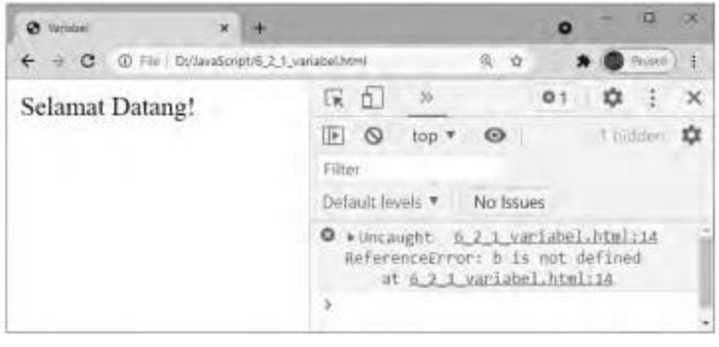
Variabel yang dideklarasikan di luar function akan menjadi variabel **global** dan variabel yang dideklarasikan di dalam function akan menjadi variabel **local**. Contohnya sebagai berikut:

File 6_2_1_variabel.html

```
<!DOCTYPE HTML>
<html>
   <head></head>
   <body>
      <script>
         var a = "Selamat Datang!";
         function selamat(){
            var b = "Selamat pagi!";
            document.write(a);
         }
         selamat();
         document.write(b);
      </script>
   </body>
</html>
```

Hasilnya seperti gambar berikut:



Penggunaan Variabel

Pada gambar di atas, variabel **a** berhasil ditampilkan walaupun dideklarasikan di luar function dan digunakan di dalam function. Ini namanya variabel **global**. Sedangkan variabel **b** tidak tampil karena

dideklarasikan di dalam function dan digunakan di luar function. Ini namanya variabel **local**. Variabel local hanya dapat digunakan di dalam function di mana variabel tersebut dideklarasikan dan tidak dapat digunakan di luar function atau di function lain.

6.2.2 Tipe Data

Setiap bahasa pemrograman pasti mengenal yang namanya tipe data. Tipe data pada JavaScript sedikit berbeda dengan bahasa pemrograman lain. Tipe data pada JavaScript bersifat seolah-olah sebagai objek. Sehingga setiap variabel pada JavaScript akan langsung memiliki method.

JavaScript mengenal 3 tipe data dasar sebagai berikut:

- **Numbers**: contoh 125, 20.50, dan sebagainya.
- **Strings**: contoh "Hallo!", "200", dan sebagainya.
- **Boolean**: hanya ada dua nilai yaitu True dan False.

Selain tiga tipe di atas, JavaScript juga mengenal tipe data **null** dan **undefined**. **Null** merupakan nilai dari sebuah variabel yang tidak diberi nilai saat didefinisikan. Sedangkan **undefined** merupakan hasil yang didapat dari proses berikut:

- Nilai dari pemanggilan variabel yang belum dideklarasikan.
- Nilai dari pemanggilan elemen array yang tidak ada.
- Nilai dari pemanggilan property objek yang tidak ada.
- Nilai dari pemanggilan function yang tidak mengembalikan nilai.
- Nilai dari parameter function yang tidak memiliki argumen.

Dapat disimpulkan bahwa perbedaan antara null dan undefined adalah bahwa null biasanya diperoleh dari kondisi normal yang sudah direncanakan, sedangkan undefined biasanya diperoleh dari kesalahan program yang tidak direncanakan.

6.2.3 Operator

Operator merupakan sebuah simbol atau kata yang digunakan untuk melakukan operasi terhadap satu atau lebih data atau variabel sehingga menghasilkan data baru. Variabel atau data yang dioperasikan disebut **operand.** Javacript mengenal beberapa jenis operator sebagai berikut:

Operator Aritmetika

Operator aritmetika merupakan operator yang digunakan untuk melakukan perhitungan data numerik. Daftar operator aritmatika dan cara penggunaanya dapat dilihat pada tabel berikut:

Operator	Keterangan
+	Untuk penjumlahan, misal A = 10 dan B = 4, maka: A + B menghasilkan 14.
-	Untuk pengurangan, misal A = 10 dan B = 4, maka: A - B menghasilkan 6.
*	Untuk perkalian, misal A = 10 dan B = 4, maka: A * B menghasilkan 40.
/	Untuk pembagian, misal A = 10 dan B = 4, maka: A / B menghasilkan 2.5.
%	Untuk mendapatkan sisa pembagian (modulus), misal A = 10 dan B = 4, maka : A % B menghasilkan 2.
++	Berarti ditambah satu, misal A = 10, maka: A++ menghasilkan 11.
--	Berarti dikurangi satu, misal A = 10, maka: A-- menghasilkan 9.

