10/18/2022

**City University**

**Assignment**

**Submitted to**

**Submitted by**

Course code:

Course title:

Assignment no:

Assignment name:

**1. Bisection Method**

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

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

int main() {

float a = 0, b = 0, error = 0, c, c\_old;

int i = 0;

printf("Input Interval: ");

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

if ((f(a) \* f(b)) > 0){

printf("Invalid Interval Exit!"); exit(1);

}else if (f(a) == 0 || f(b) == 0){

printf("Root is one of interval bounds. Root is %f\n", f(a) == 0 ? a : b); exit(0);

}

printf("Ite\ta\t\tb\t\tm\t\tf(m)\t\terror\n");

do {

c\_old = c;

c = (a + b) / 2;

printf("%2d\t%4.6f\t%4.6f\t%4.6f\t%4.6f\t", i++, a, b, c, f(c));

if (f(c) == 0) {

printf("Root is %4.6f\n", c);

} else if ((f(a) \* f(c)) < 0) {

b = c;

} else a = c;

error = fabs(c - c\_old);

if (i == 1) {

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

} else printf("%4.6f\n", error);

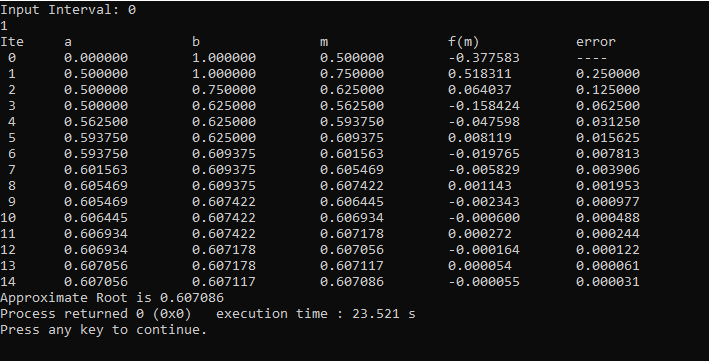
} while (error > 0.00005);

printf("Approximate Root is %4.6f", c);

return 0;

}

**Output:**

****

**2. Regular Falsi Method**

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

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

int main() {

float a = 0, b = 0, error = 0, c, c\_old;

int i = 0;

printf("Input Interval: ");

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

if ((f(a) \* f(b)) > 0) {

printf("Invalid Interval Exit!"); exit(1);

} else if (f(a) == 0 || f(b) == 0) {

printf("Root is one of interval bounds. Root is %f\n", f(a) == 0 ? a : b); exit(0);

}

printf("Ite\ta\t\tb\t\tm\t\tf(m)\t\terror\n");

do {

c\_old = c;

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

printf("%2d\t%4.6f\t%4.6f\t%4.6f\t%4.6f\t", i++, a, b, c, f(c));

if (f(c) == 0) {

printf("Root is %4.6f\n", c);

} else if ((f(a) \* f(c)) < 0) {

b = c;

} else a = c;

error = fabs(c - c\_old);

if (i == 1) {

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

} else printf("%4.6f\n", error);

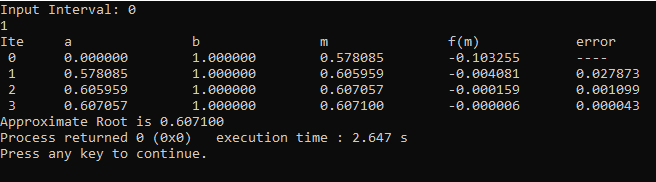
} while (error > 0.00005);

printf("Approximate Root is %4.6f", c);

return 0;

}

**Output:**

****

**3. Fixed Point Iteration**

#include <stdio.h>

#include <math.h>

#include <stdlib.h>

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

#define g(x) (cos(x) + 1) / 3

int main() {

int step = 1, N;

float x0, x1, e;

printf("Enter initial guess: ");

scanf("%f", &x0);

printf("Enter tolerable error: ");

scanf("%f", &e);

printf("Enter maximum iteration: ");

scanf("%d", &N);

printf("\nStep\tx0\t\tf(x0)\t\tx1\t\tf(x1)\n");

do {

x1 = g(x0);

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

step = step + 1;

if (step > N) {

printf("Not Convergent.");

exit(1);

}

x0 = x1;

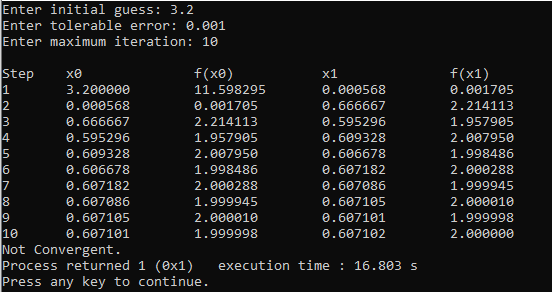
} while (fabs(f(x1)) > e);

printf("\nRoot is %f", x1);

return (0);

}

**Output:**

****

**4. Newton Raphson Method**

#include <stdio.h>

#include <math.h>

float f(float x) {

return (3 \* x - cos(x) - 1);

}

float df(float x) {

return 3 + sin(x);

}

int main() {

int itr, maxmitr;

float h, x0, x1, allerr;

printf("\nEnter x0, allowed error and maximum iterations\n");

scanf("%f %f %d", &x0, &allerr, &maxmitr);

for (itr = 1; itr <= maxmitr; itr++) {

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

printf("At Iteration no. %3d, x = %9.6f\n", itr, x1);

if (fabs(x1 - x0) < allerr) {

printf("After %3d iterations, root = %8.6f\n", itr, x1);

return 0;

}

x0 = x1;

