

Exercises 2.2

1) $\Theta(n^3)$

2)

SELECTION-SORT (A, n)

for $i = 1$ to $n - 1$

 smallest = i

 for $j = i + 1$ to n

 if $A[j] < A[\text{smallest}]$

 smallest = j

 temp = $A[\text{smallest}]$

$A[\text{smallest}] = A[i]$

$A[i] = \text{temp}$

Loop invariant :- (outer loop) :- Before each iteration
the sub array $A[1:i-1]$ is sorted
contains the smallest $i-1$ elements of
 A and it is sorted

It only has to run for the first $n-1$
elements because the ~~if~~ th by then the
~~smallest~~ first $i+n-1$ (since $i=n$) elements of the
array are already correctly sorted and
in the subarray $A[1:n-1]$ and so the
 n^{th} element naturally ends up in the
correct place

Worst case :- $\Theta(n^2)$

The best case is not any better, it is
also $\Theta(n^2)$

3) In the ~~worst~~ average case there are $\frac{1+2+3+\dots+n}{n}$ checks, wh:-

$$\begin{aligned} \frac{1+2+3+\dots+n}{n} &= \frac{n(n+1)}{2} \\ &= \frac{n^2+n}{2n} = \frac{n+1}{2} \end{aligned}$$

Where n is the number of elements in the array.

This is because if the element being searched for can be in position 1, 2, 3 all the way up to n , with each possibility equally likely. So it is also equally likely that we check 1, 2, 3 or all the way to n elements. Add all possibilities ($1+2+\dots+n$) and take average (divide by n).

The worst case occurs when the element being searched for does not exist in the array; and so we must check all n elements.

In the average case, the only loop runs $(n+1)/2$ times (the other statements are constants so we ignore them). So it is $\Theta(n)$. Similarly, in the worst case it runs n times, which is still $\Theta(n)$.

4) Do a ~~sinAdd~~ an if condition before the main algorithm body that checks if the input is already sorted by doing a single pass, and if it is, then return the input as is.