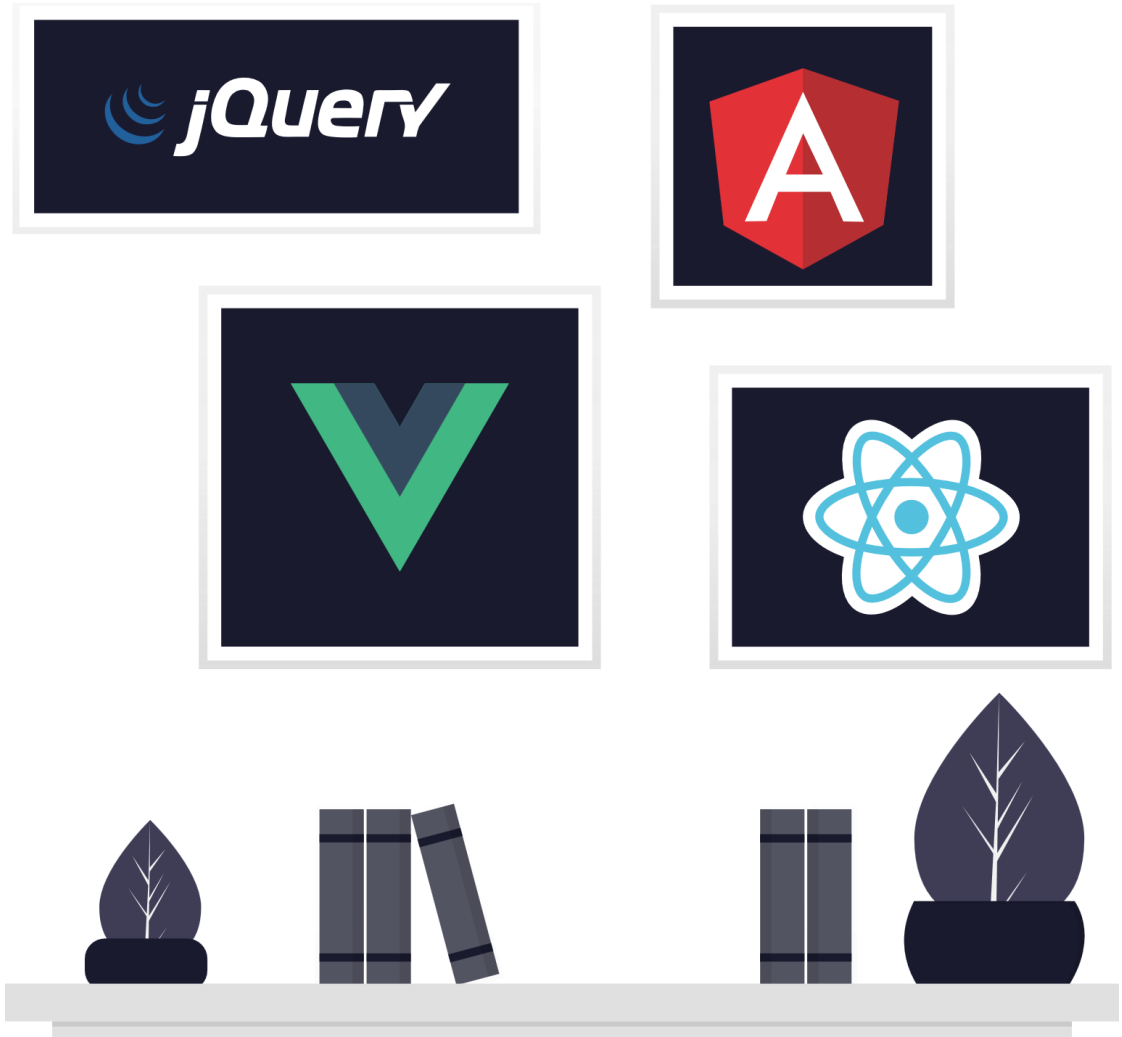
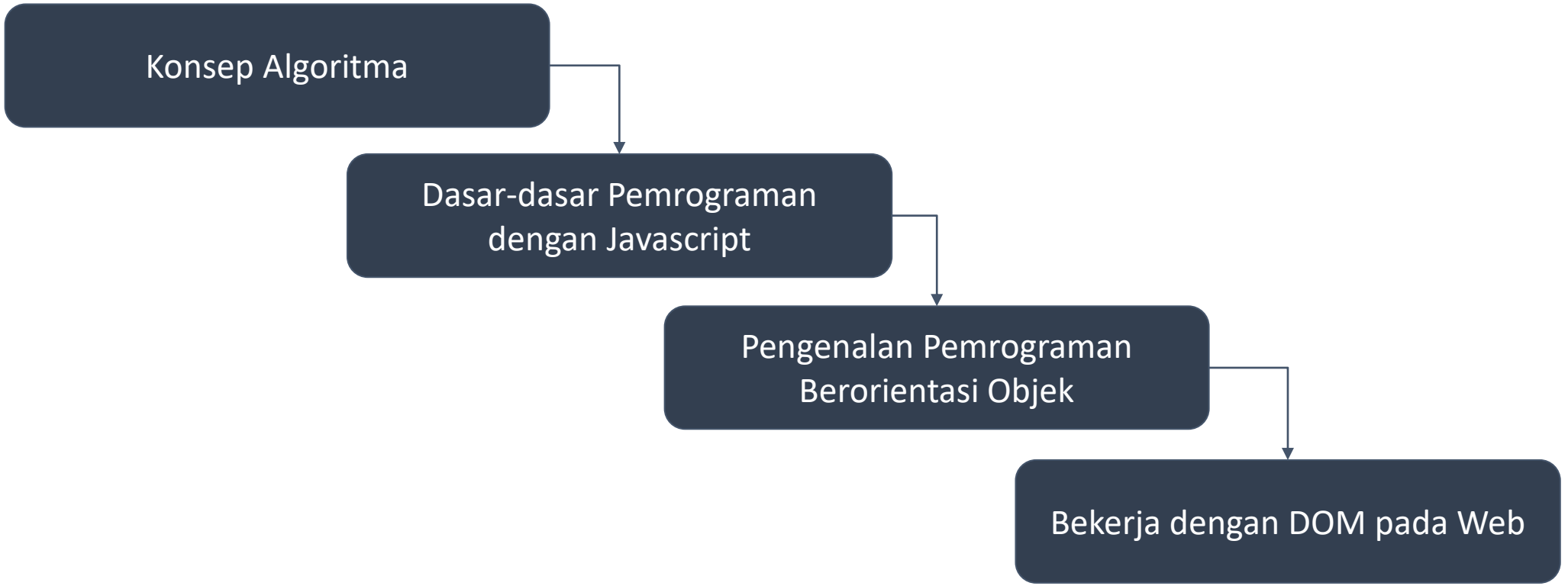
# PEMROGRAMAN JAVASCRIPT



