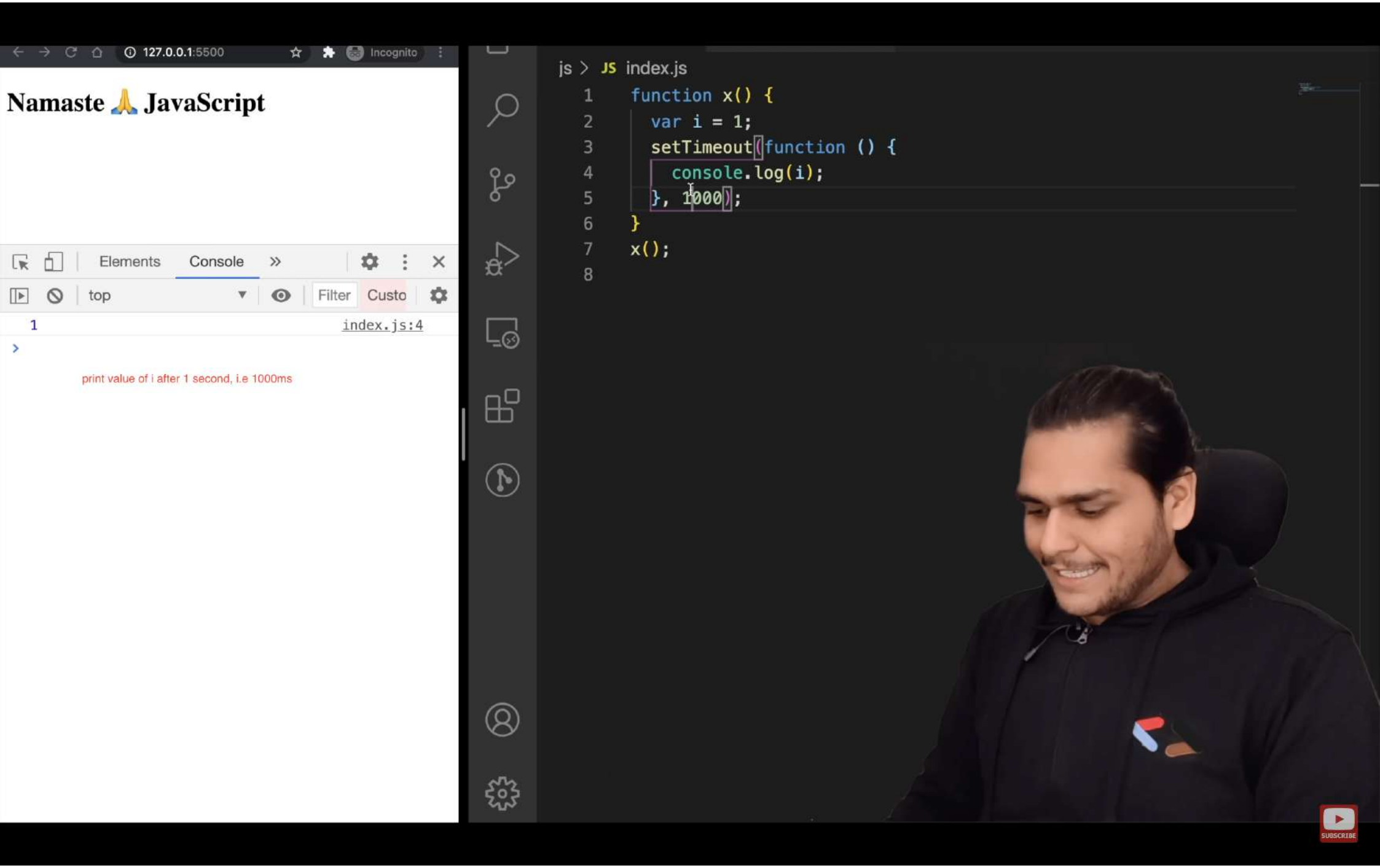
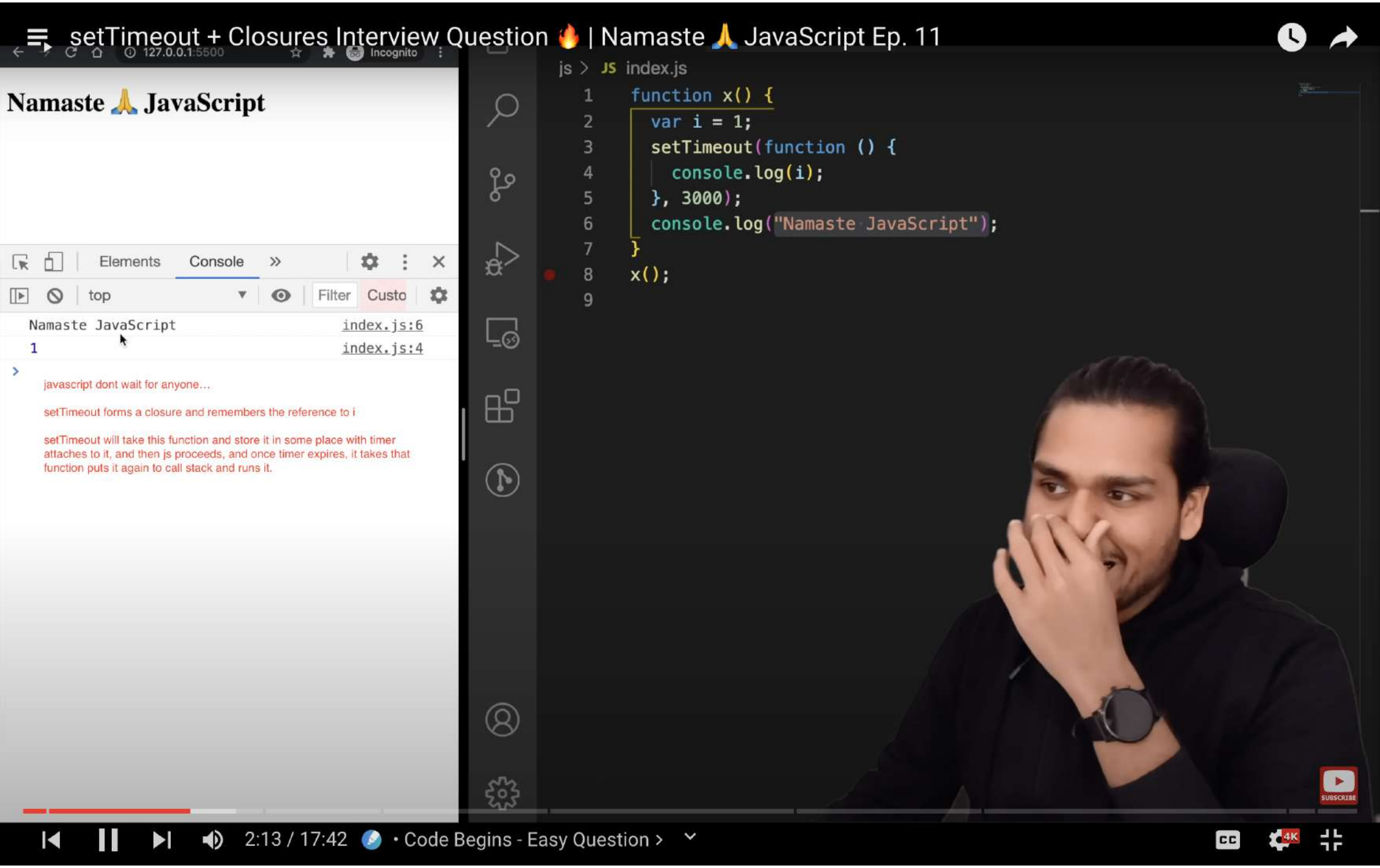
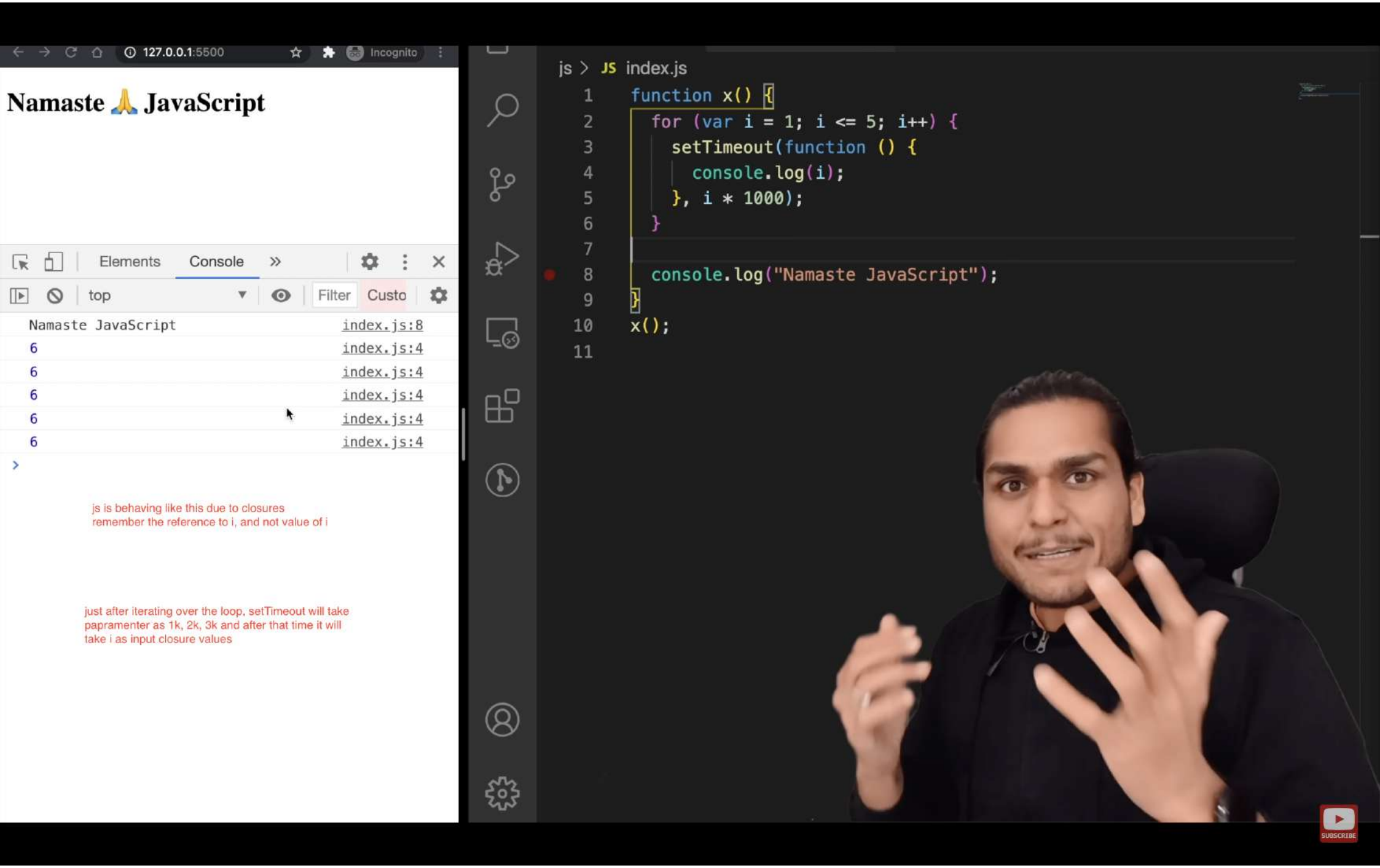
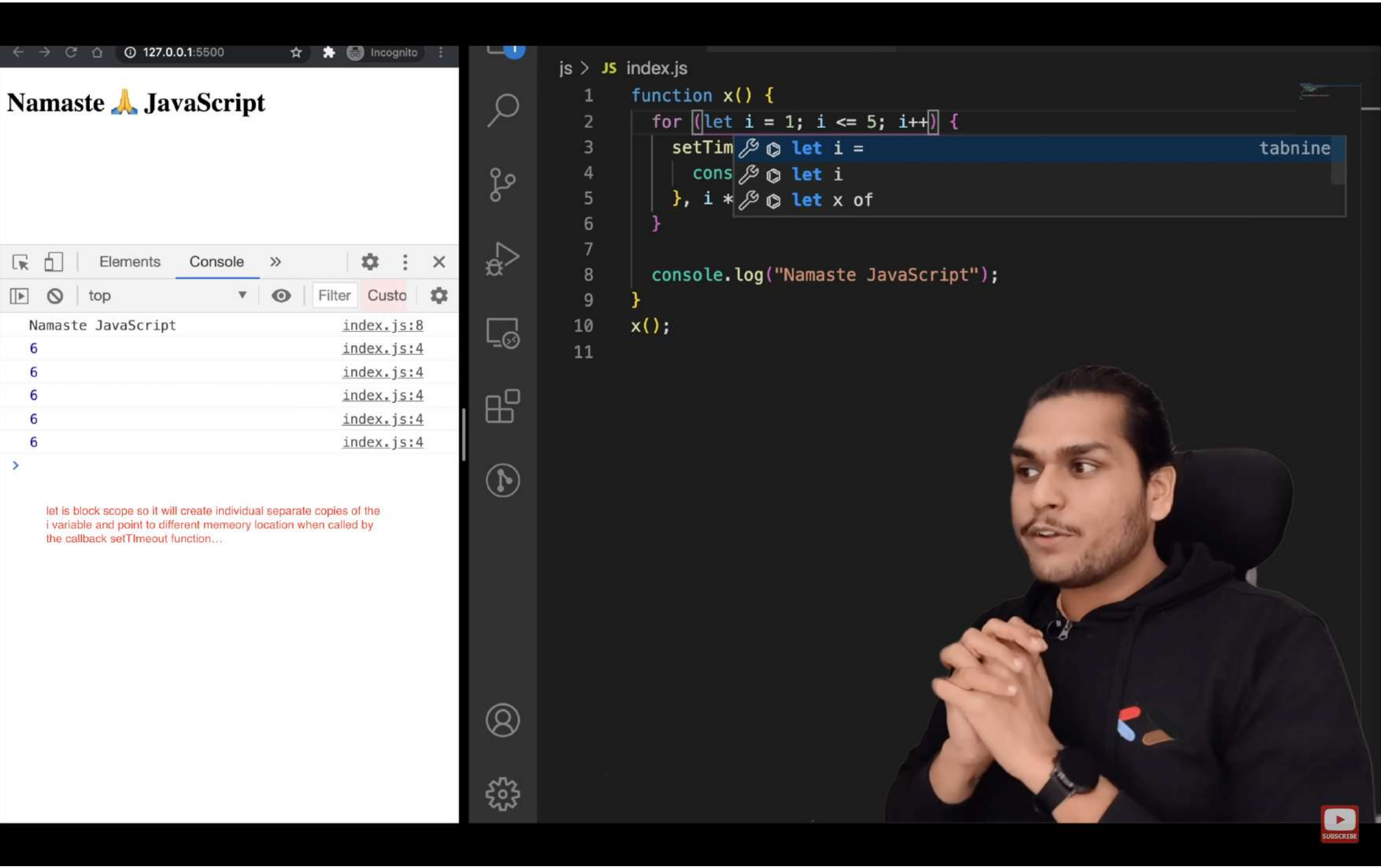
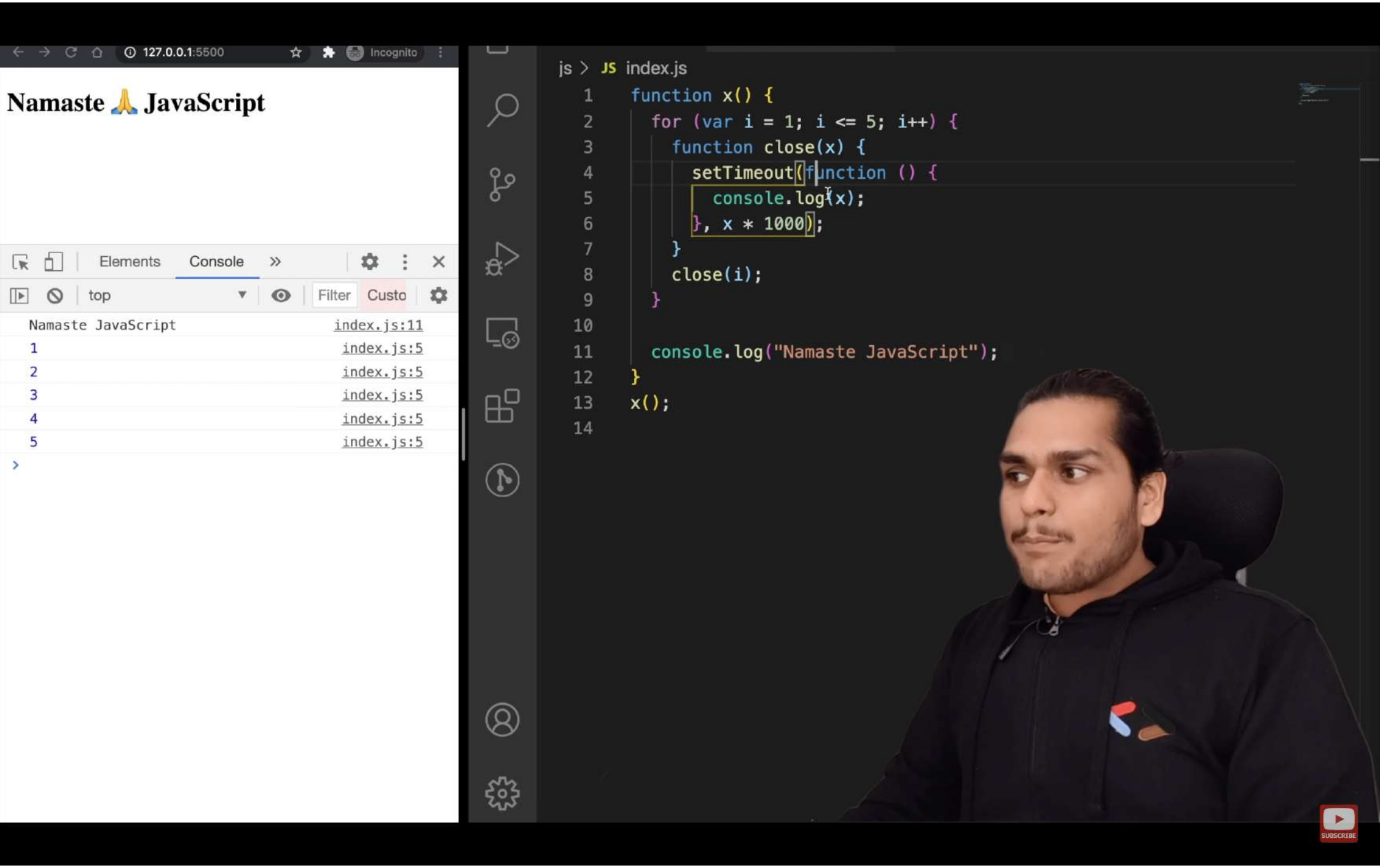

127.0.0.1:5500
Incognito
Namaste 🙏 JavaScript
Elements
Console
top
Filter
Custo
1
index.js:4
print value of i after 1 second, i.e 1000ms
js > JS index.js
function x() {
var i = 1;
setTimeout(function () {
console.log(i);
}, 1000);
}
x();
SUBSCRIBE


setTimeout + Closures Interview Question 🔥 | Namaste 🙏 JavaScript Ep. 11
