1-b)

[25, 25, 75, 8, 42, 67, 7, 21, 66, 72]

[7, 8, 21, 25, 25, 42, 66, 67, 72, 75]

1-c)

sorting 1000 element

Min time: 0

Average time: 0.16

Max time: 2

Sorting 10000

Min time: 12

Average time: 14.65

Max time: 18

1-d)

by changing the number of elements, the average runtime of the program increased nearly 2 to the power of 4 , I mean in my opininon it increased exponentially.

2-b)

[77, 46, 35, 68, 11, 10, 8, 30, 51, 31]

[8, 10, 11, 30, 31, 35, 46, 51, 68, 77]

2-c)

sorting 1000 element

Min time: 1

Average time: 0.58

Max time: 4

Sorting 10000 element

Min time: 24

Average time: 24.58

Max time: 27

2-d)

by changing the number of elements, the average runtime of the program increased almost 25 fold and it is between 2 to the power of four and two the the power of five.

3)

if the size of our array is small, ı mean we have less element, as in our example 1000 element, insertion sort performs better than selection sort. Meanwhile, if the number of elements increases, the performance of insertion sort becomes more better than selection sort. So, if ı need to sort elements, ı would use insertion sort.