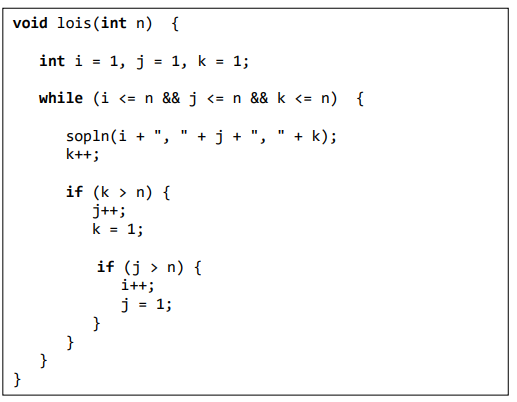
Brian Ortiz

CS-324

8/31/23

Assignment Two

1. Problem 1 (4 pts) As a function of n, what is the exact number of lines of output of the call lois(n)?

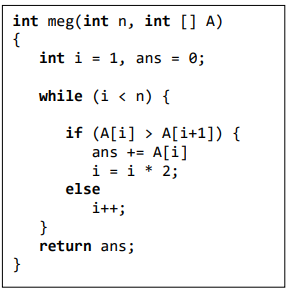


Solution:  
For this case we see 1 output line, in this case sopln(i + ", " + j + ", " + k);

However, the output is repeated based on how many times *k* increments over *n*, and how many times *j* increments over *n*, and lastly how many times *i* increments to *n.* Let’s take an example:  
if n=5  
  
we will print line 5 times for *k* + another *k* times for 5 times to *j* + another *k* times and another *j* times 5 times for *i*.   
  
Total lines of output = *n* (innermost loop iterations for each value of *j*) \* *n* (middle loop iterations) \* *n* (outermost loop iterations) = 5 \* 5 \* 5 = 125

In short, number of lines for output is *n^3*

2. Problem 2 (8 pts) Consider the following method:



(a) In terms of 𝜃-notation, what is the worst-case running time of meg?

𝜃(n) if all indexes of A[I] are all less than A[I+1] we incrementally check each pass through causing us to check every value even when ans=0 in the end for all n times.

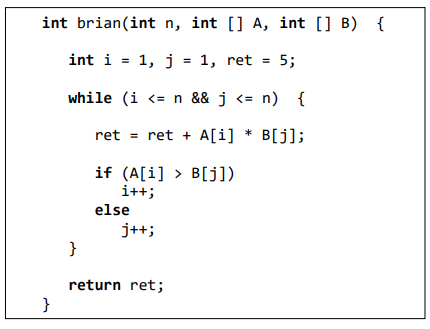
(b) In terms of 𝜃-notation, what is the best-case running time of meg?  
 as the loop runs the best case will be 𝜃 (logn). As A[I] are all greater than A[I+1] we exponentially grow I to skip indexes depending on I value while being less than n. I after being powered by two becomes larger than n the loop ends. If n=10 I=1,2,4,8,16 with only 4 passes in the loop.

