Jordan Allard
CSCI 264-01
Homework 3

HW3 Problem 4 Writeup

a) Heart of the Solution

I) WHAT: S[j] = the max sum of the sequence formed by jumping two or three spaces ending with the jth element (i.e., the jth element is included)

II) HOW: S[j] = max(S[j - 2], S[j - 3]) + A[j]

III) WHERE: max(S[n], S[n - 1])

b) Pseudocode

```
Let A be an array of ints of length n
Let S be an empty array of length n
If n >= 1: S[1] = A[1]
If n >= 2: S[2] = A[2]
If n >= 3: S[3] = S[1] + A[3]
If n >= 4: S[4] = S[1] + A[4]
If n >= 5: S[5] = S[3] + A[5]
For j = 5 to n:
      S[j] = max(S[j - 2], S[j - 3]) + A[j]
Return max(S[n], S[n - 1])
```

c) Proof of Correctness (explanation of HOW)

Each sequence must include the number either two or three indexes before it. It doesn't matter what the sequence is prior to those numbers, only that the ending numbers at those indices are included in their sequences. If they are included, then the max sum of all the numbers following the rules of the sequence can be generalized by taking the larger of the sum of the sequence up unti either the number two or three spaces behind the current index plus the amount at the current index, since the sequence can only reach the current number from one of those two other numbers. The final max total must be either the last or second to last sum, since any sequence further back will always include one of the last two numbers to increase the total. Solving the problem this way also ensures that the algorithm will end up processing every piece of input.

d) Running Time Estimate

O(n)

e) Running Time Estimate Reasoning

This algorithm only needs to run through every item in the input one time to calculate the totals. Including the time needed to process the input, the total running time is O(n) + O(n) = O(n).