**Javascript nedir?**

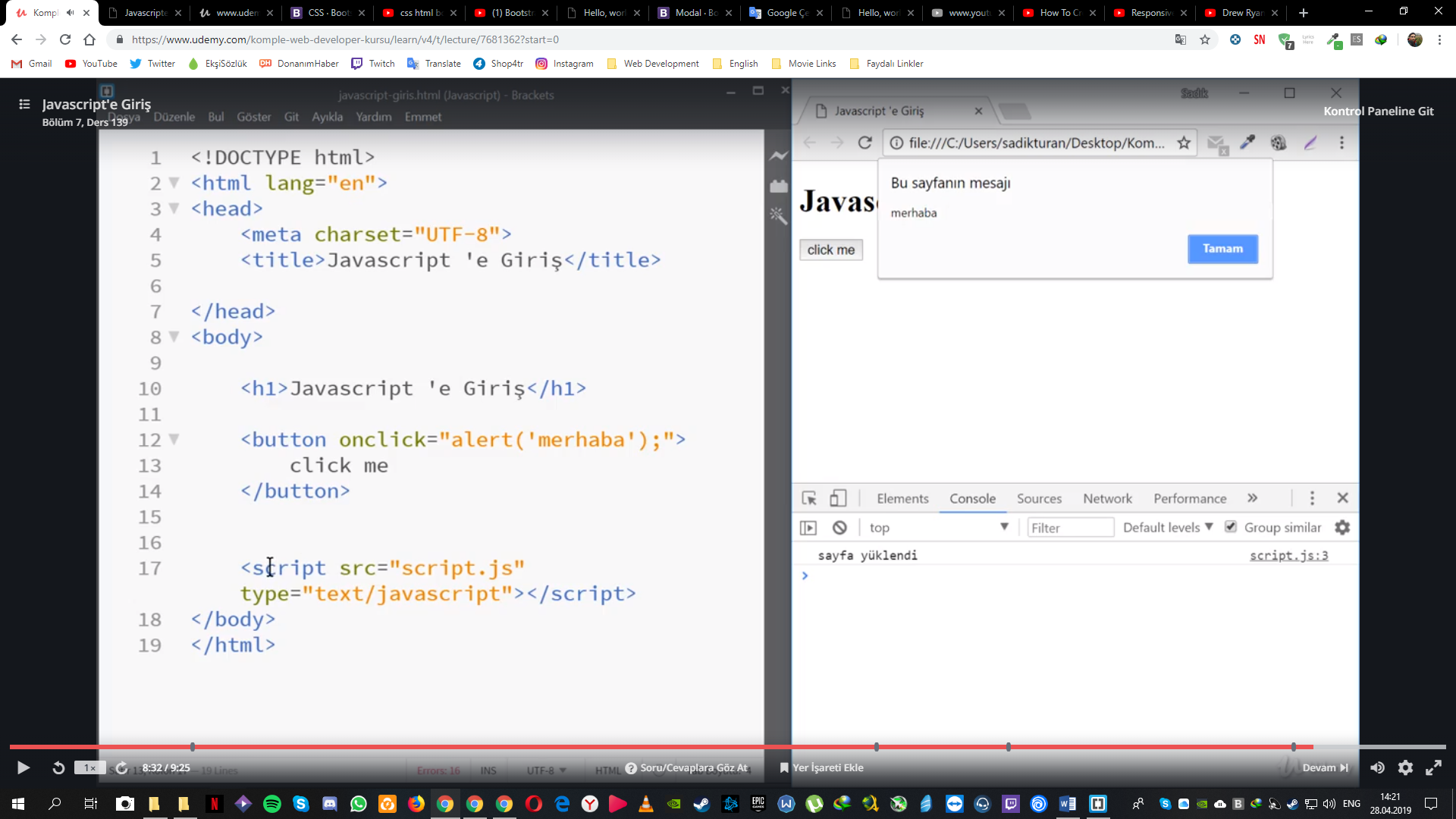
* HTML ile verilen etiketlerin CSS’le şekilelndirilmesi yapılırken. Javascript ile hazırlanan bu yapılara anismasyonlar verme, açıp kapatma yok etme gibi özellikler veriyoruz. Kısaca bu 3’üde FRONT END kısımdır web tasarımda. Örneğin;
  + Bir butona tıkladığımızda HTML ve CSS ile oluşturulmuş bir içeriğin gösterilip, tekrar tıkladığımızda kapatılması Javascript ile alakalı bir durumdur. BOOTSTRAO’teki modal kutuları buna bir örnektir. Aslında Javascript barındırırlar.
* Javascript Lightweight yani çalıştığı ortamı yormayan bir platformdur.
* Javascript cross platformdur. Yani birden fazla ortamda çalışabilir. Windows Linux IOS Android gibi.
* Javascript ayrıca object oriented language’dir. Yani Objeye nesneye yönelik bir programlama dilidir.
* Javascript artık Node.js ile backend’de de kullanılabiliyor. Yani ASP.NET ile yapılan herşey artık Node.js ilede yapılıyor. Ve Node.js içinde kullanılan tek dil Javascript’tir.
* Javascript kullanarak modern web siteleri yapabiliriz.
* Javascript ile mobil uygulamalardaki gibi hızlı uygulamalar yapabiliriz. AngularJs React tarzı uygulamalaral.

