T.C.
MİLLÎ EĞİTİM BAKANLIĞI

BİLİŞİM TEKNOLOJİLERİ

BETİK DİLİ (JAVASCRİPT)

Ankara, 2013

- Bu modül, mesleki ve teknik eğitim okul/kurumlarında uygulanan Çerçeve Öğretim Programlarında yer alan yeterlikleri kazandırmaya yönelik olarak öğrencilere rehberlik etmek amacıyla hazırlanmış bireysel öğrenme materyalidir.
- Millî Eğitim Bakanlığınca ücretsiz olarak verilmiştir.
- PARA İLE SATILMAZ.

İÇİNDEKİLER

AÇIKLAMALAR

ALAN	**Bilişim Teknolojileri**
DAL/MESLEK	**Web Programcılığı**
MODÜLÜN ADI	**Betik Dili (JavaScript)**
MODÜLÜN TANIMI	Bu modül, betik dili yardımıyla web siteleri için script oluşturabilecek yeterliliklerin kazandırıldığı bir öğrenme materyalidir.
SÜRE	40/32
ÖNKOŞUL	Programlama Temelleri dersi modüllerini tamamlamış olmak
YETERLİK	Betik dili yardımıyla script oluşturmak
MODÜLÜN AMACI	**Genel Amaç** Bu modül ile gerekli ortam sağlandığında betik dili yardımıyla web siteleri içinde kullanılacak scriptleri oluşturabileceksiniz. **Amaçlar** **1.** Değişkenleri ve mantıksal işlem operatörlerini kullanabileceksiniz. **2.** Akış kontrol, dizi ve fonksiyon yapılarını kullanabileceksiniz. **3.** İhtiyaca uygun script komutlarını kullanabileceksiniz. **4.** Olay ve nesne işlemlerini yapabileceksiniz.
EĞİTİM ÖĞRETİM ORTAMLARI VE DONANIMLARI	**Ortam:** Bilişim Teknolojileri laboratuvarı, işletme ortamı **Donanım:** Bilgisayar
ÖLÇME VE DEĞERLENDİRME	Modül içinde yer alan her öğrenme faaliyetinden sonra verilen ölçme araçları ile kendinizi değerlendireceksiniz. Öğretmen modül sonunda ölçme aracı (çoktan seçmeli test, doğru-yanlış testi, boşluk doldurma, eşleştirme vb.) kullanarak modül uygulamaları ile kazandığınız bilgi ve becerileri ölçerek sizi değerlendirecektir.

GİRİŞ

Sevgili Öğrenci,

İnternet üzerinde çalışan siteler, oyunlar ve birçok özel uygulama, betik dili ile hızlı ve kolay bir şekilde yapılmaktadır. Betik dili, hazırladığınız sayfalara dinamizm kazandırmak için tasarlanmış, farklı platformlarda ve nesne tabanlı çalışabilen bir programlama dilidir.

Betik dili ile dinamik resim galerileri, kullanıcı formları, oyunlar, duruma uygun çalışabilen uygulamalar hazırlayabilir, internette yayınlayabilir, kolayca kullanıma sunabilirsiniz.

Bu modül ile betik dilinde değişken tanımlayabilecek değişkenlerle işlemler yapabilecek, fonksiyon üretip sahnede olan her şeye hükmedebileceksiniz. İnternet sayfalarınız için özel uygulamalar hazırlayıp nesneye yönelik programlama yapabileceksiniz. Yazdığınız kodlara karar yapıları ekleyip şartlara göre değişen kod yapıları oluşturabileceksiniz.

ÖĞRENME FAALİYETİ–1

AMAÇ

Bu faaliyet sonunda değişkenleri ve mantıksal işlem operatörlerini kullanabileceksiniz.

ARAŞTIRMA

- JavaScript şeklinde arama yaparak internet siteleri için kullanılan hazır scriptleri araştırınız.

1. PROGRAMLAMA DİLİ

1.1. Betik Dili (JavaScript)

Betik dili, web sayfalarında dinamik içerik sağlamak ve kullanıcıyla iletişim kurmak için kullanılan, istemci tarafında çalışan bir dildir. Html etiketleri arasında tanımlanarak kullanılabilir. 1995 yılında Brendan Eich tarafından geliştirilmiştir.

JavaScript dilini, c dilinin web sayfalarına uyarlanmış basit sürümü olarak düşünülebilirsiniz. JavaScript dili isim benzerliğine rağmen Java ile birbirinden bağımsız ve farklı dillerdir. Java gerçek bir programlama dilidir. JavaScript ise bir script dildir.

JavaScript dili, C, Scheme, Java, Perl, Python,Self dillerinden etkilenmiş ve geliştirilmiştir. JScript, JScript .NET,Objective-J, TIScript dillerinin oluşturulması için kullanılmış ve gelişmesine yardımcı olmuştur.

Başlangıçta sadece Netscape tarafından desteklenen dil, şimdi tüm tarayıcılar tarafından desteklenmekte ve kullanılmaktadır. Temel script komutları her tarayıcıda çalışmasına rağmen ileri düzey komutlar bazı tarayıcılar tarafından tam olarak desteklenmemektedir. Script komutlarının tarayıcı üzerinde çalışıp çalışmamasını, güvenlik açısından kullanıcının belirlemesi sağlanmıştır. İsteyen kişi, tarayıcı ayarlarından script kullanımını kapatabilir.

JavaScript dili 1.0 versiyonundan başlayarak 1.8.5 versiyonuna ulaşmıştır. Script kodlarını web editör uygulamasında ya da notepad kullanarak oluşturup düzenleyebilirsiniz.

1.1.1. HTML'e JavaScript Kodlarını Tanıtma

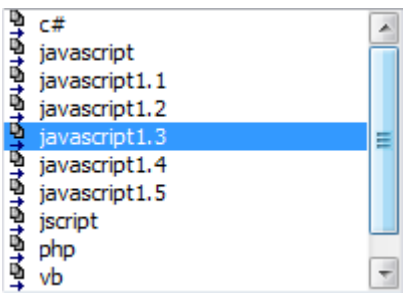
Script kodlarını html'e tanıtmak için;

```
10  <script>       </script>
```

etiketleri kullanılır.

```
10  <script type="text/javascript" language="javascript">
11
12  // çalıştırılacak kodlar bu bölüme yazılır.
13
14  </script>
```

Type parametresi script blokları arasında yazılacak kodların türünü belirtir. Language parametresi ile JavaScript kodlarının sürümünü belirtir. Tüm tarayıcılarda sorunsuz çalışması için JavaScript seçmek doğru olacaktır.



Şekil 1.1: JavaScript sürümleri

1.1.2. JavaScript Kodlarının HTML'deki Konumu

JavaScript kodları sayfa içerisinde üç bölümde kullanılabilir:

- Birinci yol, script kodlarını head etiketleri arasına yazarak kullanmaktır. <head></head> etikletleri arasında yazılan kodlar sayfa yüklenmeden derlenecektir. Sayfamızda gerçekleşen olaylar karşısında (düğme tıklanması vb.) kodların çalışmasını istiyorsak gerekli fonksiyonları ve değişkenleri bu bölümde yazmamız gereklidir.

```
3  <head>
4  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
5  <title>İlk JavaScript Kodlarım</title>
6  <script language="javascript" type="text/javascript">
7  document.write("İlk JavaScript Kodlarım");
8  </script>
9  </head>
```

- İkinci yol, script kodlarını <body></body> etiketleri arasına yazarak kullanmaktır. Body etiketleri arasına yazılan script kodları sayfa yüklendiğinde otomatik olarak derlenip çalışacaktır.

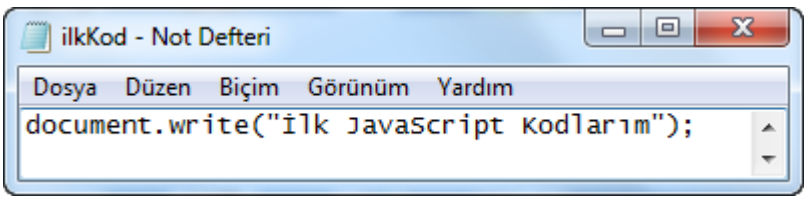
```
8  <body>
9  <script language="javascript" type="text/javascript">
10 document.write("İlk JavaScript Kodlarım");
11 </script>
12 </body>
```

İki şekilde de sayfanın çıktısı aynı olacaktır.

- Script çalıştırmanın diğer bir yolu *.js uzantılı olarak kaydettiğimiz haricî bir JavaScript dosyasını sayfaya dâhil edip kodları çalıştırmaktır.

Web sayfasına harici JavaScript dosyası dâhil etmek için;

- Notepad programını açarak script kodlarını yazarak ilkKod.js şeklinde masaüstüne kaydediniz.



Şekil 1.2: JavaScript sürümleri

- Web editör programını çalıştırarak yine masaüstüne yeni bir html sayfası oluşturunuz.

- Script komutunun src parametresini kullanarak oluşturduğunuz js dosyasını aşağıdaki şekilde web sayfasına dâhil ediniz.

```
8  <body>
9  <script src="ilkKod.js" language="javascript" type="text/javascript">
10 </script>
11 </body>
```

- Oluşturduğumuz js dosyası ve html dosyası farklı klasörlerde ise js dosyasının html dosyasına göre adresi kaynak (src) parametresi içerisinde belirtmek gerekir.

- Html sayfasını çalıştırdığımızda ekranda sola dayalı olarak “ilk JavaScript Kodlarım ” ifadesi yazdırılacaktır.

1.2. Yazım Kuralları

JavaScript kodlarını yazarken basit yazım kuralları vardır. JavaScript kodlarını yazmamızı sağlayan editörleri kullanmadan hata bulmak ve hataları düzeltmek çok fazla vakit alır.

JavaScript, html etiketleri gibi boşlukları önemsemez. Örneğin **sayi=10** ile **sayi = 10** aynı şekilde çalışacaktır.

1.2.1. İsimlendirme

JavaScript dili büyük küçük harf duyarlı bir dildir. JavaScript dilinde ad, Ad, AD, aD birbirinden farklı isimlerdir. JavaScript dilinde değişkenler, fonksiyonlar vb. nesneler;

- Sayı ile başlayamaz.
- Kelimeler arasında boşluk kullanılamaz.
- Boşluk yerine _ işareti kullanılır.
- JavaScript dili için ayrılmış kelimeler herhangi bir nesneye isim olarak verilemez.

break	else	new	var
case	finally	return	void
catch	for	switch	while
continue	function	this	with
default	if	throw	double
delete	in	try	import
do	instanceof	typeof	public
abstract	enum	int	short
boolean	export	interface	static
byte	extends	long	super
char	final	native	synchronized
class	float	package	throws
const	goto	private	transient
debugger	implements	protected	volatile

Tablo 1.1: Ayrılmış kelimeler

1.2.2. Özel İşaret Gösterimi

JavaScript kodları içerisinde özel karakter kullanmak istersek \ işaretini özel karakterin başına eklemek zorundayız. Eklenebilecek özel karakterler tablosu aşağıdadır:

Kod	Çıktı
\'	Tek tırnak
\"	Çift tırnak
\&	Ve işareti
\\	Ters taksim
\n	Yeni satır
\r	Satır başı
\t	tab
\b	backspace
\f	Besleme oluşturmak

Tablo 1.2: Özel işaretler

```
11  <script>
12  var yazi="Yaşlı kadın "volkan" diye bağırdı";
13  document.write(yazi);
14  </script>
```

Yukarıdaki kodlarda volkan kelimesi için çift tırnak işareti kullanılmak istenmiştir. Çift tırnak işareti aynı zamanda script ifadesinin bittiğini göstermektedir. Bu yüzden 12 numaralı satır hata verecek ve script kodumuz çalışmayacaktır.

```
11  <script>
12  var yazi="Yaşlı kadın \" volkan \" diye bağırdı";
13  document.write(yazi);
14  </script>
```

\ ifadesi tırnakların başına eklenerek bu hata ortadan kaldırılmıştır. Sayfamız çalıştırıldığında aşağıdaki ifade elde edilir:

Yaşlı kadın " volkan " diye bağırdı

1.2.3. Yorum Kodları

Yazılan script kodlarına diğer programcılar ya da kendiniz için açıklamalar eklemek isteyebilirsiniz ya da istediğiniz herhangi bir kodun çalışmasını istemeyebilirsiniz.

Bu durumlarda açıklama satırları eklemek ya da kodları açıklama satırı hâline getirmek faydalı olacaktır.

İki şekilde açıklama satırı eklenebilir.

