## **Step One: Simplifying Expressions**

Simplify the following big O expressions as much as possible:

1. O(n + 10) **O(n)**
2. O(100 \* n) **O(n)**
3. O(25) **O(1)**
4. O(n^2 + n^3) **O(n^3) \***
5. O(n + n + n + n) **O(n)**
6. O(1000 \* log(n) + n) **O(n) \***
7. O(1000 \* n \* log(n) + n) **O(n log n)**
8. O(2^n + n^2) **O(2^n) \***
9. O(5 + 3 + 1) **O(1)**
10. O(n + n^(1/2) + n^2 + n \* log(n)^10) **O(n^2) \***

## **Step Two: Calculating Time Complexity**

Determine the time complexities for each of the following functions. If you’re not sure what these functions do, copy and paste them into the console and experiment with different inputs!

function logUpTo(n) {

for (let i = 1; i <= n; i++) {

console.log(i);

}

}

**Time Complexity: O(n)**

function logAtLeast10(n) {

for (let i = 1; i <= Math.max(n, 10); i++) {

console.log(i);

}

}

**Time Complexity: O(n)**

function logAtMost10(n) {

for (let i = 1; i <= Math.min(n, 10); i++) {

console.log(i);

}

}

**Time Complexity: O(1)**

function onlyElementsAtEvenIndex(array) {

let newArray = [];

for (let i = 0; i < array.length; i++) {

if (i % 2 === 0) {

newArray.push(array[i]);

}

}

return newArray;

}

**Time Complexity: O(n)**

function subtotals(array) {

let subtotalArray = [];

for (let i = 0; i < array.length; i++) {

let subtotal = 0;

for (let j = 0; j <= i; j++) {

subtotal += array[j];

}

subtotalArray.push(subtotal);

}

return subtotalArray;

}

**Time Complexity: O(n^2)**

function vowelCount(str) {

let vowelCount = {};

const vowels = "aeiouAEIOU";

for (let char of str) {

if(vowels.includes(char)) {

if(char in vowelCount) {

vowelCount[char] += 1;

} else {

vowelCount[char] = 1;

}

}

}

return vowelCount;

}

**Time Complexity: O(n)**

## **Part 3 - short answer**

Answer the following questions

1. **True** or false: n^2 + n is O(n^2).
2. **True** or false: n^2 \* n is O(n^3).
3. True or **false**: n^2 + n is O(n).
4. What’s the time complexity of the .indexOf array method? **O(n)**
5. What’s the time complexity of the .includes array method? **O(n)**
6. What’s the time complexity of the .forEach array method? **O(n) at least (depends on what the callback does)**
7. What’s the time complexity of the .sort array method? **O(n log n)**
8. What’s the time complexity of the .unshift array method? **O(n)**
9. What’s the time complexity of the .push array method? **O(1)**
10. What’s the time complexity of the .splice array method? **O(n)**
11. What’s the time complexity of the .pop array method? **O(1)**
12. What’s the time complexity of the Object.keys() function? **O(n)**

### BONUS

1. What’s the space complexity of the Object.keys() function? **O(n)**