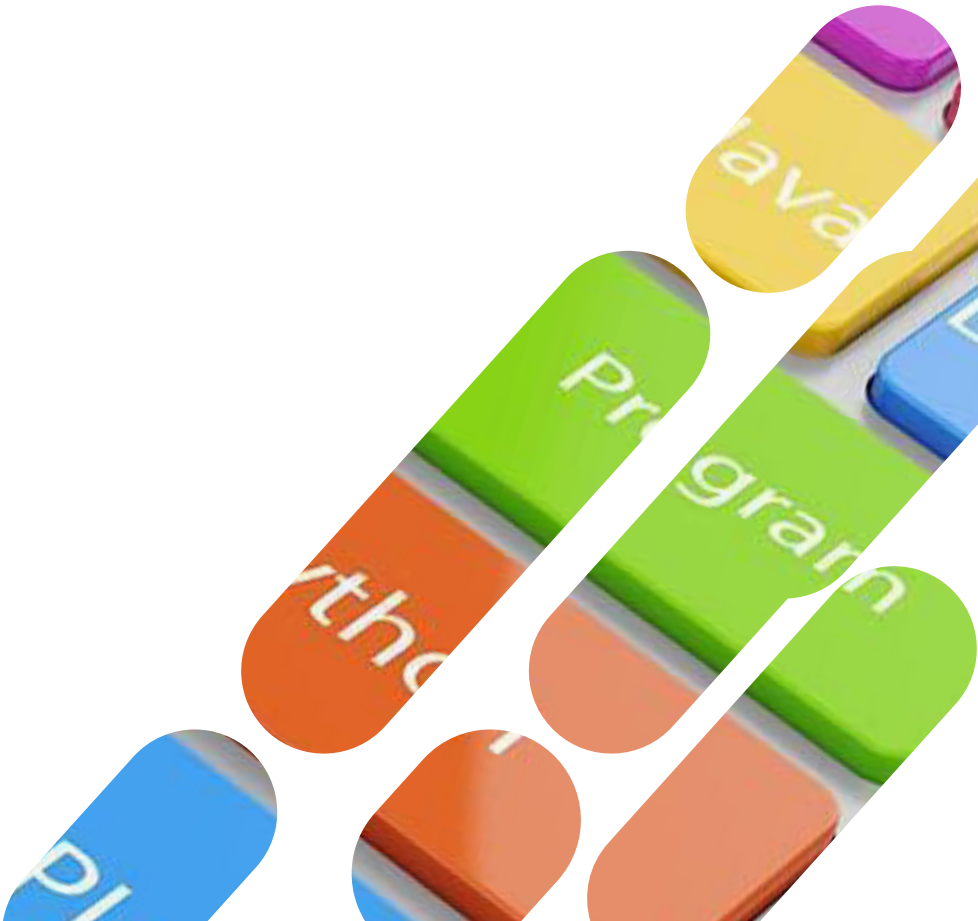
SMK Telkom Makassar

PEMROGRAMAN TERSTRUKTUR
VARIABEL DAN TIPE DATA



PEMROGRAMAN TERSTRUKTUR

VARIABLE DAN TIPE DATA

Rekaya Perangkat Lunak SMK Telkom Makas*sar*

Tujuan Pembelajaran

1. Peserta didik dapat menginstalasi javascript runtime (node.js)
2. Perserta didik dapat mengkonfigurasi javascript runtime (node.js)
3. Peserta didik dapat Memahami struktur kode javascript
4. Peserta didik dapat Memahami variabel dan tipe data

First Code

Sekarang Anda sudah siap untuk menjalankan JavaScript! Namun sebelum Anda dapat menjalankan kode apa pun, Anda harus menulisnya.