3. Problem 3: (4 pts) In terms of 𝜃-notation, what is the running time of stewie?

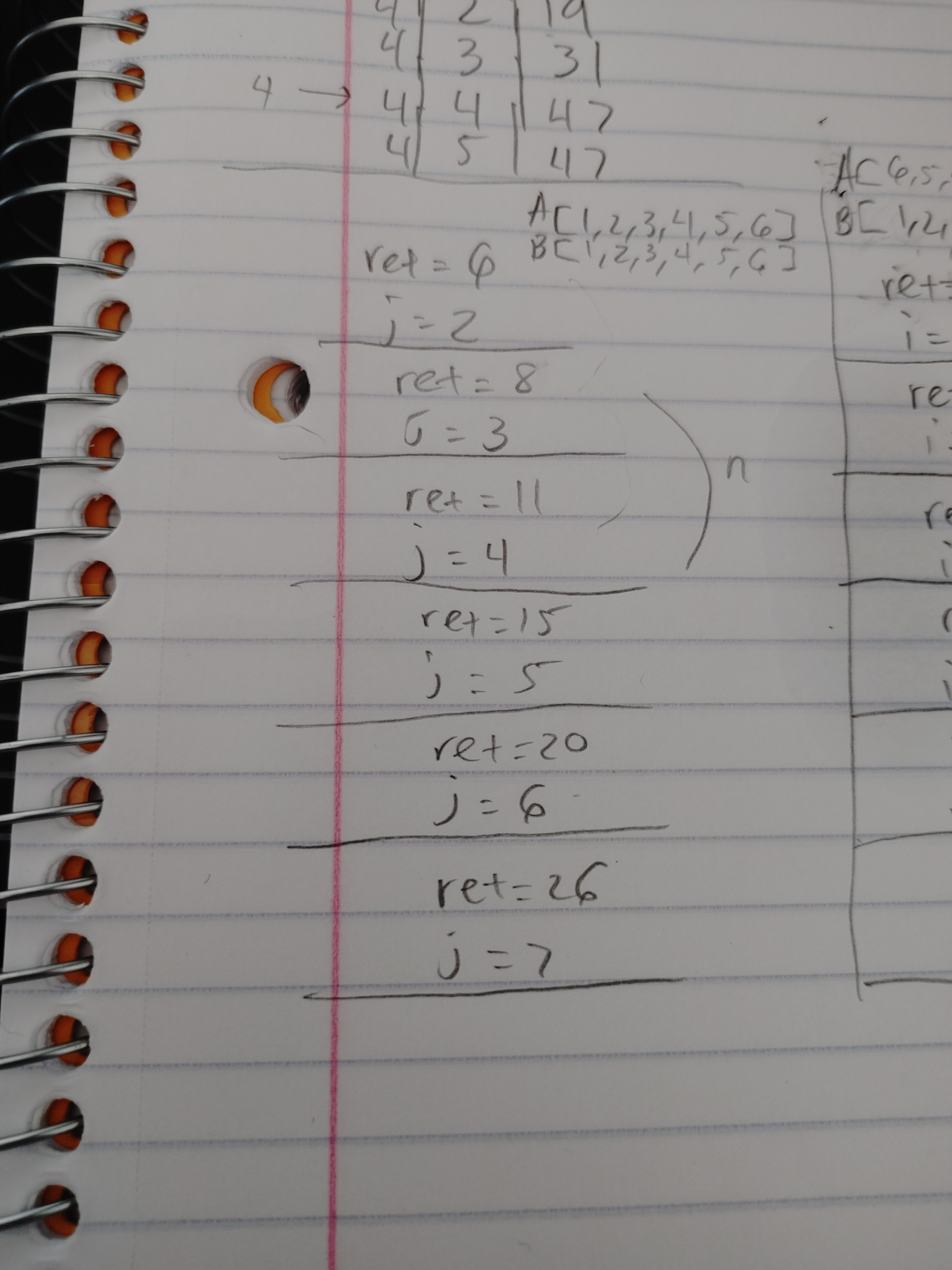


𝜃(n log n) , as the first loop runs logarithmically, the inner loop is iterating over n times. This means that n \* lg n = n log n

4. (8 pts) Consider the following method:



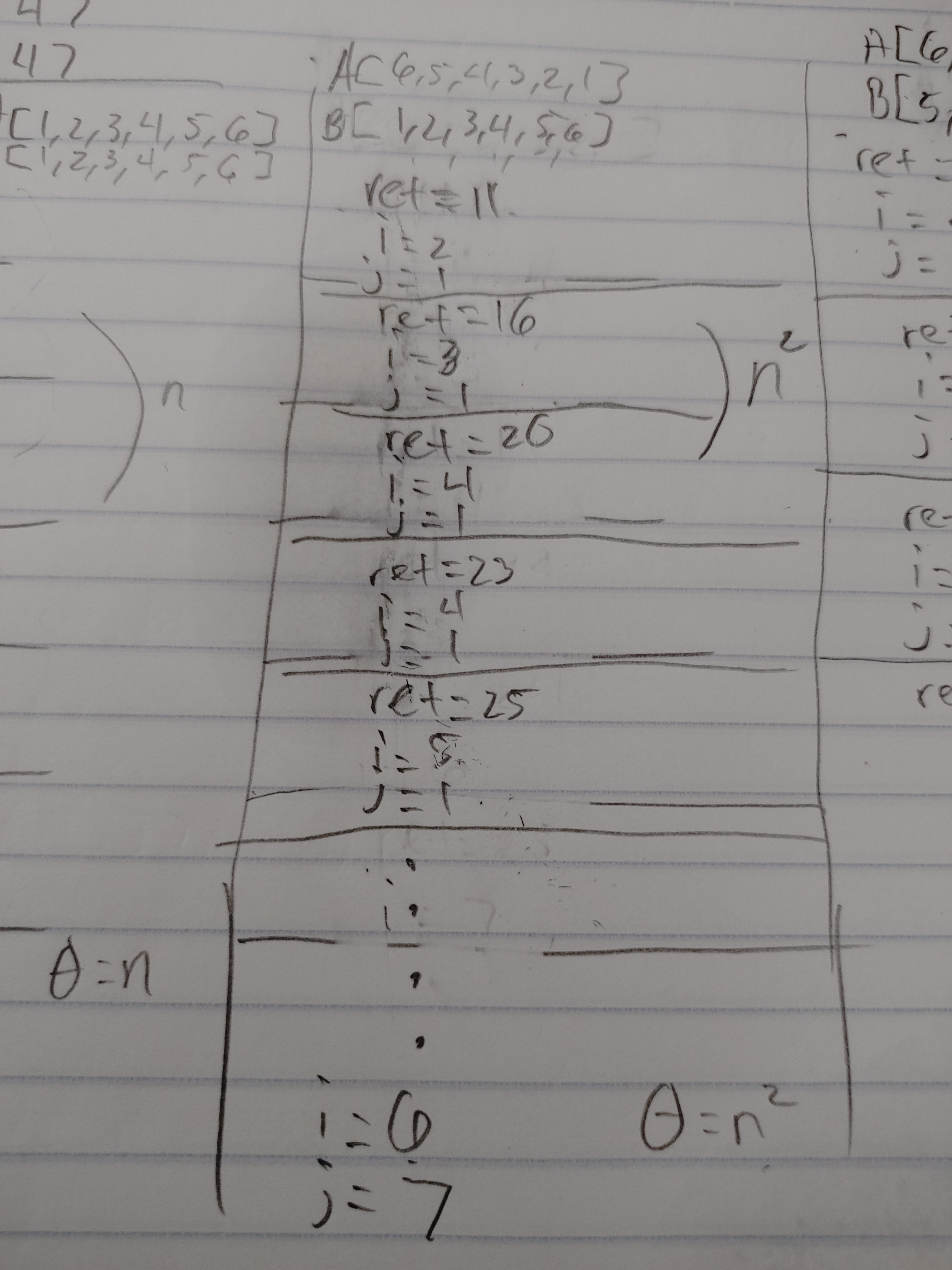
4.1) As an exact function of n, what is the best-case (fewest) number of array element comparisons made by method brian? Give arrays A and B of size 𝑛 = 6 that achieve this best case.

For best case if we gave Array A[1,2,3,4,5,6] and B[1,2,3,4,5,6] . The loop will flow as follows:  


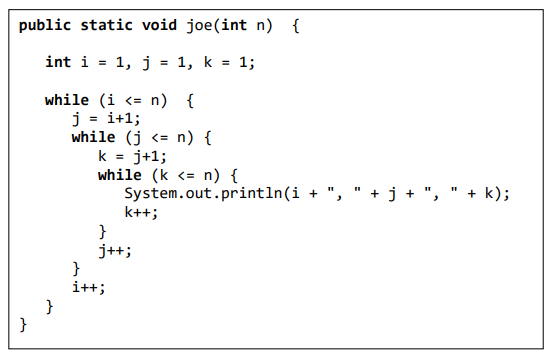
This means that while index value of I is always less than index value of j, j will increment linear times. Resulting in linear function or 𝜃(n).

4.2) As an exact function of n, what is the worst-case (largest) number of array element comparisons made by method brian? Give arrays A and B of size 𝑛 = 6 that achieve this worst case.

