St. Francis Xavier University
Department of Computer Science

CSCI 355: Algorithm Design and Analysis
Final Examination
April 16, 2024
9:00am–11:30am

Student Name: ______________________________

Email Address: ______________________________

Instructor: T. J. Smith (Section 20)

Format:

The exam is 150 minutes long. The exam consists of 6 questions worth a total of 70 marks. The exam booklet contains 9 pages, including the cover page and one blank page at the back of the exam booklet for rough work.

Reference Materials:

None.

Instructions:

1. Write your name and email address in the spaces above.
2. Answer each question either in the space provided or on a blank page. If you use a blank page to write your answer, indicate this clearly in the space provided for the question. Show all of your work.
3. Ensure that your exam booklet contains 9 pages. Do not detach any pages from your exam booklet.
4. Do not use any unauthorized reference materials or devices during this exam.
5. Sign in the space below. Your signature indicates that you understand and agree to these instructions and the university's examination policies.

Signature: ______________________________

Question	Marks	Score
1	10	
2	10	
3	14	
4	14	
5	10	
6	12	
Total	70	

Multiple Choice

[10 marks] 1. For each of the following questions, select exactly one answer by circling the associated letter. Incorrect answers will not be penalized. Answers with more than one letter circled will be marked as incorrect.

(a) Which of the following statements about stable matchings is **true**?
   - A. A stable matching guarantees that all participants are paired with their most preferred match.
   - B. There exists an instance of the stable matching problem for which it is impossible to find a stable matching.
   - C. There exists an instance of the stable matching problem where every participant is paired with their least preferred match.
   - D. In a stable matching, there is no pair who would both prefer to be matched with each other over their assigned match.

(b) Which of the following statements about Dijkstra's algorithm is **true**?
   - A. Dijkstra's algorithm can handle graphs having negative edge weights.
   - B. Once Dijkstra's algorithm finds the shortest path to a vertex, that path never changes.
   - C. Dijkstra's algorithm finds the shortest paths between all pairs of vertices in a graph.
   - D. The runtime of Dijkstra's algorithm is not dependent on the number of edges in the graph.

(c) Which of the following minimum spanning tree algorithms is most suitable for a dense graph (i.e., a graph having many edges)?
   - A. Prim's algorithm
   - B. Kruskal's algorithm
   - C. Reverse-delete algorithm
   - D. Borůvka's algorithm

(d) Which of the following recurrence relations and time complexity bounds applies to the mergesort algorithm?
   - A. $T(1) = 0$, $T(n) = 2T(n/2) + n$; $T(n) \in O(\log(n))$.
   - B. $T(1) = 0$, $T(n) = T(n/2) + n$; $T(n) \in O(n)$.
   - C. $T(1) = 0$, $T(n) = 2T(n/2) + n$; $T(n) \in O(n \log(n))$.
   - D. $T(1) = 0$, $T(n) = 2T(n/2) + n^2$; $T(n) \in O(n^2)$.

(e) Which of the following statements about the master theorem is **true**?
   - A. The master theorem can only be applied to recurrence relations having constant coefficients and linear work terms.
   - B. The master theorem gives a method of determining the exact number of operations performed by a divide-and-conquer algorithm.
   - C. The master theorem gives a method of analyzing the asymptotic runtime of a divide-and-conquer algorithm.
   - D. The master theorem is primarily used to determine the space complexity of a divide-and-conquer algorithm.

(f) True or false: dynamic programming provides a better method of solving the problem of array sorting than the traditional divide-and-conquer approach.

A. False, because you cannot write a Bellman equation to model this problem.
B. False, because the subproblems do not have the overlapping property.
C. False, because you cannot establish an ordering on the subproblems.
D. All of the above.
E. True, because dynamic programming is always better than divide-and-conquer.

(g) Consider the following strings:

```
I N T E N T I O N
E X E C U T I O N
```

If the gap penalty is 1 and the mismatch penalty is 2, what is the minimum edit distance between these strings?

