1. Insertion Sort: Worst Case

(a) In the list below, 4 passes of the insertion sort algorithm have been completed, and the double bar separates the sorted part of the list from the unsorted part. The item at index $i$ is missing. Fill in the missing item with a value that will cause $\text{insert}(L, i)$ to perform the most number of steps. (As a reminder, this is called the worst case.)

\[
\begin{array}{cccccccc}
\text{sorted} & \text{unsrt} & \text{anything} < 3 \\
L & 3 & 4 & 6 & 6 & 2 & 3 & 1 & 5
\end{array}
\]

(b) When $\text{insert}(L, i)$ is executed on the example list, how many times does the while loop iterate?

(c) When $\text{insert}(L, i)$ is called on the example list, how many assignment statements are executed?

\[2 \times 4 + 2\]

(d) In general, in the worst case, on pass $i$ of insertion sort, how many times does the while loop iterate? (Your answer should be a formula that involves $i$.)

(e) In general, in the worst case, on pass $i$ of insertion sort, how many assignment statements are executed? (Again, your answer should be a formula that involves $i$.)

\[2i + 2\]

(f) In terms of $i$, in the worst case, does function $\text{insert}$ have constant running time, linear running time, quadratic running time, or some other running time?

(a) constant  (b) linear  (c) quadratic  (d) something else

(g) In function $\text{insertion_sort}$, the first time that function $\text{insert}$ is called, $i$ is 0; the second time, $i$ is 1; and so on. What value does $i$ have the last time that function $\text{insert}$ is called?

\[n - 1\]

(h) For the call $\text{insertion_sort}(L)$, in the worst case, write a formula expressing how many comparisons are made during all the calls to $\text{insert}$.

\[
\frac{(1 + 2(n-2) + 1) + (3 + 2(n-4) + 1) + \ldots}{n/2} + 2n + 2n + \ldots + 2n = 2n \cdot n = n^2
\]

(i) In the worst case, does $\text{insertion_sort}$ have constant running time, linear running time, quadratic running time, or some other running time?

(a) constant  (b) linear  (c) quadratic  (d) something else
2. **Insertion Sort: Best Case**

(a) In the list below, 4 passes of the insertion sort algorithm have been completed, and the double bar separates the sorted part of the list from the unsorted part. The item at index $i$ is missing. Fill in the missing item with a value that will cause $\text{insert}(L, i)$ to perform the fewest number of steps. (That’s called the best case).

```
L: 1 3 3 4 9 8 6 5
```

(b) When $\text{insert}(L, i)$ is executed on the example list, how many times does the while loop iterate?

\[ \boxed{0} \]

(c) When $\text{insert}(L, i)$ is called on the example list, how many assignment statements are executed?

\[ \boxed{2} \]

(d) In general, in the best case, on pass $i$ of insertion sort, how many times does the while loop iterate?

(e) In general, in the best case, on pass $i$ of insertion sort, how many assignment statements are executed?

(f) In the best case, does $\text{insert}$ have constant running time, linear running time, quadratic running time, or some other running time?

(a) constant  (b) linear  (c) quadratic  (d) something else

(g) For the best case, write a formula expressing how many comparisons are made during all the calls to $\text{insert}$.

(h) In the best case, does $\text{insertion_sort}$ have constant running time, linear running time, quadratic running time, or some other running time?

(a) constant  (b) linear  (c) quadratic  (d) something else