JavaScript Final Quest

Total puncte 82/100

A sosit momentul testului final de cunoștințe. Prin acest quest, vei verifica informațiile pe care le-ai adunat până acum.

- Quest-ul este format din 30 întrebări de tip grilă.
- **Ai nevoie de cel puțin 50 de puncte pentru a nu pierde o viață.**
- Quest-ul nu este cronometrat.
- Poți trimite răspunsurile tale o singură dată. Odată trimise ele nu pot fi modificate.
- **Termenul limită pentru acest quest este 14 iulie 2025, ora 23:59.**

Adresă de e-mail *

antonio.soare.42005@gmail.com

Assessment form 82 din 100 puncte

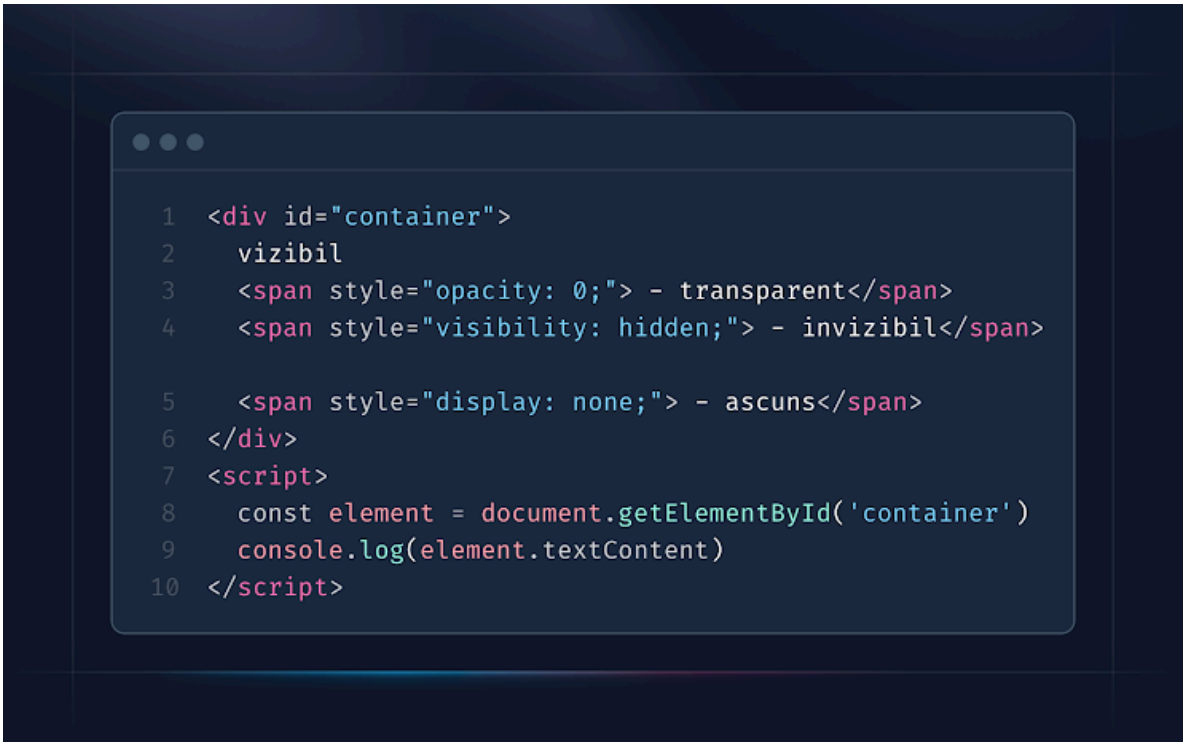
✓ Ce mesaj va fi afișat când apelăm funcția greet? 3/3

- [ ] Maria
- [ ] Bună, name
- [x] Bună, Maria ✓
- [ ] Error

Feedback

Funcția primește parametrul "Maria" și îl concatenează cu "Bună, " pentru a forma salutul.

✗ Ce va afișa textContent pentru acest element? 0/3



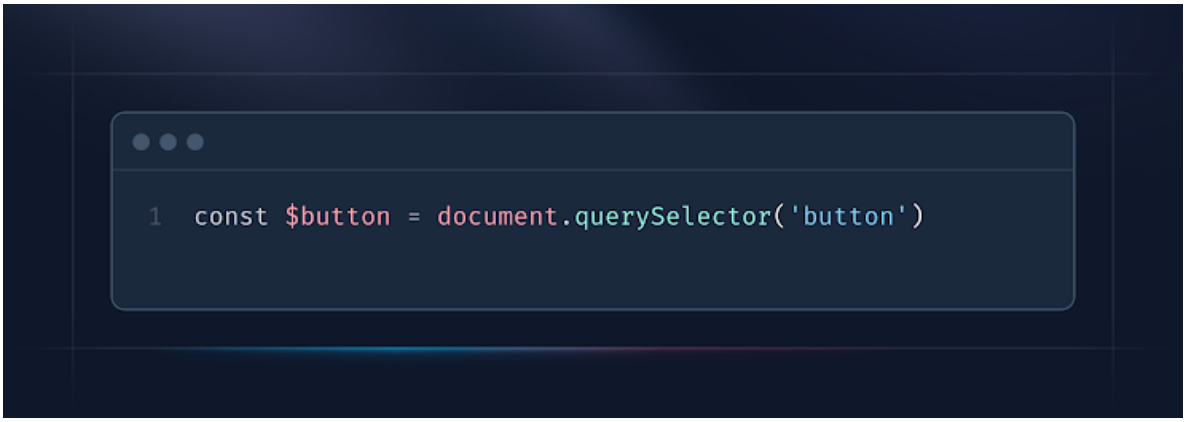
```
<div id="container">
  vizibil
  <span style="opacity: 0;"> - transparent</span>
  <span style="visibility: hidden;"> - invizibil</span>

  <span style="display: none;"> - ascuns</span>
</div>
<script>
  const element = document.getElementById('container')
  console.log(element.textContent)
</script>
```

- (●) "vizibil" ✗
- ( ) "vizibil - transparent"
- ( ) "vizibil - transparent - invizibil"
- ( ) "vizibil - transparent - invizibil - ascuns"

Feedback

textContent obține tot textul din DOM, ignorând complet stilizarea CSS - include text din elemente cu display: none, visibility: hidden, sau opacity: 0.

