



IF-ELSE STATEMENTS COURSE

APRIL 2022

YONATAN BENEZRA

- DO NOT DISTRIBUTE -

- DO NOT COPY – ALL RIGHTS RESERVED TO THE
AUTHOR –

In JavaScript, there are three forms of the if...else statement:

- if statement
- if...else statement
- if...else if...else statement

JavaScript if Statement

The syntax of the if statement is as follows:

```
if (condition) {  
    // the body of if  
}
```

The if statement evaluates the condition inside the parenthesis ().

- If the condition is evaluated to true, the code inside the body of the if is executed.
- If the condition is evaluated to false, the code inside the body of the if statement is skipped.

***Note:**

The code inside { } is the body of the if statement.

Example 1:

```
// check if the number is positive  
  
const number = 3;  
  
// check if number is greater than 0  
if (number > 0) {  
    // the body of the if statement  
    console.log("The number is positive");  
}
```

Output 1

if the number is 2
The number is positive
The if statement is true

Output 2

if the number is -1
The if statement is false.

Suppose the user entered -1. In this case, the condition, `number > 0`, evaluates to false. Hence, the body of the if statement is skipped.

JavaScript if...else Statement

An if statement can have an optional else clause. The syntax of the if...else statement is as follows:

```
if (condition) {  
    // block of code if condition is true  
} else {  
    // block of code if condition is false  
}
```

The if...else statement evaluates the condition inside the parenthesis.

If the condition is evaluated to true:

- the code inside the body of the if is executed
- the code inside the body of the else is skipped and therefore not executed

If the condition is evaluated to false:

- the code inside the body of else is executed
- the code inside the body of the if is skipped and therefore not executed

JavaScript if...else if Statement

The if...else statement is used to execute a block of code among two alternatives. However, if you need to make a choice between more than two alternatives, if...else if...else can be used.

The syntax of the if...else if...else statement is as follows:

```
if (condition1) {  
    // code block 1  
} else if (condition2){  
    // code block 2  
} else {  
    // code block 3  
}
```

- If condition1 evaluates to true, the code block 1 is executed.
- If condition1 evaluates to false, then condition2 is evaluated.
 - If the condition2 is true, the code block 2 is executed.
 - If the condition2 is false, the code block 3 is executed.

1st Condition is true

```
let number = 2;
if (number > 0) {
  // code
}
else if (number == 0){
  // code
}
else {
  //code
}
//code after if
```

2nd Condition is true

```
let number = 0;
if (number > 0) {
  // code
}
else if (number == 0){
  // code
}
else {
  //code
}
//code after if
```

All Conditions are false

```
let number = -2;
if (number > 0) {
  // code
}
else if (number == 0){
  // code
}
else {
  //code
}
//code after if
```

The above image depicts the process that takes place when executing an if...else if...else statement.

Example 3: if...else if Statement

```
// check if the number is positive, negative or zero
const number = 4;

// check if number is greater than 0
if (number > 0) {
  console.log("The number is positive");
}
// check if number is 0
else if (number == 0) {
  console.log("The number is 0");
}
// if number is neither greater than 0, nor zero
else {
  console.log("The number is negative");
}

console.log("The if...else if...else statement is easy");
```

Nested if...else Statement

You can also use an if...else statement inside of an if...else statement. This is known as a nested if...else statement.

Example 4: Nested if...else Statement

```
// check if the number is positive, negative or zero
const number = prompt("Enter a number: ");

if (number >= 0) {
  if (number == 0) {
    console.log("You entered number 0");
  } else {
    console.log("You entered a positive number");
  }
} else {
  console.log("You entered a negative number");
}
```

Suppose the user entered 5. In this case, the condition `number >= 0` evaluates to true, and the control of the program goes inside the outer if statement.

Then, the test condition, `number == 0`, of the inner if statement is evaluated. Since it's false, the else clause of the inner if statement is executed.

*Note:

As you can see, nested if...else statements can quickly become very complicated. Therefore, we should try to avoid using nested if...else whenever possible.