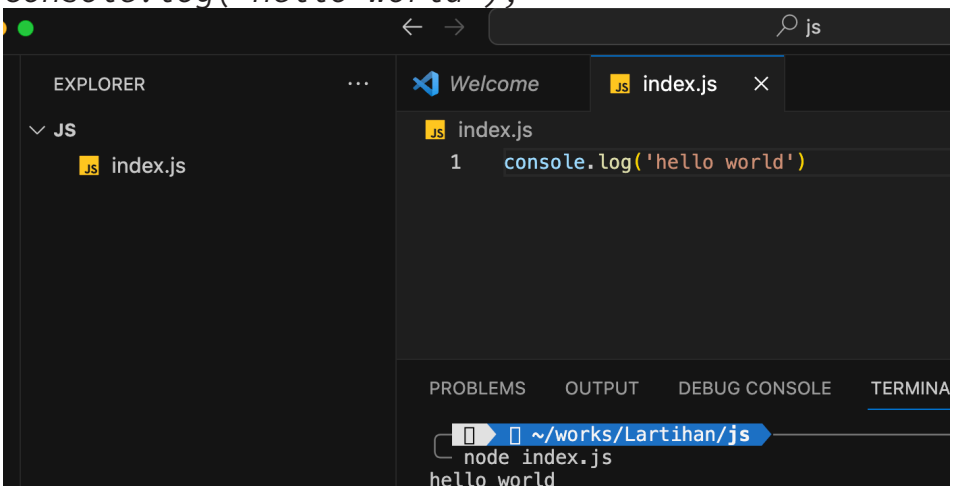
CONSOLE.LOG

Kita akan terus menggunakan sesuatu yang disebut console.log. console.log mengambil apa yang dibungkus dalam tanda kurung dan mencetaknya ke dalam baris perintah. Pada kasus ini, kita tidak perlu memahami mekanisme yang mendasari di balik console.log. kita hanya untuk mengonfirmasi bahwa komputer kita sudah siap untuk menulis JavaScript!.

First Code

Kita akan membuatnya sederhana untuk saat ini, dengan satu console.log. Ketik ini ke dalam file Anda di VS Code, tapi jangan tekan Enter. Simpan file sebagai file .js setelah Anda selesai.

```
console.log('Hello world');
```



Selamat! Anda telah secara resmi menjalankan kode pertama Anda.

Bugs

Menulis algoritma sama seperti menulis esai persuasif. Algoritma adalah sebuah tulisan yang memiliki alur logika yang mengarah pada kesimpulan yang benar. Anda akan dinilai berdasarkan kriteria berikut:

1. Terjawab: Esai Anda sampai pada kesimpulan yang benar.
2. Tata bahasa: Tidak ada aturan bahasa tertulis yang dilanggar.
3. Pembenaran yang baik: Alasan Anda di seluruh esai masuk akal.

Menulis kode memiliki persyaratan yang sama:

1. Terjawab: Anda memiliki perilaku yang diharapkan.
2. Tata bahasa: Sintaks JavaScript yang benar.
3. Pembenaran yang baik: Urutan logika algoritme Anda tahan banting.

"Merasa kelelahan? Beristirahatlah selama 15 menit. Anda akan kembali dengan pikiran yang lebih jernih dan menemukan bug lebih cepat dari yang Anda kira."

Pemrogram profesional membuat program dengan jumlah kode yang luar biasa banyak. Kode-kode tersebut disebut sebagai basis kode. Bug suka bersembunyi di celah-celah ribuan (bahkan jutaan) baris kode. Ini adalah pekerjaan yang sulit, tetapi para programmer berusaha keras untuk menulis kode yang bebas dari bug.

BUG BISA MEMBUAT FRUSTRASI

Ketika Anda mulai membuat kode, Anda akan menulis banyak kode yang tidak sempurna. Anda akan salah untuk pertama kalinya, kedua kalinya, bahkan mungkin kedua puluh kalinya. Tetapi bagian terpenting dari bug adalah tidak membiarkannya masuk ke kepala Anda. Bahkan programmer profesional pun bisa frustrasi. Itu semua adalah bagian dari proses pembelajaran, jadi jangan menyerah!

Struktur Kode

Bahasa pemrograman memiliki aturan dasar tentang bagaimana bahasa tersebut harus ditulis. Mari kita bahas sebelum Anda mengotori tangan Anda. Ini hanyalah gambaran umum tingkat tinggi untuk memahami bagaimana kode JavaScript disusun. Tidak perlu memikirkan isi dari baris-barisnya.

STATEMENTS/PERNYATAAN

Pernyataan adalah ekspresi yang melakukan suatu tindakan. Pernyataan biasanya ditulis pada satu baris.

```
console.log('I am a statement');
var statement = 'I am anotherstatement';
```

SEMICOLONS

Pernahkah Anda memperhatikan titik koma? Setiap pernyataan harus memiliki titik koma di bagian akhir. JavaScript memiliki beberapa cara licik untuk memperbaiki dirinya sendiri jika Anda lupa, tetapi JavaScript sangat rewel dan rentan terhadap kesalahan. Bermainlah dengan aman dan biasakan untuk selalu menambahkan titik koma sendiri. Ada beberapa situasi di mana titik koma di akhir pernyataan merupakan sintaks yang ilegal. Saya akan menjelaskan situasi tersebut sambil berjalan.

