Table of Contents

Problem 1(A)

```
fprintf('\n\nProblem 1(A)\n\n');
% log(1.9) = log(1-x) => 1.9 = 1-x => x = -0.9
x = -0.9
```

```
Problem 1(A)


x =

  -9.0000e-01
```

Problem 1(B)

```
fprintf('\n\nProblem 1(B)\n\n');
% need 10 significant digits
error_bound=.5E-10;
% True value : log(1.9), x= -0.9
xtrue= log(1-x);
x= -0.9;
fprintf('k \t x_n \t\t Relerr\n');
% Use a Taylor series approximation for log(1-x) to evaluate log(1.9).
n=2;
xn=-x;
Relerr=abs(xtrue-xn)/xtrue;
fprintf('%d \t %0.11f \t %0.5e\n',n,xn,Relerr);
%Start a stopwatch timer.
tic
% Keep increasing the number of terms in the Taylor approximation till
 10
% significant digits of accuracy is obtained.
while Relerr>error_bound
  xn=xn- (x.^n)/n;
  Relerr=abs(xtrue-xn)/xtrue;
  n=n+1;
end
```

```
fprintf('%d \t %0.11f \t %0.5e\n',n,xn,Relerr);
% To save the results for Problem 1(E)
p1s = n;
p1v = xn;
% Read the stopwatch timer
toc
```

```
Problem 1(B)

k   x_n    Relerr
2   0.90000000000   4.02188e-01
175   0.64185388614   4.61738e-11
Elapsed time is 0.000669 seconds.
```

Problem 1(C)

```
fprintf('\n\nProblem 1(C)\n\n');
% log(1.9) = log((1+x)/(1-x)) => 1.9 = (1+x)/(1-x) => x = 0.3103448276
x = 0.3103448276
```

```
Problem 1(C)


x =

   3.1034e-01
```

Problem 1(D)

```
fprintf('\n\nProblem 1(D)\n\n');
% need 10 significant digits
error_bound=.1E-10;
% True value : log(1.9), x= 0.3103448276
xtrue= log((1+x)/(1-x));
x= 0.3103448276;
fprintf('k \t x_n \t\t Relerr\n');
% Use a Taylor series approximation for log(1-x) to evaluate log(1.9).
n=2;
xn=2*x;
Relerr=abs(xtrue-xn)/xtrue;
fprintf('%d \t %0.11f \t %0.5e\n',n,xn,Relerr);
%Start a stopwatch timer.
tic
% Keep increasing the number of terms in the Taylor approximation till
 10
% significant digits of accuracy is obtained.
while Relerr>error_bound
  xn=xn+ 2*((x.^(2*n -1))/(2*n -1));
```

```
  Relerr=abs(xtrue-xn)/xtrue;
  n=n+1;
end
fprintf('%d \t %0.11f \t %0.5e\n',n,xn,Relerr);
% To save the results for Problem 1(E)
p2s = n;
p2v = xn;
givenv = log(1.9);
% Read the stopwatch timer
toc
```

```
Problem 1(D)

k    x_n     Relerr
2    0.62068965520    3.29736e-02
11    0.64185388620    3.46842e-12
Elapsed time is 0.000630 seconds.
```

Problem 1(E)

According to the results, part (2) is more efficient for computing log(1.9) as first the time complexity for the part (2) took 0.000842 seconds to run 11 times compared to part (1), which took 0.002057 seconds to run 175 times. Although both are't the exact same as log(1.9) in a matter of the 11th floating point.

```
fprintf('\n\n\nActual Value: %0.11f \n', givenv)
fprintf('part (1)\n')
fprintf('Steps Took: %d \t Value: %0.11f \t \n', p1s, p1v)
fprintf('part (2)\n')
fprintf('Steps Took: %d \t Value: %0.11f \t \n\n\n', p2s, p2v)
```

```
Actual Value: 0.64185388617
part (1)
Steps Took: 175    Value: 0.64185388614
part (2)
Steps Took: 11    Value: 0.64185388620
```

Problem 2(A)

```
fprintf('\n\nProblem 2(A)\n\n');
syms n
% Compute f(x) = ((((4+x)^(1/2)) - 2)/x) for decreasing values of x
 from 10^(-1) and
% 10^(-20)
fprintf('\n\nTable:\n\n');
fprintf('x \t f(x)\n');
for i=1:20,x=10^(-i);
```

```
    fprintf('%g \t %0.10f\n',x,((((4+x)^(1/2)) - 2)/x))
end
% Compute a reformulation of f(x) = ((((4+x)^(1/2)) - 2)/x) in the
 form
% Rf(x)= (1/(((4+x)^(1/2)) + 2)), for decreasing values of x from
 10^(-1) and
% 10^(-20)
fprintf('\n\nTable:\n\n');
fprintf('x \t f(x) \t \t Rf(x)\n');
for i=1:20,x=10^(-i);
    fprintf('%g \t %0.10f \t %0.10f\n',x,((4+x)^(1/2) -2)/x,1/
((4+x)^(1/2) + 2))
end
% The final results are being rounded/chopped whenever we use the give
 function as
% computers tend to do so as they have finite inputs, so to fix this
% problem we would need to rewrite the function inorder to get the
 accurate
% answer.

% For this problem, the given function is ((((4+x)^(1/2)) - 2)/x) as
 it
% tends to go 0 for decreasing valyes of x from 10(-1) and 10^(-20),
 so I
% have rewritten the function to 1/((4+x)^(1/2) + 2) as it will go to
 0.25,
% which is the correct answer
```

