انواع روش برای تعریف متغییر در javaScript

جی اس یک زبان اسکریپتی است که برای اجرا نیازی به نصب برنامه خاص و سنگینی وجود ندارد

تنها ابزاری که برای خروجی گرفتن نیاز دارید یک مرورگر و یه نوت پد ساده است

و از انجایی که یک زبان متن باز هست خوشبختانه کتابخانه و فریم ورک های بسیار خوبی برای این زبان ساخته شده

مانند jquery که سختی در ک کد ها را کمتر کرده و هم سرعت برنامه نویس را بالا میبرد

یا مانند nodejs که انقلاب عظیمی در زبان js انجام داده که این زبان به یک زبان تحت سرور نیز تبدیل کرده

یا فریم ورک هایی مانند react native x و electron که میتوان با انها برای اکثر دیوایس ها از جمله گوشی و سیستم های دسکتاپ ، اپلیکشین ساخت.

در جاوا اسکریپت بر خلاف زبان سیشارپ یا جاوا که برای تعریف متغییر باید نوع انرا مشخص کنند، در جاوا اسکریپت نیازی به مشخص کردن نوع متغییر نیست و تنها با استفاده از کلمه کلیدی var میتوان هر نوع متغییری تعریف کرد

ساختار اصلی

var <*variable-name*>;

var <*variable-name*> = <*value*>;

Var=variable

var one = 1;

متغیر مقدار عددی ذخیره میکند

var two = 'two';

متغییر مقدار رشته ای در خود ذخیره میکند

var three;

متغییر بدون مقدار اولیه

var one = 1, two = 'two', three;

تعریف چند متغییر با یک کلمه کلیدی و جدا کاما

one = 1;

two = 'two';

بدون کلمه کلیدی

var one

=

1, two

= "two"

همراه با اسپیس و اینتر

Var a=1;

مقدار عددی

Var b=’one’;

مقدار رشته ای

Var c=1.1;

مقدار دهدهی

Var d=true;

مقدار بولین

Var e=null;

مقدار تهی

ارایه ها در جاوا اسکریپت

var *array\_name* = [*item1*, *item2*, ...];

ساختار اصلی

var cars = ["Saab", "Volvo", "BMW"];

ارایه ای با 3 عضو

var cars = [  
 "Saab",  
 "Volvo",  
 "BMW"  
];

تعریف در خط

var cars = new Array("Saab", "Volvo", "BMW");

نوع دیگر

تابع رندوم در جی اس

با فرخوانی این تابع مقدار بازگشتی که تابع برای ما میفرستد یک عدد اعشاری بین 0 و 1 است(خود عدد 1 در مقادیر وجود ندارد)

Math.random();

اما اگه ما یک عدد رندوم بین 0 تا 100 بخوایم کافیست تابع رندوم را ضربدر عدد 100 بکنیم و با استفاده از تابع

Floor

اعشار اضافی را حذف کنیم

Math.floor((Math.random() \* 100));

جستجو باینری

var steps = 0;

var list = [2, 9, 4, 1, 6, 7, 3, 5, 8]

document.getElementById("Method1").innerHTML = list;

function binarySearch(list, value) {

var start = 0

var stop = list.length - 1

var middle = Math.floor((start + stop) / 2)

while (list[middle] !== value && start < stop) {

if (value < list[middle]) {

stop = middle - 1

steps++;

} else {

start = middle + 1

steps++;

}

middle = Math.floor((start + stop) / 2)

}

return (list[middle] !== value) ? -1 : middle

}

function fun() {

list.sort();

document.getElementById("Method2").innerHTML = list;

var input = parseInt(prompt("عدد مورد نظر را وارد کنید"));

document.getElementById("Method3").innerHTML = "element " + binarySearch(list, input);

document.getElementById("Method4").innerHTML = "steps " + steps;

steps=0;

جستجو خطی

var steps = 0;

var list = [2, 9, 4, 1, 6, 7, 3, 5, 8]

document.getElementById("Method1").innerHTML = list;

function linearSearch(list, value) {

for (var i = 0; i < list.length; i++) {

if (list[i] == value) {

return i;

}

steps++;

} return null;

}

function fun() {

var input = parseInt(prompt("عدد مورد نظر را وارد کنید"));

document.getElementById("Method2").innerHTML = "element " + linearSearch(list, input);

document.getElementById("Method3").innerHTML = "steps " + steps;

steps=0;

}

جمع ماتریس دوبعدی

var items = [

[1, 2],

[3, 4],

[5, 6]

];

var items2 = [

[7, 8],

[9, 10],

[11, 12]

];

for (let i = 0; i < 3; i++) {

for (let j = 0; j < 2; j++) {

var a =items[i][j]+items2[i][j];

console.log(a);

}

}

ضرب ماتریس دو بعدی

function multiplyMatrix(m1, m2) {

var result = [];

for (var j = 0; j < m2.length; j++) {

result[j] = [];

for (var k = 0; k < m1[0].length; k++) {

var sum = 0;

for (var i = 0; i < m1.length; i++) {

sum += m1[i][k] \* m2[j][i];

} result[j].push(sum);

}

}

return result;

}

var list = [[2, 4],

[3, 4]

];

var list2 = [[2, 4],

[3, 4]

];

console.log(multiplyMatrix(list, list2)[0]);

مرتب سازی حبابی

در ای نوع مرتب سازی با استفده از دو حلقه فور شروع به مرتب سازی میکنید

به این صورت که خانه اولی با خانه بعدی رو چک میکند اگه خانه بعدی بزرگ تر بود با استفاده از یک متغییر مخزن جایی محتوای دو خانه عوض میشود

این حلقه ها تا جایی ادامه پیدا میکنند که تمام خانه ها مرتب شوند

function swap(arr, first\_Index, second\_Index){

var temp = arr[first\_Index];

arr[first\_Index] = arr[second\_Index];

arr[second\_Index] = temp;

}

function bubble\_Sort(arr){

var len = arr.length,

i, j, stop;

for (i=0; i < len; i++){

for (j=0, stop=len-i; j < stop; j++){

if (arr[j] > arr[j+1]){

swap(arr, j, j+1);

}

}

}

return arr;

}

console.log(bubble\_Sort([3, 0, 2, 5, -1, 4, 1]));

مرتب سازی انتخابی

روش کار به این شکل است که بزرگ ترین خانه را انتخاب میکنیم و ان را به انتهای لیست میبریم

و دوباره شروع به انتخاب بزرگ ترین خانه میکنیم و ان را کنار خانه قبلی یعنی در انتهای لیست میگذاریم

function swap(items, firstIndex, secondIndex) {

var temp = items[firstIndex];

items[firstIndex] = items[secondIndex];

items[secondIndex] = temp;

}

function selectionSort(items) {

var len = items.length,

min;

for (i = 0; i < len; i++) {

min = i;

for (j = i + 1; j < len; j++) {

if (items[j] < items[min]) {

min = j;

}

}

if (i != min) {

swap(items, i, min);

}

}

return items;

}