COMMENTS

Ada kalanya seorang programmer ingin menulis teks yang tidak boleh dijalankan, seperti catatan untuk diri sendiri. Komentar adalah cara untuk menulis informasi dalam file yang akan diabaikan ketika kode dijalankan.
Anda dapat menulis apa pun yang Anda inginkan, untuk alasan apa pun. Algoritma yang kompleks bisa menjadi rumit, dan komentar adalah salah satu cara untuk membantu menjelaskan apa yang dilakukan kode. Sintaksnya adalah menambahkan garis miring ganda di depan teks yang bukan kode.

```
// aku adalah komentar
console.log('I am a statement');
```

INDENTATIONS
Anda akan melihat lekukan pada sebagian garis. Hal ini memang disengaja, tetapi secara teknis bersifat opsional. Pemrogram membuat indentasi kode agar mudah dibaca. Setiap editor teks pengkodean akan memandu Anda menuju lekukan yang tepat. Ketika Anda menulis lebih banyak kode, Anda akan mulai memahami aturannya.

Variabel

Bagi manusia, konsep nama adalah sebuah sistem untuk menciptakan identitas yang unik. Dengan cara yang sama seperti Anda dan saya memiliki nama yang berbeda untuk melacak satu sama lain, nama dalam pemrograman digunakan untuk mengidentifikasi dan melacak data. Nama pengkodean disebut sebagai variabel.

DECLARATION
Mengasosiasikan data dengan sebuah variabel adalah proses dua langkah. Langkah pertama adalah mendeklarasikan variabel. Proses deklarasi memberi tahu JavaScript bahwa nama tertentu harus disiapkan sebagai variabel. Kata khusus var memberi tahu JavaScript bahwa sebuah variabel harus dideklarasikan.

```
// declaration
var aHumanName;
```

ASSIGNMENT
Asosiasi data dengan variabel adalah penugasan. Setelah variabel dideklarasikan, variabel tersebut tidak perlu dideklarasikan ulang. Penugasan dieksekusi dengan operator tanda sama dengan.

```
// declaration
var aHumanName;
// assignment
aHumanName = ‘Andrew’;
```

PENUGASAN DAN PERNYATAAN DALAM SATU PERNYATAAN
Deklarasi dan penugasan adalah dua ide yang berbeda, tetapi keduanya sering dilakukan dalam satu pernyataan.

```
// declaration and assignment in one statement
var aHumanName = ‘Andrew’;
```

DATA REASSIGNMENT
Variabel juga dapat ditetapkan kembali ke nilai yang berbeda. Kemampuan untuk menetapkan ulang adalah penting untuk topik-topik yang akan datang.

```
// declaration and assignment
var aHumanName = ‘Andrew’;
// assign variable to a new value
aHumanName = ‘Charlie’;
```

Penyorotan sintaks dan rekomendasi kesalahan VS Code biasanya akan melindungi Anda dari penggunaan kata kunci sebagai variabel secara tidak sengaja. Namun terkadang kesalahan tersebut tidak langsung terlihat. Satu-satunya hal yang dijamin adalah ada sesuatu dalam kode Anda yang salah

MENGEVALUASI PERNYATAAN DENGAN VARIABEL

Setiap kali sebuah variabel digunakan, JavaScript akan mengekstrak data yang terkait dengan variabel tersebut dan menggunakannya sebagai gantinya.

```
var myBirthYear = 2000;
var currentYear = 2023;
// setara dengan 2023 - 2000
currentYear - myBirthYear;
```

KATA KUNCI TIDAK BOLEH BERUPA VARIABEL

Kata var adalah salah satu contoh kata kunci dalam JavaScript. Penggunaannya selalu berarti "deklarasi variabel". Kata ini akan mengabaikan maksud lain dan tidak boleh digunakan untuk nama variabel.

```
// will create runtime error! variable declaration
expected
var var = 2023;
```

Kode yang baru saja Anda saksikan di sini adalah permintaan untuk menyatakan tidak ada apa-apa dan tidak menetapkan apa-apa pada angka 2021. Bingung? Begitu juga dengan JavaScript.