✓ Cum adaugi o clasă CSS la un element DOM fără a șterge clasele existente? 3/3



- [ ] $button.class.push('primary')
- [ ] $button.classes.push('primary')
- [ ] $button.className = 'primary'
- [x] $button.classList.add('primary') ✓

Feedback

Metoda classList.add() adaugă o clasă la un element fără a afecta celelalte clase existente.

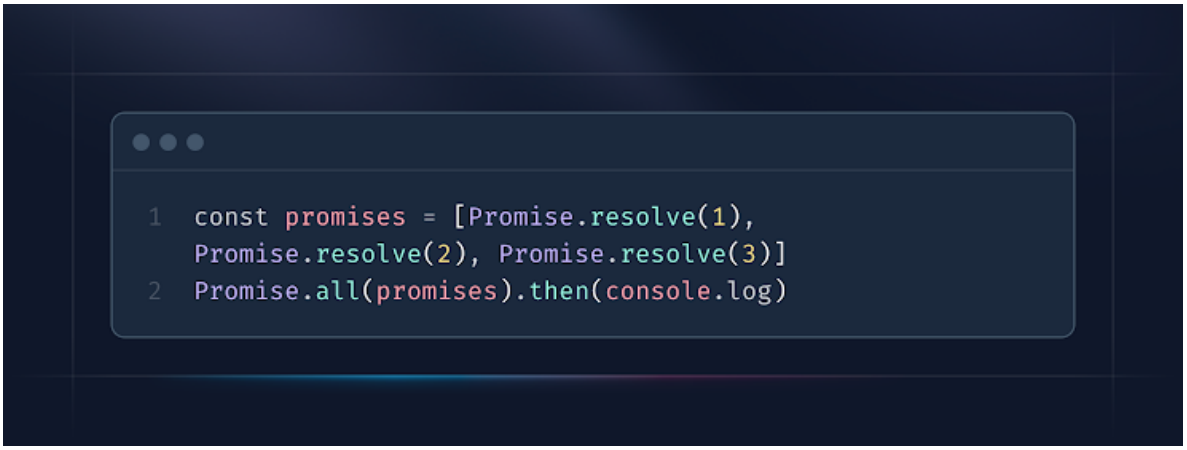
✓ Ce face operatorul optional chaining? 4/4

- ○ Afișează 'name'
- ○ Afișează null
- ◉ Afișează undefined ✓
- ○ Generează eroare

Feedback

Optional chaining (?.) returnează undefined dacă proprietatea nu există, în loc să genereze eroare.

✓ Ce face Promise.all()? 4/4



- (●) Afișează [1, 2, 3] ✓
- ( ) Afișează 1, apoi 2, apoi 3
- ( ) Afișează primul Promise care se rezolvă
- ( ) Generează eroare

Feedback

Promise.all() așteaptă ca toate Promise-urile să se rezolve și returnează un array cu rezultatele.

✓ Care este metoda corectă pentru a salva date în localStorage? 3/3

- [ ] localStorage.save('key', 'value')
- [ ] localStorage.push('key', 'value')
- [x] localStorage.setItem('key', 'value') ✓
- [ ] localStorage.addItem('key', 'value')

Feedback

Metoda setItem() este folosită pentru a stoca date în localStorage, primind doi parametri: cheia și valoarea.

✓ Ce sunt Props în React? 4/4

- ( ) Props sunt variabile globale în React
- ( ) Props sunt stiluri CSS pentru componente
- ( ) Props sunt funcții care gestionează evenimentele
- (•) Props sunt date care se trimit de la componenta părinte la componenta copil ✓

Feedback

Props (properties) sunt date care se trimit de la o componentă părinte către componentele copil, similar cu atributele HTML.

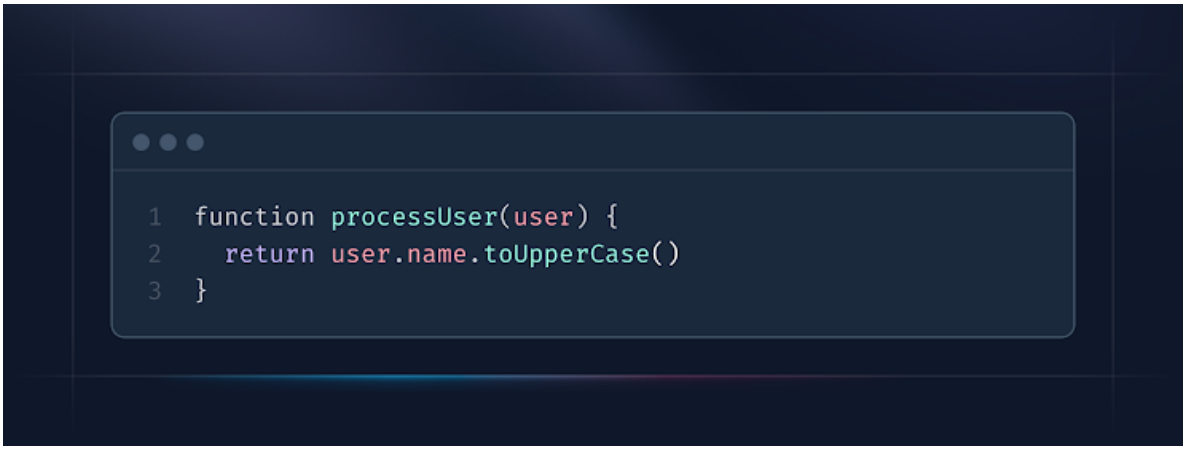
✓ Cum trimiți date în format JSON către un server folosind fetch API? 3/3

