                                       // REVISIONS

                                   // Révision1

let arrExample = [2, 10, 5, 89, 58, 9, 25, 56, 118]

let readFirstElement = true;

let index;

if (readFirstElement) {

    index = 2

} else {

    index = 4

}

console.log (arrExample[2])

                                    // Révision2

let objExample = {

    prop1: 6,

    prop2: 3,

}

console.log (objExample.prop1)

                                    // Révision 3: exercice qui débouche sur un cours d'imbrication de tableaux et d'objets

// A l'aide de l'objet précéedent, construire

// la chaine de caractère ==> Je m'appelle Paul Doazan, j'ai 35 ans et j'habite à Talence.

let identity1 = {

    firstname1:'Paul',

    lastname1: 'Doazen',

    age1: 35,

    town1:'Talence'

}

let resultFromObject = `Je m'appelle ${identity1.firstname1} ${identity1.lastname1} , j'ai ${identity1.age1}  et j'habite ${identity1.town1}`

console.log (resultFromObject)

// Pour ceux qui ont terminé, créer un deuxième objet identity2,

// stocker identity et identity 2 dans un tableau,

// et écrire la phrse précédente en utilisant le tabaleau.

let identity2 = {

    firstname2:'Régis',

    lastname2: 'Jouan',

    age2: 54,

    town2:'Bisca'

}

let arrIdentities = [identity1, identity2]

console.log(arrIdentities[0].firstname1)

                                                        // Cours 2 sur .length

// recupére le dernier éléments du tableau

console.log(arrExample[arrExample.length - 1])

                                               // Exercice d'application qui permet de chercher la valeur du milieu d'un tableau

 Récupérer l'élement du tableau placé au milieu du tableau à l'ai de la propriété .length du tableau

let arrExample = [3, 5, 8, 12, 2, 43, 21, 2, 3, 42, 2, 5, 12, 555, 2, 43, 21, 2, 42, 2, 5, 12, 434]

let middleIndex;

if(arrExample.length % 2 == 0) {

    middleIndex = arrExample.length / 2

} else {

    middleIndex = (arrExample.length - 1) /2

}

let elementInTheMiddle = arrExample[middleIndex]

console.log(elementInTheMiddle)

                                               //  autre possibilité pour l'exercice précedent

let middleIndex = Math.round(arrExample.lengh / 2); // et voilà ... rien besoin d'autre

console.log(elementInTheMiddle)

let arrExample = [3, 5, 8, 12, 2, 43, 21, 2, 3, 42, 2, 5, 12, 555, 2, 43, 21, 2, 42, 2, 5, 12, 434]

let middleIndex;

if(arrExample.length % 2 == 0) {

    middleIndex = arrExample.length / 2

} else {

    middleIndex = (arrExample.length - 1) /2

}

let elementInTheMiddle = arrExample[middleIndex]

console.log(elementInTheMiddle)

                            // ajout d'un élement au tableau avec "PUSH" ou "POP" qui retire le dernier élément du tableau

arrExample.push(51, 21, 51)

arrExample.pop()

console.log(arrExample)

                               // Exercice d'application de PUSH

let administrators =['Paul','Pierre', 'Mathilde']

ajouter Marianne dans le tableau administrators

let newAdmin1 = 'Marianne'

administrators.push(newAdmin1)

let newAdmin2 = 'Sophie'

administrators.push(newAdmin2)

let newAdmin3 = 'Lucie'

administrators.push(newAdmin3)

console.log(administrators)

                                               //  nouvel exercice

avant chaque ajout d'une administrateur on teste si le nombre d'administrateur est supperieur ou egal à 4

si c'est le cas on enlève le dernier élément on enlève le dernier léméent pour ajouter le  nouvel administrateur.

Si ce n'est pas le case, ajouter simplement le nouvel adminsitrateur

let administrators =['Paul','Pierre', 'Mathilde']

let newAdmin1 = 'Marianne'

let newAdmin2 = 'Sophie'

let newAdmin3 = 'Lucie'

        // première solution

if (administrators.length >= 4) {

    administrators.pop()

    administrators.push(newAdmin1)

}

administrators.push(newAdmin1)

console.log(administrators)

       // secondes solutions optimisées

if (administrators.length >= 4) {

    administrators.pop()

}

administrators.push(newAdmin2)

console.log(administrators)

if (administrators.length >= 4) {

    administrators.pop()

}

administrators.push(newAdmin3)

console.log(administrators)

                                       // Nouvel exercice sur la base du précédent

                                       // ajouter utilisateur si la propriété est true

let administrators =['Paul','Pierre', 'Mathilde']

let newAdmin1 =  {

    name: 'Marianne',

    admin: false,

}

let newAdmin2 =  {

    name: 'Sophie',

    admin: false,

}

let newAdmin3 =  {

    name: 'Lucie',

    admin: true,

}

if(newAdmin1.admin) {

    administrators.push(newAdmin1.name)

}

if(newAdmin2.admin) {

    administrators.push(newAdmin2.name)

}

if(newAdmin3.admin) {

    administrators.push(newAdmin3.name)

}

console.log(administrators)

             // exercice précédent remodelé avec une function ! super important

let administrators =['Paul','Pierre', 'Mathilde']

let newAdmin1 =  {

    name: 'Marianne',

    admin: false,

}

let newAdmin2 =  {

    name: 'Sophie',

    admin: false,

}

let newAdmin3 =  {

    name: 'Lucie',

    admin: true,

}

addUserIfAdmin(newAdmin1)

addUserIfAdmin(newAdmin2)

addUserIfAdmin(newAdmin3)

function addUserIfAdmin(user) {

    if(user.admin) {

        administrators.push(user.name)

    }

}

console.log(administrators)

                   // Nouvel exercice : Créer une fonction qui prend en paramètre un objet utilisateur et qui affiche en console "Bonjour $(nom) !"

