First Person:  
Family Name:  
Given Name:  
Student #:  
Email:  

Second Person:  
Family Name:  
Given Name:  
Student #:  
Email:  

Guidelines:

• You may complete this assignment alone or in groups of two. Do not get solutions from other pairs. Though you are to teach & learn from your partner, you are responsible to do and learn the work yourself. Write it up together.

• Please make your answers clear and succinct.

• Relevant Readings:
  – CLRS Ch. 1, 2, 3, 8, 12, A
  – Edmonds Ch. 1-7, 22-26

• This page should be the cover of your assignment.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (8 marks)</td>
<td></td>
</tr>
<tr>
<td>2 (15 marks)</td>
<td></td>
</tr>
<tr>
<td>3 (16 marks)</td>
<td></td>
</tr>
<tr>
<td>4 (25 marks)</td>
<td></td>
</tr>
<tr>
<td>5 (20 marks)</td>
<td></td>
</tr>
<tr>
<td>6 (16 marks)</td>
<td></td>
</tr>
<tr>
<td>Total (100 marks)</td>
<td></td>
</tr>
</tbody>
</table>
1. **Universal and Existential Qualifiers** (8 marks)

Recall that a problem is *computable* if and only if there is an algorithm that halts and returns the correct solution on every valid input. Let \( \text{Works}(P, A, I) \) be true if and only if algorithm \( A \) halts and correctly solves problem \( P \) on input instance \( I \). Let \( P = \text{Halting} \) be the Halting problem which takes a Java program \( I \) as input and tells you whether or not it halts on the empty string. Let \( P = \text{Sorting} \) be the sorting problem which takes a list of numbers \( I \) as input and sorts them.

(a) (4 marks) Express using universal and existential qualifiers (\( \forall, \exists \)) that \( \text{Sorting} \) is computable. No justification is required.

(b) (4 marks) Express using universal and existential qualifiers (\( \forall, \exists \)) that \( \text{Halting} \) is not computable. No justification is required.

2. **Bounds** (15 marks)

We use the logical connective \( \rightarrow \) to represent a logical implication, consequence or entailment. For example, the proposition Statement 1 \( \rightarrow \) Statement 2 means that if Statement 1 is true, Statement 2 must also be true.

If \( f(n) \) and \( g(n) \) are two functions of the positive integer \( n \), which of the following propositions are true? Please prove the ones you believe to be true, and provide a counterexample for those you think false.

(a) \( f(n) \in \Theta(g(n)) \rightarrow g(n) \in \omega(f(n)) \)

(b) \( f(n) \in \Omega(g(n)) \rightarrow g(n) \in \mathcal{O}(f(n)) \)

(c) \( f(n) \in \Omega(g(n)) \rightarrow g(n) \in o(f(n)) \)

(d) \( f(n) \in \mathcal{O}(g(n)) \rightarrow g(n) \in \omega(f(n)) \)

(e) \( f(n) \in \Theta(g(n)) \rightarrow f(n) \in \mathcal{O}(g(n)) \)

3. **Sums** (16 marks) Derive a \( \Theta \)-approximation for the following sums. Please show your work, but be concise.

(a) \( \sum_{i=1}^{n} i \cdot 2^i \)

(b) \( \sum_{k=5}^{n} \frac{1}{3k+7} \)

(c) \( \sum_{i=2}^{n} \sum_{j=1}^{n} \frac{\log_i}{j} \)

(d) \( \sum_{k=1}^{n} k^k \)

4. **The Carnival Coin Game** (25 marks)

You are running a game booth at your local village carnival. In your game you lay out an array of $1 coins on a table. For a $5 charge, anyone can challenge you to a game. In this game, you and your customer alternately pick coins from the table, either 1 or 2 at a time. If your customer can make you pick up the last coin, he walks away with all the coins. You graciously allow your customer to go first.

Write an iterative algorithm \( \text{CarniCoin}(n) \) that tells you how many coins to pick up on every turn, depending upon what the customer does, so that you are guaranteed to win. The algorithm takes one parameter \( n \), the number of coins on the table at the beginning of the game. The algorithm should halt with exactly one person’s turn remaining.

(You may simulate the customer with a function \( k=\text{customer}(m) \) which takes as input the number of coins \( m \) currently on the table and returns the number of coins \( k \in 1, 2 \) that the customer picks up.)

(a) (5 marks) What is the postcondition for your algorithm?

(b) (5 marks) What precondition must you place on \( n \) in order to guarantee that you will be able to meet the postcondition?
5. **Tri-Sort** (20 marks)

**Input:** An array $A[1..n]$ of employee records. $A[i].id$ and $A[i].rank$, respectively indicate the id and the rank of the corresponding employee. The id field contains the employee name and additional relevant information about that employee. The rank field is 0, 1, or 2.

**Output:** Array $A$ sorted by rank.

Design and analyze an iterative incremental and “in-place” algorithm to solve this problem in $O(n)$ worst-case time. “In-place” means that in addition to array $A$, the algorithm uses no more than $O(1)$ additional variables, each of which has size no more than a single array element. (So you are not allowed to use additional arrays, lists, etc., with more than a constant number of cells.) Describe your algorithm using loop invariants. Prove that it is $O(n)$.

6. **Longest Interval in Union** (16 marks) Given a list of intervals $[a_1, b_1], [a_2, b_2], \ldots, [a_n, b_n]$, where $a_1, \ldots, a_n, b_1, \ldots, b_n$ are integers, we are interested in the length of the longest interval in the union of the given intervals. For example, if the given intervals are $[0, 3], [2, 6], [7, 14]$ and $[8, 10]$, then their union is $[0, 6] \cup [7, 14]$ and the length of the longest interval in the union is $14 - 7 = 7$.

   (a) (12 marks) Design an algorithm to solve this problem.
   (b) (4 marks) Analyze its running time.

Your algorithm must be as efficient as possible. Algorithms with running time $\Theta(n^2)$ and above will receive only partial marks.