**Function**

**Подпрограмма (программа внутри программы)**

***Разделяй и властвуй*** – главное правило программиста.

**Функция – это значение, представляющее действие**

f(a,b,c) {}; - a,b,c – параметр

f(10,20,30) {}; - 10,20,30 – аргумент

undefined – тривиальный тип данных который обозначает что в переменной ничего нет, занимает минимальное пространство.

**Области видимости**

1) Разрешение переменных (разрешить уравнение)

var – *lacal, создает переменную для текущей области видимости*

**ошибка is not defined** – эта переменная не была найдена в своей области видимости, вышестоящей и глобальной области видимости

function sayHi() {

alert( "Привет" );

}

Function Declaration (Объявление Функции)

let sayHi = function() {

alert( "Привет" );

};

Function Expression (Функциональное Выражение)

function() {}; - Функция без имени - анонимная функция, видна в своей области видимости

function factory() {

    let count = 0;

    return function() {

        console.log(count++);

    }

}

factory();

пример замыкания count

**Во время первого пробега интерпретатора сохраняются var переменный и function declaration**

**LecsicalEnvarement** Замыкание — это комбинация функции и лексического окружения, в котором эта функция была определена. Другими словами, замыкание дает вам доступ к [Scope](https://developer.mozilla.org/ru/docs/Glossary/Scope) внешней функции из внутренней функции. В JavaScript замыкания создаются каждый раз при создании функции, во время ее создания.