CSC 464/564 Homework 1 Instructor: Jeff Ward
Covers: Dynamic Programming 25 points

Planning a Trip

Consider Problem 6.2 from the textbook (Algorithms, by Dasgupta, Papadimitriou, and Vazirani):

You are going on a long trip. You start on the road at mile post 0. Along the way there are n hotels, at mile posts $a_1 < a_2 < \ldots < a_n$, where each a_i is measured from the starting point. The only places you are allowed to stop are at these hotels, but you can choose which of the hotels you stop at. You must stop at the final hotel (at distance a_n), which is your destination.

You'd ideally like to travel 200 miles a day, but this may not be possible (depending on the spacing of the hotels). If you travel x miles during a day, the penalty for that day is $(200 - x)^2$. You want to plan your trip so as to minimize the total penalty - that is, the sum, over all travel days, of the daily penalties.

Give an efficient algorithm that determines the optimal sequence of hotels at which to stop.

The file HW1Data.txt, provided on Canvas, contains a list of one hundred prospective stops: $a_1 < a_2 < \ldots < a_{100}$.

```
66
83
99
104
109
164
263
266
276
306
337
343
418
…
4833
4905
```

Your program should read in a file like the above and create a corresponding output file, HW1Output.txt, that lists the mile posts of an optimal trip from the 0 mile post to the final mile post, along with the total penalty that has accumulated at each stop:

```
Distance: 0 Total Penalty: 0
Distance: 164 Total Penalty: 1296
Distance: 343 Total Penalty: 1737
Distance: 558 Total Penalty: 1962
Distance: 741 Total Penalty: 2251
Distance: 919 Total Penalty: 2735
…
Distance: 4735 Total Penalty: 8639
Distance: 4905 Total Penalty: 9539
```

In order to facilitate checking your results, I have provided the file HW1OutputWard.txt, also on Blackboard, which gives the results of running my solution on HW1Data.txt.

Your program should be flexible enough to handle input files of various lengths.

Dynamic Programming

With one hundred or more stops to consider, it is impossible for the program to finish in a reasonable amount of time if it uses a straightforward "brute-force search" that explicitly considers every possible combination of stops. For this assignment, you are required to use a dynamic programming solution. We will design such a solution in class. Write and test an implementation in the computer language of your choice. You should use an array (or array list or vector) to store your intermediate sub-results, and compute these sub-results "bottom-up": That is, compute answers to the smaller subproblems first.

What to turn in:
Submit your completed program on Blackboard. The finished source code can be contained in a single file or split between multiple files, as you prefer. (It should not be a very long program, so a single file probably suffices.) Be sure to include some comments at the beginning of each file giving your name and a brief description of the file contents.