Name: Jaime Arana-Rochel

Homework #5

Q1: Exercise 7 Chapter 6

First we define *lowest*(i) to be the lowest price among the first i days, and *low-day*(i) the day associated with that lowest price. So our optimal solution $OPT(i)$ is the greatest profit that can be made among the first i days. Because $OPT(i)$ is either achieved through selling the stock on the i-th day, or not selling it on that day, we get the following recurrence:

$$OPT(i) = \max(p(i) - lowest(i), OPT(i-1)) \tag{1}$$

So to find the optimal numbers i and j, we would loop over all n, computing the $OPT(i)$ and placing it in an array M. Thus M would be built from the bottom up and ultimately contain the optimal profit. In each iteration, we would also keep track of whether we sold on day i or not by keeping using another array. By the end, we will have computed i to be *low-day*(i), and j to be the value we kept track of in each iteration.

So in total, we would be doing constant time work for each iteration of the for loop where we build our array M. This gives us a total running time of $O(n)$.

Q2: Exercise 8 Chapter 6

(a)

Suppose $n = 4$, and the values of x_1-x_4 = 1,10,10,2 respectively. The values of $f(1)$-$f(4)$ = 1,2,4,8 respectively. The given algorithm would choose activate the EMP at times 2 and 4, destroying 4 robots. However, the optimal solution would be to activate at times 3 and 4, destroying 4 robots.

(b)

Algorithm

We define the recurrence relation

$$OPT(n) = \max_{0 \le i < n} [\min(x_n, f(n-i)) + OPT(i)]$$

```
Schedule-EMP(x1,...,xn):
   M[0] = 0
   For i = 1 to n
      M[i] = OPT(i) defined above
   endfor

   return M[n]
```

Correctness

We derive the recurrence relation $OPT(n)$ given above as follows: Over the course of n seconds, we always activate the EMP on the n^{th} second since there is no penalty for doing so. So on the n^{th} second, we destroy $\min(x_n, f(n-i))$ robots. Then, we just need to choose when to last activate it before the n^{th} second. For all i less than n, we compute $OPT(i)$ such that we get the maximum value for $OPT(i) + \min(x_n, f(n-i))$. That maximum value is then our optimal solution $OPT(n)$.

We proceed with a proof by induction on n:
Base case n = 1: Clearly we can only activate the EMP on the first second, giving us the maximum amount of robots we can destroy in that one second span. The algorithm in this case returns $M[1]$.
Inductive Step: Let our induction hypothesis IH be the following - Assume that for $1 \leq i < n$, the algorithm correctly writes $OPT(i)$ into the array entry $M[i]$.
Consider when $i = n$. By the IH, $OPT(i) = M[i]$. Thus,

$$OPT(n) = \max_{0 \leq i < n}[\min(x_n, f(n-i)) + OPT(i)]$$

$$= \max_{0 \leq i < n}[\min(x_n, f(n-i)) + M[i]]$$

$$= M[n]$$

This is exactly the array entry value that the algorithm computes. Since the algorithm returns this optimal solution $M[n]$ for $i = n$, by induction we are done.

Running Time

For every iteration of the for loop, the running time is $O(n)$ since we compute the maximum over all j less than i. This gives us a total running time $O(n^2)$.