***Do the below programs in anonymous function and IIFE***

**1.Print odd numbers in an array**

**AN0NYMOUS FUNCTION METHOD**

**// Getting input via STDIN**

**const readline = require("readline");**

**const inp = readline.createInterface({**

**input: process.stdin**

**});**

**const userInput = [];**

**inp.on("line", (data) => {**

**userInput.push(data);**

**});**

**inp.on("close", () => {**

**var odd = function(a){**

**var b = [];**

**for(var i = 0;i < a.length;i++){**

**if(a[i] % 2 !== 0){**

**b.push(a[i]); ;**

**}**

**}**

**return b;**

**}**

**var a = userInput[0].split(" ").map(function(val){**

**return Number(val);**

**})**

**console.log(odd(a))**

**});**

**Input:**

**10 20 39 41 50 56 73 32 21**

**Output:**

**[ 39, 41, 73, 21 ]**

**IIFE METHOD**

**// Getting input via STDIN**

**const readline = require("readline");**

**const inp = readline.createInterface({**

**input: process.stdin**

**});**

**const userInput = [];**

**inp.on("line", (data) => {**

**userInput.push(data);**

**});**

**inp.on("close", () => {**

**var a = userInput[0].split(" ").map(function(val){**

**return Number(val);**

**});**

**(function(a){**

**var b = [];**

**for(var i = 0;i < a.length;i++){**

**if(a[i] % 2 !== 0){**

**b.push(a[i]);**

**}**

**}**

**console.log(b);**

**})(a);**

**});**

**Input:**

**10 20 39 41 50 56 73 32 21**

**Output:**

**[ 39, 41, 73, 21 ]**

**2. Sum of all numbers in an array**

**AN0NYMOUS FUNCTION METHOD**

**// Getting input via STDIN**

**const readline = require("readline");**

**const inp = readline.createInterface({**

**input: process.stdin**

**});**

**const userInput = [];**

**inp.on("line", (data) => {**

**userInput.push(data);**

**});**

**inp.on("close", () => {**

**var a = userInput[0].split(" ").map(function(val){**

**return Number(val);**

**});**

**var sumOf = function(a){**

**var sum = 0;**

**for(var i = 0;i < a.length;i++){**

**sum += a[i];**

**}**

**return sum;**

**};**

**console.log(sumOf(a));**

**});**

**INPUT:**

**2 3 4 5 6 10**

**OUTPUT:**

**30**

**IIFE METHOD**

**// Getting input via STDIN**

**const readline = require("readline");**

**const inp = readline.createInterface({**

**input: process.stdin**

**});**

**const userInput = [];**

**inp.on("line", (data) => {**

**userInput.push(data);**

**});**

**inp.on("close", () => {**

**var a = userInput[0].split(" ").map(function(val){**

**return Number(val);**

**});**

**(function(a){**

**var sum = 0;**

**for(var i = 0;i < a.length;i++){**

**sum += a[i];**

**}**

**console.log(sum);**

**})(a);**

**});**

**INPUT:**

**2 3 4 5 6 10**

**OUTPUT:**

**30**

**3. Return all the prime numbers in an array**

**AN0NYMOUS FUNCTION METHOD**

**// Getting input via STDIN**

**const readline = require("readline");**

**const inp = readline.createInterface({**

**input: process.stdin**

**});**

**const userInput = [];**

**inp.on("line", (data) => {**

**userInput.push(data);**

**});**

**inp.on("close", () => {**

**var arr = userInput[0].split(" ").map(function(val){**

**return Number(val);**

**});**

**var prime = function(a){**

**var sum = 0;**

**var b = [];**

**for(var i = 0;i < a.length;i++){**

**if(a[i] !== 1){**

**var flag = 0;**

**for(var j = 2;j < a[i];j++){**

**if(a[i] % j === 0){**

**flag = 1;**

**break;**

**}**

**}**

**if(flag === 0){**

**b.push(a[i]);**

**}**

**}**

**}**

**return b;**

**}**

**var result = prime(arr);**

**console.log(result);**

**});**

**INPUT:**

**2 3 4 5 6 10**

**OUTPUT:**

**[ 2, 3, 5 ]**

**IIFE METHOD**

**// Getting input via STDIN**

**const readline = require("readline");**

**const inp = readline.createInterface({**

**input: process.stdin**

**});**

**const userInput = [];**

**inp.on("line", (data) => {**

**userInput.push(data);**

**});**

**inp.on("close", () => {**

**var arr = userInput[0].split(" ").map(function(val){**

**return Number(val);**

**});**

**(function(a){**

**var sum = 0;**

**var b = [];**

**for(var i = 0;i < a.length;i++){**

**if(a[i] !