KONVENSI NAMA VARIABEL

Programmer suka menjaga struktur variabel tetap konsisten. Hal ini biasa disebut sebagai konvensi. Konvensi JavaScript untuk nama variabel adalah camelCase: Huruf kecil untuk kata pertama, dengan semua kata berikutnya menggunakan huruf besar

```
// Konvensi camelCase yang diterima
var currentYear = 2023;
// tidak akan rusak saat runtime tetapi akan akan
lebih baik jika Anda tidak melakukannya
var current_year = 2023;
// kita bukan lagi teman
var xXx_CuRrEnTyEaR_xXx = 2023;
```

Nantinya, kita akan menggunakan konvensi yang berbeda untuk topik-topik tertentu. Untuk saat ini, selalu gunakan camelCase.

Tipe Data

Setelah Anda mengetahui cara kerja variabel, sekarang saatnya mempelajari jenis data yang dapat diberikan kepada variabel.

NUMBER

Angka mengikuti aturan aritmatika dasar. Urutan operasi adalah: tanda kurung, eksponen, perkalian, pembagian, penambahan, lalu pengurangan.

```
10 + 20; //30
20 * 10; // 200
(20 + 3) * 10; // 230
```

INCREMENT OPERATOR

Dalam pemrograman, menambah atau mengurangi nilai variabel dengan 1 adalah operasi yang umum, jadi ada singkatan untuk itu. Gunakan operator double-plus dan double-minus untuk menambah dan mengurangi.

```
var num = 1;
num++;
// num is now 2
num--;
// num is now back to 1
```

Jangan Pernah Menetapkan Kenaikan Pernyataan

Berhati-hatilah untuk tidak menetapkan saat melakukan inkrementasi, karena nilai variabel tidak akan berubah jika Anda melakukannya. Perilaku yang tidak diharapkan akan terjadi.

```
var num = 1;
num++; // good!
num = num++; // no! bad!
```

SINGKATAN MATEMATIKA YANG DAPAT DIOPERASIKAN SENDIRI

Melakukan perhitungan pada sebuah variabel dan kemudian menetapkan kembali nilai sebelumnya adalah hal yang umum dilakukan. Para programmer memiliki singkatan untuk itu. Istilah umum untuk operator ini adalah penugasan aritmatika. Ini adalah operator matematika dan operator penugasan yang digabungkan menjadi satu operator.

```
var num = 1;
num += 2; //sameasnum=num+2
num; // 3
var num = 1;
num *= 3; // bekerja dengan cara yang sama untuk
perkalian dan pembagian
num; // 3
```

NAN: BUKAN ANGKA

Angka tampak mudah, bukan? Nah, sekarang saatnya membicarakan sisi gelap dari angka. Ada jenis angka khusus yang bukan angka. Ya, Anda membacanya dengan benar. Ini adalah angka yang mewakili sesuatu yang bukan angka. Namanya adalah NaN, yang merupakan singkatan dari “Not a Number.”

NaN adalah kata kunci yang merupakan nilai numerik. Matematika dapat dilakukan padanya. Matematika apa pun yang dilakukan pada NaN akan dihitung menjadi NaN.

```
NaN * 10; // NaN
```

Strings

Kumpulan karakter disebut sebagai string. Anggap saja sebagai "untaian karakter". Membuat string itu sederhana. Anda hanya perlu membungkus teks dalam tanda kutip. Tanda kutip ganda dan tunggal berfungsi untuk mengembangkan string.

```
// both valid strings
“saya string”;
‘saya string’;
```

PENGGABUNGAN STRING
JavaScript memiliki banyak trik pintar dalam memodifikasi string. Namun untuk saat ini, satu-satunya hal yang perlu Anda ketahui adalah bahwa string dapat digabungkan dengan operator penjumlahan. Konsep menggabungkan string disebut penggabungan, atau disingkat concat. Sejumlah string dapat digabungkan dengan menggunakan operator plus.

```
‘one ‘ + ‘two’; // ‘one two’
‘one ‘ + ‘two’ + ‘three’ + ‘four’; //
‘one two three four
```

Perhatikan ruang kosong di akhir semua string, kecuali yang terakhir? Itu karena spasi juga merupakan karakter. Spasi diperlukan untuk memisahkan kata saat melakukan penggabungan.'

