🔥🔥🔥🔥🔥 Javascript Core 🔥 🔥🔥🔥🔥

🔥🔥🔥🔥🔥 SESSION 2 🔥🔥🔥🔥🔥

```
// 🔥 console yazdırma
/* Python da print(), javascript te;
document.write()
window.alert()
console.log() */
console.log("Hello World!");
console.log("I'm from file");
var x = 5;
console.log(x);

var a= "Global";
console.log(a);
{
    let a = "scope"
    console.log(a);
    {
        let a = "inner scope"
        console.log(a);
    }
    console.log(a);
}

console.log(a);

var c=3;
console.log(c);
var c = 5;
console.log(c);

/* let d = 3;
console.log(d);
let d = 5;
console.log(d); */
//yukarıdaki kullanımda hata veriyor.doğrusu;
let d = 3;
console.log(d);
d = 5;
console.log(d);

const h1 = "constant variable";
console.log(h1);
//h = 2; dersek hata veriyor çünkü const değişmez
```

```
console.log(f);//sadece bu satrırı yazarsak not defined tanımı hatası verir
var f; //diyerek üstteki hatayı ortadan kaldırıyoruz bu sefer undefined çıktısı vercek
// hoisting olayı gerçekleşmiş oluyor
/* console.log(e);
let e = 5; *///let ile yada const ile tanımlayınca var daki gibi olmaz
hata vermeye devam eder.

//🔥🔥 Data Types

var g;
console.log(typeof g);// undefined

var aa = 5;
console.log(typeof aa);
console.log(typeof 5.5);
console.log(typeof 0);
console.log(typeof 1/0);//NaN -> not a number
console.log(typeof NaN);//number
console.log(typeof (1/0)); //number işlmlerde prantez içine almak lazım
console.log(typeof typeof 5.5);//string çünkü typeof 5.5 in çıktısı number olduğu için
console.log(1/0);
console.log(typeof Infinity);//number

//🔥🔥🔥     string
var merhaba = "Merhaba";
var selam = "Sanada selam";
var instructor = 'Mark'
var myStr= `Merhaba ${instructor}`//backtickle yazılması gerekir bu gibi kullanımda.
aslında pythondaki format metodunun kullanım mantığı ile aynı sadece syntaxleri farklı
console.log(myStr);
console.log(`${2+3}`);
console.log(Boolean(""));//false
console.log(Boolean(" "));// true
console.log(Boolean("0"));//true
console.log(Boolean(0));//false
var s;
console.log(Boolean(s));//false undefined olduğu için
console.log(typeof s);//undefined
console.log(typeof null);//object
console.log(Boolean(null));//false undefined
console.log(Boolean(null) == false);//true
console.log(null == false);//false
console.log(null == undefined);//true
// ===
console.log(null === undefined);//false
console.log(2 == "2");//true burada js yorum yaparak number yazdığımızı düşünüp evet
eşit olabilir diyor
console.log(2 === "2");//false çünkü tamamen eşit mi diye soruyor
console.log(2 + "2");//22
console.log(2 + 2);//4
console.log(2 + 2.0);//4
```

🔥🔥🔥🔥🔥 SESSION 3 🔥🔥🔥🔥🔥

```
//🔥js de number işlemlerinde maksimum çıktı alınabilecek değer ve minimum değer vardır.