Operator Perbandingan

Operator perbandingan digunakan untuk membandingan dua variabel atau data. Hasil dari perbandingan tersebut akan menghasilkan nilai **true** atau **false.** Daftar operator perbandingan dan penjelasanya dapat dilihat pada table berikut:

Operator	Keterangan
==	Berarti sama dengan, misal A = 10 dan B = 4, maka: (A == B) menghasilkan **false.**
!=	Berarti tidak sama dengan, misal A = 10 dan B = 4, maka: (A != B) menghasilkan **true**.
>	Berarti operand pertama lebih besar dari operand ke dua, misal A = 10 dan B = 4, maka: (A > B) menghasilkan **true.**
<	Berarti operand pertama lebih kecil dari operand kedua, misal A = 10 dan B = 4, maka (A < B) menghasilkan **false.**
>=	Berarti lebih besar atau sama dengan, misal A = 10 dan B = 4, maka: (A >= B) menghasilkan **true** dan (10 >= A) juga menghasilkan **true.**
<=	Berarti lebih kecil atau sama dengan, misal A = 10 dan B = 4, maka: (A <= B) menghasilkan **false**, sedangkan (10 <= A) menghasilkan **true.**

Operator Logika

Operator logika digunakan untuk mengoperasikan data boolean. Hasil dari perbandingan tersebut akan menghasilkan nilai **true** atau **false.** Daftar operator logika dan penjelasanya dapat dilihat pada table berikut:

Operator	Keterangan
&&	Menghasilkan nilai true jika kedua operand bernilai true (logika And). Misal A = true dan B = false, maka: (A && B) menghasilkan false.
||	Menghasilkan nilai true jika salah satu atau kedua operand bernilai true (logika Or). Misal A = true dan B = false, maka: (A || B) menghasilkan true.
!	Membalikan nilai sebuah variabel (logika Not). Misal A = true, maka: (!A) menghasilkan false.

Operator Bitwise

Operator bitwise akan membandingkan nilai biner dua bilangan. Misalnya 2 akan dikonversi ke biner menjadi 10 dan 3 akan dikonversi ke biner menjadi 11. Jika 2 dan 3 dibandingkan dengan operator bitwise, maka yang dibandingkan adalah nilai biner-nya. Hasil dari operasi berupa biner yang akan dikonversi lagi dan ditampilkan dalam bentuk desimal. Daftar operator bitwise dan penjelasanya dapat dilihat pada table berikut:

Operator	Keterangan
&	Menghasilkan angka 1 jika kedua angka yang dibandingkan 1 dan menghasilkan 0 jika salah satu atau kedua angka 0 (bitwise And). Misal, A = 2 (binernya 10) dan B = 3 (binernya 11), maka: (A & B) menghasilkan 2. Maksudnya: digit pertama 1 dan 1 menghasilkan 1, digit kedua 0 dan 1 menghasilkan 0. Jika digabungkan hasilnya 10 jika didesimalkan menjadi 2.
|	Menghasilkan angka 1 jika salah satu atau kedua angka yang dibandingkan 1 dan menghasilkan 0 jika kedua angka 0 (bitwise Or). Misal, A = 2 (binernya 10) dan B = 3 (binernya 11), maka: (A | B) menghasilkan 3.
^	Menghasilkan angka 1 jika salah satu angka yang dibandingkan 1 dan menghasilkan 0 jika kedua angka 0 atau kedua angka 1 (bitwise Xor). Misal, A = 2 (binernya 10) dan B = 3 (binernya 11), maka: (A ^ B) menghasilkan 1.
~	Membalikkan suatu angka yang 1 menjadi 0 dan yang 0 menjadi 1 (bitwise Not). Biasanya hasilnya lebih besar satu angka tapi bernilai negatif. Misal, A = 3 (binernya 11) , maka: (~A) menghasilkan -4.
<<	Menggeser ke kiri bit operand pertama sejumlah angka operand kedua dan diisi dengan 0 (Left Shift). Misal A = 4 (binernya 100), maka: (A<<2) menghasilkan 16 (binernya 10000).
>>	Menggeser ke kanan bit operand pertama sejumlah angka operand kedua (Right Shift). Misal A = 9 (binernya 1001), maka: (A>>1) menghasilkan 4 (binernya 100).

