

**NAME:** \_\_\_\_\_

## ALGORITHMS QUALIFYING EXAM

**This exam is open:**

- books
- notes

**and *closed*:**

- neighbors
- calculators

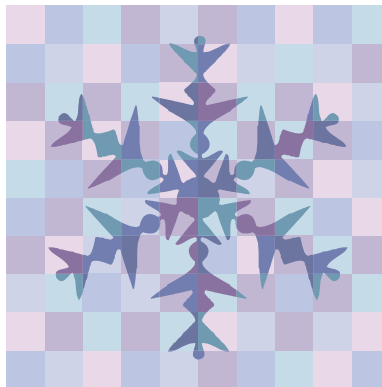
**The *upper bound* on exam time is 3 hours.**

**Please write your name at the top of each page.**

**Please put all your work on the exam paper.**

**(Partial credit will only be given if your work is shown.)**

**Good luck!**



**Part 1**  
(50 points)

**1: (10 points) TRUE/FALSE**

**For each of the 5 statements below:**

- **Circle TRUE if the statement is TRUE.**
- **Circle FALSE if the statement is FALSE**
- **Justify your answer in each case.**

**a) (2 points) If A and B are two algorithms for solving the same problem, then it is possible for a lower bound on the best-case asymptotic running time of A to be larger than an upper bound on the worst-case asymptotic running time of B.**

**TRUE**

**FALSE**

**b) (2 points) If  $P$  is a problem with a worst-case lower bound in  $\Omega(f(n))$  and  $A$  is an algorithm solving  $P$  and  $A$  has a worst-case lower bound in  $\Omega(g(n))$ , then  $f(n)$  must be in  $O(g(n))$ .**

**TRUE**

**FALSE**

**c) (2 points) Given an NP-complete problem  $A$  and a problem  $B$ , a reduction from  $A$  to  $B$  can be used to establish NP-hardness of  $B$  if the transformation from  $A$  to  $B$  has a worst-case asymptotic upper bound on its running time in  $O(2^n)$ , where  $n$  is the input size of both  $A$  and  $B$ .**

**TRUE**

**FALSE**

**d) (2 points) An approximation scheme whose worst-case asymptotic**

**running time is in  $O\left(\left(\frac{1}{\varepsilon}\right)^3 n^{(2+\varepsilon)}\right)$  is a fully polynomial-time approximation scheme.**

**TRUE**

**FALSE**

**e) (2 points) Section 17.4.2 of the Cormen algorithms textbook uses the following potential function to analyze the cost of table expansion and contraction:**

$$\Phi(T) = \begin{cases} 2\text{num}[T] - \text{size}[T] & \text{if } \alpha(T) \geq 1/2 \\ \text{size}[T]/2 - \text{num}[T] & \text{if } \alpha(T) < 1/2 \end{cases}$$

**The potential function  $\Phi'(T)$  below can be used in place of  $\Phi(T)$ .**

$$\Phi'(T) = \begin{cases} 2\text{num}[T] - \text{size}[T] & \text{if } \alpha(T) \geq 1/2 \\ \text{size}[T]/3 - \text{num}[T] & \text{if } \alpha(T) < 1/2 \end{cases}$$

**TRUE**

**FALSE**

**2: (10 points) Pseudocode Analysis**

```
MYSTERY( $A, p, r$ )
   $q \leftarrow \text{RANDOM}(1, \text{length}[A])$ 
  if  $p \geq r$ 
    then return  $q$ 
   $m2 \leftarrow \left\lfloor \frac{(p+r)}{2} \right\rfloor$ 
   $m4 \leftarrow \left\lfloor \frac{(3p+r)}{4} \right\rfloor$ 
  if  $q \bmod 2 = 0$ 
    then return MYSTERY( $A, p, m2$ )
    else return MYSTERY( $A, p, m4$ ) + MYSTERY( $A, m4 + 1, m2$ )
```

Provide a tight upper bound on the worst-case asymptotic running time of the function MYSTERY above. You may assume that a call to RANDOM uses the uniform distribution and that the call requires  $\Theta(1)$  time in the worst case.

*this page left blank for work*

**3: (5 points) String Matching**

**Design a string-matching finite automaton that scans a text string for all occurrences of the text pattern abbbab.**

**4: (5 points) Number Theory**

a) (3 points) List all the elements of the set  $Z_{42}^*$  for the *finite multiplicative group modulo 42*:  $(Z_{42}^*, \cdot_{42})$ .

b) (2 points) Calculate the number of elements of  $Z_{42}^*$  using Euler's phi function.

**5: (10 points) Flow Networks**

**For a flow network with vertex set  $V$ , source vertex  $s$  in  $V$ , and a vertex  $y$  in  $V$  that is connected to  $s$ , consider the following statement about the flow network:**

$$f(y, V) = f(y, s) - \sum_{v \in V - \{s\}} f(v, y)$$

**Either prove the statement or provide a counterexample.**

**6: (10 points) Linear Programming**

**An independent set of a graph  $G = (V, E)$  is a subset  $V' \subseteq V$  of vertices such that each edge in  $E$  is incident on at most one vertex in  $V'$ . The independent-set problem is to find a maximum-sized independent set in  $G$ .**

**An integer program is a linear program in which one or more variables are constrained to have integer values.**

**Formulate an integer program that solves the independent-set problem.**

***Hint: Use variables that can only have the value 0 or 1.***

**Can your integer program be solved in polynomial time? Justify your answer.**

*this page left blank for work*

**Part 2**

(25 points)

The Euclidean Traveling-Salesman problem is the problem of determining the shortest closed tour that connects a given set of  $n$  points  $\langle p_1, \dots, p_n \rangle$  in the plane. The general problem is NP-complete. However, the bitonic version of the problem can be solved in  $O(n^2)$  time using dynamic programming. In the bitonic version of the problem, a tour that starts at the leftmost point goes strictly left to right to the rightmost point and then goes strictly right to left back to the leftmost point. In other words, the tour consists of two pieces. The endpoints of each piece are the leftmost point and the rightmost point. Each piece is monotonic with respect to the  $x$ -axis. That is, if we project the points down onto the  $x$ -axis the points retain their order.

An  $O(n^2)$  time dynamic programming algorithm that constructs an optimal bitonic tour scans left to right, maintaining optimal possibilities for the two parts of the tour (left to right part and right to left part). The algorithm first sorts the points by non-decreasing  $x$  coordinate. (Assume that no two points have the same  $x$  coordinate.) The algorithm is based on a characterization of the optimal substructure of the problem. Describe the optimal substructure by showing that, at a given point  $p_i$  in the left to right scan, the optimal bitonic tour involving  $\langle p_1, \dots, p_i \rangle$  can be determined by examining no more than  $(i-1)^2$  subproblems and then calculating the distance from  $p_i$  to each of  $\langle p_1, \dots, p_{i-1} \rangle$ .

Hint: The algorithm must assign each point to one or both parts of the tour. Consider the different possibilities for a sequence of assignments for the left to right part and the right to left part. When considering  $p_i$ , focus on the last point added to each of the two parts of the bitonic tour.

*this page left blank for work*

**Part 3**  
(25 points)

Design an algorithm that determines whether or not an undirected graph  $G = (V, E)$  is 2-colorable. If  $G$  is 2-colorable, the algorithm should 2-color it.

(a) (8 points) PseudoCode:

(b) (8 points) Correctness:

(c) (8 points) Upper Bound on Asymptotic Worst-Case Running Time:

(d) (1 point) Can your approach be generalized to efficiently determine whether or not a graph is 3-colorable? Why or why not?