== 1){**

**var flag = 0;**

**for(var j = 2;j < a[i];j++){**

**if(a[i] % j === 0){**

**flag = 1;**

**break;**

**}**

**}**

**if(flag === 0){**

**b.push(a[i]);**

**}**

**}**

**}**

**console.log(b);**

**})(arr);**

**});**

**INPUT:**

**2 3 4 5 6 10**

**OUTPUT:**

**[ 2, 3, 5 ]**

**4. Remove duplicates from an array**

**ANONYMOUS FUNCTION**

**const readline = require("readline");**

**const inp = readline.createInterface({**

**input: process.stdin**

**});**

**const userInput = [];**

**inp.on("line", (data) => {**

**userInput.push(data);**

**});**

**inp.on("close", () => {**

**var a = userInput[0].split(" ").map(function(val){**

**return Number(val);**

**})**

**var removeDuplicates = function(arr){**

**var b = new Array(arr.length);**

**var c = [];**

**b.fill(false);**

**for(var i=0;i<arr.length;i++){**

**for(var j=i+1;j<arr.length;j++){**

**if(!b[j] && arr[i] === arr[j] ){**

**b[j] = true;**

**}**

**}**

**}**

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

**if (!b[i]){**

**c.push(arr[i]);**

**}**

**}**

**return c;**

**}**

**var c = removeDuplicates(a);**

**console.log(c);**

**});**

**INPUT:**

**10 20 30 20 20 30 40 50 60**

**OUTPUT:**

**[ 10, 20, 30, 40, 50, 60 ]**

**IIFE METHOD:**

**const readline = require("readline");**

**const inp = readline.createInterface({**

**input: process.stdin**

**});**

**const userInput = [];**

**inp.on("line", (data) => {**

**userInput.push(data);**

**});**

**inp.on("close", () => {**

**var a = userInput[0].split(" ").map(function(val){**

**return Number(val);**

**});**

**(function(arr){**

**var b = new Array(arr.length);**

**var c = [];**

**b.fill(false);**

**for(var i=0;i<arr.length;i++){**

**for(var j=i+1;j<arr.length;j++){**

**if(!b[j] && arr[i] === arr[j] ){**

**b[j] = true;**

**}**

**}**

**}**

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

**if (!b[i]){**

**c.push(arr[i]);**

**}**

**}**

**console.log(c);**

**})(a);**

**});**

**INPUT:**

**10 20 30 20 20 30 40 50 60**

**OUTPUT:**

**[ 10, 20, 30, 40, 50, 60 ]**

**5. Return all the palindromes in an array**

**ANONYMOUS METHOD**

**// Getting input via STDIN**

**const readline = require("readline");**

**const inp = readline.createInterface({**

**input: process.stdin**

**});**

**const userInput = [];**

**inp.on("line", (data) => {**

**userInput.push(data);**

**});**

**inp.on("close", () => {**

**var reverse = function(n){**

**var digit, rev = 0;**

**while(n>0){**

**digit = n % 10;**

**rev = (rev \* 10) + digit;**

**n = parseInt(n / 10);**

**}**

**return rev;**

**}**

**var isPalindrome = function(n){**

**if(n === reverse(n)){**

**return true;**

**}**

**else {**

**return false;**

**}**

**}**

**var findPalindromes = function(arr){**

**var result = [];**

**for (var i = 0; i < arr.length; i++){**

**if (isPalindrome(arr[i])){**

**result.push(arr[i]);**

**}**

**}**

**return result;**

**}**

**var a = userInput[0].split(" ").map(function(val){**

**return Number(val);**

**});**

**var b = findPalindromes(a);**

**console.log(b);**

**});**

**INPUT:**

**1221 345 234 2442**

**OUTPUT:**

**[ 1221, 2442 ]**

**IIFE METHOD:**

**// Getting input via STDIN**

**const readline = require("readline");**

**const inp = readline.createInterface({**

**input: process.stdin**

**});**

**const userInput = [];**

**inp.on("line", (data) => {**

**userInput.push(data);**

**});**

**inp.on("close", () => {**

**var a = userInput[0].split(" ").map(function(val){**

**return Number(val);**

**});**

**var reverse = function(n){**

**var digit, rev = 0;**

**while(n>0){**

**digit = n % 10;**

**rev = (rev \* 10) + digit;**

**n = parseInt(n / 10);**

**}**

**return rev;**

**};**

**var isPalindrome = function(n){**

**if(n === reverse(n)){**

**return true;**

**}**

**else {**

**return false;**

**}**

**};**

**(function(arr){**

**var result = [];**

**for (var i = 