**Вопросы 💎**

1. Какие есть способы объявления функций?
   1. Function Declaration (основной)- функция объявлена, как отдельная инструкция в основном потоке кода
   2. Function Expression- функция создается как часть выражения
   3. Стрелочные функции (ES6).
2. Приведите примеры вызова одной и той же функции всеми известными вам способами.
3. // Function Declaration
4. function sum(a,b){
5. return a+b;
6. }
7. const result = sum(2,3);
8. console.log (result);
9. // Function Expression
10. const sum = function(a,b){
11. return a+b;
12. };
13. const result = sum(2,4);
14. console.log (result);
15. // Стрелочные функции
16. const sum = (a, b) => a + b;
17. console.log( sum(1, 2) );
18. В чём разница между тестированием и отладкой (дебаггингом)? А что такое логирование?

Тестирование направлено на проверку работоспособности программы и выявление ошибок, в то время как отладка фокусируется на идентификации и исправлении этих ошибок. Однако эти два процесса взаимосвязаны, поскольку отладка обычно осуществляется на основе результатов тестирования, где ошибки были обнаружены.

Логирование- фиксирование происходящих действий в консоль.

1. В чём разница между Function Expression и Function Declaration?

Основная разница между Function Expression (функциональным выражением) и Function Declaration (объявлением функции) в JavaScript связана с тем, как интерпретатор обрабатывает эти конструкции во время фазы подъема (hoisting) и доступности функции в коде.

*Function Declaration:*

* Объявление функции поднимается (hoisted) вверх области видимости, поэтому функцию можно вызывать до ее фактического объявления в коде.
* Функция доступна внутри всей области видимости, включая код, написанный перед объявлением функции.
* Не требуется заключать объявление функции в выражение или присваивать его переменной.

*Function Expression:*

* Функциональное выражение не поднимается вверх области видимости, поэтому функцию можно вызывать только после ее фактического объявления в коде.
* Функция доступна только после того, как переменной, к которой она присвоена, было присвоено значение.
* Функциональное выражение требует присвоения его переменной или использования в выражении.

1. Что делает функция console.log()?

Выводит сообщение или значение в консоль разработчика. Вывод может быть текстовым сообщением, строкой, числом, объектом, массивом или другими типами данных.

1. По какой причине, мы можем использовать до объявления функцию, объявленную через Function Declaration, но не можем ту, которая объявлена через Function Expression ?

Отличие состоит в том, когда создаётся функция движком JavaScript.

**Function Expression создаётся, когда выполнение доходит до него, и затем уже может использоваться.**

После того, как поток выполнения достигнет правой части выражения присваивания let sum = function… — с этого момента, функция считается созданной и может быть использована (присвоена переменной, вызвана и т.д.).

С Function Declaration всё иначе.

**Function Declaration можно использовать во всём скрипте (или блоке кода, если функция объявлена в блоке).**

Другими словами, когда движок JavaScript готовится выполнять скрипт или блок кода, прежде всего он ищет в нём Function Declaration и создаёт все такие функции. Можно считать этот процесс «стадией инициализации».

И только после того, как все объявления Function Declaration будут обработаны, продолжится выполнение.

В результате, функции, созданные, как Function Declaration, могут быть вызваны раньше своих определений.

1. Что такое глобальная зона видимости функций?

В глобальной зоне видимости функции и переменные доступны из любой части программы, включая другие функции или блоки кода. Они могут быть объявлены на верхнем уровне скрипта, за пределами каких-либо функций.

1. Что вернёт код? **28**

function showX(x)

{

return x;

}

console.log(showX(28));

1. Что выведет этот код? **5**

function sum(x, y=0) {

return x+y;

}

console.log(sum(5));

1. Что выведет код? Ошибку , foo не объявлена

console.log(foo);

let foo = "Cat";

console.log(foo);

**Практическое задание 💎**

**\*\*\* Задание под звездочкой**