>>>	Hampir sama dengan >> hanya di sebelah kiri selalu 0.

Operator Assignment

Operator ini digunakan untuk memberikan nilai kepada suatu variabel menggunakan tanda sama dengan. Daftar operator assignment dan penjelasanya dapat dilihat pada tabel berikut:

Operator	Keterangan
=	Memberi nilai pada suatu variabel dengan nilai, variabel lain atau hasil perhitungan di sebelah kanan. Misal A = 3 dan B = 4, maka: C = A + B menghasilkan nilai C = 3 + 4 yaitu 7.
+=	Menambah nilai suatu variabel dengan nilai atau variabel lain. Misal A = 3 dan B = 4, maka: A += B artinya nilai A ditambah nilai B, sehingga nilai A menjadi 7.
-=	Mengurangi nilai suatu variabel dengan nilai atau variabel lain. Misal A = 3 dan B = 4, maka: B -= A artinya nilai B dikurangi nilai A, sehingga nilai B menjadi 1.
*=	Mengalikan nilai suatu variabel dengan nilai atau variabel lain. Misal A = 3 dan B = 4, maka: B *= A artinya nilai B dikalikan nilai A, sehingga nilai B menjadi 12.
/=	Membagikan nilai suatu variabel dengan nilai atau variabel lain. Misal A = 12 dan B = 4, maka: A /= B artinya nilai A dibagi nilai B, sehingga nilai A menjadi 3.
%=	Mengambil sisa bagi suatu variabel dengan nilai atau variabel lain. Misal A = 12 dan B = 4, maka: A %= B artinya nilai sisa bagi A dengan B, sehingga nilai A menjadi 0.

Operator Kondisi

Operator ini digunakan untuk memberikan nilai pada suatu variabel sesuai kondisi yang ditentukan. Jika kondisi **true** maka akan diberi nilai dengan nilai di sebelah kiri tanda titik dua, sedangkan jika kondisi **false** maka akan diberi nilai dengan nilai di sebelah kanan tanda titik dua.

Kondisi diletakkan di dalam kurung diikuti dengan tanda **?** yang memisahkan dengan nilai.

Operator	Keterangan
? ... : ...	Tanda tanya memisahkan kondisi dengan nilai, sedangkan tanda titik dua memisahkan nilai pertama dan nilai kedua. Nilai pertama dipakai jika kondisi true dan nilai kedua dipakai jika kondisi false.

Berikut contoh beberapa penggunaan operator:

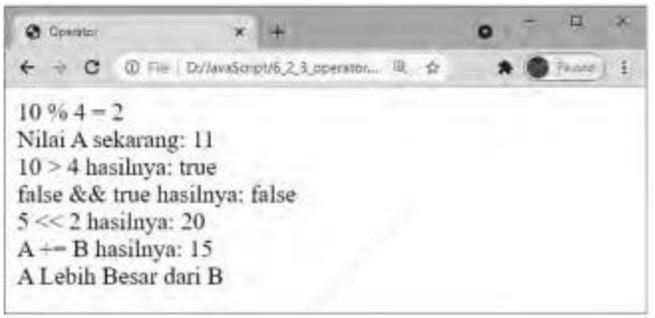
File 6_2_3_operator.html

```
<!DOCTYPE HTML>
<html>
   <head></head>
   <body>
      <script>
         var A = 10, B = 4;
         var C = A % B;
         document.write("10 % 4 = " + C + '<br>');

         A++;
         document.write('Nilai A sekarang: ' + A + '<br>');

         var D = A > B;
         document.write('10 > 4 hasilnya: ' + D + '<br>');

         var E = false, F = true;
         var G = E && F;
         document.write('false && true hasilnya: ' + G +
'<br>');

         var H = 5, I = 2;
         var J = 5 << 2;
         document.write('5 << 2 hasilnya: ' + H + '<br>');

         A += B;
         document.write('A += B hasilnya: ' + A + '<br>');
```



```
         var hasil = (A>B) ? "A Lebih Besar dari B" : "A Lebih
Kecil dari B";
         document.write(hasil);
      </script>
   </body>
</html>
```

