Problem 1 (30 points). For each pairs of functions below, indicate one of the three: $f = O(g)$, $f = \Omega(g)$, or $f = \Theta(g)$.

1. $f(n) = 100 \cdot n$, $g(n) = n + 10^4$
2. $f(n) = n^{1.01}$, $g(n) = n^{0.99} \cdot \log n$
3. $f(n) = n$, $g(n) = n^{0.99} \cdot (\log n)^2$
4. $f(n) = 100 \cdot \log(100 \cdot n^3)$, $g(n) = (\log n)^2$
5. $f(n) = n^2 \cdot \log n^2$, $g(n) = n \cdot (\log n)^3$
6. $f(n) = n^{2.01} \cdot \log n^3$, $g(n) = n^2 \cdot (\log n)^2$
7. $f(n) = (\log n)^{\log n}$, $g(n) = n^2$
8. $f(n) = (\log n)^{\log n}$, $g(n) = n^{\log \log n}$
9. $f(n) = (n \cdot \log n)^{\log n}$, $g(n) = n^{(\log n)^2}$
10. $f(n) = n^2 \cdot 2^n$, $g(n) = 3^n$
11. $f(n) = 2^{n \cdot \log n}$, $g(n) = 3^n$
12. $f(n) = n!$, $g(n) = (\log n)^n$
13. $f(n) = n!$, $g(n) = n^n$
14. $f(n) = \sum_{k=1}^{n} (1/k)$, $g(n) = \log n$
15. $f(n) = \sum_{k=1}^{n} k^2$, $g(n) = n^2 \cdot \log n$

Problem 2 (20 points). Solve each of the following recursions.

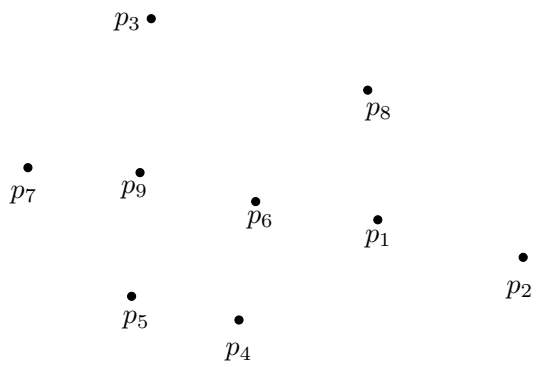
1. $T(n) = 9 \cdot T(n/2) + n^3$
2. $T(n) = 8 \cdot T(n/3) + n^2$
3. $T(n) = 2 \cdot T(n/4) + n^{0.5}$
4. $T(n) = 2 \cdot T(n/2) + n^{1.5}$
5. $T(n) = 4 \cdot T(n/2) + n \cdot \log n$

Problem 3 (10 points). You are given an array with n integers in which some integers might appear more than once. Design an algorithm to remove all integers that appear more than once. For example, if you are given $(6,3,4,2,4,3,5,3)$, then your algorithm should return $(6,2,5)$. Your algorithm should run in $O(n \cdot \log n)$ time.

Problem 4 (20 points). You are given m sorted arrays, each of which contains n integers. You are asked to merge them into a single sorted array. You may assume that $m = 2^k$.

1. Consider the following recursive algorithm: use the linear-time algorithm (the "merge" step in the divide-and-conquer merge-sort algorithm) to merge the first 2 sorted arrays to obtain a single sorted array, then use the same algorithm (again the "merge" step in merge-sort) to merge this sorted array with the third array, and so on, until all arrays are processed. Analyze the running time of this algorithm.
2. Design a faster algorithm using divide-and-conquer technique, and give its running time.

Problem 5 (10 points). Run the Graham-Scan algorithm on the following instance: draw the status of the stack as you process each point.



Problem 6 (10 points).

1. Design an instance of the convex hull problem such that the run of the Graham-Scan algorithm on your instance will not execute the *pop* operation. Your instance should contain 5 points on 2D plane and any three of them are not on the same line.

2. Design an instance of the convex hull problem such that the run of the Graham-Scan algorithm on your instance will execute the *pop* operations exactly twice. Your instance should contain 5 points on 2D plane and any three of them are not on the same line.