Slide 12 – Matrix Sum, Worst Case:

Number of statements per inner loop iteration: 2
Number of inner loop iterations: n
Total cost of inner loop: $2n + 1$ (extra +1 for when loop condition is false)

Number of statements per outer loop iteration: 1+ cost of inner loop $= 2n + 2$
Number of outer loop iterations: n
Total cost of outer loop: $n(2n + 2) + 1 = 2n^2 + 2n + 1$ which is $O(n^2)$.

Slide 12 – Matrix Sum, Best Case:

Same as the worst case. Algorithm runs same number of statements for any input of size n. Thus we can say that the algorithm is $\Theta(n^2)$.

Slide 13 – Prefix Averages, Worst Case:

Number of statements per inner loop iteration: 3
Number of inner loop iterations: $i + 1$
Total cost of inner loop: $3i + 4$ (extra +1 for when loop condition is false)

Number of statements per outer loop iteration: 5+ cost of inner loop $= 3i + 9$
Number of outer loop iterations: n, runs for values of i from 0 to $n - 1$
Total cost of outer loop: $\sum_{i=0}^{n-1} 3i + 9$
Total cost:

$$\begin{aligned}
\left(\sum_{i=0}^{n-1} 3i + 9\right) + 3 &= 9n + 3 + \sum_{i=0}^{n-1} 3i \\
&= 9n + 3 + 3\sum_{i=0}^{n-1} i \\
&= 9n + 3 + 3\frac{(n-1)(n)}{2} \\
&= 9n + 3 + 3\frac{(n^2 - n)}{2} \\
&= 9n + 1.5n^2 - 1.5n + 3 \\
&= 1.5n^2 + 7.5n + 3 \\
&\quad \text{which is } O(n^2).
\end{aligned}$$

Slide 13 – Prefix Averages, Best Case:

Same as the worst case. Algorithm runs same number of statements for any input of size n. Thus we can say that the algorithm is $\Theta(n^2)$.

Slide 14 – Binary Search, Worst Case: Worst case will be when searching for an item not in the array.

How many statements per loop iteration? In the worst case, the first if-statement is always false, and the second one is always true, resulting in 5 statements per iteration.

How many loop iterations are there? (How many times is the loop condition true?) Each time through the loop, the value of `hi-lo` will be halved.

First loop iteration: `hi-lo` $= n = n/2^0$
Second loop iteration: `hi-lo` $= n/2^1$
Third loop iteration: `hi-lo` $= n/2^2$
i-th loop iteration: `hi-lo` $= n/2^i$

When $i = \log(n)$, `hi-lo` $= n/2^{\log n} = n/n = 1$. Thus, on the $\log(n) + 1$-th iteration, it must be the case that `hi` $=$ `lo`. During this iteration, `mid` $=$ `hi` $=$ `lo`. Then either `hi` is decreased by 1, or `lo` is increased by 1. Either way, the condition `lo <= hi` will be false by the end of the iteration, ending the loop. Thus the loop condition is true at most $\log(n) + 1$ times.

So the entire loop, therefore, results in $5(\log(n) + 1)$ statements plus one for when the loop condition is false for $5\log(n) + 6$ statements. Finally add in the 3 statements outside the loop and we get $5 \log n + 9$ which is $O(\log(n))$.