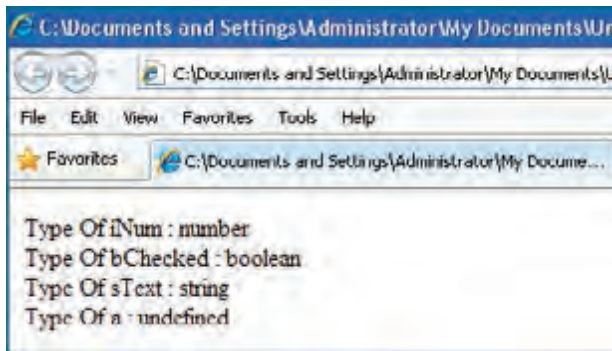


حال می‌خواهیم با تابعی آشنا شویم که یک متغیر را گرفته و نوع داده‌ای آن را برمی‌گرداند؛ این تابع جزو توابع پیش‌ساخته جاوا اسکریپت است و به صورت `typeof(var)` نوشته می‌شود.



خروجی کد زیر چیست؟

```
<body>
<script type="text/javascript">
var iNum=10, bChecked=false, sText="This is a js Code" , a ;
document.write("Type Of iNum: " + typeof(iNum) + "<br/>");
document.write("Type Of bChecked: " + typeof(bChecked) + "<br/>");
document.write("Type Of sText: " + typeof(sText) + "<br/>");
document.write("Type Of a: " + typeof(a) + "<br/>");
</script>
</body>
```



**بررسی کد:**

در این کد با استفاده از تابع `typeof()` نوع داده‌ای متغیرهای تعریف شده برگردانده شده که نتیجه برای متغیرهای `iNum`، `bChecked` و `sText` مطابق انتظار است. اما برای متغیر `a`، نوع `undefined` به معنی «تعریف نشده» برگردانده شده است. دلیل این مسأله را باید در مقداری که به متغیر نسبت داده شده جستجو کرد. همان‌گونه که در کد فوق می‌بینید، متغیر `a` تعریف شده اما هیچ مقداری به آن منتسب نشده بنابراین نوع داده‌ای آن برای برنامه مجهول مانده است.



قطعه کد زیر را به انتهای کد مثال قبل اضافه و نتیجه را در مرورگر مشاهده نمایید.

```
a=iNum;
document.write("Type Of a: " + typeof(a) + "<br/>");
```

```
Type Of iNum : number
Type Of bChecked : boolean
Type Of sText : string
Type Of a : undefined ←
Type Of a : number ←
```

### بررسی کد:

همان‌طور که در این مثال می‌بینید، نوع داده‌ای متغیر a ابتدا undefined است اما وقتی متغیر iNum را به آن نسبت می‌دهیم، نوع داده‌ای آن همانند این متغیر می‌شود و با اجرای دستور typeof()، عبارت number برگردانده می‌شود.

## ۲-۶-۱۹ انواع داده‌ای ارجاعی

وقتی صحبت از انواع داده‌ای ارجاعی به میان می‌آید، مفاهیم شیء‌گرایی و در رأس همه آن‌ها مفهوم کلاس (Class) خودنمایی می‌کند. انواع داده‌ای ارجاعی در جاوا اسکریپت عبارتند از:

■ کلاس **Boolean**: برای ایجاد یک شیء منطقی

■ کلاس **Number**: برای ایجاد شیء عددی

■ کلاس **String**: برای ایجاد شیء عددی

در بخش قبل با انواع داده‌ای اولیه و روش تعریف آن‌ها با استفاده از کلمه کلید var آشنا شدید. حال می‌خواهیم چگونگی تعریف یک نوع داده‌ای ارجاعی را فرابگیریم. در حالت کلی برای تعریف یک شیء از نگارش زیر استفاده می‌شود:

```
var o=new object();
```

var مشخص می‌کند که در حال تعریف یک متغیر هستیم و کلمه کلیدی new هم یک نمونه از کلاس object را ایجاد می‌کند تا به این ترتیب شیء 0 ایجاد شود. مزیت استفاده از انواع داده‌ای ارجاعی این است که به راحتی می‌توانیم به متدها و خصوصیت‌های تعریف شده برای آن‌ها دسترسی پیدا کرده و عملیات موردنظر را به سادگی انجام دهیم.



قصد داریم یک شیء عددی ایجاد نموده و آن را مقدار دهی کنیم.

```
var n=new Number();
n=100;
```



مقداردهی اولیه به اشیاء را می‌توانیم در حین تعریف هم به صورت زیر انجام دهیم:

```
var n=new Number(100);
```

اکنون شیء عددی ایجاد و با عدد ۱۰۰ مقداردهی شده است و می‌توانیم با کد زیر، مقدار آن را روی صفحه

بنویسیم:

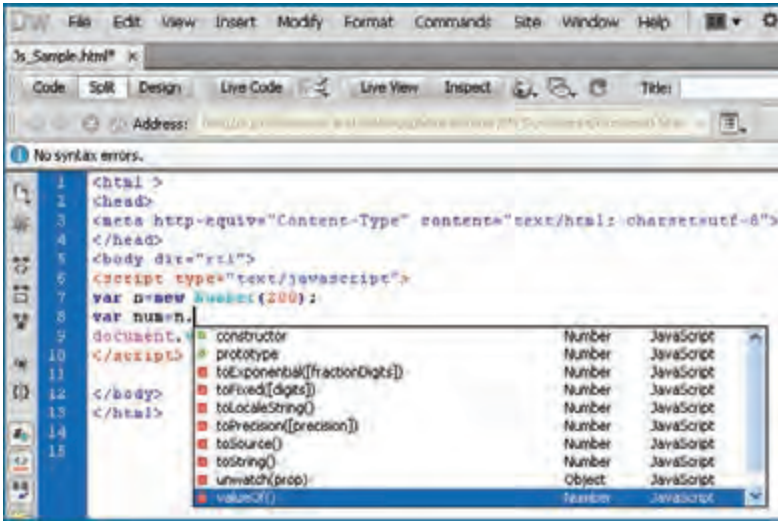
```
document.write(n);
```



مقدار یک شیء عددی را درون یک متغیر عددی قرار دهید.

ابتدا یک شیء عددی ایجاد و آن را با یک عدد مقداردهی می‌کنیم. حال برای انتساب مقدار این شیء به متغیر عددی از متد valueOf() استفاده می‌کنیم. هنگامی که این کد را درون محیط Dreamweaver می‌نویسید، با نوشتن نام شیء و تایپ نقطه، لیستی از متدها و خصوصیت‌های آن شیء ظاهر می‌شود.

```
<script type="text/javascript">
var n=new Number(200);
var num=n.valueOf();
document.write(num);
</script>
```



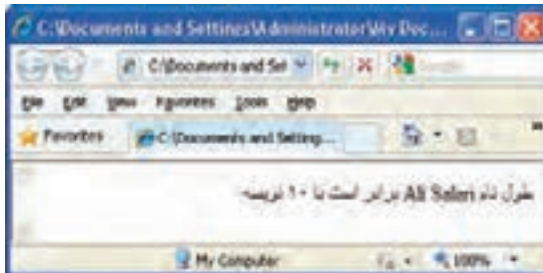
با نمایش صفحه فوق در مرورگر عدد ۲۰۰ نمایش داده می‌شود.



