PROBLEM 2 *Crossing the Bridge*

n people need to cross a narrow rope bridge as quickly as possible, and each respective person crosses at speeds $s_1, s_2, \ldots, s_n$ (you can assume these are integers and are sorted in descending order). You must also follow these additional constraints:

1. It is nighttime and you only have a single flashlight. One requires the flashlight to cross the bridge.
2. A max of two people can cross the bridge together at one time (and they must have the flashlight).
3. The flashlight must be walked back and forth, it cannot be thrown, mailed, raven'd, etc.
4. A pair walking across together crosses at the speed of the slowest individual. They must stay together!

Describe a greedy algorithm that solves this problem optimally. State the runtime of your algorithm and prove your algorithm always returns the optimal solution. *NOTE: The obvious greedy algorithm does NOT work here. Be careful! This is more complicated than it appears.*

Answer

In this problem we have n people who need to cross the bridge as quickly as possible.

We need ALL the people to cross that bridge, so we always deal with the case that we must wait for each individual to cross with their own pace. What makes the difference **here would be the time that it takes for the person who is holding the flashlight**. That person should be as fast as possible and **not waste time.**

If we have the array with speeds of persons, S = [s1, s2, s3 … sn] in descending order, then it means that the fastest person is the first. The greedy algorithm which always select the first person (fastest) to deliver other persons one by one wastes some valuable time. We can consider the case when he delivers the two slowest persons. Here, we lose time. Maybe it would be better if the two slowest persons walk together rather than accompanied by the fastest.

Let's try to solve by analyzing some subproblems. There are two ways to go with the slowest person to the bridge:

- Fastest with slowest then fastest goes back -> total cost would be time for fastest + time for slowest.
- The two fastest go together, second fastest goes back, two slowest go together, fastest who stayed there goes back -> total cost would be slowest + fastest + 2 * second Fastest.

If the optimal time to gather all the people would be T, then T would be equal to the minimum of taking two kind of steps explained above in each step.

The correct solution would have a Non-Polynomial time as we consider a lot of pairs that can go in the same time.

If there were only 4 people, then a correct solution would be for the first two to go first, the fastest goes back, the slowest go together, the second fastest who already crosses to go back, then for the two fastest to go back again.