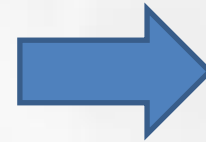
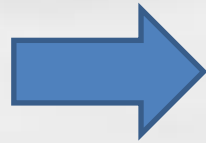


JAVA SCRIPT

JAVA SCRIPT

**הוצג לראשונה בשנת 1995 כדרך להוסיף תוכניות לדפי אינטרנט
בדפדפן Netscape Navigator.
מאז כל הדפדפנים המודרניים אימצו את השפה להרצת קטעי קוד
ללא "הידור" לצד הלקוח.
כיום שפה JS הינה השפה הפופולרית ביותר בעולם הפרונט-אנד.**



Create SCRIPT

ישנם 2 דרכים ליצור סקריפט:

1. ניתן ליצור אלמנט script בכל חלק בעמוד שלנו (BODY OR HEAD).

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

2. ניתן ליצור קובץ נפרד עם סיומת js. ולהוסיף אותו לעמוד שלנו.

```
<script type="text/javascript" src="Filename.js"> </script>
```

****מיקום הסקריפט מאוד קריטי. רצוי למקם אותו בסוף העמוד לאחר טעינת כל האלמנטים בעמוד.**

OUTPUT

הצגת ערכים: ישנן דרכים רבות להצגת ערכים ב JS. אנחנו נתייחס ל 3 עיקריות:

1. הצגת תוכן ב Console.Log

2. הקפצת הודעה חיצונית למשתמש.

3. הכנסת ערכים לתוך אלמנט ספציפי.

`console.log('...');`

הדפסת ערכים ל console:

`console.log('hello world');`

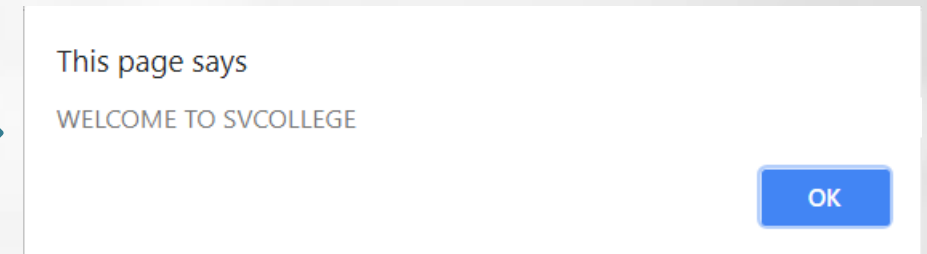
ניתן לראות את התוצאה ב devTools
בדפדפן. F12

window.alert

הקפצת הודעה חיצונית למשתמש:

`window.alert('תוכן');`

```
<body>  
.....  
<script type="text/javascript">  
window.alert("WELCOME TO SVCOLLEGE");  
</script>  
</body>
```



****לא ניתן לשנות את עיצוב ההודעה. תלוי בדפדפן בלבד!**

innerHTML

הכנסת ערכים לתוך אלמנט ספציפי:

```
document.getElementById('idName').innerHTML=value;
```



חיפוש הערך לפי ID



ID ספציפי



איזה ערך
יוחלף בתוך האלמנט

****ניתן לחפש לפי ID, CLASS, NAME.**
****הערך יחליף את הערך שהיה בתוך האלמנט.**

Data Types

יחידות זיכרון: שימוש בשפת תוכנה מתבצעת בעזרת הקצאת זיכרון לטובת חישובים ושימושים של התוכנה.

לדוגמא:

נניח וניצור "שדה" של סיסמא ונרצה לוודא את תקינותו, נשמור את השדה בתוך יחידת זיכרון וכך נוכל לבצע על הסיסמא כל פעולה שנרצה.

דוגמא נוספת:

נניח ויצרנו משחק ובמשחק אנחנו יכולים לצבור נקודות, את הנקודות נשמור בתוך יחידת הזיכרון וכך נוכל במהלך המשחק להוסיף עוד נקודות למאזן ולהציג את מספר הנקודות בסוף המשחק.

Data Types

יצירת יחידת זיכרון:

let **name**;

let – מתאר את הטיפוס של יחידת הזיכרון.
name – שם המשתנה שהגדרנו.

const **info**;

const – הגדרת טיפוס **קבוע**, משתנה שלא ניתן לשנות לאחר היצירה.
Info – שם המשתנה שהגדרנו.

****שם המשתנה יכול להיות כל שם, ללא מספרים בתחילתו וללא רווחים.**

****שם המשתנה צריך להיות שם משמעותי כדי שנוכל להבין מה הוא משמו.**

****כתיבה של משתנים תעשה ב camelCase.**

Operator Assignment

=

אופרטור השמה: בניגוד למתמטיקה ששם משתמשים בסימן ה"שווה" לבדיקת שוויון, אופרטור זה משמש להכנסת נתונים לתוך המשתנה שלנו. כל מה שמימין לשווה יישמר במשתנה משמאל.