**Javascript Giriş**

<!—Javascrit--> : Bu sekilde yorum satirlari olustururuz.

Script.js : Birden fazla sayfayi ilgilendiren script kodlari, soldaki gibi ayri bir dosya olusturularak icine alinirlar. Html sayfalarinin icerisinde olusturdugumuz **<script>** etiketlerini olusturmamiza bu dosya icinde gerek yoktur. Html icinde bu **script.js**`yi yine body`de script etiketinin yazildigi yerde **<script src=”script.js” type=”text/javascript”></script>** yazarak tanimlariz.

**<button onclick="alert('merhaba');"> click me</button>** buradada onclick ozelligi icine javascrip kodu yaziyoruz ve ekranda bir uyaru notu cikiyor. Burada javascript kodu nesne icindeki gerceklesmesi istenilen bir durumda kullaniliyor. Asagidaki gibi:



**<script>** KOD ARAYA **</script>**

Ve ya asagidaki gibi root dizinindeki script.js I secer gibi seceriz.

**<script src=”./script.js” >** **<script>**

Asagidaki gibi script tagi head icinde yazilirsa once sayfanin head kismi yuklenecegi icin sonra yuklenen body kismini secemeyiz onun icin script tagini body nin altina yerlestirmeliyiz.

<!DOCTYPE *html*>

<html *lang*="en">

  <head>

    <meta *charset*="UTF-8" />

    <title>Document</title>

    <script>

      document.querySelector("p");

    </script>

  </head>

  <body>

    <p>Deneme</p>

  </body>

</html>

**Javascript: Degiskenler**

* HTML ve CSS deki bazi kisimlarin gecici vir bellekte hesaplanip tutularak islem yapilmasi olayini degiskenler ile yapariz.
* Degisken tanimlamak icin once var degiskenini girmemiz gerekmektedir. Asagidaki gibi

|  |
| --- |
| <script>  var marka;  console.log(marka);  </script> |

* Yukarda bir degisken tanimladik ama tanimladigimiz marka degiskenine herhangi bir bilgi atmadik bu yuzden tarayici konsol bunu underfined olarak isaretler.

|  |
| --- |
| <script>  var marka;  console.log(marka);  **marka = "Opel";**  **console.log(marka);**  </script> |

* Yukardada once tanimlama yaptik daha sonra cikti istedik. Bu bir string tanimlamasidir.

|  |
| --- |
| <script>  var marka;  console.log(marka);  marka = "Opel";  console.log(marka);  var model='Astra';  console.log(model);  **console.log(marka + ' ' + model);**  </script> |

* Bu sekildede birlestirebiliriz.

|  |
| --- |
| var yakitdeposu=100;  var yakitltfiyati=7;  console.log(yakitdeposu\*yakitltfiyati); |

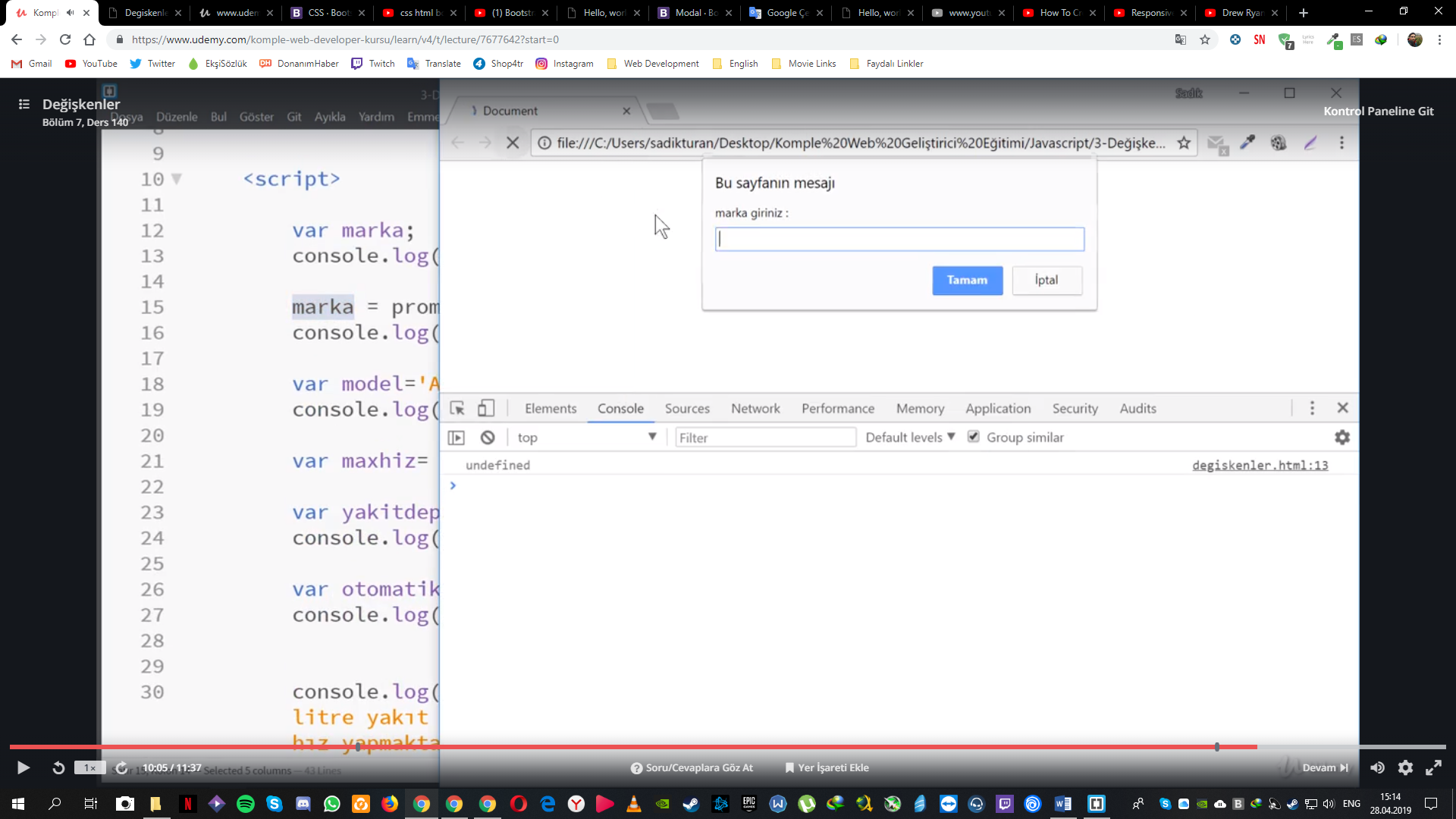
* Yukaridaki gibi var degiskenimize yani degiskenimzi yakitdeposu’na tirnak isaretlerini koymadigimiz zaman sayilar bir deger tanimlayabiliriz.
* Ve bunlarla daha sonra yukaridaki gibi bir islem yapip sonuclar elde edebiliriz.

