1. Consider the function \( f(x) = \log x \). The condition number is \( \kappa_f(x) = \left| \frac{1}{\log x} \right| \), which is large for \( x \approx 1 \). Numerically demonstrate that a small relative change in \( x \) can produce a much larger relative change in \( \log x \) for \( x \approx 1 \). Use the rule of thumb:

\[
\text{(relative forward error)} \lesssim \text{(condition number)} \times \text{(relative backward error)}.
\]

to explain your numerical results.

2. In this problem, we explore the conditioning of root-finding. Suppose \( f(x) \) and \( p(x) \) are smooth functions of \( x \in \mathbb{R} \), and \( x_* \) is a root of \( f(x) \), i.e., \( f(x_*) = 0 \).

(a) Due to error in evaluating \( f(x) \), we might compute roots of a perturbation \( f(x) + \varepsilon p(x) \).

If \( f'(x_*) \neq 0 \), for small \( \varepsilon \) we can write a function \( x(\varepsilon) \) such that \( f(x(\varepsilon)) + \varepsilon p(x(\varepsilon)) = 0 \) with \( x(0) = x_* \). Assuming such a function \( x(\varepsilon) \) exists and is differentiable, show that

\[
\frac{dx}{d\varepsilon} \bigg|_{\varepsilon=0} = -\frac{p(x_*)}{f'(x_*)}
\]

(b) Consider Wilkinson’s polynomial

\[
f(x) = (x - 1)(x - 2) \cdots (x - 20).
\]

We could have expanded \( f(x) \) in the monomial basis as \( f(x) = a_0 + a_1 x + \cdots + a_{20} x^{20} \). If we express the coefficient \( a_{19} \) in accurately, we could use the model from (a) with \( p(x) = x^{19} \) to predict how much root-finding will suffer. Show that

\[
\frac{dx}{d\varepsilon} \bigg|_{\varepsilon=0,x_*=j} = -\prod_{k \neq j} \frac{j}{j-k}
\]

(c) Compare \( \frac{dx}{d\varepsilon} \) from (b) for \( x_* = 1 \) and \( x_* = 20 \). Which root is more stable to the perturbation?

3. The power series for \( \sin x \) is

\[
\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots
\]

Here is a Matlab function that uses this series to compute \( \sin x \).

```matlab
function s = powersin(x)
% POWERSIN(x) tries to compute sin(x) from a power series
s = 0;
t = x;
n = 1;
while s + t ~= s;
s = s + t;
t = -x.^2/((n+1)*(n+2)).*t;
n = n + 2;
end
```
(a) What cases the while loop to terminate?

(b) Answer each of the following questions for $x = \pi/2, 11\pi/2, 21\pi/2$ and $31\pi/2$.
   - How accurate is the computed results?
   - How many terms are required?
   - What is the largest term (i.e, the last $t$) in the series?

(c) What do you conclude about the use of floating point arithmetic and power series to evaluate functions?

4. The roots of the quadratic function $ax^2 + bx + c$ are given by

   $$x_* = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

(a) Prove the alternative formula

   $$x_* = \frac{-2c}{b \pm \sqrt{b^2 - 4ac}}.$$

(b) Propose a numerical stable algorithm for finding the roots.

(c) Compare/comment your computed results for the following set of data:

   1) $a = 1, b = -56, c = 1$
   2) $a = 1, b = -10^8, c = 1$

5. Consider the function

   $$f(x) = \frac{e^x - 1}{x},$$

   which arises in various applications. By L'Hopital’s rule, we know that

   $$\lim_{x \to 0} f(x) = 1.$$

(a) Compute the values of $f(x)$ for $x = 10^{-n}$, $n = 1, 2, \ldots, 16$. Do your results agree with theoretical expectations? Explain why.

(b) Perform the computation in part (a) again, this time using the mathematically equivalent formulation

   $$f(x) = \frac{e^x - 1}{\log(e^x)},$$

   (evaluated as indicated with no simplification). If this works any better, can you explain why?

6. Program the five algorithms discussed in the class for the matrix-matrix multiply $C = C + AB$, where $A$, $B$ and $C$ are $n \times n$ matrix. Time them for random matrices for a set of dimensions. Verify that they yield the same solution but takes different amount of time (and different rates of flops).