bunlarıda aşağıdaki şekilde sorgulayabiliriz:
console.log(Number.MAX_SAFE_INTEGER);//maksimum çalışcak dğer
console.log(Number.MIN_SAFE_INTEGER);
//🔥 JS de noktalı sayılar ile işlemler
var x = 0.1 + 0.2;
console.log(x);//0.30000000000000004
console.log(x.toFixed(5));//0.30000 noktadan sonra 5 değer getir
console.log(typeof x.toFixed(5));//string çıktı veriyor
console.log(+x.toFixed(5));// başındaki + numbera çeviriyor.ve noktadan sonraki sıfırları
 çıktı vermez.0.3 tür
console.log(15+25);//40
console.log(015+025);//8li sayı sisitemine göre işlem yapar çıktı 34 olur.octal
// 015 : 13 e 025 : 21 e eşit
console.log(08 + 08);//16 çünkü 8 ve 8den büyük sayılar olursa decimal olarak görür
var a=2;
var b = a;
console.log(a,b);//primitive tipte olduğu için a değişince b değişmez.primitiv tipte
veriler kopyalanır
var a = 3;
console.log(a,b);// b değişmedi.nesne tipinde veriler link ile irtibatlanır
console.log("araba"+3);//pythontadki gibi değil stringe yapıştıryor:araba3
console.log(null+3);//3
var a = 3;
var b = a--
console.log(b);//3 işlme önceliğinden
console.log(null==null);//true
console.log(NaN==NaN);// false

//🔥🔥🔥       Operators
var x = 2;
var y= 5;
var z = 7;
var t = "11";
var c = 55;
console.log(x+y+z);
console.log(t - x);//pythondaki gibi hata vermiyor string olan t number gibi davranır
console.log(t*y);// number gibi davranır
console.log(t/x);
console.log(c/y);
console.log(c%7);//kalanı verir
console.log(5/0);// Infinity çıktı verir
console.log(typeof Infinity);// number
console.log(5**2);//25
console.log(100+50*3);// 250,işlem önceliği için:
//https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Operator_
Precedence
var b = 5;
console.log(b);
console.log(++b);//6
```

```
console.log(--b);//5
var h = 8;
console.log(h--);//çıktı 8
console.log(h);//7öncelikten kaynaklı
console.log(--h);//6

/* var input= +prompt("enter age")//+ ile number tipine çevirdik
console.log(input);
console.log(input, typeof input); */
var g = 8;
console.log(-g);// -8
var a = 4;
var b = "4";
var c = a == b;
console.log(c);//true
console.log(a === b);//false === hem değeri hem tipi eşit mi

console.log(a!=b);//false
console.log(a !== b);//true

var a = "a";
var b = "b";
console.log(a>b);//false ascii koda göre
console.log(a<b);//true
console.log(a.charCodeAt());//97.ascikodu verir
console.log(b.charCodeAt());//98
console.log("2".charCodeAt());//50

//  🔥🔥🔥           Logical Operators
/*and = %%
  or = ||
  not = !
*/
var  a = (true || false);
console.log(a);true
console.log(2 && 5);// 5 i ddöndürür
console.log(null && NaN);//null

var capitalletter = true;
var symbol = true;
var passlength = false;
var result = capitalletter && symbol && passlength;
console.log(result || "Try again");
//🔥        ??
// uzun uzun if statement ile kontrol yapmaktansa böyle kısa bir yöntem geliştirilmiş
var k;

console.log(k ?? "Null");//çıktı:null
var b = null;
console.log(b ?? k);// çıktı:undefined. null olmayanı çıktı verir ama yukarıdaki string
olduğu için "null" çıktısı verdi
// ilk değer null sa diğerini çıktı verir.ilk değer null değilse ikinciyi çıktı verir.
```

🔥🔥🔥🔥🔥 SESSION 4 🔥🔥🔥🔥🔥

```
//🔥🔥🔥   Conditionals
let score = 51;
if (score >= 50) {
    console.log ("Tebrikler, geçtiniz");
}
if (score >= 50)
    console.log ("Tebrikler, geçtiniz");
// tek satırda da yazılabiliyor.şartı tek bir statement varsa
let grade = score >= 52
if (grade) console.log("Tebrikler geçtiniz");
/* let scor = prompt("Notunuzu giriniz : ")
let grades = scor >= 52
if (grades) console.log(`Tebrikler, notunuz ${scor} geçtiniz`); */

if (grade){
    console.log(`Tebrikler notunuz ${score} geçtiniz`);
} else {
    console.log(`üzgünüm kaldınız`);
}
if (score>80){
    console.log("Tebrikler çok başarılısınız");
} else if (score >= 50) {
    console.log("Tebrikler geçtiniz");
}else {
    console.log(`üzgünüm kaldınız`);
}
// 🔥🔥 nested if
if (score >= 50) {
    if (score>80){
        console.log("Tebrikler çok başarılısınız");
    } else {
        console.log("Tebrikler geçtiniz");
    }
}else{
    console.log("üzgünüm kaldınız")
}
//🔥🔥  Ternary if
// syntaxi => condition ? true(şartı sağlıyorsa burası çıktı vercek) : false(şartı
 sağlamıyorsa burası çıktı vercek yani else kısmı)
// önemli bir yapı genel olarak tek satırlık if else yapısı yerine tercih edilir
var scores = 49;
scores>=50 ? console.log("Tebrikler geçtiniz") :
