```
// LECTURE 17: Decision Making in JS
// Write a program that outputs if a person is able to
// get a driving license
// or the number of years he/she will be required to wait
// to obtain a driving license.
const age = 19;
// Assume that the legal age to get a driving license
// is 18.
const isOldEnough = age >=18;

// Syntax similar to C
if (isOldEnough) console.log(1); // 1 denotes old enough
else
{
    console.log(0); // 0 denotes not old enough
    console.log(18 - age); // Output number of years required to wait.
}
// LECTURE 18: Type Conversion and Type Coersion
let inputYear = '1991';
// What is going to happen below is the conversion of the number 18 into a string, and then the concatenation of the string "18" to the inputYear string.
console.log(inputYear + 18); // ==> 199118
console.log(18 + inputYear); // ==> 181991
```

```
console.log(typeof inputYear); // ==>
  string

// To fix this, we have to convert
  inputYear into a string.
// Type Conversion:
inputYear = Number(inputYear); // converts
  `1991` to 1991
console.log(inputYear + 18); // ==> 1991

// What if we convert a string that is
  filled with
// characters into a number?
console.log(Number(`Jonathan`)); // ==>
  NaN (Not a Number)
// It means an invalid number.
console.log(typeof NaN); // Number --> NaN
  is still a number, but it represents an
  invalid number.

// converting numbers to string
console.log(String(123)); // ==> "123"

// Syntax is: NewType(Exp)
/*
In JS, we can convert
- Number --> string OR Number --> boolean
We cannot convert to undefined or NULL.
In practice, we rarely have to do type
  conversion, because JS automatically
  does type COERCION in many situations.
```

```
Type Coercion - Type coercion happens when
  an operator is dealing with two
  different types. JS will then, under the
  hood convert one of the values to match
  the other, so that the operation can be
  executed. (We already seen an example of
  this above.)
*/
console.log("I am " + 23 + " years old"); /
  / I am 23 years old.
// The + operator in JS has a type
  coercion to strings - whenever there is
  an operation between a string and a
  number, it will convert the number to a
  string.
// Because of type coercion, we don't have
  to write this:
console.log("I am " + String(23) + " years
  old");

// Not all operators do type coercion to
  string:
// The minus operator actually has a type
  coercion to numbers:
console.log("23" - "10" - 3); // ==> 10
console.log("23" - "10"); // ==> 13?
console.log("23" + "10" - 3); // ==> 2307

// The multiplication and division
  operator also has a type coercion to
  numbers:
```

```
60  console.log("23" * "2"); // ==> 46
61  console.log("22" / "2"); // ==> 11
62
63  // The logical operators also have a type
 •    coercion to numbers:
64  console.log("23" > "2"); // ==> true
65  console.log("23" < "2"); // ==> false
66
67  // Guess the following output
68  let n = '1' + 1;
69  n = n -1;
70  console.log(n); // ==> 10
71
```