- // simgesini kullanarak tek satır hâline açıklama satırı eklenebilir. İfade alt satıra geçtiğinde açıklama satırından çıkmış olur.

```
11  <script>
12  // bu ifadeler tek satır açıklama için kullanılır
13  ifade alt satıra geçtiğinde açıklama olmakdan çıkar
14  </script>
```

- /* */ ifadeleri arasına yazılan her şey açıklama olarak alınır. Satır sınırlaması yoktur.

```
11  <script>
12  /* bu ifadeler
13  ile
14  istediğimiz kadar  açıklama satırı oluşturabiliriz.*/
15  </script>
```

1.3. Değişkenler

Programlama dillerinde belirlediğimiz isimlerle oluşmuş ve hafızada yer kaplayan alanlara ***değişken*** denir. Tüm programlama dillerinde olduğu gibi JavaScript içerisinde de değişken tanımlaması yapılarak kullanılır.

1.3.1. Değişken Tanımlama ve Değişken Türleri

Değişken tanımlamak için variable (değişken) kelimesinin kısaltılmışı olan var kelimesi kullanılır. Var kelimesinden sonra değişken ismi yazılarak değişken tanımlanmış olur.

Değişken tanımlanırken dikkat edilmesi gereken kurallar bulunmaktadır. Bu kurallardan isimlendirme başlığı altında bahsetmiştik. Bu kuralların yanında Türkçe karakter kullanmak değişkenler için herhangi bir sorun oluşturmasa da yazdığınız kodları çalıştıracak bilgisayarlarda (örneğin Çin'deki bir bilgisayarda) nasıl çalışacağını düşünerek Türkçe için kullanılan high-ASCII (ı, İ, ğ, Ğ, Ş, Ş, ü, Ü, ö, Ö, ç, Ç) karakterlerini kullanmamamız daha uygun olacaktır.

Değişken tanımlamak için kullanacağımız kalıp aşağıdadır:

```
12  var değişken_ismi
```

Değişkenler için kullanabileceğimiz üç temel değişken türü bulunur:

- Number veri tipi

Sayıları tutmak için number veri türü kullanılır. Tüm sayılar için tanımlanan değişken kullanılabilir.

- Boolean veri tipi

Doğru (true) ve yanlış (false) olmak üzere iki adet lojik değer tutar. Genellikle karşılaştırma işlemlerinde kullanılır.

- String veri tipi

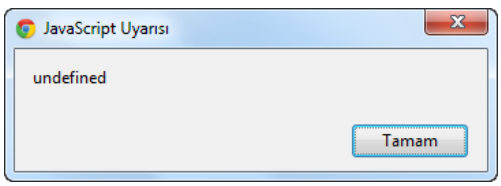
Karakter, metin ya da metin gruplarında kullanılır. String ifadeler tanımlanırken tırnak işareti kullanılır. Sayı değerleri tırnak içerisinde tanımlanmış ise string ifade olarak alınır ve matematiksel işlem yapılmaz.

1.3.2. Değişkenlere Değer Verme

JavaScript dilinde tanımlanan değişken, herhangi bir değer verilmediğinde boş olarak bekleyecektir.

```
11  <script>
12  var sayilar;
13  alert(sayilar);
14  </script>
```

İfadesi sayilar isminde bir değişken oluşturmuş fakat herhangi bir değer atamamıştır. İfade çalıştırıldığında alert komutu değişkenin değerini uyarı kutusu içerisinde gösterecektir.



Şekil 1.3: Tanımsız uyarısı

JavaScript dilinde değişken tanımlarken aynı satır içerisinde değişkene değer verme imkânı sunmuştur. Bu sayede değişken tanımlanırken değer aktarılmış olur.

```
11  <script>
12  var sayilar=10;
13  alert(sayilar);
14  </script>
```

Değişkene eşittir (=) ifadesi kullanılarak değer atanmıştır. Eşittir ifadesi atama operatörüdür ve **eşitliğin sağ tarafındaki değeri sol taraftaki değişkene aktarır**. Değişken tanımlanırken unutmamamız gereken kural, tanımlamanın değişken kullanılmadan önce yapılmasıdır.

```
10  <script>
11  var sayilar;
12  sayilar=10;
13  alert(sayilar);
14  </script>
```

11. satırda sayilar isminde bir değişken oluşturulmuş 12. satırda değişkene değer atanmıştır.

Farklı türlerde değişken tanımlayarak değer vermek için;

- Editör programını açarak yeni bir html sayfa oluşturunuz.
- Kod bölümüne gelerek sayı, string ve boolean türünde değişkenler tanımlayıp rastgele değerler veriniz.

```
10  <script>
11  var sayi1,sayi2;
12  sayi1=10;
13  sayi2=20;
14  var sayi3=30;
15  var isim="Volkan GÖK";
16  var cinsiyet='E';
17  var yas="30";
18  var dogrumu=true;
19  </script>
```

Tanımladığımız değişkenlerde isim değişkeninde çift, cinsiyet değişkeninde tek tırnak kullanılmıştır. İşlevsellikte herhangi bir fark yoktur. Fakat kod okunurluğu açısından iç içe kullanımlarda farklı tırnakları kullanmak faydalı olacaktır.

Yas değişkeni tırnaklar arasına alınarak string ifadeye çevrilmiştir.

1.3.3. Tür Dönüşümleri

JavaScript dinamik bir dildir. Veri türlerini kendisi otomatik olarak yapabilir. Tanımlayıp sayısal değer verdiğiniz bir değişkene daha sonra string türünden değer atayabilirsiniz.

Sayi isminde 50 değerinde bir değişkenin değerini Merhaba yapmak için;

- Editör programını açarak yeni bir html sayfası oluşturunuz.
- Sayi isminde bir değişken tanımlayarak 50 değerini veriniz.

```
10  <script>
11  var sayi=50;
```

- Sayi değişkeninin değerini Merhaba olarak değiştiriniz.

```
10  <script>
11  var sayi=50;
12  sayi="Merhaba";
13  </script>
```

JavaScript Number türünü otomatik olarak string türüne çevirecektir.

1.3.4. Türler Arası İşlemler

Toplama ve çıkartma operatörleri ile farklı veri türleri arasında işlemler yapılabilir.

- **Toplama operatörü ile yapılan işlemler**

```
10  <script>
11  var yazi="merhaba"//string türünde
12  var sayi=40;//Number türünde
13  alert(yazi+sayi);
14  </script>
```

Yazi ve sayi değişkenlerini toplatıp sayfayı çalıştırdığımızda sonuç olarak

```
merhaba40
```

çıktısını verecektir.

```
10  <script>
11  var yazi="merhaba"//string türünde
12  var dogrumu=false;//Boolean türünde
13  alert(yazi+dogrumu);
14  </script>
```

Boolean türü ile string türünü topladığımızda

```
merhabafalse
```

çıktısını verecektir.

```
10  <script>
11  var dogrumu=true;//Boolean türünde
12  var sayi=40;//Number türünde
13  alert(sayi+dogrumu);
14  </script>
```

Boolean türü ile number türünü topladığımızda sayı ile mantıksal türün true (1) ve false (0) değerlerini toplar.

```
41
```

Sonuç olarak 41 değerini verir.

```
10  <script>
11  var dogrumu=false;//Boolean türünde
12  var sayi=40;//Number türünde
13  alert(sayi+dogrumu);
14  </script>
```

False değeri yani 0 ile sayi türünü topladığımızda

```
40
```

40+0 = 40 değerini verecektir.

➢ **Fark operatörü ile yapılan işlemler**

Sayı türü ile string türü arasında fark işlemi yapıldığında sayı değil (NaN) uyarısını verecektir.

```
<script>
var yazi="merhaba"//string türünde
var sayi=40;//Number türünde
alert(yazi-sayi);
</script>
```

```
NaN
```

Sayı türü ile Boolean türü arasında işlem yapıldığında

```
<script>
var dogrumu=true;//Boolean türünde
var sayi=40;//Number türünde
alert(sayi-dogrumu);
</script>
```

40-true (1)=39 değerini verecektir.

```
39
```

1.4. Operatörler ve İşlem Öncelik Sırası

1.4.1. Aritmetik Operatörler

➢ **Toplama operatörü (+)**

Bu operatör, sayısal ifadeleri ve mantıksal ifadeleri matematiksel olarak toplar.

```
<script>
var sayi1=10;
var sayi2=20;
document.write(sayi1+sayi2);//30 çıktısını verecektir.
</script>
```

```
<script>
var mantiksal1=true
var mantiksal2=true
document.write(mantiksal1+mantiksal2);// 2 çıktısını verir.
</script>
```

Toplama operatörü string ifadeleri yan yana yazdırır.

```
var yazi1="Mesleki ve Teknik Eğitim"
var yazi2="Genel Müdürlüğü"
document.write(yazi1+yazi2);//Mesleki ve Teknik EğitimGenel Müdürlüğü
</script>
```

- **Fark operatörü (-)**

Sadece matematiksel ifadelerde kullanılır. Çıkarma işlemi yapılır.

```
10  <script>
11  var sayi1=10;
12  var sayi2=20;
13  document.write(sayi1-sayi2);//-10 çıktısını verecektir.
14  </script>
```

- **Çarpma operatörü(*)**

Matematiksel çarpma işlemi yapılır.

```
10  <script>
11  var sayi1=10;
12  var sayi2=20;
13  document.write(sayi1*sayi2);//200 çıktısını verecektir.
14  </script>
```

- **Bölme operatörü (/)**

Bölme işlemi yapılır.

```
10  <script>
11  var sayi1=10;
12  var sayi2=20;
13  document.write(sayi1/sayi2); //0.5 çıktısını verecektir
14  </script>
```

- **Mod operatörü (%)**

Bölme işlemindeki kalanı bulmak için kullanılır.

```
10  <script>
11  var sayi1=100;
12  var sayi2=12;
13  document.write(sayi1%sayi2); //4 çıktısını verecektir
14  </script>
```

- **Arttırma operatörü (++)**

Sayının sonuna ya da başına eklenmesi durumunda sayının değerini 1 arttırır.

```
10  <script>
11  var sayi1=10;
12  sayi1++;
13  document.write(sayi1);//11 çıktısını verecektir.
14  </script>
```

- **Azaltma operatörü (--)**

Sayının sonuna ya da başına eklenmesi durumunda sayının değerini 1 azaltır.

```
<script>
var sayi1=10;
sayi1--;
document.write(sayi1);//9 çıktısını verecektir.
</script>
```

1.4.2. Atama Operatörleri

- **Atama operatörü (=)**

Sağ taraftaki değeri sol taraftaki değişkene atama işlemi yapar.

```
<script>

var sayi1 = 10;

</script>
```

- **Ekle ve ata operatörü(+=)**

Yığmalı toplam işlemi yapacaktır. Soldaki değişkenin değerini sağdaki değer kadar arttırıp değişkene yeni değer verir.

```
<script>
var sayi1 = 10;
document.write(sayi1);  //10 çıktısı verir
sayi1+=5;
document.write(sayi1);  //15 çıktısı verir
sayi1=sayi1+5;
document.write(sayi1);  //20 çıktısı verir
</script>
```

Böl ve ata(/ =) ,çıkar ve ata (- =), çarp ve ata (* =), mod al ve ata(%=) operatörleri de atama operatörleridir. Çalışma şekli, ekle ve ata ile aynıdır.

1.4.3. Karşılaştırma Operatörleri

Değerleri karşılaştırıp sonuç olarak boolean bir değer üretir. Akış denetimi yaparken sıklıkla kullanılır.

- **< (küçük), > (büyük), <= (küçük eşit) , >= (büyük eşit) Karşılaştırma operatörleri**

İki ifadenin karşılaştırmasını yapar, sonuç olarak boolean ifade elde edilir.

```
10  <script>
11  var sayi1 = 10;
12  var sayi2 = 20;
13  document.write(sayi1< sayi2);    //true
14  document.write(sayi1> sayi2);    //false
15  document.write(sayi1<=sayi2);    //true
16  document.write(sayi1>=sayi2);    //false
17  </script>
```

- **Eşittir operatörü = =**

Atama operatörü ile çok karıştırılır. Matematikteki eşitlik, programlamada = = ifadesi ile gösterilir. = = karşılaştırma operatörüdür.

```
10  <script>
11  var sayi1 = 10;
12  var sayi2 = 20;
13  document.write(sayi1==sayi2);    //false
14  </script>
```

- **Eşit değildir (!=)**

İfadelerin eşit olmaması durumunda true değerini döndürecektir.

```
10  <script>
11  var sayi1 = 10;
12  var sayi2 = 20;
13  document.write(sayi1!=sayi2);    //true
14  </script>
```

1.4.4. Mantıksal Operatörler

- **Mantıksal VE operatörü (&&)**

Koşullardan hepsinin gerçekleşmesi durumunu sınar yani şartlardan birisi gerçekleşmezse yanlış (false) değerini döndürecektir.