- [ ] post(url, JSON.stringify(data))
- [ ] fetch(url, { method: 'POST' }, data)
- [ ] fetch(url, { method: 'POST', body: data })
- [x] fetch(url, { method: 'POST', headers: { 'Content-Type': 'application/json' }, body: JSON.stringify(data) }) ✓

Feedback

Pentru a trimite date JSON într-un request POST, trebuie să specificăm metoda, să adăugăm header-ul 'Content-Type': 'application/json' și să convertim datele în format JSON folosind JSON.stringify().

✓ Ce problemă are această funcție? 3/3

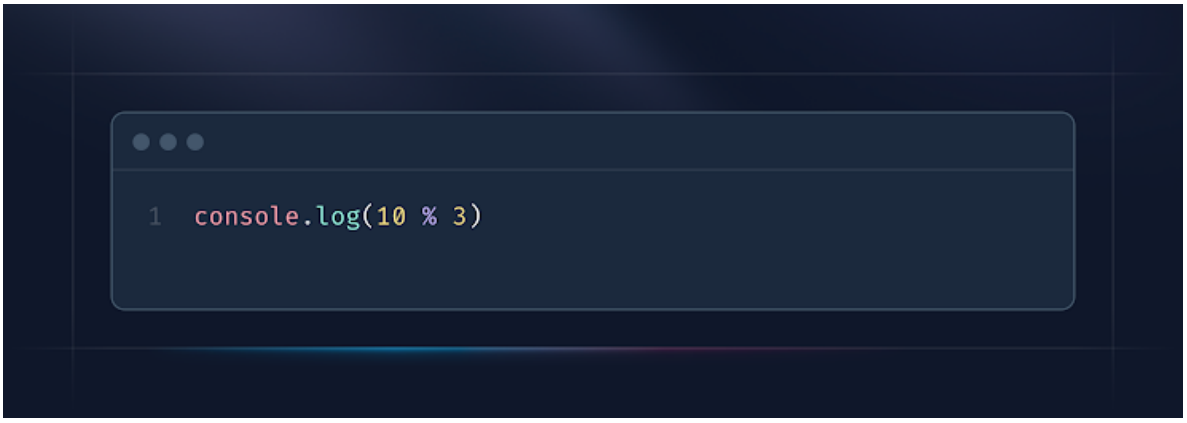


- ○ Sintaxa este greșită
- ○ toUpperCase() nu există
- ○ Funcția nu returnează nimic
- ◉ Nu verifică dacă user și user.name există ✓

Feedback

Funcția ar genera eroare dacă user este null/undefined sau dacă user.name nu există.

✓ Ce rezultat va returna operatorul modulo în următoarea expresie? 3/3



- [x] 1 ✓
- [ ] 3
- [ ] 3.(3)
- [ ] 10

Feedback

Operatorul % (modulo) returnează restul împărțirii. 10 împărțit la 3 dă 3 cu rest 1.

✓ Ce va afișa acest cod cu scope-ul variabilelor? 3/3

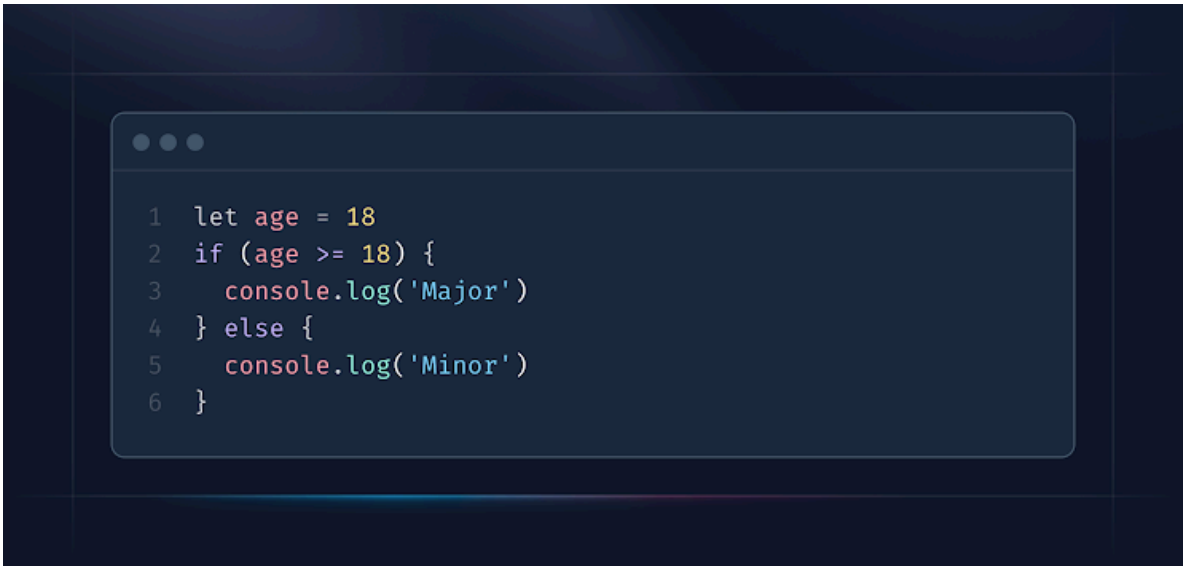
```
function outer() {
  let a = 1
  function inner() {
    let a = 2
    console.log(a)
  }
  inner()
  console.log(a)
}
outer()
```

- ○ 1, 1
- ◉ 2, 1 ✓
- ○ 2, 2
- ○ Error

Feedback

Variabilele let au block scope. Variabila a din funcția inner este diferită de cea din outer.

