Salesforce.JavaScript-Developer-I.v2026-07-07.q122

Exam Code:	JavaScript-Developer-I
Exam Name:	Salesforce Certified JavaScript Developer (JS-Dev-101)
Certification Provider:	Salesforce
Free Question Number:	122
Version:	v2026-07-07
# of views:	116
# of Questions views:	1220
https://www.exam-tests.com/JavaScript-Developer-I-exam/Salesforce.JavaScript-Developer-I.v2026-07-07.q122.html	

NEW QUESTION: 1
A developer has code that calculates a restaurant bill, but generates incorrect answers while testing the code:
function calculateBill (items) {
let total = 0;
total += findSubTotal(items);
total += addTax(total);
total += addTip(total);
return total;
}
Which option allows the developer to step into each function execution within calculateBill?
A. Wrapping findSubtotal in a console.log() method.
B. Calling the console.trace (total) method on line 03.
C. Using the debugger command on line 03
D. Using the debugger command on line 05.
Answer: D (LEAVE A REPLY)

NEW QUESTION: 2
A developer wants to use a module named universalContainerslib and then call functions from it. How should a developer import every function from the module and then call the functions foo and bar?
A. import * as lib from ' /path/universalContainerslib.js ' ;
lib.foo();
lib.bar();
B. import * from ' /path/universalContainerslib.js ' ;
universalContainerslib.foo();
universalContainerslib.bar();
C. import all from ' /path/universalContainerslib.js ' ;
universalContainerslib.foo();

universalContainerslib.bar();
D. import {foo, bar} from ' /path/universalContainerslib.js ' ;
foo();
bar();
Answer: A (LEAVE A REPLY)
* import * as lib from ' ... ' imports all named exports from the module into the namespace object lib.
* You then call:
lib.foo();
lib.bar();
Option D correctly imports only foo and bar, not every function. The question explicitly says "import every function... and then call foo and bar", so A best matches.
Options B and C are invalid syntax or reference the wrong identifier.

NEW QUESTION: 3
Given the JavaScript below:

```
01 function filterDOM(searchString){
02   const parsedSearchString = searchString && searchString.toLowerCase();
03   document.querySelectorAll('.account').forEach(account => {
04     const accountName = account.innerHTML.toLowerCase();
05     account.style.display = accountName.includes(parsedSearchString) ? /* Insert code here */;
06   });
07 }
```

Which code should replace the placeholder comment on line 06 to hide accounts that do not match the search string?
A. 'Block' : 'none'
B. 'Visible : 'hidden'
C. 'Hidden, visible
D. 'None' : 'block'
Answer: (SHOW ANSWER)

NEW QUESTION: 4
At Universal Containers, every team has its own way of copying JavaScript objects. The code Snippet shows an implementation from one team:
Function Person() {
this.firstName = "John";
this.lastName = 'Doe';
This.name =() => (
console.log('Hello $(this.firstName) $(this.firstName)');
)}
Const john = new Person ();
Const dan = JSON.parse(JSON.stringify(john));
dan.firstName ='Dan';

dan.name();
What is the Output of the code execution?
A. Hello Dan Doe
B. TypeError: Assignment to constant variable.
C. Hello John DOe
D. TypeError: dan.name is not a function
Answer: D (LEAVE A REPLY)

NEW QUESTION: 5
developer uses the code below to format a date.

```
01 const date = new Date(2020, 05, 10);
02 const dateDisplayOptions = {
03   year: 'numeric',
04   month: 'long',
05   day: 'numeric'
06 };
07
08 const formattedDate = date.toLocaleDateString('en', dateDisplayOptions);
```


After executing, what is the value offormattedDate?
A. November 05, 2020
B. June 10, 2020
C. May 10, 2020
D. October 05, 2020
Answer: B (LEAVE A REPLY)

