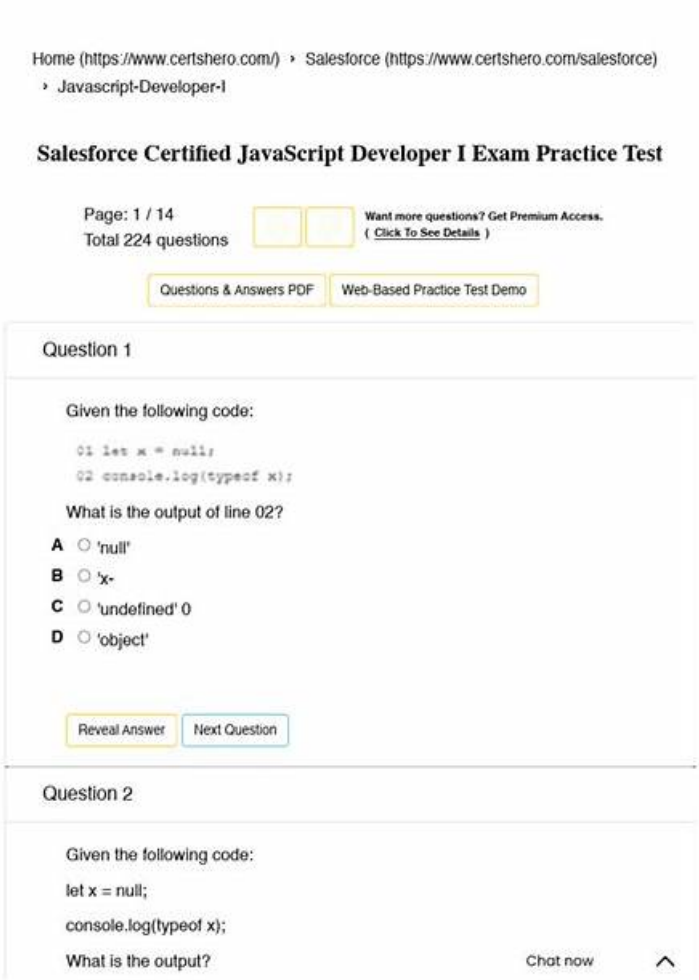
Salesforce JS-Dev-101: Salesforce Certified JavaScript Developer - Multiple Choice braindumps PDF & Testking echter Test



Laden Sie die neuesten ExamFragen JS-Dev-101 PDF-Versionen von Prüfungsfragen kostenlos von Google Drive herunter: https://drive.google.com/open?id=1WU90qdpbL8GBXj6s0Ug6GUIgrmBdKwi-

Wenn Sie die Produkte von ExamFragen benutzen, setzten Sie dann den ersten Fuß auf die Spitze der IT-Branche und nähern Ihrem Traum. Die Quizfragen und Antworten von ExamFragen können Ihnen nicht nur helfen, die Salesforce JS-Dev-101 Zertifizierungsprüfung zu bestehen und Ihre Fachkenntnisse zu konsolidieren. Außerdem bieten wir Ihnen auch einen einjährigen kostenlosen Update-Service.

Salesforce JS-Dev-101 Prüfungsplan:

Thema	Einzelheiten
Thema 1	• Debugging and Error Handling: Covers proper error handling techniques and the use of the console and breakpoints to debug code.
Thema 2	• Testing: Covers evaluating unit test effectiveness against a block of code and modifying tests to improve their coverage and reliability.

Thema 3	• Variables, Types, and Collections: Covers declaring and initializing variables, working with strings, numbers, dates, arrays, and JSON, along with understanding type coercion and truthy • falsy evaluations.
Thema 4	• Server Side JavaScript: Covers Node.js implementations, CLI commands, core modules, and package management solutions for given scenarios.

>> JS-Dev-101 German <<

JS-Dev-101 Dumps Deutsch & JS-Dev-101 Deutsch Prüfung

Seit Jahren ist Salesforce JS-Dev-101 Prüfung eine sehr populäre Prüfung. Heutzutage wird Salesforce Zertifizierung immer wichtiger. Als von IT-Industrie international anerkannte Prüfung wird JS-Dev-101 eine der wichtigsten Prüfungen in Salesforce. Sie können viele Vorteile bekommen, wenn Sie das JS-Dev-101 Zertifikat bekommen. Salesforce JS-Dev-101 Dumps von ExamFragen gilt als das unentbehrliche Gerät, womit Sie die Salesforce JS-Dev-101 Prüfung vorbereiten, weil es den besten Nachschlag für Salesforce JS-Dev-101 Zertifizierungsprüfung ist.