console.log("üzgünüm kaldınız(ternary if yapısıyla yapıldı)");
// 🔥🔥 switch case yapısı
// bu da önemli bir yapı if-elseif-else yapısı yerine kullanılabilir
// syntaxi if-elseif-else e göre daha kolaydır
var text;
var fruits = "Banana";
switch (fruits) {
    case "Banana":
```

```
        text = "Banana is good"
        break;
    case "Orange":
        text = "I am not a fan of orange.";
        break;
    case "Apple":
        text = "How you like them aplles?";
        break;
    default:
        text = "I have never heard of that fruit...";
        break;
}
console.log(text);
// her case den sonra break koymak gerekir. yoksa diğer caselere geçmeye devam eder
default kısmını bile çalıştır
// bu durumda da kodumuzdan istediğimiz çıktıyı alamayabiliriz
var text;
var fruits = "APPLE";
switch (fruits.toLowerCase()) {
    case "banana":
        text = "Banana is good"
        break;
    case "orange":
        text = "I am not a fan of orange.";
        break;
    case "apple":
        text = "How you like them apples?";
        break;
    default:
        text = "I have never heard of that fruit...";
        break;
}
console.log(text);
//.toLowerCase ya da .toUpperCase string değerimizi küçük ve büyük harf yapar.
// js case sensitive olduğu için koşul ifadelerimizde yada diğer işlemlerimizde
bu metodları kullanabiliriz.
/* var text;
var fruits = prompt("Enter your favorite fruit");
switch (fruits.toLowerCase()) {
    case "banana":
        text = "Banana is good"
        break;
    case "lime":
    case "orange":
        text = `I am not a fan of ${fruits.toLowerCase()}.`;
        break;
    case "apple":
        text = "How you like them aplles?";
        break;
    default:
        text = "I have never heard of that fruit...";
        break;
}
console.log(text);
```

```
// bu örnekte olduğu gibi case yapısını peşpeşe kullanabiliriz.yani diyoruz ki
caselerde belirlediğimiz şartlarımızdan hangisi gelirse gelsin sen aynı şeyi döndür.
// ${fruits.toLowerCase()} bu yapı ile de hangi case gerçekleştiyse onun değerini
bu şekilde çıktımıza monte edebiliriz

*/

//🔥 güne göre o gün programda ne olduğunu gösteren switch case yapısı:
/* var text;
var days = prompt("Enter day");
switch (days.toLowerCase()) {
    case "monday":
    case "wednesday":
    case "thursday":
    case "saturday":
        text = "In class";
        break;
    case "tuesday":
    case "friday":
        text = "Teamwork";
        break;
    case "sunday":
        text = "Holiday";
        break;
    default:
        text = "enter wrong day"
        break;
}
console.log(text); */

//🔥 leap year olup olmadığına göre şubat ayının gün sayısını bulma
var year = 2152;
var month = 2;
var dayCount;
switch (month) {
    case 1:
    case 3:
    case 5:
    case 7:
    case 8:
    case 10:
    case 12:
        dayCount = 31;
        break;
    case 4:
    case 6:
    case 9:
    case 11:
        dayCount = 30;
        break;
    case 2:
        if (((year % 4 == 0) && !(year % 100 == 0))
            || (year % 400 == 0))
            dayCount = 29;
```

```
        else
            dayCount = 28;
        break;
    default:
        dayCount = -1; // invalid month
}
console.log(dayCount); // 29

//🔥🔥🔥                  Loops

//🔥🔥 while

let i = 10;
let sum = 0;
while (i<6){
    sum += i;
    i++;
}
console.log(sum); //15
// while da bulunan koşul false olana kadar döngü çalışmaya devam eder
// do while yapısında do içinde buluna her türlü başlangıçta çalışır sonra whiledeki
 koşul doğruysa çalışmaya devam eder false olana kadar
let sums = 0;
do {
    sums +=i ;
    i++;
} while (i<6);
console.log(sums,{i}); // sums 10 olark çıktı verir. while koşuluna uymadığı için durur

/* let j = +prompt ("Bir sayı giriniz : ");
console.log(isNaN(j),j)// isNaN sayı olup olmadığına bakar sayıysa flse verir */
/* while (isNaN(j)) {
    j = +prompt ("Bir sayı giriniz : ");

}
console.log("Bir sayı girdiniz") */

/* let j;
do {
    j = +prompt ("Bir sayı giriniz : ");//+ koayarak kullanıcın gireceği değerin türünü
 number yapıyoruz(pythondfaki int( mantığı))
} while (isNaN(j));
console.log(`girlen değer ${j}`) */
// userın sayı girmesini sağlamış olduk
// user harf girerse döngümüz sormaya devam edecek numara girdiğinde do çalışcak ve
 durcak

//🔥🔥    For Loop
// farklı kullanım yolları;
// i örneğinde i yi for yapısı içinde de declare edebiliriz.