کدی بنویسید که ابتدا یک نام را ذخیره نموده و سپس نام را به همراه طول آن را روی صفحه وب بنویسد.

```
<script type="text/javascript">
var s=new String("Ali Salari");
```

```
document.write("طول نام "+ s + " برابر است با " + s.length + " نویسه ");
</script>
```



بررسی کد:

در این قطعه کد، ابتدا یک شیء رشته‌ای تعریف و با عبارت "Ali Salari" مقداردهی می‌شود. سپس محتوای رشته و نیز طول آن روی صفحه نمایش داده می‌شود. برای استخراج طول رشته از خصوصیت length

استفاده شده که جز خصوصیت‌های تعریف شده توسط جاوا اسکریپت برای کلاس String است. بدون وجود چنین خصوصیتی مجبور بودیم با نگارش یک قطعه کد، طول آن را محاسبه کنیم و این جاست که مزایای استفاده از روش شیء‌گرایی برای ایجاد برنامه آشکار می‌شود.



نتیجه اجرای کد زیر چیست ؟

```
<script type="text/javascript">
var b1=new Boolean(1)
var b2= b1.valueOf();
document.write(b2);
</script>
```

عبارت true روی صفحه ظاهر می‌شود چون شیء منطقی b1، با عدد ۱ (معادل true) مقداردهی شده است و سپس این مقدار درون متغیر منطقی b2 قرار گرفته و نهایتاً مقدار b2 روی صفحه نوشته شده است.



آیا اجرای کد زیر با خطا مواجه می‌شود؟

```
<body dir="rtl">
<script type="text/javascript">
var name="Hamid Razavi";
document.write(name.length);
</script>
```

در نگاه اول ممکن است این‌طور به نظر برسد که یک متغیر رشته‌ای (نه شیء رشته‌ای) تعریف شده و سپس از خصوصیت length آن استفاده شده است در صورتی‌که خصوصیت‌ها و متدها مربوط به انواع داده‌ای ارجاعی است نه اولیه! بنابراین اجرای کد باید با خطا مواجه شود اما نکته این جاست که جاوا اسکریپت حاوی سازوکاری درونی برای تبدیل ضمنی متغیرها به اشیاء است لذا شما می‌توانید یک متغیر رشته‌ای تعریف نموده و در آن از متدها و خصوصیت‌های اشیاء رشته‌ای استفاده نمایید.

## ۷-۱۹ تبدیل انواع داده‌ای

هر زبان برنامه‌نویسی قدرتمندی باید دارای سازوکاری کارآمد برای تبدیل انواع داده‌ای به یک‌دیگر باشد.

برای نمونه ممکن است هنگام بررسی مقادیر وارد شده در یک فرم وب، شماره ملی کاربر را به صورت رشته‌ای دریافت کنید اما لازم باشد برای ارزیابی آن، رشته دریافت شده تبدیل به عدد شود. به این کار تبدیل انواع داده‌ای یا Data Type Conversion گفته می‌شود.

### ۱-۷-۱۹ تبدیل به رشته

برای تبدیل اعداد یا مقادیر منطقی به رشته، می‌توانید از متد `toString()` که برای اشیاء عددی (و به تبع آن متغیرهای عددی) و نیز متغیرهای منطقی تعریف شده استفاده نمایید.



کد زیر چه مقداری را نمایش می‌دهد؟

```
<script type="text/javascript">
var n=100;
var s=n.toString();
document.write(typeof(s));
</script>
```

عبارت `string` را برمی‌گرداند چون عدد `n` با استفاده از متد `toString()` تبدیل به یک متغیر رشته‌ای شده و در `s` ذخیره گردیده است بنابراین اکنون نوع داده‌ای `s`، عبارت است از `string`.



خروجی کد زیر چیست؟

```
<script type="text/javascript">
var b =new Boolean(0);
document.write(b.toString());
</script>
```

عبارت `false` روی صفحه نوشته می‌شود چون شیء منطقی `b` با مقدار صفر (معادل `false`) مقداردهی شده است و سپس محتوای آن توسط متد `toString()` به رشته تبدیل و روی صفحه نمایش داده شده است.



کد زیر چه عبارتی را برمی‌گرداند؟

```
<script type="text/javascript">
```

```
var n = 7;
document.write(n.toString(2));
</script>
```

وقتی در پرانتز متد `toString()` عدد ۲ درج شود، نتیجه به صورت دودویی نمایش داده می‌شود و لذا به جای عدد ۷، معادل دودویی آن یعنی ۱۱۱ نمایش داده خواهد شد. عدد ۱۶ هم برای نمایش هگزادسیمال استفاده می‌شود.

## ۲-۷-۱۹ تبدیل به عدد

در جاوا اسکریپت برای تبدیل مقادیر رشته‌ای به عدد صحیح از تابع `parseInt(string)` استفاده می‌شود. تابع `parseInt` که در ادامه کتاب با روش نگارش و فراخوانی آن بیش‌تر آشنا خواهید شد، حاوی مجموعه‌ای از دستورات است که صفر، یک یا چند ورودی را پذیرفته و حداکثر یک خروجی را برمی‌گرداند.



کد زیر چه مقادیری را برمی‌گرداند؟

```
<script type="text/javascript">
document.write(parseInt("15") + "<br />");
document.write(parseInt("15.18") + "<br />");
document.write(parseInt("20 10 30") + "<br />");
document.write(parseInt(" 90 ") + "<br />");
document.write(parseInt("40 Books") + "<br />");
document.write(parseInt("Number 40") + "<br />");
</script>
```

مقادیر زیر برگردانده می‌شود:

15	
15	→ قسمت اعشاری حذف شده است
20	→ اولین عدد قبل از فاصله خالی تبدیل می‌شود
90	
40	
NaN	→ از عبارت Not a Number گرفته شده یعنی ورودی نامعتبر است

چنانچه می‌خواهید رشته موردنظر را به عدد اعشاری تبدیل کنید باید به جای تابع `parseInt()` از تابع `parseFloat()` استفاده نمایید.




با اجرای کد زیر چه اعدادی روی صفحه نمایش داده می‌شود؟


```
<script type="text/javascript">
document.write(parseFloat("20") + "<br />");
document.write(parseFloat("15.33") + "<br />");
document.write(parseFloat("34.5 11 18") + "<br />");
</script>
```


اعداد 20، 15.33 و 34.5 روی صفحه وب ظاهر می‌گردد.

### ۳-۷-۱۹ استفاده از توابع تبدیل<sup>۱</sup>

روش دیگر برای تبدیل انواع داده‌ای به یکدیگر از توابع تبدیل است. در جاوا اسکریپت سه تابع تبدیل زیر تعریف شده‌اند که مقدار دریافتی را به نوع داده‌ای متناظر تبدیل می‌کنند:

Boolean (value) 

Number (value) 

String (value) 



نتیجه اجرای کد زیر چیست؟

```
<script type="text/javascript">
var a1 = Boolean(1);
var a2 = Number(false);
var a3 = String(122.33);
document.write(a1 + "<br />");
document.write(a2 + "<br />");
document.write(a3 + "<br />");
</script>
```



عبارت‌های true، 0 و 122.33 نمایش داده می‌شود.

true	→	عدد 1 توسط تابع Boolean() تبدیل به مقدار منطقی true شده است
0	→	مقدار منطقی false توسط تابع Number() تبدیل به عدد 0 شده است
122.33	→	عدد درون پرانتز توسط تابع String() تبدیل به رشته شده است

## ۸-۱۹ شناخت عمل‌گرها

در جاوا اسکریپت همانند هر زبان برنامه‌نویسی یا اسکریپت‌نویسی دیگری، تعدادی عملگر<sup>۱</sup> وجود دارد که بر روی عملوند(ها)<sup>۲</sup>، عملیات خاصی را انجام می‌دهند. این عملیات می‌تواند انجام یک محاسبه ریاضی، اجرای یک مقایسه و ... باشد.

