1 Completing Homeworks (33 points)

Suppose (hypothetically) that you were taking a class, possibly called “Algorithms”, in which the homeworks were extremely difficult. After enough complaining, the professor decided to make the following changes. There are two homework assignments each week rather than one, an “easy” assignment and a “hard” assignment. The hard assignment is worth more points, but it is in fact so difficult that you can only complete it if you’re completely rested and prepared, meaning that you cannot have done either of the assignments the week before.

More formally, let \( n \) be the number of weeks in the class, let \( h_i \) be the number of points for the hard assignment in week \( i \), and let \( e_i \) be the number of points for the easy assignment in week \( i \). Note that \( h_i \) does not have to be equal to \( h_j \) for \( i \neq j \) (although it might be), and similarly with \( e_i \) and \( e_j \). Assume that you know all of these values in advance. Then the goal is compute a schedule which in each week tells you whether to do nothing, the easy assignment, or the hard assignment and maximizes the total number of points, subject to the restriction that if you do a hard assignment in week \( i \) you cannot have done any assignment in week \( i - 1 \).

(a) One obvious approach would be to choose a hard assignment in week \( i \) if we get more points than if we completed the easy assignments for weeks \( i \) and \( i - 1 \). This would be the following algorithm:

\[
i = 1
\text{while}(i < n) \{
\text{if}(h_{i+1} \geq e_{i+1} + e_i) \{
\text{choose no assignment in week } i,
\text{choose the hard assignment in week } i + 1,
\text{ } i = i + 2
\}
\text{else} \{
\text{choose the easy assignment in week } i,
\text{ } i = i + 1
\}
\}
\]
Give an instance in which this algorithm does not return the optimal solution. Also say what
the optimal solution is (and its value) and what the algorithm finds instead.

(b) Give an efficient (polynomial time) algorithm which takes as input the values $e_1, \ldots, e_n$ and
$h_1, \ldots, h_n$ and returns the value of the optimal schedule. Prove its correctness and running
time.

2 Longest Path (33 points)

Let $G = (V, E)$ be a directed graph on vertices $v_1, \ldots, v_n$. We say that it is a forward-graph if
all edges are of the form $(v_i, v_j)$ where $j > i$, i.e. edges only go from smaller vertices to bigger
ones. Suppose that we want to find the longest path from $v_1$ to $v_n$ (or output NULL if none exists)
in a forward-graph. Give the fastest algorithm that you can find for this problem, and prove its
correctness and running time.

3 Mobile Business (33 points)

Let’s say that you have a great idea for a new food truck, and in order to save money you decide
to run it out of your RV so you can live where you work. Each day $i$ there is some demand for your
food in Baltimore and some demand in Washington – let’s say you would make $B_i$ dollars by being
in Baltimore and $W_i$ dollars by being in Washington. However, if you wake up in one city (due to
being there the previous day) and want to serve in the other city, it costs you $M$ dollars to drive
there.

The goal in this problem is to devise a maximum-profit schedule. A schedule is simply an
assignment of locations to days – for each day $i$, the schedule says whether to serve in Baltimore
or Washington. The profit of a schedule is the total profit you make, minus $M$ times the number
of times you have to move between cities.

For example, let $M = 10$ and suppose that $B_1 = 1, B_2 = 3, B_3 = 20, B_4 = 30$ and
$W_1 = 50, W_2 = 20, W_3 = 2, W_4 = 4$. Then the profit of the schedule (Washington, Washington, Baltimore, Baltimore) would be $W_1 + W_2 + B_3 + B_4 - M = 110$, while the profit of the schedule (Washington, Baltimore, Baltimore, Washington) would be $W_1 + B_2 + B_3 + W_4 - 2M = 50 + 3 + 20 + 4 - 20 = 57$.

Given the fixed driving cost $M$ and profits $B_1, \ldots, B_n$ and $W_1, \ldots, W_n$, devise an efficient algo-

rithm to compute the profit of an optimal schedule. Prove correctness and running time.