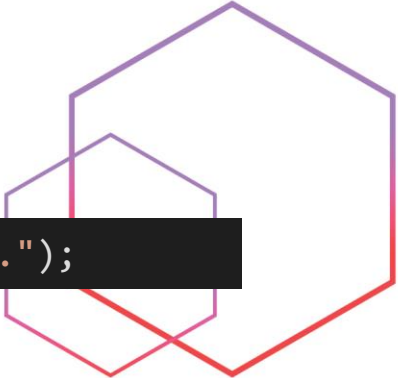
Body of if...else With Only One Statement

If the body of if...else has only one statement, we can omit `{ }` in our programs. For example, instead of the following:

```
const number = 2;
if (number > 0) {
  console.log("The number is positive.");
} else {
  console.log("The number is negative or zero.");
}
```

You can simply write the following:

```
const number = 2;
if (number > 0)
  console.log("The number is positive.");
else
```



```
console.log("The number is negative or zero.");
```

What is a Ternary Operator?

A ternary operator evaluates a condition and executes a block of code based on the condition. Its syntax is:

```
condition ? expression1 : expression2
```

The ternary operator evaluates the test condition.

- If the condition is true, expression1 is executed.
- If the condition is false, expression2 is executed.

The ternary operator takes three operands, hence the name ternary operator. It is also known as a conditional operator.

Let's write a program to determine if a student passed or failed the exam based on their obtained marks.

Example: JavaScript Ternary Operator

```
// program to check pass or fail  
  
let marks = 78;  
  
// check the condition  
let result = (marks >= 40) ? 'pass' : 'fail';  
  
console.log(`You ${result} the exam.`);
```

Output 1

Enter your marks: 78

You pass the exam.

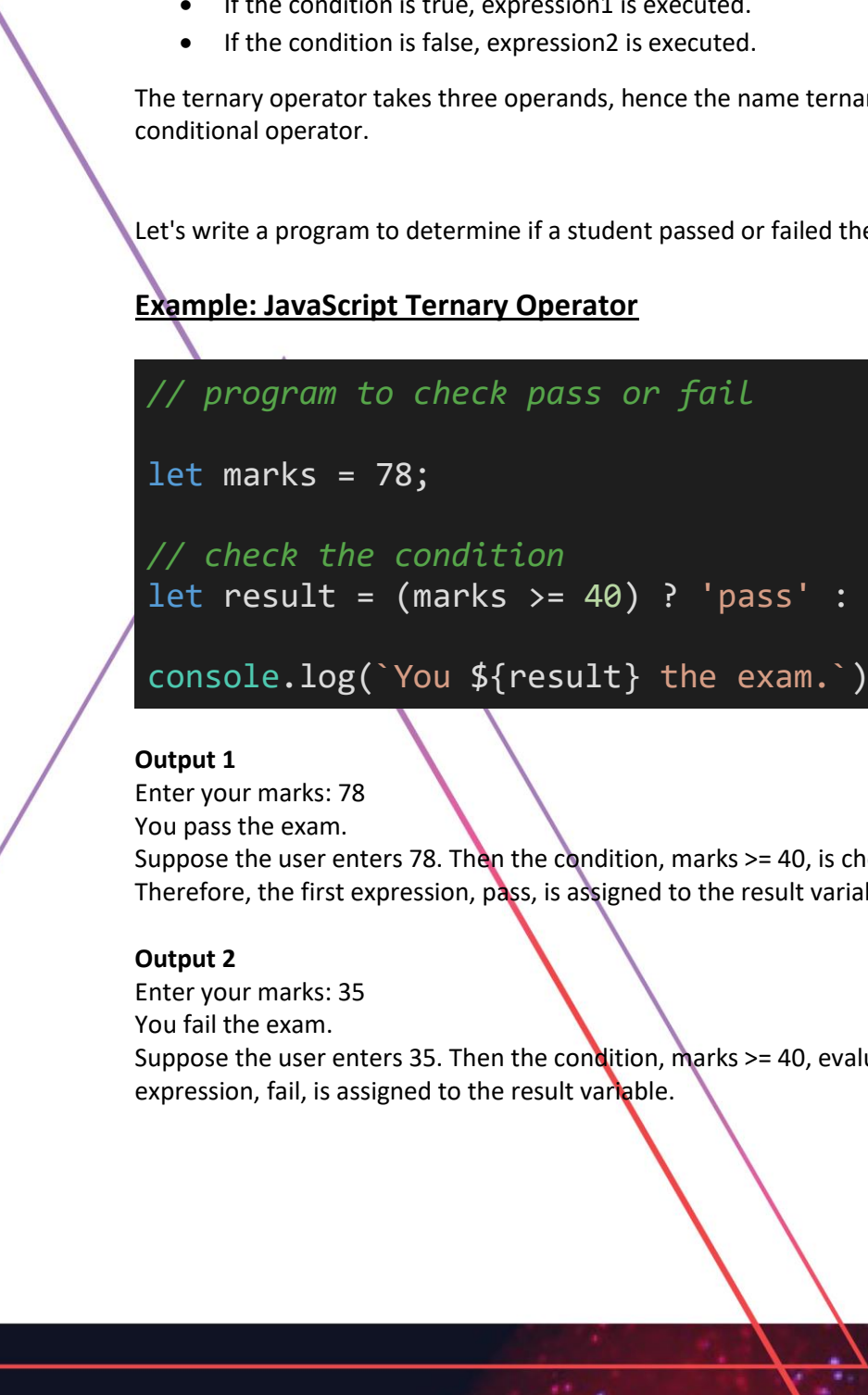
Suppose the user enters 78. Then the condition, marks >= 40, is checked, which evaluates to true. Therefore, the first expression, pass, is assigned to the result variable.