A. 5
B. 8
C. 10
D. 14

(h) Consider a graph $G = (V, E)$, where $c(e)$ denotes the capacity of an edge e and $f(e)$ denotes the flow across an edge e. Which of the following properties is **not** required of a flow in G?

A. For each $e \in E$, $0 \leq f(e) \leq c(e)$.
B. For each $v \in V \setminus \{s, t\}$, $\sum_{e \text{ into } v} f(e) = \sum_{e \text{ out of } v} f(e)$.
C. For each $e \in E$, $c(e) \geq 0$.
D. For each $v \in V \setminus \{s, t\}$, $\sum_{e \text{ into } v} f(e) > 0$.

(i) If we know that Y is a decision problem that can be solved in polynomial time, and we also know that $X \leq_{\text{P}} Y$ for some decision problem X, then what can we conclude about X?

A. X can be solved in polynomial time.
B. X may or may not be able to be solved in polynomial time.
C. X cannot be solved in polynomial time.
D. X is NP-complete.

(j) Which of the following problems is **not** currently known to belong to the complexity class P?

A. Primality testing problem
B. Planar 4-colourability problem
C. Knapsack problem
D. 2-satisfiability problem

Short Answer

[10 marks] 2. For each of the following questions, give a 1–2 sentence answer.

(a) The master theorem considers divide-and-conquer recurrence relations of the form

$$T(n) = a\,T\left(\frac{n}{b}\right) + f(n).$$

In this expression, what do the terms a, b, and $f(n)$ correspond to?

(b) What is memoization? Is memoization used in top-down or bottom-up dynamic programming?

(c) Every max flow in a flow network has an associated min cut. Is this min cut guaranteed to be unique? That is, for a given max flow, could there be more than one min cut? Either explain why the min cut is unique or give a counterexample.

(d) Suppose you are discussing a decision problem X with Prof. Smith. Prof. Smith claims that we require $\Theta(2^n)$ time for solving X, but you find a reduction from X to another decision problem Y, where Y is solvable in $\Theta(n^6)$ time. If your reduction takes $\Theta(n^4)$ time, what does this say about Prof. Smith's claim and why?

(e) The complexity classes P and NP are both defined in terms of algorithms for decision problems. What is the primary distinction between the algorithm for a decision problem in P and the algorithm for a decision problem in NP?

[14 marks] 3. (a) Suppose you have three candidate algorithms for a problem you're trying to solve:

- Algorithm A takes as input the problem of size n, divides it into four subproblems each of size $n/2$, recursively solves each subproblem, and then combines the solutions in linear time.
- Algorithm B takes as input the problem of size n, recursively solves two subproblems each of size $n-1$, and then combines the solutions in constant time.
- Algorithm C takes as input the problem of size n, divides it into nine subproblems each of size $n/3$, recursively solves each subproblem, and then combines the solutions in cubic time.

What are the time complexities of each of these algorithms, and which algorithm should you choose?

(b) A length-n array A is said to have a majority element if more than $n/2$ of the entries in A are the same. For example, in the array $[2, 1, 4, 2, 2, 5, 2, 3, 2, 2]$, the majority element is 2. The elements of A are not guaranteed to be from some ordered domain, so we cannot make $>$ or $<$ comparisons or sort the array. However, we can test for equality between two elements.

Complete the following divide-and-conquer algorithm that determines whether a majority element exists in an array A. The algorithm returns the element if it exists, and returns "null" otherwise.

```
procedure MAJORITY($A, p, q$)                ▷ initially called as MAJORITY($A, 1, n$)
    if $p = q$ then
        return $\boxed{\phantom{xxxxxxxxxx}}$
    else
        $mid \leftarrow \lfloor (p + q)/2 \rfloor$
        $tmp1 \leftarrow$ MAJORITY($A, p, mid$)
        $tmp2 \leftarrow$ MAJORITY($A, mid + 1, q$)
        if $tmp1 = tmp2$ then
            return $\boxed{\phantom{xxxxxxxxxx}}$
        else if $tmp1 =$ "null" then
            $\boxed{\phantom{xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx}}$
        else                                  ▷ i.e., $tmp1 \neq$ "null" in this else block
            $\boxed{\phantom{xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx}}$
```

[14 marks] 4. The next total solar eclipse visible in Canada will occur on August 22, 2044. Thinking ahead, you have decided to plan a trip to see the event. It will be a long drive, so you are figuring out which hotels to stay at during your trip.