Pada skrip tersebut, masing-masing jenis operator diwakili oleh satu operator. Untuk mencoba operator lain pada masing-masing jenis operator silakan bereksperimen sendiri. Hasil skrip tersebut seperti gambar berikut:



Hasil Penggunaan Operator

6.2.4 Statement Control

Statement control pada bahasa pemrograman digunakan untuk menjalankan suatu program sesuai dengan kondisi tertentu. Ada berbagai jenis statement control pada JavaScript yang dapat digunakan sesuai kebutuhan.

Percabangan dengan if

Percabangan ini digunakan untuk menjalankan suatu program jika kondisi terpenuhi (**true**) dan jika kondisi tidak terpenuhi tidak ada alternatif lain untuk dijalankan. Penulisan kondisi biasanya menggunakan variabel dengan tipe **boolean** atau menggunakan operator **logika** yang akan menghasilkan tipe data boolean.

Penggunaan **if** dapat ditulis dengan format sebagai berikut:

```
if(kondisi) pernyataan;
```

Jika pernyataan lebih dari satu baris, cara di atas tidak dapat digunakan, tapi menggunakan format berikut:

```
if(kondisi){
   pernyataan baris pertama;
   pernyataan baris kedua;
}
```

Berikut ini contoh percabangan menggunakan if pada JavaScript:

File 6_2_4_percabangan_if.html

```
<!DOCTYPE HTML>
<html>
   <head>
      <script>
         var umur = 20;
         var dewasa;

         if(umur > 17) dewasa = true;
         if(umur > 20) document.write("Anda sudah boleh
menikah");

         if(dewasa){
            document.write("<br>Anda sudah bisa buat KTP");
            document.write("<br>Anda sudah bisa buat SIM");
         }
      </script>
   </head>
   <body>
   </body>
</html>
```


Hasil dari skrip tersebut seperti gambar berikut:

Hasil Percabangan dengan if

Percabangan dengan if ... else ...

Percabangan ini digunakan untuk menjalankan program jika kondisi terpenuhi, dan jika kondisi tidak terpenuhi ada alternatif untuk dijalankan. Cara menggunakanya yaitu dengan format sebagai berikut:

```
if(kondisi) pernyataan jika terpenuhi;
else pernyataan jika tidak terpenuhi;
```

Jika pernyataan lebih dari satu baris, maka formatnya menjadi seperti berikut:

```
if(kondisi){
  pernyataan jika terpenuhi;
}else{
  pernyataan jika tidak terpenuhi;
}
```

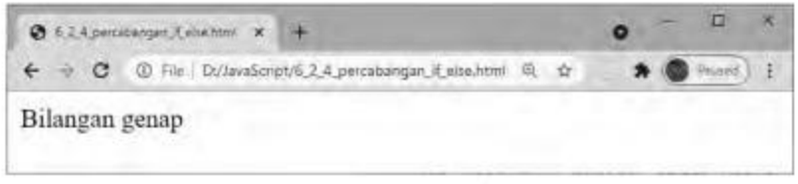
Berikut ini contoh penggunaan percabangan dengan if .... Else .... pada JavaScript:

File 6_2_4_percabangan_if_else.html

```
<!DOCTYPE HTML>
<html>
   <head>
```

```
      <script>
         var angka = 20;
         if(angka%2 == 0) document.write("Bilangan genap");
         else document.write("Bilangan ganjil");
      </script>
   </head>
   <body>
   </body>
</html>
```

Silakan ubah nilai variabel **angka** untuk menampilkan hasil yang berbeda. Hasil dari perintah di atas seperti gambar berikut:



Hasil Percabangan dengan if...else...

Percabangan dengan if ... else if ... else ...