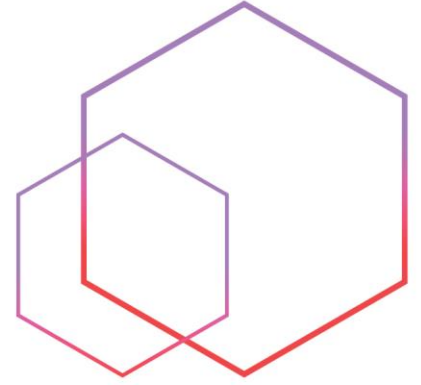
Output 2

Enter your marks: 35

You fail the exam.

Suppose the user enters 35. Then the condition, marks >= 40, evaluates as false. Therefore, the second expression, fail, is assigned to the result variable.





Ternary Operator Used Instead of if...else

In JavaScript, a ternary operator can be used to replace certain types of if..else statements. For example, you can replace the following code:

```
// check the age to determine the eligibility to vote
let age = 15;
let result;

if (age >= 18) {
    result = "You are eligible to vote.";
} else {
    result = "You are not eligible to vote yet.";
}

console.log(result);
```

With the below:

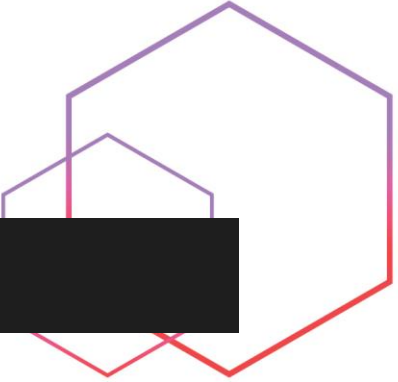
```
// ternary operator to check the eligibility to vote
let age = 15;
let result =
    (age >= 18) ? "You are eligible to vote." : "You are
not eligible to vote yet";
console.log(result);
```

The output of both programs will be the same.

Nested Ternary Operators

You can also nest one ternary operator as an expression inside another ternary operator. For example:

```
// program to check if number is positive, negative or
zero
let a = 3;
let result = (a >= 0) ? (a == 0 ? "zero" : "positive") :
```



```
"negative";  
console.log(`The number is ${result}.`);
```

Note:

You should try to avoid nested ternary operators whenever possible as they make your code hard to read.

