Homework 1 Solution

1.
   - 10, 2^{10}
   - $4n$, $2^{\log n}$, $3n + 100 \log n$
   - $n \log n$, $4n \log n + 2n$
   - $n^2 + 10n$
   - n^3
   - 2^n

2.
```
InsertionSortRev(A):
    for j = 2 to len(A):
        key <- A[j]
        i <- j - 1
        while i > 0 and A[i] < key:
            A[i + 1] <- A[i]
            i <- i - 1
        A[i + 1] <- key
```

3. $5n^2 + n \log n + 7 \leq 5n^2 + n^2 + 7n^2 = 13n^2$ when $n \geq 1$. Thus $c = 13$ and $n_0 = 1$. We can verify this because $5n^2 + n \log n + 7 - 13n^2 < 0$ when $n = 1$ and the derivative $-16n + log(n) + 1 < 0$ for every $n \geq 1$.

4. If $f = O(g(n))$ then $f(n) \leq c \cdot g(n)$ when $n \geq n_0$ for some constants c and n_0. This also tells us that $\frac{1}{c} f(n) \leq g(n)$ (or $g(n) \geq \frac{1}{c} g(n)$) when $n \geq n_0$. Since $\frac{1}{c}$ is just another constant, $g = \Omega(f(n))$.

5. We know that $f(n) \leq \max(f(n), g(n))$ and that $g(n) \leq \max(f(n), g(n))$. Thus $f(n) + g(n) \leq 2 \max(f(n), g(n))$. This establishes that $f(n) + g(n) = O(\max(f(n), g(n)))$.

   We also need to show the reverse. For positive functions, $f(n) = O(f(n) + g(n))$ and $g(n) = O(f(n) + g(n))$. Thus $\max(f(n), g(n)) = O(f(n) + g(n))$. Together, these establish that $O(\max(f(n), g(n)) = O(f(n) + g(n)))$.

6. The loop header and conditional will each run at most n times and one or the other return statement will execute once. This makes the total running time $O(n)$.

```
Contains(A, k):
    for i = 1 to len(A): #n+1 times
        if A[i] = k: #n times
            return true #1 time
    return false #1 time
```

7. This has a runtime of $O(\log n)$ as each iteration halves the number of items for consideration. Thus k iterations can differentiate 2^k items. If $n = 2^k$ then $k \approx \log n$.

```
Seek(A, k):
    l <- 1
    r <- len(A)
    while l < r:
        m <- (l + r) / 2
        if A[m] < k:
            r = m
        elif A[m] > k:
            l = m + 1
        else:
            return true
    return false
```