1. Consider the following algorithm.

```
0   def long_sum(lst: list[int], t: int) -> int:
1       """ Return the maximum length of any slice of lst whose sum is
2           at most t. Preconditions: t > 0; lst is non-empty.
3       """
4       m = 0  # max length found so far
5       for i in range(len(lst)):                # Loop 1
6           j = i
7           s = 0  # sum of lst[j+1:i+1]
8           while j ≥ 0 and s + lst[j] ≤ t:  # Loop 2
9               s = s + lst[j]
10              j = j - 1
11          if i - j > m:
12              m = i - j
13      return m
```

Find, with proof, an input sequence (aka "input family") $(\texttt{lst}_1, t_1), (\texttt{lst}_2, t_2), (\texttt{lst}_3, t_3), \ldots$ with $\text{len}(\texttt{lst}_n) = n$, such that $RT(\texttt{lst}_n, t_n) \in \Theta(n \log n)$. Show your work.

CAREFUL! The pair $(\texttt{lst}_n, t_n)$ makes up **one** input to LONG_SUM. Also, remember that "$RT(\texttt{lst}_n, t_n)$" refers to