CSCI 200

HW #1: Insertion sort, analyzing algorithms, growth of functions

Directions: *Complete your work on a separate sheet of paper. Submit the physical copy of your work at the beginning of class on the specified due date. Show your work. You may work in groups of up to 3 students provided that all students participate in each question. Unless otherwise stated, assume that logarithmic functions are using base 2.*

1. For each function $f(n)$ and time t in the following table, determine the largest size n of a problem that can be solved in time at most t, assuming that the algorithm to solve the problem takes $f(n)$ **seconds**.

	1 minute	1 hour	1 day	1 year
$\log_2(n)$				
$\sqrt{n}$				
n				
n^2				
$n!$				

2. Using Figure 2.2 as a model, illustrate the operation of `INSERTION-SORT` on the array $A = [10, 42, 30, 28, 53, 42]$.

3. Rewrite the `INSERTION-SORT` procedure to sort into nonincreasing instead of nondecreasing order.

4. Consider sorting n numbers stored in an array A, indexed 1 to n, by first finding the smallest element of A and exchanging it with the element in $A[1]$. Then, find the second smallest element of A, and exchange it with $A[2]$. Continue in this manner for the first $n - 1$ elements of A (i.e., up to index $n - 1$).
   (a) Write pseudocode for this algorithm, which is known as `SELECTION-SORT`
   (b) What loop invariant does this algorithm maintain? Why does it need to run for only the first $n - 1$ elements, rather than all n elements?
   (c) What is the best-case and worst-case run time in Θ-notation?

```
function Foo(a, n)
    k ← 0
    b ← 1
    while k < n do
        k ← k + 1
        b ← b * a
    return b
```

```
function Bar(a, n)
    k ← n
    b ← 1
    c ← a
    while k > 0 do
        if k mod 2 = 0 then
            k ← k/2
            c ← c * c
        else
            k ← k − 1
            b ← b * c
    return b
```

```
function Baz(n)
    k ← 1
    m ← 1
    while k ≤ 6 do
        m ← m * k
        k ← k + 1
    return m
```

```
function Moo(n)
    k ← 1
    m ← 1
    x ← 1
    while k ≤ n do
        while m ≤ n do
            x ← m * k
            m ← m + 1
        k ← k + 1
    return x
```

5. (a) Consider the algorithm `Foo`, which takes as input integers a and n. Describe what the algorithm `Foo` does. *Hint: 'It computes ______.' Fill in the blank with a mathematical expression.*
   (b) Analyze the worst-case running time of `Foo` and express it in O-notation. Justify.

6. (a) Consider the algorithm `Bar`, which takes as input integers a and n. Describe what the algorithm `Bar` does. *Hint: 'It computes ______.' Fill in the blank with a mathematical expression.*
   (b) Analyze the worst-case running time of `Bar` and express it in O-notation. Justify.

7. (a) Consider the algorithm `Baz`, which takes as input integer n. Describe what the algorithm `Baz` does. *Hint: 'It computes ______.' Fill in the blank with a mathematical expression.*

   (b) Analyze the worst-case running time of `Baz` and express it in O-notation. Justify.

8. (a) Consider the algorithm `Moo`, which takes as input integer n. Describe what the algorithm `Moo` does. *Hint: 'It computes ______.' Fill in the blank with a mathematical expression.*

   (b) Analyze the worst-case running time of `Moo` and express it in O-notation. Justify.

9. Express each of the following functions using O-notation. That is, find the slowest growing function $g(n)$ such that $f(n) \in O(g(n))$.

   (a) $f(n) = 10n + 32$

   (b) $f(n) = 5\log(n) + 42$

   (c) $f(n) = 18 + n^5 + 2^n$

   (d) $f(n) = 4 - 4n + 4n^2 + 4n^3$

   (e) $f(n) = 5\log(n) + 42n$

10. Rank the following functions by order of growth, from slowest-growing to fastest-growing. That is, find an arrangement $f_1, f_2, f_3, ..., f_{10}$ of the following functions satisfying $f_1 \in O(f_2)$, $f_2 \in O(f_3)$, etc.

$2^{\log_2(n)}$	n^{100}	100	$n^{0.001}$	$log_2(n)$
$\log_2^3(n)$	$n\log_2(n)$	2^n	n^2	$\sqrt{n}$