Submit your typed answers to the questions below using the department's Gitlab server by October 30, 2015 @ 11:59pm. Put your PDF into a folder named `theory_assignment4`.

1. The following code fragment implements Horner's Method for evaluating a polynomial

$$\begin{aligned} P(X) &= \sum_{k=0}^{n} a_k x^k \\ &= a_0 + x(a_1 + x(a_2 + \cdots + x(a_{n-1} + xa_n)\cdots)). \end{aligned}$$

given the coefficients $a_0, a_1, \ldots, a_n$ and a value for x:

$y \leftarrow 0$
for $i = n$ downto 0 **do**
  $y = a_i + x \cdot y$
end for

   (a) In terms of Θ–notation, what is the running time of this code fragment for Horner's Method?

   (b) Write pseudocode to implement the naive polynomial–evaluation algorithm that computes each term of the polynomial from scratch. What is the running time of this algorithm? How does it compare to Horner's Method?

   (c) Consider the following loop invariant:
   At the start of each iteration of the for loop,

$$y = \sum_{k=0}^{n-(i+1)} a_{k+i+1} x^k.$$

   Interpret a summation with no terms as equaling 0. Use this loop invariant to show that, at terminationn, $y = \sum_{k=0}^{n} a_k x^k$.

   (d) Conclude by arguing that the given code fragment correctly evaluates a polynomial characterized by the coefficients $a_0, a_1, \ldots, a_n$.

2. Let $A[1..n]$ be an array of n distinct numbers. If $i < j$ and $A[i] > A[j]$, then the pair (i, j) is called an *inversion* of A.

   (a) List the five inversions of the array $\langle 2, 3, 8, 6, 1 \rangle$.

   (b) What array with elements from the set $\{1, 2, \ldots, n\}$ has the most inversions? How many does it have?

   (c) What is the relationship between the running time of insertion sort and the number of inversions in the input array? Justify your answer.

   (d) Give an algorithm that determines the umber of inversion in any permutation on n elements in $\Theta(n \lg n)$ worst–case time. Hint: Modify merge sort.

3. Prove that $o(g(n)) \bigcap \omega(g(n))$ is the empty set.

4. Let $f(n)$ and $g(n)$ be asymptotically nonnegative functions. Using the basic definition of Θ–notation, prove that $\max(f(n), g(n)) = \Theta(f(n) + g(n))$.

5. Rank the following functions by order of growth; that is, find an arrangement $g_1, g_2, \ldots, g_{30}$ of the functions satisfying $g_1 = \Omega(g_2)$, $g_2 = \Omega(g_3)$, $\ldots$, $g_{23} = \Omega(g_{24})$. Partition your list into equivalence classes such that functions $f(n)$ and $g(n)$ are in the same class if and only if $f(n) = \Theta(g(n))$.

$(\sqrt{2})^{\lg n}$	n^2	$n!$	$(\lg n)!$	$\left(\frac{2}{3}\right)^n$	n^3
$\lg^2 n$	$\lg(n!)$	2^{2^n}	$n^{1/\lg n}$	$\ln \ln n$	$n \cdot 2^n$
$n^{\lg \lg n}$	$\ln n$	$n \lg n$	$2^{\lg n}$	$(\lg n)^{\lg n}$	e^n
$4^{\lg n}$	$(n+1)!$	$\sqrt{\lg n}$	$2^{\sqrt{2 \lg n}}$	n	2^n