WORKSHEET 2

Activity One.

Integer Number entered.	(Runtime) Factorial	(Runtime) Fibonacci.	Factorial Result.	Fibonacci Result.
5	862ms	825ms	120	5
31	1007ms	23111ms	738197504	1346269
32	636ms	30207ms	-2147483648 (invalid)	2178309
41		1820493ms		165580141
42		infinite		No result given.

Looking at the above table we can see that higher the integer number entered the more time it takes. This happens due to recursion. In recursion we always sort the smaller problem before the bigger problem and bring about results. Therefore, when int 5 is entered the method recurse 5 times while int 31 is entered the method recurse for 31 times to get the result .This is why more time is taken to get results of higher integers.

The time complexity of Factorial is (O)n because the method has to recurse n times to get the result. While the space complexity of factorial is O(N) since there is no extra space needed during recursive calls.

But as we can see the highest value we can enter to get a valid result is 31 because an int is 32 bits so the times the method can recurse should be less than 32. After 31 when we enter number 32 we get invalid results due to a stack overflow.

But in Fibonacci the time complexity is 2 ^(O)n. The Fibonacci tree is much larger than the input we give because it has two instances (Does an addition of two numbers to get the final result). The first case time complexity is O(n). The second case time complexity is 2^O(n). Therefore, the time complexity of Fibonacci recursive method is 2^O(n).

Fibonacci method only gives results up to 40 which means that it can recurse only 40 times.