**Tut 1 Q4**

Type out your Math analysis of the individual code segments.

Instead of writing if you have a tablet or copy-pasting equations made from LaTex (for those familiar), you may want to try typing using the Microsoft Equation editor (press Alt + = for Windows users, or go to Insert -> Equation). Do try it!

|  |  |
| --- | --- |
| **Code Segment** | **Your Math Analysis** |
| public static void Q1a(int n) {  int a = 0;  for(int i=0; i<n; i++)  a += Q1b(n);  System.out.println(a);  }  public static int Q1b(int n) {  int b = 0;  while(n>0) {  b += n;  n = n / 2;  }  return b;  } | The while loop in Q1b is executed times. The time complexity of Q1b is thus O(logn). The for loop in Q1b is executed n times. Thus the time complexity of Q1a is O(nlogn) . |
| for(int i = 0; i < n; i++){  for(int k=0; k<n; k++){  /\*some sequence of O(1) steps\*/  } |  |
| for(int x = 1; x < n; x \*= 2){  /\* some sequence of O(1) steps \*/  } | At the ith iteration of the loop, , thus when x=n . Thus the time complexity is |
| // loop 1  for(int i = 0; i < n; i++)  // loop 2  for(int j = i+1; j > i; j--)  // loop 3  for(int k = n; k > j; k--) System.out.println(“\*”); | Loop 2 only executes once since j=i+1 will not be greater than i when decremented. In loop 3, j=i+1 thus the loop will execute n-j=n-i-1 times.  Thus the time complexity is O(n^2) |