You are given a linked list, and you need to compute the **rank** of each element of the list, i.e. the distance of that element from the end of the list.

Give a high-level idea of how would you solve this problem in parallel using pointer skipping. You can assume that the list is stored in a contiguous array, with a pointer to the next element in the list being a simple index of that element. For example, the following array:

```
A 0 1 B 0 3 F 0 C 0 5 E 0 3 D 0 4
```

Represents the following list:

```
A 0 ----> B 0 ----> C 0 ----> D 0 ----> E 0 ----> F 0
```

What is the total WORK that your solution would perform (integer addition counts as WORK(1), everything else is ignored)?
Worksheet: Pointer Skipping

Assume $d[i] = 1$ for all nodes

**Algorithm:**

1. Repeat $\log N$ times:
   1. Create an async task for each list node
   2. In each task $i$:
      1. set $d[i] += d[\text{succ}[i]]$
      2. set $\text{succ}[i] = \text{succ}[\text{succ}[i]]$

What is the big-O for total WORK and CPL that your solution would perform (integer addition counts as WORK(1), everything else is ignored)?

\[
\text{WORK} = O(N \log N), \quad \text{CPL} = O(\log N)
\]