Percabangan ini digunakan untuk menjalankan program jika kondisi terpenuhi, dan jika kondisi tidak terpenuhi ada pengecekan kondisi berikutnya hingga tidak ada lagi kondisi untuk dicek baru dijalankan alternatif terakhir. Cara menggunakanya dengan format sebagai berikut:

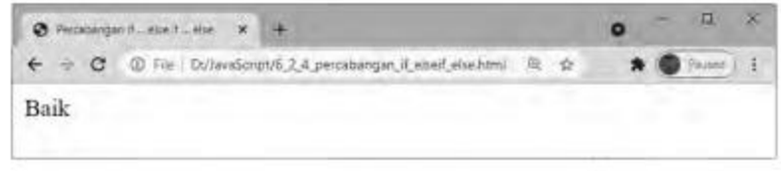
```
if(kondisi){
   pernyataan jika kondisi terpenuhi;
}else if(kondisi kedua){
   pernyataan jika kondisi kedua terpenuhi;
}else if(kondisi ketiga){
   pernyataan jika kondisi ketiga terpenuhi;
}else{
   pernyataan jika tidak ada kondisi terpenuhi;
}
```


Berikut ini contoh penggunaanya:

File 6_2_4_percabangan_if_elseif_else.html

```
<!DOCTYPE HTML>
<html>
   <head>
      <title>Percabangan if ... else if ... else</title>
      <script>
         var nilai = 80;
         if(nilai > 90) document.write("Sangat Baik");
         else if(nilai > 75) document.write("Baik");
         else if(nilai > 60) document.write("Cukup");
         else document.write("Kurang");
      </script>
   </head>
   <body>
   </body>
</html>
```

Silakan ubah nilai variabel **nilai** untuk menampilkan hasil yang berbeda. Hasilnya seperti gambar berikut:



Hasil Percabangan dengan if...else if...else...

Percabangan dengan switch

Percabangan ini merupakan alternatif untuk menggantikan percabangan dengan if ... else if ... else. Namun, jika variabel yang dicek sama, cara ini lebih disarankan daripada menggunakan if ... else if ... else. Cara penggunaanya sebagai berikut:

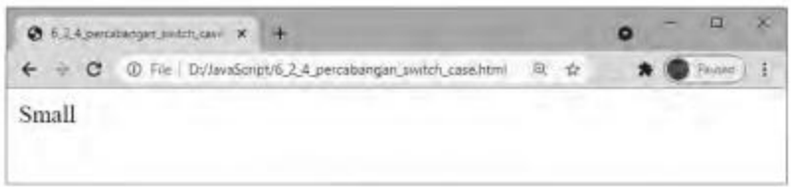
```
switch (variabel){
```

```
    case kondisi 1: pernyataan;
         break;
    case kondisi 2: pernyataan;
         break;
    case kondisi 3: pernyataan;
         break;
    default: pernyataan default;
}
```

Contoh penggunaanya, perhatikan pada skrip berikut:

```
<!DOCTYPE HTML>
<html>
   <head>
      <script>
         var ukuran = "S";
         switch(ukuran){
            case "L": document.write("Large");
               break;
            case "M": document.write("Medium");
               break;
            default: document.write("Small");
         }
      </script>
   </head>
   <body>
   </body>
</html>
```

Ubah nilai variabel **ukuran** dengan **L** atau **M** untuk menampilkan hasil yang berbeda. Hasilnya seperti gambar berikut:



Hasil Percabangan dengan Switch

Perulangan dengan while

Perulangan ini digunakan untuk menjalankan program berulang-ulang selama kondisi yang ditentukan terpenuhi. Cara penggunaanya sebagai berikut:

```
while(kondisi){
   pernyataan;
}
```

Berikut contoh penggunaan perulangan dengan while:

6_2_4_perulangan_while.html

```
<!DOCTYPE HTML>
<html>
   <head>
      <script>
         var hitung = 0;
         while(hitung < 10){
            document.write("Hitungan ke: " + hitung + "<br>");
            hitung++;
         }
         document.write("Hitungan terakhir: " + hitung +
"<br>");
      </script>
   </head>
   <body>
   </body>
</html>
```

Skrip tersebut akan menghasilkan seperti gambar berikut:

Hasil Perulangan dengan While

Perulangan dengan do ... while

Statement control ini digunakan untuk menjalankan program secara berulang-ulang selama kondisi yang ditentukan di akhir terpenuhi. Hampir sama dengan while, hanya kondisi ditulis di akhir. Cara penggunaanya sebagai berikut:

```
do{
  Pernyataan;
} while(kondisi);
```

Contoh penggunaanya dapat dilihat pada skrip berikut:

File 6_2_4_perulangan_do_while.html

```
<!DOCTYPE HTML>
<html>
   <head>
      <script>
         var hitung = 0;
         do{
            document.write("Hitungan ke: " + hitung + "<br>");
            hitung++;
         } while (hitung<10);
         document.write("Hitungan mencapai " + hitung + "<br>");
      </script>
   </head>
   <body>
   </body>
</html>
```


Hasilnya sama dengan perulangan menggunakan **while.**

Perulangan dengan for

Perulangan ini digunakan untuk menjalankan program berulang-ulang dengan menentukan nilai awal, nilai akhir, dan penambahan/pengurangan nilai. Cara penggunaanya sebagai berikut:

```
for(inisialisasi counter; kondisi; increment/decrement counter){
   pernyataan;
}
```

Contoh penggunaanya seperti skrip berikut:

File 6_2_4_perulangan_for.html

```
<!DOCTYPE HTML>
<html>
   <head>
      <script>
         var hitung;
         for(hitung=0; hitung<10; hitung++){
            document.write("Hitungan ke: " + hitung + "<br>");
         }
         document.write("Hitungan mencapai " + hitung + "<br>");
      </script>
   </head>
   <body>
   </body>
</html>
```

Hasilnya sama dengan perulangan menggunakan **while.**

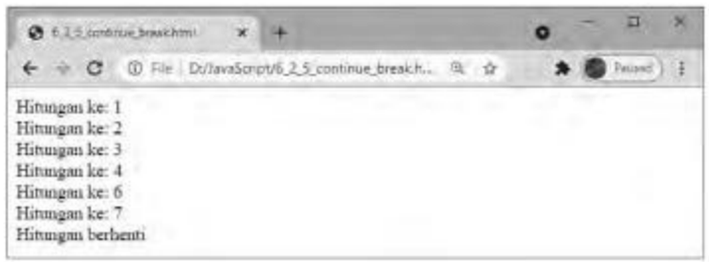
6.2.5 Continue dan Break

Continue digunakan untuk melewati suatu perulangan karena kondisi tertentu. Sedangkan **break** digunakan untuk menghentikan paksa suatu perulangan sebelum kondisi perulangan berakhir karena kondisi tertentu. Contoh penggunaanya dapat diperhatikan pada skrip berikut:

File 6_2_5_continue_break.html

```
<!DOCTYPE HTML>
<html>
   <head>
      <script>
         var hitung = 0;
         while(hitung<10){
            hitung++;
            if(hitung == 5) continue; //dilewati
            if(hitung == 8) break; //berhenti
            document.write("Hitungan ke: " + hitung + "<br>");
         }
         document.write("Hitungan berhenti");
      </script>
   </head>
   <body>
   </body>
</html>
```

Hasil dari skrip tersebut seperti gambar berikut:



Contoh Penggunaan Continue dan Break

Pada gambar tersebut tampak tidak ada hitungan ke-5 dan pada hitungan ke-8 perulangan dihentikan.

6.3 Function

Function merupakan blok program yang dapat digunakan secara berulang-ulang dengan memanggil nama function. Function dapat mengembalikan sebuah nilai dengan perintah **return** atau tanpa mengembalikan nilai. Pemanggilan function yang mengembalikan nilai dapat digunakan sebagai nilai suatu variabel. Cara menulis function sebagai berikut:

```
function nama_fungsi(parameter1, parameter 2){
   pernyataan;
   return variabel;
}
```

Berikut ini contoh dari penggunaan function.

File 6_3_function.html

```
<!DOCTYPE HTML>
<html>
   <head>
      <script>
         function hitung_luas(p, l){
            var L = p * l;
            return L;
         }
         function tampilkan_hasil(){
            var teks = "Luas tanah " + hitung_luas(200, 150) +
"m2";
            document.write(teks);
         }
         tampilkan_hasil();
      </script>
   </head>
```