SINGKATAN PENGGABUNGAN STRING YANG BEROPERASI SENDIRI
Dengan cara yang sama seperti matematika yang biasanya dihitung dan ditugaskan kembali ke variabel yang sama, string juga digabungkan dan ditugaskan kembali. Secara teknis, string juga disebut penugasan penjumlahan. Operator penugasan penjumlahan dapat digunakan untuk keduanya.

```
var numWords = ‘one two’; // ‘one two’
numWords += ‘three four’; numWords; // ‘one two three
four’
```

ESCAPE CHARACTERS
Membungkus teks dalam tanda kutip menunjukkan sebuah string. Tetapi bagaimana jika Anda menginginkan kutipan di dalam string?

```
‘hello, my cat’s name is Suzy’; // runtime error
```

JavaScript membacanya sebagai sebuah string yang berhenti di cat. Tanda kutip pada "cat's" membuangnya. Teks selanjutnya ditafsirkan sebagai kode yang tidak masuk akal dan akan rusak pada saat runtime. Untuk menyatakan bahwa apostrof pada "cat's" harus menjadi bagian dari string, kita harus menghilangkannya. Hal ini dilakukan dengan menggunakan operator backslash.

```
'hello, my cat\'s name is Suzy'; // much better
```

ANGKA DAN STRING: PERBEDAANNYA

Sebuah string dianggap sebagai teks. Dalam hal ini, kita dapat memiliki string yang terlihat seperti angka, tetapi sebenarnya bukan. Angka sepuluh (10) dan string yang terdiri dari "1" dan "0" ("10") ditafsirkan secara berbeda. Angka dapat dioperasikan dengan matematika. Tetapi konsep operasi matematika pada teks tidak masuk akal di dunia nyata, bukan?

```
10; // the number ten
'10'; // text with two characters: 1 and 0
```

Boolean

Sederhananya, boolean adalah tipe data yang memiliki kata kunci benar atau salah. Kata kunci masing-masing berarti persis seperti yang Anda pikirkan.

```
// boolean keywords
true;
false;
```

Pengecekan data adalah cara yang paling umum untuk mendapatkan nilai boolean. Kita akan melakukan banyak pengecekan data ketika kita belajar tentang kondisional. Untuk saat ini, cukup ketahui bahwa kata kunci benar dan salah itu ada.

null

Kadang-kadang Anda ingin mengekspresikan gagasan "tidak ada". Di sinilah kata kunci null berperan. Secara harfiah berarti "tidak ada".

```
// keyword null
null;
```

undefined

Kata kunci lain yang secara longgar tidak mewakili apa pun adalah tidak terdefinisi. Lebih khusus lagi, kata kunci ini mewakili "sesuatu yang belum diberi nilai."

```
// keyword undefined
undefined;
```

Sejauh ini, satu-satunya cara yang telah kita bahas untuk membuat nilai tidak terdefinisi adalah dengan mendeklarasikan sebuah variabel tetapi tidak memberikan nilai.

```
var aName; // undefined
```

Tipe Paksaan

Pembaca yang budiman, maafkan saya, tapi saya tidak bisa hidup dengan kebohongan ini lagi. Sebenarnya, Anda dapat melakukan matematika dengan campuran angka dan string-kadang-kadang.

```
// angka 30 dikurangi string ‘20’ sama dengan angka
10
30 - ‘20’; // 10
```

Setelah JavaScript melihat operasi yang tidak masuk akal, seperti matematika dengan angka dan string, terkadang JavaScript akan mencoba melakukan sesuatu yang disebut pemaksaan tipe. Ini adalah konsep melakukan operasi pada dua tipe data yang berbeda secara fundamental yang menghasilkan output yang sebenarnya.
Sebagian orang mungkin menganggap ini sebagai pendapat yang kuat, tetapi menurut saya, pemaksaan jenis merupakan perwujudan dari pepatah “Hanya karena Anda bisa, bukan berarti Anda harus melakukannya.”
Banyak aturan di balik pemaksaan tipe, terus terang saja, benar-benar konyol. Mari kita lihat contoh dari apa yang saya maksud. Di bawah ini adalah daftar contoh yang tidak komprehensif yang akan melelehkan otak Anda jika Anda mencoba menulis ribuan baris kode yang sangat bergantung pada pemaksaan jenis.

Struktur Data

Dalam skenario dunia nyata, programmer beroperasi pada sejumlah besar data. Banyak program modern yang menangani ribuan, bahkan jutaan, potongan data.
Karena itu, sangat masuk akal bagi para programmer untuk membuat strategi manajemen data. Daripada menetapkan setiap bagian data ke variabel, satu per satu, mengapa tidak membuat sesuatu yang merupakan kumpulan dari banyak bagian data? Inilah yang dimaksud dengan struktur data.
JavaScript menyediakan dua struktur data dasar untuk mengelola data. Keduanya disebut array dan objek.