Aşağıdaki örnekte sayi1değişkeninin değerinin 10'dan küçük ve sayi2 değişkenin değerinin 3'ten büyük olması durumunda, ekrana şartlar doğru ifadesi yazdırılacaktır.

```
10  <script>
11  var sayi1 = 9;
12  var sayi2 = 8;
13  if(sayi1<10 && sayi2>3)
14  {
15      document.write("şartlar doğru")
16  }
17  else
18  {
19      document.write("şartlar yanlış")
20  }
21  </script>
```

- **Mantıksal VEYA operatörü (||)**

Koşullardan herhangi birisinin gerçekleşmesi durumunu sınar. Şartlardan birisinin gerçekleşmesi doğru (true) değerini döndürecektir.

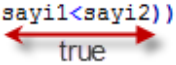
```
10  <script>
11  var sayi1 = 3;
12  var sayi2 = 8;
13  document.write(sayi1>5 || sayi2>5); // true
14  </script>
```

Sayi1 ve sayi2 değişkenin değerlerinden herhangi biri 5'ten büyükse ***true*** değerini döndürecektir.

- **Değil operatörü (!)**

İfadenin boolean karşılığını tersine çevirir.

```
10  <script>
11  var sayi1 = 3;
12  var sayi2 = 8;
13  document.write(!(sayi1<sayi2)); // false
14
15  </script>
```



1.4.5. İşlem Önceliği

JavaScript kodları çalıştırılırken aşağıdaki tabloya göre operatör öncelikleri belirlenir. Parantez ile bu önceliklerin sırasını değiştirebilirsiniz.

Tablonun üstten aşağıya doğru en düşük öncelikten en yüksek önceliğe doğru sıralanmıştır.

Operatör türü	Operatörler
Virgül(comma)	,
Atama	= += -= *= /= %= <<= >>= >>>= &= ^= |=
Koşul	?:
Mantıksal-veya	||
Mantıksal-ve	&&
Bit düzeyinde-veya	|
Bit düzeyinde-özel veya	^
Bit düzeyinde-ve	&
Eşitlik	== != === !==
İlişki	< <= > >= in instanceof
Bit düzeyinde öteleme	<< >> >>>
Ekleme/eksiltme	+ -
Çarpma/bölme/mod	* / %
Etkisiz duruma getirme/arttırma	! ~ – + ++ — typeof ,void ,delete
Çağırma/örnek yaratma	( ) new
Üyelik	. []

Tablo 1.3: Operatör öncelikleri

UYGULAMA FAALİYETİ

Html dökümanı oluşturmak, değişken tanımlamak, operatörleri değişkenlerle kullanmak ve sayfayı test etmek için aşağıdaki uygulamayı yapınız.

İşlem Basamakları	Öneriler
➢ Yeni html belgesi oluşturunuz.	➢ Editör programlarını kullanabilirsiniz.
➢ Html dokümanına script kodlarını yazmak için gerekli etiketleri yazınız.	➢ <script></script> etiketleri arasına yazılan her türlü kod JavaScript kodu olarak algılanır.
➢ Ad, soyad değerlerini tutacak bir değişken ismi bulunuz.	➢ Değişken ismi bulunurken Türkçe karakter kullanmamaya dikkat ediniz.
➢ Veri türü olarak metinsel veri türünü kullanınız.	➢ Çift tırnak içerisine yazılan değişken değerleri, değişkeni string yapacaktır.
➢ Değişkene kendi isminizi değer olarak atayınız.	➢ Atama yaparken = operatörünü kullanınız.
➢ Yaşınız bilgisini tutacak bir değişken ismi bulunuz.	➢ Değişken ismi belirlenirken en kısa ve en anlamlı kelimeyi bulmaya çalışınız.
➢ Yaş bilgisi tutan değişkene değer olarak yazı ile yazılmış yaş bilgisini veriniz.	➢ yas="onyedi" şeklinde otomatik tür dönüşümü yaptırabilirsiniz.
➢ İki değişkeni birbirine ekleyiniz.	➢ İçerisinde string ifade bulunan değişkenleri + operatörü ile birbirine ekleyebilirsiniz.
➢ Sayi isminde bir değişken tanımlayıp değerini bir arttırınız.	➢ ++ operatörlerini kullanarak ifadelere 1 ekleyebilirsiniz.
➢ Ad soyad, yas, sayi bilgisini tutan değişkenleri sayfaya yazdırınız.	➢ write metodu ile sayfaya değişkenleri yazdırabilirsiniz.
➢ Sayfayı test ederek çalıştırınız.	➢ Sayfayı test etmeden kaydetmeyi unutmayınız.

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için Evet, kazanamadığınız beceriler için Hayır kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

	Değerlendirme Ölçütleri	Evet	Hayır
1.	Html sayfada script kodlarını çalıştırabildiniz mi?		
2.	Ayrılmış kelimelerin neler olduğunu biliyor musunuz?		
3.	Değişken tanımlama kurallarını biliyor musunuz?		
4.	String türünden ifadeleri değişken değeri olarak kullandınız mı?		
5.	Atama operatörünü kullanarak isminizi değer olarak verdiniz mi?		
6.	Yazdığınız kodlar arasına açıklama satırları ekleyebildiniz mi?		
7.	İşlem önceliği sırasını öğrendiniz mi?		
8.	Atama operatörünün nasıl çalıştığını öğrendiniz mi?		
9.	++ operatörünün nasıl çalıştığını öğrendiniz mi?		

DEĞERLENDİRME

Değerlendirme sonunda "**Hayır**" şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız "**Evet**" ise "Ölçme ve Değerlendirme"ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

1. ( ) Betik dili; C,scheme, java, perl vb. dillerden etkilenmiştir.

2. ( ) Betikleri not defterinde düzenleyebilirsiniz.

3. ( ) Script kodları sayfa içerisinde üç bölümde kullanılır.

4. ( ) Değişken tanımlarken değişken isimleri sayı ile başlamak zorundadır.

5. ( ) for ismini değişken ismi olarak kullanabiliriz.

6. ( ) Yorum kodları uygulamanın akışını değiştirebilir.

7. ( ) Değil operatörü boolean değeri tersine çevirir.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ–2

AMAÇ

Bu faaliyet sonunda akış kontrol, dizi ve fonksiyon yapılarını kullanabileceksiniz.

ARAŞTIRMA

- Akış diyagramlarında kullanılan kontrol ve döngü işlemlerini araştırınız.

2. KODLAMA YAPISI

2.1. Akış Denetimi

Programın akışını karar ifadeleri ile denetleriz. Karar ifadeleri-belli bir şart sonucu-yazılan script kodlarının istediğimiz şekilde çalışmasını sağlar.

2.1.1. "if –else" Koşul Deyimi

Şartın doğru olup olmadığını denetler, şart doğruysa parantezler arasındaki kodları çalıştırır.

```
10  <script>
11  if(şart)
12  {
13  şart doğruysa çalıştırılacak kodlar
14  }
15  else
16  {
17  şart yanlışsa çalıştırılacak kodlar
18  }
19  </script>
```

Normal parantezler arasındaki ifade true değerini döndürüyorsa yani koşul ifadesi doğru ise küme parantezleri arasındaki kodlar çalışacaktır.

```
12  if(2==2)
13  {document.write("2 == 2")}
14  else
15  {document.write("2 != 2")}
```

Birden fazla şartın olduğu durumlarda şart sayısı kadar if yazılabilir.

Web sayfamızın ziyaretcilerine, saate göre günaydın, iyi öğlenler, iyi akşamlar, iyi geceler, iyi uykular uyarılarını veren script yazmak için;

- Editör programını açıp yeni bir html sayfası oluşturunuz.

- Zaman isminde bir date nesnesi oluşturunuz. Saat isminde bir değişken oluşturup bilgisayarın sistem saatinden gelen saat bilgisini saat isimli değişkene aktarınız.

```
5  var zaman = new Date();
6  var saat = zaman.getHours();
```

- Saat değişkeninin değeri artık bilgisayarın sistem saatine göre belirlenmektedir. İf kontrol deyimleri ile sayfanın açılışında kullanıcılara iyi dileklerinizi iletebiliriz. Şart tablosunu oluşturup script kodlarını yazmaya başlayabilirsiniz.

06-11	Günaydın
11-19	İyi öğlenler
19-23	İyi akşamlar
23-06	İyi geceler

Tablo 2.1: Şart tablosu

- Birinci şartım saat değişkeninin 6 değerinden büyük ve 11'den küçük olması. Şartların birden fazla olduğu durumlarda ve-veya operatörleri ile istediğiniz kontrolleri yaptırabilirsiniz.

```
14 if (saat>=6 && saat<11)
15 {
16 alert("günaydın");
17 }
```

- Diğer şartlar da tabloya uygun şekilde yazarak kodların yazımını tamamlıyoruz.

```
20 if (saat>=11 && saat<19)
21 {
22 alert("iyi öğlenler");
23 }
24 if (saat>=19 && saat<23)
25 {
26 alert("iyi akşamlar");
27 }
```

- 23'ten sonra bilgisayar saati 0 değerini aldığı için son şartım olan 23-06 arasını 6 değerinden küçük şeklinde değiştiriyor ve tek şart olarak kontrol deyimi içerisine yazıyoruz.

```
30 if (saat<6)
31 {
32 alert("iyi geceler");
33 }
```

İki sayıdan büyük olanını bulan uygulamayı hazırlamak için;

- Editör programını açarak yeni bir html sayfası oluşturunuz.
- Kullanıcıdan iki sayıyı sayfa üzerinden girmesini sağlamak için ***promt*** komutunu kullanacağız. Sayfa üzerinden gireceğimiz sayıları sayi1 ve sayi2 değişkenlerine atayacağız.

```
11  var sayi1=prompt("birinci sayıyı giriniz  :");
12  var sayi2=prompt("ikinci sayıyı  giriniz  :");
```

- Girdiğimiz sayıları artık kontrol deyimleri ile kontrol edebiliriz.

```
15  if(sayi1>sayi2)
16  {
17  alert("birinci girdiğin sayı büyük");
18  }
19  else
20  {
21  alert("ikinci girdiğin sayı büyük");
22  }
```

- Alert komutu ile uyarı ekranında hangi sayının büyük olduğunu yazdırıyoruz.

2.1.2. Ternary Operatörü

İf kontrol deyiminin yaptığı işlemi tek satırda yapmamızı sağlar. Genellikle kıyaslama işlemlerinde hızlıca ve yerden tasarruf etmek için kullanılır fakat kodlarda karmaşa yarattığı düşünülerek çok tercih edilmez.

Kullanımı çok kolaydır.

```
14  (şart)? şart doğruysa çalışacak kodlar : şart yanlışsa çalışacak kodlar;
```

Sayfadan girdiğimiz sayının tek mi çift mi olduğunu tespit eden script kodlarını yazmak için;

- Editör programında yeni bir html sayfası açınız.
- Sayıyı kendimiz gireceğimiz için promt komutunu kullanmalıyız. Prompt komutu ile girilen sayıyı, sayi isminde bir değişkene aktaralım.

```
11  var sayi=prompt("birinci sayıyı giriniz  :");
```

- Ternary operatörünü kullanarak kodlarımızı yazalım.

```
14  (sayi%2==0)?alert("sayı çift"):alert("sayı tek")
```

- Bir sayının çift olma şartı, sayının 2 ile tam bölünmesidir. Bu yüzden sayıya mod2 işlemi yaptık, sayının mod2 ile sonucu 0 ise sayı tam bölünüyor yani çift demektir.

2.1.3. "switch" Koşul Deyimi

Birden fazla koşul için gerekli kontrolü yapar fakat şartın boolean bir ifade döndürmesine bakmaz. Değer kısmına yazılan ile kontrol ifadesinin aynı olup olmamasını kontrol eder.

Switch kullanımında dikkat etmemiz gereken en önemli husus, case ifadesi içerisinde kodlarımızı yazdıktan sonra bir diğer case ifadesine geçiş yaparken break komutu kullanmamız gerekir.

```
30  switch(değer)
31  {
32  case "1. kontrol ifadesi ":
33  değer ifadeye eşitse çalışacak komutlar
34  break;
35  case "2. kontrol ifadesi ":
36  değer ifadeye eşitse çalışacak komutlar
37  break;
38  case "3. kontrol ifadesi ":
39  değer ifadeye eşitse çalışacak komutlar
40  break;
41  case "4. kontrol ifadesi ":
42  değer ifadeye eşitse çalışacak komutlar
43  break;
44  case "5. kontrol ifadesi ":
45  değer ifadeye eşitse çalışacak komutlar
46  break;
47  .
48  .
49  .
50  default:
51  değer ifadelerin hiç birine eşit değilse çalışacak kodlar.
52  break;
53  }
```

Yazdığımız yazıyı istediğimiz renkle ekrana yazdıran uygulama hazırlamak için ;

- Yeni bir html sayfası oluşturunuz.
- Öncelikle renk seçimini kullanıcının yapmasını sağlayalım. Promt komutu ile kullanıcıya seçebileceği renklerin listesini hazırlayalım.

```
11  var renk=prompt("kırmızı için :1 yeşil için :2 mavi için :3");
```

- Kullanıcı istediği renk için seçim yaptıktan sonra ekrana yazdırmak istediği yazı için yazi isminde bir değişken daha tanımlayınız.

```
14  var yazi=prompt("yazıyı giriniz.");
```

- Switch kullanarak kullanıcıdan gelen seçime göre ekrana yazımızı yazdıralım.

```
16  switch(renk)
17  {
18  case "1":
19  document.write("<font color='red'>"+yazi+"</font>");
20  break;
21
22  case "2":
23  document.write("<font color='green'>"+yazi+"</font>");
24  break;
25
26  case "3":
27  document.write("<font color='blue'>"+yazi+"</font>");
28  break;
29  }
```

- Document.write komutu parantezin içeriğini ekrana yazdırmamızı sağlar. Html etiketlerini yazdırmak istersek komutları yazmamız yeterlidir. Etiketlerin parametrelerini yazarken tek tırnak kullanmak, hata oluşmasını engelleyecektir.

