# Exercise 1

## Task

Write a python script which finds the minimum of a given function using a steepest decent line search algorithm with optimal step size. The function to be minimized is

$$f(\mathbf{x}) = f(x_0, x_1) = (x_0 - 2x_1)^2 + (x_0 - 2)^2$$

A possible outline of the algorithm is given as follows:

**initialize** $i = 0$, $\mathbf{x}_i = \mathbf{x}_0$

**compute** $\nabla f(\mathbf{x}_i)$

**while** $\nabla f(\mathbf{x}_i) >$ tol:

> **set search direction** $\mathbf{s}_i = -\nabla f(\mathbf{x}_i)$
>
> **set** $j = 0$, $\alpha_j = 0$
>
> **while** $\nabla_\alpha f(\mathbf{x_i} + \alpha \mathbf{s}_i) >$ tol
>
> > **set** $a_{j+1} = a_j - \frac{\nabla_\alpha f(\mathbf{x}_i + \alpha_j \mathbf{s}_i)}{\nabla^2_\alpha f(\mathbf{x}_i + \alpha_j \mathbf{s}_i)}$
> >
> > **set** $j = j + 1$
>
> **set** $\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha_j \mathbf{s}_i$
>
> **set** $i = i + 1$

## Hints

- Use the template `Exercise1.py`
- Note that f is a two parametric function.
- Write separate subroutines for the function, its gradient and its hessian.
- The inner loop is a 1D-Newton iteration, which requires gradient and hessian with respect to $\alpha$, which also should be coded as separate subroutines.
- Try to visualize the minimization procedure.
- Replace the Newton loop with the Armijo-backtracking algorithm or a damped-Newton algorithm.
- Compare the performance of the different algorithms.
- Try other functions such as:
  - Simple quadratic functional: $f(\mathbf{x}) = x_0^2 + x_1^2$
  - Rosenbrook function: $f(\mathbf{x}) = (1 - x_0)^2 + 100 * (x_1 - x_0^2)^2$