**ФУНКЦИИ И ИХ ВИДЫ**

Определение и синтаксис

Функция – конструкция для повторного вызова одной и той же части кода, выполнения одних и тех же действий, в разных частях программы.  
Одним из принципов программирования является принцип DRY (don’t repeat yourself) – в программе следует избегать дублирования кода.

function functionName([argument1], [argument2]) {

statements

[return]

}

function – ключевое слово для объявления функции;  
functionName – имя функции для дальнейшего обращения к ней;  
argument – аргументы передаваемые в функцию (необязательно);  
statements – код, который является телом функции (код выполняемый после вызова функции);  
return – ключевое слово после которого указывается возвращаемое значение (необязательно).

function sayHello(name) {

let upperCaseName = name.toUpperCase();

return "Wellcome Mr. " + name;

}

let yourName = "Johnson";

sayHello(yourName); // "Wellcome Mr. JOHNSON"

// some code

sayHello(yourName); // "Wellcome Mr. JOHNSON"

// some code

sayHello("Smith"); // "Wellcome Mr. SMITH"

При именовании функции следует придерживаться такой рекомендации: первое слово это обязательно глагол, который выражает производимое действие, далее – описание.

getElementById();

loadData();

bindEvents();

generateUID();

Функция должна выполнять только то действие которое заложено в ее названии.

Функция без аргументов

function getCurrentYear() {

let date = new Date();

return date.getFullYear();

}

const currentYear = getCurrentYear();

console.log(currentYear); // текущий год

Функция без return

function noReturn() {

var sum = 0;

for(var i = 0; i < 10; i++) {

sum += i;

}

console.log(sum);

}

noReturn(); // undefined

Если в функции отсутствует ключевое слово return, то она вернет специальное значение undefined.  
Код, после возвращения значения из функции, то есть после return, не выполняется.

function checkReturn() {

console.log("Before return");

return true;

console.log("After return");

}

checkReturn(); // "Before return"

Function declaration и expression

Существует несколько способов объявления функции, это Function Declaration и Function Expression  
Синтаксис function declaration

function funcDeclaration() {

console.log("Function declaration");

}

Синтаксис function expression

var funcExpression = function () {

console.log("Function expression");

}

Отличия function declaration и function expression:

* • функциональные выражения (function expression) используются когда тело функции надо присвоить в переменную, например, для передачи функции как параметра в другую функцию;

var firstFunction = function () {

console.log("Run firstFunction.");

};

function runFunction(param) {

console.log("Run runFunction");

param();

};

runFunction(firstFunction);

* • или сделать ее свойством объекта;

var showFullName = function (firstName, secondName) {

return firstName + " " + secondName;

}

var obj = {

fullName: showFullName

}

obj.fullName("Keanu", "Reeves");

* • function declaration можно вызывать как до, так и после ее объявления;

showFullName ("John", "Karter"); // "John Karter"

function showFullName(firstName, secondName){

console.log(firstName + " " + secondName);

}

showFullName ("Garry", "Miller"); // "Garry Miller"

* • function expression доступна для вызова после того как объявлена;

showFullName ("John", "Karter"); // ошибка!

var showFullName = function(firstName, secondName){

console.log(firstName + " " + secondName);

}

showFullName ("Garry", "Miller"); // "Garry Miller"

Анонимная функция

Анонимная функция – функциональное выражение, которое не записывается в переменную, у которого отсутствует имя.

setTimeout(function(){

console.log("Anonymous function.");

}, 2000);

Анонимная функция используется когда отсутствует необходимость вызвать функцию по имени где-то в коде, то есть она объявляется в месте ее использования.

Самовызывающаяся функция (IIFE)

Самовызывающаяся функция (Immediately Invoked Function Expression(IIFE)) – синтаксическая конструкция с помощью которой можно вызвать функцию в месте ее определения.

(function () {

console.log("IIFE");

}());

(function () {

console.log("IIFE");

})();

В такую функцию можно так же передавать параметры.

