Birla institute of technology Mesra,Ranchi

Numerical method

Prepared by : Robin

ROLL NO : BTECH/15138/19

Faculty in charge

Dr. Prakash Chandra srivastava

Patna off campus

2021

Simpson’s 1/3rd rule

Algorithm:

1. Start.

2. Define an equation for f(x).

3. Define a method by the name of simpsonsRule().

4. Take the values of lower and upper limits of integration as well as the number of sub-intervals as inputs from the user.

5. Initialize a variable ifx with 0.

6. Find the value of h. [h = (b-a)/n]

7. Add the values of f(a) and f(b) to ifx.

8. Set the value of i = a+h.

9. Multiply the value of f(i) by 4 and add the result to ifx.

10. Add (2*h) to the existing value of i.

11. Repeat steps 8, 9 and 10 till the value of i is less than b.

12. Set the value of j = a+(2*h).

13. Multiply the value of f(j) by 2 and add the result to ifx.

14. Add (2*h) to the existing value of j.

15. Repeat steps 12, 13 and 14 till the value of j is less than b.

16. Multiply the final value of ifx with h and divide the result by 3 and store it back in ifx.

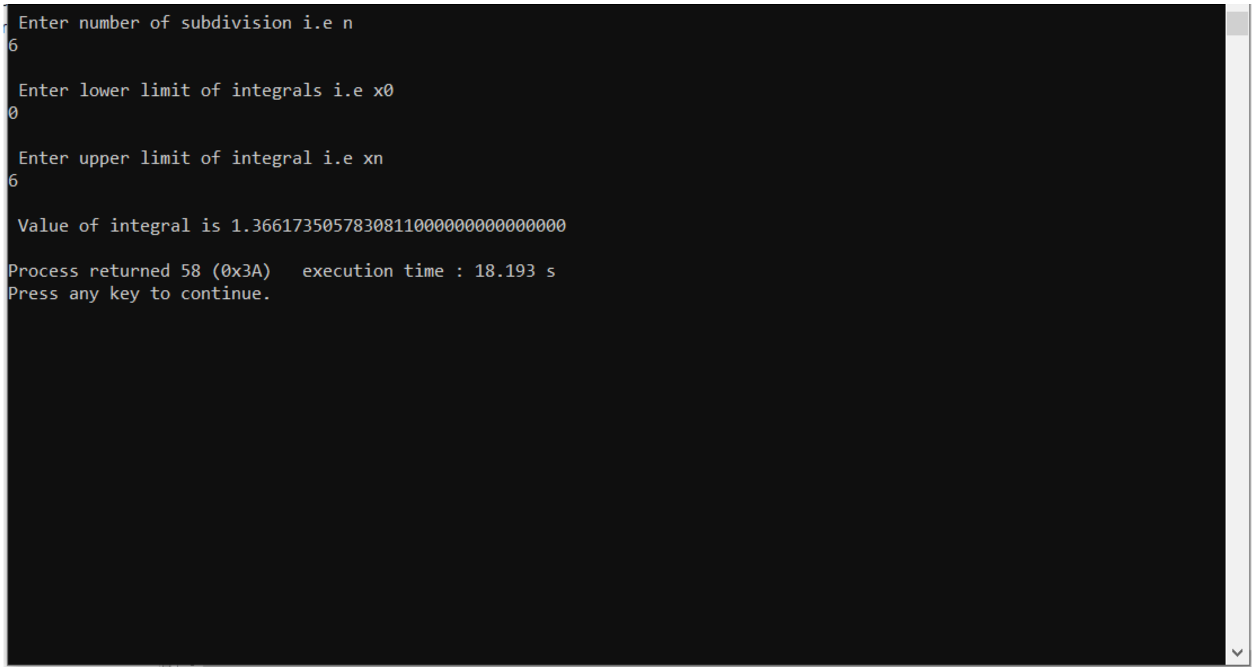
17. Print the value of integration ifx.

