**function** writer(title, price, description) - это пустой шаблон

**function –** функция. Writer – название функции. title, price, description – аргументы.

Если прописать writer(‘milk’, 32, ‘white’ ) – подтянется шаблон и будет выведено milk, 32, white.

milk = title, 32 = price, white = description

Аргументом функции может быть целый массив.

function writer(title, price, description) {  
 document.write(`  
 <div>  
 <h2>${title} ${price}</h2>  
 <p>${description}</p>  
 </div>`  
 )  
 }  
 writer('milk', 23, 'white')

Функция **return**. Что бы вывести результат функции в аргумент(переменную) и использовать в будущем, используем return.

function calc (a, b) {  
 let result = a+b;  
 return result;  
}  
***console***.log(calc(10,20));

Функция может возвращать множественные вычисления. Для этого, математические действия заводим в обьект:

function supercalc (num1, num2) {  
 let obj = {  
 dif: num1 - num2,  
 add: num1 + num2  
 };  
 return obj;  
}  
***console***.log(supercalc(10, 20))

# Переменная arguments. Позволяет выводить в консоль неопределенное кол-во аргументов: Можно вывести 3, а можно изменить на любое другое кол-во:

function calcArgum () {  
 let basket = 0;  
 for (const num of arguments) {  
 basket = basket + num;  
 }  
 return basket;  
}  
***console***.log(calcArgum(10, 20, 30))

**rest** аргумент. Выводит в массив остаток от определенной функции. X - остаток

function rees (y, z, ... x) {  
 ***console***.log(x)  
}  
rees (1,2,3,4,5);

Функции **замыкания.** Это возможность функции сохранять посылание на внешние обьекты. Возможность сделать приватность, т.е. недоступность изменять параметры переменной:

Выводится функция outer, которая ссылается на inner. Inner имеет внешнюю переменную 100. В данном случае происходит зымыкание inner на переменную 100.

function outer() {  
 let x = 100;  
 function inner() {  
 return x;  
 }  
 return inner;  
}  
***console***.log(outer()())

Функция **this.** Работает в пределах своего обьекта. Для того, что бы эта функция заменяла собою имя данного обьекта. Вместо this.name могло стоять z

let ***z*** = {  
 name: 'vasya',  
 greeting: function () {  
 ***console***.log('hello I ' + this.name);  
 // console.log(this)  
 }  
}  
***z***.greeting();

**get**Name() – get это закрытая функция, по нему можно достать например имя и его нельзя изменить. Name – свободное название елемента объекта, к которому обращается get. user

function foo(name, age) {  
 let user = {name, age};  
 return {  
 getName() {  
 return user.name;  
 }  
 }  
}  
let ***name2*** = foo('Vasya', 20)  
***console***.log(***name2***.getName());