SAMPLE PROBLEMS

— Read all of the instructions on this page.

— You can consult the following resources for this exam: your notes, content that is hosted on the course website. You are not allowed to use the internet or ask anyone else for help on these problems (except the course staff for clarification).

— Use exactly 2 pages for each problem.

PROBLEM 1 *Solve [15 pts]*

Solve the recurrences in parts **(a)** to **(d)**, expressing your answers using Θ-notation. Whenever possible, apply the Master Theorem and state which case you used. Assume that $T(n) = \Theta(1)$ for small n, and that (when applicable) the Regularity Condition is met.

(a) $T(n) = 3T(n/3) + \sqrt{n}$

(b) $T(n) = 4T(n/2) + 3n^2$

(c) $T(n) = T(n^{1/2}) + \log\log n$

(d) $T(n) = 4T(n/5) + n^2$

(e) $T(n) = T(n/2) + T(4n/5) + T(7n/10) + n$

PROBLEM 2 *Divide and Conquer [20 points]*

An array $A[1..n]$ is ***has one mode*** if it consists of an increasing sequence followed by a decreasing sequence, or more precisely, if there is an index $m \in \{1, 2, ..., n\}$ such that

- $A[i] < A[i+1]$ for all $i < m$, and
- $A[i] > A[i+1]$ for all $i \geq m$

In particular, $A[m]$ is the maximum element, and it is the unique element surrounded by smaller elements ($A[m-1]$ and $A[m+1]$). For example, the array 3,7,9,10,15,6,5,4,1 has one mode.

(a) Give a divide-and-conquer algorithm to compute the maximum element input array $A[1..n]$ that has one mode in $O(\log n)$ time. First, explain your algorithm concisely in English. Second, specify your algorithm using pseudocode.

(b) Explain in a couple of sentences why your algorithm is correct.

(c) Give (and solve) a recurrence for the running time of your algorithm.

PROBLEM 3 *Analyze [20 pts]*

Your goal is to understand the following algorithm and analyze its running time.

Initial call: MYSTERY-ALG$(A, 1, length(A))$

MYSTERY-ALG(A, p, q)

▷ A is an array of integers (could be negative);
▷ p and q are indices between 1 and $length(A)$ such that $p \leq q$.

1 **if** $p = q$ **then return** $|A[p]|$
▷ Here $|\cdot|$ denotes absolute value.

2 $m \leftarrow \lfloor \frac{p+q}{2} \rfloor$
3 $left \leftarrow$ MYSTERY-ALG(A, p, m)
4 $right \leftarrow$ MYSTERY-ALG$(A, m+1, q)$

5 Create two new arrays $B[p\,..\,m]$ and $C[m+1\,..\,q]$
▷ Initialize B:
6 $B[m] \leftarrow A[m]$
7 **for** $i \leftarrow m$ **downto** p
8 **do** $B[i] \leftarrow B[i+1] + A[i]$
▷ Initialize C:
9 $C[m+1] \leftarrow A[m+1]$
10 **for** $i \leftarrow m+1$ **to** q
11 **do** $C[i] \leftarrow C[i-1] + A[i]$

12 MERGESORT(B, p, m)
13 MERGESORT$(C, m+1, q)$
14 $middle \leftarrow$ LIGHTEST-SUM(B, C, p, m, q)
▷ LIGHTEST-SUM takes two sorted arrays of integers and three indices p, m, q,
▷ and returns $\min\limits_{\substack{p \leq i \leq m \\ m+1 \leq j \leq q}} \left|B[i] + C[j]\right|$ (i.e., it returns the absolute value of the closest
▷ number to zero one can get by adding a number from B to a number from C).

15 **return** $\min(left, middle, right)$

(a) Suppose $A = [\,2\ \ 3\ \ 4\ \ 5\ \ \text{-}15\ \ 2\ \ 3\ \ 4\,]$. What is the output of the initial call?

(b) What does MYSTERY-ALG compute in general?

(c) Assume that LIGHTEST-SUM runs in time $O(n)$ in the worst case. State and solve a recurrence which describes the running time of the MYSTERY-ALG algorithm.

PROBLEM 4 *Frogger*

Frogger must cross the street. Frogger can jump either 1 foot straight ahead, diagonal-left (i.e., one one foot ahead and one foot to the left), and diagonal-right. The street is n feet wide and m feet long. Each spot $(x, y) \in [1, n] \times [1, m]$ on the street has an assocated danger level $d(x, y)$—dangers can be potholes, a car (ultimately dangerous), men-at-work, chewing gum, etc. Design an algorithm, that, on input $(n, d(\cdot, \cdot))$, determines the safest path for Frogger.

PROBLEM 5 *Best Trade in Hindsight*

You are given an array $A[1, \ldots, n]$ with the closing stock price of AAPL for the first n days of this year. You would like to find the best day i to buy the stock and the best day to $i < j$ to sell the stock (i.e., you cannot short-sell the stock).

(a) Bob solves this problem by finding the min stock price, and buying on that day, and finding the max stock price and selling on that day. What is wrong with this strategy?

(b) Devise a $\Theta(n \log n)$ divide-and-conquer algorithm to solve this problem.

(c) Devise an $\Theta(n)$ dynamic programming algorithm to solve the problem.

PROBLEM 6 *Euclid*

The greatest common divisor (gcd) of two numbers $a, b \in \mathbb{N}$ is the largest integer x that divides both a and b. Euclid's algorithm below computes this value efficiently:

EUCLID(a, b) where $a \geq b$

```
1  if b = 0 return a
2  d ← EUCLID(b, a mod b)
3  return d
```

We will (loosely) analyze this recursive algorithm in this problem. Let (a_i, b_i) denote the arguments that are passed to the i^{th} invocation of EUCLID where $(a_1, b_1) = (a, b)$ are the initial call. Let $t_i = a_i + b_i$ represent the size of the input on the i^{th} invocation.

(a) (1pt) List the recursive calls made on input EUCLID$(10, 6)$.

(b) (3pt) Prove: for any $a, b \in \mathbb{N}$ where $a \geq b$, $a \bmod b < a/2$.

(c) (2pt) Prove that $t_{i+2} < t_i/2$.

(d) Write a recurrence that upper-bounds $T(t_0)$ (Hint: if the algorithm terminates, what is the value of t?)

(e) Provide an O-expression for the running time of EUCLID in terms of t.

(f) Let F_n be the n^{th} Fibonacci number. Analyze the running time of EUCLID(F_n, F_{n-1}) as precisely as you can.