Operator Assignment

=

name = value



שם המשתנה



ערך

Operator Assignment

=

דוגמאות להכנסת ערכים:

```
let name = 'svcollege';  
let number = 35;  
name = 'sv college';  
let newName = 'SBCOLLEGE';  
name = newName;
```

Arithmetic Operators

אופרטורים מתמטיים: ניתן לבצע פעולות מתמטיות לפני הכנסתם למשתנה.

לדוגמא:

```
let sum = 5+5;
```

1. המערכת תחבר $5+5$.
2. התוצאה 10 תישמר לתוך המשתנה `sum`.

Arithmetic

OPERATOR	Description
()	סוגריים
* / %	כפל חילוק שארית
+ -	חיבור חיסור

```
let num = (5+5*12)/2;
```

מהי התוצאה לדעתכם?

** גם בתכנות יש דגש לסדר פעולות חשבון

Shortcuts

```
let num = 1;  
num = num + 1;
```

בקטע קוד זה אנו רואים הוספה של 1 למשתנה:

```
let num = 1;
```

ניתן לקצר זאת בדרך הבאה:

```
num +=1;
```

ואפילו כך:

Shortcuts



svcollege
ללמוד. לדעת. לעבוד.

קיצור דרך	ללא קיצור
num+=value	num = num + value
num-=value	num = num - value
num*=value	num = num * value
num/=value	num = num / value
num%=value	num = num % value
num++	num = num + 1
num--	num = num - 1

Math

הינה ספריה אשר מכילה פונקציות מתמטיות רבות:

PI() - מחזיר מספר פאי

```
Math.PI(); // 3.1459...
```

round() – מעגלת את המספר

```
Math.round(92.5); // 93
```

```
Math.round(92.7); // 93
```

pow(number , exponent) - מבצעת חזקה של המספר

```
Math.pow(5,2); // 25
```

sqrt(number) - מבצעת שורש על מספר מסוים

```
Math.sqrt(81); // 9
```

String

מחרוזת הינה רצף של תווים המורכב מכל תו במקלדת – אותיות מספרים וסימנים מיוחדים.
כל ערך שנכתב ב ' ' נחשב למחרוזת.
** כולל רווח!

```
let firstName = 'orgad';  
let lastName = 'mahluf';  
let name = firstName + ' ' + lastName;
```

name

orgad mahluf

String Actions

פעולות: ישנם פונקציות (פעולות שנלמד בהמשך) אשר מובנות עם השפה וניתן להיעזר בהם.

length – מחזיר את אורך המילה.

```
let str = 'svcollege' ;  
let len = str.length;
```

0 1 2 3 4 5 6 7 8 = length 9
s v c o l l e g e

String Actions

indexOf(...) - מחזיר את **מיקום** האות/מחרוזת

```
let str = 'hello world';  
let index = str.indexOf('world');
```

מיקום התחלה 6 \rightarrow n

****** במידה והערך לא נמצא יוחזר -1

String Actions

substring(start, end) - פעולה אשר מחזירה חלק מהמחרוזת.

```
let str = 'hello world' ;
```

```
let subStr = str.substring(6 , 11);
```

subStr → world

String Actions

slice(**start** , **end**) - פעולה אשר מחזירה חלק מהמחרוזת.

```
let str = 'hello world' ;
```

```
let strSlice = str.slice(6 , 11);
```

```
strSlice → world
```

String Actions

replace(**source** , **new**) - פעולה אשר מחליפה ערך בערך.

```
let str = 'hello world';
```

```
let newStr = str.replace('world', 'orgad');
```

newStr → hello orgad

String Actions

charAt(index) - פעולה אשר מחזירה אות במיקום ספציפי.

```
let str = 'hello world';  
let letter = str.charAt(0);
```

letter → h

Condition IF

במידה ונרצה שתנאי יקרה רק כאשר ... נשתמש בתנאי
"אם".

הקוד שבבলוק יתבצע רק כאשר התנאי "אמת".

```
if(true) {
```

```
.....
```

```
}
```

Comparison Operators

פירוש	סימן
שווה ל..	==
שונה מ..	!=
גדול מ..	>
קטן מ..	<
גדול או שווה	>=
קטן או שווה	=<
NOT	!

פירוש	סימן
או	
וגם	&&

Comparison Operators

EXAMPLE



svcollege
ללמוד. לדעת. לעבוד.

```
let age = 18;  
  
if(age<=18) {  
    alert('access denied');  
}  
if(age>18){  
    alert('access complete');  
}
```

מה התוכנית עושה?

else

**אחרת – במידה ותנאי "אם" לא מתקיים, ניתן לבצע תנאי אחר.
לתנאי else אין תנאי!**

```
if(true) {
```

```
.....
```

```
}
```

```
else{
```

```
}
```

if / else

EXAMPLE

```
let age = 18;
```

```
if(age<=18) {  
    alert('access denied');  
}  
else {  
    alert('access complete');  
}
```

else if

אחרת אם – במידה ונצטרך לעבור על יותר מ 2 תנאים
נוכל להוסיף "אחרת אם" לסקריפט שלנו.

```
let bestCollege = 'svcollege';

if(bestCollege.length == 2) {
    alert('To short');
}
else if (bestCollege.length<2) {
    alert('?!?');
}
else{
    if(bestCollege == 'svcollege'){
        alert( ' 😊 ');
    }
    else {
        alert(' 😞 ');
    }
}
```



Debugger

בעזרת devTool, ניתן לרוץ על הסקריפט "צעד אחר צעד".
בשביל לבצע זאת יש להגדיר breakpoint בקוד.

```
<script>  
.... code ....  
debugger  
.... code ....  
</script>
```



מרגע שהקטע קוד מגיע ל breakpoint
ניתן לרוץ על הקוד step by step
F10 - המשך צעד אחד קדימה
F11 - הכנס לתוך הפונקציה

function

פונקציה הינה קטע קוד שניתן לשלוח ולקבל ממנה ערכים.

יצירת פונקציה:

```
function name(){  
  
}
```

function

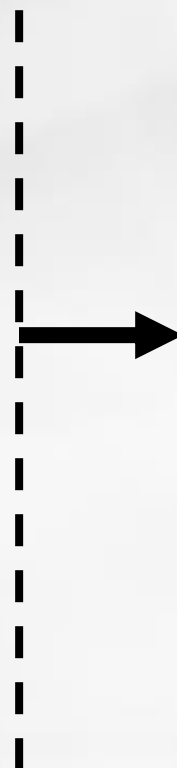


svcollege
ללמוד. לדעת. לעבוד.

דוגמא:

```
function alertHelloWorld(){  
    alert( 'hello world' );  
}
```

```
alertHelloWorld();  
alertHelloWorld();  
alertHelloWorld();
```



בדוגמא זו יקפצו 3
הודעות בזו אחר זו עם
המשפט hello world

function

דוגמא לפונקציה שמקבלת ערכים:

```
function alertSomething (printMe){  
    alert(printMe);  
}
```

```
alertSomething('hello');  
alertSomething('class');
```

return value

פונקציה יכולה להחזיר ערכים מכל סוג:

```
function func ( ){  
    var result = 'hello world' ;  
    return result;  
}
```

```
var res = func();
```

****שימו לב שהערך המוחזר מהפונקציה חייב להישמר במשתנה או להיות מודפס.**

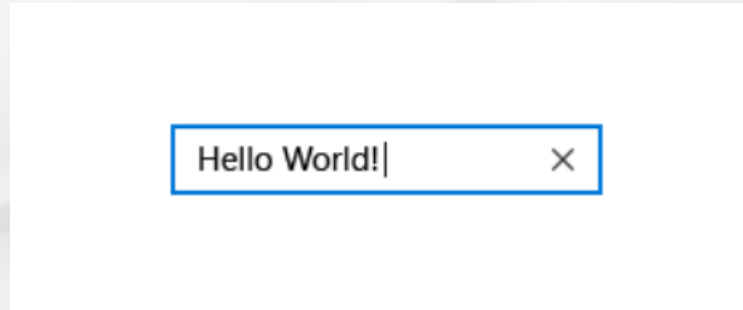
function

הפעלת פונקציה בעזרת כפתור:

```
<button onClick="funcName()"> CLICK PLZ </button>
```

הגדרה זו מפעילה את הפונקציה בעת לחיצה על הכפתור

Value from input (type text)



```
let inputV = document.getElementById(' ').value;
```

פקודה זו תאפשר את שמירת הנתונים מהערך שהתקבל ב INPUT למשתנה.

****שימו לב להפעיל אופציה זו בתוך פונקציה בלבד.**

While loop

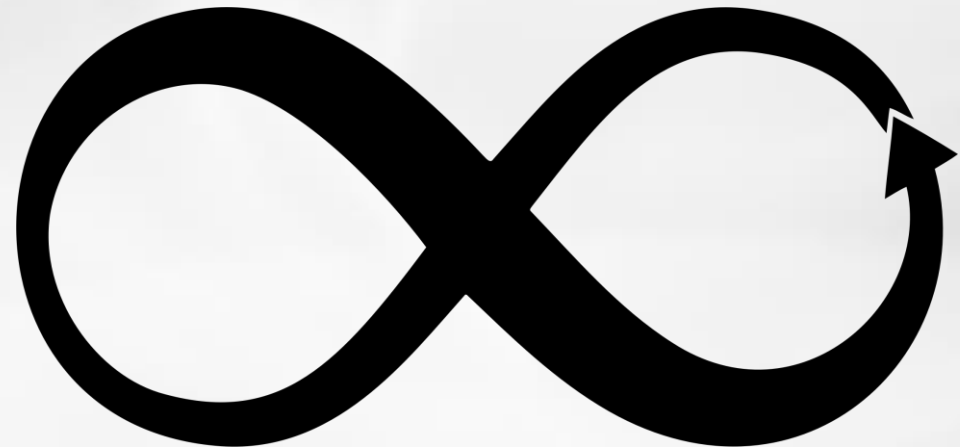
לולאה – קטע קוד אשר ממשיך להתבצע כל עוד התנאי נכון.

```
while(true){
```

```
.....
```

```
.....
```

```
}
```



****יש לשים לב שבמידה ולא נחשוב על תנאי עצירה הלולאה תמשיך לעד.**

While example

```
let i = 0;  
while(i<4){  
  alert('hello');  
  i++;  
}
```

i	While(i<4)
0	True
1	True
2	True
3	True
4	False

for

לולאת for – נשתמש בה כאשר אנו יודעים מראש את מספר
הסיבובים שהלולאה אמורה לרוץ.

for(קידום ; תנאי ; אתחול)

for example

**דוגמא ללולאה שרצה על כל האותיות במחרוזת, במידה
והיא מוצאת ערך מספרי היא מקפיצה הודעה על כך.**

```
let fullName = 'Shem1Bar';  
let i;  
for(i=0 ; i<fullName.length ; i++){  
    if(fullName.charAt(i)>='0' && fullName.charAt(i)<='9'){  
        alert('number exists');  
    }  
}
```

for example

```
function checkPrime(num){  
  let i;  
  if(num == 1 || num == 2)  
    return true;  
  
  for(i=2 ; i<num ; i++){  
    if(num%i==0){  
      return false;  
    }  
  }  
  return true;  
}
```

דוגמא לפונקציה שמקבלת מספר
ומחזירה אם הוא ראשוני או לא.



svcollege
ללמוד. לדעת. לעבוד.

break & continue

break – פקודה אשר "שוברת" את הלולאה.

דוגמא לשימוש ב break:

```
for(i = 11 ; i != 0 ; i--){  
    if(i%3 == 0){  
        break;  
    }  
}
```

מתי הלולאה תעצור?

break & continue

continue – פקודה אשר ממשיכה לסיבוב הבא של הלולאה.

דוגמא לשימוש ב continue:

```
while(num < 1500){  
    if(num < 0){  
        num*=-1;  
        continue;  
    }  
    num++;  
}
```

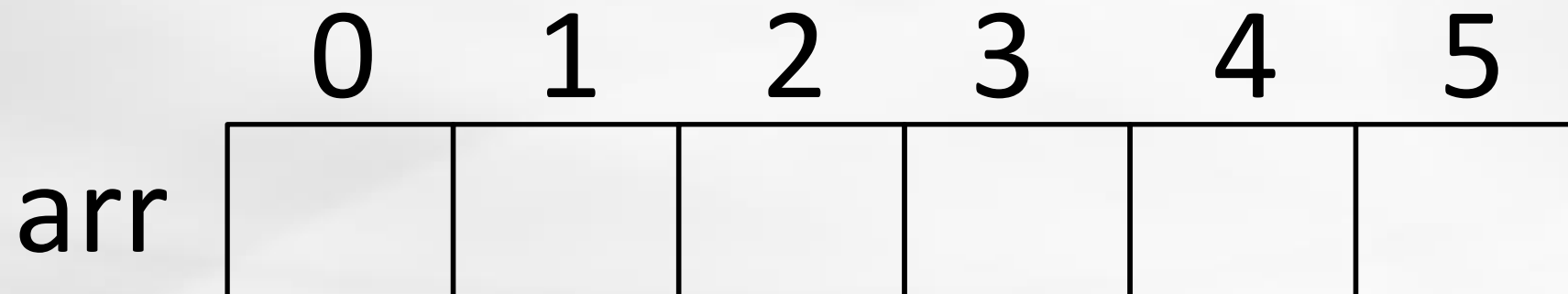
מה הלולאה עושה?

Array

מערך הוא סוג של מבנה נתונים.

מערך הינו רצף של תאים צמודים בזיכרון. לכל התאים יש את אותו שם ומיקום יחסי שונה.

דוגמא למערך בעל 5 תאים:



Array

יצירת מערך:

```
let arr = [value1, value2, value3, value4, value5];
```

גישה לתא ספציפי במערך:

```
arr[index] = newValue;
```

דוגמא להכנסת ערך מספרי לתא השלישי במערך:

```
arr[2] = 26;
```

Array – methods

toString() – מדפיסה את הערכים במערך עם " " בין כל ערך.

```
let arr = [value1, value2, value3, value4, value5];  
alert(arr.toString());
```

push() & pop() – הוצאת והכנסת ערכים לסוף המערך.

```
let x = arr.pop(); // מוציא את הערך האחרון מהמערך  
arr.push("newValue"); // מוסיף את הערך לסוף המערך
```

```
let arr = [1, 4, 77, 22, 200];  
let len = arr.push(8); // כעת במשתנה נשמר האורך 6
```

shift() & unshift() – הוצאת והכנסת ערכים לתחילת המערך.

Array – methods

`sort()` – פונקציה אשר ממיינת את המערך מהקטן לגדול:

```
let arr = ['b', 's', 'a', 'e', 'b'];  
arr.sort(); // a, b, b, e, s
```

שימוש במיון למספרים:

```
arr.sort(function(a, b){return a-b}); // מיון מערך בסדר עולה  
arr.sort(function(a, b){return b-a}); // מיון מערך בסדר יורד
```

Array- matrix

מערך של מערכים – כל תא במערך מחזיק מערך.

`mat[row][col]` // פניה לאיבר במערך.

```
let mat = [  
mat[0] ← [1, 5, 9],  
mat[1] ← [2, 4, 7],  
mat[2] ← [6, 8, 3]  
];
```

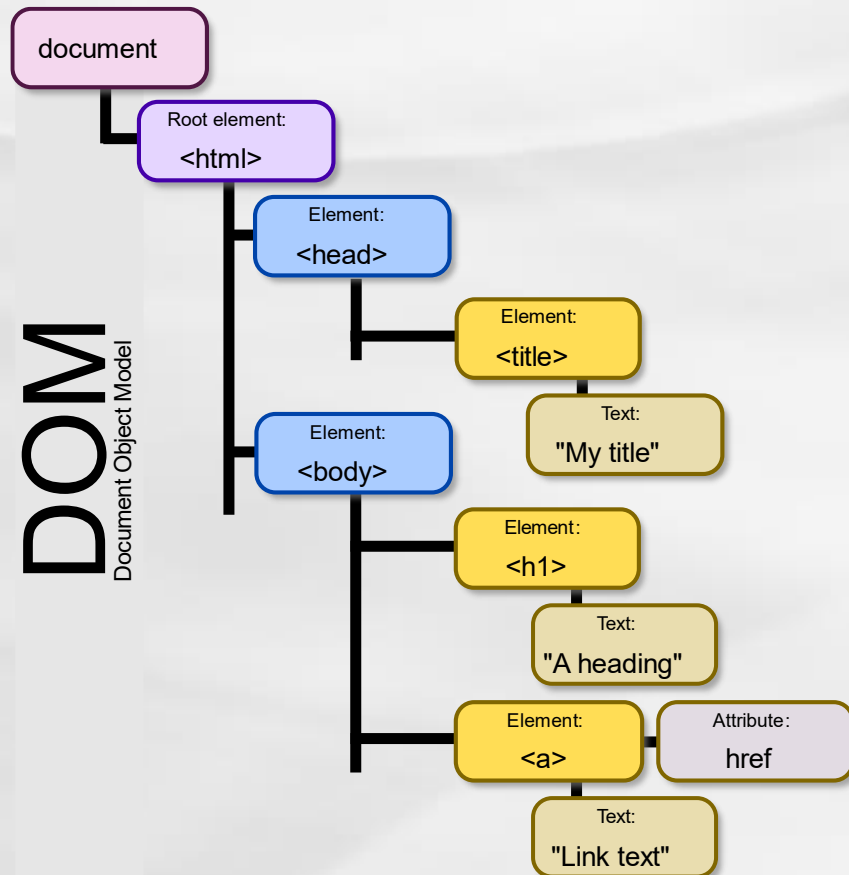
`mat[1][1]` // 4

`mat[2][0]` // 6



DOM

Document object model



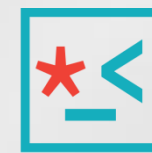
באמצעות DOM לJS יש את היכולת לגשת ולשנות את כל האלמנטים של מסמך ה HTML.

הדפדפן יוצר דיאגרמה של עץ המציג אובייקטים יחד עם הקשרים והמידע של כל אלמנט בדף HTML.

ובכך מקבל JavaScript את הכוח ליצירת דף HTML דינמי.

Document Methods

Find Element



svcollege
ללמוד. לדעת. לעבוד.

הסבר	פעולה
מוצא אלמנט לפני id	<code>document.getElementById(id)</code>
מוצא אלמנט לפני תגית	<code>document.getElementsByTagName(name)</code>
מוצא אלמנט לפני class	<code>document.getElementsByClassName (name)</code>

Document Property

Edit Element

הסבר	פעולה
משנה את התוכן של האלמנט	<code>element.innerHTML = new html content</code>
משנה את ערך התכונה של האלמנט	<code>element.attribute = new value</code>
	<code>element.setAttribute(attribute, value)</code>
משנה את העיצוב של האלמנט	<code>element.style.property = new style</code>

Document Methods

Adding and Deleting Elements

הסבר	פעולה
צור אלמנט חדש	<code>document.createElement(element)</code>
הסר אלמנט	<code>document.removeChild(element)</code>
צרף אלמנט	<code>document.appendChild(element)</code>
החלף אלמנט ישן באלמנט חדש	<code>document.replaceChild(new, old)</code>

exception

try / catch / throw

טיפול בשגיאות - ככל שנדע "לטפל" בשגיאות מראש, כך
נוכל להימנע מקריסות עתידיות.

```
try{  
    .... Code ....  
}catch(e){  
    alert(e.name);  
}
```

exception

try / catch / throw

דוגמא לאופציה בשימוש במשתנה שלא קיים / חלוקה ב 0:

```
var num1 = document.getElementById("..1").value;  
var num2 = document.getElementById("..2").value;  
try{  
    alert(num1/num2);  
}catch(e){  
    alert(e.name);  
}
```


Random number

יצירת מספר אקראי -

$\text{Math.floor}(\text{Math.random()} * (\text{max} - \text{min})) + \text{min};$

דוגמא למספר רנדומלי בין 2 – 10 לא כולל!

$\text{Var res} = \text{Math.floor}(\text{Math.random()} * (10 - 2)) + 2;$

דוגמא למספר רנדומלי בין 2 – 10 כולל!

$\text{Math.floor}(\text{Math.random()} * (\text{max} - \text{min} + 1)) + \text{min};$

Object

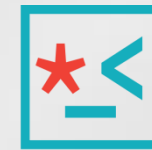
גישה לכל מאפיין מתבצעת באופן הבא:

nameOfObject.**property**

(בהמשך לדוגמא הקודמת):

```
student.lastName = 'levi';  
document.getElementById('..').innerHTML = student.lastName;
```

Object



svcollege
ללמוד. לדעת. לעבוד.

אובייקטים – סוג של מבנה נתונים אשר מחזיק כמה מאפיינים ובכך
מאפשר שמירה של נתונים תחת ארגומנט אחד.

```
let student = {  
  firstName: 'dor',  
  lastName: 'dekel',  
  ID: '123456789',  
  GPA: 82.7  
};
```

Object – function

ניתן להוסיף פונקציות בתוך אובייקט.

```
let objectName = {  
  variable1 : value,  
  variable2 : value,  
  sum : function(){  
    return this.variable1+ this.variable2;  
  }  
};
```

this = מתייחס לאלמנט הנוכחי.

```
document.getElementById('.').value = objectName.sum();
```

Classes

מחלקות נוספו לשפה כאשר יצא העדכון של ES6, (נלמד בהרחבה בשיעור הבא)
מחלקות בJS הן תבניות לאובייקטים של JS, כלומר כאשר יש מחלקה ניתן ליצור
אובייקט בעזרתו, להשתמש במאפייניו ופעולותיו.

בכדי ליצור אובייקט עם כל המאפיינים, נשתמש בבנאי constructor שיכריח
להזין את הערכים לכל המאפיינים.
הבנאי נקרא אוטומטית כאשר נוצר אובייקט חדש.

Class Constructor & Properties



```
class Product {  
  constructor(name, price) {  
    this.name = name;  
    this.price = price;  
  }  
}
```

בדוגמה זו יצרנו מחלקת מוצר ובנאי
ש מקבל 2 ערכים: שם מוצר ומחיר
המוצר.

Class Methods

```
class Product {  
    constructor...  
  
    discount() {  
        return this.price * 0.9;  
    }  
}
```

נוסיף למחלקה גם מתודה לחישוב הנחה, כך שכל מוצר יקבל את אותה המתודה ויציג את המחיר לאחר ההנחה, בהתאם למחיר של המוצר. הרי המחירים של המוצרים שונים וכך גם ההנחה.

Object Instance

```
let p1 = new Product("Bamba", 5);  
let p2 = new Product("Bisli", 4);
```

בעת יצירת מוצר חדש העברנו
לבנאי את הערכים המתאימים.
כעת יש לנו 2 מוצרים: במבה וביסלי
עם המחירים 5 ו4 בהתאמה.

```
console.log(`Product name: ${p1.name}, the price after discount is ${p1.discount()}`)  
console.log(`Product name: ${p2.name}, the price after discount is ${p2.discount()}`)
```

Console

```
Product name: Bamba, the price after discount is 4.5  
Product name: Bisli, the price after discount is 3.6
```


Inheritance

בעת הורשה אנו "מורישים" (מעבירים) את כל המאפיינים והמתודות למחלקה היורשת בעת יצירת אותה מחלקה חדשה, הורשה שימושית עבור שחזור קוד.

לשם יצירת הורשה נכיר את המילה השמורה **extends** אשר בעזרתה נגדיר את המחלקה היורשת ממחלקת הבסיס (המורשה).

Inherits Constructor

```
class Category {  
    constructor(categoryName) {  
        this.categoryName = categoryName;  
    }  
    getCategoryName() {  
        return this.categoryName;  
    }  
}
```

```
class Product extends Category {  
    constructor(name, price, categoryName) {  
        super(categoryName)  
        this.name = name;  
        this.price = price;  
    }  
    discount() {  
        return this.price * 0.9;  
    }  
}
```

פקודת `super()` מעבירה את הערך אל הבנאי של מחלקת הבסיס
כעת כאשר נשתמש בבנאי `Product` בעת יצירת אובייקט נצטרך להעביר 3 ערכים.

Use Inherits Methods

לאחר שירשנו את המחלקה Category אתה ירשנו גם את המתודות והמאפיינים.
מסיבה זו נוכל להשתמש במתודה הקיימת במחלקת הבסיס לאחר שניצור
אובייקט של Product. הנה דוגמה:

```
let p1 = new Product("Bamba", 5, "Snacks")
```

```
let p2 = new Product("Yogurt", 4, "Dairy Products")
```

```
console.log(`${p1.name}, belongs to the category ${p1.getCategoryName()}`)
```

```
console.log(`${p2.name}, belongs to the category ${p2.getCategoryName()}`)
```

```
Bamba, belongs to the category Snacks
```

```
Yogurt, belongs to the category Dairy Products
```

ES5 & ES6

JS פותחה בשנת 1995 והפכה ל ECMAScript ב1997, הוגדר סטנדרט אשר הגדיר את הכללים לכתיבה נכונה ואחידה של השפה.

במהלך השנים, יצאו המון עדכונים וגרסאות שכללו תוספות ופיצ'רים חדשים.

ES6 ES5 – אלו שני הגרסאות הגדולות ביותר שיצאו.

ES6 - ECMAScript 2015

- let
- const
- arrow Function
- classes
- spread

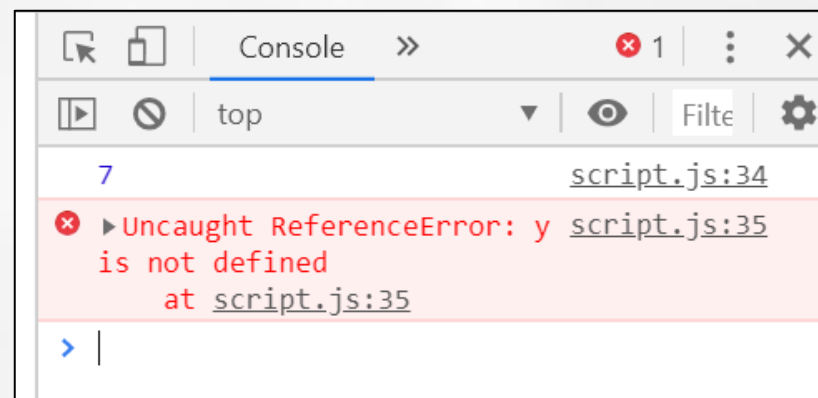
ES5 - ECMAScript 2009

- forEach()
- filter()
- map()
- array.indexOf()

הטיפוסים `let` / `var`

`let` הינו משתנה לוקאלי (מוגדר ו"מת" בתוך ה- scope).
`var` הינו משתנה גלובלי (נוכל "לקרוא" לו מכל מקום).

```
if(true){  
    var x = 7;  
    let y = 5;  
}  
console.log(x);  
console.log(y);
```



arrow function

דרך נוספת ליצור פונקציה



svcollege
ללמוד. לדעת. לעבוד.

function

```
function func(name){  
    Console.log('hello' + name);  
}  
func();
```

arrow function

```
let func = (name) => {  
    console.log(`hello ${name}`);  
};  
func();
```

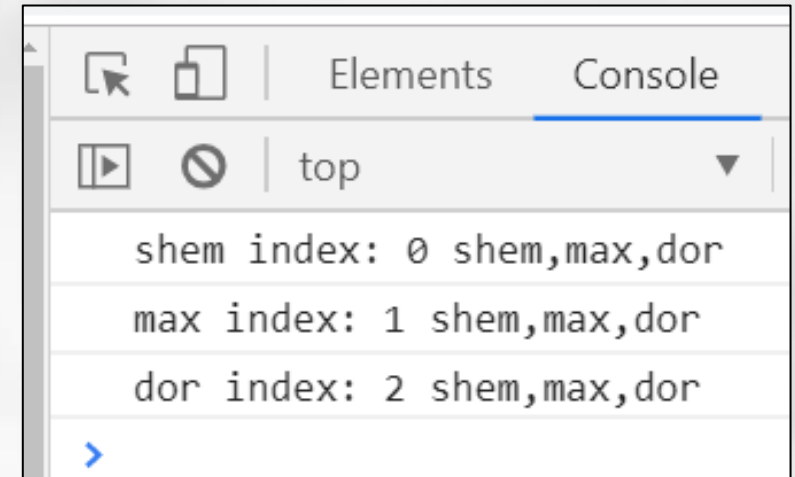
******הפונקציה נשמרת בתוך משתנה, לאחר מכן ניתן לקרוא/לזמן את המשתנה
כמו פונקציה רגילה.

forEach

לולאה ייעודית אשר רצה על כל האברים המערך.

```
const names = ['shem', 'max', 'dor'];
```

```
names.forEach((item, i, names)=>{  
  console.log(`${item} index: ${i} ${names}`)  
});
```



item – ערך האיבר במערך.
i – אינדקס האיבר במערך.
names – שם המערך (כל המערך).

forEach

דוגמא ללולאת forEach ששולחת רשימה לתוך אלמנט שיוצג באתר.

```
script.js > ...  
  
const names = ['shem', 'max', 'dor'];  
names.forEach((item, i, names) => {  
    document.getElementById("myDiv").innerHTML +=  
        `<p>${item}</br> index: ${i}<br> ${names}</p>`;  
});
```

index.html

shem
index: 0
shem,max,dor

max
index: 1
shem,max,dor

dor
index: 2
shem,max,dor

- item – ערך האיבר במערך.
- i – אינדקס האיבר במערך.
- names – שם המערך.

filter



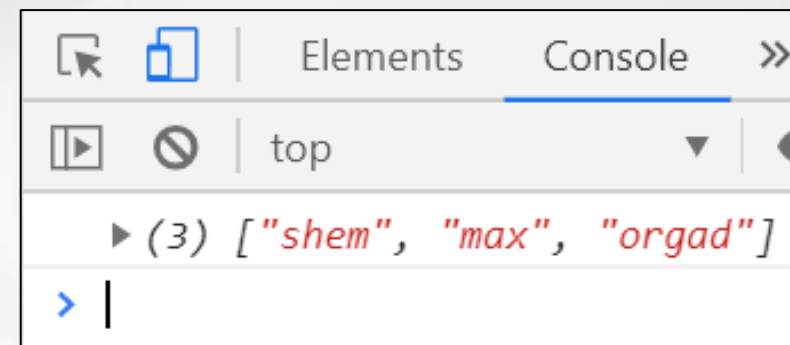
svcollege
ללמוד. לדעת. לעבוד.

יצירת מערך חדש עם סינון של ערכים ספציפיים.

```
const names = ['shem', 'max', 'dor', 'orgad'];
```

```
let afterFilter = names.filter((item)=>  
    item !== 'dor');
```

```
console.log(afterFilter);
```



item – ערך האיבר במערך.

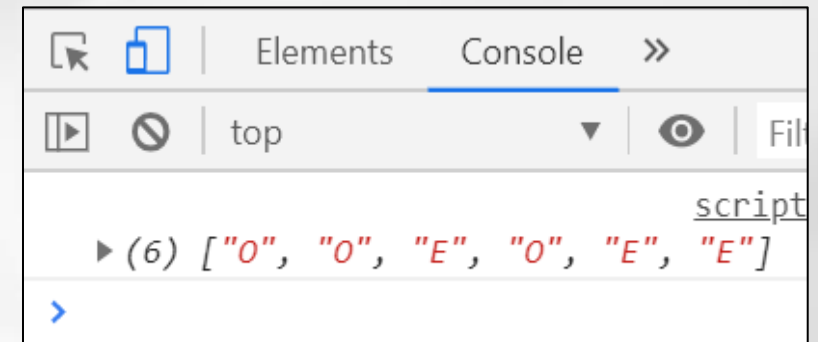
names – שם המערך.

map

יצירת מערך חדש עם ערכים שונים.

```
let arr = [1, 3, 4, 5, 6, 6];  
let arrNew = arr.map((item) => {  
  if (item % 2 == 0) {  
    return 'E'  
  }  
  return 'O';  
})
```

```
console.log(arrNew);
```



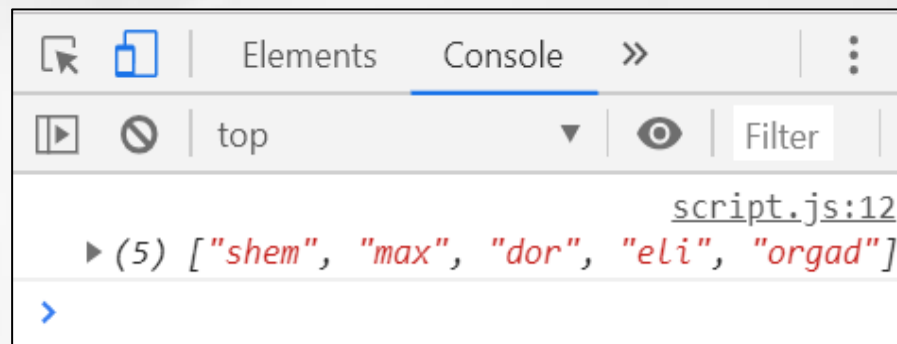
E = Even O = Odd

spread

אפשרות העתקת איברים ממערך אחד למערך אחר.

```
const names = ['shem', 'max', 'dor'];  
let moreNames = [...names, 'eli', 'orgad'];  
// '...' is spread syntax
```

```
console.log(moreNames);
```



svcollege
ללמוד. לדעת. לעבוד.

הצגת פרויקטים