18. Stop.

Program:

```
//1/1+x^2
#include<stdio.h>
#include<conio.h>
 float y(float x)
 {
     return x*x/(1+x*x*x);
 }
void main()
{
    float x0,xn,h,s,sum;
    int i,n;
    puts("\n Enter number of subdivision i.e n");
    scanf("%d",&n);
    puts("\n Enter lower limit of integrals i.e x0");
    scanf("%f",&x0);
    puts("\n Enter upper limit of integral i.e xn");
    scanf("%f",&xn);

    h = (xn-x0)/n;
    s = y(x0)+ y(xn)+ 4*y(x0+h);

    for(i=3;i<=n-1;i+=2)
        s+=4*y(x0+i*h)+2*y(x0+(i-1)*h);
    sum=s*(h/3);
    printf("\n Value of integral is %0.31f \n",sum);
}
```

Output:



```
 Enter number of subdivision i.e n
6

 Enter lower limit of integrals i.e x0
0

 Enter upper limit of integral i.e xn
6

 Value of integral is 1.3661735057830811000000000000000

Process returned 58 (0x3A)   execution time : 18.193 s
Press any key to continue.
```

Trapezoidal rule

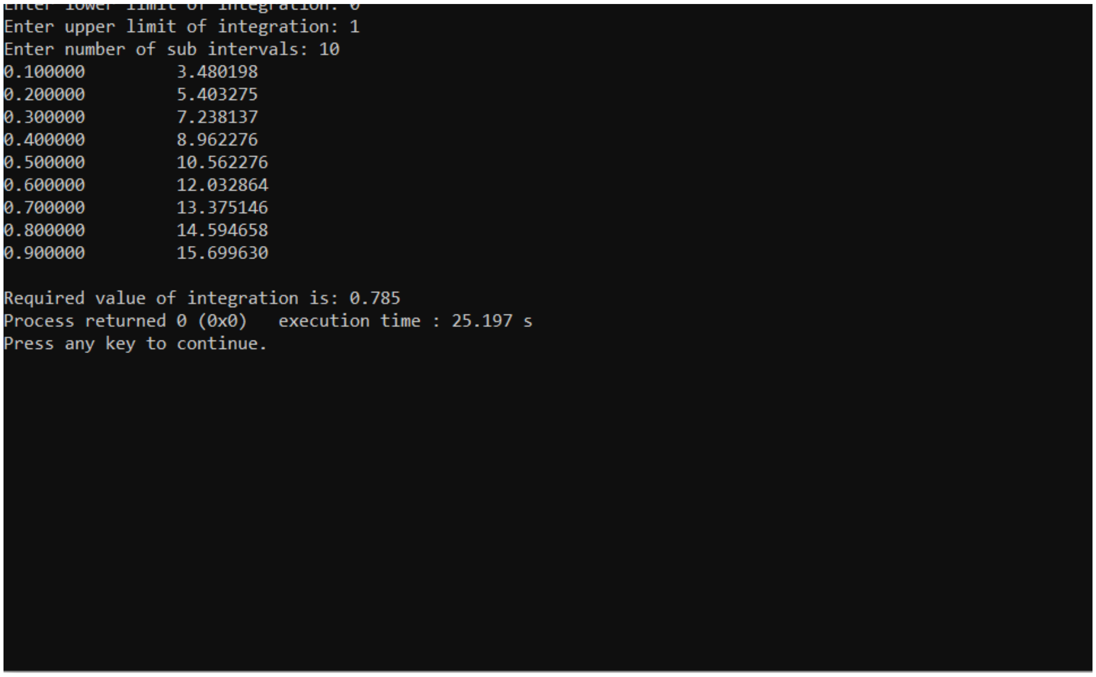
Algorithm:

```
1. Start
2. Input Lower limit a
3. Input Upper Limit b
4. Input number of sub intervals n
5. h=(b-a)/n
6.sum=0
7.sum=fun(a)+fun(b)
8.fori=1; i<n; i++
9.sum +=2*fun(a+i)
10. End Loop i
11.result =sum*h/2;
12. Print Output result
13. End of Program
14. Start of Section fun
15.temp = 1/(1+(x*x))
16. Return temp
17. Stop
```

code:

```
#include<math.h>
#define f(x) 1/(1+pow(x,2))

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;
integration = integration + 2 * f(k);
printf("%f\t %f\n",k,integration);
}
integration = integration * stepSize/2;
printf("\nRequired value of integration is: %.3f", integration);
getch();
return 0;
}
```

Output:



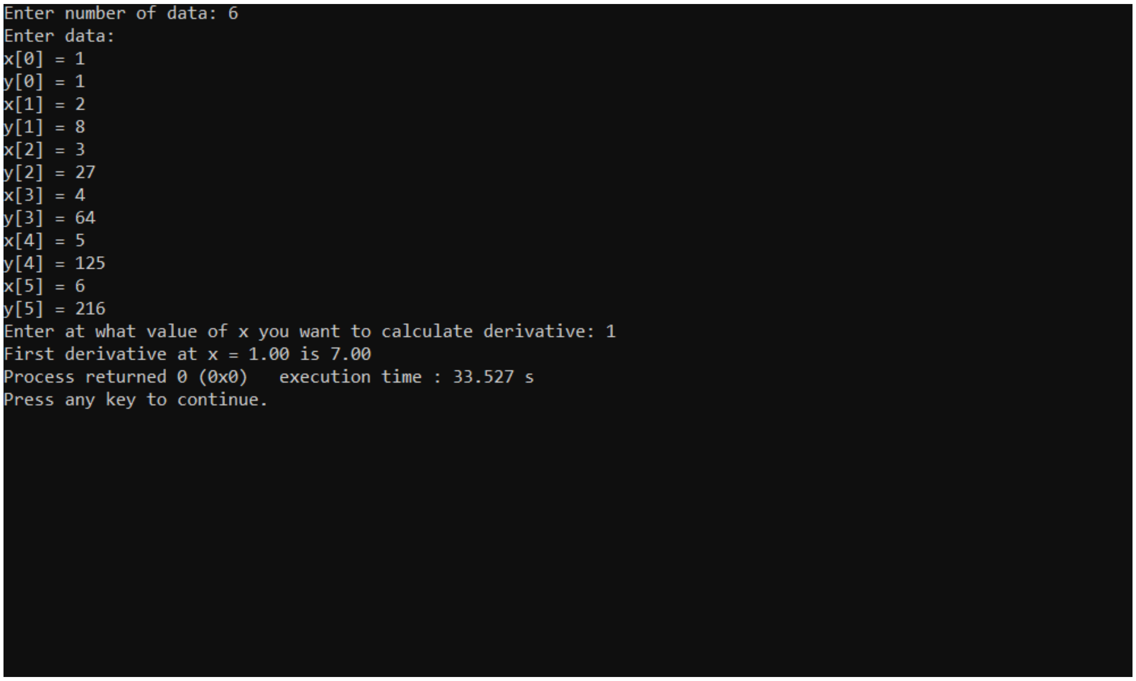
```
Enter lower limit of integration: 0
Enter upper limit of integration: 1
Enter number of sub intervals: 10
0.100000         3.480198
0.200000         5.403275
0.300000         7.238137
0.400000         8.962276
0.500000         10.562276
0.600000         12.032864
0.700000         13.375146
0.800000         14.594658
0.900000         15.699630

Required value of integration is: 0.785
Process returned 0 (0x0)   execution time : 25.197 s
Press any key to continue.
```

Numerical Differentiation

Code:

```
#include<stdio.h>

#include<stdlib.h>

#include<math.h>

#include<conio.h>

int main()

{

 float x[20], y[20][20], xp, h, sum=0.0, term, first_derivative;

 int i,j, n, index, flag = 0, sign=1;


 printf("Enter number of data: ");

 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]);
 }
 printf("Enter at what value of x you want to calculate derivative: ");
 scanf("%f", &xp);
 for(i=0;i< n;i++)
 {
     if (fabs(xp - x[i])< 0.0001)
     {
        index = i;
        flag = 1;
        break;
     }
 }
 if (flag==0)
 {
     printf("Invalid calculation point. Program exiting...");
     exit(0);
 }

 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];
  }
 }
 h = x[1] - x[0];
 for(i=1; i< n-index; i++)
 {
     term = pow(y[index][i], i)/i;
     sum = sum + sign*term;
     sign = -sign;
 }
 first_derivative = sum/h;
printf("First derivative at x = %0.2f is %0.2f", xp, first_derivative);
    getch();
 return 0;
}
```

Output:



```
Enter number of data: 6
Enter data:
x[0] = 1
y[0] = 1
x[1] = 2
y[1] = 8
x[2] = 3
y[2] = 27
x[3] = 4
y[3] = 64
x[4] = 5
y[4] = 125
x[5] = 6
y[5] = 216
Enter at what value of x you want to calculate derivative: 1
First derivative at x = 1.00 is 7.00
Process returned 0 (0x0)   execution time : 33.527 s
Press any key to continue.
```