// i yi for yapısı dışında da declare edebiliriz.
let k1 = 0;
for (k1 = 0; k1<10; k1++) {
    console.log(k1);
```

```
}
let jk = 0;
for (;jk<4;jk++){
    console.log(jk);
}
```

🔥🔥🔥🔥🔥 SESSION 5 🔥🔥🔥🔥🔥

```
let strOne = "clarusway";
console.log(strOne.length);//9 pythondaki len() metodu ile aynı
console.log(strOne[0]);//c indexleme python ile aynı []içinde belirtiyoruz ve 1 den
değil 0 dan başlar
// pythondan farklı olarak - indexleme yoktur son elemana length-1 diyererk erişebiliriz

for(let i=0 ; i<strOne.length; i++){
    console.log(strOne[i]);
}
for (let j = strOne.length-1 ; j >= 0; j--) {
    console.log(strOne[j]);
}// tersten yazdırma
console.log(strOne.slice(3-7));//aralıklı yazdırma
//🔥  continue
for (let i=0; i<=100;i++){
    if (i % 5 == 0) continue;
    console.log(i);
}

/* while (true) {
    let x = prompt(`bir sayı giriniz \n veya q ile çıkış yapınız`);
    if (x.toLowerCase() == `q`){
        console.log("çıkış yapıldı")
        break
    } else if(isNaN(x)){
        continue;
    } else {
        console.log(`${x}'in karesi = ${x*x}`)
    }
} */
console.log(Math.random()*6+1);//bu sonraki konularda anlaşılcaktır.built-in fonksi
yondur.0 ve 1 arasında rastgele sayı döndürür. 1 çıkmaz
console.log(Math.trunc(Math.random()*6+1));//trunc da küsuratı keser.burada +1 deme
mizin sebebi random gelcek sayı 1 olmayacağı için koyduk yani değerimiz en fazla 5,9999
 küsuratlı olabilir
// trunc ile bu küsuratı kesiyoruz ve 1 ekleyerek 6 ya ulaşabiliyoruz
// 🔥 zar atma
let xx,yy;
let count = 0;
let cift = 0;
while(true){
    xx = Math.trunc(Math.random()*6)+1;
```

```
    yy = Math.trunc(Math.random()*6)+1;
    if (xx == 6 && yy ==6) {
        count++;
        console.log("Kazandınız");
        break;
    } else if (xx == yy && xx != 6){
        cift++;
        console.log(xx,yy,`${cift}. çift zar`);
        if (cift ==3) {
            console.log("Kaybettiniz");
            break;
        }
    } else {
        console.log(xx,yy);
    }
    count++;
}
console.log(count);

//🔥🔥🔥             Functions
//function keywordü ile tanımlanır
sayHi(); // birkere  tanımlayınca ööncesinde de sonrasında da çağırır
// hoisting ten dolayı
function sayHi(){
    console.log("Hi");
}
sayHi();
let userName = "Münir";
function sayHi(name){
    console.log(`hello ${name}`);
}
sayHi(userName);
sayHi("Mark");

function sayHi2(name) {
    return ("Hello "+ name)
}
console.log(sayHi2("veli"));
// pythondan farklı olarak fonksiyonu yine return ile tanımlıyoruz ama çağırırken
 console.log içerisinde çağırıyoruz
console.log(typeof sayHi2());// string
console.log(typeof sayHi());// function tanımlarken console.logla tanımladığımız
 için undefined veriyor,pythonda da aynıydı

function add100(num1) {
    return num1+100
}
console.log(add100(23));//123
function add(num1,num2) {
    return num1+ num2
}
console.log(add(23));//NaN verir çünkü num2 yi göndermedik belirlediğimiz paramtere
 kadar argüman göndermemiz gerekir,bunu engellemek için parametrelerimze default
değer verebiliriz;
```

```
function add1(num1,num2=0) {
    return num1+ num2
}
console.log(add1(23));// num2 ye default olrak 0 verdik çıktımız 23 oldu
console.log(add1(23,25));//48
//🔥🔥 function expressions
