Problem 1

Order the following function by growth rate. $n!$, $n^2 + \sqrt{n}\ log^{10}n$, $n^{1/3}$, $log^{100}n$, n^3, 2^n, $10^{\sqrt{n}}$, 2^{logn}, 2^{2logn}, $2^{\sqrt{logn}}$, 128, $128n$. Indicate which functions at the same rate (all logarithms are base 2). For example, if you are asked to order n, $2n$, $2n^2$, then your answer should be "$n = \Theta(2n), 2n = o(2n^2)$".

1. 128
$$\lim_{n\to\infty}(\frac{128}{log^{100}n}) = 0,\ 128 = \mathcal{O}(log^{100}n)$$
2. $log^{100}n$
$$\lim_{n\to\infty}(\frac{log^{100}n}{2^{\sqrt{logn}}}) = 0,\ log^{100}n = \mathcal{O}(2^{\sqrt{logn}})$$
3. $2^{\sqrt{logn}}$
$$\lim_{n\to\infty}(\frac{2^{\sqrt{logn}}}{n^{1/3}}) = 0,\ 2^{\sqrt{logn}} = \mathcal{O}(n^{1/3})$$
4. $n^{1/3}$
$$\lim_{n\to\infty}(\frac{n^{1/3}}{128n}) = 0,\ n^{1/3} = \mathcal{O}(128n)$$
5. 2^{logn}
$$\lim_{n\to\infty}(\frac{2^{logn}}{128n}) = 0,\ 2^{logn} = \mathcal{O}(128n)$$
6. $128n$
$$\lim_{n\to\infty}(\frac{128n}{n^2 + \sqrt{n}log^{10}n}) = 0,\ 128n = \mathcal{O}(n^2 + \sqrt{n}log^{10}n)$$
7. 2^{2logn}
$$\lim_{n\to\infty}(\frac{2^{2logn}}{n^2 + \sqrt{n}log^{10}n}) = 0,\ 2^{2logn} = \mathcal{O}(n^2 + \sqrt{n}log^{10}n)$$
8. $n^2 + \sqrt{n}log^{10}n$
$$\lim_{n\to\infty}(\frac{n^2 + \sqrt{n}log^{10}n}{n^3}) = 0,\ n^2 + \sqrt{n}log^{10}n = \mathcal{O}(n^3))$$
9. n^3
$$\lim_{n\to\infty}(\frac{n^3}{10^{\sqrt{n}}}) = 0,\ n^3 = \mathcal{O}(10^{\sqrt{n}})$$
10. $10^{\sqrt{n}}$
$$\lim_{n\to\infty}(\frac{10^{\sqrt{n}}}{2^n}) = 0,\ = 10^{\sqrt{n}}\mathcal{O}(2^n)$$
11. 2^n
$$\lim_{n\to\infty}(\frac{2^n}{n!}) = 0,\ 2^n = \mathcal{O}(n!)$$
12. $n!$
$$\lim_{n\to\infty}(\frac{n!}{2^n}) = \infty,\ n! = o(2^n)$$

Problem 2

Solve the following recurrence equations, expressing the answer in Big-Oh notation. Assume that Tn is constant for sufficiently small n.

(a) $T(n) = T(n/2) + 100$

(b) $T(n) = 8T(n/2) + n^2$

(c) $T(n) = 8T(n/2) + n^3$

(d) $T(n) = 8T(n/2) + n^4$

(e) $T(n) = T(n-1) + logn$

(f) $T(n) = T(n-3) + n$

Problem 3

You implemented a quadratic time algorithm for a problem P. On a test run, your algorithm takes 50 seconds for inputs of size 1000. Your classmate found a clever algorithm solving the same problem with a running time $O(n^{3/2})$. However, the faster algorithm takes 150 seconds for input of size 1000. Explain how can this happen. If you need to solve a problem of size 4000, which algorithm you should use? What about input of size 10,000? Explain your answers (assume low-order terms are negligible).

Problem 4

Recall that in the *testing safe height to drop a cellphone* problem we discussed in the class, the goal is to find out the maximum safe height to drop a cellphone without breaking it. In the class we saw that if the maximum safe height is n, then in the worst case we can perform n tests if there is only one cellphone and $2\sqrt{n}$ tests if there are two cellphones. Give an algorithm to minimize the number of tests if there are k cellphones available (assume k is constant). How many tests do you need to perform?

Problem 5

You are given a set of n numbers. Give an $O(n^2)$ algorithm to decide if there exist three numbers a, b, and c in the set such that $a + b = c$ (Hint: sort the numbers first).