### ۱-۸-۱۹ عمل‌گرهای ریاضی

از عملگرهای ریاضی برای انجام محاسبات ریاضی ساده نظیر جمع، ضرب، محاسبه باقی‌مانده و ... استفاده می‌شود. در جدول زیر عملگرهای ریاضی قابل استفاده در جاوا اسکریپت را مشاهده می‌کنید.



مقدار اولیه متغیر X در همه ردیف‌ها برابر با عدد ۶ است.

عملگر	عملیات	مثال	نتیجه
+	جمع	$Y=X+2$	$Y=8$
-	منها	$Y=X-2$	$Y=4$
*	ضرب	$Y=X*3$	$Y=18$
/	تقسیم	$Y=X/4$	$Y=1.5$
%	باقی‌مانده	$Y=X\%5$	$Y=1$
++	افزایش یک‌واحدی	$Y=++X$	$Y=7$
--	کاهش یک‌واحدی	$Y=--X$	$Y=5$

1 . Operator  
2 . Operand



کدی بنویسید که هنگام باز شدن صفحه وب، یک عدد از کاربر گرفته شود و رقم سمت راست آن در یک پنجره پیغام به نمایش در بیاید.

```
<html ><head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
</head>
<body>
<script type="text/javascript">
var n= window.prompt(یک عدد صحیح وارد کنید);
window.alert("رقم سمت راست عدد وارد شده "+ n%10);
</script>
</body></html>
```

نتیجه اجرای کد را در تصویر زیر می‌بینید.



### بررسی کد:

ابتدا با استفاده از متد `window.prompt()` پنجره ورود عدد ظاهر می‌گردد. هنگامی که کاربر عدد را وارد و روی دکمه OK کلیک می‌کند، مقدار وارد شده درون متغیر `n` ذخیره می‌شود. سپس با استفاده از متد `window.alert()` باقی‌مانده این عدد بر ۱۰ (یعنی رقم سمت راست) به همراه یک پیغام نمایش داده می‌شود.



کد زیر چه نتیجه‌ای را برمی‌گرداند؟

```
<script type="text/javascript">
var x=11,y=15;
y=++x;
x=(y*x)/4;
document.write(x + "<br/>");
document.write(y + "<br/>");
</script>
```

مقدار اولیه  $x$  برابر با ۱۱ و  $y$  برابر با ۱۵ است. دستور  $++x$  مقدار متغیر  $x$  را یک واحد افزایش می‌دهد و بنابراین برابر با ۱۲ می‌شود. همین مقدار هم در متغیر  $y$  قرار می‌گیرد و مقدار قبلی آن یعنی ۱۵ پاک می‌شود. حال حاصل ضرب  $x*y/4$  (یعنی  $12*12/4$ ) درون  $x$  قرار می‌گیرد. بنابراین خروجی کد فوق

12

36

است.



خروجی کد زیر چیست؟

```
<script type="text/javascript">
var x=11,y=0, z=0;
y=x++;
z=++x;
++z;
document.write("X= " + x + "<br/>");
document.write("Y= " + y + "<br/>");
document.write("Z= " + z + "<br/>");
</script>
```

**بررسی کد:**

وقتی عملگرهای  $++$  و  $-$  بعد از یک متغیر قرار می‌گیرند، ابتدا عملیات انتساب با مقدار فعلی متغیر انجام می‌شود و سپس عمل افزایش یک واحدی انجام خواهد شد.

بنابراین در کد

```
y=x++;
```

ابتدا مقدار x (یعنی ۱۱) درون y قرار می‌گیرد و سپس مقدار x یک واحد افزایش می‌یابد. اکنون مقدار x برابر است با ۱۲. در ادامه، دستور

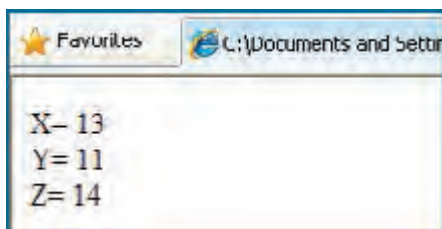
```
z=++x;
```

ابتدا مقدار x را یک واحد افزایش داده و سپس عدد ۱۳ را درون z قرار می‌دهد.

نهایتاً هم دستور

```
++z;
```

باعث افزایش یک واحدی مقدار z می‌شود. لذا خروجی کد فوق به صورت زیر است:



عملگر + این توانایی را نیز دارد که رشته‌ها را به یکدیگر متصل نماید. برای نمونه خروجی کد زیر:

```
var s1="Java" , s2="Script";
```

```
document.write(s1+s2);
```

عبارت JavaScript خواهد بود.

در فصول بعد هنگام بررسی اشیاء پیش‌ساخته جاوا اسکریپت، این کار را از طریق متدهای شیء String انجام خواهیم داد.



نتیجه اجرای کد زیر چیست؟

```
var x;
```

```
x=6+6;
```

```
document.write(x + " " + typeof(x) + "<br/>");
```

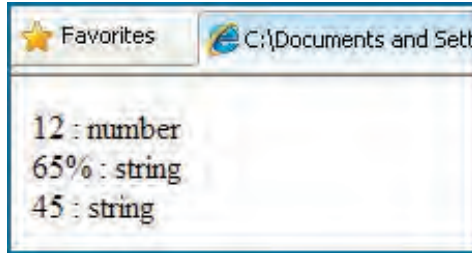
```
x="6"+"5"+"%";
```

```
document.write(x + " " + typeof(x) + "<br/>");
```

```
x=4+"5";
```

```
document.write(x + ": " + typeof(x) + "<br/>");
```

خروجی زیر روی صفحه وب ظاهر می‌شود.



نکته قابل توجه در کد این است که وقتی یک عدد با یک رشته جمع می‌شود، خروجی از نوع رشته خواهد بود.

## ۲-۸-۱۹ عمل‌گرهای انتساب

پیش از این با عملگر = که برای انتساب یک مقدار به یک متغیر مورد استفاده قرار می‌گیرد آشنا شدید. در جدول زیر با روش‌های دیگری از کاربرد این عملگر که برای خلاصه‌نویسی دستورات مورد استفاده قرار می‌گیرد آشنا می‌شوید.



مقدار اولیه متغیر X در همه ردیف‌ها برابر با عدد ۶ و مقدار Y برابر با ۱۲ است.