/* console.log(square(4));// aşağıdaki örnekte function expressions ile tanımladığımız
 için burada fonksiyonumuz çalışmadı bunun amacı houstingdden kurtarmak*/
let square = function(a) {return a*a};// function expressions
console.log(square(3));//9

let adder = new Function ("a","b","return a+b");
console.log(adder(2,6));//8
console.log(typeof adder);//function

let sumAdd = function (a,b){return a+b;};
let addTwo = function (num1){
    return sumAdd(num1,2);
};// function içinde diğer bir functionu çağırabiliriz
console.log(addTwo(5));// ikinci fonskiyonda sum  fonksiyonumuzu çağırıyoruz ve çıktı 7
console.log(addTwo(7));// 9

// 🔥faktöriyel bulma
let faktör = function (num2){
    if (num2 == 0) {
        return 1
    } else {
        return num2 * faktör(num2-1);
    }
}
console.log(faktör(5));

function sayHello(name){
    name && console.log(`Hello ${name}`);
}
sayHello("deli");

//Fonksiyon kendine verilen değeri değiştirmeyecek;
function square2(num3){
    num3 *= num3;
    return num3;

}
let myNum = 4;//myNum global bir değişken
//aşağıda myNum un değerini argüman olarak kullanmış olduk
console.log(square2(myNum));//16
console.log(myNum);//4
// bu şekilde myNum değişmedi
//ama localde oluşturursak
function square3(num4){
    myNum1= num4 * num4;
    return myNum1;

}
```

```
let myNum1 = 4;
console.log(square3(myNum1));//16
console.log(myNum1);//16 oldu
//ancak function içinde yani localde let ile tanımlarsak globaldeki değişmez
function square4(num5){
    let myNum2= num5 * num5;
    return myNum2;

}
let myNum2 = 4;
console.log(square4(myNum2));//16
console.log(myNum2);//4

let student = {};
student.name = 'Mesut';
function sayHi3(student) {
    console.log(`Hello! ${student.name} from entry point`);
    student.name = 'Zeynep';
    console.log(`Hello! ${student.name} inside function`);
    student = {name : "Leon"};
    console.log(`Hello! ${student.name} inside function`);
}
sayHi3(student);
console.log(`Hello! ${student.name} from outside`);
// gloablin değiştirilmesi istenmez genel olarak
//o yüzden fonksiyon içindeki leonu tanımladığımız gibi tanımlamamız gerekiyor
// aşağıdaki şekildeki gibi tanımlayabiliriz
const student1 = { name: 'Mesut'};
function f1(st) {
   console.log(`this is ${st.name}`);
   st.name = 'Zeynep';
   console.log(`this is ${st.name}, should be zeynep?`);
}

f1(student1);
console.log(student1);
```

🔥🔥🔥🔥🔥 SESSION 6 🔥🔥🔥🔥🔥

```
let student5 = "Mustafa";
function sayHi(studentName) {
    console.log(`Welcome ${studentName}`);
}
sayHi(student5);//değişken yollayarak aslında değişkenimizn değerini yani 'Mustafa' yı
 argüman yapmış oluyoruz

let students = ["John","Jane","Joe"];//array yapısı
function sayHello(studentName) {
    console.log(`Welcome ${studentName}`);
}
```

```
for (let i = 0; i<students.length;i++){
    sayHello(students[i]);
}
function sayHi1(student){
    for(let i = 0;i < student.length;i++){
        console.log(`Welcome ${student[i]}`);
    }
}
sayHi1(students);

function sumNew(){
    let sum = 0;
    for (let i = 0; i< arguments.length; i++){
        sum += arguments[i];
    }
    return sum;
}
//🔥 arguments default olarak bulunur fonksiyonun metodu gibi.pythonda *args gibi
// yani function tanımlarken parametre belirtmezsekfonksiyonu çağırırken gönderdiğimiz