2.2. Döngü Deyimleri

İstediğimiz sayıda tekrar eden işlemleri bilgisayara yaptırmak için kullandığımız kodlara ***döngü*** denir. Örneğin 1000 satırlık bir tabloyu oluşturmak için defalarca satır komutu yazmak yerine döngü kurmak çok mantıklı olacaktır.

2.2.1. "for" Döngü Deyimi

Koşul sağlandığı sürece komutların tekrar tekrar çalıştırılmasını sağlar. For döngüsü için bir döngü değişken tanımlamalıyız. Döngü değişkeninin başlangıç değerinden itibaren değişken artış miktarı kadar değerini değiştirir. Koşul doğru olduğu sürece döngü çalışır.

```
18  for(başlangıç değeri; koşul ; artış)
19  {
20  çalıştırılacak kodlar;
21  }
```

Ekrana 1'den 10'a kadar olan sayıları alt alta yazdıran script kodlarını yazmak için;

- Editör programını açarak yeni bir html sayfa oluşturunuz.

- For döngümüz için 1'den başlayarak 10'a kadar değeri değişecek bir değişken oluşturmamız gerekli değişkeni dışarıda tanımlayabileceğimiz gibi döngü içerisinde de tanımlayabiliriz.

```
13  for(var dongudegiskeni=1;dongudegiskeni<10;dongudegiskeni++)
14  {
15  document.write(dongudegiskeni+"<br>");
16  }
```

- Alt alta yazdırmak için
 html etiketini kullandık.

- Bu işlemi yapmak için 10 defa document.write komutunu kullanabilirdik fakat bizden 10 değil de 1000 sayı istenseydi 1000 defa komut yazmak çok zor olurdu.

Prompt komutu ile kullanıcı tarafından girilen sayı kadar kullanıcıdan aldığımız string ifadeyi yazdıran script kodlarını yazmak için;

- Kullanıcıdan girmesini istediğimiz sayıyı *sayi* değişkenine, string ifadeyi de *txt* değişkenine atayalım.

```
11  var sayi=prompt("döngü sayısını giriniz    :");
12  var  txt=prompt("döngü ifadesini giriniz   :");
```

- Döngü değişkeni olarak *dd* kısaltmasını kullanalım. dd, 1'den başlayarak girilen sayı kadar çalışsın.

```
14  for(var dd=1;dd<sayi;dd++)
15  {
16  document.write(txt);
17  }
```

- Sayfayı çalıştırdığımızda girilen sayı kadar girilen ifadeyi ekrana yazdıracaktır.

2.2.2. "While" Döngü Deyimi

Şartı sağladığı sürece while döngüsü çalışacaktır. Genellikle çalışma sayısının bilinmediği durumlarda while kullanılır.

```
25  while(şart)
26  {
27  çalıştırılacak kodlar
28  }
```

şeklinde kullanılır.

Klavyeden 0 sayısı girilene kadar girilen sayıları toplayan script kodlarını yazmak için;

- Editör programını açarak yeni bir html sayfası oluşturunuz.

- Sayıların toplamını tutmak için *toplam*, kaç sayının toplandığı bilgisi için *say*, girilen sayıları tutmak içinde *sayi* değişkenlerini tanımlayıp ilk değerlerini verelim.

```
12  var toplam=0;
13  var say=1;
14  var sayi;
```

- While ile döngümüzü kuralım. Şartımız girilen sayının sıfırdan farklı olması yani sayi değişkeni sıfırdan farklı olduğu sürece döngü çalışsın.

```
17  while(sayi!=0)
18  {
```

- Prompt komutu ile kullanıcıya bir sayı girmesi için iletişim penceresi oluşturalım. Kaçıncı sayıyı girmesi gerektiğini say değişkeni ile komutun içerisinde göstereceğiz.

```
20  sayi=prompt(say+". sayı giriniz.   :");
```

- Girilen sayıları yığmalı toplam yaparak toplam değişkenin içerisine yazdıracağız. Say değerini arttırıyoruz.

```
22  toplam=toplam+Number(sayi);
23  say++
24  }
```

- 0 rakamı girildiğinde döngüyü bitirip ve ekranda sayıların toplamını yazdırıyoruz. Yazdırma komutunu döngü içerisinde yazarsak her girilen sayıdan sonra sayfaya toplamı yazdırır ama döngü bittiğinde yazılırsa tüm sayılar toplanıp son toplam değeri sayfaya yazdırılacaktır.

```
26  document.write(say+" sayının toplamı :"+toplam);
```

2.2.3. "do-while" Döngü Deyimi

While döngüsünden tek farkı, şartın kontrolünün çalıştırılacak komutların en sonunda yapılmasıdır. Önce çalıştırılacak komutlar çalıştırılır, sonrada döngü şartı kontrol edilecektir. Her şartta döngü en az bir kere çalışacaktır.

```
12  do
13  {
14  çalıştırılacak kodlar
15  }
16  while(şart)
```

Sayfaya i değişkeninin 20'den küçük olduğu sürece ekrana merhaba yazdıracağı script kodlarını yazmak için;

- Yeni bir html sayfası açarak kodları yazınız.
- Document.write ile sayfamıza merhaba yazdıracağız.
- Dikkat etmemiz gereken i değerinin başlangıç değerini döngünün dışında vermek ve döngünün içerisinde i değerini arttırmaktır.

```
12  var i=0;
13  do
14  {
15  document.write("merhaba");
16  i++;
17  }
18  while(i<20);
```

2.3. Diziler

Aynı türdeki verileri tek yerde saklamamızı sağlayan değişkenlerdir. İçlerinde birden fazla değeri tutabilir.

2.3.1. Dizi Tanımlama

Dizi tanımlanın birden fazla yolu bulunmaktadır.

- Boş bir dizi oluşturup içeriğini sonradan ekleyebilirsiniz.

```
12  var dizi_ismi = new Array();
```

Dizi tanımlamak için var kelimesini daha sonra dizinin ismini yazmamız gereklidir. Yukarıdaki tanımlamada dizimizi oluşturduk fakat içerisine herhangi bir dizi elemanı atamadık.

Gunler isminde bir dizi oluşturup diziye değer girmek için;

```
12  var gunler = new Array();
13  gunler[0]="pazartesi";
14  gunler[1]="salı";
15  gunler[2]="çarşamba";
```

Komutları gunler dizisini oluşturdu, dizi elemanlarını tek tek 0.1.2. index numaralı yerlere yerleştirdik.

Diziler 0. değerden başlar.

- Dizilere değer vermenin ikinci yolu, diziyi oluştururken değeri aynı satırda vermektir.

```
12  var gunler = new Array("pazartesi","salı","çarşamba");
```

Değerler virgül ile birbirinden ayrılır. İsimler string değer olduğu için tırnak içerisine alınarak yazılır. Diziye string,number,boolean veri türlerini karışık olarak değer verilebilirsiniz. Dizi tanımlarken ve değer verirken tür sınırlaması yoktur.

Dizi tanımladıktan sonra alert ya da document sınıfının write metodu ile diziyi kontrol edebilirsiniz.

```
14  alert(gunler);          //pazartesi, salı, çarşamba
15  document.write(gunler); //pazartesi, salı, çarşamba
```

Dizilerin eleman sayısını bulmak için length komutunu kullanabiliriz. Length komutu string ifadelerinde karakter sayısını bulmak için kullanılabilir.

```
12  var gunler = new Array("pazartesi","salı","çarşamba");
13  alert(gunler.length);          //3 çıktısını verir
```

2.3.2. Çok Boyutlu Diziler

Dizilere eleman olarak değişik türde değerler atayabiliyorduk. Bir dizinin herhangi bir elemanına bir dizi değişkeni atarsak çok boyutlu bir dizi elde etmiş oluruz.

Örneğin ogrenci isimli bir diziye iki dizi elemanı ekleyelim:

```
12  var ogrenci = new Array();
13  ogrenci[0]=new Array("volkan","melek");
14  ogrenci[1]=new Array("web tasarım","grafik ve Animasyon");
```

Dizi elemanına ulaşmak istersek alert komutunu kullanabiliriz. Örneğin öğrenci dizisinin 1. kaydının 0. elemanına ulaşmak istersek

```
19  alert(ogrenci[1][0]);   //web tasarım
```

komutunu kullanabiliriz.

```
16  alert(ogrenci[0][0]);   //volkan
17  alert(ogrenci[0][1]);   //melek
18  alert(ogrenci[1][0]);   //web tasarım
19  alert(ogrenci[1][1]);   //grafik ve animasyon
```

2.3.3. Diziler Üzerinde İşlemler

- **Push( )**

Dizinin sonuna yeni bir dizi elemanı eklemek için kullanılır.

```
12  var sayilar = new Array("1","2","3","4");
13  sayilar.push("5");
14  sayilar.push("6");
15  alert(sayilar)  //1,2,3,4,5,6
```

- **Pop( )**

Dizinin sonundaki elemanı kaldırmak için kullanılır.

```
12  var sayilar = new Array("1","2","3","4");
13  sayilar.pop();
14  alert(sayilar);     //1,2,3
```

- **Unshift( )**

Dizinin başına eleman ekler. Birden fazla eleman bir kerede eklenebilir.

```
var sayilar = new Array("1","2","3","4");
sayilar.unshift("0");
alert(sayilar);      //0,1,2,3,4
```

- **Shift( )**

Dizinin başındaki elemanı kaldırır.

```
var sayilar = new Array("1","2","3","4");
sayilar.shift();
alert(sayilar);      //2,3,4
```

- **Delete**

Dizide istediğimiz elemanı silmek için kullanılır. Delete komutu ile dizi elemanı silinir fakat yeri dizi içerisinde kalır.

```
var sayilar = new Array("1","2","3","4");
delete sayilar[1]
alert(sayilar);      //1,,3,4
```

Dizinin 1 index numaralı elemanını için silme komutu verdik. Dizinin 1 index numaralı bölümünün içini boşaltmış olduk. Dizi elemanı kontrol edildiğinde dizinin yine 4 elemanlı olduğunu görürüz.

- **Splice( )**

Dizide istenilen bölüme eleman eklememizi ya da eleman silmemizi sağlar.

Dizi içerisinden eleman silmek için;

```
var sayilar = new Array("1","2","3","4");
sayilar.splice(0,2);
alert(sayilar);      //3,4
```

komutları kullanılır. Splice komutunda parantez içerisinde yazılan *ilk parametre* hangi dizi elemanından sonra silineceği, *ikinci parametre* ise kaç eleman silineceğini belirtir. Yukarıdaki kullanımda ad dizisinde 0. elemanından sonra 2 eleman siler. İkinci rakam kullanılmasaydı birinci rakamın gösterdiği dizi elemanından sonraki tüm elemanlar silinecekti.

Dizi içerisine eleman eklemek için;

```
var sayilar = new Array("1","2","3","4");
sayilar.splice(2,0,"bir","iki");
alert(sayilar);      //1,2,bir,iki,3,4
```

Yukarıdaki kullanımda splice komutu ikinci elemandan sonra "bir" ve "iki" kelimelerini eklemek üzere çalışacaktır. Eklemek ve silmek arasındaki fark ikinci rakamın 0 olarak belirlenmesidir. İkinci rakam, silinecek eleman sayısını belirlediği için slice, kelimeleri eklemek için çalışır.

- **indexOf**

Dizi içerinde geçen kelimeyi aramamızı sağlar. Uzun dizilerde kelimenin, dizinin kaçıncı elemanı olduğunu gösterir.

```
12  var harfler = new Array("a","b","c","d","e","f","g","h");
13  alert(harfler.indexOf("e"));
```

Alert komut 4 çıktısını verecektir. Diziler 0 index numarasından başlar e'nin bulunduğu yer 4 index numaralı yerdir.