نتیجه	دستور معادل	مثال	عملگر
Y=6		Y=X	=
Y=18	Y=Y+X	Y+=X	+=
Y=6	Y=Y-X	Y-=X	-=
Y=72	Y=Y*X	Y*=X	*=
Y=2	Y=Y/X	Y/=X	/=
Y=0	Y=Y%X	Y%=X	%=



کد زیر چه نتیجه‌ای را برمی‌گرداند؟

```
<script type="text/javascript">
var x=4,y=5, z=6;
z%=x;
y+=z++;
document.write("Y= " + y + "<br/>");
document.write("Z= " + z + "<br/>");
</script>
```

دستور  $z\%=x$  به صورت  $z=z\%x$  تفسیر شده و عدد ۲ به عنوان باقی‌مانده تقسیم  $z$  بر  $x$  درون  $z$  قرار می‌گیرد. حال عبارت  $y+=z++$  به صورت  $y=y+z++$  در می‌آید یعنی ابتدا  $z$  با  $y$  جمع شده و درون  $y$  قرار می‌گیرد و سپس  $z$  یک واحد اضافه می‌شود. بنابراین خروجی به صورت زیر است:

Y = 7  
Z = 3



اگر در مثال قبل دستور  $y+=z++$  به صورت  $y+=++z$  نوشته شود چه تغییری در نتایج ایجاد خواهد شد؟ این بار نتایج زیر برگردانده می‌شود چون  $z$  قبل از جمع شدن با  $y$  یک واحد اضافه شده است

Y=8  
Z = 3



خروجی کد زیر چیست؟

```
var x=4,y=5, z=6;
x=y=z;
document.write("X= " + x + " , ");
document.write("Y= " + y + " , ");
document.write("Z= " + z );
```

مقدار  $z$  در  $y$  و مقدار  $y$  در  $x$  قرار می‌گیرد بنابراین عبارت زیر روی صفحه وب ظاهر خواهد شد.

X=6 , Y=6 , Z=6

### ۳-۸-۱۹ عمل‌گرهای مقایسه‌ای

عملگرهای مقایسه‌ای، دو عملوند را با هم مقایسه نموده و یک خروجی منطقی (true یا false) برمی‌گردانند. جدول صفحه بعد، عملگرهای منطقی را در زبان جاوا اسکریپت همراه با کاربرد آن‌ها نشان می‌دهد.



مقدار اولیه متغیر X در همه ردیف‌ها برابر با عدد ۶ است.

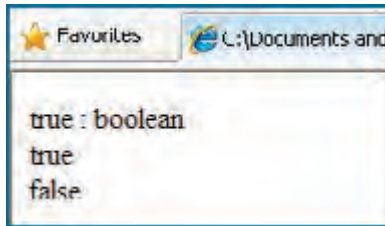
نتیجه	مثال	توضیح	عملگر
false	X==5	معادل است با	==
true	X=="6"		
true	X===6	از نظر نوع و مقدار معادل است با	===
false	X==="6"		
true	X!=5	معادل نیست با	!=
true	X>3	بزرگ‌تر است از	>
false	X<3	کوچک‌تر است از	<
true	X>=6	بزرگ‌تر یا مساوی است	>=
false	X<=5	کوچک‌تر یا مساوی است	<=



کد زیر چه مقادیری را برمی‌گرداند؟

```
<script type="text/javascript">
var x=10;
var result = (x*2) >= 20;
document.write(result + ": " + typeof(result) + "<br/>");
document.write(x==10);
document.write("<br/>");
document.write(x==="10");
</script>
```

نتایج زیر هنگام نمایش صفحه وب ظاهر می شود:



از آن جا که عبارت  $(x*2) \geq 20$  صحیح است، مقدار منطقی true در متغیر result قرار می گیرد. همچنین به دلیل صحیح بودن عبارت  $x == 10$  مقدار true برای درج توسط متد write ارسال می گردد. علاوه بر این چون عدد ۱۰ توسط عملگر  $===$  با رشته ۱۰ مقایسه شده، نتیجه نادرست است. بدیهی است اگر عبارت  $x === "10"$  به عبارت  $x === 10$  تبدیل شود، نتیجه این عبارت هم صحیح خواهد بود.

#### ۴-۸-۱۹ عمل گرهای منطقی

این نوع عملگرها که معمولاً برای بررسی نتیجه نهایی چند مقایسه مورد استفاده قرار می گیرند، یک خروجی منطقی (true یا false) برمی گردانند. در جدول زیر روش به کارگیری آن ها توضیح داده شده است.



مقدار اولیه متغیر X در همه ردیف ها برابر با عدد ۶ و مقدار Y برابر با ۱۰ است.

عملگر	کارکرد	مثال	نتیجه
&&	«و» منطقی	$X < 5 \ \&\& \ Y > 9$	true
	«یا» منطقی	$X >= 7 \    \ Y <= 9$	false
!	«نه» منطقی	$(x == y)!$	true



خروجی کد زیر چیست؟

```
<script type="text/javascript">
```



```
var x=10,y=12,z=false;
document.write((x>y) || (y*2)<10);
document.write("<br/>");
document.write(!z && y>x);
</script>
```

خروجی به صورت زیر است

```
true
false
```

چون عبارت  $x > y$  نادرست است و «یا»ی منطقی آن با عبارت نادرست  $(y * 2) < 10$  نادرست خواهد بود. همچنین عبارت  $!z$  درست و طبیعتاً «و» منطقی آن با عبارت درست  $y > x$ ، درست است.



کد زیر چه نتیجه‌ای را برمی‌گرداند؟

```
<script type="text/javascript">
var a,b,c;
a=(15%7==1);
b=(14!=2*7);
c=!((a && b) || a)
document.write(c==true);
</script>
```

**بررسی کد:**

از آن‌جا که باقی‌مانده ۱۵ بر ۷ برابر یک است بنابراین:  $a = \text{ture}$

عبارت  $b = 14 \neq 2 * 7$  نادرست است لذا:  $b = \text{false}$

$a \&\& b$  نادرست است اما «یا»ی منطقی آن با  $a$  صحیح می‌شود. اما چون قبل از پرانتز عملگر ! قرار

گرفته بنابراین:  $c = \text{false}$

در پایان هم، متد write نتیجه مقایسه  $c$  را با  $\text{true}$  بر می‌گرداند که طبیعتاً  $\text{false}$  خواهد بود.

## ۹-۱۹ تعریف و فراخوانی تابع

تا این‌جا کتاب با روش نوشتن کدهای جاوا اسکریپت و اجرای آن‌ها در حین بارگذاری صفحه آشنا شدید. حال اگر بخواهیم قطعه کدی، هم‌زمان با باز شدن صفحه به اجرا در نیاید و اجرای آن مشروط به وقوع یک رویداد مانند کلیک شدن دکمه باشد، آیا راه‌حلی وجود دارد؟

خوشبختانه پاسخ مثبت است. شما می‌توانید کد موردنظر را درون یک تابع بنویسید و فراخوانی تابع را منوط به وقوع یک رویداد کنید. این کار علاوه بر این که اجرای عملیات‌های متنوع و پیچیده را امکان‌پذیر می‌سازد، یک مزیت مهم دیگر هم دارد. عملیات‌ها، درون توابع پیاده‌سازی می‌شوند و در زمان موردنظر فراخوانی و اجرا می‌گردند؛ استفاده از این روش، کدهای نوشته شده را منظم‌تر خواهد کرد و ردگیری خطاهای احتمالی به سادگی انجام خواهد شد. علاوه بر این اگر لازم باشد عملیاتی درون یک صفحه چندین بار اجرا شود، لازم نیست کدها چندین بار تکرار شوند؛ بلکه صرفاً عملیات فراخوانی انجام خواهد گرفت.

