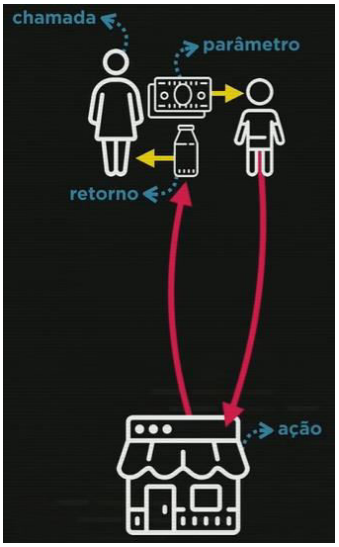
AULA 16 – FUNÇÕES

Exemplo prático: mãe

→ O seu filho tem um conjunto de tarefas para fazer (varrer quintal, arrumar o quarto). Na linguagem de programação, chamamos isso de **funções**

Contexto: sua mãe entrega dinheiro para você ir ao mercado e comprar leite



→ Quando a sua mãe te chama e te passa a ordem, chamamos isso de **chamada**
OBS: as vezes você pode ter uma funcionalidade de por exemplo toda quarta-feira 8h da manhã pegar o leite, ou seja, não precisa ter o "disparo" de sua mãe. A chamada nem sempre é verbalizada, as vezes é automatizada

→ Você não pode ir ao mercado comprar leite sem dinheiro, ou seja, sem uma coisa de entrada. Chamamos isso de **parâmetro**. Nesse contexto existe dois parâmetros: o dinheiro (que usamos para comprar) e o que sua mãe queria (o produto)

→ Ir até o mercado (da maneira que for), chamamos de **ação**

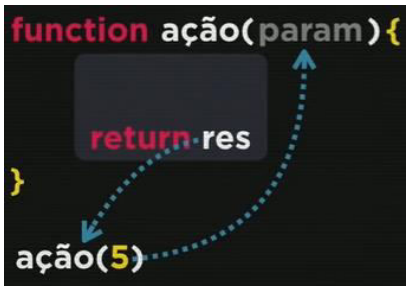
→ No final, quando você volta para casa e entrega o leite para sua mãe, temos o

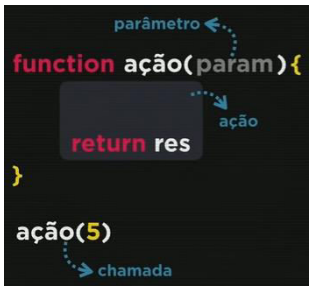
"Toda função pode ter os seguintes fatores: uma chamada, um conjunto de parâmetros, uma ação e um retorno. Nem toda função usa parâmetros, nem toda função tem retornos, mas elas são muito importantes para dar mais funcionalidades às nossas tarefas

Funções são **ações** executadas assim que são **chamadas** ou em decorrência de algum **evento**

Uma **função** pode receber **parâmetros** e retornar um **resultado**

Lembrando: quando utilizávamos o DOM, não tínhamos retorno!





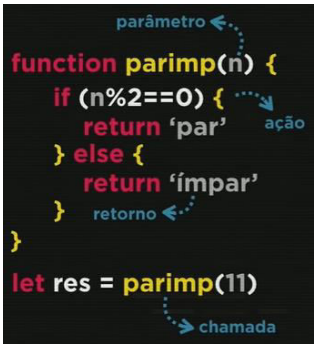
→ Agora damos um upgrade nesse conceito de função

→ Dentro dos parênteses eu sou capaz de colocar **parâmetros**, esses parâmetros vão ser processados pela **ação** e podem ter um **resultado**. Essa ação não executa por padrão, eu tenho que ter uma **chamada**!

→ Essa chamada eu coloco na linha de baixo: ação(5) → isso significa que eu estou chamando a ação (que é o nome da minha função). Então, na hora que eu executar esse código, ele vai pular a parte da função e vai executar somente o ação(5). Eu vou executar esse ação passando um valor (que no caso é 5). Esse valor 5 vai para parâmetro, então o parâmetro vai valer 5 nesse exemplo

→ No final, eu posso dar o retorno de um **resultado**. Esse retorno de resultado vai ser voltado de novo para a chamada

Exemplo: verificar se o número é par ou ímpar



Se eu mandar mostrar o res na tela, ele vai me retornar ímpar!

Exemplos no Visual Studio:

```
function parimpar(n){
    if (n%2==0) {
        return 'par'
    } else {
        return 'impar'
    }
}
let res = parimpar(4012)
console.log(res)
```

```
Info: Start process (5:26:38 PM)
par
Info: End process (5:26:39 PM)
```

```
function soma(n1,n2) {
    return n1 + n2
}
console.log(soma(2,5))
```

```
Info: Start process (5:27:42 PM)
7
Info: End process (5:27:42 PM)
```

```
function soma(n1,n2) {
    return n1 + n2
}
console.log(soma(2))
```

```
Info: Start process (5:29:01 PM)
NaN
Info: End process (5:29:01 PM)
```

→ Como a chamada só veio de um número, o JavaScript definiu o n2 como indefinido, por isso o resultado foi NaN (Not a Number)

Isso pode ser resolvido: considerar 0 caso não passe o n1 ou o n2

```
function soma(n1=0,n2=0) {
    return n1 + n2
}
console.log(soma(13))
```

```
Info: Start process (5:31:06 PM)
13
Info: End process (5:31:06 PM)
```