Thomas X Mei

911647469

CSC 340

Hui Yang

11/09/2014

Anything not mentioned is assumed to have ran once.

CSC 340 HW 6

Question One:

1. double sum\_fifth (double array[], int n)

2. //n: size of the array. Assume n is divisible by 5

3. {

4. double sum=0;

5. for (int i=0; i<n; i=i+5)

6. sum = sum + array[ i ];

7. return sum;

8. }

Line 5: (n/5) +1

Since it increments by 5 each time we divide by 5 from n, and the + 1 is for the final check of if i < n

Line 6: (n/5)

Same number of times as line 5 but since it will not run after the final check, which results in failure, so it’s whatever number of times line 5 had minus 1.

Line 7: (n/5)

Same number of times as Line 6

Question Two

1. double sum\_exponentials(int n)

2. //n is a power of 10, i.e., n=10k or k=log10n

3. {

4. int sum=0;

5. for (int i=1; i<n; i=i\*10){

6. sum = sum + i;

7. for (j=0; j< n; j++)

8. sum = sum + j;

9. }

10. return sum;

11. }

Line 5: (log\_10(n)+1)

It is incremented itself \*10 each time, therefore log\_10(n) and the final check which results in failure is added

Line 6: (log\_10(n))

Same idea as question one line 6, it is one less than what ever line 5 was.

Line 7: (n+1)(log\_10(n)), in a normal circumstance, as in if this line was the first line, it would have been just (n+1) but since this is a nested for loop it will be (n+1) multiplied by how ever many times it occurred prior to this, which is (log\_10(n)) therefore (n+1)(log\_10(n)) being the answer

Line 8: (n(log\_10(n))

It runs however many times the previous one runs minus one due to not running again if the check results in failure ( j < n)

Line 10: (log\_10(n))

Since it is in the outer loop and not in the inner loop.

Question Three

1. for (int i=n; i>=0; i--)

2. for (int j=i; j<n; j++)

3. cout << i << “,” << j <<endl;

Line 1: (n+2)

Since it includes 0 and needs the check which results in failure, it is n + 1+1 = (n+2

Line 2: (n+1)((n/2)+1)

Since this nest for loop is the standard one, it will be (n+1) multiplied by whatever the previous one was, BUT, int j =I, if you were to draw out a grid of it every single time you increment j by one, you get one less times ran on the next one, it looks somewhat like a triangle, or a square cut in half diagonally, therefore the ½. So we take n and divide it by 2 and add one for the check which in results in failure and multiply it by (n+1)

Line 3: (n)((n/2)+1)

Since it only goes through n times even though it checks it n+1 times, it only really executes this line n((n/2)+1) times instead of (n+1)((n/2)+1 times

Question Four

1. for (int i=0; i<n; i++) //n is even

2. for (int j=i; j<n/2; j++)

3. cout << i << “,” << j <<endl;

Line 1: (n+1)

This is just the simple iterator incremented once per try and last check which results in failure

Line 2: ½ (n/2)+1

½ because it only takes even numbers and n/2 because it was int j=i , refer to prior problems for explanation ( square cut in half diagonally) and of course the +1at the end for final check which results in failure.

Line 3: ½(n/2)

Last check on line 2 fails so the last one will not go through, therefore simply minus one from however many times Line 2 ran, which his ½ (n/2)