|  |
| --- |
| var otomatikvites=true;  console.log(otomatikvites); |

* Yukaridaki gibi degiskenlerimize true yada false degeri verip console log da daha sonra true veya false oldugunu gorebiliriz.

|  |
| --- |
| var marka;  console.log(marka);    marka = **prompt("marka giriniz : ")**  console.log(marka); |

* Yukaridaki gibi prompt tanimlamasini yaparakda asagidaki gibi bir goruntu aliriz. Daha sonra console loga yazdirgimiz seyler de buraya girdigimiz tanimlama gozukur.



|  |
| --- |
| var yazilibir=prompt('Lutfen 1. yazili ortamanizi giriniz');  var yaziliiki=prompt('Lutfen ikinci yazili ortamini giriniz');  var ortalama=(**Number**(yazilibir)+**Number**(yaziliiki))/2; |

* Prompt ile sayisal deger alip islem yaparken tanimladigimiz degerlerin sayi oldugunu belirtmek icin Number kullanmamiz gerekiyor.

**Javascript Operatorler**

|  |
| --- |
| //Matematiksel operatorler  var x = 10;  var y = 20;  var z = 30;  console.log(x \* y + z);  x++ = x+1 demektir.  x-- = x-1 demektir. |

|  |
| --- |
| //Karsialstirma operatorleri  var sonuc = x>y;  var sonuc2 = x<y;  var sonuc3 = x>=y;  console.log(sonuc);  console.log(sonuc2);  console.log(sonuc3);  console.log(x==z);  console.log(x==='10'); = Burada x ve ‘10’ un once tip olarak esit olup olmadiklari kontrol edilir daha sorna sayisal olarak esitligi kontrol edilir.  console.log(x!=z); = Burada != Esit degildir anlamina gelmektedir.   * Console komutlarimiz konsolda dogruysa **true** yanlissa **false** diye gozukur. |

|  |
| --- |
| //Mantiksal operatorler  // && (And) Operatoru  // || (Or) Operatoru  // ! (Not) Operatoru  console.log((20>10) && (10>20)); Yani burada her iki kosulda saglanirsa konsolo true degeri gelir. Bu yuzden burasi false.  console.log((20>10) && (30>20)); Burada her iki kosulda dogru oldugundan bu yuzden burasi true.  console.log((20>10) || (30>20)); Burada her iki kosuldan biri dogru oldugunda konsol true yazar. Bu yuzden burasi true.  console.log((10>10) && (10>20)); Bu yuzden burasida false cunku her iki kosuldan ikiside yanlis.  console.log(!(x>z)); Burada ise parantez icerisinde x>z denilmis ama disaridaki o isaret degildir oldugu icin bu komut konsola true yazdirir. |

|  |
| --- |
| console.log('Sayi1 en buyuk mu? : ', + sayi1>sayi2>sayi3);  Console komutu yazarken iceride iki deger varsa virgul kullanmayi unutmuyoruz yukaridaki ornekte kullanildigi gibi. |

**Javascript Kosul Ifadeleri**

|  |
| --- |
| **// If/else kosul ifadeleri(statement)**  var marka = 'Opel';  var model = 'Astra';  var otomatik = 'yes';  if(otomatik=='yes'){  console.log('Arac otomatiktir.');  }else{console.log('arac otomatik degildir.')}; |

* Yukarida belli degerler atadik ve altinda kosul cumleleriyle kullanimini gosterdik. Onemli olan kosul cumlenin yapisini anlamak

|  |
| --- |
| otomatik=true;  if(otomatik){  console.log('Arac otomatiktir);  }  otomatik=true;  if(**otomatik=='true'**){  console.log("true");  }  if(10=='10'){  console.log("Sayilar esittir");  }  if(10==='10'){  console.log("Sayilar esittir");  } |

