Objective : Program for Lagrange’s Interpolation Method

Code :

#**include**<stdio.h>

**void** main()

{

**float** x[100], y[100], xp, yp=0, p;

**int** i,j,n;

clrscr();

/\* Input Section \*/

printf("Enter number of data: ");

scanf("%d", &n);

printf("Enter data:\n");

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

{

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

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

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

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

}

printf("Enter interpolation point: ");

scanf("%f", &xp);

/\* Implementing Lagrange Interpolation \*/

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

{ p=1;

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

**if** (i!=j) { p = p\* (xp - x[j])/(x[i] - x[j]); }

}

yp = yp + p \* y[i]; }

printf("Interpolated value at %.3f is %.3f.",xp,yp);

return 0; }

Output :

Enter number of data: 5

Enter data:

x[1] = 5

y[1] = 150

x[2] = 7

y[2] = 392

x[3] = 11

y[3] = 1452

x[4] = 13

y[4] = 2366

x[5] = 17

y[5] = 5202

Enter interpolation point: 9

Interpolated value at 9.000 is 810.000.

Algorithm :

1. Start

2. Read number of data (n)

3. Read data Xi and Yi for i=1 ton n

4. Read value of independent variables say xp

whose corresponding value of dependent say yp is to be determined.

5. Initialize: yp = 0

6. For i = 1 to n

Set p = 1

For j =1 to n

If i ≠ j then

Calculate p = p \* (xp - Xj)/(Xi - Xj)

End If

Next j

Calculate yp = yp + p \* Yi

Next i

6. Display value of yp as interpolated value.

7. Stop

Objective : Program for Trapezoidal Rule.

Code :

#**include**<stdio.h>

#**include**<math.h>

/\* Define function here \*/

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

**int** main()

{

**float** lower, upper, integration=0.0, stepSize, k;

**int** i, subInterval;

clrscr();

/\* Input \*/

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);

/\* Calculation \*/

/\* Finding step size \*/

stepSize = (upper - lower)/subInterval;

/\* Finding Integration Value \*/

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

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

{ k = lower + i\*stepSize;

integration = integration + 2 \* f(k); }

integration = integration \* stepSize/2;

printf("\nRequired value of integration is: %.3f", integration);

**return** 0; }

Output :

Enter lower limit of integration: 0

Enter upper limit of integration: 1

Enter number of sub intervals: 6

Required value of integration is: 0.784

Algorithm :

1. Start

2. Define function f(x)

3. Read lower limit of integration, upper limit of

integration and number of sub interval

4. Calcultae: step size = (upper limit - lower limit)/number of sub interval

5. Set: integration value = f(lower limit) + f(upper limit)

6. Set: i = 1

7. If i > number of sub interval then goto

8. Calculate: k = lower limit + i \* h

9. Calculate: Integration value = Integration Value + 2\* f(k)

10. Increment i by 1 i.e. i = i+1 and go to step 7

11. Calculate: Integration value = Integration value \* step size/2

12. Display Integration value as required answer

13. Stop