Aranılan kelime dizi içerisinde yok ise -1 değerini döndürecektir.

- **Reverse ( )**

Dizi elemanlarını ters çevirmemizi sağlar.

```
12  var harfler = new Array("a","b","c","d","e","f","g","h");
13  alert(harfler);      //a,b,c,d,e,f,g,h
14  harfler.reverse();
15  alert(harfler);      //h,g,f,e,d,c,b,a
```

Dizi reverse komutundan önce tanımlandığı gibi ekrana yazdırılıyor. Reverse komutu ile birlikte dizi elemanları ters çevrilip yeniden listeleniyor.

- **Sort( )**

Dizi elemanlarını sıralamak için kullanılır. Komut kullanıldığında a-z, 0-9 aralıklarında sıralama işlemini yerine getirir.

```
12  var harfler = new Array("a","z","c","f","e","k","r","v");
13  var sayilar = new Array("3","5","2","1","7","6","8","9")
14  harfler.sort();
15  sayilar.sort();
```

Dizilerin içeriğini kontrol için alert komutlarını kullanabiliriz.

```
17  alert(harfler);      //a,c,e,f,k,r,v,z
18  alert(sayilar);      //1,2,3,5,6,7,8,9
```

2.4. Fonksiyonlar

Tekrarlanan işin yapılması için gerekli işlem ve komut gruplarına "fonksiyon" adı verilir. Uygulama içerisinde bir komut ya da komut grubunu tekrar tekrar kullanmak isteyebiliriz. Bu gibi durumlarda aynı satırları defalarca yazmak zorunda kalabiliriz. Tekrar

ettiğimiz kodlarda belli bir değeri değiştirmek istediğimizde tüm tekrar eden satırları yeniden düzenlemek gerekecektir.

Kısaca fonksiyonlar tekrar eden kod bloklarını alıp paketlememize yarar. Uygulama içerisinde artık bu paketin ismini kullanarak istediğimiz yerde bu kodları çalıştırabiliriz. Bu sayede daha sonra üstüne yapılacak değişiklikleri ve hata kontrolünü hızlı bir biçimde yapmamıza imkân sağlar.

2.4.1. Fonksiyon Tanımlamak

Fonksiyon tanımlamak için aşağıdaki kalıp uygulanır. Fonksiyon tanımlandığında editör penceresinin istenilen yerinde fonksiyon çalıştırılabilir.

- **Fonksiyonu tanımlamak için;**

```
10  function fonksiyon_ismi()
11  {
12  fonksiyon içerisinde çalıştırılacak kodlar
13  };
```

- **Fonksiyonu çalıştırmak için;**

```
15  fonksiyon_ismi();
```

şeklinde yazarak istenilen yerde fonksiyon çalıştırılır.

Sayfadan girilen ismi 20 defa alt alta yazdıran fonksiyon hazırlamak için;

- Editör programını açarak yeni bir html sayfası oluşturunuz.

- Fonksiyonumuzun ismi yazdir olsun. Prompt komutu ile dışarıdan alınan ifadeyi tutmak için isim değişkenini kullanalım.

```
11  function yazdir()
12  {
13  var isim=prompt("isminizi yazınız :");
```

- İsim değişkenini 20 defa alt alta yazdırmak için for döngüsü kullanıp fonksiyonu kapatalım.

```
17  for(var i=1;i<20;i++)
18  {
19  document.write(isim+"<br>");
20  }
21  };
```

- Fonksiyonumuz artık hazır. Fonksiyonun ismi ile sayfada istediğimiz yerde fonksiyonu çağırıp kullanabiliriz.

```
23  yazdir();
```

2.4.2. Fonksiyonlara Veri Gönderme-Alma

Yazdığımız fonksiyonlara isteğimize göre değer verip işlenmiş verinin sonucunu geri alabiliriz. Böylelikle fonksiyonlarımız anlık olarak verileri işleyip sonucu programa dâhil edebilir.

Fonksiyona parametre göndermek için yapmamız gereken fonksiyon isminden sonra parantez içerisine gelecek veri için değişken ismi yazmak olacaktır.

Kullanıcının girdiği yaş bilgisine göre kaç gün yaşadığını hesaplayan script kodlarını yazmak için;

- Yeni html sayfası oluşturun. Fonksiyonumuzu *head* etiketleri arasında tanımlayıp *body* etiketleri arasında kullanacağız.

- Kullanıcıdan isim ve yaş bilgilerini almak için *x* ve *y* değişkenlerini kullanacağız.

- *Prompt* komutunu kullanarak bilgileri değişkenlere atayalım.

```
19  var x = prompt ( "Lütfen isminizi giriniz : ", "" );
20  var y = prompt ( "lütfen yaşınızı giriniz.: ", "" );
```

- Fonksiyonumuzu *head* etiketleri arasında yazmaya başlayalım. Fonksiyonumuz ad ve yas olarak iki parametre alacak.

```
5   <title>Yaş Hesap Uygulaması</title>
6   <script type="text/javascript">
7   function ayBul(ad,yas)
8   {
```

- Fonksiyona parametre olarak girilen yas bilgisini gün sayısına çeviriyoruz.

```
10  var gun = yas * 12 * 30;
```

- Write metodu ile sayfaya yazdırıyoruz ve fonksiyonu kapatıyoruz.

```
12  document.write ("Merhaba "+ad+", Toplamda  " + gun+ " gün yaşamışsınız ");
13  }
```

- Body etiketleri arasında fonksiyonu ismi ile çağırarak çalışmasını sağlıyoruz.

```
26  ayBul(x,y);
```

Yukarıdaki örnekte fonksiyon, kullanıcıdan aldığı bilgileri işleyip ekrana çıktı olarak vermektedir. Fonksiyonun çalışması sonucu bir veri almak istiyorsak *return* kelimesini fonksiyon içerisinde kullanmak zorundayız.

Örneğin girilen sayının karesini hesaplayan ve uygulamaya sayısal değer olarak dâhil eden bir fonksiyon yazalım.

- Script etiketleri arasında karesi isminde bir fonksiyon tanımlayalım. Fonksiyonumuz dışarıdan tek parametre alsın.

```
<script type="text/javascript">
function karesi(x)
{
```

- Fonksiyon, dışarıdan aldığı x değerinin kendi içerisinde karesini alma işlemi yapsın.

```
11  var kare=x*x;
```

- Hesapladığı kare değerini fonksiyon dışına veri olarak göndermek için return komutunu kullanalım.

```
13  return kare;
14  }
```

- Fonksiyonumuz hazır artık. Sayfada istediğimiz işlemde fonksiyonumuzu kullanabiliriz.

```
22  alert(karesi(2));       //4
23  alert(karesi(4));       //16
24  alert(karesi(5));       //25
```

2.4.3. Hazır Fonksiyonlar

- **parseInt( )**

String ifadenin sayısal kısmını alarak tam sayıya çevirir.

```
14  document.write(parseInt("1980volkan")); //1980
```

İfade ondalıklı ise parseInt fonksiyonu sadece tam olan kısmını alır.

```
16  document.write(parseInt("212.45"));     //212
```

String ifade sayısal bir değer ile başlamıyorsa parseInt fonksiyonunun döndüreceği değer NaN olacaktır.

```
18  document.write(parseInt("volkan1980")); //NaN
```

- **parseFloat( )**

parseInt fonksiyonundan farkı sayısal ifadeleri ondalıklı olarak da alabilir.

```
16  document.write(parseFloat("212.45"));   //212.45
```

- **String( )**

İfadeyi string ifade yapar.

```
14  var sayi1=10;
15  var sayi2=15;
16
17  var toplam1=sayi1+sayi2;
18  alert(toplam1);          //25
19
20  var toplam2=String(sayi1)+sayi2;
21  alert(toplam2);          //1015
```

- **Number**( )

String ifadeyi sayısal değer yapar

```
14  var sayi1="10";
15  var sayi2=15;
```

```
17  var toplam1=sayi1+sayi2;
18  alert(toplam1);          //1015
19
20  var toplam2=Number(sayi1)+sayi2;
21  alert(toplam2);          //25
```

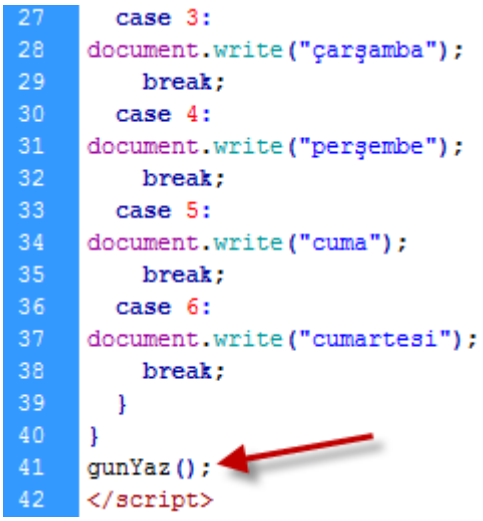
UYGULAMA FAALİYETİ

Web sayfasına hangi günde olduğumuzu yazdıran fonksiyonu switch yapısını kullanarak hazırlayınız.

İşlem Basamakları	Öneriler
➢ Yeni bir html sayfası oluşturunuz.	➢ <script></script> etiketlerini kullanınız.
➢ gunYaz isimli bir fonksiyon tanımlayınız.	➢ Fonksiyon tanımlarken function komutunu kullanabilirsiniz.
➢ Gün bilgisini tutmak için değişken tanımlayınız.	➢ gun değişken ismini kullanabilirsiniz.
➢ gun değişkenine sistem saatinden gün numarasını atayınız.	`var gun=new Date().getDay();` ➢ Date nesnesinden getDay özelliği ile gun değişkenine değer atayabilirsiniz.
➢ gun değişkenine göre ekrana günleri yazdıracak switch deyimini yazınız.	➢ switch deyimi gün değişkenine göre çalışmalıdır. `switch (gun)`

➢ gun değerinin alacağı değerler 0-6 arası olacaktır. Değere göre çalışacak switch komutlarını yazınız.	➢ Uygulama gun değeri 0 olduğunda ekrana pazar yazdırmalıdır.
	`case 0:`
	`document.write("pazar");`
	`break;`

➢ Switch deyimini diğer günler için tamamlayıp fonksiyonu kapatınız.	➢ Uygulamanın son hâli bu şekilde olmalıdır.
	`<script type="text/javascript">`
	`function gunYaz()`
	`{`
	`var gun=new Date().getDay();`
	`switch (gun)`
	`{`
	`case 0:`
	`document.write("pazar");`
	`break;`
	`case 1:`
	`document.write("pazartesi");`
	`break;`
	`case 2:`
	`document.write("salı");`
	`break;`

➢ Fonksiyon ismini script içerisine yazıp çalışmasını sağlayınız.	
➢ Sayfayı kaydederek çalıştırınız.	➢ Sayfada sistem saatinizin gösterdiği günü görmelisiniz. Sistem tarihini değiştirerek ekrandaki gün değerini değiştirebilirsiniz. Hazırladığımız uygulama sunucu tabanlı olsaydı sunucu bilgisayara göre işlem yapar ve gün değerini sistem tarihi ile değiştiremezdik.

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet**, kazanamadığınız beceriler için **Hayır** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri		Evet	Hayır
1.	if-else koşul deyimine göre uygulamayı yönlendirebiliyor musunuz?		
2.	Switch deyimi ile uygulamayı yönlendirebiliyor musunuz?		
3.	Ternary operatörünü kullanabiliyor musunuz?		
4.	Uygulamayı istenilen sayıda tekrar ettirebiliyor musunuz?		
5.	Uygulamayı şarta bağlı olarak tekrar ettirebiliyor musunuz?		
6.	Dizi oluşturup dizi elemanları üzerinde işlemler yapabiliyor musunuz?		
7.	Fonksiyon tanımlayıp veri gönderip alabiliyor musunuz?		
8.	Fonksiyonu uygulama içerisinde kullanabiliyor musunuz?		
9.	Hazır fonksiyonları uygulama içerisinde kullanabiliyor musunuz?		

DEĞERLENDİRME

Değerlendirme sonunda "**Hayır**" şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız "**Evet**" ise "Ölçme ve Değerlendirme"ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

1. ( ) Bütünü parçalara ayırmak için fonksiyonları kullanabiliriz.

2. ( ) Fonksiyon tanımlayarak aynı kod blokunu hızlı bir şekilde başka bölümlerde kullanabiliriz.

3. ( ) Ternary operatörü ile if deyiminin yaptığı işleri yapabiliriz.

4. ( ) Switch tek koşul için çalışır.

5. ( ) Delete ile dizi içerisinde istenilen eleman silinebilir.

6. ( ) Fonksiyon tanımlamak için function kelimesinden sonra fonksiyon ismi yazılmalıdır.

