Name: ______________________________

For each of the following $T(n)$, write the corresponding Big O time complexity. Some series may require research.

1. (2 points) $T(n) = n^2 + 3n + 2$

   1. $O(n^2)$

2. (2 points) $T(n) = (n^2 + n)(n^2 + \frac{\pi}{2})$

   2. $O(n^4)$

3. (2 points) $T(n) = 1 + 2 + 3 + \ldots + n - 1 + n$

   3. $O(n^2)$

4. (2 points) $T(n) = 1^2 + 2^2 + 3^2 + \ldots + (n-1)^2 + n^2$

   4. $O(n^3)$

5. (2 points) $T(n) = 10$

   5. $O(1)$

6. (2 points) $T(n) = 10^{100}$

   6. $O(1)$

7. (2 points) $T(n) = n + \log n$

   7. $O(n)$

8. (2 points) $T(n) = 12\log(n) + \frac{n}{2} - 400$

   8. $O(n)$

9. (2 points) $T(n) = (n+1) \cdot \log(n) - n$

   9. $O(n\log(n))$

10. (2 points) $T(n) = \frac{n^4+3n^2+2n}{n}$

    10. $O(n^4)$

_______ 20 points

11. (4 points) What is the time complexity to insert or remove an item in the middle of an ArrayList?

Solution: $O(n)$

12. (4 points) Why?

Solution: Because we shift half the items in the list.

13. (4 points) What is the **average** time complexity to add an item to the end of an ArrayList?

Solution: $O(1)$ on average

14. (4 points) What is the **worst case** time complexity to add an item to the end of an ArrayList?

Solution: $O(n)$

15. (4 points) Taking this all into account, what situations would an ArrayList be the appropriate data structure for storing your data?

Solution: When we are doing a lot of adding to the end of the arraylist and doing lots of get or set operations.

______________ 20 points

```
public static int[] allEvensUnder(int limit){
        if (limit <= 0){
                return new int[0];
        }
        if (limit < 2){
                return new int[1];
        }
        int[] vals =  new int[(limit+1)/2];
        for(int i  = 0;  i <(limit+ 1)/2  ; i++ ) {
                vals[i] = i*2;
        }
        return vals;
}
```

16. (5 points) What is the **time** complexity of the above algorithm?

    Solution: $O(n)$

17. (5 points) What is the **space** complexity of the above algorithm? In other words, how much space is used up based on the input size?

    Solution: $O(n)$

```
/*
*  https://rosettacode.org/wiki/Sorting_algorithms/Insertion_sort#Java
*/
public static void insertSort(int[] A){
        for(int i = 1; i < A.length; i++){
                int value = A[i];
                int j = i - 1;
                while(j >= 0 && A[j] > value){
                        A[j + 1] = A[j];
                        j = j - 1;
                }
                A[j + 1] = value;
        }
}
```

18. (10 points) What is the time complexity of the above algorithm?

    Solution: $O(n^2)$

`bogosort` attempts to sort a list by shuffling the items in the list. If the list is unsorted after shuffling, we continue shuffling the list and checking until it is finally sorted.

19. (5 points) What is the worst case run time for `bogosort`?

Solution: Student can answer Infinite or $O(n!)$

20. (5 points) Why?

Solution: Student can answer it either goes on forever as there's no guarantee to end or that it runs through all possible permutations of list.

21. (5 points) What is the average case run time for `bogosort` (Hint: think about a deck of cards)?

Solution: $O(n!)$

22. (5 points) Why?

Solution: We run through all possible permutations and find a sorted one approximately half way through.

23. (20 points) For each of the methods you wrote in Lab 2, figure out the time complexity of the method you wrote. To turn in this portion, attach a printout of the code and specify the time complexity of each.

______________ 20 points