✓ Cum va clasifica următorul sistem o persoană de 18 ani? 3/3



- [ ] 18
- [x] Major ✓
- [ ] Minor
- [ ] Error

Feedback

Condiția age >= 18 se evaluează la true deoarece age este 18, astfel codul din blocul if este executat.

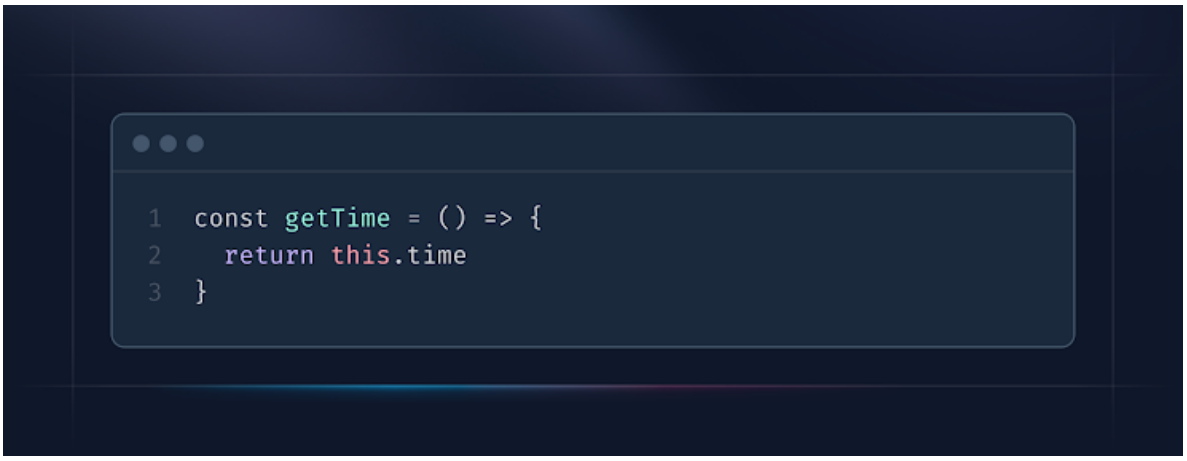
✕ Ce face Promise.race()? 0/4

- ○ Afișează 'fast'
- ○ Afișează 'slow'
- ◉ Afișează ['fast', 'slow'] ✕
- ○ Generează eroare

Feedback

Promise.race() se rezolvă cu primul Promise care se finalizează.

✓ Ce valoare are "this" într-un arrow function? 3/3

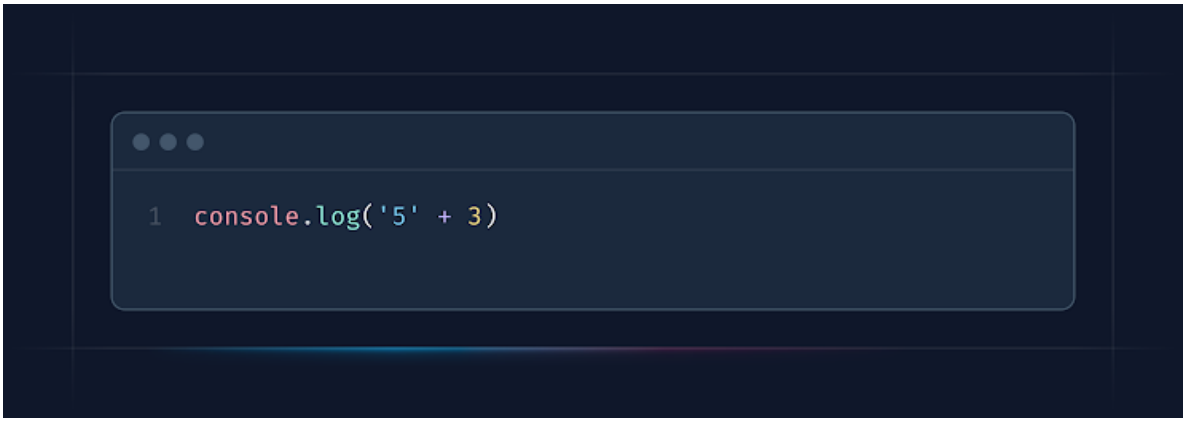


- [ ] Obiectul global (window în browser)
- [ ] Obiectul care a apelat funcția
- [x] Valoarea lui this din contextul în care a fost creată funcția ✓
- [ ] Întotdeauna "undefined"

Feedback

Arrow functions nu au propriul lor this; ele moștenesc this din contextul lexical în care sunt definite.

✓ Cum se comportă operatorul + când lucrăm cu un string și un număr? 3/3



- [ ] 8
- [ ] "8"
- [x] "53" ✓
- [ ] NaN

Feedback

Când operatorul + este folosit cu un string și un număr, JavaScript convertește numărul într-un string și le concatenează.

✓ Ce se întâmplă cu această funcție arrow? 3/3

```
const obj = {
  name: 'Test',
  regularFunction: function () {
    console.log(this.name)
  },
  arrowFunction: () => {
    console.log(this.name)
  },
}
obj.regularFunction()
obj.arrowFunction()
```

- ○ Test, Test
- ◉ Test, undefined ✓
- ○ undefined, Test
- ○ undefined, undefined

Feedback

Funcțiile arrow nu au propriul lor this, ci îl moștenesc din contextul lexical (în acest caz, obiectul global).

