Instructions: Try to solve all problems on your own. If you have difficulties, ask the instructor or TAs.

Exercise 1: Steepest Descent for Unconstrained Minimization
Suppose we want to find stationary points of a function $f : \mathbb{R}^n \to \mathbb{R}$, starting from a given point $x^0 \in \mathbb{R}^n$, using the steepest descent algorithm (covered in IE 601):

$k \leftarrow 0$;
while *not converged* **do**
  $d^k \leftarrow -\nabla f(x^k)$;
  $\alpha_k \leftarrow \text{argmin}_{\alpha \geq 0} f(x^k + \alpha d^k)$;
  $x^{k+1} \leftarrow x^k + \alpha_k d^k$;
  $k \leftarrow k + 1$;
end

1. What convergence criteria would you use for this algorithm?
2. Implement the above algorithm in a Python function (or a language of your choice). Your implementation should be independent of the function used. You may assume that the function you want to minimize is available as another Python function (evalf) that you can call from your program. Also assume that the function to evaluate gradient is available as well (gradf). **For solving the inner minimization problem to compute α_k, use your own implementation from your previous lab.**
3. Test your algorithm for the function $f : \mathbb{R}^2 \to \mathbb{R}$, defined as $f(x) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2$, starting from point $x^0 = (-1, -1)$.
4. [**R**] What are the stationary points of f above?
5. [**R**] How many iterations does your algorithm take to get close (according to your convergence criteria) to a stationary point?
6. [**R**] Plot the sequence of points $x^0, x^1, \ldots$ generated by your algorithm and report your observations.
7. [**R**] Repeat the same computations with different starting points: (0, 0), (3, 0), and (0.5, 0.5). Do you observe any pattern in the number of iterations required to converge to a stationary point?

Exercise 2:
Use your method to find the minimum value of each of the functions

$$
\begin{aligned}
f_1(x) &= \sum_{i=1}^{1000}(-40x_i + 3.0) + \sum_{i=1}^{999}(x_i^2 + x_{1000}^2)^2 \\
f_2(x) &= \sum_{i=1}^{10} \frac{x_i^2}{4000} - \prod_{i=1}^{10} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \\
f_3(x) &= \left(1 + (x_1 + x_2 + 1)^2 \left(19 - 14x_1 + 13x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)\right) \\
&\quad \times \left(30 + (2x_1 - 3x_2)^2 \left(18 - 32x_1 + 12x_1^2 - 48x_2 - 36x_1x_2 + 27x_2^2\right)\right)
\end{aligned}
$$

[**R**] For each of the functions, make a plot of the function values at each point that is visited. Use two different icons to identify the function values at the end of each major (outer) iteration and minor (inner) iteration. You should make one plot for each problem. Which of these problems is convex?

Exercise 3: Instead of heading in the direction given by the gradient, let us just search along only the coordinates one by one, i.e. our search directions are the n unit vectors $e_1, e_2, \ldots, e_n$, one for each variable in the problem.

$k \leftarrow 0$;
while *not converged* **do**
  $y^1 \leftarrow x^k$;
  for $i = 1, \ldots, n$ **do**
    $\alpha^* \leftarrow \text{argmin}_\alpha f(y^i + \alpha e_i)$;
    $y^{i+1} \leftarrow y^i + \alpha^* e_i$;
  end
  $k \leftarrow k + 1$;
  $x^k \leftarrow y^{n+1}$;
end

Implement this algorithm and test it for all the above instances. Compare the number of line searches required in this algorithm to those in steepest descent. Compare the number of iterations of steepest descent to number of outer loops of this algorithm. Note that α^* obtained in the inner loop can be either negative or positive. Explain how you find α^*. Discuss whether you need the the gradient of f at any stage of this algorithm.