Suppose the drive is n kilometres long, and there are $n-1$ hotels along the highway: exactly one hotel per kilometre excluding the destination. Each day, you can drive at most k kilometres before either stopping at a hotel or reaching the destination. Since hotels are expensive, you want to find the minimum possible amount of money you spend on hotels before reaching the destination.

To formalize this problem:

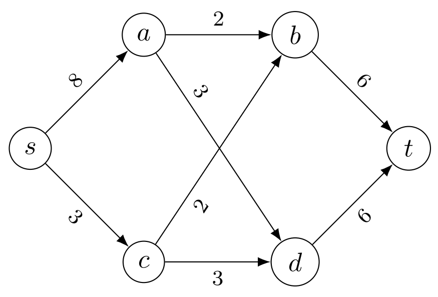
- The inputs are two integers n and k, and an array A where, for all $i < n$, $A[i]$ is the cost of staying one night at the hotel at kilometre i of the highway.
- The output is the least number c such that there exist indices $i_1 < i_2 < \cdots < i_\ell$ (i.e., the hotels you stay at) satisfying the following constraints:
  - $(i_1 - 0)$, $(i_2 - i_1)$, $\ldots$, $(i_\ell - i_{\ell-1})$, and $(n - i_\ell)$ are all $\leq k$ (i.e., you don't drive more than k kilometres per day); and
  - $A[i_1] + A[i_2] + \cdots + A[i_\ell] = c$ (i.e., the total cost of staying at hotels is c).

(a) As a small example, let $n = 10$, $k = 3$, and the costs of the nine hotels along the highway are $A = [3, 1, 4, 5, 2, 3, 7, 8, 1]$. What is the minimum possible cost of hotels in this instance?

(b) For each $i \leq n$, let $C(i)$ denote the cumulative cost of driving i kilometres and staying at the hotel located at kilometre i. What is an appropriate formulation of $C(i)$ for this problem?

(c) Design a dynamic programming algorithm that takes as input integers $n, k \geq 1$ and the array A and outputs the minimum possible cost of hotels c. Your algorithm must run in time polynomial in n and k, but you do not need to analyze your algorithm.

[10 marks] 5. Consider the following flow network.



(a) What is the greatest amount of flow we can route through this network? Draw the flow routing through the network.

(b) Find a cut in this network having a capacity matching the value of the flow you found in part (a). Either draw the cut in the network or give the cut set.

(c) Using what you know from parts (a) and (b), does there exist a cut in this network with a capacity smaller than the one you found? Why or why not?

[12 marks] 6. Recall some of the NP-complete decision problems we defined in class:

A.	Independent set	D.	Knapsack
B.	Vertex cover	E.	3-colour
C.	Set cover	F.	Subset sum

Each of the problem statements below can be formulated as an instance of one of the above decision problems. For each problem statement below, match it to the most appropriate decision problem, and give a brief justification of your decision.

(a) A software developer uses a measurement tool on a program to record which lines of code are covered by each of the test cases in the developer's testing suite. The developer wants to run as few test cases as possible while still testing as many lines of code in the program as possible.

(b) A communications company owns a number of network transmission towers, and each tower covers a certain area of land. If the area covered by a particular tower overlaps with the area covered by another tower, those towers cannot broadcast on the same frequency. Since the government charges the company for each frequency it uses, the company wants to minimize the number of frequencies used by its network.

[2 marks] *Bonus.* What was your favourite part of this course, and why?

This blank page may be used for rough work.