else

{

x1=x1+0.1;

}

}

x0=x1-0.1;

f0=f(x0);

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

printf("\n\t\t ITERATIONS \t x2 \t\t F(x) \n");

printf("\t\t---------------------------------------------");

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

{

x2=x0-((x1-x0)/(f1-f0))\*f0;

f2=f(x2);

if(f0\*f2>0)

{

x1=x2;

f1=f2;

}

else

{

x0=x2;

f0=f2;

}

if(fabs(f(2))>EPS)

{

printf("\n\t\t %d\t%f\t%f\n",i+1,x2,f2);

}

}

printf("\t\t---------------------------------------");

printf("\n\t\t\t\t Roots=%7.4f \n",x2);

printf("\t\t---------------------------------------");

getch();

}

**BISECTION METHOD**

#include<stdio.h>

#include<math.h>

#include<conio.h>

#include<process.h>

#include<string.h>

#define EPS 0.00000005

#define F(x) (x)\*log10(x)-1.2

void Bisect();

int count=1,n;

float root=1;

int main()

{

printf("\n Solution of Bisection method");

printf("\n eqaution is");

printf("\n\t\t\t (x)\*log10(x)-1.2=0\n\n");

printf("enter the number of iterations\n");

scanf("%d",&n);

Bisect();

getch();

}

void Bisect()

{

float x0,x1,x2;

float f0,f1,f2;

int i=0;

for(x2=1;;x2++)

{

f2=F(x2);

if(f2>0)

{

break;

}

}

for(x1=x2-1;;x2--)

{

f1=F(x1);

if(f1<0)

{

break;

}

}

printf("\t\t--------------------");

printf("\n\t\t iterations \t\t roots\n");

printf("\t\t----------------------");

for(;count<=n;count++)

{

x0=(x1+x2)/2.0;

f0=F(x0);

if(f0==0)

{

root=x0;

}

if(f0\*f1<0)

{

x2=x0;

}

else

{

x1=x0;

f1=f0;

}

printf("\n\t\t Iteration %d",count);

printf("\t :\t %f",x0);

if(fabs((x1-x2)/x1)<EPS)

{

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

printf("\n\t\t root=%f",x0);

printf("\n\t\t Iterations = %d\n",count);

printf("\t\t------------------------");

getch();

exit(0);

}

}

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

printf("\n\t\t\t root = %7.4f",x0);

printf("\n\t\t Iterations = %d\n ",count-1);

printf("\t\t-----------------------");

getch();

}

**SECANT METHOD**

#include<stdio.h>

#include<conio.h>

#include<math.h>

#include<stdlib.h>

/\* Defining equation to be solved.

Change this equation to solve another problem. \*/

#define f(x) (x)\*log10(x)-1.2

int main()

{

float x0, x1, x2, f0, f1, f2, e;

int step = 1, N;

/\* Inputs \*/

printf("\nEnter initial guesses:\n");

scanf("%f%f", &x0, &x1);

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

scanf("%f", &e);

printf("Enter maximum iteration:\n");

scanf("%d", &N);

/\* Implementing Secant Method \*/

printf("\nStep\t\tx0\t\tx1\t\tx2\t\tf(x2)\n");

do

{

f0 = f(x0);

f1 = f(x1);

if(f0 == f1)

{

printf("Mathematical Error.");

exit(0);

}

x2 = x1 - (x1 - x0) \* f1/(f1-f0);

f2 = f(x2);

printf("%d\t\t%f\t%f\t%f\t%f\n",step,x0,x1,x2, f2);

x0 = x1;

f0 = f1;

x1 = x2;

f1 = f2;

step = step + 1;

if(step > N)

{

printf("Not Convergent.");

exit(0);

}

}while(fabs(f2)>e);

printf("\nRoot is: %f", x2);

getch();

}