Big-O Notation Practice

In this exercise, you'll analyze expressions and code to figure out the time complexity.

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) $O(n \cdot \log n)$ ✓
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);
  }
}
```

$O(n)$ ✓

Time Complexity:

```
function logAtLeast10(n) {
  for (let i = 1; i <= Math.max(n, 10); i++) {
    console.log(i);
  }
}
```

$O(10) \rightarrow O(1)$ x

Max → n
⇒ $O(n)$

Time Complexity:

```
function logAtMost10(n) {
  for (let i = 1; i <= Math.min(n, 10); i++) {
    console.log(i);
  }
}
```

$O(\leq 10) \rightarrow O(1)$ ✓

Time Complexity:

```
function onlyElementsAtEvenIndex(array) {
  let newArray = [];
  for (let i = 0; i < array.length; i++) {
    if (i % 2 === 0) {
      newArray.push(array[i]);
    }
  }
  return newArray;
}
```

$T \to O(n)$ ✓

$S \to O(n)$

Time Complexity:

```
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;
}
```

$T \to O(n^2)$ ✓

$S \to O(n)$

Time Complexity:

```
function vowelCount(str) {
  let vowelCount = {};
  const vowels = "aeiouAEIOU";

  for (let char of str) {        n
    if(vowels.includes(char)) {  1
      if(char in vowelCount) {   1
        vowelCount[char] += 1;
      } else {             n-1
        vowelCount[char] = 1;
      }
    }
  }

  return vowelCount;
}
```

$O(n)$ ✓

Time Complexity:

Part 3 - short answer

Answer the following questions

1. True or false: n^2 + n is O(n^2). T $O(n^2)$ ✓
2. True or false: n^2 * n is O(n^3). F $O(n^2)$ ✗ $\rightarrow n \times n^2 = n^3$ $O(n^3)$ $n^3 > n^2$ so n^3
3. True or false: n^2 + n is O(n). F $O(n^2)$ ✓
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? $O(n)$ ✓
7. What's the time complexity of the .sort array method? $n \times \log n$ ✓
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? $O(n)$

$\rightarrow$ worse cases, so $O(n)$

BONUS

13. What's the space complexity of the Object.keys() function? $O(n)$ ✓

Solution

View our solutions <solution/index.html>