                   // Exemple "Bonjour Lucie !"

 let newAdmin2 =  {

        name: 'Sophie',

        admin: false,

}

displayName(newAdmin2)

function displayName(param) {

         console.log(`Bonjour,  ${param.name} !`)

}

               // Ajouter une condition pour afficher Bonjour administratrice ... !

               // Si la propriété est true

let newAdmin1 =  {

    name: 'Marianne',

    admin: false,

}

let newAdmin2 =  {

    name: 'Sophie',

    admin: true,

}

let newAdmin3 =  {

    name: 'Lucie',

    admin: true,

}

displayName(newAdmin1)

displayName(newAdmin2)

displayName(newAdmin3)

function displayName(param) {

    if (param.admin) {

        console.log(`Bonjour,  ${param.name} !`)

    }   else {

        console.log (`${param.name} , tu n'as pas l'autorisation necessaire`)

    }

}

                           // autre méthode de l'exercice précédent

let newAdmin1 =  {

    name: 'Marianne',

    admin: false,

}

let newAdmin2 =  {

    name: 'Sophie',

    admin: true,

}

let newAdmin3 =  {

    name: 'Lucie',

    admin: true,

}

displayName(newAdmin1)

displayName(newAdmin2)

displayName(newAdmin3)

function displayName(param) {

    let strEsult = `Bonjour `

    if (param.admin) {

        strEsult += ' adminsitratrice '

    }

    strEsult += param.name + `!`

    console.log(strEsult)

}

                            //  Créer une fonction à deux paramètres qui affiche Bonjour Sophie et Lucie

let newAdmin1 =  {

    name: 'Marianne',

    admin: false,

}

let newAdmin2 =  {

    name: 'Sophie',

    admin: true,

}

let newAdmin3 =  {

    name: 'Lucie',

    admin: true,

}

buildSentence(newAdmin2, newAdmin3)

function buildSentence(user1, user2) {

        console.log(`bonjour, ${user1.name} et ${user2.name} !`)

}

                                     // Nouvel exercice : objet, clés et une valeurs

const town1 = {

    name:'Bordeaux',

    population: 300000,

    superficy:50

}

const town2 = {

    name:'Lyon',

    population: 500000,

    superficy:48

}

const town3 = {

    name:'Paris',

    population: 2000000,

    superficy:105

}

                             //// Ecrire une fonction qui prend deux paramètres et qui affiche dans la console le nom de la ville la plus peuplée des deux

compareTowns(town1,town2)

function compareTowns(popu1, popu2 ) {

    if(popu1.population > popu2.population) {

        console.log(popu1.name);          // afficher le nom de la ville la plus peuplée

    } else {

        console.log (popu2.name)

    }

}

                                // // autre méthode avec return qui meyt fin à la fonction

const biggerTown =  compareTowns(town1,town2)

                                function compareTowns(popu1, popu2 ) {

                                    if(popu1.population > popu2.population) {

                                        return popu1          // afficher le nom de la ville la plus peuplée

                                    } else {

                                        return popu2

                                    }                     // met fin à la fonction ...

                                }

                                // ecrire la fonction qui retourne la somme de 2 paramètres

const nb1 = 5

const nb2 = 7

const sum = addTwoParameters(nb1, nb2)

console.log ('Le résultat de la somme de ${nb1} et ${nb2} est égal à ${sum}')

function addTwoParameters(a, b) {

        const result = a+b

        return result

}

                            // Ecrire une fonction qui retourne le produit de deux paramètres

const resulProduct = productOfParameters (4, 6)

console.log(resulProduct)

function productOfParameters(a, b) {

  const result = a\*b

  return result

}

// Nouvel exercice : variante avec exercice avec les towns: Ecrire la fonction qui compare

// 2 villes en paramètres et qui me retourne celle qui a le plus jumelages.

const town1 = {

    name:'Bordeaux',

    population: 300000,

    superficy:50,

    twinnings: ['Berlin', `Londres`,'Madrid', 'Libourne' ]

}

const town2 = {

    name:'Lyon',

    population: 500000,

    superficy: 48,

    twinnings: ['Berlin', `Londres`,'Madrid', 'Libourne', 'Libourne', 'Libourne' ]

}

const town3 = {

    name:'Paris',

    population: 2000000,

    superficy:105,

    twinnings: ['Berlin', `Londres`,'Madrid', 'Rochefort' ]

}