7. ( ) ParseInt fonksiyonu ifadenin ondalıklı kısmını alacaktır.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ–3

AMAÇ

Olaylarla ilgili düzenlemeleri yapabilecek ve sahnede olayları kontrol edebileceksiniz.

ARAŞTIRMA

- İnternetteki ActionScript ile yapılmış oyun ve uygulamaları inceleyip klavye ve fareye verdikleri tepkileri gözlemleyiniz.

3. OLAYLAR (EVENTS)

İnternet sayfamızdaki her şey olaylarla gözlenebilir. Farenin herhangi bir yere tıklaması, klavyeden bir şeyler yazılması, sayfanın yüklenmesi vb. eylemler olaylarla kontrol edilebilir. Olayların gerçekleşmesi ile yazılmış fonksiyonların tetiklenmesi sağlanır.

Olay yöneticileri, olay gerçekleştiğinde hazırladığımız fonksiyonları tetiklememizi sağlar.

Kullanımı basittir. Html etiketlerine olay yöneticisi eklerken etiketi kapatmadan

```
28  eventhandler="script kodları"
```

komut satırı eklenir.

Html etiketlerinin neredeyse tamamına uygulanabilir. Örneğin <a> etiketine olay yöneticisi eklersek yazım şu şekilde olacaktır:

```
28  <a eventhandler="script kodları">
```

3.1. onClick Olayı

Html etiketlerinin üstüne tek tıklandığında gerçekleşen olaylara onClick olayları denir.

Sahneye eklediğimiz düğmeye tıklama sonucu uyarı veren script kodlarını yazmak için;

- Yeni html sayfası oluşturunuz. Head etiketleri arasına uyari fonksiyonunu tanımlayınız.

```
6   <title>Olay Örnek Sayfası</title>
7   <script>
8   function uyari()
9   {
```

- Fonksiyonumuz ekrana "Düğme Tıklandı" uyarısını versin.

```
10  alert("Düğme Tıklandı");
11  }
12  </script>
```

- Fonksiyonumuz hazır. Artık body etiketleri arasına düğme oluşturmak için gerekli html komutlarını yazıp olayımızı düğme etiketinin içerisinde tanımlayabiliriz.

```
21  <input type="button" name="dugme" value="düğme" onclick="uyari()" />
```

- Sayfayı çalıştırınız. Sayfamızdaki düğmeye tıkladığımızda fonksiyonumuz çalışacak ve uyarı penceresinde *Düğme Tıklandı* uyarısını verecektir.

- Aynı işlemi link etiketi ile yapalım.

```
19  <a href="http://www.meb.gov.tr" onclick ="uyari()">MEB</a>
```

- Olaydan sonra fonksiyon yerine direk olarak script kodlarını yazabiliriz.

```
19  <a href="http://www.meb.gov.tr" onclick ="alert('meb.gov.tr açılıyor.')">MEB</a>
```

- Yukarıda hem sayfaya link sağlanıyor hem de ekranda bir uyarı penceresi oluşuyor.

3.2. onDblClick Olayı

Html etiketlerinin üstüne çift tıklandığında gerçekleşen olaylara onClick olayları denir.

```
23  <a href="http://www.volkangok.com" ondblclick="alert('Sayfa Açılıyor')">
24  Volkan GOK</a>
```

3.3. onLoad Olayı

Sayfamızdaki herhangi bir nesnenin tamamıyla yüklenme olayını temsil eder. **Body, img, frame** ve **frameset** etiketlerinde kullanılabilir.

```
16  <body onload="alert('sayfanın tamamı yüklendi')">
```

Sayfa tam yüklendiğinde yüklendi uyarısı veren onload olayını sayfanın yüklenme olayı olduğu için body etiketi içerisine yazdık. Resim yüklenmesine göre kontrol ettirseydik <img> etiketi içerisine yazmamız gerekirdi.

3.4. onUnLoad Olayı

Nesnenin kaldırılması olayını gözler. Body etiketi içerisine parametre olarak yazılırsa sayfanın kapatılma olayını gözlemiş oluruz.

```
16  <body onunload="alert('Yine Bekleriz.')">
```

3.5. onFocus Olayı

Sayfa üzerinde html elemanı işlem yapılmak üzere seçildiğinde yani o nesneye odaklanıldığında çalışan olay tipidir.

Örneğin bir metin kutusu içerisine yazı yazılmak üzere seçildiğinde uyarı vermesini sağlayabiliriz.

- Yeni bir html sayfası açarak head etiketleri arasına olay gerçekleştiğinde çalışacak fonksiyonu yazalım.

```
 6  <title>Olay Örnek Sayfası</title>
 7  <script>
 8  function uyari()
 9  {
10  alert("kullanıcı adını giriniz. ");
11  }
12  </script>
```

- Sayfanın gövdesinde form elemanını oluşturup onFocus olayını gerçekleştirelim.

```
18  <input type="text" name="kullaniciAdi" onfocus="uyari()" />
```

- Metin kutusunun içine yazı yazmaya çalıştığımızda artık uyarı verecektir.

3.6. onBlur Olayı

Seçilen html nesnesinin seçilme özelliğini kaybettiğinde çalışan olaydır. Seçilmiş eleman seçimden çıkartılmışsa tetiklenir.

3.7. onMouseOver Olayı

Html nesnesinin üstüne gelme olayını gözleyen olaydır.

Sayfada bulunan foto.jpg isimli resmin üzerine fare ile gelindiğinde uyarı veren script kodları aşağıdadır:

```
20  <img src="foto.jpg" height="50" width="50" onmouseover="alert('üstünde')" />
```

3.8. onMouseOut Olayı

Farenin nesne üzerinden ayrılması olayıdır.

3.9. onMouseMove Olayı

Fare nesne üzerinde gezdirildiğinde meydana gelen olaydır.

3.10. onMouseDown Olayı

Fare nesne üzerinde basıldığı anda gerçekleşen olaydır. Click ile farkı click olayının bırakılma eylemidir.

3.11. onMouseUp Olayı

Farenin basılı olan tuşu bırakması olayıdır.

3.12. onKeyDown Olayı

Klavyeden tuşa basma olayıdır. Metin kutusu nesnesinde klavyeden tuşa basıldığında olay dinleyicisi çalışır.

Klavyeden basılan tuşu ekranda göstermek için;

- Yeni bir html sayfası açınız.
- Klavyeden basılan tuşu tespit etmek için olayın event özelliğini ve keyCode komutunu kullanacağız.

```
19  <input onkeydown="alert(String.fromCharCode(event.keyCode))" />
```

- Event.keyCode basılan tuşun ASCII kodunu okuyacaktır. Örneğin boşluk(Spacebar) tuşuna bastığımızda keyCode bize 32 değerini verecektir.
- String.fromCharCode ile okunan ASCII değeri tuşun ismine çevirecektir.

3.13. onSelect Olayı

Metin kutuları içerisindeki yazı seçildiğinde meydana gelen olaydır.

```
19  <input type="text" value="Merhaba" onselect="alert('Yazıyı seçtiniz.')" />
```

3.14. onResize Olayı

Tarayıcı penceresinin boyutunu değiştirdiğinde gerçekleşen olaydır. Body etiketine yazılır.

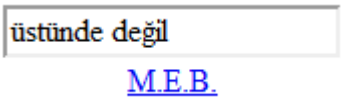
```
17  <body onresize="alert('tarayıcının boyutunu değiştirdiniz.')">
```

UYGULAMA FAALİYETİ

Link ile sahneye eklenmiş metin kutusunun içeriğini onClick, onMouseOver, onMouseOut, onMouseDown olayları ile değiştiren uygulamayı yapınız.

İşlem Basamakları	Öneriler
➢ Yeni bir html sayfası oluşturunuz.	➢ Not defterini kullanabilirsiniz.
➢ Sayfaya id değeri durum olan bir metin kutusu oluşturunuz.	`<input type="text" id="durum" /> `
➢ Metin kutusunun altına link ekleyip onClick, onMouseOver, onMouseOut, onMouseDown olayları için gerekli komutları yazınız.	`<a href="#" onClick="tiklandiVeBirakildi();" onMouseOver="ustunde();" onMouseOut="ustundeDegil();" onMouseDown="tiklandi();"> M.E.B.</a>`
➢ Olayların tetikleyeceği fonksiyonları oluşturunuz.	➢ Fonksiyonları <head></head> etiketleri arasında oluşturun. Tüm fonksiyonları script etiketleri arasına yazmalısınız.
➢ onClick olayının tetikleyeceği tiklandiVeBirakildi fonksiyonunu oluşturunuz.	➢ getElementById metodu bir sonraki öğrenme faaliyetinde ayrıntılı işlenecektir. Id değeri durum olan nesnenin değerini değiştirmek için kullanılmalıdır. `function tiklandiVeBirakildi() { document.getElementById('durum'). value="tıklandı ve bırakıldı"; }`
➢ onMouseOver olayının tetikleyeceği ustunde fonksiyonunu oluşturunuz.	`function ustunde() { document.getElementById('durum'). value="üstünde"; }`
➢ onMouseOut olayının tetikleyeceği ustundeDegil fonksiyonunu oluşturunuz.	`function ustundeDegil() { document.getElementById('durum'). value="üstünde değil"; }`

➢ onMouseDown olayının tetikleyeceği tiklandi fonksiyonunu oluşturunuz.	`function tiklandi()`
	`{`
	`document.getElementById('durum').`
	`value="tıklandı";`
	`}`

➢ Sayfayı kaydedip çalıştırınız.	➢ Aşağıdaki sayfa görünümü ile karşılaşmalısınız.  M.E.B.

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet**, kazanamadığınız beceriler için **Hayır** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri	Evet	Hayır
1. Html etiketlerine olay yöneticisi ekleyebildiniz mi?		
2. onClick olayını kullanabildiniz mi?		
3. onMouseOver olayını link etiketi ile kullanabildiniz mi?		
4. onLoad olayını sayfa içerisinde kullanabildiniz mi?		
5. onFocus olayını sayfa içerisinde kullanabildiniz mi?		
6. onUnLoad olayını sayfa içerisinde kullanabildiniz mi?		
7. onMouseOut olayını sayfa içerisinde kullanabildiniz mi?		
8. onMouseMove olayını sayfa içerisinde kullanabildiniz mi?		
9. onMouseDown olayını sayfa içerisinde kullanabildiniz mi?		
10. onKeyDown olayını sayfa içerisinde kullanabildiniz mi?		

DEĞERLENDİRME

Değerlendirme sonunda "**Hayır**" şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız "**Evet**" ise "Ölçme ve Değerlendirme" ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

1. ( ) OnkeyDown olayı klavye tuşları ile ilgili olayları gözler.

2. ( ) onClick ve onMouseDown olayları aynı işlemi yapar

3. ( ) Metin kutuları içerisindekileri seçtiğimizde onSelect olayı meydana gelir.

4. ( ) Fare nesnenin üzerinde dolaştırıldığında onMouseMove olayı meydana gelir.

5. ( ) Fare nesne üzerine tıklandığında onClick olayı meydana gelir.

6. ( ) onLoad olayı <a> etiketi için kullanılabilir.

7. ( ) Odaklanma meydana geldiğinde onLoad olayı meydana gelir.

8. ( ) Olaylar sonucu sadece olaylar için hazırladığımız fonksiyonlar kullanılabilir.

9. ( ) Tarayıcı penceresinin boyutunu onResize olayı ile değiştirebiliriz.

10. ( ) Farenin basılı olan tuşu bırakma olayı onMouseUp olayıdır.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ–4

AMAÇ

Animasyon yazılımını kullanarak Sınıf (class) işlemlerini gerçekleştirebilirsiniz.

ARAŞTIRMA

- Nesneye yönelik programlama nedir? Araştırınız.

4. NESNELER

JavaScript tüm web sayfalarını ve tarayıcılarını nesne olarak görür. Kendimiz nesne oluşturabileceğimiz gibi birçok işi kolayca yapabilmemiz için bünyesinde hazır nesneler de bulundurmaktadır.

Dizi işlemleri için array, tarih saat için date, aritmetiksel işlemler için math nesnesi vb. içlerinde birçok özelliği ve metodu barındırmaktadır. Bu sayede kolay ve hızlıca işlemlerimizi yapabiliriz.

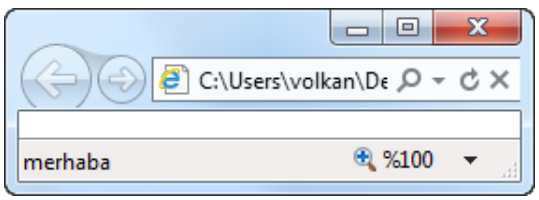
4.1. Window Nesnesi

Tarayıcı ile ilgili özellik ve metodun bulunduğu tarayıcı nesnesidir.

