CS325 Group Assignment 1 Solutions

By Lance Umagat & Garrett Pement

Algorithm 1 Pseudocode
MaxSubarray (A [1.. n]):
 res = 0
 for i = 1 , ... , n
  for j = i , ... , n
   sum = 0
   for k = i , ... , j
    sum = sum + A [k]
   res = max (sum , res)
 return res

Algorithm 1 Runtime analysis: The number of time that the algorithm adds two numbers together is $\sum_{i=1}^{n} \sum_{j=i}^{n} \sum_{k=i}^{j} 1 = \Theta(n^3)$. The number of times that this algorithm finds the max of two numbers is $\sum_{i=1}^{n} \sum_{j=i}^{n} 1 = \Theta(n^2)$.

Algorithm 2 Pseudocode
MaxSubarray (A [1.. n]):
 res = 0
 for i = 1 , ... , n
  sum = 0
  for j = i , ... , n
   sum = sum + A [j]
   res = max (res , sum)
 return res

Algorithm 2 Runtime analysis: The number of time that the algorithm adds two numbers together is $\sum_{i=1}^{n} \sum_{j=i}^{n} 1 = \Theta(n^2)$. The number of times that this algorithm finds the max of two numbers is $\sum_{i=1}^{n} \sum_{j=i}^{n} 1 = \Theta(n^2)$.

Algorithm 3 Pseudocode
MaxSubarray (A [1.. n]):
 res = A [1]
 mem = A [1]
 for i = 2 , ... , n
  sum = mem + A [i]
  mem = max (sum , mem)
  res = max (res , mem)
 return res

Algorithm 3 Runtime analysis: The number of time that the algorithm adds two numbers together is $\sum_{i=1}^{n} 1 = \Theta(n)$. The number of times that this algorithm finds the max of two numbers is $\sum_{i=1}^{n} 1 = \Theta(n)$.

Experimental Runtime Analysis:

Graphs are in attached excel file
Based on the graphs. We found that as the input sizes got bigger the run times for each of the algorithms became more clearly differentiable. Although as for algorithms 3 and 2 between inputs 100-900 the run times are fairly close. However, after about 1000 the runtimes became much clearer. Algorithm 1 started to deviate after about 400 inputs.

Notes:
We couldn't get the read from file to work properly for more than one array. As such we have decided just to hard code the arrays into the code and run as needed. We talked to TA Evgenia about this already.