* Otomatige sadece true diyoruz. Ve if’de parantez icinde sadece otomatik varsa direk EGER DOGRUYSA diye anlar kod kismini. Yukaridada eger dogruysa ‘Arac otomatiktir’ yaz dedik. Ve usttede dogru oldugunu belirttigimiz icin konsol bunu yazdi.
* Alt kisimda ise bir string ile karsilastirmak istedigimizde console bize hicbirsey gostermez cunku string ile true degerini bu sekilde karsilastiramayiz.
* Ama sonraki ornekte bir sayi ile bir string sayisini karsilastirmak istersek burada bunu yapabiliriz. Ve console bize ciktiyi “Sayilar esittir” diye verecektir.
* Ama 3 esittir simgesi olursa buda kabul edilmez ve console cikti vermezdi.

|  |
| --- |
| **// Switch kosul ifadeleri(statement)**  var vites = '3';  if(vites=='1'){  console.log('arac manueldir');  }else if(vites=='2'){  console.log('Arac otomatiktir');  }else{  console.log('Yanlis bir deger');  }  switch(vites){  case '1':  console.log('arac manueldir.');  break;  case '2':  console.log('arac otomatiktir');  break;  default:  console.log('Yanlis deger');  } |

* Yukarida ise **else if** ile istedigimiz kadar ekstra kosul belirleyebilir ve en son bunlarin hicbiri degilse icinde sadece **else** ekleyebiliriz. Veyahutta asagisindaki SWITCH yapimizi kullanabiliriz.
* DEFAULT = ELSE.
* Break ıse eger altında oldugu kosul dogruysa dırek breakten ıtıbaren cıkar ve dıger kodlar sorgulanmaz.

**Javascript – Donguler**

* 1’den 10000’e kadar sayi yazmak gibi uzun isleri tek tek kod ile yazmaktan ise dongu kullanarak kisa sekilde bunu yazmamizi saglayan sistemlerdir. 2 cesit dongu vardir. Bunlar: FOR ve WHILE dongusudur.

|  |
| --- |
| for(var i=0; i<10; i++){  console.log(i);  } |

* Yukaridaki kod ile soldaki gibi bir goruntu aliriz.

|  |
| --- |
| var sehirler=["kocaeli","istanbul","ankara","izmir","adana"];  for(var i=0; i<5; i++){  console.log(sehirler[i]);  }  for(var i=0; i<**sehirler.lenght**; i++){  console.log(sehirler[i]);  } |

* Bu sekilde bir kod yazdikmida solundaki gibi bir goruntu elde ederiz. Select boxlara bu sekilde rahat ve koalylikla veri isleyebiliriz.
* Her zaman listenin uzunlugunu bilmeyebiliriz. İste bu zamanlardada listemizin adi ve .lenght yani uzunlugu kadar seklinde sinirlandirmamizi yapabiliriz. Bu sekilde liste uzunlugunu saymamiz hesaplamamiz gerekmez.

|  |
| --- |
| var a=0;  while(a<10){  console.log(a);  a++;  } |

* Bu kod ilede ilk ornekteki gibi 1’den 10’a kadar bir sayi ciktisi alabiliriz.

|  |
| --- |
| var b=0;  while(b<sehirler.length){  console.log(sehirler[b]);  b++;  } |

* Bu kod ilede for dongusundeki ikinci ornekteki ayni ciktiyi goruntuyu elde ederiz.

|  |
| --- |
| for(var i=0; i<100; i++){  if(i%2==0){  console.log(i);  }  } |

* Buradaki % isareti bolumunden kalan anlamina geliyor. Yani orada i’nin 2’ye bolumunden kalan 0 ise eger o sayilari consola yazdir demisiz. Yani bu sekilde 1 ile 100 arasindaki sayilardan cift olanlari hizlica secebilmisiz.
* Eger i%2==1 yapsaydikda kalani 1 olan sayilari bulacagi icin tek sayilari bulmus ve yazdirmis olurduk.

|  |
| --- |
| for(var i=0; i<100; i++){  if(i==50){  break;  }  console.log(i);  } |

* Bu kod ilede 49 dahil olmak uzere cikti veren ve 50nin ciktisini vermeden cikip break olan bir islem yapariz. Yani 49’a kadar cikti aliriz. Oraya 50 haric baska bir deger yazip o degerinde ciktisini alabiliriz.

|  |
| --- |
| for(var i=0; i<100; i++){  if(i==50){  continue;  }  console.log(i);  } |

* Break yerine continue kullansaydik egerde dongu 49’a kadar yine gelir ve 50’yi atlar sonra sonsuza kadar devam ederdi. Bunun ciktisida bu sekilde olurdu. Yani burada sadece secilen sayiyi atlatmis olduk.

|  |
| --- |
| var kullaniciadi='guest';  var sifre='12345';  var girisyapildi=false;  while(girisyapildi==false){  var id=prompt('Lutfen kullanici adinizi giriniz.');  var pw=prompt('Lutfen sifrenizi giriniz');  if((id==kullaniciadi)&&(pw==sifre)){  console.log('Basari ile giris yapildi');  girisyapildi=true;  }  } |

* Bu sekildede bir kullanici giris paneli olusturabiliriz.