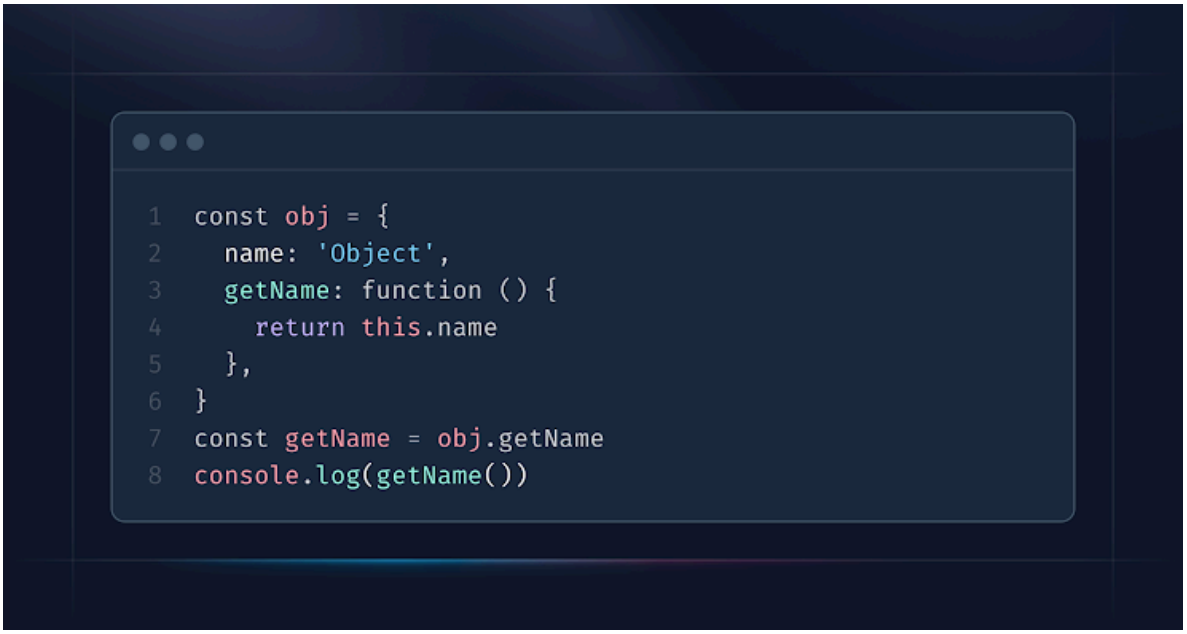
✓ Ce returnează o funcție async? 3/3

- ○ 'data'
- ○ undefined
- ◉ Promise object ✓
- ○ Error

Feedback

Funcțiile async returnează întotdeauna un Promise, chiar dacă returnezi o valoare directă.

✗ Ce afișează acest cod cu "this" context? 0/4



```
const obj = {
  name: 'Object',
  getName: function () {
    return this.name
  },
}
const getName = obj.getName
console.log(getName())
```

- (●) 'Object' ✗
- ( ) null
- ( ) undefined
- ( ) Error

Feedback

Când funcția este atribuită unei variabile și apelată independent, this nu mai pointează la obiect.

✓ Ce face event delegation? 3/3

- ○ Șterge automat event listeners
- ○ Adaugă event listener la toate butoanele
- ○ Împarte evenimentul în mai multe funcții
- ◉ Folosește un singur event listener pe părintele pentru a gestiona evenimente de la copii ✓

Feedback

Event delegation folosește un singur listener pe un element părinte pentru a gestiona evenimente de la descendenți.

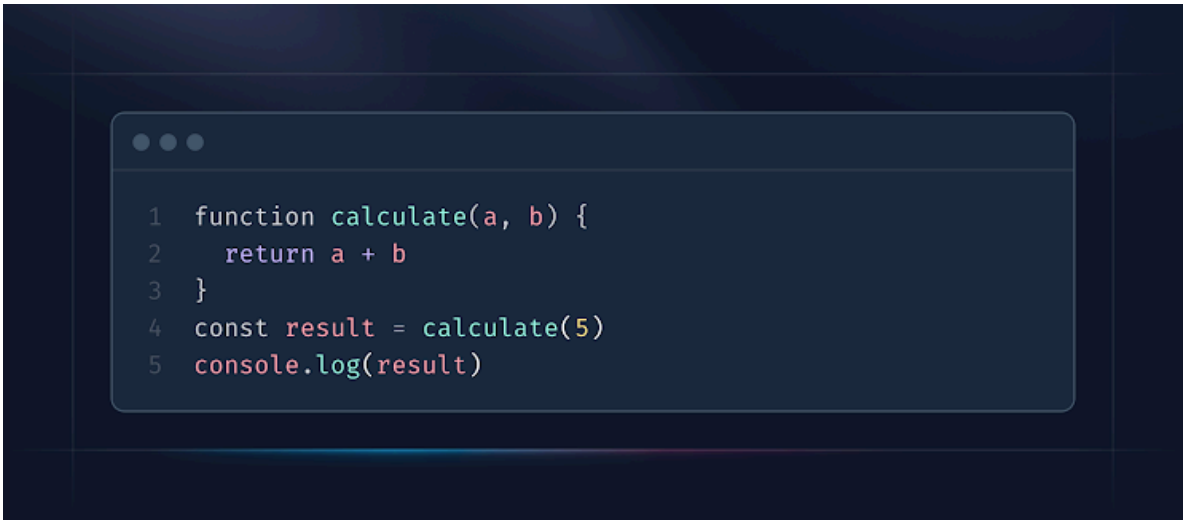
✗ Cum poți accesa valoarea orașului dintr-un obiect imbricat? 0/3

- [x] user.address.city ✓
- [ ] address.city
- [ ] user['address']['city']
- [ ] user[address][city]

Feedback

Proprietățile obiectelor în JavaScript pot fi accesate fie prin notația cu punct (dot notation), fie prin notația cu paranteze pătrate (bracket notation). Ambele metode sunt corecte pentru acest exemplu.