Objects

Objek adalah struktur data. Objek menyimpan data mereka dalam pasangan kunci-nilai. Kunci juga disebut sebagai properti objek.

```
var catBreedAndName = { terrier: ‘suzy’, // key:
terrier,value: ‘suzy’
shiba: ‘yin-yin’, // key: shiba, value: ‘yin-yin’
goldenRetriever: ‘sparky’ // key: goldenRetriever,
value: ‘sparky’
};
```

Pengaksesan nilai dilakukan dengan merujuk pada nama kunci yang terkait. Hal ini dapat dilakukan dengan operator titik (titik).

```
// mendapatkan nilai ‘suzy’ dengan mengakses kunci
‘terrier’
catBreedAndName.terrier; // ‘suzy’
```

Notasi kurung juga dapat digunakan. Ini mengakses nilai menggunakan nilai string dari kunci.

```
// mendapatkan nilai ‘suzy’ dengan mengakses kunci
‘terrier’
catBreedAndName[‘terrier’]; // ‘suzy’
```

OBJECTS AND UNDEFINED
Setiap upaya untuk mengakses kunci yang tidak ada akan menghasilkan output yang tidak terdefinisi.

```
var cats = { terrier: ‘suzy’
};
// No key ‘shiba’ exists in the object cats.shiba;
// undefined.
```

Arrays

Array adalah struktur data dengan sintaks berupa tanda kurung siku, dengan koma yang memisahkan setiap bagian data dari bagian lainnya.

```
[1, 2, 3]; // an array of numbers
```

Larik memiliki indeks. Indeks adalah “slot” yang dapat memasukkan data ke dalamnya. Setiap item di dalam larik dapat diakses dengan indeks menggunakan notasi kurung. Indeks dalam larik dimulai dari angka 0.

```
var array = [30, 21, 43, 34]; array[0]; // 30
array[1]; // 21
array[2]; // 43
array[3]; // 34
```

ARRAYS AND UNDEFINED
Upaya untuk mengakses indeks yang berada di luar batas ukuran larik akan menyebabkan masalah pada kode Anda. Keluaran Anda tidak akan terdefinisi.

```
var array = [30, 21, 43, 34];
// array is not long enough to have an index of 4
array[4]; // undefined
```

ARRAY PROPERTY: LENGTH
Lebih lanjut akan dijelaskan nanti dalam buku ini, tetapi array juga memiliki konsep properti. Yang paling penting yang akan digunakan adalah properti panjang.

```
var array = [30, 21, 43, 34]; array.length; // 4
```

In Action

```
npm i readline-sync
```

Dalam JavaScript, membuat program dasar dengan interaktivitas pengguna ternyata sangat rumit. Dalam banyak latihan di dalam buku ini, Anda akan menggunakan paket praktis yang telah dibuat dengan susah payah oleh orang lain, yaitu readline-sync.

Untuk tantangan pertama Anda, mari kita tinjau kembali proses pembuatan file dan menjalankan kode. Buat sebuah file dan beri nama catNames.js. Berikut adalah tampilan VS Code explorer saya saat ini. Tambahkan potongan JavaScript pertama Anda untuk mempersiapkan latihan.

```
// the content of our file
var prompt=require('readline-sync');
var pertanyaan = prompt.question('apa jenis kucing-
mu?');
```

Jangan terlalu memusingkan detailnya. Ketahuilah bahwa dua baris ini membuat JavaScript dapat berinteraksi. Menjalankan kode ini akan membuat sebuah prompt teks. Setelah memasukkan teks, teks tersebut akan disimpan ke dalam variabel jenis.
Mengambil langkah demi langkah akan membantu Anda menemukan bug lebih cepat. Sebelum mencoba memecahkan masalah, cetaklah variabel input breed.
File tersebut untuk sementara akan terlihat seperti ini. Ingatlah untuk menyimpan file tersebut sebelum menjalankannya.

```
// the content of our file
var prompt=require('readline-sync');
var pertanyaan = prompt.question('apa jenis kuc-
ingmu?');
console.log(pertanyaan)
```

Dengan pengaturan yang cepat, buatlah struktur data dengan informasi kucing yang indah. Setiap pasangan nilai kunci dapat ditambahkan. Sebagai contoh, saya akan memberikan nama-nama yang sangat bagus yang akan membuat kucing mana pun senang dipanggil.

```
var prompt=require('readline-sync');
var pertanyaan = prompt.question('apa jenis kuc-
ingmu?');
var kucing={
    'anggora':'susi',
    'persia':'boy',
    'burma':'meong',
};
console.log('kucingmu cocoknya bernama '+kuc-
ing[pertanyaan]+' terima kasih');
```