**Javascript - Diziler**

* Amaci ayni olan verileri tek bir degisken icine saklama olayini diziler denilir.

|  |
| --- |
| var marka=["Opel", "Toyota", "Mazda", "Renault"];  var model=new Array("Corsa", "Yaris", "CX-3", "Clio");  console.log(marka);  console.log(model); |

* Yukaridaki kod ile 2 farkli yontem ile dizi olusturduk ve cikti aldik farklar ortada. İki yontemden birini kullanabiliriz bnize kalmis.

|  |
| --- |
| var marka=["Opel", "Toyota", "Mazda", "Renault"];  var model=new Array("Corsa", "Yaris", "CX-3", "Clio");  console.log(marka**[0]**);  console.log(model**[1]**);  model[2]=”Mazda”;  console.log(model); |

* Yukaridaki ornektede console ile marka dizisinin 0. Elemanini yazdirmayi istedik yani Opel’i. Model dizisinin ise 1. Elemani Yaris’i yazdirmak istedik.
* Veya model[2]=”Mazda” bu sekilde kullanarak tek bir degeri degistirebilirizde sonradan.

|  |
| --- |
| var auris=["Toyota", "Auris", 42, true];  console.log(auris); |

* Veya bu sekilde farkli ozellikleri bir yerde toplayip yazabiliriz. Bazi ozelliklerde cift tirnak isaretinede gerek yoktur.

|  |
| --- |
| var auris=["Toyota", "Auris", 42, true];  auris.push('blue');  auris.unshift('HB');  auris.pop();  auris.shift();  console.log(auris); |

* Push ozelligi ile dizimizin en sonuna bir eleman ekleyebiliriz. Ornegin yukarida ‘bkue’ eklennmis.
* Unshift ozelligi ilede dizimizin en sonuna bir eleman ekleyebiliriz. Ornegin yukaridaki ornegimizde ‘HB’ eklenmis.
* auris.pop(); ile listenin en sonundan bir eleman sileriz.
* auris.shift(); ile listenin en basindan bir eleman sileriz.

|  |
| --- |
| var marka=["Opel", "Toyota", "Mazda", "Renault"];  for(i=0; i<marka.length; i++){  console.log(marka[i]);  } |

* Bu sekildede dizilerimiz ve for dongusunu ic ice kullanabiliriz.

|  |
| --- |
| var marka=["Opel", "Toyota", "Mazda", "Renault"];  if(marka.indexOf('Mazda')!==**-1**){  console.log(marka[marka.indexOf('Mazda')]);  } |

* marka.indexOf('Mazda') yazdigimizda bize donen sonuc -1 ise marka dizimizde ‘Mazda’ diye bir eleman olmadiginiz gosterir. Eger marka dizisinde Mazda varsa marka dizisinin kacinci sirasinda oldugunu yazar. Ornek yukardaki marka dizisinda Mazdanin sirasi 2, Renault’unki 3 gibi.
* Bizde burada eger marka’nin index’inde ‘Mazda’, -1 degerini almiyorsa ve fazladan esittirde turu ayniysa dedigi icin, console.log yapip markanin, ‘Mazda’ nin dize sayisinin degerinin marka dizesindeki karsiligini sordurmusuz. Buda yine bize ‘Mazda’ ciktisini sonucunu verecektir.

|  |
| --- |
| Var marka=[“Opel”, “Toyota”, “Mazda”, “Renault”];  if(marka.indexOf(‘Mercedes’)!==-1){  console.log(‘Bulundu’);  }else{  console.log(‘Aradiginiz model elimizde bulunamadi’)  } |

* Bu sekildede bir dizemizden arama yaptirma mekanizmasi olusturabiliriz.

|  |
| --- |
| models.forEach(function(**elements**){  console.log(**elements**);  });  for(m=0; m<models.length; m++){  console.log(models[m]);  } |

* ForEach’i cagirdigimiz zaman dizinin her bir elemani sira sira element’in icine kopyalaniyor ve alttada console ile sirayla yaziliyor. Sonucta bize for ile olusturdugumuz altindaki kodun ayni sonucunu veriyor.
* **Function’dan sonra gelen element ile console.log’daki element ayni elementler. DIKKAT**

**Javascript - Object**

* Bazen diziler ihtiyacimizi karsilamayabilirler. O zaman objeleri kullanarak cok daha basit sekilde atamalar yapabiliriz.

|  |
| --- |
| var opel={  marka : 'Opel',  model : 'Corsa',  yil : '2015',  renk : 'mavi',  otomatik : true,  maxhiz : 320  }  console.log(opel); |

* Yukaridaki kodu yazdigimizda altindaki gibi bir konsol ciktisi alacagimiz bir object olustururuz.

|  |
| --- |
| console.log(opel.marka);  console.log(opel.model);  console.log(opel.maxhiz);  console.log(opel[‘renk’]);  console.log(opel[‘model’]);  console.log(opel[‘yil’]); |

* Yukaridaki sekildede opel icindeki degerlere erisim saglayabiliriz.

|  |
| --- |
| var toyota = new Object();  toyota.marka='Toyota';  toyota.model='Yaris';  toyota.yil='2012';  toyota.marka='kirmizi';  toyota.otomatik='false';  toyota.maxhiz=250;  console.log(toyota); |

* Veya bu sekildede object tanimlayabiliriz. Yukaridakini veya bunu tercih etmek tamamen bize kalmis birsey.

