UMC 201 practice sheet 1

August 2026

Assume logarithm to base 2 in all the problems.

1. Let $f(n)$ and $g(n)$ be asymptotically positive functions. Prove or disprove the following claims:
   (a) If $f(n) = O(h_1(n))$ and $g(n) = O(h_2(n))$, then $f(n) + g(n) = O(\max\{h_1(n), h_2(n)\})$.
   (b) $f(n) + o(f(n)) = \Theta(f(n))$.
   (c) $f(n) = \Theta(f(n/3))$.
   (d) $f(n) = O(g(n))$ implies $\log(f(n)) = O(\log(g(n))$ where, for sufficiently large n, $\log(g(n)) \geq 1$ and $f(n) \geq 1$.
   (e) $f(n) = O(f^2(n))$.

2. Arrange the following functions in increasing order of asymptotic complexity, that is, the functions should be ordered as $g_1, g_2 \ldots$ where $g_i = O(g_{i+1})$.

2026	2^n	$2^{\sqrt{\log n}}$	$\log(\log(n))$
$n \log n$	n^{90}	$n^{1.5}$	$2^{5 \log \log n}$
$n2^{2n}$	$n^{\log^2 n}$	$(\log n)^{\log n}$	$n^{3/\log(n)}$

3. Let $P = \{p_1, p_2, \ldots, p_n\}$ and $Q = \{q_1, q_2, \ldots, q_n\}$ be two sets of n points each, where the points in P and Q lie on two parallel lines. Create the set $L = \{\ell_1, \ell_2 \ldots, \ell_n\}$ of n line segments by connecting p_i to q_i. Give a divide and conquer algorithm to determine the number of intersecting line segments in $O(n \log n)$ time.

4. In this problem, we look at a generalisation of heaps called d-ary heaps. A d-ary heap is like a binary heap, but where the nonleaf nodes have d children with one possible exception. Assume a $O(1)$ time cost per operation for maintaining the mapping between objects and heap elements.
   (a) Represent a d-ary heap in an array.
   (b) Use Θ-notation to express the height of a d-ary heap of n elements in terms of n and d.

(c) Implement the EXTRACT-MAX, INCREASE-KEY and INSERT operations in a d-ary max-heap. Analyse the runtime of all operations in terms of d and n.

5. In mergesort, you saw that merging two sorted lists formed the combine part of the "divide, conquer and combine" strategy. In this problem, you will show a lower bound of $2n-1$ on the worst-case number of comparisons required to merge two sorted lists, each containing n items.

   (a) Given $2n$ numbers, compute the number of possible ways to divide them into two sorted lists, each with n numbers.

   (b) Using a decision tree and your answer to part (a), show that any algorithm that correctly merges two sorted lists must perform at least $2n - o(n)$ comparisons.

   (c) Show that if two elements are consecutive in the sorted order and from different lists, then they must be compared.

   (d) Use your answer to part (c) to show an improved lower bound of $2n - 1$ comparisons for merging two sorted lists.

6. The quicksort procedure as described in the lectures makes two recursive calls to itself. This can lead to Quicksort using $\Theta(n)$ memory because the recursion depth can be $\Theta(n)$. In this problem, you will see a variant of quicksort called the *tail recursive* quicksort, which uses less memory by eliminating the second recursive call.

Algorithm 1 Tail Recursive Quicksort

```
1: function TRE-QSort(A, p, r)
2:     while p < r do
3:         q = PARTITION(A, p, r)
4:         TRE-QSort(A, p, q − 1)
5:         p = q + 1
6:     end while
7: end function
```

   (a) Argue that Algorithm 1 correctly sorts the array A.

   (b) Describe a scenario in which Algorithm 1 makes $\Theta(n)$ recursive calls on an n-element input array

   (c) Modify Algorithm 1 so that the number of recursive calls in the worst-case is $\Theta(\log n)$. Maintain the $O(n \log n)$ expected running time of the algorithm.