✓ Ce se întâmplă cu această funcție? 3/3

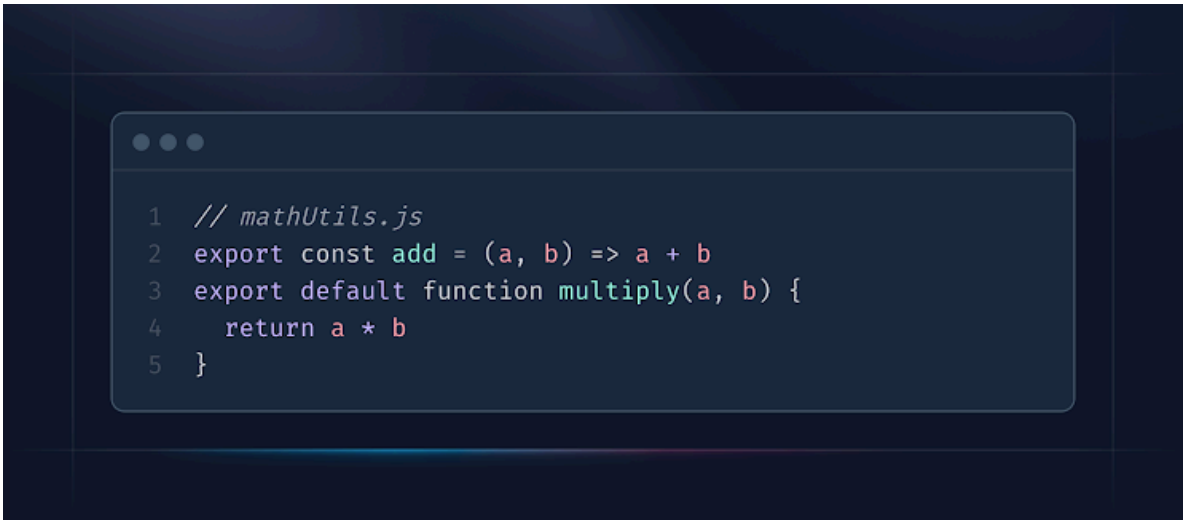


- ○ 5
- ○ 10
- ◉ NaN ✓
- ○ undefined

Feedback

Parametrul b este undefined, și 5 + undefined = NaN în JavaScript.

✓ Care este sintaxa corectă pentru module ES6? 4/4



- ○ import { add, multiply } from './mathUtils.js'
- ◉ import multiply, { add } from './mathUtils.js' ✓
- ○ import * as math from './mathUtils.js'
- ○ import './mathUtils.js'

Feedback

Export default se importă fără acolade, named exports se importă cu acolade.

✗ Ce va afișa acest cod de React? 0/4

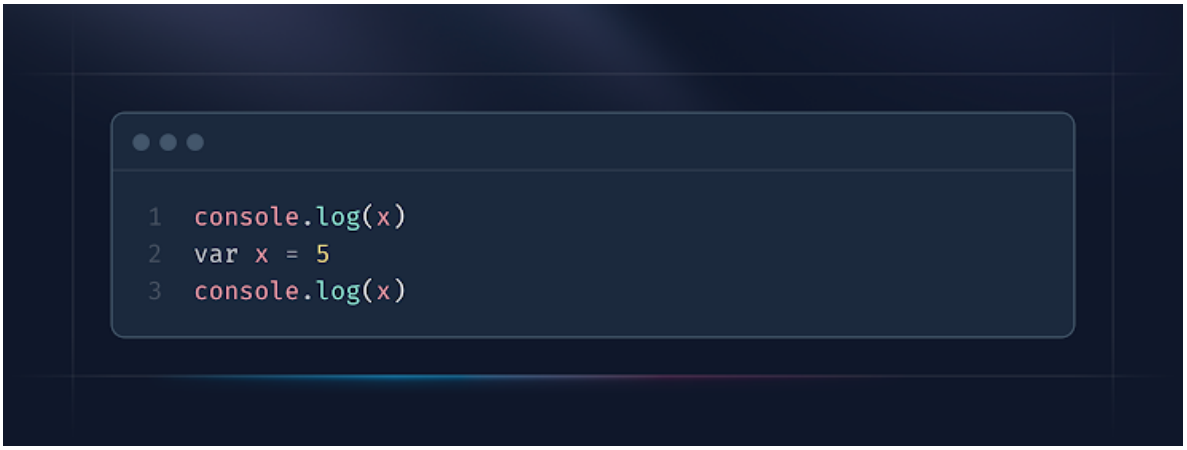
```jsx
function Parent() {
  const [count, setCount] = useState(0)

  const handleClick = () => {
    setCount(count + 1)
    setCount(count + 1)
    setCount(count + 1)
  }

  return (
    <div>
      <p>Count: {count}</p>
      <button onClick={handleClick}>Increment</button>
    </div>
  )
}
```

- ○ Count se incrementează cu 0 la fiecare click
- ○ Count se incrementează cu 1 la fiecare click
- ◉ Count se incrementează cu 3 la fiecare click ✗
- ○ Generează eroare

Feedback

React batchează update-urile de state, și toate cele 3 setState folosesc aceeași valoare inițială (count). Pentru increment corect, trebuie să folosești forma funcțională: setCount(prev => prev + 1).