|  |
| --- |
| toyota.model='Auris';  opel.model='Astra'; |

* Yukaridaki sekillerdede objectlerimizin tanimlanmis degerlerini sonradan degistirebiliriz.

|  |
| --- |
| var arabalar=**[**  **{**marka : 'Opel',  model : 'Corsa',  yil : '2015',  renk : 'mavi',  otomatik : true,  maxhiz : 320**},**    **{**marka : 'Toyota',  model : 'Auris',  yil : '2012',  renk : 'mavi',  otomatik : false,  maxhiz : 250**}**  **]**  console.log(arabalar); |

* Veya yukaridaki gibi objectlerimizi bir dize icerisine yerlestirip sonra console komutu verip arabalar’i cagirdigimizda console bize bu goruntuyu verir. Kullanimi bu sekildedir.

|  |
| --- |
| Console.log(arabalar[0].marka);  Console.log(arabalar[1].model); |

* Bu sekildede objectlerimizi bir dizi icerisine aldigimizda iclerindeki diger birimlere ulasip console’a gonderip yazdirabiliriz.

|  |
| --- |
| for(s=0; s<arabalar.length; s++){  console.log(arabalar[s]);  } |

* Veya for dongusuyle yukaridaki gibi yazip arabalar dizisinin icindekileri alt alta sirayla yazdirabiliriz. Yukaridaki ornekteki gibi.

**UYGULAMA**

|  |
| --- |
| var personel ={  ad : 'Ali',  soyad : 'Yilmaz',  cinsiyet : 'erkek',  telefon : '02623337788',  adres : 'pendik',  meslek : 'bilgisayar muhendisi',  diller : ['ingilizce', 'almanca'],  hobiler : ['sinema', 'spor', 'kitap okumak']  } |

* Yukaridaki gibi birden fazla eklenecek ozellikleri object icinde dizilenerek yaparak yapmak daha mantiklidir.

|  |
| --- |
| var sayi = prompt('Bir sayi girin ve biz size onun tek mi yoksa cift mi oldugunu soyleyelim');  function tekmiciftmi(hesapla){  if(hesapla%2==0){  console.log('Girdiginiz sayi cifttir.');  }else{  console.log('Girdiginiz sayi tektir.');  }    }  tekmiciftmi(sayi); |

* Tekmi yoksa ciftmi hesaplama islemini bu kodlarla yapabiliriz.

|  |
| --- |
| var kelimeler = cumle.split(' '); = bu kod ile cumle yazisindaki metinin bosluklar referans alinarak bolunmesini istiyor. Oraya i yazsaydik ileri keserek oralardan kesecekti. Ve kesim isleminden sonra bunlari var kelimelere tanimlamis oluyoruz. |

|  |
| --- |
| **console.log(ozetCumle.join(' ')+'...');** = bu kod ilede kelime aralarinda olan ifadeleri ayarliyoruz. Mesela burada kelime aralarinda olan kesme isareti virgul gibi ifadelerin olmamasi icin **join(' ')** yapiyoruz ve joinin tirnak isareti icerisindeki kesme isaretine arasina gelmesi gereken seyi yaziyoruz. Eger bosluk birakmasaydik tum kelimeler bitisik yazilirdi.   * Sonunda mavi ile isaretlenmis tirnak icindeki yerede sonuna gelecek isareti veya yaziyi koyabiliriz. Ben 3 nokta tercih ettim. Alttaki orneklerde bunun ciktisi yok.   ILKA HALI    **join(' ')** LI HALI |

|  |
| --- |
| var cumle = prompt('Bir cumle giriniz');  function ozet(cumle, maxKarakter) {  var ozetCumle = [];  if (maxKarakter > cumle.length) {  return cumle;  } else {  var kelimeler = cumle.split(' ');  var toplamKarakter = 0;  for (i = 0; i < kelimeler.length; i++) {  ozetCumle.unshift(kelimeler[i]);  toplamKarakter += kelimeler[i].length;  if (toplamKarakter > maxKarakter) {  break;  }  }  }    return ozetCumle.join(' ') + '...'  }  console.log(ozet(cumle,20)); |