- **KODE** : MK03
- **MATAKULIAH** : PEMROGRAMAN JAVASCRIPT
- **SKS** : 4 SKS
- **SEMESTER** : 1 (SATU)


Konsep Algoritma
Dasar-dasar Pemrograman dengan Javascript
Pengenalan Pemrograman Berorientasi Objek
Bekerja dengan DOM pada Web

# RENUNGAN ISLAMI

"Aku berkumpul dengan kaum Sufi. Manfaat yang dapat kuambil dari mereka hanya dua nasihat. Pertama, waktu bagaikan pedang. Jika tak kau gunakan untuk memotong, kaulah yang akan terpotong. Kedua, jiwamu, jika tidak kau sibukkan untuk kebenaran pasti akan balik menyibukkan dirimu sendiri dengan kebatilan.“

**- Imam Asy-Syafi’I -**

# JAVASCRIPT FUNCTION

# FUNCTION

**Function** pada JavaScript adalah sekumpulan kode yang dirancang untuk melakukan tugas tertentu. Sebuah fungsi JavaScript dijalankan ketika ada yang memanggilnya.

# FUNCTION

## Dua Jenis Function :

- Prosedur mengeksekusi sejumlah perintah secara berurutan (istilah lainnya subroutine/subprogram)
- Fungsi/Function menerima inputan, mengeksekusi sejumlah perintah, dan mengembalikan nilai

