TITLE: TO FIND FINITE INTEGRATION OF A FUNCTION USING NEWTON COTES'S FORMULA

Definition:

The **Newton-Cotes formula** is a numerical integration method that approximates the integral of a function by evaluating the function at equally spaced points and fitting a polynomial through these points to estimate the area under the curve.

Algorithm for Newton Cotes Formula:

1. **Input:**
   - Function $f(x)$ to integrate.
   - Lower limit a.
   - Upper limit b.
   - Number of subintervals n (equal to b in the code).
2. **Initialization:**
   - Calculate step size $h = \frac{b-a}{n}$.
   - Generate $n+1$ equally spaced points $x_i = a + (i-1) \times h$ for $i = 1$ to $n+1$.
   - Evaluate the function at these points to get $y_i = f(x_i)$.
3. **Integration using different Newton-Cotes rules:**
   - **Trapezoidal Rule:**
     - Approximate integral as

$$I = \frac{h}{2}\left(y_1 + y_{n+1} + 2\sum_{i=2}^{n} y_i\right)$$

   - **Simpson's 1/3 Rule:**
     - Approximate integral as

$$I = \frac{h}{3}\left(y_1 + y_{n+1} + 4\sum_{\text{even } i} y_i + 2\sum_{\text{odd } i} y_i\right)$$

   - **Simpson's 3/8 Rule:**
     - Approximate integral as

$$I = \frac{3h}{8}\left(y_1 + y_{n+1} + 3\sum_{i \not\equiv 1 \pmod 3} y_i + 2\sum_{i \equiv 1 \pmod 3} y_i\right)$$

Code:

```
%Title:-
% To find finite integration of a function using Newton's Cotes Quadrature
% Formula
%Developed by:-Arpan Adhikari
%Date: July 10,2025
%----------Three critical statements----------
close all;
clear variables;
clc;
%------------- User I/P section -------------
func = input('Enter function f(x)=');
f = inline(func);
%------------- User I/P section -------------
a = input('Enter lower limit a =');
b = input('Enter upper limit b =');
```

```
n = b;
h = (b-a)/n;
x = zeros(1, n+1);
y = zeros(1, n+1);
x(1) = a;
y(1) = f(x(1));
for i=2:n+1
    x(i) = x(1) + (i-1) * h;
    y(i) = f(x(i));
end
out = [x;y];
disp(out);
%------- Calculation section --------
%------- Trapezoidal Rule --------
I = 0;
for i = 2:n
    I = I + 2*y(i);
end
I = h/2*(y(1) + y(n+1) + I);
result = strcat('The integration by Trapezoidal rule is I = ', num2str(I));
disp('__________________________________');
disp(result);

%------- Simpsons 1/3 Rule --------
I = 0;
for i = 2:n
    if(mod(i,2) == 0)
        I = I + 4*y(i);
    else
        I = I + 2*y(i);
    end
end
I = h/3*(y(1) + y(n+1) + I);
result = strcat('The integration by Simpsons 1/3 rule is I = ', num2str(I));
disp(result);

%------- Simpsons 3/8 Rule --------
I = 0;
for i = 2:n
    if(mod(i-1,3) == 0)
        I = I + 2*y(i);
    else
        I = I + 3*y(i);
    end
end
I = (3*h)/8*(y(1) + y(n+1) + I);
result = strcat('The integration by Simpsons 3/8 rule is I = ', num2str(I));
disp(result);
```

Output:

```
Command Window
'1/(1+x^2)'
Enter lower limit a =
0
Enter upper limit b =
6
         0    1.0000    2.0000    3.0000    4.0000    5.0000    6.0000
    1.0000    0.5000    0.2000    0.1000    0.0588    0.0385    0.0270


__________________________________
The integration by Trapezoidal rule is I =1.4108
The integration by Simpsons 1/3 rule is I =1.3662
The integration by Simpsons 3/8 rule is I =1.3571
>>
```

Conclusion:

Newton-Cotes formulas allow us to numerically estimate definite integrals by evaluating the function at equally spaced points and applying polynomial-based rules. This practical demonstrates their effectiveness for approximating finite integrals when exact solutions are difficult.