To Students: This file contains the solutions to the term test together with the marking scheme and comments for each question. Please read the solutions and the marking schemes and comments carefully. Make sure that you understand why the solutions given here are correct, that you understand the mistakes that you made (if any), and that you understand *why* your mistakes were mistakes.

Remember that although you may not agree completely with the marking scheme given here it was followed the same way for all students. We will remark your test only if you can clearly demonstrate that the marking scheme was not followed correctly. We will make *no* exception to the marking scheme, unless you can clearly demonstrate that it is somehow incorrect.

For all remarking requests, please submit your request **in writing** directly to your instructor. For all other questions, please don't hesitate to ask your instructor during office hours or by e-mail.

Summary of Marking Guidelines:

- Form (structure) $\approx 1/2$ marks;
  content (idea) $\approx 1/2$ marks.
- Mostly correct answers deserve at least $\approx 1/2$ marks (no matter how many small mistakes they contain);
  mostly incorrect answers deserve at most $\approx 1/2$ marks (no matter how many correct elements they contain).
- Honesty deserves extra credit (*e.g.*, recognizing incorrect answers, admitting they don't know);
  confusion deserves extra penalties (*e.g.*, adding incorrect justification to mostly correct answer).
- Mark leniently (*i.e.*, give benefit of the doubt),
  but correctly (*i.e.*, don't give undeserved marks).
- Annotate test to make it easy for students to understand their mistakes.

Marking Guidelines: In general, having the right idea should count for approximately half of the marks (even if it isn't expressed perfectly), and having the correct structure/format should count for approximately half of the marks (even if the content is not correct). Try not to be too picky on the details except when they are critical to the solution. Start from the assumption that students know what they are doing, unless they give you evidence to the contrary. In particular, a correct answer followed or preceded by an incorrect justification or explanation deserves fewer marks than the correct answer alone. In contrast, an incorrect answer accompanied by a correct explanation or followed by an acknowledgement that the answer is incorrect deserves more marks than the incorrect answer alone. The important goal is consistency: even if you feel your choice is a little lenient, or a little harsh, stick with it for everyone. As a "sanity check", keep in mind that no matter how the marking scheme is written, no matter how many part marks a student earns or loses, a solution that is mostly correct should get at least roughly 1/2 the marks, and a solution that is mostly incorrect should get at most roughly 1/2 the marks.

Remember that students should get 20% of the marks for a question (don't round) if they write "I don't know" or something similar, as long as they write nothing else (or that they cross off anything else they wrote to make it clear it should not be marked). If a student writes "I don't know" before or after an attempted solution, mark only the attempted solution and ignore the sentence stating that they do not know.

While you mark, please keep track of common student errors and how they were marked, as well as any interpretations of or minor modifications to the marking scheme (including any further breakdown of the marks that you decide to use), so that it is easy to figure out later how a question was marked. Also, please make note of how well each question or each part of a question was answered in general, and of any serious misconceptions or apparent gaps in student's knowledge that you noticed. (These comments will be typeset with the solutions and posted on the course website so that students can find out where and why they lost marks.)

Finally, please remember to give students enough feedback on their copy of the test so that they can easily figure out what they did wrong (if anything) from the solutions, marking schemes, and comments, together with your feedback—keep in mind that a student who misunderstood a question or who didn't know how to answer correctly is unlikely to be able to figure out what they did wrong purely from the sample solutions and marking scheme; it's up to you to provide them with additional feedback to help them understand their mistake(s). In particular, you may find it convenient to use codes (like the letter codes next to each point of the marking scheme) to report common errors on a student's paper. If you do so, please list any additional codes and their meaning in your marking comments.

Question 1. [10 MARKS]

Write a divide-and-conquer algorithm that determines the minimum difference between any two elements of a sorted array of real numbers, in linear time.

For example, on input $A = [-5\frac{2}{3}, -2, 1, \pi, 7.5]$, your algorithm should return 3.

Justify briefly that your algorithm is correct and runs within the required time bound.
(For your reference, the Master Theorem states that a recurrence of the form $T(n) = aT(n/b) + \Theta(n^d)$ has solution $\Theta(n^d)$ if $a < b^d$, $\Theta(n^d \log n)$ if $a = b^d$, and $\Theta(n^{\log_b a})$ if $a > b^d$.)

Note: For full marks, your answer must make use of the divide-and-conquer method.

SAMPLE SOLUTION

The minimum difference is simply the smallest difference between adjacent elements in a sorted list. We can divide the list in two, recursively (using divide-and-conquer), find the smallest difference in each sublist, and find the difference between elements crossing the partition. We'd simply return the minimum of these three values.

```
MinDiff(A):
    if n = 1:        // n = size of A
        return ∞        // can't take min of one element
    else:
        m := ⌊n/2⌋        // need floor, because n might not be a power of 2
        m_1 := MinDiff(A[1...m])
        m_2 := MinDiff(A[(m+1)...n])
        return min{m_1, m_2, A[m+1] − A[m]}
```

Procedure MinDiff runs in time $\Theta(n)$ because its worst-case running time satisfies the recurrence $T(n) = 2T(n/2) + \Theta(1)$.

Why is this algorithm useful? (After all, we could just walk down the list comparing adjacent elements and find the answer in linear time.) Well, our version is parallelizable: if we have a number of computers or processors, we're dividing up the list and can assign small pieces to each machine. We might solve the problem more quickly than linear time if we have enough machines.

MARKING SCHEME:

A. 2 marks: correct idea
B. 5 marks: correct divide-and-conquer algorithm
C. 3 marks: correct runtime, brief justification of correctness and runtime analysis
D. -4 marks: solution that does not use divide-and-conquer, even if completely correct (additional deductions as per marking scheme)

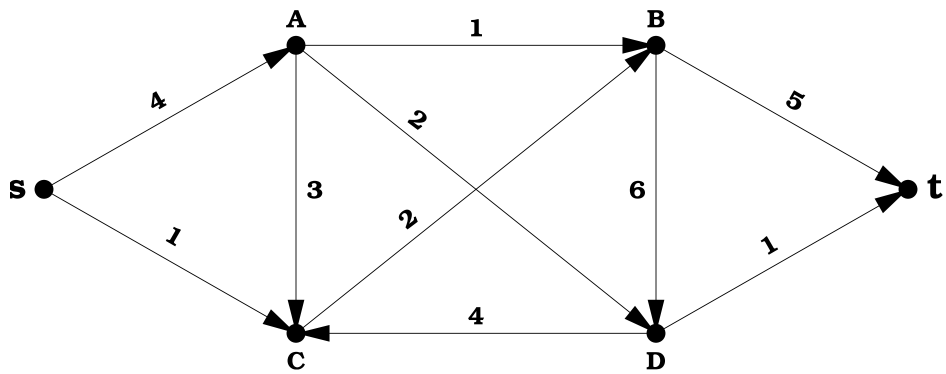
- Use roughly the same marking scheme for other ideas with partial marks for correct divide-and-conquer implementations of reasonable, but incorrect, high-level ideas (the exact number of marks will depend on how close the idea gets to finding a correct answer).

MARKER'S COMMENTS:

- The biggest problem on this question was handling the pair consisting of the largest element in the first half of the array and the smallest element in the second half of the array. Some forgot about this altogether, and others thought of it but were confused about how to handle it. I took off 2 marks for this.
- Another common problem was handling the case $|A| = 1$. Some students returned 0 in this case. I took off 1 mark for this.

Question 2. [5 MARKS]

Use the Ford-Fulkerson method to compute a maximum flow in the network below (where each edge has the indicated capacity). State precisely each augmenting path that you use and its residual capacity; you **must** use the path given below the picture as your first augmenting path. Justify that your flow is maximum; for full marks, your justification should *not* be based on the non-existence of a larger flow or of an augmenting path, but rather on the existence of something else.



Augmenting path: $s \xrightarrow{0/4} A \xrightarrow{0/2} D \xrightarrow{0/4} C \xrightarrow{0/2} B \xrightarrow{0/5} t$ Residual capacity: 2 (*fill this in*)

SAMPLE SOLUTION

Augmenting path: $s \xrightarrow{2/4} A \xrightarrow{0/1} B \xrightarrow{2/5} t$ Residual capacity: 1
Augmenting path: $s \xrightarrow{0/1} C \xleftarrow{2/4} D \xrightarrow{0/1} t$ Residual capacity: 1

The resulting flow (of $2 + 1 + 1 = 4$ units) is maximum because $(\{s, A, C, D\}, \{B, t\})$ is a cut with capacity $c(A, B) + c(C, B) + c(D, t) = 1 + 2 + 1 = 4$.

MARKING SCHEME:

A. 3 marks: correct augmenting paths and maximum flow
B. 2 marks: correct justification through minimum cut

MARKER'S COMMENTS:

- Generally well done.

Question 3. [8 MARKS]

Fires and tunnel collapses are a common hazard in underground mining. Mines establish emergency shelter areas (refuges) where trapped miners can reach emergency supplies and communicate with the surface.

We want to determine whether a given distribution of location of miners in the mine is "safe" or not. Each miner can reach only some nearby refuges, and each refuge can only support a limited number of miners. For each miner we want to pick two nearby refuges (in case the way to one is blocked). The shift is "safe" if we can do this such that no refuge can be overloaded, no matter which of the two options each miner chooses.

You are given a list of n miners and k refuges, with $k < n$. For each miner i, R_i is the set of refuges which the miner could safely reach. Refuge j can support at most C_j miners.

Part (a) [3 MARKS]

Describe precisely how to model this problem as a network flow problem. (Don't forget to specify clearly all edge directions and capacities in your network.)

SAMPLE SOLUTION

Make vertex m_i for each miner i and a vertex r_j for each refuge j. Add an edge (m_i, r_j) with capacity 1 for every $r_j \in R_i$, an edge (s, m_i) with capacity 2 for $1 \leq i \leq n$, and an edge (r_j, t) with capacity C_j for $1 \leq j \leq k$. (all edges directed left-to-right)

MARKING SCHEME:

- 3 marks: clear and correct description, including all edge directions and all capacities

Part (b) [5 MARKS]

Write a polynomial time algorithm based on your network from part (a) to determine whether a safe assignment of miners to emergency refuges exists. Justify briefly that the answer given by your algorithm is correct.

SAMPLE SOLUTION

First we run an efficient integer maximum flow algorithm on our network to compute the maximum value of a flow. If the max flow is $2n$, a safe assignment exists (assign miner i to the two refuges with flow over edge (m_i, r_j)) where no refuge can be overloaded (we can't get more than C_j miners coming to refuge, since at most C_j flow comes in and out).

If the max flow is less than $2n$, we need to show that no safe assignment is possible. First, find the "first" min cut as per the Ford-Fulkerson proof. This cut must include at least one miner vertex on the source side of the cut. Either all of its edges to refuges are saturated (and has less than two possible refuges, proving no solution can exist) or there is at least one refuge vertex on the source side of the cut. Suppose X is the set of miner vertices and Y is the set refuge vertices in the source side of the cut. Then we see that $2|x| > \sum_{y \in Y} C_y$, proving no solution can exist.

MARKING SCHEME:

- 2 marks: clear and correct algorithm, checks for equality to $2n$
- 3 marks: correct justification that problem constraints are satisfied, and that correct answer is found

MARKER'S COMMENTS:

- Many students had trouble justifying the correctness of their algorithm. Many of them did not even attempt to explain why their algorithm is correct when the max flow is less than $2n$.
- Another common mistake was having the algorithm look for a flow of n (rather than $2n$). I took off 1 mark for this.

Question 4. [7 marks]

Let N be a network with positive integer capacities and maxflow value F. Let N' be the network obtained by subtracting 1 from the capacity of every edge in N.

For each of the claims below, decide whether it is true or false. If true, give a brief justification; if false, give a counterexample.

Part (a) [4 marks]

The maxflow value in N' is *at most* $F - 1$.

Sample Solution

True. Consider any min cut in N. Its capacity in N is F and has at least one forward edge. Thus the capacity of this cut in N' is at most $F - 1$, hence by Ford-Fulkerson Theorem, the max flow in N' is at most $F - 1$.

Marking Scheme:

A. 1 mark: for saying true
B. 3 marks: for the justification.

Marker's Comments:

- The most common mistake was trying to justifying this by arguing about edges from s or to t, rather than edges crossing a min cut. I took off 2 marks for this.

Part (b) [3 marks]

The maxflow value in N' is *exactly* $F - 1$

Sample Solution

False. Simple counter example: let N be the network on four vertices $\{s, t, A, B\}$ with edges $E = \{(s, A), (s, B), (A, t), (B, t)\}$ (directed left to right) with all edges with capacity 2. Then $F = 4$ and a minimum cut is $(\{s\}, \{A, B, t\})$. If we subtract 1 from the capacity of every edge, the capacity of the above cut is 2, which is less than $F - 1 = 3$.

Marking Scheme:

A. 1 mark: for saying false
B. 2 marks: for the counterexample.