0; i < arr.length; i++){**

**if (isPalindrome(arr[i])){**

**result.push(arr[i]);**

**}**

**}**

**console.log(result);**

**})(a);**

**});**

**INPUT:**

**1221 345 234 2442**

**OUTPUT:**

**[ 1221, 2442 ]**

**6.Convert all the strings to title caps in a string array**

**ANONYMOUS FUNCTION**

**// Getting input via STDIN**

**const readline = require("readline");**

**const inp = readline.createInterface({**

**input: process.stdin**

**});**

**const userInput = [];**

**inp.on("line", (data) => {**

**userInput.push(data);**

**});**

**inp.on("close", () => {**

**var convertAllStringsToTitleCase = function(strArr){**

**var resultArr = [];**

**for(var i = 0; i < strArr.length ; i++){**

**resultArr.push(convertOneStringToTitleCase(strArr[i]));**

**}**

**return resultArr;**

**}**

**convertOneStringToTitleCase = function(str){**

**var arr = str.split(" ");**

**for(var i = 0; i < arr.length ; i++){**

**arr[i] = arr[i].charAt(0).toUpperCase() + arr[i].slice(1);**

**//console.log(arr[i]);**

**}**

**var resultString = arr.join(" ");**

**return resultString;**

**}**

**var strArray = userInput[0].split(",");**

**//console.log(strArray);**

**var resultStringArray = convertAllStringsToTitleCase(strArray);**

**console.log(resultStringArray);**

**});**

**INPUT:**

**siddharrth Karthikeyan, mongo db, node js**

**OUTPUT:**

**[Siddharrth Karthikeyan, Mongo Db, Node Js]**

**IIFE METHOD**

**// Getting input via STDIN**

**const readline = require("readline");**

**const inp = readline.createInterface({**

**input: process.stdin**

**});**

**const userInput = [];**

**inp.on("line", (data) => {**

**userInput.push(data);**

**});**

**inp.on("close", () => {**

**var strArray = userInput[0].split(",");**

**var convertOneStringToTitleCase = function(str){**

**var arr = str.split(" ");**

**for(var i = 0; i < arr.length ; i++){**

**arr[i] = arr[i].charAt(0).toUpperCase() + arr[i].slice(1);**

**}**

**var resultString = arr.join(" ");**

**return resultString;**

**};**

**(function(strArr){**

**var resultArr = [];**

**for(var i = 0;i < strArr.length;i++){**

**resultArr.push(convertOneStringToTitleCase(strArr[i]));**

**};**

**console.log(resultArr);**

**})(strArray);**

**});**

**INPUT:**

**siddharrth Karthikeyan, mongo db, node js**

**OUTPUT:**

**[Siddharrth Karthikeyan, Mongo Db, Node Js]**

**7. Rotate an array by k times and return the rotated array**.

**ANONYMOUS FUNCTION**

**// Getting input via STDIN**

**const readline = require("readline");**

**const inp = readline.createInterface({**

**input: process.stdin**

**});**

**const userInput = [];**

**inp.on("line", (data) => {**

**userInput.push(data);**

**});**

**inp.on("close", () => {**

**var k = parseInt(userInput[0]);**

**var a = userInput[1].split(" ").map(function(val){**

**return Number(val);**

**});**

**var rotateArrayOnce = function(a){**

**var temp = a[0];**

**for (var i = 0; i < a.length; i++){**

**a[i] = a[i+1];**

**}**

**a[a.length - 1] = temp;**

**}**

**var rotateArrayKTimes = function(a, k){**

**for(var i = 0; i < k; i++){**

**rotateArrayOnce(a);**

**}**

**}**

**rotateArrayKTimes(a,k);**

**console.log(a);**

**});**

**INPUT:**

**2**

**1 2 3 4 5**

**OUTPUT:**

**[3,4,5,1,2]**

**IIFE METHOD**:

**// Getting input via STDIN**

**const readline = require("readline");**

**const inp = readline.createInterface({**

**input: process.stdin**

**});**

**const userInput = [];**

**inp.on("line", (data) => {**

**userInput.push(data);**

**});**

**inp.on("close", () => {**

**var k = parseInt(userInput[0]);**

**var a = userInput[1].split(" ").map(function(val){**

**return Number(val);**

**});**

**var rotateArrayOnce = function(a){**

**var temp = a[0];**

**for (var i = 0; i < a.length; i++){**

**a[i] = a[i+1];**

**}**

**a[a.length - 1] = temp;**

**};**

**(function(a, k){**

**for(var i = 0; i < k; i++){**

**rotateArrayOnce(a);**

**}**

**console.log(a);**

**})(a, k);**

**});**