4.1.1. DefaultStatus Özelliği

Tarayıcının alt kısmındaki durum çubuğunda gösterilen mesajdır. Script komutları arasında herhangi bir yere yazılabilir. Tarayıcıların bir kısmı bu özelliği desteklememektedir.

```
12  <script>
13  window.defaultStatus="merhaba";
14  </script>
```



Şekil 4.1: DefaultStatus uyarısı

4.1.2. Alert Metodu

Daha önceki örneklerde defalarca kullandığımız alert, kullanıcıyı bilgilendirmek için uyarı pencereleri oluşturur. Gereksiz yere kullanmaktan kaçınmak gerekir.

Ekranda 10 defa uyarı kutusu oluşturacak script kodları için;

- Editör penceresini açarak yeni bir html sayfası oluşturunuz.
- Ekranda 10 kez uyarı vereceği için for döngümüzü kurarak alert komutunu döngümüz içerisine yazıyoruz.

```
14  for (var i=0;i<10;i++)
15  {
16  alert(i+". uyarı kutusu");
17  }
```

- Sayfamızı çalıştırdığımızda ekranda 10 adet uyarı penceresi oluşacak ve tek tek kapatmamız beklenecek.

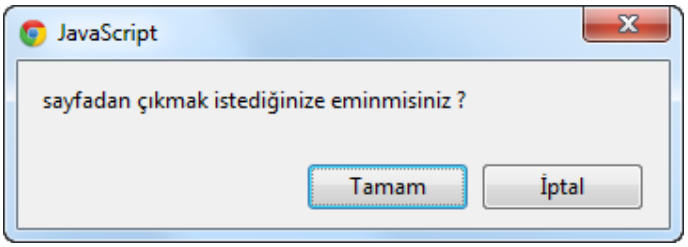
4.1.3. Confirm Metodu

Yapılan işlem sonucu onay kutusu oluşturur. Onay kutusunun sonucu bir boolean olarak geri döner. True ya da false şeklinde alınan cevap ile programın akışı belirlenir.

Sayfaya eklediğimiz kapat düğmesine basıldığında sayfayı kapatmak için onay isteyen script kodları için;

- Editör programını açarak yeni bir html sayfası oluşturunuz.
- Head etiketleri arasında, düğmeye basıldığında çalışmak üzere bir fonksiyon oluşturalım.
- Fonksiyonumuz düğmeye bastığında bir onaylama penceresi oluştursun. Kullanıcının vereceği cevaba göre sayfayı kapatıp kapatmamaya karar versin.
- Onaylama kutusu için karar isminde bir boolean değişken oluşturalım. Confirm kutusundan gelecek true ya da false bilgisi bu değişkene atanacaktır.

```
10  var karar=confirm('sayfadan çıkmak istediğinize eminmisiniz ?');
```



Şekil 4.2: Onay penceresi

Onay kutusunda tamam tıklandığı takdirde, karar değişkenimiz true, iptal tıklandığında false değerini alacaktır.

- Karar değişkenin alacağı değere göre if kontrol deyimini kullanarak sayfanın kapanıp kapanmama işlemini gerçekleştireceğiz. Kullanıcı tamam düğmesine basarsa ekranda "Yine Bekleriz" uyarısı çıkacak ve sayfa kendini kapatacaktır.

```
12  if(karar==true)
13  {
14  alert('Yine Bekleriz')  ;
15  window.close();
16  }
```

- Kullanıcı iptal düğmesine basarsa "Bizimle kaldığınız için teşekkürler" uyarısı ekranda çıkıp kullanıcıya teşekkür ediyoruz.

```
18  else
19  {
20  alert('Bizimle kaldığınız için teşekkürler');
21  }}
22  </script>
```

- Fonksiyonumuz hazır. Body etiketleri arasında düğmemizi oluşturup fonksiyonumuzu düğmemizin onclick olayına ekliyoruz.

```
28  <input type="button" name="s_kapat" value="Kapat" onclick="kapat()" />
```

- Sayfamızı çalıştırarak test edelim.

4.1.4. Close Metodu

Aktif olan sayfayı kapatmaya yarar.

```
16  window.close();
```

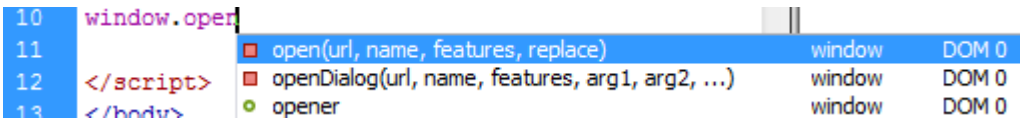
4.1.5. Prompt Metodu

Uygulamaya dışarıdan veri girmek için kullanılır. Promt ile alınan bilgi string türündedir. Herhangi bir değer girilmediği takdirde dönecek değer null olacaktır.

```
11  var gelen_deger=prompt("İsminiz Nedir ?");
```

4.1.6. Open Metodu

Yeni bir pencere açmak için kullanılır. Editör programı betik dili desteklediği için nesnenin özellik ve metotlarına ulaşmak çok kolaylaşır. Window yazıp nokta koyduğunuzda tüm metotlar karşınıza çıkacaktır.



Open metodu üç adet parametre alır.

- URL parametresi: Açılacak sayfanın adresini yazdığımız bölümdür.
- Name parametresi: Açılacak sayfanın target özelliği içindir. _blank,_self,_parent,_top ve kendi tanımladığımız herhangi bir ismi verebiliriz.
- Özellik parametresi: Açılacak sayfanın özelliklerini belirlediğimiz bölümdür. Buradaki parametreler ile sayfanın tam ekran açılması, boyutunun değiştirilip değiştirilemeyeceği, adres çubuğunun gösterilip gösterilmeyeceği gibi birçok özellik değiştirilebilir.

Buradaki parametreler birlikte kullanıldığı gibi tek başlarına da kullanılabilir. Örneğin sadece link oluşturmak için URL parametresi kullanılabilir ya da boş bir sayfa açmak için özellik parametreleri tek başlarına kullanılabilir.

Aşağıdaki örnekte sadece tanımlanan bağlantıyı açan bir fonksiyon oluşturulmuştur.

```
 6  <title>Sayfa Açma</title>
 7  <script type="text/javascript">
 8  function baglanti()
 9  {
10  window.open("http://mtegm.meb.gov.tr/");
11  }
12  </script>
```

Sayfaya eklenen düğmeye onclick olayı ile bağlanti fonksiyonu tanımlanmıştır.

```
18  <input type="button" value="MTEGM" onclick="baglanti()"/>
```

Open metodunun diğer özellikleri aşağıdaki tabloda verilmiştir:

Özellik	Açıklama
fullscreen=yes|no|1|0	Sayfanın tam ekran açılmasını sağlar.
height=pixels	Sayfanın yüksekliğini belirler. Piksel cinsindendir.
left=pixels	Açılan pencerenin sola göre mesafesini belirler.
location=yes|no|1|0	Adres çubuğunun gösterilip gösterilmeyeceğini ayarlar.
menubar=yes|no|1|0	Menü üzerindeki araç çubuğunun gösterilmesini belirler.
resizable=yes|no|1|0	Sayfanın yeniden boyutlandırılmasını ayarlar.
scrollbars=yes|no|1|0	Sayfadaki kaydırma çubuklarını ayarlar.
status=yes|no|1|0	Durum çubuğunun gösterilmesini ayarlar.
titlebar=yes|no|1|0	Başlık çubuğunun gösterilip gösterilmeyeceğini ayarlar.
toolbar=yes|no|1|0	İleri, geri, dur düğmelerinin gösterilmesini ayarlar.
top=pixels	Sayfanın yukarıdan mesafesini ayarlar.
width=pixels	Sayfanın genişliğini ayarlar.Piksel cinsindendir.

Tablo 4.1: Open metodu özellik tablosu

4.1.7. Print Metodu

Yazdırma penceresini açmaya yarar.

```
18  <input type="button" value="yazdır" onclick='window.print()'/>
```

4.1.8. Find Metodu

Arama penceresinin açılmasını sağlar. Çoğu tarayıcı bu metodu desteklememektedir.

```
18  <input type="button" value="bul" onclick="window.find();"/>
```

4.1.9. moveTo Metodu

Sayfayı ekran üzerinde istediğimiz koordinata taşımaya yarar.

200*200 px olarak açtığımız sayfayı istediğimiz yere götürmemizi sağlayan script kodları için;

- Editör programını açarak yeni bir html sayfası oluşturunuz.
- Yeni pencere açmak için open metodundan faydalanacağız.
- Sayfaya eklediğimiz bir düğme ile yeni bir pencere açalım. Pencereyi açarken isim verirsek diğer nesneler ile etkileşime sokmak kolay olacaktır. yeniPencere isimli bir pencere oluşturduk.

```
 9  function pencereAc()
10  {
11  yeniPencere=window.open('','','width=200,height=100');
12  yeniPencere.document.write("bu sayfa yeni oluşturuldu");
13  }
14  </script>
```

- yeniPencere isimli nesnemize artık write metodu ile sayfa üzerine yazı yazdırabiliriz.
- Sayfamıza düğme ekleyip oluşturduğumuz fonksiyonu düğmeye event olarak atayalım.

```
21  <input type="button" value="AÇ" onclick="pencereAc()"/>
```

- Pencereyi taşımak için moveTo metodundan faydalanacağız. moveTo metodunu bir fonksiyon içerisinde tanımlayıp düğmeye fonksiyonu atayabiliriz ya da direkt olarak düğmenin onclick olayına kodları tanımlayabiliriz.

```
23  <input type="button" value="TAŞI" onclick="window.moveTo(400,300)" />
```

4.2. Navigator Nesnesi

Tarayıcı hakkında kullanıcıya bilgi verir.

4.2.1. appName Özelliği

Sayfanın açıldığı tarayıcı ismini verir.

```
27  document.write("Tarayıcınızın İsmi: " + navigator.appName);
```

şeklinde kullanılır.

4.2.2. appCodeName Özelliği

Tarayıcının kod adını verir.

```
27  document.write("Tarayıcınızın İsmi: " + navigator.appCodeName);
```

4.2.3. appVersion Özelliği

Tarayıcının versiyonunu verir.

```
27  document.write("Tarayıcınızın yazılım Versiyonu:" + navigator.appVersion);
```

4.2.4. BrowserLanguage Özelliği

Tarayıcının dilini verir.

```
27  document.write("Tarayıcınızın dili:" + navigator.browserLanguage);
```

4.2.5. javaEnabled Metodu

Tarayıcıda java ayarının açık olup olmadığını kontrol eder.

```
27  document.write("Tarayıcının Java Ayarı: " + navigator.javaEnabled());
```

4.3. Document Nesnesi

Tüm html dokümanı document nesnesi ile ifade edilmektedir. Sayfa içerisindeki herhangi bir elemana document nesnesi ile erişebilir özelliklerini değiştirebiliriz.

4.3.1. getElementById Metodu

Sayfa içerisinde id özelliği verilmiş herhangi bir elemanın özelliklerine ulaşmak için kullanılır.

Sayfada bulunan metin kutusunun içeriğini istenilen şekilde değiştiren script kodları için;

- Editör programını açarak yeni bir html sayfası oluşturunuz.
- Metin kutusunu oluşturunuz. Metin kutusu oluşturulurken id etiketini vermemiz gereklidir. getElementByid metodunu kullanabilmemiz için özelliği değiştirilecek html elemanının mutlaka bir id değeri olması gerekir.

```
22  <input type="text" name="txt" value="" id="txt_kutu"/>
```

- İçi boş bir metin kutusu oluşturduk. Düğme oluşturup düğmenin tıklanma olayına metodu tanımlayalım. Metin kutusu içerisine yazılacak bilgiyi prompt ile kullanıcıdan alalım.

```
24  <input type="button" value="değiştir" onclick="
25  document.getElementById('txt_kutu').value=
26  prompt('kutuya yazılacak bilgiyi yazın')" />
```

- 24. satırda düğmemizi oluşturduk ve onClick olayını yazmaya başladık. 25. satırda id değeri txt_kutu olan elemanın value değerini kullanıcıdan prompt metodu ile alınan bilgi olarak değiştirdik.

4.3.2. getElementByName Metodu

getElementById ile aynı şekilde çalışır. Tek farkı id bilgisi yerine name bilgisini kullanır. Bazı html etiketleri grup olarak aynı ismi alır (radio düğmeleri, checkbox) bu elemanlar için kullanılırken item( ) metodu ile bu elemanlara ulaşılabilir.

4.3.3. Write( ) Metodu

Sayfaya metinleri ve html içeriklerini yazdırmak için kullanılır.

4.4. Form Nesnesi

Kullanıcıdan bilgi almak, kullanıcılarla etkileşime geçmek, sitemize dinamik bir yapı katabilmek için kullanılan html elemanlarıdır.