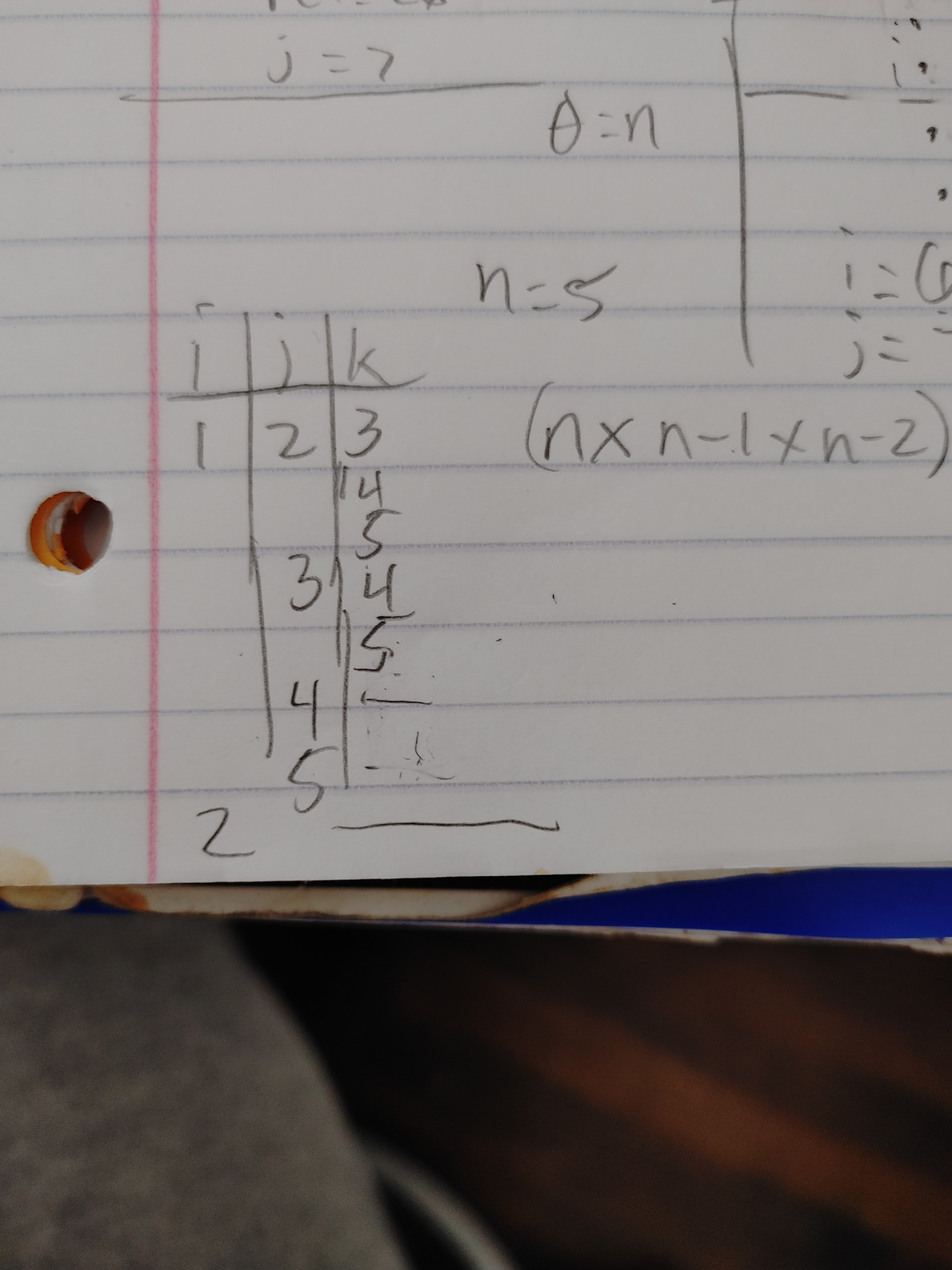
For worst case we can have state where function must iterate through all I and j indexes when comparing for increments. For example, if A[6,5,4,3,2,1] and B[1,2,3,4,5,6] we that the iterations would go thru n^2 times resulting in worst case being 𝜃(n^2)



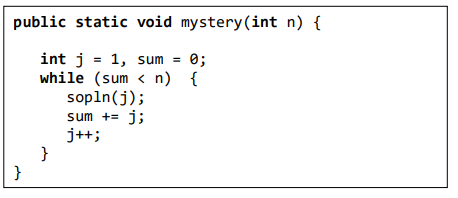
5. (+3 pts extra credit) As a function of n, what is the exact number of lines of output of the call joe(n)?



Answer:  
This one was tricky and might not be exactly true, but since we are stating for every I there is j-1 iterations and k-2 iterations. Maybe our exact run time can be measured as (n \* (n-1) \* (n-2)). I cannot say for sure but here is how I wrote the steps down:



6. (+3 pts extra credit) In terms of 𝜃-notation, what is the running time of "mystery"?



Answer: the running time for this function is 𝜃(n) as each time we iterate thru j the value of j adds into sum. if j is 1 sum is 1, if j is 2 sum is 3 . If n was 6 the loop would stop after just 3 iterations.  
See details below:

