Arrays in JavaScript

فلنفترض بأن أسم المصفوفة Array

-1 لإضافة عنصر في بداية المصفوفة :

Array.unshift('A' , 'B');

-2 لإضافة عنصر في نهاية المصفوفة :

Array.push('A' , 'B');

Array[ Array.length ] = 'A';

-3 لإضافة عنصر في نصف المصفوفة :

Array.splice( 3 , 0 , 'A' );

في المعامل الأول نضع رقم خانت العنصر الذي نريد أن نضيف قبله وهنا وضعنا 3 بمعنى أننا نريد أن نضيف قبل العنصر 3 ,

في المعامل الثاني نحدد كم عنصر نريد أن نحذف وهنا وضعنا 0 بمعنى أننا لا نريد أن نحذف شيء (فقط نريد أن نضيف), و إذا وضعنا رقم 1 بدلاً من 0 سيستبدل العنصر 3 بالعنصر الجديد , في المعامل الثالث وما بعده نضع العناصر الجديدة الذي نريد أضافتها .

-4 لحذف عنصر من بداية المصفوفة :

Array.shift( );

-5 لحذف عنصر من نهاية المصفوفة :

Array.pop( );

-6 لحذف عنصر من نصف المصفوفة :

Array.splice( 1 , 3 );

في المعامل الأول نضع رقم خانت العنصر الذي نريد أن يبدأ الحذف من عنده ,

في المعامل الثاني نضع كم عنصر نريد حذفه ,

ملاحظة : إذا وضعنا متغير قبل عملية الحذف سيتم حفظ العنصر المحذوف في المتغير , مثال :

let i = Array.pop( );

إذا قمنا بطباعة المتغير i في الكونسول سيكون الناتج هوا العنصر المحذوف من المصفوفة .

-7 لاستبدال عنصر من المصفوفة :

Array[ 4 ] = 'A';

رقم 4 هوا رقم خانت العنصر الذي نريد استبداله ,

لمعرفة طول المصفوفة و ارقام خانات عناصرها نقوم بطباعتها في الكونسول و سيظهر لنا طولها و ارقام عناصرها

-8 لعكس عناصر المصفوفة :

Array.reverse( );

-9 للبحث عن عنصر ما ضمن المصفوفة نطبع في الكونسول :

Array.includes('A');

A هوا مثال للعنصر الذي نريد أن نبحث عنه ,

إذا كانت النتيجة true يعني أن العنصر موجود في المصفوفة , اما إذا كانت النتيجة false يعني أن العنصر غير موجود في المصفوفة.

-10 للبحث عن رقم خانت عنصر ما في المصفوفة :

Array.indexOf('A');

A هوا مثال للعنصر الذي نريد أن نبحث عن رقم خانته ,

إذا كان العنصر موجود ضمن المصفوفة سيكون الناتج رقم الخانة الخاصة به , اما إذا كان العنصر غير موجود سيكون الناتج -1

ملاحظة : يجب أن يكون حرف O كبير (Capital O)

-11 للبحث عن عنصر يحمل قيمة معينة :

let result = Array.find( function( i ){

return i > 10;

});

وضعنا return لإرجاع شرط البحث وهوا أن يكون الرقم الذي نبحث عنه أكبر من 10 ,

ملاحظة : سيتم البحث في عناصر المصفوفة من اليسار إلى اليمين , و أول عنصر تجده الدالة يحقق الشرط أي أول عنصر أكبر من 10 ستتوقف عنده و ترجعه في المتغير result الذي وضعناه في بداية التابع ,

و إذا اردنا أن نطبع النتيجة في الكونسول نقوم بطباعة result وهوا المتغير الذي وضعناه في البداية لأننا أرجعنا فيه القيمة .

-12 أما إذا اردنا أن لا تتوقف و أن تعطينا جميع العناصر الذي أكبر من 10 نستبدل find ب filter :

let result = Array.filter( function( i ){

return i > 10;

)};

-13 و إذا اردنا أن نبحث عن رقم خانت العنصر نستبدل find ب findIndex :

let result = Array.findIndex( function( i ){

return i > 10;

)};

ملاحظة : هنا يجب أن يكون حرف I حق Index كبير(Capital I)

-14 للعبور على عناصر المصفوفة :

Array.forEach( i => console.log( i ));

و إذا اردنا أن نعبر على عناصر المصفوفة مع ارقام خاناتها نظيف معامل آخر , مثال :

Array.forEach(( i , x ) => console.log( i , x ));

ملاحظة : هنا يجب أن يكون حرف E كبير (Capital E)

هناك طرق اخرى للعبور على عناصر المصفوفة باستخدام حلقات التكرار for in و for of , و الفرق بن for in و for of هوا أن for of تقوم بطباعة عناصر المصفوفة أما for in تقوم بطباعة ارقام خانات المصفوفة ,

مثال لل for of :

for( let i of Array ){

console.log( i );

};

مثال لل for in :

for( let i in Array ){

console.log( i );

};

ولكن في الحلقة for in إذا وضعنا اسم المصفوفة ثم اسم المتغير داخل اقواس المصفوفة بهذا الشكل Array[i] في أمر الطباعة في الكونسول سيكون استخدام for in مثل استخدام for of و يتم طباعة العناصر بدلاً من ارقام خاناتها ,

مثال :

for( let i in Array ){

console.log( Array[i] );

};

-15 لترتيب عناصر المصفوفة :

Array.sort( );

ملاحظة: هذه الدالة ترتب العناصر اعتمادًا على رموز اليونيكود ,

يعني سترتب النصوص فقط بشكل صحيح أما الارقام فلن ترتبها بشكل صحيح , إذا اردنا أن ترتب الارقام بشكل صحيح نقوم بتمرير دالة داخل sort( ) و نضع فيها الشرط المطلوب و يجب أن تمتلك هذه الدالة معاملين ليتم المقارنة بينهم ,

-16 هذه الدالة ترتب ارقام المصفوفة :

Array.sort( (a , b) => {

return a - b

});

و لأننا كتبناها بالدالة السهمية يمكننا اختصارها بالاستغناء عن return و الأقواس المجعدة و تصبح بهذا الشكل :

Array.sort( (a , b) => a - b );

و إذا اردنا ترتيب ارقام المصفوفة بشكل تنازلي أي من الرقم الأكبر إلى الرقم الأصغر نقوم بعكس عملية الطرح a - b و تصبح b - a

-17 لترتيب مصفوفة تحتوي على نصوص (string) و ارقام (number) :

Array.sort( (a , b) => {

return String (a).localeCompare( b, undefined, { numeric : true, sensitivity : 'base' });

});

ملاحظة : حرف الC في كلمت Compare يجب أن يكون كبير ( Capital C )

و حرف S في كلمت String يجب أن يكون كبير (Capital S)