# FUNCTION

- Deklarasi

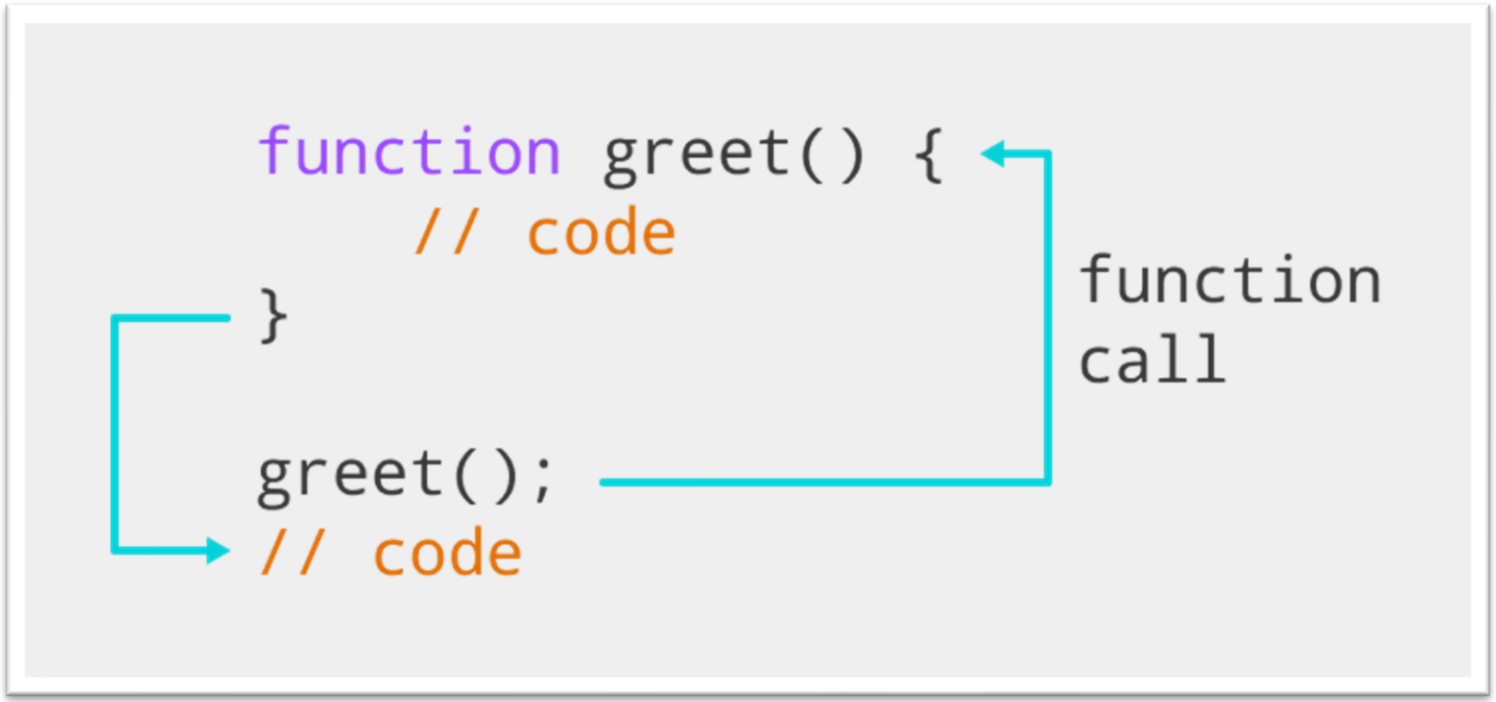
```
function namaFungsi ( <parameterFungsi> )
{
// isi dari fungsi
}
```

- Pemanggilan

```
kirimPesan();
```

Dalam beberapa bahasa pemrograman, prosedur dan fungsi dibedakan (contohnya pascal), tetapi di javascript (dan beberapa bahasa lainnya) perbedaan itu tidak dinyatakan eksplisit. Jadi keduanya bisa dianggap sama di Javascript (sebagai function).

# FUNCTION



# FUNCTION

- **Return Function**

Jenis fungsi yang mengembalikan sebuah nilai

```
function tambah(a1, a2){
    let hasil = a1 + a2;
    return hasil;
}
let x = 10;
let y = 20;
console.log(tambah(50,30)); // output = 80
console.log(tambah(x,y)); // output = 30
```

# FUNCTION

- **Return Function**

  Kita bisa mengembalikan nilai dari fungsi (fungsi return) dengan cara memberikan directive return.