Salesforce Certified JavaScript Developer - Multiple Choice JS-Dev-101 Prüfungsfragen mit Lösungen (Q122-Q127):

122. Frage
Code:
01 const sayHello = (name) => {
02 console.log('Hello ', name);
03 };
04
05 const world = () => {
06 return 'World';
07 };
08
09 sayHello(world);
This does not print "Hello World".
What change is needed?

- A. Change line 5 to function world() {
- B. Change line 7 to }();
- C. Change line 9 to sayHello(world)();
- D. Change line 2 to console.log('Hello', name());

Antwort: D

Begründung:
Comprehensive and Detailed Explanation From Exact Extract JavaScript Knowledge:
Currently:
sayHello expects a value name and prints it.
world is a function that returns 'World'.
sayHello(world); passes the function object itself, not the result of calling it.
So name inside sayHello is a function, not the string "World".
To keep the call sayHello(world) yet get "World", sayHello must call the function parameter:
const sayHello = (name) => {
console.log('Hello', name());
};
Now:
sayHello(world) passes the function.
Inside sayHello, name() calls world(), which returns "World".
The console logs "Hello World".
Why the others are wrong:

B: Changing line 7 to }(); would attempt to IIFE the function definition and break the declaration.
C: sayHello(world)() would try to call the return value of sayHello, which is undefined, causing an error.
D: Changing world to a function declaration does not change the fact that it is passed as a function reference; sayHello still prints the function object, not 'World'.

---

123. Frage
Refer to thecode below:
Const pi = 3.1415326,
What is the data type of pi?

- A. Float
- B. Decimal
- C. Double
- D. Number

Antwort: D

124. Frage
Given code below:
setTimeout (()=> (
console.log(1);
). 0);
console.log(2);
New Promise ((resolve, reject)) = > (
setTimeout(() => (
reject(console.log(3));
). 1000);
)).catch(() => (
console.log(4);
));
console.log(5);
What is logged to the console?

- A. 1 2 43 5
- B. 2 1 4 3 5
- C. 1 2 5 3 4
- D. 2 5 1 3 4

Antwort: D

125. Frage
Which two code snippets show working examples of a recursive function?

- A. function factorial(numVar) {
  if (numVar < 0) return;
  if (numVar === 0) return 1;
  return numVar - 1;
  }
- B. const sumToTen = numVar => {
  if (numVar < 0)
  return;
  return sumToTen(numVar + 1);
  };
- C. let countingDown = function(startNumber) {
  if (startNumber > 0) {
  console.log(startNumber);

return countingDown(startNumber - 1);
} else {

- D. const factorial = numVar => {
  if (numVar < 0) return;
  if (numVar === 0) return 1;
  return numVar * factorial(numVar - 1);
  };

Antwort: C,D

Begründung:
return startNumber;
}
};
(Note: Option D is shown here with corrected syntax: lowercase return and matching parentheses.) Explanation:
Comprehensive and Detailed Explanation From Exact Extract JavaScript knowledge:
A recursive function is a function that calls itself and has a base case to terminate the recursion.
Evaluate each option:
Option A:
const sumToTen = numVar => {
if (numVar < 0)
return;
return sumToTen(numVar + 1);
};
This function calls itself: sumToTen(numVar + 1) - so it is recursive.
However, the base condition is if (numVar < 0) return;.
If you call sumToTen(0):
numVar < 0 is false, so it calls sumToTen(1), then sumToTen(2), and so on, incrementing forever.
There is no condition to stop the recursion when numVar increases; it will eventually cause a stack overflow.
This code does not represent a properly working recursive function with a valid termination for increasing values and is not a good example of correct recursion.
Option B:
function factorial(numVar) {
if (numVar < 0) return;
if (numVar === 0) return 1;
return numVar - 1;
}
This function does not call itself anywhere.
It has conditional returns, but there is no recursive call such as factorial(numVar - 1).
Therefore, it is not recursive at all.
Option C:
const factorial = numVar => {
if (numVar < 0) return;
if (numVar === 0) return 1;
return numVar * factorial(numVar - 1);
};
This is a classic recursive factorial implementation.
It calls itself with a smaller argument: factorial(numVar - 1).
Base cases:
If numVar < 0, it simply returns (could be treated as invalid input).
If numVar === 0, it returns 1, which is the mathematical definition of 0! (zero factorial).
For positive integers, it correctly multiplies numVar by factorial(numVar - 1) until it reaches the base case.
This is a correct and working recursive function.
Option D (corrected):
let countingDown = function(startNumber) {
if (startNumber > 0) {
console.log(startNumber);
return countingDown(startNumber - 1);
} else {
return startNumber;
}