 bütün argümanlar arguments içinde toplanır, bu bir array yapısı oluşturur. örnekte
 olduğu gibi elemanlara ulaşmak için array metodları kullanılır
console.log(sumNew(1));//1
console.log(sumNew(1,2,3,4,5));//15

function sum1(a,b,...others){
    console.log(arguments);
    console.log(others);
    let sum = 0;
    for (let i = 0; i< arguments.length; i++){
        sum += arguments[i];
    }
    return sum;
}
//bu yöntemde rest operatoru kullanmış oluyoruz yani bu örnek üzerinden söyleyecek
 olursak a, b den sonrasını arraye çevirmiş oluyor
console.log(sum1(1,2,3,4));//çıktımız:
/* {0: 1, 1: 2, 2: 3, 3: 4}
[3, 4]
10 */
function sum2(a,b,...others){
    console.log(arguments);
    console.log(others);
    let sum = 0;
    for (let i = 0; i< others.length; i++){
        sum += others[i];
    }
    return sum;
}
console.log(sum2(1,2,3,4));// a ve b den sonra olanlara işlem yapıyor sadece çktı 7
const bill = function(tax,...list){
    let total = 0;
    for(let i =0; i < list.length; i++){
        total += list[i]+list[i]*tax;
    }
```

```
    return total;
}
//🔥vergi hesaplama
console.log(bill(0.18,10,15,20));

function bolme(num1,num2){
    if(num2 === 0){
        return "Zero Division Error"
    } else {
        return num1 / num2
    }
}
console.log(bolme(12,2));
console.log(bolme(12,0));
function bolme1(num1,num2){
    if(num2 === 0) return "Zero Division Error"
    return num1 / num2
}
console.log(bolme1(12,3));
console.log(bolme1(12,0));
//🔥🔥 ternary
function bolme2(num1,num2){
   return num2==0 ? "Zero Division Error" : num1/ num2
}
console.log(bolme2(12,4));
function bolme3(num1,num2){
    return num2 ? num1 / num2 : "Zero Division Error"
}
console.log(bolme3(24,4));
//🔥example
function pascalNumber1(n){
    let sum = 0;
    for(let i = 1; i < n+1; i++){
    sum +=i
    }
    return sum
}
console.log(pascalNumber1(5));//15
function pascalNumber2(n){
    return (n*(n+1)/2)
}
console.log(pascalNumber2(6));//21
function pascalNumber3(n){
    if (n===1) return 1;
    return n + pascalNumber3(n-1);
}
console.log(pascalNumber3(7));//28 recursive işlem yani fonksiyon içinde aynı fonksiyonu
 çalıştırma
// recursive işlemlerde mutlaka çıkış kapısı konulması gerekir
// faktöriyel
function faktör11(n){
    if (n===1) return 1;// çıkış kapısı
    return n * faktör11(n-1);
}
```

```
console.log(faktör11(5));//120
function faktör1(n){
    return n===1 ? 1 : n * faktör1(n-1);
}
console.log(faktör1(5));//120

//🔥🔥  arrow function

let xz = (a) => a * 5;//tek satır olduğu için paranteeze filan gerek yok
//lambda ile aynı
console.log(xz(5));// 25
let toplam = a => a + 50;
console.log(toplam(17));// 67

let iife = (function trian(num) {
    if (num === 1) return 1;
    return num + trian(num-1);
})(3);
console.log(iife);


//  🔥🔥🔥         String Methods
let str1 = "Hello ";
let str2 = "World";
let str3 = `${str1 + str2}`;
console.log(typeof str1,str2);
console.log(typeof str2,str3);
console.log(typeof str3,str3);
let str4 = new String("A string object");
console.log(str4);
//string immutabledir
//🔥🔥 concat
let str6 = "Hello ";
let str5 = "World";
let str7 = str6.concat(str5);
console.log(str7);// Hello World

//🔥🔥 charAt()
// The charAt() method returns the char value at the specified index in a string.