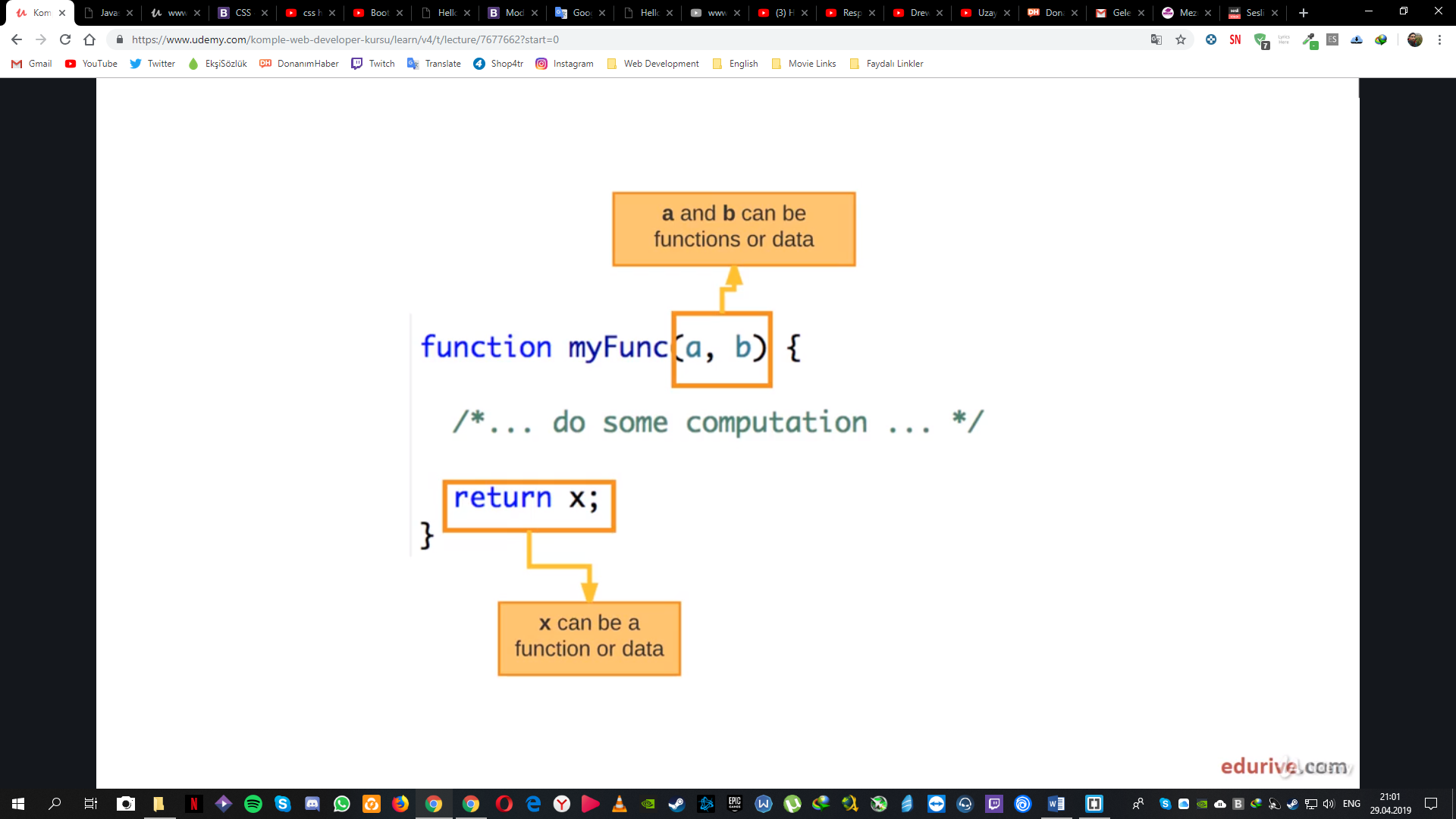
* Bu uygulama ile girilen bir cumleyi 20 karakteri gectikten sonraki cumle bitimine kadar ki olan cumleleri alarak sonuna … koyarak bir cikti verir. Tipki konunun bloglardaki kisa bilgi veren icerigi gibi. 20 karakter sinirini degistirmek bizim elimizde, istersek oraya girdili bir degerde tanimlayabiliriz prompt ile.

|  |
| --- |
| var product = {  name: 'Samsung S8',  price: 3000,  description: 'lorem ada ada ada tasa yasa gasa rata yata kanyak yankat kalaka slak',  summary: function ozet() {  var ozetCumle = [];  var maxKarakter = 20;  **var cumle = this.description;**  if (maxKarakter > cumle.length) {  return cumle;  } else {  var kelimeler = cumle.split(' ');  var toplamKarakter = 0;  for (i = 0; i < kelimeler.length; i++) {  ozetCumle.unshift(kelimeler[i]);  toplamKarakter += kelimeler[i].length;  if (toplamKarakter > maxKarakter) {  break;  }  }  }  return ozetCumle.join(' ') + '...'  }  console.log(ozet(cumle, 20));  } |

* (**var cumle = this.description;**) Buradathis ile objemizin en ust yoluna gelip daha sonra . koyarak onun alt klasorlerine erisimi saglayip descriptionu secme islemini yaptik.

**Javascript - Fonksiyonlar**

* Bir matematik islemini bir cok kez yazip yapmak yerine bir islem formulu olusturuyoruz. Ve bu formul kisayoldan bu islemi yapiyor. Bunlara fonksiyon diyoruz.



* Yukarida fonksiyonun ingilizce olarak tanimini goruyoruz. Biz bir ave b tanimliyoruz daha sonra fonksiyonun icinde islem yapiliyor ve fonksiyon bu islem sonucunu bize x olarak donduruyor. Kisaca anlami bu.

|  |
| --- |
| function topla(a, b){  return a+b;  }  var sonuc = topla(20,10);  console.log(sonuc); |

* Yukarida basit bir fonksiyon ornegi yaptik. Once bir fonksiyon olsuturduk ve buna a ve b yi tanimladik. Daha sonra return olarak a+b islemini yaparak bize vermesini istedik. Ve daha sonra sonuc degiskeni tanimladik ve topla fonksiyonunu calistirip parantez icine islemlerin yapilacagi degerleri tanimladik. Daha sonrada console ile sonucu istedik.

|  |
| --- |
| var now = 2019;  var ad = prompt('Isminizi giriniz')  var dogumtarihi = prompt('Dogum tarih yilinizi 4 haneli olacak sekilde giriniz?')  function bilgilendir(ad, dogumtarihi){  yas=now-dogumtarihi;  var emeklilik = 65 - yas;  console.log(ad + ' ' + yas + ' yasinda ve ' + emeklilik + ' yil sonra emekli olabilir.')  }  bilgilendir(ad, dogumtarihi); |

