Jordan Allard
CSCI 264-01
Homework 4

HW4 Problem 1 Writeup

a) Heart of the Solution

I) WHAT: S[j] = the max sum of an increasing subsequence with gap p chosen from the first j elements ending with a[j]

II) HOW: S[j] = max(S[k]) + a[j], where j - p <= k < j and a[k] < a[j]

III) WHERE: max value of S

b) Pseudocode

```
Given an array of ints A with length n and an int p
Let S be an array of ints with length n
maxSum = 0
For j = 1 to n:
      S[j] = a[j]
      maxSeqSum = 0;
      For k = 1 to p:
            If j - k >= 0 and a[j - k] < a[j]:
                  If S[j - k] > maxSeqSum:
                        maxSeqSum = S[j - k]
      S[j] += maxSeqSum
      If S[j] < maxSum:
            maxSum = S[j]
Return maxSum
```

c) Proof of Correctness (Explanation of HOW)

By the rules of the algorithm, a valid subsequence that includes an element at index i must be either a continuation of a subsequence from one of the previous p elements, in which the current element would be the largest number, or a brand new subsequence. By checking each of these possibilities and picking the option with the largest value, the algorithm is guaranteed to find the subsequence that produces the largest possible sum for each "end" element. The maximum sum possible for any element is just the largest value among the maximum sums possible for each specific end element.

d) Running Time Estimate

O(np)

e) Running Time Estimate Reasoning

Each step in this algorithm is constant time except for the two for loops, which will run n times and p times respectively. Since these loops are nested, the final running time will be O(np). (Including the input loop, the time complexity is still O(n) + O(np) = O(np).)