Homework #4
Due Date:
Reading Assignment: Chapter 4
Problems:

1. Prove that
\[ \sum_{k \mid m} \frac{1}{k} \sum_{d \mid k} \mu\left(\frac{k}{d}\right)c^d = \frac{1}{m} \sum_{d \mid m} \varphi(d)c^{\frac{m}{d}}. \]

2. 4–29
3. 4–38
4. 4–41
5. The object of this problem is to determine the average execution time of the Insertion Sort algorithm. Suppose we want to sort \( n \) integers stored in an array \( X[0..n] \). We assume that \( X[0] \) contains garbage and that the \( n \) integers are stored as \( X[1], X[2], \ldots, X[n] \).

Insertion sort is the method commonly used by people to sort bridge hands: consider the integers one at a time and insert each in its proper place among those already considered. The following program does exactly that.

\[
\begin{align*}
\text{#times executed} & \quad n \\
\text{for } i := 2 \text{ to } n \text{ do } & \begin{align*}
& n - 1 \quad \text{value} := X[i]; \\
& n - 1 \quad \text{place} := i - 1; \\
& A_n + n - 1 \quad \text{while } (\text{place} \geq 1 \text{ and } X[\text{place}] \geq \text{value}) \text{ do begin } \\
& A_n \quad X[\text{place} + 1] := X[\text{place}]; \\
& A_n \quad \text{place} := \text{place} - 1; \\
& \text{end; } \\
& n - 1 \quad X[\text{place} + 1] := \text{value}; \\
& \text{end;} 
\end{align*}
\]

Then number of times each statement is executed is given in the left column. Assume each statement takes one time unit to execute, except
for the while statement. It takes two time units to execute, because there are two logical expressions to evaluate. The total running time of this algorithm is just

$$4A_n + 6n - 5.$$  

To find the average running time, it suffices to find the average value of $A_n$, the number of times the inner loop is executed.

Assume that the $n$ integers are $\{1, 2, \ldots, n\}$ and that all $n!$ permutations of the $n$ integers are equally likely. The average (or expected) value of $A_n$ is $E(A_n) = B_n/n!$, where

$$B_n = \sum_{\text{each permutation } X[1], X[2], \ldots, X[n]} \left( \text{value of } A_n \text{ when the program is run on input } X[1], \ldots, X[n] \right).$$

For example, when $n = 3$, assume that we want to sort the integers $\{1, 2, 3\}$. For the $3!$ possible permutations, we get

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>

Therefore, $B_3 = 0 + 1 + 1 + 2 + 2 + 3 = 9$, so $E(A_3) = 9/3! = 3/2$.

(a) Find a simple expression for $E(A_n)$ in terms of $n$. Do that by deriving a recurrence relation for $B_n$ in terms of $B_{n-1}$, then solving it by using the appropriate “summation factor.” Compute the average running time $4(E(A_n)) + 6n - 5$.

(b) One way to speed up the algorithm is to get rid of the “place $\geq 1$” expression, since it is seldom needed. To do that, we must initialize $X[0]$ to $-\infty$. Now the while statement takes only one time unit to execute. Does this alteration affect the value of $A_n$? Why or why not? What is the resulting average running time of the algorithm?
(c) Is this algorithm stable? If not, can you modify the algorithm to make it stable? A sorting method is stable if equal keys remain in the same relative order in the sorted sequence as they were in originally.