(4) a) $h(x) = g(f(x))$, both $f(x)$ and $g(x)$ are convex over the domain

$$(g \cdot f)(\lambda x + (1-\lambda)y) = g(f(\lambda x + (1-\lambda)y))$$
$$\circledast \le g(\lambda f(x) + (1-\lambda) f(y))$$
$$\le \lambda g(f(x)) + (1-\lambda) g(f(y))$$
$$= \lambda (g \cdot f)(x) + (1-\lambda)(g \cdot f)(y) //$$

applying the convex definition

$\circledast$ this only holds assuming g preserves the monotony of the underlying argument, or in other words, if g is non-decreasing.

(5) a) $f(x_1, x_2) = 0.5 x_1^2 + x_2^2 + 2x_1 + x_2 + \cos(\sin(\sqrt{\pi}))$

$\nabla f = (x_1 + 2, 2x_2 + 1)$, since the function is convex we can say that $(-2, -1/2)$ is the minimizer without extra calculations

$$\nabla f = 0 \Leftrightarrow \begin{cases} x_1 + 2 = 0 \\ 2x_2 + 1 = 0 \end{cases} \Leftrightarrow \begin{cases} x_1 = -2 \\ x_2 = -1/2 \end{cases}$$

b) $x^{(1)} = x^{(0)} - \tau \cdot \nabla f(x^{(0)})$
$= (0,0) - 1 \cdot (2,1) = (-2,-1)$

$x^{(2)} = x^{(1)} - \tau \nabla f(x^{(1)})$
$= (-2,-1) - (0,-1)$
$= (-2,0)$

c) No, because it's going to keep alternating between $(-2,-1)$ and $(-2,0)$.
this happens because the learning rate is too high so we overshoot the actual solution to the problem and get stuck in a loop.
An obvious solution to this problem would be to use other methods of implementing gradient descent, such as adaptive g.d. or other more advanced methods, or simply using a smaller learning rate such as 0.1.

(7) a) The shaded region isn't convex since you can draw a line between two points in S and verify that the resulting line isn't contained in S. For example, $(1, 3.5) \longrightarrow (3.5, 6)$.

Take the formal definition:

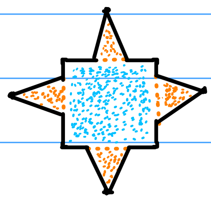
- S is convex iff $\forall x,y \in S: \lambda x + (1-\lambda)y \in S, \ \lambda \in [0,1]$

$\Rightarrow$ we can clearly confirm that taking $x = (1, 3.5)$; $y = (3.5, 6)$; $\lambda = 1/2$

$$1/2 \cdot (1, 3.5) + 1/2\,(3.5, 6) = (0.5 + 1.75 \ , \ 1.75 + 3) = (2.25, 4.75)$$

does not result in a point contained in S, which is equivalent to saying S is not convex.

b) Since the algorithm only works in convex regions, we assume it ignores the existence of local minimums. Such that the only way of using it to compute the global minimum of S, would be to break S into convex regions and applying the algorithm individually to each region s'. By comparing the solutions computed, it would be possible to compute the global minimum over S.



Example split of S resulting in only convex regions