1. We need to find constant $c > 0$ and $n_0 \geq 1$ interger such that $\frac{1}{n^2} \leq c \cdot \frac{1}{n}$ for all $n \geq n_0$

Simplify: $\frac{1}{n^2} \cdot n \leq c \cdot \frac{1}{n} \cdot n$ for all $n \geq n_0$

$\frac{1}{n} \leq c$ for all $n \geq n_0$

choose $c = 1$ to get

$\frac{1}{n} \leq 1$ for all $n \geq n_0$

the inequality is valid for all values of n,

so we can choose $n_0 = 1$

Since we have found constant $c = 1$, $n_0 = 1$ that make inequality true, then we have proven that $\frac{1}{n^2}$ is $O(\frac{1}{n})$.

2. We need to find constant $c > 0$ and $n_0 > 1$ integer such that $\frac{1}{g(n)} \leq c \cdot \frac{1}{f(n)}$ for all $n \geq n_0$.

Simplify: given that $f(n)$ is $O(g(n))$, then we can rewrite $c \cdot \frac{1}{f(n)}$ as $c \cdot \frac{1}{k \cdot g(n)}$ where k is a constant.

$\frac{1}{g(n)} \leq c \cdot \frac{1}{k\, g(n)}$

$(\frac{c}{k} - 1) \frac{1}{g(n)} \geq 0$. for all $n \geq n_0$.

choose $c = 2, k = 1$. then we can get

$\frac{1}{g(n)} \geq 0$ for all $n \geq n_0$.

the inequality is valid for all values of n, so we pick $n_0 = 1$

Since we have found constant $c=2$, $n_0=1$ that make inequality true, then we have proven that $\frac{1}{g(n)}$ is $O(f(\frac{1}{n}))$.

3. Assume that $n-1$ is $O(1)$. If $n-1$ is $O(1)$, there is a constant $c>0$ and $n_0 \geq 1$ such that

$n-1 \leq c$ for all $n \geq n_0$.

Simply the inequality, then we can have

$n \leq c-1$ for all $n \geq n_0$.

The inequality $n \leq c-1$ is valid only for values of n there are at most $c-1$, so this inequality can not be true for all vakes n larger than some constant n_0. Specifically, if we choose $n \geq c+1+n_0$, then note that these values of n are larger than or equal than n_0 but they are not at most c.

Therefore, we have reached a controdiction as there're no constant values $c>0$ and $n_0 \geq 1$ such that $n-1 \leq c$ for all $n \geq n_0$. Consequently, $n-1$ is not $O(1)$.

5. The algorithm will never terminate.

Assuming value x is not in L.

Then the iteration starts at $i=0$, the beginning of the array. It will keep increasing the value of i by 2 since x is not in L and loop through all even number position in L. Finally, i will be either n (while n is even) or $n-1$ (while n is odd). Then, the value of i is reset to 1 and still satisified $i<n$. Then, i will be 1, 3, 5 ... and loop through every odd number position in L. Finally, i will be either $n-1$ (while n is even) or n (while n is odd). Then i will still be reset to 1 and this algorithm will never terminate in any case.

6. The algorithm may not produce the correct output. For example, given the array $L=[1,2,3,4]$, find(L, 4, 4) would return -1 rather than 3.

n	Linear Search
5	104
10	152
100	542
1000	2689
10000	5602.
100000	18604

n	Quadratic Search
5	224.
10	363
100	5462
1000	76015
10000	6577391.

n	Factorial Search
7	1181300
8	8046700
9	36857500
10	348500600
11	3819685200
12	50206406000