تعریف تابع همانند سایر کدهای جاوا اسکریپت می‌تواند در بخش سرصفحه، بدنه صفحه وب و یا درون فایل خارجی با پسوند js صورت گیرد. برای ایجاد یک تابع باید مراحل زیر را انجام دهید:

- کلمه کلیدی function را بنویسید. توجه داشته باشید که این عبارت حتماً باید با حروف کوچک انگلیسی نوشته شود.

- نام منحصر به فردی را برای تابع تایپ کنید. سعی کنید نام تابع با کارکرد آن تناسب داشته باشد تا در بررسی کد دچار سردرگمی نشوید.

- پارامترهایی که تابع دریافت می‌کند را مشخص نمایید.

- بدنه تابع را درون علامت‌های { } بنویسید.

- مقداری را که باید توسط تابع برگردانده شود (در صورت وجود) تعیین نمایید.



می‌خواهیم تابعی بنویسیم که طول و عرض یک مستطیل را دریافت کند و مساحت آن را برگرداند. کد زیر را درون ویرایش‌گر متنی وارد کنید.

```
<html >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<script type="text/javascript">
function CalculateArea(width,height)
{
    var s= width*height;
    return (s);
}
</script>
```

} تعریف تابع

```

</head>
<body dir="rtl">
<script type="text/javascript">

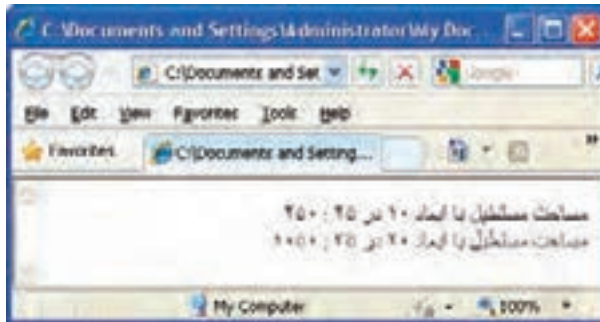
document.write("مساحت مستطیل با ابعاد ۱۰ در ۲۵: " + CalculateArea(10,25));
document.write("<br/>");
document.write("مساحت مستطیل با ابعاد ۲۰ در ۳۵: " + CalculateArea(30,35));
</script>
</body>
</html>

```

فراخوانی تابع

فراخوانی تابع

هنگام باز کردن صفحه درون مرورگر، عبارت زیر نوشته می‌شود.



در مثال قبل، هنوز هم فراخوانی تابع، هم‌زمان با بارگذاری صفحه انجام می‌شود. قصد داریم با تغییراتی در کد، این کار را مشروط به کلیک شدن یک دکمه کنیم. کد را به صورت زیر تغییر دهید:

```

<html >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<script type="text/javascript">
function CalculateArea(width,height)
{
    var s= width*height;
    return (s);
}
function WriteResults()
{

```

```

document.write("مساحت مستطیل با ابعاد ۱۰ در ۲۵ : " + CalculateArea(10,25));
document.write("<br/>");

document.write("مساحت مستطیل با ابعاد ۲۰ در ۳۵ : " + CalculateArea(30,35));
}
</script>
</head>
<body dir="rtl">
<input type="button" value="کلیک کنید" onClick="WriteResults()" />
</body>
</html>

```

در این صفحه، تابع دیگری با نام WriteResults() تعریف گردیده و وظیفه نوشتن عباراتی که در مثال قبل مشاهده کردید به آن واگذار شده است. همان طور که مشاهده می کنید، تابع می تواند فاقد پارامتر ورودی باشد و ضمناً مقداری را برنگرداند؛ این توابع عمدتاً برای انجام کاری خاص مثل نوشتن یک متن مورد استفاده قرار می گیرند. حال به بدنه صفحه دقت کنید. یک کنترل از نوع دکمه روی صفحه قرار گرفته و مشخصه onClick آن با نام تابع مقداردهی شده است. منظور از کد "onClick=functionName()" این است که وقتی رویداد onClick دکمه به وقوع پیوست (یعنی دکمه کلیک شد)، تابعی که نام آن مشخص شده فراخوانی شود. تابع WriteResults برای محاسبه مساحتها، دوبار تابع CalculateArea را فراخوانی می کند؛ بنابراین فراخوانی تابع از درون تابع دیگر امکان پذیر است.

این بار پس از نمایش صفحه وب، شما صرفاً یک دکمه را خواهید دید و برای فراخوانی تابع و مشاهده نتایج باید روی این دکمه کلیک نمایید.





صفحه‌ای ایجاد کنید تا کاربر با کلیک روی یک دکمه، پیغام مناسبی را دریافت کند و پنجره بسته شود.

```
<html >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<script type="text/javascript">
function CloseWindow()
{
    window.alert("منتظر دیدار مجدد شما هستیم");
    window.close();
}
</script>
</head>
<body dir="rtl">
<input type="button" value="بستن" onClick="CloseWindow()" />
</body>
</html>
```

در این مثال، رویداد onClick دکمه با نام تابع CloseWindow مقداردهی شده است و وقتی این تابع فراخوانی شود، کاربر پیغامی را مشاهده خواهد کرد و پس از کلیک روی دکمه OK، پنجره مرورگر بسته خواهد شد.

نکته قابل توجهی که در این مثال‌ها با آن آشنا شدید، مفهوم رویداد بود. هر یک از کنترل‌هایی که بر روی صفحه وب قرار می‌گیرند دارای رویدادهای مخصوص به خود هستند. خوشبختانه این رویدادها توسط جاوا اسکریپت شناسایی می‌شوند و با استفاده از کدهای جاوا اسکریپت می‌توانید پاسخ‌های مناسبی را برای وقوع این رویدادها ایجاد نمایید.

برای نمونه می‌توانید تابعی را به رویداد onMouseOver یک پاراگراف نسبت دهید تا وقتی کاربر، اشاره‌گر ماوس را روی نوشته‌های پاراگراف برد، عملیاتی مثل تغییر رنگ یا جابه‌جایی نوشته‌ها انجام شود. به این ترتیب می‌توان صفحاتی را ایجاد کرد که با کاربر تعامل برقرار نموده و صفحات وب را از محیطی ساده و یکنواخت به محیطی پویا تبدیل نمایند.

فعلاً برای پیگیری مثال‌های خود به رویداد onClick دکمه‌ها بسنده می‌کنیم اما در فصل‌های بعدی کتاب، رویدادهای کنترل‌های صفحات وب را به صورت دقیق‌تر و با ذکر مثال بررسی خواهیم نمود تا توانایی شما برای کدنویسی جاوا اسکریپت به حد مطلوب برسد.