NEW QUESTION: 6
developer creates a new web server that uses Node.js. It imports a server library that uses events and callbacks for handling server functionality.
The server library is imported with require and is made available to the code by a variable named server. The developer wants to log any issues that the server has while booting up.
Given the code and the information the developer has, which code logs an error at boost with an event?
A. Server.on ('error', (error) => {
console.log('ERROR', error);
});
B. Server.error ((server) => {
console.log('ERROR', error);
});
C. Server.catch ((server) => {
console.log('ERROR', error);
});
D. Try{
server.start();
} catch(error) {

console.log('ERROR', error);
}
Answer: (SHOW ANSWER)

NEW QUESTION: 7
A developer initiates a server with the file server,js and adds dependencies in the source codes package,json that are required to run the server.
Which command should the developer run to start the server locally?
A. start server,js
B. npm start server,js
C. node start
D. npm start
Answer: D (LEAVE A REPLY)

NEW QUESTION: 8
A developer wants to leverage a module to print a price in pretty format, and has imported a method as shown below:
Import printPrice from '/path/PricePrettyPrint.js';
Based on the code, what must be true about the printPrice function of the PricePrettyPrint module for this import to work ?
A. printPrice must be the default export
B. printPrice must be a multi exportc
C. printPrice must be be a named export
D. printPrice must be an all export
Answer: A (LEAVE A REPLY)

NEW QUESTION: 9
A developer executes:
document.cookie;
document.cookie = ' key=John Smith ' ;
What is the behavior?
A. Cookies are read, but the key value is not set because the value is not URL encoded.
B. Cookies are read and the key value is set, the remaining cookies are unaffected.
C. Cookies are read and the key value is set, and all cookies are wiped.
D. Cookies are not read because line 01 should be document.cookies, but the key value is set and all cookies are wiped.
Answer: B (LEAVE A REPLY)

NEW QUESTION: 10
A developer copied a JavaScript object:

```
01 function Person() {
02   this.firstName = "John";
03   this.lastName = "Doe";
04   this.name = () => `${this.firstName},${this.lastName}`;
05 }
06
07 const john = new Person();
08 const dan = Object.assign(john);
09 dan.firstName = 'Dan';
```

How does the developer access dan's forstName,lastName? Choose 2 answers

A. dan,name

B. dan,firstname ( ) + dan, lastName ( )

C. dan,name ( )

D. dan, firstName = dan.lastName

Answer: C,D (LEAVE A REPLY)

NEW QUESTION: 11

Given the code below:

```
01 setTimeout(() => {
02   console.log(1);
03 }, 1100);
04 console.log(2);
05 new Promise((resolve, reject) => {
06   setTimeout(() => {
07     reject(console.log(3));
08   }, 1000);
09 }).catch(() => {
10   console.log(4);
11 });
12 console.log(5);
```

What is logged to the console'

A. 1 2 5 3 4

B. 1 2 3 4 5

C. 2 5 1 3 4

D. 2 5 3 4 1

Answer: D (LEAVE A REPLY)

NEW QUESTION: 12
Refer to the following code:
function test (val) {
If (val === undefined) {
return 'Undefined values!' ;
}
if (val === null) {
return 'Null value! ';
}
return val;
}
Let x;
test(x);
What is returned by the function call on line 13?
A. Line 13 throws an error.
B. 'Undefined values!'
C. 'Null value!'
D. Undefined
Answer: (SHOW ANSWER)

NEW QUESTION: 13
Refer to the code below:
01 let country = {
02 get capital() {
03 let city = Number(" London ");
04
05 return {
06 cityString: city.toString(),
07 }
08 }
09 }
Which value can a developer expect when referencing country.capital.cityString?
A. An error
B. NaN
C. undefined
D. ' London '
Answer: (SHOW ANSWER)
* In the getter:
* let city = Number(" London ");

The Number() constructor attempts to convert the string " London " into a numeric value.
* " London " is not a valid numeric string. When JavaScript attempts numeric conversion of a non- numeric string:
* Number(" London ") # NaN
* Next line:
* city.toString()
When city is NaN, calling .toString() yields:
NaN.toString() # " NaN "
* The getter returns:
* {
* cityString: " NaN "
* }
* The question asks: Which value can a developer expect? The multiple-choice options include NaN , but not " NaN " .
Given the available choices, the intended correct answer is B (NaN) because the root cause is the Number(" London ") conversion returning NaN.
JavaScript Knowledge References (text-only)
* Number(nonNumericString) returns NaN.
* NaN.toString() produces " NaN " .
* Getters compute and return their values each time the property is accessed.

