**Question 1**

**1 (a and b)**

Algorithm thirdLargestByThreeLoops(arr) #operations

firstIdx <- 0 1

secondIdx <- 0 1

thirdIdx <- 0 1

for int i <- 0 to arr.length do 2 + n

if arr[i] > arr[firstIdx] then 3n

firstIdx <- i n

for int i <- 0 to arr.length do 2 + n

if i != firstIdx && arr[i] > arr[secondIdx] then 4n

secondIdx <- i n

for int i <- 0 to arr.length do 2 + n

if i != firstIdx && i != secondIdx && arr[i] > arr[thirdIdx] then 5n

thirdIdx <- i n

return arr[thirdIdx] 1

**Total 18n**

**1 (c)** Algorithm 1 by three loops: time complexity: O(n), space complexity O(n). Used extra 2 loops to determine the third max. Speed is slower than Algo2 because it runs two additional n times for loops.

**2 (a and b)**

findFirstThreeMax(A,n) #operations

max<-minInt 1

preMax<-minInt 1

prePreMax<-minInt 1

for i<-0 to n-1 do 2 + n

if A[i]>prePreMax then 2n

if (A[i]>=max) then 2n

prePreMax = preMax 1n

preMax = max 1n

max = A[i] 2n

else if (A[i]>=preMax && A[i]<max) then 4n

prePreMax = preMax 1n

preMax = A[i] 2n

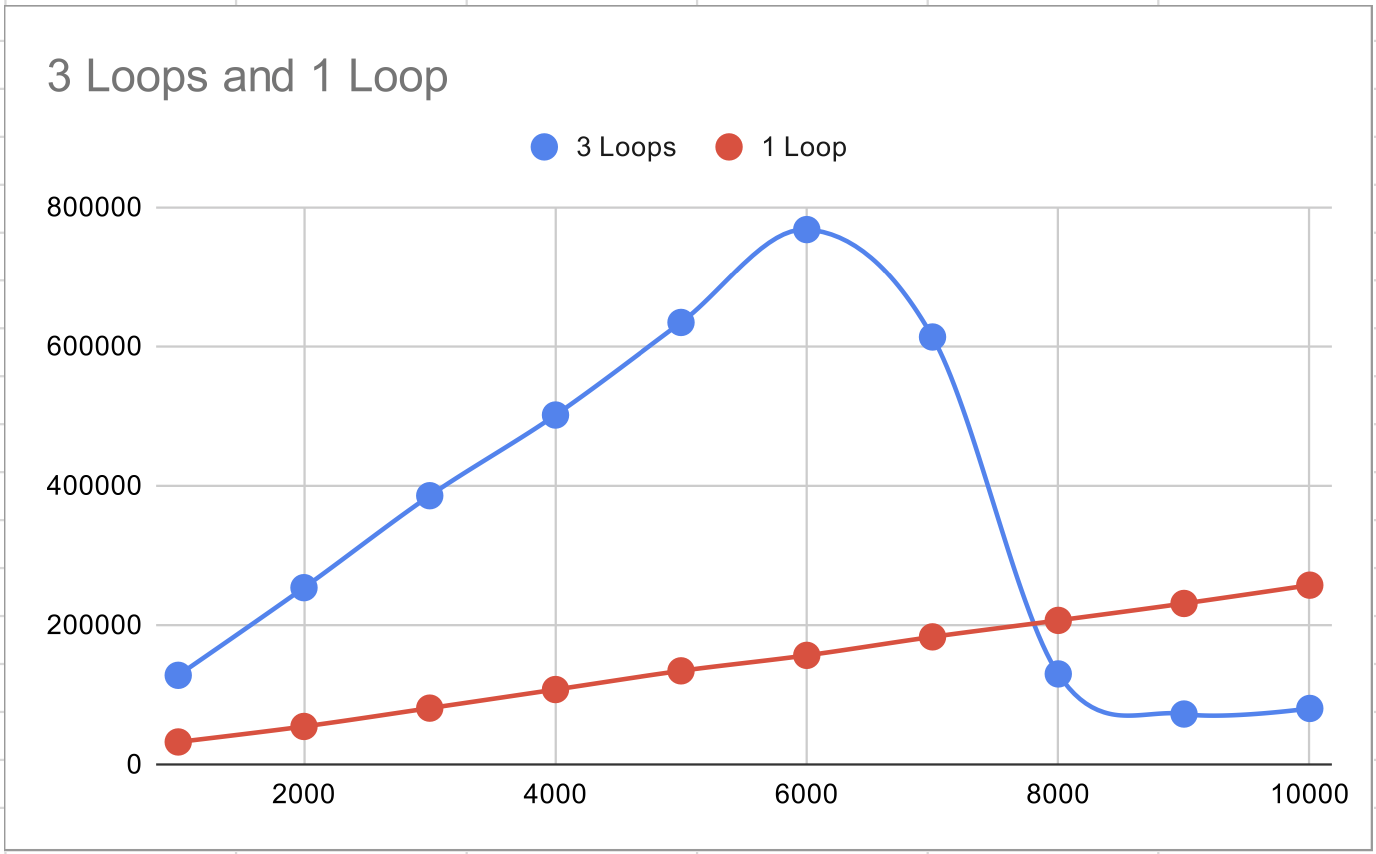
else

prePreMax = A[i] 2n

**Total 18n**

**2 (c)** Algorithm 2 by 1 loop: time complexity: O(n) and space complexity is also O(n). Only used n times for loop to determine the third max. Speed is faster than an algorithm 1.

**1 and 2 (d)**

****

**Question 2**

|  |  |
| --- | --- |
| 10, 1 | 𝚯(1) |
| log(log(n)) | 𝚯(log(log(n))) |
| n1/3logn | 𝚯(n1/3log n) |
| n1/2logn | 𝚯(n1/2log n) |
| logn, ln n | 𝚯(logn) |
| lognn, nlogn | 𝚯(nlogn) |
| n1/k(k>3) | 𝚯(n1/k) |
| n1/3 | 𝚯(n1/3) |
| n1/2 | 𝚯(n1/2) |
| n2 | 𝚯(n2) |
| n3 | 𝚯(n3) |
| nk(k>3) | 𝚯(nk) |
| 2n | 𝚯(2n) |
| 3n | 𝚯(3n) |
| n! | 𝚯(n!) |
| nn | 𝚯(nn) |