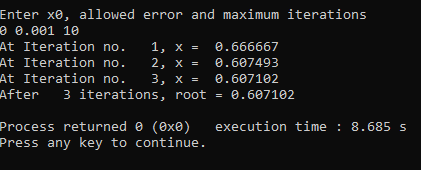
}

printf("The required solution does not converge or iterations are insufficient\n");

return 1;

}

**Output:**

****

**5. Forward Difference Table**

#include <stdio.h>

int main() {

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

int i, j, n;

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]);

}

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

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

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

}

}

printf("\n Forward Difference Table \n\n");

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

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

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

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

}

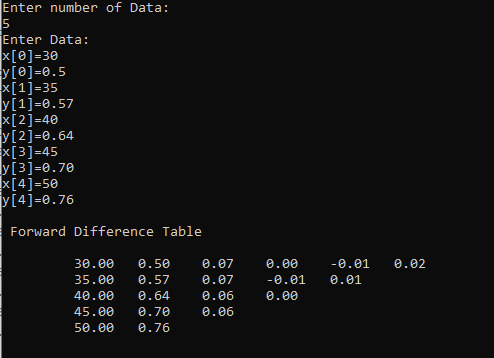
printf("\n");

}

return 0;

}

**Output:**



**6. Trapezoidal Rule**

#include <stdio.h>

#include <math.h>

float fn(float x) {

return x \* x + 1;

}

int main() {

int i, n;

float a, b, s = 0, y = 0, h;

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

scanf("%d", &n);

printf("Enter the lower limit: ");

scanf("%f", &a);

printf("Enter the upper limit: ");

scanf("%f", &b);

h = (b - a) / n;

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

s = s + fn(a + i \* h);

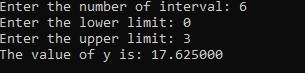
}

y = (fn(a) + fn(b) + 2 \* s) \* h / 2;

printf("The value of y is: %f", y);

}

**Output:**



**7. Simpsons Rule**

#include <stdio.h>

#include <math.h>

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

int main() {

float lower, upper, integration = 0.0, stepSize, k, int i, subInterval;

printf("Enter lower limit of integration : ");

scanf("%f", &lower);

printf("Enter upper limit of integration : ");

scanf("%f", &upper);

printf("Enter number of sub intervals : ");

scanf("%d", &subInterval);

stepSize = (upper - lower) / subInterval;

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

for (i = 1; i <= subInterval - 1; i++) {

k = lower + i \* stepSize;

if (i % 2 == 0) integration = integration + 2 \* f(k);

else integration = integration + 4 \* f(k);

}

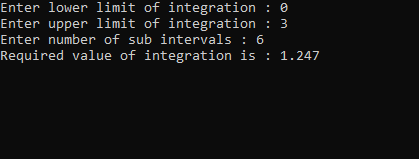
integration = integration \* stepSize / 3;

printf("Required value of integration is : %0.03f", integration);

return 0;

}

**Output:**



**8. Gaussian Elimination**

#include<stdio.h>

int main() {

int i, j, k, n;

float A[20][20], c, x[10], sum;

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

scanf("%d", &n);

printf("Enter the element of augmented matrix row-wise: \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(i=1; i<=n; i++) {

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

if(j>i) {

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

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

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

}

}

}

}

printf("The upper triangular matrix is: \n3");

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

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

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

}

printf("\n");

}

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

x[i] = 0;

}

printf("After applying backward substitution: \n");

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

sum = 0;

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

if(i!=j) sum = sum + A[i][j] \* x[j];

}

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

}

printf("The solution is: ");

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

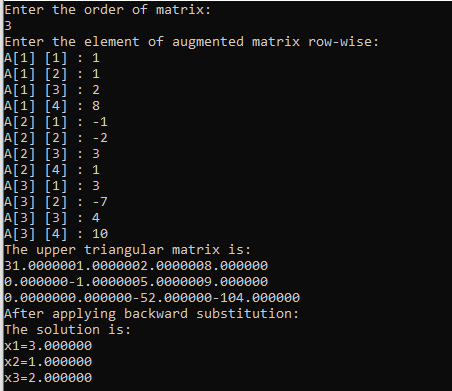
printf("\nx%d=%f\t", i, x[i]);

}

return 0;

}

**Output:**



**9. Forward Backward and Central Difference Method**

#include<stdio.h>

#include <math.h>

void fordwardDiff();

void backwardDiff();

void centraldDiff();

void options();

double f(double x) {

return sin(x);

}

int main() {

int c;

options();

scanf("%d", &c);

switch (c) {

case 1: fordwardDiff();

break;

case 2: centraldDiff();

break;

case 3: backwardDiff();

break;

default: return 0;

}

return 0;

}

void options () {

printf("Enter Choice \n");

printf("\t Enter 1 (forward Diff)\n ");

printf("\t Enter 2 (central Diff)\n ");

printf("\t Enter 3 (backward Diff)\n ");

printf(" Enter Choice ");

}

void fordwardDiff() {

double a, h;

printf("Enter a and h separator by space \n");

scanf("%lf %lf", &a, &h);

double res = (f(a + h) - f(a)) / (2 \* h);

printf("f(a)=%lf", res);

}

void centraldDiff() {

double a, h;

printf("Enter a and h separator by space \n");

scanf("%lf %lf", &a, &h);

double res = (f(a + h) - f(a - h)) / (2 \* h);

printf("f(a)=%lf", res); }

void backwardDiff() {

double a, h;

printf("Enter a and h separator by space \n");

scanf("%lf %lf", &a, &h);

double res = (f(a + h) - f(a)) / (2 \* h);

printf("f(a)=%lf", res);}

**Output:**