(function sum(a, b) {

console.log(a + b); // 5

})(2, 3);

Стрелочная функция (arrow function)

Стрелочные функции (arrow function) – это более короткая синтаксическая запись обычных функций. Базовый синтаксис:

var arrowFunction = (argument) => statements;

arrowFunction – имя функции;  
argument – аргументы передаваемые в функцию (необязательно);  
statements – выполняемый код или возвращаемое значение.  
Если функция короткая, ее можно записать в одну строку, без return и фигурных скобок:

let getSum = (a, b) => a+b;

Без фигурных скобок, функция возвращает значение, даже без return.  
Функцию с одним параметром можно записать без скобок:

let getSum = a => a + 2;

В случае использования фигурных скобок необходимо указывать ключевое слово return.

let getSum = (a, b) => { return a+b; }

Эта же функция записанная как function expression:

let getSum = function(a, b) { return a+b; }

Функцию без параметров записывают с пустыми фигурными скобками.

let getYear = () => {

return new Date().getFullYear();

}

Стрелочная функция как анонимная функция.  
Самый распространненый вариант использование анонимных стрелочных функций, это использование в функциях (методах) итераторах:

[1,2,3,4].map( item => item \* 2 ); // [2, 4, 6, 8]

item => item \* 2 – анонимная стрелочная функция переданная методу map.  
Дательнее некоторые из них будут рассмотрены далее в курсе.

Часть1 Веберете правильный ответ

1. Зачем нужны функции в JavaScript?

Служат для математических вычислений

Для связывания переменных между собой

Для повторного вызова одного и того же кода в разных частях программы

**7**Баллы

2. Выберите синтаксически корректную JavaScript команду для вызова функции "callFunction"

function callFunction()

callFunction()

new callFunction()

$echo callFunction()

**7**Баллы

3. Выберите метод JavaScript позволяющий выполнять произвольный код через заданные промежутки времени

callCode()

timer()

setInterval()

setTimeOut()

**7**Баллы

4. Выберете подходящее имя функции которая выполняет получение ссылки

link

setLink

getUrl

getData

**7**Баллы

5. Сколько аргументов можно передать функции?

Один

Два

Сколько угодно

Ни одного

**7**Баллы

6. Почему в консоль выводится undefined?

[code lang="javascript"]function getSum(first, second) { let sum = 0; sum = first + second; } console.log("result = " + getSum(5,6)); // result = undefined [/code]

неправильный вызов функции

не объявлены переменные first и second

отсутствует ключевое слово return

**7**Баллы

7. Как правильно объявить функцию getData?

function getData: { }

func getData = function () {}

new getData = { }

function getData() { }

**7**Баллы

8. Что вернет функция getSecondName?

[code lang="javascript"]let secondName; function getSecondName() { secondName = "Smith"; return secondName; secondName = "Damon"; return secondName; }[/code]

Smith

Damon

Ошибку

**7**Баллы

9. Какое утверждение верно для function expression?

Функция доступна для вызова после того как объявлена

Функцию доступна для вызова, до того как объявлена

Оба варианта верны

Ни один из вариантов

**7**Баллы

10. Что выведет этот код?

[code lang="javascript"]var func = function () { console.log(func === func); } func();[/code]

false

true

undefined

ReferenceError

**7**Баллы

11. Функциональное выражение, которое не записывается в переменную, у которого отсутствует имя, называется…

function expression

function declaration

анонимная функция

функция не имеющая аргументов

**7**Баллы

12. В этой функции допущена ошибка, в какой строке?

[code lang="javascript"] 1. function resetTextArea(event) { 2. gettext.value = settext.value = ""; 3. counterLetters.innerHTML = 0; 4. counterLetters.style.color = ''; 5. }();[/code]

1

5

2, 4

1, 5

**7**Баллы

13. Выберете правильный вариант записи следующей функции:

[code lang="javascript"]function getFullName (firstName, secondName) { return firstName + “ “ + secondName; }[/code]

var getFullName = (firstName, secondName) => firstName + " " + secondName;

var getFullName = function (firstName, secondName) { return firstName + " " + secondName; }

var getFullName = (firstName, secondName) => {return firstName + " " + secondName; }

Все вышеперечисленные

**7**Баллы

14. Бывают ли стрелочные функции анонимными?

Нет, так как это функциональное выражение, которое обязательно записывается в переменную

Да, как и обычные функции

Стрелочные функции будут введены только в ES7

**7**Баллы

15. Что выведет данный код?

[code lang="javascript"]let firstName = "Matt"; let secondName = "Damon"; let fullName = (firstName, secondName) => firstName + " " + secondName; console.log(fullName());[/code]

Matt Damon

undefined undefined

Ошибку