1 Exercise 3

1.1

Algorithm 1 Recursive insertion sort

```
procedure INSERTION_SORT_NEW(A, n)                ▷ A[1...n]
    for j = 2 to A.length do
        key ← A[j]
        i ← j − 1
        index ← binary_search(A, 1, i, key)
        while i ≥ index do
            A[i + 1] ← A[i]
            i ← i − 1
        end while
        A[i + 1] ← key
    end for
    return A
end procedure
```

1.2

이진 탐색을 사용하면, $O(n \log n)$ 복잡도를 갖는다. j번째 iteration에서 이진 탐색 procedure는 $log_2(j-1)$ 만큼의 비교를 진행하기 때문에,

$$\begin{aligned}\sum_{j=2}^{n} \log_2(j-1) &= \sum_{j=1}^{n-1} \log_2(j) \\ &= n\log_2(n+1) + 2^{\log_2(n+1)+1} + 2 \\ &= O(n \log n)\end{aligned} \tag{1}$$

으로 정리할 수 있다.

1.3

$T(n) = c_1 n + c_2(n-1) + c_3(n-1) + c_4 \log(n-1) + c_5 \sum_{j=2}^{n} t_j + c_6 \sum_{j=2}^{n}(t_j - 1) + c_7 \sum_{j=2}^{n}(t_j - 1) + c_8(n-1)$

에서 최악의 상황일 때 $t_j = j$가 된다. 따라서 $\sum_{j=2}^{n}(j-1)n(n-1)/2$ 이므로 수행시간은 $an^2 + bn + c + dlogn$로 나타낼 수 있다. 따라서 최악의 상황에서 수행시간은 $O(n^2)$