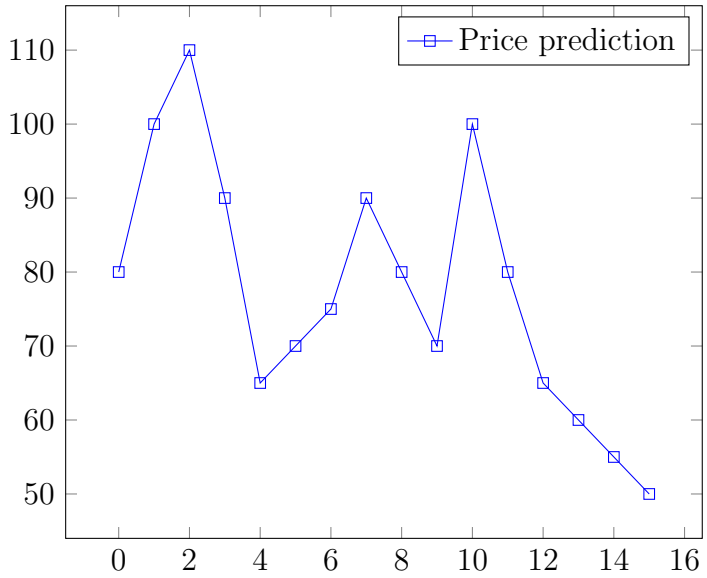
ECSE210L: Design and Analysis of Algoithms

Lab 2 (Week 2: January, 13 - 17, 2020)

Instructors: ***Shakti Sharma and Raghunath Reddy M***

1. The maximum-subarray problem

   You are predicting the price of a share by using a 'share price future prediction tool. You want to maximize the profit by following simple strategy 'buy low, sell high'. Consider following example, where horizontal axis indicates number of days, and vertical axis shows the price.



   Write a program to identify the best time to buy and sell the share in order to maximise profit.

   Try on your own, calculate the complexity of your algorithm, improve the complexity to O(nlogn).

   After the implementation of your method, implement the brute-force method explained in the Hint 1.

   Hint1:

   Note: Only see if you are unable to solve this problem.

   One simple strategy can be to identify the lowest and highest price. Then work left from the highest price to search prior lowest price and similarly work right from the lowest price to find highest price. This strategy will not always work.

   Check this example, will above mentioned strategy work on this example? Write in a word file.

Another **brute-force strategy** is to check every possible pair i.e. all cases where buy date precedes the sell date. Implement brute-force method. What is the complexity of this algorithm. Explain it in a word file.

Hint 2: If you are unable to solve the problem in O(nlogn) time after multiple attempts then only proceed with this section.

Use the following algorithms, implement the method, show and explain functionalities of algorithm.

Algorithm 1: Find-Max-Crossing-Subarray(A,low,mid,high)

```
left-sum = - ∞ ;
sum = 0 ;
for ( i = mid downto low ) {
    sum = sum + A[i];
    if sum  left-sum then
        left-sum =sum;
        max-left = i;
    end
    right-sum = - ∞ ;
    sum = 0 ;
}
for ( j = mid+1 to high ) {
    sum = sum + A[i];
    if sum  right-sum then
        right-sum =sum;
        max-right = j;
    end
}
```

Result: max-left, max-right, left-sum+right-sum