CP312 Algorithm Design and Analysis I **Winter 2024**

Assignment 1

Instructor: Dariush Ebrahimi *Due Date: 04-Feb-2024*

Instructions: You must submit your solutions as a single PDF file to MyLS. Make your solutions as detailed as possible by clearly stating every step in your answers. The assignment must be done individually. Any **COPYING** of solutions from external sources will result in a **ZERO** grade.

Problems

1. **Algorithm Analysis.** In this question, you will need to analyze three algorithms and prove the required correctness and/or running time properties.

(a) You are given the following algorithm which accepts as input a list of positive integers $A = (a_1, ..., a_n)$ and we need to determine whether all elements are *unique* (that is, there are no repeated elements in the list). If all elements are unique, the algorithm outputs True, otherwise it will output False.

```
procedure CheckUnique(A, n)
    for i = 1 to n − 1 do
        for j = i + 1 to n do
            if a_i == a_j then
                return FALSE
            end if
        end for
    end for
    return TRUE
end procedure
```

(i) (3 points) State the primitive operations and the data structure(s) to be used in this algorithm.

(ii) (2 points) Compute the **worst-case** running time $T(n)$ to output the final result using Θ-notation.

(b) You are given the following algorithm which accepts as input an $n \times n$ square matrix *A* and we need to compute $C = A^2 = AxA$ where $A[i, j]$ is the element in row *i* and column *j*.

```
procedure MatrixSquared(A, n)
    for i = 1 to n do
        for j = 1 to n do
            for k = 1 to n do
                C[i, j] = C[i, j] + (A[i, k] * A[k, j])
            end for
        end for
    end for
    return TRUE
end procedure
```

(i) (3 points) State the primitive operations and the data structure(s) to be used in this algorithm.

(ii) (4 points) Compute the **worst-case** running time $T(n)$ to output the final result. Write your answer **exactly**, **without** using asymptotic notation.

(iii) (2 points) Write your answer for part(ii) using Θ-notation.

(c) Consider the following algorithm for computing the factorial of some integer $n \geq 2$.

```
1: procedure Factorial(n)
2:     i = 1
3:     j = 1
4:     while j < n do
5:         j = j + 1
6:         i = i * j
7:     end while
8:     return i
9: end procedure
```

(i) (8 points) To prove the correctness of this algorithm, the following loop invariant for the **while** loop in lines 4-7 can be defined:

Loop Invariant: At the start of iteration *j*, it must be that *i* = *j*!

Prove correctness by showing that this loop invariant is true using the initialization, maintenance, and termination properties.

(ii) (2 points) Compute the worst-case running time.

2. (20 points) **Asymptotic Notation Exercises.** Indicate whether each of the following statements is True or False then justify your answers; if true, **show the necessary constants** (e.g., c, n_0, etc.) that satisfy the notation, and if false, argue that no such constants exist.

(a) $3n^2 + n + 10 = \Theta(n^2)$

(b) $\sqrt{n} = \Omega(n^2)$

(c) $\dfrac{\lg n}{10} = \Omega(1)$

(d) $\lg n^8 = O(\lg n)$

(e) $3^n = O(2^n)$

(f) $100n^3 = o(n^3 \lg n)$

(g) $5n^{10} = \omega(n^{10})$

(i) $\log^2(n) = O(\log(n^2))$ (Note: $\log^2(n) = \log(n) \times \log(n)$)

(h) $3\log_7 n = \Theta(\log_2 n)$

(j) $\dfrac{1}{n^2} = O(1)$

3. **Asymptotic Notation Properties.** Let f(n) and g(n) be asymptotically non-negative functions. Using the definitions of asymptotic notations:

(a) (4 points) Prove that if $f_1(n) = O(g_1(n))$ and $f_2(n) = O(g_2(n))$ then $f_1(n) + f_2(n) = O(g_1(n) + g_2(n))$.

(b) (4 points) Prove that if $f(n) = O(n) + O(n^2) + O(n^3)$ then $f(n) = O(n^3)$.

4. **Solving Recurrences (I).** Consider the following recurrence:

$$T(n) = T(n - 1) + 10$$

   (a) Assume that the base case is $T(1) = \Theta(1)$. Show that the solution to this recurrence is $O(n)$ using:
      (i) (5 points) The substitution method (the starting guess is $T(n) = O(n)$)
      (ii) (5 points) The recursion tree method

   (b) (2 points) Is $T(n) = \Omega(n)$? Explain your answer.

   (c) (3 points) Solve the recurrence using the Master Theorem, if possible.

5. **Solving Recurrences (II).** Consider the following recurrence:

$$T(n) = 2T(n/4) + T(n/8) + n$$

   Assume that the base case is $T(1) = \Theta(1)$. Using the **recurrence tree method**:

   (a) (6 points) Find an upper bound (i.e. $O(.)$) on the solution to this recurrence.

   (b) (3 points) State the lower bound (i.e. $\Omega(.)$) for T (n) and justify your answer.

6. (24 points) **Recurrences (Master Method).** Solve each recurrence below using the Master Method and write your final answer using Θ-notation. Make sure that you show all your work including the corresponding case and the values of ϵ or k used. If it is not possible to solve a recurrence using the Master Method, prove it by showing that the form is inapplicable or by showing that all 3 cases cannot be satisfied.

   (a) $T(n) = 64T(n/8) + 3n$
   (b) $T(n) = 8T(n/2) + n^3$
   (c) $T(n) = T(2n) + n^2$
   (d) $T(n) = T(3n/10) + n$
   (e) $T(n) = 2T(n/2) + \sqrt{n}$
   (f) $T(n) = T(n/7) + \lg^3 n$