127.0.0.1:5500
Incognito
Namaste 🙏 JavaScript
Elements
Console
top
Filter
Custo
Namaste JavaScript
index.js:6
1
index.js:4
javascript dont wait for anyone...
setTimeout forms a closure and remembers the reference to i
setTimeout will take this function and store it in some place with timer attaches to it, and then js proceeds, and once timer expires, it takes that function puts it again to call stack and runs it.
js > JS index.js
function x() {
var i = 1;
setTimeout(function () {
console.log(i);
}, 3000);
console.log("Namaste JavaScript");
}
x();
SUBSCRIBE
2:13 / 17:42
• Code Begins - Easy Question ›
4K


127.0.0.1:5500
Incognito
Namaste 🙏 JavaScript
Elements
Console
top
Filter
Custo
Namaste JavaScript index.js:8
6 index.js:4
6 index.js:4
6 index.js:4
6 index.js:4
6 index.js:4
js is behaving like this due to closures
remember the reference to i, and not value of i
just after iterating over the loop, setTimeout will take
papramenter as 1k, 2k, 3k and after that time it will
take i as input closure values
js > JS index.js
function x() {
for (var i = 1; i <= 5; i++) {
setTimeout(function () {
console.log(i);
}, i * 1000);
}
console.log("Namaste JavaScript");
}
x();
SUBSCRIBE


127.0.0.1:5500
Incognito
Namaste 🙏 JavaScript
Elements
Console
top
Filter
Custo
Namaste JavaScript index.js:8
6 index.js:4
6 index.js:4
6 index.js:4
6 index.js:4
6 index.js:4
let is block scope so it will create individual separate copies of the i variable and point to different memeory location when called by the callback setTImeout function…
js > JS index.js
function x() {
for (let i = 1; i <= 5; i++) {
setTim
let i =
tabnine
cons
let i
}, i *
let x of
}
console.log("Namaste JavaScript");
}
x();
SUBSCRIBE


127.0.0.1:5500
Incognito
Namaste 🙏 JavaScript
Elements
Console
top
Filter
Custo
Namaste JavaScript index.js:11
1 index.js:5
2 index.js:5
3 index.js:5
4 index.js:5
5 index.js:5
js > JS index.js
function x() {
for (var i = 1; i <= 5; i++) {
function close(x) {
setTimeout(function () {
console.log(x);
}, x * 1000);
}
close(i);
}
console.log("Namaste JavaScript");
}
x();
SUBSCRIBE