NEW QUESTION: 14
Refer to the code below:

```
01 let requestPromise = client.getRequest;
02
03 requestPromise().then((response) => {
04   handleResponse(response);
05 });
```

A developer uses a client that makes a GET request that uses a promise to handle the request, getRequest returns a promise.
Which code modification can the developer make to gracefully handle an error?
A)

```
01 let requestPromise = client.getRequest;
02
03 try {
04   requestPromise().then((response) => {
05     handleResponse(response);
06   }).then((error) => {
07     throw new Error(error.name);
08   });
09 } catch(error) {
10   handleError(error);
11 }
```

B)

```
01 let requestPromise = client.getRequest;
02
03 requestPromise().then((response) => {
04   handleResponse(response);
05 }).catch((error) => {
06   handleError(error);
07 });
```

C)

```
01 let requestPromise = client.getRequest;
02
03 try {
04   requestPromise().then((response) => {
05     handleResponse(response);
06   });
07 } catch(error) {
08   handleError(error);
09 }
```

D)

```
01 let requestPromise = client.getRequest;
02
03 requestPromise().then((response) => {
04   handleResponse(response);
05 }).finally((error) => {
06   handleError(error);
07 });
```

A. Option D

B. Option B

C. Option A

D. Option C

Answer: C (LEAVE A REPLY)

NEW QUESTION: 15

A developer wrote the following code to test a sum3 function that takes in an array of numbers and returns the sum of the first three numbers in the array. The test passes:

01 let res = sum3([1, 2, 3]);

02 console.assert(res === 6);

03

04 res = sum3([1, 2, 3, 4]);

05 console.assert(res === 6);

A different developer made changes to the behavior of sum3 to instead sum all of the numbers present in the array.

Which two results occur when running the test on the updated sum3 function?

A. The line 02 assertion fails.

B. The line 05 assertion passes.

C. The line 05 assertion fails.

D. The line 02 assertion passes.

Answer: (SHOW ANSWER)

New behavior: sum3 now returns the sum of all elements .

* Line 01: sum3([1, 2, 3]) # 1 + 2 + 3 = 6

* Assertion on line 02: res === 6 # passes.

* Line 04: sum3([1, 2, 3, 4]) # 1 + 2 + 3 + 4 = 10

* Assertion on line 05: res === 6 # 10 === 6 is false, so the assertion fails.

So:

* Line 02 assertion passes # D.

* Line 05 assertion fails # C.

NEW QUESTION: 16
Refer to the following code that imports a module named utils:
import (foo, bar) from '/path/Utils.js';
foo() ;
bar() ;
Which two implementations of Utils.js export foo and bar such that the code above runs without error?
Choose 2 answers
A. const foo = () => { return 'foo';}
const bar = () => {return 'bar'; }
Export default foo, bar;
B. const foo = () => { return 'foo' ; }
const bar = () => { return 'bar' ; }
export { bar, foo }
C. Export default class {
foo() { return 'foo' ; }
bar() { return 'bar' ; }
}
D. // FooUtils.js and BarUtils.js exist
Import (foo) from '/path/FooUtils.js';
Import (boo) from ' /path/NarUtils.js';
Answer: (SHOW ANSWER)

NEW QUESTION: 17
Given the following code:

```
01 let x = null;
02 console.log(typeof x);
```

is the output of line 02?
A. "undefined"
B. "object"
C. "null"'
D. "x"

Answer: B (LEAVE A REPLY)

NEW QUESTION: 18
Refer to the code below:
Let textvalue = '1984';
Which code segment shows a correct way to convert this string to an integer?
A. Let numberValue = Integer (textValue);
B. Let numberValue = (Number) textvalue;
C. Let numberValue = Number(textvalue);
D. Let numberValue = textValue. Tointeger ( ) ;
Answer: C (LEAVE A REPLY)

