The grader will grade the pledge problem and one randomly chosen problem.

(1) (25 pts) How would you perform the following calculations to avoid cancellation? Justify your answers.
   
i. Evaluate $\sqrt{x + 1} - 1$ for $x \simeq 0$.
   
ii. Evaluate $\sin(x) - \sin(y)$ for $x \simeq y$.
   
iii. Evaluate $\frac{1 - \cos(x)}{\sin(x)}$ for $x \simeq 0$.

(2) (25 pts)
Consider the polynomial $p(x) = (x - 2)^9 = x^9 - 18x^8 + 144x^7 - 672x^6 + 2016x^5 - 4032x^4 + 5376x^3 - 4608x^2 + 2304x - 512$.

   i. 5 pts Plot $p(x)$ for $x = 1.920, 1.921, 1.922, \ldots, 2.080$ (i.e. $x = [1.920 : 0.001 : 2.080]$) evaluating $p$ via its coefficients.
   
   ii. 5 pts Produce the same plot again, now evaluating $p$ via the expression $(x - 2)^9$.
   
   iii. 15 pts What is the difference? What is causing the discrepancy? Which plot is correct?

(3) (25 points) Rigorous theoretical proofs are required for these problems.
   
i (12 points) Sharpen the estimate shown in class by showing that if $x$ lies in the range of possible floating point numbers then
   
   $$fl(x) = x(1 + \delta), \quad |\delta| \leq \frac{1}{2}2^{-p-1},$$
   
   where $p$ is the precision.
   
   ii (13 points) Assume $fl(ab) = (ab)(1 + \delta_1)$, $fl(a + b) = (a + b)(1 + \delta_2)$, and $fl(a - b) = (a - b)(1 + \delta_3)$, where $\delta_i < \epsilon_M$ for $i = 1, 2, 3$. Which is the more accurate way to compute $x^2 - y^2$ : as $x^2 - y^2$ or $(x + y)(x - y)$? ($\epsilon_M$ denotes machine $\epsilon$.)

(4) (30 pts)
This problem is pledged! You may not discuss this with anyone but your instructor. You may not consult any source other than NA, NLA, Prof. Embree’s lecture notes or your in-class notes to help you with the problem.

   i. (10 pts) Code your own safeguarded Newton method and apply it to solve $f(x) = 0$ where

   $$f(x) = (x - 10)(x - 20)(x - 30).$$

   You should use the bisection method as a safeguard mechanism. You may assume the user will supply inputs $a, b$ with $a < b$ such that $f(a) * f(b) < 0$. Of course, your code must check this input and take appropriate action if this condition is not satisfied.
Your code should accept a general function name \textit{funder} such that the call \([f, df] = funder(x)\) returns the values \(f(x)\) in \(f\) and \(f'(x)\) in \(df\). Thus, the user should be able to input a function of their choice.

Your implementation should return a point \(x_k\) after \(k\) steps which satisfies \(|f(x_k)| < tol\) and \(|x_k - x_*| < tol\) where \(tol < 1\) is a user specified stopping tolerance and \(f(x_*) = 0\).

ii. (20 pts) Prove that the number of steps \(k\) needed to converge to a certain tolerance is finite and that once the bounding interval has been reduced to contain exactly one root \(x_*\), that eventually the iteration will be a pure Newton iteration which will achieve quadratic convergence to \(x_*\) if appropriate conditions are satisfied at \(x_*\). Please state what these conditions must be.

Your function call should be of the form

\[[x, xhist, niter] = SGNewton(funder, a, b, tol, maxit);\]

Input:
- \(funder\): a user supplied function
- \(a, b\): real numbers with \(a < b\) and \(f(a) \cdot f(b) < 0\)
- \(tol\): a positive real number (the stopping tolerance)
- \(maxit\): a positive integer specifying the max number of iterations allowed.

Output:
- \(x\): a real number (the approximate solution to \(f(x) = 0\) )
- \(xhist\): an array (a vector) of the values of \(x_k\) at each iteration \(k = 1, 2, \ldots\) up to the final iterate
- \(niter\): the number of iterations taken

Run the code three times with the same starting interval \(a = 0, b = 40\), but with three different values of \(tol = tol_1, tol_2, tol_3\). Set these valued of \(tol\) to assure that on return, the approximate solution \(x\) will have respectively 2, 8, 12 correct digits of accuracy. Take the right endpoint \(x_0 = 40\) as your starting guess for the Newton iteration in each of the three runs (Note: Each run should terminate in as few iterations as possible to meet the accuracy requirement).

Show the results in tabular form (one for each of the three values of \(tol\)) and visually verify that quadratic convergence of the iterates is taking place in the last table (for \(tol_3\)). The table should be of the following form with the \(x\) values displayed in Matlab’s long e format. Verify that quadratic convergence occurs for \(tol_3\).

<table>
<thead>
<tr>
<th>iteration</th>
<th>x value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(x_1)</td>
</tr>
<tr>
<td>2</td>
<td>(x_2)</td>
</tr>
<tr>
<td>3</td>
<td>(x_3)</td>
</tr>
</tbody>
</table>