Exercises on Big O Analysis

|  |
| --- |
| **// 1. Even or odd** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **function isEven(value){** |
|  |

|  |
| --- |
| **if (value % 2 == 0){** |
|  |

|  |
| --- |
| **return true;** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **else** |
|  |

|  |
| --- |
| **return false;** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **/\*** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **// 2.** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **function areYouHere(arr1, arr2) {** |
|  |

|  |
| --- |
| **//let ticks1, ticks2 = 0;** |
|  |

|  |
| --- |
| **for (let i=0; i<arr1.length; i++) {** |
|  |

|  |
| --- |
| **const el1 = arr1[i];** |
|  |

|  |
| --- |
| **//ticks1++;** |
|  |

|  |
| --- |
| **for (let j=0; j<arr2.length; j++) {** |
|  |

|  |
| --- |
| **const el2 = arr2[j];** |
|  |

|  |
| --- |
| **//ticks2++;** |
|  |

|  |
| --- |
| **if (el1 === el2) return true;** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **//console.log(ticks1);** |
|  |

|  |
| --- |
| **//console.log(ticks2);** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **return false;** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **/\*** |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **// 3.** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **function doubleArrayValues(array) {** |
|  |

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

|  |
| --- |
| **array[i] \*= 2;** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **return array;** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **/\*** |
|  |

|  |
| --- |
|  |

|  |
| --- |
| **\*/** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **// 4. Naive Search** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **function naiveSearch(array, item) {** |
|  |

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

|  |
| --- |
| **if (array[i] === item) {** |
|  |

|  |
| --- |
| **return i;** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **/\*** |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **// 5. Creating Pairs** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **function createPairs(arr) {** |
|  |

|  |
| --- |
| **//let ticks = 0;** |
|  |

|  |
| --- |
| **for (let i = 0; i < arr.length; i++) {** |
|  |

|  |
| --- |
| **for(let j = i+1; j < arr.length; j++) {** |
|  |

|  |
| --- |
| **console.log(arr[i] + ", " + arr[j] );** |
|  |

|  |
| --- |
| **//ticks++;** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **//console.log(ticks);** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |

|  |
| --- |
| **\*/** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **// 6. Computing Fibonacci Numbers** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **function generateFib(num) {** |
|  |

|  |
| --- |
| **let result = [];** |
|  |

|  |
| --- |
| **//let ticks = 0;** |
|  |

|  |
| --- |
| **for (let i = 1; i <= num; i++) {** |
|  |

|  |
| --- |
| **//ticks++;** |
|  |

|  |
| --- |
| **if (i === 1) {** |
|  |

|  |
| --- |
| **result.push(0);** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **else if (i == 2) {** |
|  |

|  |
| --- |
| **result.push(1);** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **else {** |
|  |

|  |
| --- |
| **result.push(result[i - 2] + result[i - 3]);** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **//console.log(ticks);** |
|  |

|  |
| --- |
| **return result;** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **// 7. Efficient Search** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **function efficientSearch(array, item) {** |
|  |

|  |
| --- |
| **let minIndex = 0;** |
|  |

|  |
| --- |
| **let maxIndex = array.length - 1;** |
|  |

|  |
| --- |
| **let currentIndex;** |
|  |

|  |
| --- |
| **let currentElement;** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **while (minIndex <= maxIndex) {** |
|  |

|  |
| --- |
| **currentIndex = Math.floor((minIndex + maxIndex) / 2);** |
|  |

|  |
| --- |
| **currentElement = array[currentIndex];** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **if (currentElement < item) {** |
|  |

|  |
| --- |
| **minIndex = currentIndex + 1;** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **else if (currentElement > item) {** |
|  |

|  |
| --- |
| **maxIndex = currentIndex - 1;** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **else {** |
|  |

|  |
| --- |
| **return currentIndex;** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **return -1;** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **// 8. Random element** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **function findRandomElement(arr) {** |
|  |

|  |
| --- |
| **return arr[Math.floor(Math.random() \* arr.length)];** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **// 9. Is It Prime?** |
|  |

|  |
| --- |
|  |
|  |

|  |
| --- |
| **function isPrime(n) {** |
|  |

|  |
| --- |
| **if (n < 2 || n % 1 != 0) {** |
|  |

|  |
| --- |
| **return false;** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **for (let i = 2; i < n; ++i) {** |
|  |

|  |
| --- |
| **if (n % i == 0) return false;** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
| **return true;** |
|  |

|  |
| --- |
| **}** |
|  |

|  |
| --- |
|  |
|  |