NEW QUESTION: 19
After user acceptance testing, the developer is asked to change the webpage background based on user's location. This change was implemented and deployed for testing.
The tester reports that the background is not changing, however it works as required when viewing on the developer's computer.
Which two actions will help determine accurate results?
Choose 2 answers
A. The tester should disable their browser cache.
B. The developer should inspect their browser refresh settings.
C. The developer should rework the code.
D. The tester should dear their browser cache.
Answer: B,D (LEAVE A REPLY)

NEW QUESTION: 20
Refer to the code below:

```
01 function changeValue(obj)
02   obj.value = obj.value * 2;
03 }
04 const objA = {value: 10};
05 const objB = objA;
06
07 changeValue(obj
08 const result =
```

What is the value of result after the code executes?
A. undefined
B. 5
C. 10
D. NaN
Answer: B (LEAVE A REPLY)

NEW QUESTION: 21
A developer needs to debug a Node.js web server because a runtime error keeps occurring at one of the endpoints.

The developer wants to test the endpoint on a local machine and make the request against a local server to look at the behavior. In the source code, the server.js file will start the server. The developer wants to debug the Node.js server only using the terminal.
Which command can the developer use to open the CLI debugger in their current terminal window?
(With corrected typing errors: node_inspect # node inspect, node_start_inspect # node start inspect.)
A. node inspect server.js
B. node server.js --inspect
C. node start inspect server.js
D. node -i server.js
Answer: A (LEAVE A REPLY)

NEW QUESTION: 22
Corrected code:
let a = " * " ;
let b = " ** " ;
// x = 3;
console.log(a);
What is displayed when the code executes?
A. ReferenceError: a is not defined
B. *
C. undefined
D. null
Answer: B (LEAVE A REPLY)
The correct answer is B because variable a is declared and initialized before it is printed.
let a = " * " ;
This creates a block-scoped variable named a and assigns it the string value:
" * "
The line:
// x = 3;
is a comment. JavaScript ignores comments during execution, so this line has no effect.
Then this line runs:
console.log(a);
Since a already exists and contains " * " , the console displays:
* Option A is incorrect because a is defined.
Option C is incorrect because a has an assigned value, so it is not undefined.
Option D is incorrect because a was never assigned null.
Therefore, the verified answer is B .

NEW QUESTION: 23
Given two expressions, exp1 and exp2, which two valid ways return the logical AND of the two expressions and ensure it is a Boolean?
A. Boolean(exp1 & & exp2)

B. Boolean(exp1) & & Boolean(exp2)
C. exp1 & exp2
D. Boolean(var1) & & Boolean(var2)
Answer: (SHOW ANSWER)
The correct answers are A and B .
The original question contains a typing error where 66 should be corrected to the logical AND operator:
& &
A is correct because it first evaluates the logical AND expression and then converts the final result to a Boolean:
Boolean(exp1 & & exp2)
The & & operator returns the first falsy value or the last truthy value. Wrapping the result in Boolean() ensures the final result is strictly either:
true
or:
false
Example:
Boolean(" hello " & & 123);
This returns:
true
B is also correct because it converts both expressions to Boolean values first, then applies logical AND:
Boolean(exp1) & & Boolean(exp2)
This guarantees that both sides of the & & operation are Boolean values.
Example:
Boolean(" hello ") & & Boolean(123);
This returns:
true
C is incorrect because it uses the bitwise AND operator:
&
This is not the same as logical AND:
& &
Bitwise AND converts values into numbers and compares their binary representation. It does not reliably return a Boolean logical result.
D is incorrect as written because the question gives the expressions as exp1 and exp2, but option D uses var1 and var2. After correction, it would be the same idea as option B.
Therefore, the verified answers are A and B .