- Directive return bisa diletakkan di mana saja pada badan fungsinya. Pada saat eksekusi fungsinya mencapai return, maka fungsinya akan berhenti dan mengembalikan nilai yang ada di sebelah kanan directive return tersebut.
- Jika fungsi tidak mengembalikan nilai apapun, maka sama seperti fungsi tersebut mengembalikan nilai undefined

# FUNCTION

- **Void Function**

Jenis fungsi yang tidak mengembalikan nilai apapun

```
function salam(nama) {
    console.log(“Selamat Pagi ", nama)
}
let siswa = "Fulan";
salam(siswa); // output : Selamat Pagi Fulan
salam(“Siswa"); // output : Selamat Pagi Siswa
```

# PARAMETER FUNCTION

- Parameter fungsi adalah nilai yang bisa dimasukkan pada fungsinya, seperti nama pada program salam, atau nilai a dan b yang akan dijumlahkan pada program penjumlahan.

  salam("Inaya")

  tambah(50,30)

- Parameter fungsi bersifat opsional pada saat fungsi dideklarasikan, artinya boleh ada ataupun tidak. Tetapi setelah fungsi dibuat, pemanggilan fungsi harus mengikuti format tersebut.
- Argumen adalah nilai yang diberikan pada saat fungsinya dipanggil. Contoh pada kasus di atas adalah Inaya, 50, dan 30.

# PARAMETER FUNCTION

Saat fungsi dipanggil tanpa diberikan argumen, maka nilai parameter fungsi yang berkaitan akan menjadi undefined.

```
> function salam(nama="Anda")
  {
      console.log("Assalamu'alaikum, apa kabar",nama,"?")
  }
<- undefined
> salam()
  Assalamu'alaikum, apa kabar Anda ?
<- undefined
>
```

```
> function salam(nama)
  {
      console.log("Assalamu'alaikum, apa kabar",nama,"?")
  }
<- undefined
> salam()
  Assalamu'alaikum, apa kabar undefined ?
<- undefined
>
```

Kita bisa menambahkan nilai (default) di dalam parameternya sebagai contoh pada gambar di kanan.

## Standarisasi Function

Fungsi biasanya dituliskan dengan kata kerja (get, check, create, dll)

Sebuah fungsi mengerjakan 1 hal tertentu. Jika proses yang dikerjakan banyak/panjang, ada baiknya dibuat fungsi terpisah yang dipanggil dalam fungsi tersebut.