## چکیده‌ی فصل



- متغیر، مخزنی برای ذخیره داده‌ها در طول اجرای برنامه است.
- برای نام‌گذاری متغیرها نباید از کلمات کلیدی یا رزرو شده استفاده نمود.
- برای مقداردهی به متغیرها از عملگر = استفاده می‌شود.
- انواع داده‌ای در جاوا اسکریپت به دو دسته اولیه و ارجاعی تقسیم‌بندی می‌شوند.
- برای تبدیل رشته به عدد صحیح می‌توان از متد `parseInt()` و برای تبدیل عدد به رشته از متد `toString()` استفاده نمود.
- برای انجام عملیات‌های موردنظر می‌توان از عملگرهای ریاضی، انتساب، مقایسه‌ای و منطقی استفاده نمود.
- تعریف تابع در برنامه باعث صرفه‌جویی در زمان کدنویسی و نیز تسهیل اشکال‌زدایی برنامه می‌شود.

## پرسش‌ها و تمرین‌ها



۱. آیا استفاده از کلیدواژه `var` برای تعریف متغیرها اجباری است؟
۲. کدام‌یک از موارد زیر برای نام‌گذاری متغیرهای جاوا اسکریپت معتبر نیست؟
  - `_12a`
  - `var`
  - `2abc`
  - `&sss`
  - `num`
  - `return`

۳. تفاوت متغیرهای محلی و سراسری را توضیح دهید.

۴. انواع داده‌ای اولیه را با ذکر مثال شرح دهید.

۵. تابع `typeof` چه کاربردی دارد؟

۶. کد زیر چه مقداری را برمی‌گرداند؟ چرا؟

```
<script type="text/javascript">
var b =new Boolean(100);
document.write(b.toString());
</script>
```

۷. کدی بنویسید که سال تولد کاربر را دریافت نموده و سن وی را نمایش دهد.

۸. کد زیر چه مقداری را برمی‌گرداند؟

```
<script type="text/javascript">
var x=15,y=2, z=6;
y=x--;
z=++x;
++z;
document.write("X= " + x + "<br/>");
document.write("Y= " + y + "<br/>");
document.write("Z= " + z + "<br/>");
</script>
```

۹. خروجی کد زیر چیست؟

```
var x=15,y=11,z=true;
document.write((x<y) || (y*3)>38);
document.write("<br/>");
document.write(!(z && y>x));
```

۱۰. صفحه‌ای عبارت «خروج» ایجاد کنید تا وقتی کاربر اشاره‌گر ماوس را روی آن می‌برد، پنجره مرورگر بسته شود.







## فصل بیستم



کنترل

روند اجرای برنامه

## هدف‌های رفتاری



پس از مطالعه این فصل از فراگیر انتظار می‌رود:

۱. روش کار با عبارتهای شرطی را فرا بگیرد.
۲. با نحوه کار حلقه‌های تکرار آشنا شود.
۳. با استفاده از عبارتهای شرطی و حلقه‌های تکرار، روند اجرای برنامه را کنترل نماید.

## کلیات

در همه زبان‌های برنامه‌نویسی و اسکریپت‌نویسی، روش‌هایی برای کنترل روند برنامه پیش‌بینی شده تا کدنویس بتواند عملیات خاصی را به تعداد دفعات مشخصی انجام دهد یا ادامه یک عملیات را منوط به تحقق یک شرط نماید. به همین دلیل یادگیری روش استفاده از «عبارت‌های شرطی» و نیز «حلقه‌های تکرار»، نقش تعیین‌کننده‌ای در افزایش مهارت و توانایی برنامه‌نویس خواهد داشت. در این فصل این دو مفهوم را با ذکر مثال‌های متعدد و گویا مرور خواهیم کرد.

### ۱-۲۰ عبارت‌های شرطی

در هنگام نگارش کد با موقعیت‌هایی مواجه خواهید شد که باید بسته به درست یا نادرست بودن یک یا چند شرط، عملیات خاصی را انجام دهید. فرض کنید سن کاربر را از وی دریافت کرده‌اید و می‌خواهید تعیین کنید که این کاربر در کدام یک از رده‌های سنی نوجوان، جوان، میان‌سال و سالمند قرار می‌گیرد. در این موقعیت باید مقدار ورودی را با بازه‌های سنی مقایسه نموده و در صورت تطبیق با هر یک از شروط، مقدار خاصی را برگردانید.

برای مدیریت حالت‌های شرطی در برنامه از سه دستور زیر استفاده می‌شود:

if

if...else

switch

### ۱-۱-۲۰ دستور if

در شرایطی که لازم است یک شرط بررسی شود و در صورت درستی شرط، قطعه کدی اجرا گردد از دستور if استفاده می‌شود. نگارش کلی این دستور به صورت زیر است:

```
if (condition)
{
    execute code if condition is true
}
```



کدی بنویسید که مقدار یک متغیر عددی را خوانده و اگر کوچک‌تر از ۱۰ است پیغام مناسبی را روی صفحه نمایش دهد.

```
var n=9;
if (n<10)
{
    document.write( n + " از ۱۰ کوچکتر است ");
}
```



اگر متغیر x را با عدد ۲۰ مقارنه‌ی کنیم چه نتیجه‌ای چاپ می‌شود؟

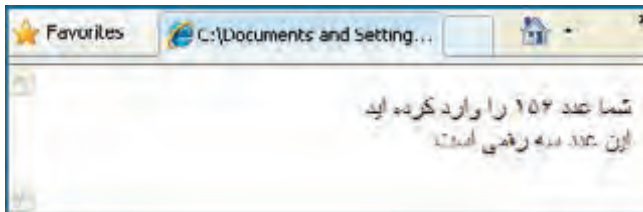


هیچ عبارتی روی صفحه نمایش داده نمی‌شود چون شرط نادرست است و عبارت درون علامت‌های { } به اجرا در نمی‌آید.



کدی بنویسید که یک عدد را از کاربر دریافت نموده و اگر عدد سه رقمی است، پیام مناسبی را روی صفحه بنویسد.

```
<script type="text/javascript">
var n=window.prompt("• عدد موردنظر را وارد کنید.");
if (n>=100 && n<=999)
{
document.write(" شما عدد " + n + " را وارد کرده اید "<br/>");
document.write("این عدد سه رقمی است.");
}
</script>
```



در این کد با استفاده از عملگر منطقی &&، درستی هر دو شرط بررسی شده است.



اگر در مثال قبلی علامت‌های { } را برداریم چه اتفاقی می‌افتد؟

- 1 if (n>=100 && n<=999)
- 2 document.write(" شما عدد " + n + " را وارد کرده اید "<br/>");
- 3 document.write("این عدد سه رقمی است.");

همان‌طور که قبلاً اشاره شد، اگر شرط دستور if صحیح باشد، تمامی دستورات موجود در بین علامت‌های { } به اجرا درخواهد آمد اما اگر سهواً یا عمداً این علامت‌ها را وارد نکنید، فقط اولین دستور بعد از if به عنوان

کدی که باید اجرا شود تلقی خواهد شد و سایر کدها جزو بدنه برنامه در نظر گرفته می‌شوند. بنابراین در صورت صحیح بودن شرط، خط ۲ اجرا می‌گردد و خط ۳ هم (که حالا جزو دستور if نیست) در هر صورت به اجرا درخواهد آمد. بنابراین حتی اگر عدد دو رقمی هم وارد کنید، عبارت «این عدد سه رقمی است» روی صفحه نوشته می‌شود و لذا منطق برنامه نادرست خواهد بود.