Problem 2(A)

Table:

```
x   f(x)
0.1   0.2484567313
0.01   0.2498439450
0.001   0.2499843770
0.0001   0.2499984375
1e-05   0.2499998438
1e-06   0.2499999843
1e-07   0.2499999985
1e-08   0.2499999763
1e-09   0.2500000207
1e-10   0.2500000207
1e-11   0.2499778162
1e-12   0.2500222251
1e-13   0.2486899575
1e-14   0.2220446049
1e-15   0.0000000000
1e-16   0.0000000000
1e-17   0.0000000000
```

```
1e-18   0.0000000000
1e-19   0.0000000000
1e-20   0.0000000000


Table:

x   f(x)     Rf(x)
0.1   0.2484567313   0.2484567313
0.01   0.2498439450   0.2498439450
0.001   0.2499843770   0.2499843770
0.0001   0.2499984375   0.2499984375
1e-05   0.2499998438   0.2499998438
1e-06   0.2499999843   0.2499999844
1e-07   0.2499999985   0.2499999984
1e-08   0.2499999763   0.2499999998
1e-09   0.2500000207   0.2500000000
1e-10   0.2500000207   0.2500000000
1e-11   0.2499778162   0.2500000000
1e-12   0.2500222251   0.2500000000
1e-13   0.2486899575   0.2500000000
1e-14   0.2220446049   0.2500000000
1e-15   0.0000000000   0.2500000000
1e-16   0.0000000000   0.2500000000
1e-17   0.0000000000   0.2500000000
1e-18   0.0000000000   0.2500000000
1e-19   0.0000000000   0.2500000000
1e-20   0.0000000000   0.2500000000
```

Problem 2(B)

```
fprintf('Problem 2(B)');
% Compute f(x) = ((1- exp(-x))/x) for decreasing values of x from
 10^(-1) and
% 10^(-20)
fprintf('\n\nTable:\n\n');
fprintf('x \t f(x)\n');
for i=1:20,x=10^(-i);
    fprintf('%g \t %0.10f\n',x,((1- exp(-x))/x))
end
% Compute a reformulation of f(x) = ((1- exp(-x))/x) in the form
% Rf(x)= symsum( ( ((-1)^(n-1))(x^(n-1)) )/factorial(n), n, 0, 20 ),
 for decreasing values of x from 10^(-1) and
% 10^(-20)
fprintf('\n\nTable:\n\n');
fprintf('x \t f(x) \t \t Rf(x)\n');
for i=1:20,x=10^(-i);
    fprintf('%g \t %0.10f \t %0.10f\n',x,(1- exp(-x))/x, symsum((-
x)^(n-1)/factorial(n),n,1,20))
end
% The final results are being rounded/chopped whenever we use the give
 function as
% computers tend to do so as they have finite inputs, so to fix this
```

```
% problem we would need to rewrite the function inorder to get the
 accurate
% answer.

% For this problem, the given function is (1- exp(-x))/x as it
% tends to go 0 for decreasing valyes of x from 10(-1) and 10^(-20),
 so I
% have rewritten the function to symsum((-x)^(n-1)/
factorial(n),n,1,20) as
% it will go to 1, which is the correct answer

% Note that the reformulation uses symsum function in matlab, which
 aims to
% behave as a sum series
```

Problem 2(B)

Table:

x	f(x)
0.1	0.9516258196
0.01	0.9950166251
0.001	0.9995001666
0.0001	0.9999500017
1e-05	0.9999950000
1e-06	0.9999995000
1e-07	0.9999999495
1e-08	0.9999999939
1e-09	0.9999999717
1e-10	1.0000000827
1e-11	1.0000000827
1e-12	0.9999778783
1e-13	1.0003109452
1e-14	0.9992007222
1e-15	0.9992007222
1e-16	1.1102230246
1e-17	0.0000000000
1e-18	0.0000000000
1e-19	0.0000000000
1e-20	0.0000000000

Table:

x	f(x)	Rf(x)
0.1	0.9516258196	0.9516258196
0.01	0.9950166251	0.9950166251
0.001	0.9995001666	0.9995001666
0.0001	0.9999500017	0.9999500017
1e-05	0.9999950000	0.9999950000
1e-06	0.9999995000	0.9999995000
1e-07	0.9999999495	0.9999999500
1e-08	0.9999999939	0.9999999950
1e-09	0.9999999717	0.9999999995

```
1e-10   1.0000000827   0.9999999999
1e-11   1.0000000827   1.0000000000
1e-12   0.9999778783   1.0000000000
1e-13   1.0003109452   1.0000000000
1e-14   0.9992007222   1.0000000000
1e-15   0.9992007222   1.0000000000
1e-16   1.1102230246   1.0000000000
1e-17   0.0000000000   1.0000000000
1e-18   0.0000000000   1.0000000000
1e-19   0.0000000000   1.0000000000
1e-20   0.0000000000   1.0000000000
```

Published with MATLAB® R2017b