};
This function also calls itself: countingDown(startNumber - 1).
Base case:
When startNumber is not greater than 0 (i.e., 0 or negative), it returns startNumber and stops recursing.
For example, countingDown(3) would:
Log 3, call countingDown(2)
Log 2, call countingDown(1)
Log 1, call countingDown(0)
At 0, it hits the else branch and returns 0, ending the recursion.
This is a valid working recursive function structure (once syntax is corrected).
Therefore, the snippets that show working recursive functions are:
Answe r: C, D
Study Guide / Concept Reference (no links):
Definition of recursion: a function calling itself
Base case vs recursive step
Recursive factorial implementation
Recursive countdown example
Importance of a terminating condition to avoid infinite recursion

126. Frage
myArraym can have one level, two levels, or more levels.
Which statement flattens myArray when it can be arbitrarily nested?

- A. myArray. reduce ((prev, curr) => prev.concat(curr) [ ]);
- B. [ ] .concat {. . .myArray) ;
- C. myArray.flat(Infinity);
- D. myArray. join (",").split (",");

Antwort: A

127. Frage
.....

Die Materialien zur Salesforce JS-Dev-101 Zertifizierungsprüfung von ExamFragen werden speziell von dem IT-Expertenteam entworfen. Sie sind zielgerichtet. Durch die Zertifizierung können Sie Ihren internationalen Wert in der IT-Branche verwirklichen. Viele Anbieter für Antwortenspeicherung und die Schulungsunterlagen versprechen, dass Sie die Salesforce JS-Dev-101 Zertifizierungsprüfung mit ihren Produkten bestehen können. ExamFragen sagen mit den Beweisen. Der Moment, wenn das Wunder vorkommt, kann jedes Wort von uns beweisen.

101 Testfagen

- JS-Dev-101 Schulungsunterlagen □ JS-Dev-101 Fragenkatalog □ JS-Dev-101 Tests □ Suchen Sie jetzt auf ☀ www.itzert.com □☀□ nach 「 JS-Dev-101 」 um den kostenlosen Download zu erhalten □JS-Dev-101 Prüfungen
- JS-Dev-101 Online Prüfung □ JS-Dev-101 PDF □ JS-Dev-101 Prüfungsfrage □ Sie müssen nur zu [www.zertsoft.com] gehen um nach kostenloser Download von ☀ JS-Dev-101 □☀□ zu suchen □JS-Dev-101 Prüfungs
- JS-Dev-101 Übungsfragen: Salesforce Certified JavaScript Developer - Multiple Choice - JS-Dev-101 Dateien Prüfungsunterlagen □ Erhalten Sie den kostenlosen Download von ➥ JS-Dev-101 □ mühelos über （ www.itzert.com ） □JS-Dev-101 Tests
- JS-Dev-101 PrüfungGuide, Salesforce JS-Dev-101 Zertifikat - Salesforce Certified JavaScript Developer - Multiple Choice □ Suchen Sie auf □ www.deutschpruefung.com □ nach kostenlosem Download von [JS-Dev-101] □JS-Dev-101 Schulungsangebot
- www.stes.tyc.edu.tw, www.stes.tyc.edu.tw, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, www.stes.tyc.edu.tw, learn.csisafety.com.au, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, www.stes.tyc.edu.tw, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, Disposable vapes

P.S. Kostenlose und neue JS-Dev-101 Prüfungsfragen sind auf Google Drive freigegeben von ExamFragen verfügbar: https://drive.google.com/open?id=1WU90qdpbL8GBXj6s0Ug6GUIgrmBdKwi-