کدی بنویسید که یک عدد را از ورودی دریافت کند و زوج یا فرد بودن آن را مشخص نماید.

```
<body dir="rtl">
<script type="text/javascript">
var n=window.prompt("عدد موردنظر را وارد کنید","0");
if (n%2==0)
document.write("عدد وارد شده زوج است");
if (n%2!=0)
document.write("عدد وارد شده فرد است");
</script>
</body>
```

### بررسی کد:

- عدد وارد شده توسط کاربر درون متغیر n ذخیره می‌شود.
- دستور if اول، باقی‌مانده این عدد را محاسبه و با صفر مقایسه می‌کند، اگر برابر با صفر بود، اعلام می‌کند که عدد وارد شده زوج است.
- دستور if دوم هم در صورتی که باقی‌مانده عدد بر دو مخالف صفر باشد، فرد بودن آن را اعلام خواهد کرد.
- توجه داشته باشید که در هر یک از دستورات if فقط یک سطر کد وجود دارد بنابراین قرار دادن علامت { }، تأثیری در نتیجه نهایی نخواهد داشت.

### ۲-۱-۲ دستور if...else

دستور if...else شکل کامل‌تر دستور if است و با استفاده از آن می‌توان در صورت نادرست بودن شرط هم مجموعه‌ای از کدها را اجرا نمود. نگارش کلی این دستور به صورت زیر است:

```
if (condition)
{
```

```

execute code if condition is true
}
else
{
execute code if condition is false
}

```



کدی بنویسید که یک عدد را از ورودی دریافت کند و زوج یا فرد بودن آن را مشخص نماید

```

<script type="text/javascript">
var n=window.prompt("عدد موردنظر را وارد کنید");
if (n%2==0)
document.write("عدد وارد شده زوج است");
else
document.write("عدد وارد شده فرد است");
</script>

```



کدی بنویسید که نام کاربر را بخواند، در صورتی که نام وارد شده بین ۳ تا ۱۵ نویسه است، آن را بنویسد، در غیراین صورت پیغامی مبنی بر نامعتبر بودن نام وارد شده ظاهر نماید.

```

var s=window.prompt("نام خود را وارد کنید، طول نام باید بین ۳ تا ۱۵ نویسه باشد");
if (s.length>=3 && s.length<=15)
{
document.write("نام وارد شده " + s);
}
else
{
window.alert("نام وارد شده نامعتبر است");
}

```



صفحه‌ای حاوی یک دکمه ایجاد کنید تا کاربر با کلیک روی آن، پنجره‌ای حاوی دکمه‌های OK و Cancel مشاهده نماید. سپس در صورت کلیک روی هر یک از این دو دکمه، پیغامی مبنی بر کلیک شدن دکمه ظاهر گردد.

```
<html >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<script type="text/javascript">
function show_confirm()
{
var result=window.confirm("یکی از دکمه ها را فشار دهید");
if (result==true)
{
window.alert("OK کلیک کردید");
}
else
{
window.alert("Cancel کلیک کردید");
}
}
</script>
</head>
<body>
<input type="button" onclick="show_confirm()" value="نمایش پنجره تأیید" />
</body>
</html>
```

### بررسی کد:

■ در این کد تابعی به نام show\_confirm() ایجاد شده که پس از فراخوانی با استفاده از متد window.confirm() پنجره تأیید را به همراه یک پیغام نشان می‌دهد.

■ در این نوع پنجره، چنانچه کاربر روی دکمه OK کلیک کند، مقدار منطقی true و در صورت کلیک روی دکمه Cancel، مقدار false برگردانده می‌شود.

■ این مقدار در متغیر result ذخیره و با استفاده از دستور if...else ارزیابی می‌گردد تا پیغام مناسب



برای کاربر نمایش داده شود.

■ درون بدنه صفحه وب هم دکمه‌ای قرار داده شده و مشخصه onClick آن با نام تابع مقاردهی شده است تا با کلیک کاربر روی دکمه، فراخوانی تابع انجام شود.

دستور if...else را می‌توان به صورت زنجیره‌ای از شرطها نیز مورد استفاده قرار داد تا بتوان حالت‌های مختلف را مورد بررسی قرار داد و متناسب با هر حالت، عملیات موردنظر را انجام داد. نگارش حالت زنجیره‌ای به صورت زیر است:

```
if (condition_1)
{
    execute code if condition_1 is true
}
else if (condition_2)
{
    execute code if condition_2 is true
}
else
{
    execute code if condition_1 and condition_2 are false
}
```



کدی بنویسید که سن کاربر را دریافت نموده و بسته به حالت‌های زیر، پیغام مناسب را چاپ کند.

پیغام	سن
خوش آمدید	کمتر از ۱۰ سال
از بخش بازی‌های وبسایت دیدن نمایید	بین ۱۰ و ۱۸
از فروشگاه آن‌لاین ما بازدید فرمایید	بین ۱۸ تا ۳۰
خوش آمدید	بیشتر از ۳۰

<script type="text/javascript">

```

var n=window.prompt("کاربر عزیز، سن خود را وار نمایید");
var msg="";
if (n>=10 && n<=18)
{
    msg="از بخش بازی های وب سایت دیدن نمایید";
}
else if (n>18 && n<30)
{
    msg="از فروشگاه آنلاین ما بازدید فرمایید";
}
else
{
    msg="خوش آمدید";
}
window.alert(msg);
</script>

```

در این کد برای جلوگیری از تکرار بی مورد متد alert، پیغام مناسب درون متغیر msg ذخیره شده و پس از بررسی شرطها و مقداردهی با پیغام مناسب، متغیر برای متد ارسال شده است.

### ۳-۱-۲۰ دستور switch

وقتی تعداد شرطهایی که می‌خواهیم متناظر با آن‌ها عملیات خاصی را انجام دهیم زیاد می‌شود بهتر است به جای استفاده از if های متعدد، از دستور switch استفاده نماییم. نگارش کلی دستور switch به صورت زیر است.

```

switch (expression)
{
case value_1:
execute code if expression is equal to value_1
break;
case value_2:
execute code if expression is equal to value_2
break;
case value_3:
execute code if expression is equal to value_3
break;

```

default:

execute code if expression is not equal to value\_1, value\_2 and value\_3

}

برای استفاده از این دستور، ابتدا مقداری که باید مورد مقایسه قرار گیرد به جای expression قرار داده می‌شود. سپس برای هر مقایسه، کلمه کلیدی case و سپس مقدار موردنظر درج می‌شود. پس از علامت : نیز مجموعه دستوراتی که باید در صورت معادل بودن expression و مقدار اجرا گردند قرار می‌گیرند و دستور break هم اعلام می‌کند که در صورت صحیح بودن شرط، نیازی به بررسی سایر موارد نیست. نهایتاً چنانچه نتیجه هیچ‌یک از مقایسه‌ها صحیح نباشد، کدی که پس از default قرار گرفته به اجرا در خواهد آمد.



