Computer Excercise 1.1.1

The following program will evaluate the finite difference quotient of $\sin(x)$ for different values of h as an approximation for the first derivative of sine; the program will loop through different values of h multiplying the previous value of h by 0.25 on each iteration. The program will then evaluate the error of the approximation by using sine's known derivative, cosine, and will seek out and store a smallest error value and its corresponding index.

On each iteration, desired values will be stored in an array which will then be casted as a table for display.

```
%Initialize values

n=30;
x=0.5;
h = 1;
emin = 1;

%Initialize output arrays which will then be used to display a table
%This portion is for display purposes

i_out = (1:30)';
h_out = zeros(30,1);
y_out = zeros(30,1);
error_out = zeros(30,1);

%approximation program loop for sin(x)

for i = 1:n
    h = 0.25*h;
    y = (sin(x+h) - sin(x))/h;
    error = abs( cos(x) - y );

    %update the output arrays
    h_out(i) = h;
    y_out(i) = y;
    error_out(i) = error;

    %seek out minimum error and record its index
    if error < emin
        emin = error;
        imin = i;
    end
end



%cast output arrays as a table

T = table(i_out, h_out, y_out, error_out);
disp(T)
```

i_out	h_out	y_out	error_out
1	0.25	0.808852885676524	0.0687296762138483
2	0.0625	0.862034158909074	0.0155484029812989
3	0.015625	0.873801417582083	0.0037811443082898
4	0.00390625	0.876643953269792	0.000938608620581149
5	0.0009765625	0.877348327919719	0.000234233970653364
6	0.000244140625	0.87752402954743	5.85323429422857e-05
7	6.103515625e-05	0.877567930439	1.46314513731483e-05
8	1.52587890625e-05	0.877578904128313	3.65776205946133e-06
9	3.814697265625e-06	0.877581647451734	9.14438638588422e-07
10	9.5367431640625e-07	0.877582333283499	2.28606873875492e-07
11	2.38418579101562e-07	0.877582504646853	5.7243520146244e-08
12	5.96046447753906e-08	0.87758254725486	1.46355123575859e-08
13	1.49011611938477e-08	0.877582557499409	4.3909640368156e-09
14	3.72529029846191e-09	0.877582564949989	3.05961656010822e-09
15	9.31322574615479e-10	0.877582550048828	1.18415446337394e-08
16	2.3283064365387e-10	0.877582550048828	1.18415446337394e-08
17	5.82076609134674e-11	0.877582550048828	1.18415446337394e-08
18	1.45519152283669e-11	0.877582550048828	1.18415446337394e-08
19	3.63797880709171e-12	0.877578735351562	3.82653881025874e-06
20	9.09494701772928e-13	0.8775634765625	1.90853278727587e-05
21	2.27373675443232e-13	0.87744140625	0.000141155640372759
22	5.6843418860808e-14	0.8779296875	0.000347125609627241
23	1.4210854715202e-14	0.87890625	0.00132368810962724
24	3.5527136788005e-15	0.875	0.00258256189037276
25	8.88178419700125e-16	0.875	0.00258256189037276
26	2.22044604925031e-16	0.75	0.127582561890373
27	5.55111512312578e-17	0	0.877582561890373
28	1.38777878078145e-17	0	0.877582561890373
29	3.46944695195361e-18	0	0.877582561890373
30	8.67361737988404e-19	0	0.877582561890373

```
%display the minimum error and its index

fprintf('\n The minimum error occurs at i = %d has a value of %d \n', imin, emin)
```

```
 The minimum error occurs at i = 14 has a value of 3.059617e-09
```

From the above table, we see that more iterations does not necessarily correspond to less error. According to the table, the error decreases from i=1 until the "sweet spot" which occurs at i =14 from which it starts to increase for indicies larger than 14. This routine demonstrates an inherent limitation of computers' capabilities to deal with infinities and infinitesimals; theoretically, the error should approach zero as h approaches zero.