Homework 6: Q2

Name: Xinkai Lin, xinkaili

*Don't forget to input your list of collaborators and sources on **AutoLab**.*
Please submit this file as a PDF.

1 Algorithm Idea

In this problem, I'm going to use greedy algorithm obviously just as the Hint point out. First of all, the input giving is a list of which store the distance of houses to the first house. I will sort the input list so it will store the distance in an ascending order. Now we have all the distance correctly sorted. Assume the first house would have the distance of 0, we place the first tower at here. Since the tower will cover 100 miles ahead on the road from its location, coverage will be the distance + 100. Since the distance of the first house is 0, 0+100=100. I initialize the coverage to be 100. From first house to last house, checking its distance, now this tower will cover up to distance from 0 to 100. If the distance of house is greater than the coverage, add the new tower to the house before this and add that house to the list and set the coverage to be the distance at that house + 100. In the same pattern, we will loop through all the distance of the houses to check if the tower can cover both house and the road. Finally, return the tower list which will tell us where to put the tower.

2 Algorithm Details

```
Input: giving a list of h which store the distance of houses to the first house with n house
sort h in an ascending order according to its distance
tower = {s} // initial tower list contains the first house with 0 distance
coverage = 100
for (i=1; i<n; i++) do
    if h[i] > coverage do
        add h[i-1] to tower
        coverage = h[i-1] + 100
    end if
end for
return tower
```

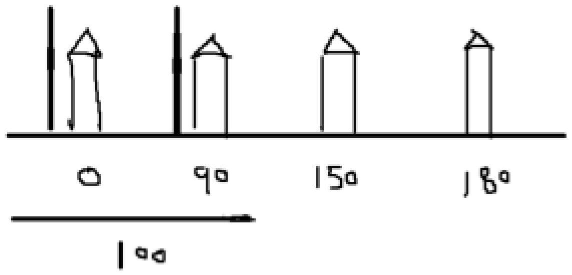
3 Proof of Correctness Idea

This algorithm is correct because it will cover both the house and the road with the minimum number of towers. First of all, I sorted the list, so it would show to be from first house to last house according to the distances. Starting at the first house with the distance of 0 and add the

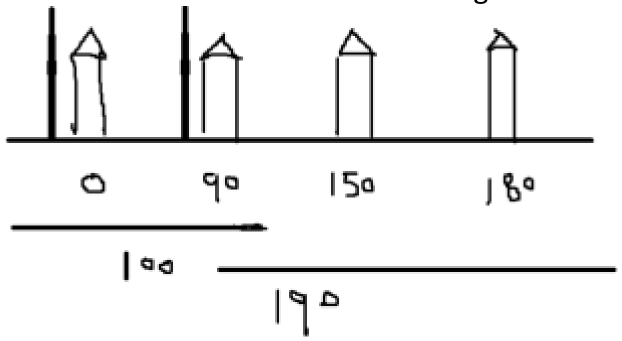
first tower. Since one cell tower can provide high speed internet access to all houses within 100 miles ahead. Whenever I had to add a tower I will add 100 to the distance where I set the tower. And because No two consecutive houses are more than 99 miles apart.
This make sure that the tower will cover all houses within 100 miles ahead. For the optimization of the algorithm, I will compare the coverage with the distance of each house. If the distance is greater than the coverage let say the coverage is 100 now, and the distance is at 210, this means the high-speed internet only cover until the house and road up to 200. In this case, I would add the tower at the house one position previous to 210. Let say it's 150, now the coverage will be 150+100=250. 210 and the road between is now cover. Then, I will compare the next house in the same way and so on. When the algorithm finished, we will get the minimum number of the tower which covers all the houses and road. Which prove that this is correct.

4 Proof Details

I'll showing some examples to prove that this is correct. For instance, if I have the house as



showing:
I would add my first tower at 0, which has the initial coverage of 100.
Then I will compare to the other houses. The next one is 90, which has been covered. The one after it has distance of 150, which is greater than 100, so I will add the tower at the house one position before which is 90 as showing below:



Now it will cover 90+100=190 miles. Which will cover all the road and the houses with minimum number of towers.

5 Runtime Analysis

The runtime of this algorithm is O(nlog(n)) time. The sort will have the runtime of O(nlog(n)). Then, it has a for loop which loop through all the n houses. Each operation inside for loop will take O(1), which will take up to O(n). Finally, the final runtime for this algorithm is O(nlog(n)) + O(n), which is O(nlog(n)).