اسکرپتی بنویسید که یک پیش‌شماره را از کاربر دریافت نموده و مشخص نماید این پیش‌شماره مربوط به کدام یک از شهرهای کشور است.

```
<body dir="rtl">
<script type="text/javascript">
var code=window.prompt("کد شهر موردنظر را وارد کنید");
var city;
switch (code)
{
case "021":
    city="تهران";
    break;
case "0311":
    city="اصفهان";
    break;
case "0411":
    city="تبریز";
    break;
case "0511":
    city="مشهد";
    break;
default:
    city="سایر شهرها";
```

```

}
document.write(" + city" کد وارد شده متعلق است به");
</script>
</body>

```

## ۲-۲۰ حلقه‌های تکرار

حلقه‌های تکرار در زبان‌های برنامه‌نویسی اهمیت فوق‌العاده‌ای دارند چون اغلب اوقات لازم است عملیاتی در برنامه به تعداد دفعات مشخصی تکرار شود یا تکرار آن‌قدر ادامه یابد تا شرط خاصی محقق گردد. در زبان جاوا اسکریپت برای ایجاد حلقه‌های تکرار از دستورات `while`، `do...while` و `for` استفاده می‌شود که هر یک کاربرد خاصی دارند.

### ۲-۲۰-۱ دستور `while`

شکل کلی نگارش این دستور به صورت زیر است:

```

while (condition)
{
    execute code while condition is true
}

```

دستور `while` یک شرط را دریافت می‌کند و تا زمانی که این شرط درست است، کدهای بین علامت { } را اجرا خواهد کرد. در صورت نادرست شدن شرط، کنترل برنامه به خارج از حلقه منتقل می‌شود.



کدی بنویسید که عبارت «خوش آمدید» را پنج بار روی صفحه بنویسد.

```

<body>
<script type="text/javascript">
var i=0;
while (i<5)
{
    document.write("خوش آمدید<br/>");
    ++i;
}
</script>
</body>

```

## بررسی کد:

■ برای ایجاد حلقه‌های تکرار، ابتدا باید یک شمارنده (مانند  $i$ ) تعریف نمایید. سپس شرط اتمام حلقه `while` را عبور مقدار  $i$  از سقف مجاز قرار دهید. درون بدنه حلقه هم دستورات موردنظر را نوشته و ضمناً فراموش نکنید که مقدار شمارنده را تغییر دهید.

■ در این کد، ابتدا شمارنده  $i$  با عدد صفر مقداردهی می‌شود.

■ از آن‌جا که شرط حلقه یعنی  $(i < 5)$  صحیح است، یک‌بار عبارت «خوش آمدید» روی صفحه نوشته خواهد شد و ضمناً مقدار  $i$  یک‌واحد افزایش می‌یابد.

■ مجدداً اجرای برنامه به ابتدای حلقه برمی‌گردد. هنوز هم شرط صحیح است و بنابراین دستورات اجرا می‌شود. این کار تا جایی ادامه می‌یابد که مقدار  $i$  به عدد ۵ می‌رسد و این‌بار شرط  $5 < 5$  نادرست است؛ لذا اجرای حلقه به پایان می‌رسد.



کدی بنویسید که ۵ کادر متنی (TextBox) روی صفحه قرار داده و شناسه آن‌ها را با مقادیر `txt1`، `txt2` و ... نام‌گذاری نماید.

```
<script type="text/javascript">
var i=1;
while (i<6)
{
    document.write("<input type='text' id=txt" + i + "'><br/>");
    ++i;
}
</script>
```

## بررسی کد:

■ در این کد شمارنده  $i$  با عدد یک مقداردهی شده است و با هر بار اجرای حلقه، عبارت سازنده کادر متنی روی صفحه تولید می‌شود و شناسه آن با عبارت `txt` بعلاوه مقدار شمارنده مقداردهی می‌شود.

■ توجه داشته باشید که برای ایجاد یک کادر متنی باید کد

```
<input type="text" id="..." />
```

وارد صفحه وب شود. اما اگر در ورودی متد `write` علامت " را قرار دهیم به معنی ابتدا یا انتهای رشته ورودی خواهد بود. برای جلوگیری از این مشکل و نادیده گرفته شدن این علامت به عنوان ابتدا و انتهای رشته، باید قبل از آن‌ها نویسه \ را قرار دهیم. به این ترتیب علامت‌های " به عنوان بخشی از رشته تلقی خواهند شد.



کد مثال قبل را به گونه‌ای تغییر دهید که این کار با کلیک شدن یک دکمه و از طریق تابع انجام شود.

```
<html >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<script type="text/javascript">
function CreateTextBox(n)
{
var i=1;
while (i<=n)
{
document.write("<input type='text' id=txt" + i + "'/><br/>");
++i;
}
}
</script>
</head>
<body>
<input type="button" value="کلیک کنید" onClick="CreateTextBox(5)" />
</body>
</html>
```

**بررسی کد:**

- در این مثال تابعی با نام CreateTextBox ایجاد کرده‌ایم که یک عدد دریافت می‌کند و به تعداد آن، کادر متنی روی صفحه قرار می‌دهد.
- سپس رویداد onClick دکمه‌ای که روی صفحه ایجاد کرده‌ایم با این تابع مقداردهی شده و ضمناً عدد ۵ برای تابع ارسال می‌شود.



اجرای کد زیر چه نتیجه‌ای دربر دارد؟

```
<script type="text/javascript">
```

```

var i=1;
while (2<3)
{
    document.write( i +"<br/>");
    ++i;
}
</script>

```

از آن جا که شرط while همواره صحیح است، اجرای دستورات حلقه هیچ‌گاه متوقف نمی‌شود و اصطلاحاً یک «حلقه بی‌پایان» ایجاد می‌شود. در این حالت مرورگر پیغامی مبنی بر تقاضای توقف اجرای اسکریپت را مطرح خواهد کرد که با کلیک روی دکمه Yes می‌توانید به این حلقه بی‌انتهای پایان دهید. کلیک روی دکمه No باعث می‌شود اعداد صحیح بزرگ‌تر از یک روی صفحه وب نوشته شوند.



## ۲-۲-۲۰ دستور do...while

کارکرد این دستور شباهت زیادی به دستور while دارد، با این تفاوت که بررسی شرط در انتهای حلقه انجام می‌گیرد و در نتیجه، کد درون حلقه حداقل یک‌بار به اجرا در می‌آید. نگارش کلی این دستور به صورت زیر است:

```

do
{
    execute code
}
while (condition)

```

مثال ۱: نتیجه اجرای کد زیر چیست؟

```

<script type="text/javascript">
var i=0;
while (i<0)

```

```

{
    document.write( i +"<br/>");
    ++i;
}
do
{
    document.write( i +"<br/>");
    ++i;
}
while (i<0)
</script>

```

حلقه while اجرا نمی‌شود چون شرط  $i < 0$  نادرست است و لذا مقدار  $i$  هم تغییر نمی‌کند. اما حلقه do... while یک بار اجرا می‌شود و عدد صفر روی صفحه نوشته می‌شود اما با افزایش یک واحدی  $i$  دیگر شرط  $i < 0$  صحیح نیست و بنابراین، اجرای اسکریپت به انتها می‌رسد.



کدی بنویسید که یک عدد غیرصفر ( $n$ ) و یک رشته ( $s$ ) را از کاربر بگیرد و  $n$  بار روی صفحه بنویسد.

```

<body>
<script type="text/javascript">
var n= window.prompt("یک عدد بزرگ تر از صفر وارد کنید");
var s= window.prompt("یک عبارت متنی وارد نمایید");
if (n>0)
{
    do
    {
        document.write(s + "<br/>");
        --n;
    }
    while (n>0)
}
else
{

```