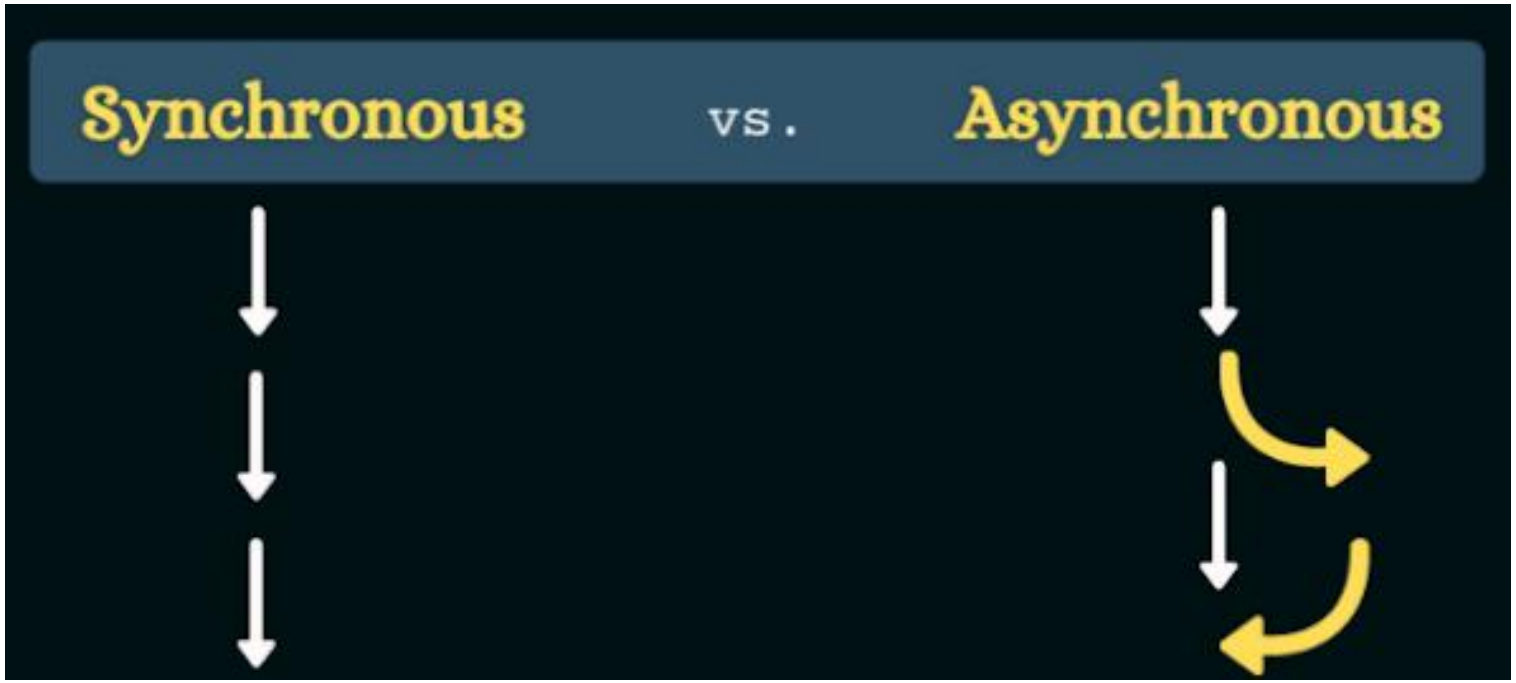
# Standarisasi Function

Nama fungsi memang sebaiknya jelas dan deskriptif, sehingga bisa menjelaskan untuk apa fungsi tersebut digunakan dari namanya. Tetapi ada beberapa pengecualian.

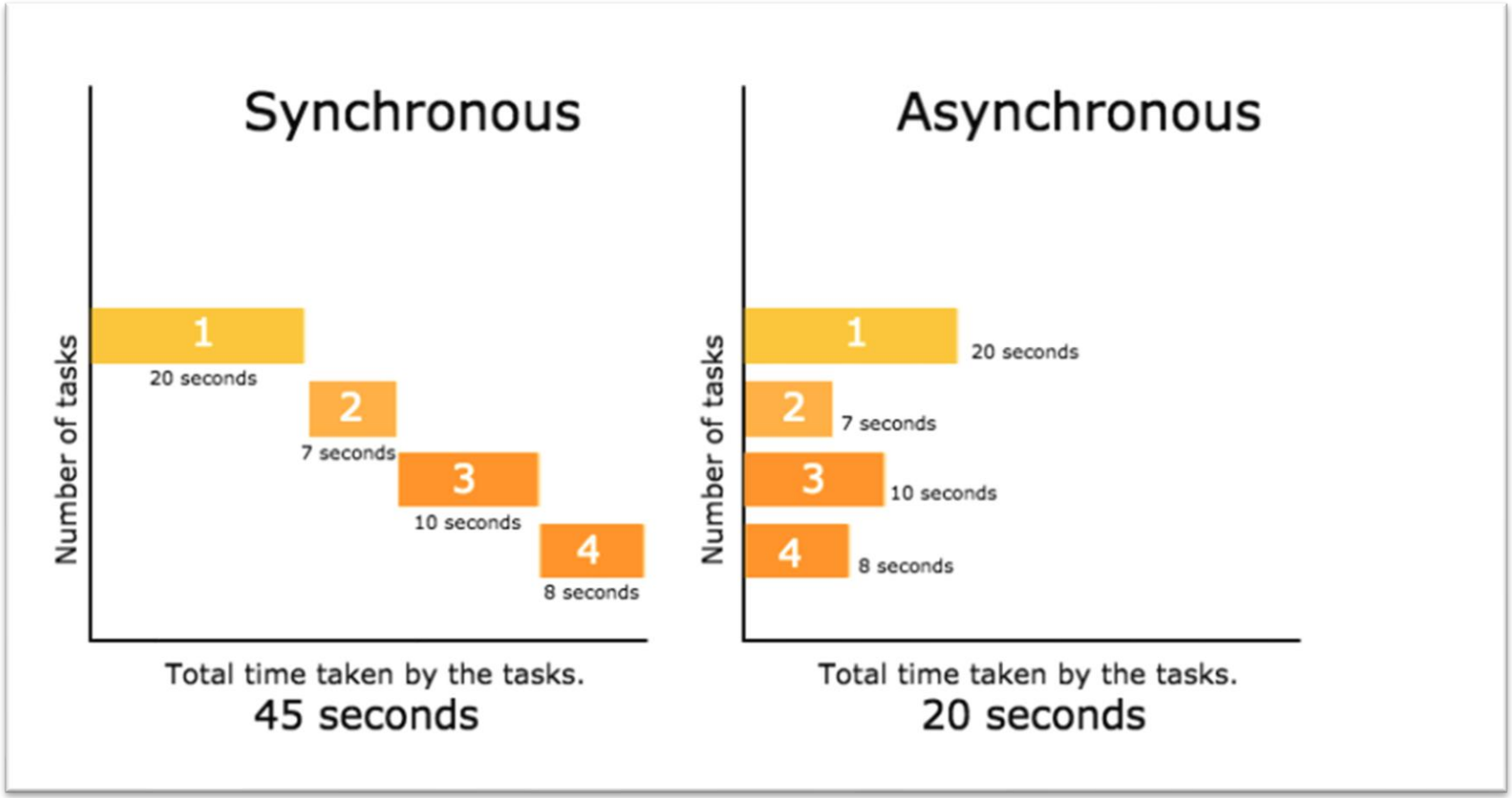
Contohnya fungsi yang sangat sering digunakan. Fungsi yang sangat sering digunakan akan lebih baik jika dinamai dengan nama yang singkat, contonya :