4.4.1. Action Özelliği

Hazırladığımız form ile kullanıcıdan alınan bilgilerinin gönderileceği ve işleneceği adres için kullanılır. Genellikle sunucu taraflı programlama dilleri ile hazırlanmış (ASP, PHP, ASP.NET, JSP, ColdFusion vb.) bir dosyasıdır.

```
10  <form action="http://www.mtegm.meb.gov.tr/formisle.php">
```

4.4.2. Method Özelliği

Form içerisinden girilen bilgilerin gönderileceği sayfaya nasıl gönderileceğini belirler. İki şekilde gönderim yapar.

- **GET** ile gönderdiğinde adres satırında gönderilen bilgiler açık şekilde gösterilir.
- **POST** ile iletim yapıldığında bilgiler gizli olarak iletilir. Get'e göre oldukça güvenlidir.

4.5. Date Nesnesi

Betik dili ile tarih ve saat işlemleri için kullanılan metot ve özelliklerin bulunduğu nesnedir. Date nesnesi oluşturduktan sonra metotlarla nesne içerisinden bilgiler rahatlıkla alınabilir.

```
12  var tarih=new Date();
```

4.5.1. getDate( ) Metodu

Sistem saatinde ayın kaçıncı günü olduğu bilgisini verir.

```
12  var tarih=new Date();
13  document.write(tarih.getDate());
```

Çalışan kodlar sonucunda sistem saatine göre 1-31 arası bir değer ekrana yazdırılacaktır.

4.5.2. getDay( ) Metodu

Sistem saatine göre haftanın kaçıncı günü olduğu bilgisini verir.

```
12  var tarih=new Date();
13  document.write(tarih.getDay());
```

Pazar günü için 0 değerinden başlayarak cumartesi günü 6 değerinde son bulur.

4.5.3. getMonth( ) Metodu

0-11 arası ay bilgisini verir.

```
12  var tarih=new Date();
13  document.write(tarih.getMonth());
```

4.5.4. getFullYear( ) Metodu

Sistem saatine göre yıl bilgisini verir.

```
12  var tarih=new Date();
13  document.write(tarih.getFullYear());
```

4.5.5. getHours( ) Metodu

0-23 arası saat bilgisini verir.

```
12  var tarih=new Date();
13  document.write(tarih.getHours());
```

4.5.6. getMinutes( ) Metodu

0-59 arası dakika bilgisini verir.

```
12  var tarih=new Date();
13  document.write(tarih.getMinutes());
```

4.6. Math Nesnesi

Betik dilinin matematiksel işlemleri yapmak için kullandığı metot ve özelliklerin bulunduğu nesnedir.

4.6.1. Math Nesnesinin Özellikleri

Matematikte kullanılan evrensel sabit değerleri verir.

```
12  document.write(Math.E);//eular sabiti
13  document.write(Math.PI);//pi değeri
14  document.write(Math.LN2);//logaritma 2 değeri
15  document.write(Math.SQRT1_2);// 1 sayısının karekökü
16  document.write(Math.SQRT2); //2 sayısının karekökü
```

4.6.2. Random( ) Metodu

0 ile 1 arası rastgele sayı üretir.

```
12  document.write(Math.random());
```

4.6.3. Round( ) Metodu

Sayıyı yuvarlamak için kullanılır.

```
12  document.write(Math.round(12.45));// 12 çıktısını verir.
```

4.6.4. Pow(x,y) Metodu

x sayısının y. kuvvetini bulur

```
12  document.write(Math.pow(5,2));// 25 çıktısını verir.
```

4.6.5. Sqrt( ) Metodu

Sayının karekökünü bulur.

```
12  document.write(Math.sqrt(25));// 5 çıktısını verir.
```

UYGULAMA FAALİYETİ

Sayfa yüklendiğinde sayfa ile birlikte başka sayfalarında otomatik açılmasını sağlayan uygulamayı yapınız.

İşlem Basamakları	Öneriler
➢ Yeni bir html sayfası oluşturunuz.	➢ Tasarım editörlerini kullanabilirsiniz.
➢ Açılacak sayfalar için fonksiyon oluşturunuz.	➢ Fonksiyonu head etiketlerini arasında oluşturunuz.

➢ Fonksiyonun ismi sayfaac olsun.	`<script type="text/javascript">`
	`function sayfaac()`
	`{`

➢ Fonksiyonumuz çalıştığında http://www.meb.gov.tr -ve http://mtegm.meb.gov.tr sayfalarını açsın.	`window.open("http://www.meb.gov.tr/");`
	`window.open("http://mtegm.meb.gov.tr/");`
	`}`
	`</script>`

➢ Sayfa yüklendiğinde fonksiyonu çalıştırması için uygun olay yöneticisini sayfaya ekleyiniz.	➢ onload olay yöneticisini kullanmalısınız.
➢ Sayfayı kaydedip çalıştırınız.	➢ onload olayını body etiketine eklemelisiniz. `<body onload="sayfaac()">`
➢ Tarayıcınızın popup engelleyicisi açıksa uygulamanız çalışmayacaktır.	➢ Tarayıcı ayarlarından popup ayarlarını düzenleyebilirsiniz.

UYGULAMA FAALİYETİ

Düğmeye basıldığında rastgele 1-49 arası 6 adet rastgele sayı üreten uygulamayı hazırlayınız.

İşlem Basamakları	Öneriler
➢ Yeni bir html sayfası oluşturunuz.	➢ Tasarım editörlerini kullanabilirsiniz.
➢ Rastgele sayılar için fonksiyon oluşturunuz.	➢ Fonksiyonu head etiketlerini arasında oluşturunuz.
➢ Fonksiyonun ismi sayisal olsun.	<script type="text/javascript"> function sayisal() {
➢ Fonksiyon çalıştığında id değeri sayi1 olan html etiketinin içerisinde yazan değeri değiştirsin.	➢ innerHTML etiketin içeriğini değiştirmek için kullanılır. document.getElementById("sayi1").innerHTML
➢ 0-49 arası rastgele değer üretiniz.	➢ Random metodu 0-1 arası değer üretir. Üretilen sayıyı 49 ile çarparsanız 49 katı bir sayı elde edersiniz.
➢ Üretilen sayıyı yuvarlatıp tam sayı hâline getiriniz.	➢ Round( ) metodunu kullanabilirsiniz.
➢ Sayfaya id değeri sayi1-6 olan altı adet etiketi ekleyiniz.	➢ Etiketi body etiketleri arasına eklemeyi unutmayınız.
➢ Fonksiyon içerisine 6 etiket için altı ayrı komut yazınız.	➢ Fonksiyon içeriği aşağıdaki gibi olmalıdır: document.getElementById("sayi1").innerHTML =Math.round(Math.random()*49); document.getElementById("sayi2").innerHTML =Math.round(Math.random()*49); document.getElementById("sayi3").innerHTML =Math.round(Math.random()*49); document.getElementById("sayi4").innerHTML =Math.round(Math.random()*49); document.getElementById("sayi5").innerHTML =Math.round(Math.random()*49); document.getElementById("sayi6").innerHTML =Math.round(Math.random()*49);
➢ Fonksiyonu tetiklemek için düğme kullanınız.	➢ Input etiketi ile sayfaya düğme ekleyebilirsiniz. <input type="button" onClick="sayisal()" value="Tahmin ET">

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet**, kazanamadığınız beceriler için **Hayır** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri		Evet	Hayır
1.	Window nesnesini sayfada kullanabildiniz mi?		
2.	Math nesnesini sayfada kullanabildiniz mi?		
3.	Date nesnesini sayfada kullanabildiniz mi?		
4.	Navigator nesnesini sayfada kullanabildiniz mi?		
5.	Document nesnesini sayfada kullanabildiniz mi?		
6.	Form nesnesini sayfada kullanabildiniz mi?		

DEĞERLENDİRME

Değerlendirme sonunda "**Hayır**" şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız "**Evet**" ise "Ölçme ve Değerlendirme" ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

1. ( ) Tarayıcı ile ilgili özellik ve metotların bulunduğu nesne document nesnesidir.

2. ( ) Alert metodu uyarı pencereleri oluşturur.

3. ( ) Confirm metodu ile onay pencereleri oluşturulur.

4. ( ) Uygulamaya dışarıdan veri girmek için close metodu kullanılır.

5. ( ) Tarayıcının versiyonunu appVersiyon özelliği ile öğrenebiliriz.

6. ( ) Sayfa içerisinde id özelliği verilmiş bir etiketin içeriğini getElementById metodu ile değiştirebiliriz.

7. ( ) GetDate metodu 1-24 arası değer üretir.

8. ( ) GetMonth metodu ile 0-11 arası değer üretir.

9. ( ) Random( ) metodu 0-100 arası değer üretir.

10. ( ) Round( ) metodu ile girilen sayıların karesi alınır.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise "Modül Değerlendirme"ye geçiniz.

MODÜL DEĞERLENDİRME

Aşağıdaki soruları dikkatlice okuyunuz ve doğru seçeneği işaretleyiniz.

1. JavaScript yazılımı aşağıdaki hangi dilden etkilenmiştir?

A) .Net
B) Jscript
C) Perl
D) Objective-j

2. Aşağıdakilerden hangisi uygun değişken ismidir?

A) case
B) try
C) tree
D) const

3.

I. Sayı ile başlayamaz.
II. Kelimeler arası boşluk kullanılamaz.
III. Boşluk kullanılabilir.

Yukarıdakilerden hangisi ya da hangileri değişken oluşturma kurallarındandır?

A) Yalnız II
B) I - II
C) II - III
D) I – II - III

4. Yorum kodları ile ilgili aşağıdakilerden hangisi yanlıştır?

A) Yorum kodları uygulamanın istediğimiz yerine açıklama eklememizi sağlar.
B) İki şekilde açıklama satırları eklenebilir.
C) Yorum kodları uygulamanın işleyişini değiştirir.
D) /**/ ifadeleri çok satırlı açıklama satırları oluşturur.

5.

I. Atama
II. üyelik
III. ilişki
IV. eşitlik
V. koşul

Yukarıdaki operatörleri öncelik sırasına göre yüksek öncelikten düşük önceliğe doğru sıralayınız.

A) II-III-IV-V-I
B) I-II-III-IV-V
C) II-III-V-IV-I
D) I–II–III-V-IV

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

6. ( ) Program akışını karar ifadeleri ile denetleriz.

7. ( ) if-else kontrol deyiminde normal parantezler arasındaki ifade true değeri döndürüyorsa küme parantezleri arası kodlar çalışır.

8. ( ) Ternary operatörü if kontrol deyiminin yaptığı işi tek satırda yapar.

9. ( ) Splice metodu ile dizi elemanlarını silme ve ekleme yapabiliriz.

10. ( ) onBlur fare olaylarını kontrol eder.

11. ( ) Confirm metodu ile boolean bir değer döner.

12. ( ) Open metodu window nesnesine aittir.

13. ( ) Location özelliği ile sayfa tam ekran açtırılabilir.

14. ( ) Tarayıcı ismini appBrowser özelliği ile öğrenebiliriz.

15. ( ) Math nesnesi ile trigonometrik işlemler yapılabilir.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki modüle geçmek için öğretmeninize başvurunuz.

CEVAP ANAHTARI

ÖĞRENME FAALİYETİ-1'İN CEVAP ANAHTARI

1	**Doğru**
2	**Doğru**
3	**Doğru**
4	**Yanlış**
5	**Yanlış**
6	**Yanlış**
7	**Doğru**

ÖĞRENME FAALİYETİ-2'NİN CEVAP ANAHTARI

1	**Doğru**
2	**Doğru**
3	**Doğru**
4	**Yanlış**
5	**Doğru**
6	**Doğru**
7	**Yanlış**

ÖĞRENME FAALİYETİ-3'ÜN CEVAP ANAHTARI

1	**Doğru**
2	**Yanlış**
3	**Doğru**
4	**Doğru**
5	**Doğru**
6	**Yanlış**
7	**Yanlış**
8	**Yanlış**
9	**Doğru**
10	**Doğru**

ÖĞRENME FAALİYETİ-4'ÜN CEVAP ANAHTARI

1	**Yanlış**
2	**Doğru**
3	**Doğru**
4	**Yanlış**
5	**Doğru**
6	**Doğru**
7	**Yanlış**
8	**Doğru**
9	**Yanlış**
10	**Yanlış**

MODÜL DEĞERLENDİRMENİN CEVAP ANAHTARI

1	**C**
2	**C**
3	**B**
4	**C**
5	**A**
6	**Doğru**
7	**Doğru**
8	**Doğru**
9	**Doğru**
10	**Yanlış**
11	**Doğru**
12	**Doğru**
13	**Yanlış**
14	**Yanlış**
15	**Doğru**

KAYNAKÇA

- http://www.w3schools.com/jsref/default.asp (03.05.2012/13.00)
- https://developer.mozilla.org/en/JavaScript/Reference (08.05.2012/13.00)