// index of tan farklı olarak tek argüman alır içine ve index numarasını alır
var a = 'primitive.\nlerin properti veya metodu olmaz.';
console.log(a.charAt());// boş girilirse ilk indexi getirir
console.log(a.charAt(9));// . noktayı veriyor
console.log(a.charAt(10));// boş yani \n i burada birşey var ama ne var bilmiyorum diyor
 ve boş çıktı verir
console.log(a.charAt(11));// l
// 🔥🔥  includes
// belirtilen ifadenin variable içinde olup olmadığını kontrol eder. varsa true yoksa
false
// case sensitivedir
var str = "Lorem Ipsum is simply dummy text of the printing and typesetting industry.";
var n =str.includes("simply");
var n1 =str.includes("Simply");
console.log(n);//true
```

```
console.log(n1);//false

// 🔥🔥  indexOf
// belitilen ifadenin indexini döndürür
var n2 = str.indexOf("simply");//bulduğu ilk yerde bu örnek üzerinden konuşacak olursak
 s nin indexini verir
console.log(str.indexOf("o", 2));
```

🔥🔥🔥🔥🔥 SESSION 7 🔥🔥🔥🔥🔥

```
//🔥🔥 String Replace
var s = "Lorem Ipsum is simply dummy text of the printing and 'printing' typesetting
industry.";
console.log(s.replace("dummy", "hello"));
// orjinal variables değişmez
//replace metodu case sensitivedir.bunu önlemek için /i metodu kullanırız
console.log(s.replace(/Dummy/i,"oldu"));//regex yöntemi
// bütün harfler içinse
console.log(s.replace(/e/g,"a"));//regex yöntemi
//🔥🔥 Search
console.log(s.search("text"));// ilk başlangıç indeksini döndürür.
console.log(s.search("araba"));//olmayan birşey olursa -1 döndürür
// 2.argümanı almaz

//🔥🔥 Slice
console.log(s.slice(0,5));// balşagıçtan bitiş-1 e kadar kesti
console.log(s.slice(10,-1));// başlangıç 10 belirtielene kadar buradaki örnekte sondaki
 karakter çıkmayacak
console.log(s.slice(-31,-10));//printing' typesetting
console.log(s.slice(-31,76));//printing' typesetting
console.log(s.slice(54,76));//printing' typesetting

//🔥🔥 Split
// içine belirtilen argümana göre metini böler.ve array döndürür
console.log(s.split(''));//tek tek harflerer göre bulur
console.log(s.split(' '));//boşluklara göre böler
console.log(s.split());//metini tek eleman yaapar

//🔥🔥 substr (şuan artık kullanılmıyor)
// başlangıçtan başla belirtilen karakter kadar al
console.log(s.substr(26,10));// y text of

//🔥🔥 substring
// başlangıçtan başla sondan -1e kadar al
console.log(s.substring(22,33));//dummy text
console.log(s.substring(33,22));//dummy text
console.log(s.substring(33,-22));//Lorem Ipsum is simply dummy text
// negatif sayıyı 0 olarak kabul ediyor
// yani negatif index almıyor
console.log("     Arrays");
```

```
// 🔥🔥🔥             Arrays
// [] içine alarak veya new Array metodu ile oluşturabiliriz
var cars = ["Opel", "Audi", "BMW"];
console.log(cars);
var cars2 = Array.of("Opel", "Audi", "BMW")
console.log(cars2);
var cars3 = new Array("Opel", "Audi", "BMW")
console.log(cars3);
var num1 = new Array(2,10)
console.log(num1);
var num2 = new Array(10)
console.log(num2);
// const cars =["Opel","Audi",[1,2,true],"BMW"]
cars[4] = 'Porsche'
console.log(cars);
cars = []
console.log(cars);
cars[cars.length] = 'Ferrari'
console.log(cars[8]);
const cars1 = ["Opel", ,"Audi",,,,"BMW"];
console.log(typeof cars1);//object
// variable ın array olup olmadığını aşağıdaki gibi kontrol edebiliriz:
console.log(cars1 instanceof Array);//true
console.log(Array.isArray(cars1));//true
//length
const fruits1 = ["Banana", "Orange", "Apple"];
// fruits[6] = "Lemon";

console.log(fruits1);
console.log(fruits1.length);

var a = ['dog','lion','hen'];
a[100] = 'horse';
// js burada bu atamayı yapmaak için 100. indexe kadar undefined olarak eleman
ataması yapar
// 100.indexe gelince de atamayı gerçekleştirir
const vegetables = ['Broccoli','Celery','Parsley','Artichoke']

console.log(fruits1.concat(vegetables));

//🔥🔥 arrayi stringe çevirme yöntemleri;
console.log(""+fruits1);
console.log(fruits1.toString());

// 🔥🔥  Sort
const daltones = ['Joe','Jack','William','Avarel'];
console.log(daltones.sort());// ['Avarel', 'Jack', 'Joe', 'William']
const num5 = [40,100,1,5,25,10];
console.log(num5.sort());// [1, 10, 100, 25, 40, 5] ilk karaktere göre sıralama yapıyor.
