Problem set 1.

1. Which of the following functions are $O(n^2)$ and which are $\Omega(n^2)$?
   $f_1(n) = 11n^2 + 100000$ $\qquad$ $f_2(n) = 8n^2 \log_2 n$ $\qquad$ $f_3(n) = 1.5n + 3\sqrt{n}$

2. Assume that in the following pseudocode, print one star counts as one step. Show that the number of steps is $O(n^3)$.

```
for i = 0 to n-1:
    for j = i+1 to n:
        print  j  copies of *
```

3. (a) We will check the running time of Bubble sort is $O(n^2)$. (This was introduced in Prog1, the pseudocode is repeated below). Is it also true that the running time is $O(n^3)$?
   (b) Further check that the number of steps is $\Omega(n^2)$.
   (c) Is it true that the number of steps or the running time is $\Theta(n^2)$?

```
for i = n to 2:
    for j = 1 to (i-1):
        if   A[j] > A[j+1]:
            swap A[j] and A[j+1]
```

---

4. Put the following functions in order of non-decreasing order of magnitude, i.e., if f_i is immediately follwed by f_j in the order, then $f_i(n) = O(f_j(n))$ holds.
   $f_1(n) = 8n^3$ $\qquad$ $f_2(n) = 5\sqrt{n} + 1000n$ $\qquad$ $f_3(n) = 2^{(\log_2 n)^2}$ $\qquad$ $f_4(n) = 1514n^2 \log_2 n$

5. The insertion sort of an array $A[1:n]$ (using Binary search) consists of $n-1$ loops: in the i^{th} loop, where $1 \le i \le n-1$, we search for the position of the $A[i+1]^{th}$ element in the array $A[1:i]$ using Binary search. We then swap $A[i+1]^{th}$ element with its neighbors until it reaches its desired position.
   (a) Convince yourself that the Binary search algorithm done in the lecture can be modified to obtain the position of the new element when its not in the array.
   (b) If the running time is counted in the number of comparisons and swaps, check that the running time of such a Insertion sort is $O(n^2)$.
   (c) Show that the running time is infact $\Omega(n^2)$ and hence, $\Theta(n^2)$.

6. Let $f_1(n) = 1.5\,n!$ and $f_2(n) = 200\,(n-1)!$. Which of the following is true?
   $f_1 \in O(f_2)$ $\qquad$ $f_2 \in O(f_1)$ $\qquad$ $f_1 \in \Omega(f_2)$ $\qquad$ $f_2 \in \Omega(f_1)$ $\qquad$ $f_1 \in \Theta(f_2)$ $\qquad$ $f_2 \in \Theta(f_1)$

7. Give a bound for the following functions using the Big-O notation:
   $(n^2+8)(n+1)$ $\qquad$ $(n\log n + n^2)(n^3+2)$ $\qquad$ $(n! + 2^n)(n^3 + \log(n^2+1))$ $\qquad$ $(2^n + n^2)(n^3 + 3^n)$

---

8. There are two algorithms for the same problem, A and B. Let the functions describing their worst case running times be f_A and f_B. It is known that $f_A(n) = O(f_B(n))$. Does it follow that
   a) A is faster on every input than B?
   b) except for finitely many inputs A is faster than B?
   c) for large enough inputs A is faster than B?

9. Show that if f_1, f_2, g_1, g_2 are positive valued functions, and $f_1(n) \in O(g_1(n))$ and $f_2(n) \in O(g_2(n))$, then,
   (a) $f_1(n) + f_2(n) \in O(\max(g_1(n), g_2(n))$
   (b) $f_1(n) \cdot f_2(n) \in O(g_1(n) \cdot g_2(n))$.

10. For which $a, b > 1$ natural numbers are the following true?
    $n^a \in O(n^b)$ $\qquad$ $2^{an} \in O(2^{bn})$ $\qquad$ $\log_a n \in O(\log_b n)$

11. Assume that n is a power of 2. Show that we can search for the two largest elements in an array of n distinct elements with $n + \log_2 n - 2$ comparisons.

12. An array $A[1:n]$ contains red and green elements. We would like to rearrange the elements so that all the red elements preceed the green elements, or all of them will come after the green elements. A legal step in rearranging them is swapping the position of two adjacent elements. We count swapping two adjacent elements as a step. Give an optimal (upto a constant factor) algorithm for this.

13. We are given n chips that can test each other in the following way: if two of the chips are connected, then both determine whether the other one is faulty or correct. A correct chip always answers properly about the other, while a faulty chip can give an arbitrary answer. Assume, that more than half of the chips are correct. Give an algorithm that uses less than n tests of the above type to find a correct chip.

Final Answers

1. yes for both, yes for Ω no otherwise, yes for O no otherwise.
2.
3. -,-, yes.
4. 2,4,1,3
5. no, yes, yes, no, no, no
6. $O(n^3), O(n^5), O(n^3 n!), O(6^n)$
7. no, no, no.
8.
9. $a \leq b$, $a \leq b$, for all values.