**INPUT:**

**2**

**1 2 3 4 5**

**OUTPUT:**

**[3,4,5,1,2]**

**8. Return median of two sorted arrays of same size**

**AN0NYMOUS FUNCTION METHOD**

**// Getting input via STDIN**

**const readline = require("readline");**

**const inp = readline.createInterface({**

**input: process.stdin**

**});**

**const userInput = [];**

**inp.on("line", (data) => {**

**userInput.push(data);**

**});**

**inp.on("close", () => {**

**function mergeSort(arr1, arr2){**

**let result = [];**

**let i = 0;**

**let j = 0;**

**while(i < arr1.length && j < arr2.length){**

**// compare the elements one at a time.**

**if(arr1[i] > arr2[j]) {**

**result.push(arr2[j]);**

**j++;**

**} else {**

**result.push(arr1[i]);**

**i++;**

**}**

**}**

**while(i < arr1.length){**

**result.push(arr1[i]);**

**i++;**

**}**

**while(j < arr2.length){**

**result.push(arr2[j]);**

**j++;**

**}**

**return result;**

**}**

**var calculateMedian = function(arr){**

**var n = arr.length;**

**if (n === 0)**

**return -1;**

**if (n%2 === 0)**

**return (parseInt)(arr[n/2] + arr[n/2-1])/2;**

**return arr[n/2];**

**}**

**var a = userInput[0].split(" ").map(function(val){**

**return Number(val);**

**});**

**var b = userInput[1].split(" ").map(function(val){**

**return Number(val);**

**});**

**a.sort(function(n1, n2){return n1 - n2});**

**b.sort(function(n1, n2){return n1 - n2});**

**var mergedArray = mergeSort(a,b);**

**var median = calculateMedian(mergedArray);**

**console.log(median);**

**});**

**INPUT:**

**10 11 9 8 15**

**12 23 13 50 5**

**OUTPUT:**

**11.5**

**IIFE METHOD:**

**// Getting input via STDIN**

**const readline = require("readline");**

**const inp = readline.createInterface({**

**input: process.stdin**

**});**

**const userInput = [];**

**inp.on("line", (data) => {**

**userInput.push(data);**

**});**

**inp.on("close", () => {**

**var a = userInput[0].split(" ").map(function(val){**

**return Number(val);**

**});**

**var b = userInput[1].split(" ").map(function(val){**

**return Number(val);**

**});**

**a.sort(function(n1, n2){return n1 - n2});**

**b.sort(function(n1, n2){return n1 - n2});**

**var mergedArray = mergeSort(a,b);**

**function mergeSort(arr1, arr2){**

**let result = [];**

**let i = 0;**

**let j = 0;**

**while(i < arr1.length && j < arr2.length){**

**// compare the elements one at a time.**

**if(arr1[i] > arr2[j]) {**

**result.push(arr2[j]);**

**j++;**

**} else {**

**result.push(arr1[i]);**

**i++;**

**}**

**}**

**while(i < arr1.length){**

**result.push(arr1[i]);**

**i++;**

**}**

**while(j < arr2.length){**

**result.push(arr2[j]);**

**j++;**

**}**

**return result;**

**}**

**(function(arr){**

**var median;**

**var n = arr.length;**

**if (n === 0)**

**median = -1;**

**if (n%2 === 0)**

**median = (arr[n/2.0] + arr[n/2-1])/2.0;**

**median = (arr[n/2.0]);**

**console.log(median);**

**})(mergedArray);**

**});**

**INPUT:**

**10 11 9 8 15**

**12 23 13 50 5**

**OUTPUT:**

**11.5**

*Generate an API key with given link below https://openweathermap.org/guide Print the current weather data in console- By lat lang*

var request = new XMLHttpRequest();

request.open('GET','https://restcountries.eu/rest/v2/all',true);

request.send();

request.onload = function(){

    var countryData = JSON.parse(this.response);

    for(let ele of countryData){

         //console.log(ele.name + ":" + ele.latlng[0] + ele.latlng[1]);

         var url = 'http://api.openweathermap.org/data/2.5/weather?q=' + ele.name + '&units=metric&&appid=be9c5bf3107b9ea80c938aab05af6344';

        var request2 = new XMLHttpRequest();

         request2.open('GET', url, true);

        request2.send();

        request2.onload  = function(){

            var weatherData = JSON.parse(this.response);

            if ( request2.status != 404) {

        console.log(ele.name + ":" + weatherData.coord.lat + " " + weatherData.coord.lon);

        //console.log(request2.status);

            }

    }

}

}