* Bu sekildede bir islem yapabiliriz. Anlatmak gerekirse;
  + Simdinin tarihini 2019 olarak ayarliyoruz
  + Sayfa acildiginda kullanicimizdan bir ad ve dogum tarihi girmesini istyoruz
  + Ve en altta sayfa acildiginda bilgilendir fonksiyonunu calistiriyoruz. Eger calistirmasaydik yani yazmasaydik yukardaki kodlar sayfa acildiginda istemezdi.
  + Fonksiyonu calistirdigimizda ad ve dogum tarihini sayfa basinda belirliyoruz.
  + Daha sonra bu belirledigimzi degerler fonksiyon icerisine giriyor ve once yas hesaplaniyor.
  + Daha sonra emeklilik tanimlaniyor ve hesaplaniyor.
  + Daha sonra bu hesaplanan sonuclar console ile return degerine donusturuluyor.

**JavascriptDOM – Nedir**

* Document(DOM) objesi araciligiyla html vs css’e ulasip onlar uzerinde degisiklik yaptigimiz ozelliktir.

**JavascriptDOM - Selectors**

|  |  |
| --- | --- |
| **document.getElementById('mavi')**   * Bu kodu konsola yazdigimizda soldaki id’si mavi olan divi seceriz. Ve istedigimiz degiskenleri tanimlayabiliriz.   **document.getElementsByTagName('div')**   * Bu kod ilede sayfadaki tum divlere ulasip degisiklikler yapabiliriz. Bu bize divlerin olusturdugu bir html collection dizisi getirir.   **document.getElementsByClassName('kutu')**   * Bu kod ilede sayfadaki classlari seceriz. Yukaridaki ornekte soldaki sayfanin class’i kutu olanlari secilmek istenmsitir. Buda yine bir htmlcollection olarak bize doner ve icinde tum kutu classlari bulunur.   **document.querySelector('#mavi')**   * Veya yukaridaki gibi hizli secimler yapip yine secimlerimizi yapabiliriz. Burada sececegimiz seye uygun olarak onlarin css htmldeki karsiliklari gelir.   **document.querySelector('.kutu')**   * Bu sekildede class’i kutu olan ilk elemani seceriz.   **document.querySelectorAll('.kutu')**   * Bu sekildede class’i kutu olan tumu secilir. | <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <title>JavascriptDOM</title>  <style>  \*{  padding: 0;  margin: 0;  }  .kutu{  height: 200px;  width: 200px;  float: left;  margin: 20px;  }  #mavi{  background-color: blue; }  #kirmizi{  background-color: red; }  #sari{  background-color: yellow; }  </style>  </head>  <body>  <div class="kutu" id="mavi"> </div>  <div class="kutu" id="kirmizi"> </div>  <div class="kutu" id="sari"> </div>  </body>  </html> |

|  |
| --- |
| var isBlue = false;  var blue = document.getElementById('mavi');    **setInterval**(function(){  if(isBlue){  blue.style.background='green';  isBlue=false;  }else{  blue.style.background='blue';  isBlue=true;  }  }, **1000**) |

* **setInterval** icerisine aldigi seyleri belli basli saniyede bir tekrar tekrar calistiriyor. Sureyi icine elemani yazdiktan sonra virgul atip saniyeyi yaziyoruz. Mesela yukaridaki ornekte saniye 1 saniye olarak en altta 1000 yazilarak ayarlanmis.
* Veya isBlue false veya true yerine, bunlari kaldirarak elsenin hizasindan **isBlue!=isBlue**’de yapabilirdik.
* elemani sectikten sonra blue.style.background='green'; bu sekildede styleden ozellikler verebiliriz.

Uygulama

|  |  |
| --- | --- |
| var todoList = document.getElementById('todoList');  todoList.style.listStyle='none';   * Bu kod tum liste elemanlarinin basindaki noktalama liste isaretleri gider.   var todoitem = document.querySelectorAll('li');  todoitem.forEach(function(item){  item.style.textDecoration='line-through';  });   * Bu kod ile tum liste elemanlarinin ustu soldaki ilk siradakinin ustu cizildigi gibi cizilir. Checked dahil degildir.   var date = new Date();  var hour = date.getHours();  var element;    for (var i=hour; i>=0; i--){  element=document.getElementById('item\_'+i);  element.style.textDecoration='line-through';  element.querySelector('input').checked=true;  }   * Bu kod ile bilgisayar saati referans alinir. Ve bilgisayar saarinin sadece saat kismi alinir. Yani saat 23:00’sa 23 dur degerimiz. Ve saatimize esit ve altindaki tum liste elemanlari cizilir.   + Date’ye new Date() ile pc tarihi tanimlanir.   + Hour’a ise datenin saati getHours() ile cekilerek tanimlanir.   + Daha sonra for dongusu kurulur.   + element=document.getElementById('item\_'+i); İle islem yapilacak nesneler belirlenir.   + element.style.textDecoration='line-through'; İle belirlenen nesnelerin uzerleri saat ilskisine gore isaretlenir.   + element.querySelector('input').checked=true; İle sollarindaki checkbix’lar saat ilskisine gore isaretlenir. |  |