✓ Ce va afișa acest cod cu hoisting? 3/3

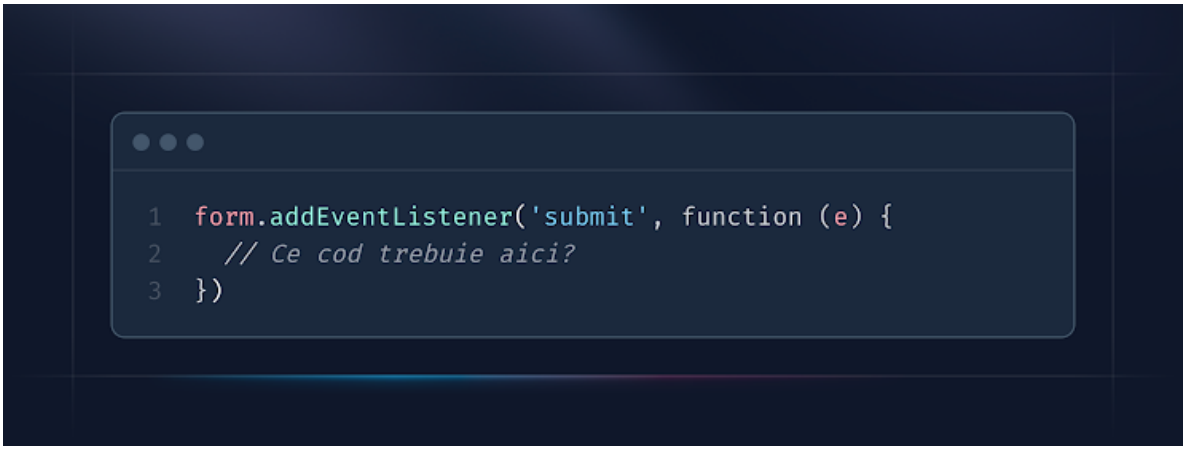


- ○ 5, 5
- ○ null, 5
- ◉ undefined, 5 ✓
- ○ Error, 5

Feedback

Declarația var x este ridicată (hoisted) în partea de sus, dar inițializarea rămâne în același loc, rezultând undefined prima dată.

✓ Cum previi comportamentul implicit al unui formular? 3/3



- ○ e.stopPropagation()
- ◉ e.preventDefault() ✓
- ○ e.stopDefault()
- ○ return false

Feedback

preventDefault() oprește comportamentul implicit al evenimentului (în acest caz, submit-ul formularului).

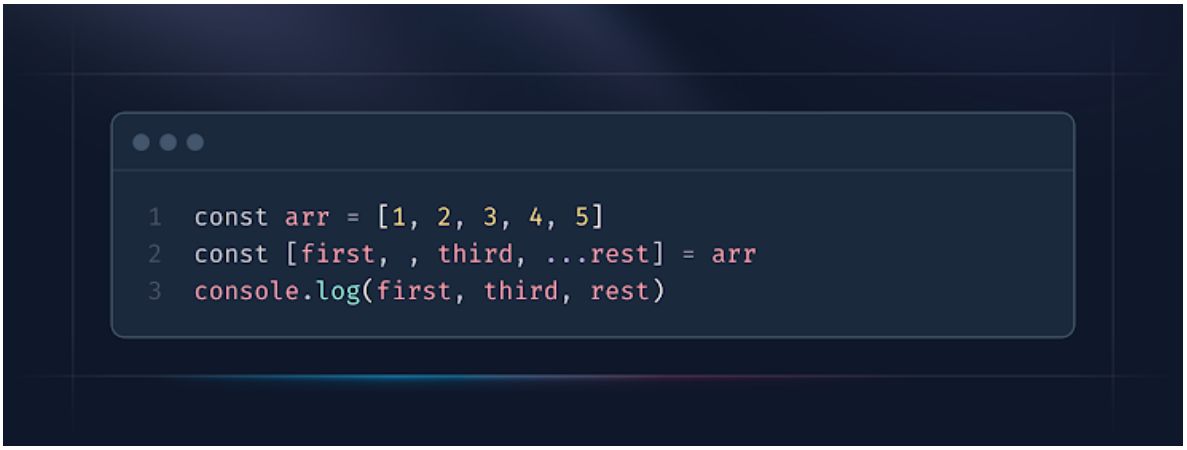
✓ Care este sintaxa corectă JSX? 4/4

- (●) return Hello {props.name} ✓
- ( ) return Hello ${props.name}
- ( ) return Hello + props.name
- ( ) return Hello $props.name

Feedback

JSX folosește acolade {} pentru a evalua expresii JavaScript, nu template literals cu ${}.

