Casey Levy – CS 325 – HW 4

Problem 1

- Iterate through all hotels (**h**) and compare their distances to drive distance (**d**)
- When hotel distance is **> d**, take **h – 1** (previous hotel) to stay for the night. This new **h** is starting point for the next day. Repeat these steps until $\mathbf{X_n}$ is reached (last hotel)
- Runtime: **O(n)**

Problem 2

$\mathbf{j_i}$ = job, $\mathbf{d_i}$ = deadline, $\mathbf{p_i}$ = penalty

Knowing $\mathbf{1 \le i \le n}$ …

```
- sort penalties in descending order
- if $d_i = i$ or $i = n$
      schedule the job
- else continue algorithm call on the rest of the remaining jobs
```

- Runtime: **O(nlogn)**

Problem 3

- The proposed problem is simply the same as the original problem of always selecting the first activity to finish, just in reverse this time and we're now selecting the last activity that is compatible. Since it's essentially the same, just in reverse, this proves it is still a greedy algorithm since we're still taking the "best-looking choice at each step", just as we would if the problem was iterating in ascending order. Additionally, this information proves that the proposed problem also yields an optimal solution.

Problem 4

a) The algorithm will take in the data from the "act.txt" file and then pass it into the greedy algorithm in descending/reverse order. Similar to problem 3, the job ending the latest will be taken first.

- **Pseudocode:**

```
def activities(arr):
    append activity info such activity number, start time, and end time into acts[]
    for i in range (len(arr)):
        if len(acts) == 0:
            append activities with the first one in acts[]
        elif arr[activity][end] <= acts[-1][start]:
            append acts with arr[acts]
    activities taken = new list from completed acts list
```

```
            reverse the taken list and return
            print data to terminal
```

b) Runtime: **O(nlogn)**