Joel Miller
CPSC 3120
09/27/2023

1.

1. $1/n$
2. 2^{100}
3. $\log \log n$
4. sqrt($\log n$)
5. $\log^2 n$
6. $n^{0.01}$
7. sqrt(n), $3\ n^{0.5}$
8. $2^{\log n}$, $5\ n$
9. $n \log_4 n$, $6\ n \log n$
10. $2\ n \log^2 n$
11. $4\ n^{3/2}$
12. $4^{\log n}$
13. $n^2 \log n$
14. n^3
15. 2^n
16. 4^n
17. 2^{2n}

2.

	1 Second	1 Hour	1 Month	1 Century
log n	10^{300000}	$2^{36x10\text{^}8}$	$2^{2.5x10\text{^}12}$	$2^{3.1x10\text{^}16}$
√n	10^{12}	$1296x10^{16}$	$6.7x10^{24}$	$9.9x10^{32}$
n	10^6	$36x10^8$	$2592x10^9$	$3.1x10^{15}$
n log n	62746	$1.3x10^8$	$7.1x10^{10}$	$6.8x10^{13}$
n^2	1000	60000	1609968	56175382
n^3	100	1532	13736	146677
2^n	19	31	41	51
n!	9	12	15	17

3. Create an array to store starting indices for finding maximum prefixes. In the first loop, update the array[t] to "t-1" if "Mt-1 + A[t] > 0," otherwise set it to "t." In the second loop, if index "m" corresponds to the maximum value in array "M," then the starting index of the maximum sub-array is "startIndex[m]," and the ending index is "m."

Joel Miller
CPSC 3120
09/27/2023

4. To modify the MaxsubFastest algorithm for a single-loop approach, initialize M and the current sum to 0. Traverse the array and add each element to the current sum. If the current sum turns negative, reset it to 0. During each iteration, update M to the maximum of M and current sum.

5. In this case, when a counter variable increments by three, the time complexity becomes logarithmic, LOG(n), with the base determined by the size of the counter increment; however, if the counter increment is a constant value, it remains O(n) in overall time complexity.

6. This is possible because big O notation only describes asymptotic behavior and doesn't consider constant factors or smaller input sizes, which can favor Bill's O(n^2) algorithm for n < 100, despite Al's O(n log n) algorithm having better theoretical complexity for larger inputs.