✓ Care este rezultatul acestei destructurări? 4/4

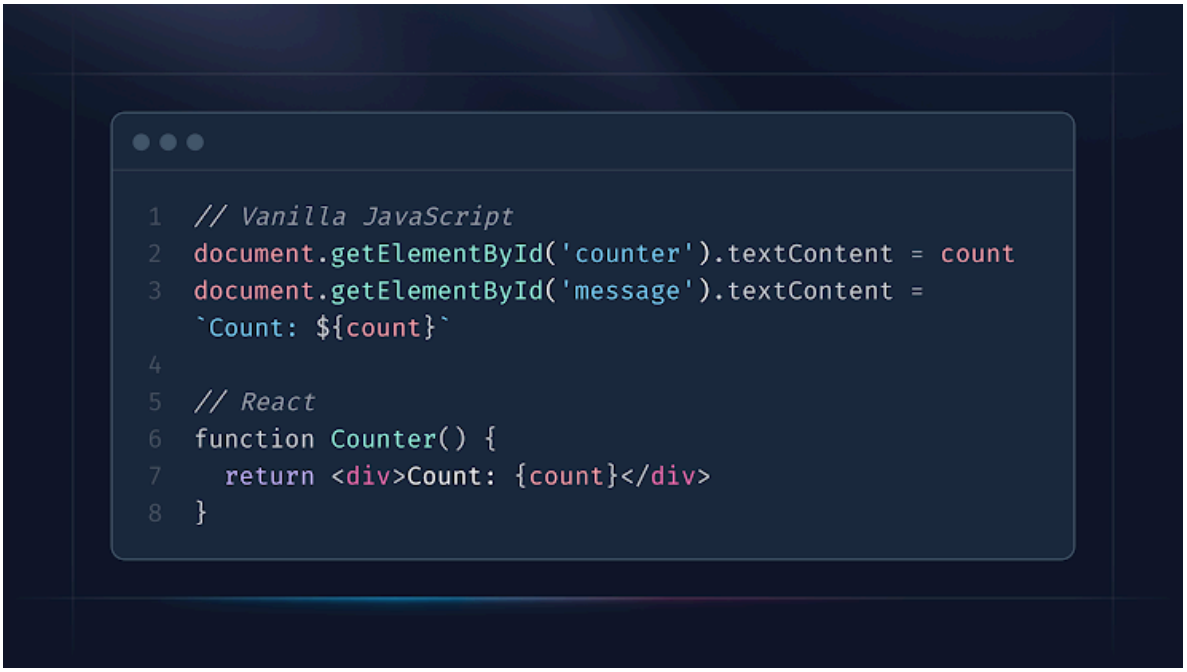


- (●) 1, 3, [4, 5] ✓
- ( ) 1, 2, [3, 4, 5]
- ( ) 1, undefined, [4, 5]
- ( ) Error

Feedback

Destructurarea sare peste al doilea element și colectează restul într-un array.

✓ Ce este React și care este avantajul principal? 4/4

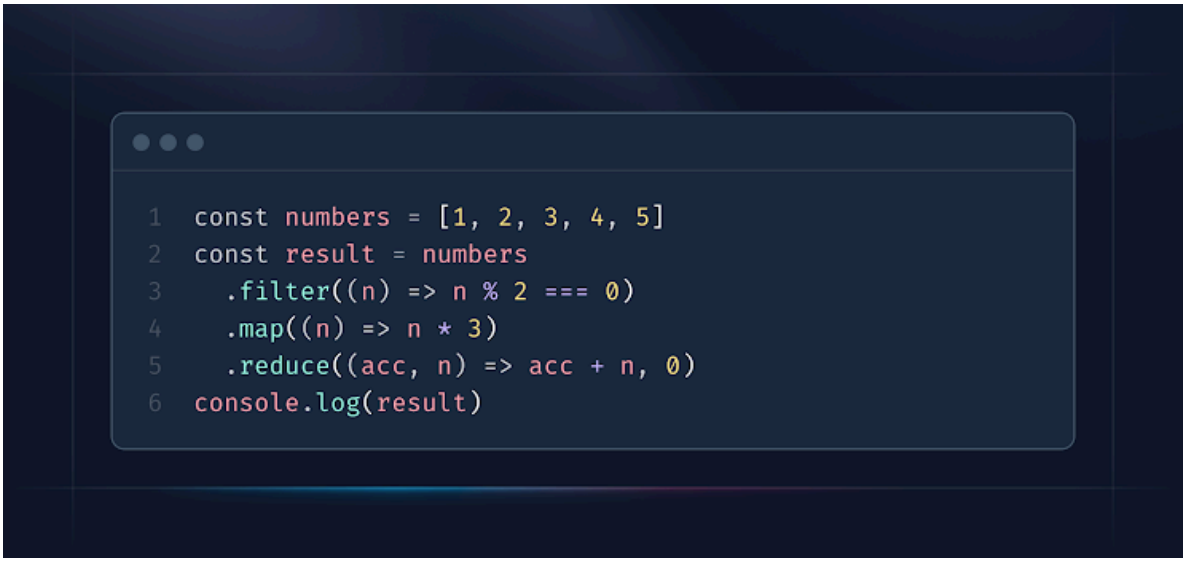


- ○ React funcționează doar pe server
- ○ React înlocuiește complet HTML și CSS
- ○ React este un framework complet pentru aplicații web
- ◉ React este o bibliotecă pentru construirea interfețelor utilizator ✓

Feedback

React este o bibliotecă JavaScript pentru construirea interfețelor utilizator, focusată pe componente reutilizabile și gestionarea eficientă a stării.

✓ Care este rezultatul acestei prelucrări de array? 3/3



```
const numbers = [1, 2, 3, 4, 5]
const result = numbers
  .filter((n) => n % 2 === 0)
  .map((n) => n * 3)
  .reduce((acc, n) => acc + n, 0)
console.log(result)
```

- (●) 18 ✓
- ( ) 30
- ( ) [2, 4]
- ( ) [6, 12]

Feedback

Filter păstrează numerele pare [2, 4], map le înmulțește cu 3 [6, 12], reduce le adună: 6 + 12 = 18.

✓ Care element va fi afișat la accesarea indexului 1 dintr-un array? 3/3

- [ ] mere
- [x] pere ✓
- [ ] banane
- [ ] Error

Feedback

Indexarea array-urilor începe de la 0, deci fruits[1] se referă la al doilea element din array, care este "pere".