- Di framework jQuery ada fungsi yang namanya $
- Di library Lodash ada fungsi utamanya yang bernama _
- Di java fungsi utama disebut main, sedangkan model menggunakan Setter dan Getter

# ASYNCHRONOUS



# ASYNCHRONOUS



# ASYNCHRONOUS

## Synchronous (Sinkron)

Synchronous berarti berada dalam urutan, yaitu setiap pernyataan kode dieksekusi satu per satu. Jadi, pada dasarnya sebuah pernyataan harus menunggu pernyataan sebelumnya untuk dieksekusi.

```
> // This function is synchronous
  function log(arg) {
      console.log(arg)
  }
  log("Sinkron")
  Sinkron
```

## Synchronous (Sinkron)

- Menghentikan eksekusi kode lebih lanjut sampai ini selesai.
- Karena ini penghentian eksekusi lebih lanjut, kode sinkron disebut 'pemblokiran'. Pemblokiran dalam arti tidak ada kode lain yang akan dieksekusi.

# ASYNCHRONOUS

## Asynchronous (Non-Sinkron)

Kode asynchronous memungkinkan program untuk dieksekusi segera di mana kode sinkron akan memblokir eksekusi lebih lanjut dari kode yang tersisa sampai menyelesaikan yang sekarang.

```
// Asynchronous Function

function angka1() {
    console.log('angka-1');
}

function angka2() {
    setTimeout(function(){
        console.log("angka-2");
    }, 2000);
}

angka2();
angka1();
```

## Asynchronous (Non-Sinkron)

- Eksekusi ini ditangguhkan ke loop acara, ini adalah konstruksi di mesin virtual JS yang menjalankan fungsi asinkron (setelah tumpukan fungsi sinkron kosong).
- Kode asinkron disebut non-blocking karena tidak memblokir kode lebih lanjut agar tidak berjalan.

## Expression Function

Sintaks lain untuk menggambarkan fungsi. Karena pada JS, fungsi adalah bentuk value yang khusus, yaitu value yang merepresentasikan aksi (action). Penulisan berikut disebut sebagai ekspresi fungsi (function expression). Kadang disebut juga sebagai fungsi anonim (anonymous function).

```
let kirimPesan = function(nama) {
  alert( "Halo semuanya, terutama kamu,", nama );
};
```

## Callback Function

Callback pada Javascript adalah sebuah fungsi yang dikirimkan sebagai parameter fungsi lainnya. Fungsi diatas adalah sebuah contoh callback yang berjalan secara Synchronous karena fungsi callback langsung dieksekusi pada sebuah proses fungsi yang memiliki sifat synchronous.

# EXPRESSION FUNCTION

```
function ask(question, yes, no) {
  if (confirm(question)) yes()
  else no();
}

function showOk() {
  alert( "You agreed." );
}

function showCancel() {
  alert( "You canceled the execution." );
}

// Melakukan Callback pada fungsi showOk dan showCancel
ask("Do you agree?", showOk, showCancel);
```

# EXPRESSION FUNCTION

## Arrow Function

Ada tiga bagian Arrow Functions atau Fungsi Lambda:

- Parameter: Setiap fungsi dapat secara opsional memiliki parameter.
- Notasi panah gemuk / notasi lambda: Ini adalah notasi untuk panah (=>).
- Pernyataan: Ini mewakili set instruksi fungsi.

```
> const tambah = (a1, a2) => a1 + a2;
  tambah(5, 5)
< 10
```

# TUGAS JAVASCRIPT

1. Buatlah Program Kalkulator Sederhana
2. Buatlah Function Penjumlahan, Pengurangan, Perkalian, dan Pembagian
3. Gunakan console.log sebagai input dan output
4. Simpan dalam file index.js
5. Upload pada akun github kalian dengan nama repository “Pertemuan-6-JS”

# Terima Kasih
# Syukron Katsiron
# Arigatou Gozaimasu