istenilen bir durum değil
//bunu engellemek için sortun içine fonksiyon tanımlarız
console.log(num5.sort((a,b)=> a-b));//[1, 5, 10, 25, 40, 100]
// burdaki fonksiyon da sonuç negatif çıkarsa b nin pozisyonunu algılıyor
```

```
console.log(num5.sort((a,b)=> b-a));// bu da büyükten küçüğe
console.log(daltones.reverse());//
console.log(daltones);

//🔥🔥 push() ve pop()
// push arrayin sonua ekler,pop ta arrayin sonundan siler,içlerine argüman almaz
// pop sildiğini döner
// pythondan append ve pop ile aynı
fruits1.push("pear");
console.log(fruits);
fruits1.push("mango","pineapple",2)
console.log(fruits);//['Banana', 'Orange', 'Apple', 'pear', 'mango', 'pineapple', 2]
fruits1.pop();
console.log(fruits1);//['Banana', 'Orange', 'Apple', 'pear', 'mango', 'pineapple']
console.log(fruits1.pop());//pineapple bu şekilde kullanırsak sildiğini verir
// yani bu gibi metotlarda bu şekilde ataamaa yapabiliriz

//🔥🔥 Shift and unshift
// shift sildiğini döndürür.silinen elemanı herhangi bir değişkene tanımlama imkanı
verir bize pop gibi
// shift baştan siler unshift başa ekler
const deleted = fruits1.shift();
console.log(deleted);//Banana baştan sildi
console.log(fruits1);
const uns = fruits1.unshift("Chery");// burada örnekten yola çıkacak olursak fruits
arrayinin eleman sayısını döndürmüş oldu
console.log(uns);//5
console.log(fruits1);//['Chery', 'Orange', 'Apple', 'pear', 'mango']

//🔥🔥 splice
// sildiğini döndürür
var names = ["John","Edward","Victor"];
var deleted1 = names.splice(2,20,"Mark","James");
// bu örnekte 2.indexe git 20tane sil(olup olmaması etkilemez) ve 2.index ve sonrasında
 şunları ekle
console.log({deleted1});// {} object gösterimi için kullaınılıyor:{deleted1: Array(1)}
(1elemanlı array demek)
console.log(deleted1);// ['Victor']
console.log(names);//['John', 'Edward', 'Mark', 'James']
names.splice(1,1,"Mahmut","Jamal");
console.log(names);//['John', 'Mahmut', 'Jamal', 'Mark', 'James']
//splice da araay değişir

//🔥🔥 slice
// orjinal array değişmez
const months = ['Jan', 'March', 'April', 'June',"july"];
const springs1 = months.slice(1,3);
const springs2 = months.slice(-4,-2);
const springs3 = months.slice(1,-2);
console.log(springs1);//(2) ['March', 'April']
console.log(springs2);//(2) ['March', 'April']
console.log(springs3);//(2) ['March', 'April']
const springs4 = months.slice(2,-7);
console.log(springs4);// bulamazsa boş array döndürür []
```

```
//🔥🔥 indexOf ve lastIndexOf
//başlangıç parametresi alabilir ve dahil eder
const colors2 = ["Red", "Yellow", "Green", "Blue", "Pink", "Green"];
const x5 = colors2.indexOf("Green", 3); // 3ten başla sağa git indexi ver
console.log(x5);//5

const colors1 = ["Red", "Yellow", "Green", "Blue", "Green","Red","Yellow","Blue"];
const last = colors1.lastIndexOf("Blue")
console.log(last)//7
const last1 = colors1.lastIndexOf("Blue",6)//yani 6dan başla sola git ve index i ver
console.log(last1)//3
```