NEW QUESTION: 24
Refer to the code:
01 const event = new CustomEvent(
02 // Missing code
03);
04 obj.dispatchEvent(event);

A developer needs to dispatch a custom event called update to send information about recordId.
Which two options can be inserted at line 02?
A. { type: ' update ' , recordId: ' 123abc ' }
B. ' update ' , { detail: { recordId: ' 123abc ' } }
C. ' update ' , ' 123abc '
D. ' update ' , { recordId: ' 123abc ' }
Answer: B,D (LEAVE A REPLY)
The correct constructor signature for CustomEvent is:
new CustomEvent(eventName, optionsObject)
Where:
* eventName is a string.
* optionsObject may include:
* detail # used to pass custom data
* bubbles
* cancelable, etc.
Example:
new CustomEvent(' update ' , {
detail: { recordId: ' 123abc ' }
});
Now evaluate each option:
Option A
{ type: ' update ' , recordId: ' 123abc ' }
Incorrect: The constructor requires (eventName, options), not a single object. type is not used this way.
Option B
' update ' , { detail: { recordId: ' 123abc ' } }
Correct format. detail is the proper place for custom event data.
Option C
' update ' , ' 123abc '
Incorrect: The second argument must be an object (options), not a string.
Option D
' update ' , { recordId: ' 123abc ' }
Acceptable because any extra properties on the options object are still allowed, even though best practice is to use detail.
This still creates a valid CustomEvent, and the event will dispatch successfully.
Thus the two correct answers are B and D .
JavaScript Knowledge References (text-only)
* new CustomEvent(name, options) is the required syntax.
* The detail property of the options object is the standard location for custom data.
* The second argument must be an object; other types are invalid.

NEW QUESTION: 25
Refer to code below:

Function muFunction(reassign){
Let x = 1;
var y = 1;
if(reassign) {
Let x= 2;
Var y = 2;
console.log(x);
console.log(y);}
console.log(x);
console.log(y);}
What is displayed when myFunction(true) is called?
A. 2 2 1 1
B. 2 2 1 2
C. 2 2 2 2
D. 2 2 undefined undefined
Answer: B (LEAVE A REPLY)

NEW QUESTION: 26
Refer of the string below:
Const str = 'sa;esforce'=;
Which two statement result in the word 'Sale'?
Choose 2 answers
A. str, substr(0,5) ;
B. str, substring(1,5) ;
C. str, substring (0,5) ;
D. str, substr(1,5) ;
Answer: A,C (LEAVE A REPLY)

NEW QUESTION: 27
Which two implementations of utils.js support foo and bar?
A. const foo = () = > { return ' foo ' ; };
const bar = () = > { return ' bar ' ; };
export { foo, bar };
B. const foo = () = > { return ' foo ' ; };
const bar = () = > { return ' bar ' ; };
export default { foo, bar };
C. import { foo, bar } from " ./helpers/utils.js " ;
export { foo, bar };
D. export default class {
foo() { return ' foo ' ; }
bar() { return ' bar ' ; }

}
Answer: A,C (LEAVE A REPLY)
The correct answers are A and C because both use named exports for foo and bar.
A module can export multiple values by name:
export { foo, bar };
This allows another file to import them like this:
import { foo, bar } from ' ./utils.js ' ;
Option A is correct because it defines both functions locally and exports both of them as named exports.
const foo = () = > { return ' foo ' ; };
const bar = () = > { return ' bar ' ; };
export { foo, bar };
Option C is also correct after correcting the typing issue so that both foo and bar are imported first:
import { foo, bar } from " ./helpers/utils.js " ;
export { foo, bar };
This is called re-exporting. The module imports foo and bar from another module, then exports them again from utils.js.
Option B is not the best answer for named access because it exports one default object:
export default { foo, bar };
That would require a default import first:
import utils from ' ./utils.js ' ;
utils.foo();
utils.bar();
It does not directly support named imports in the same way as:
import { foo, bar } from ' ./utils.js ' ;
Option D exports a default class. Its foo() and bar() are class instance methods, not named exported functions.
It would require usage like:
import Utils from ' ./utils.js ' ;
const utils = new Utils();
utils.foo();
utils.bar();
So for direct multiple function exports, the correct implementations are A and C .

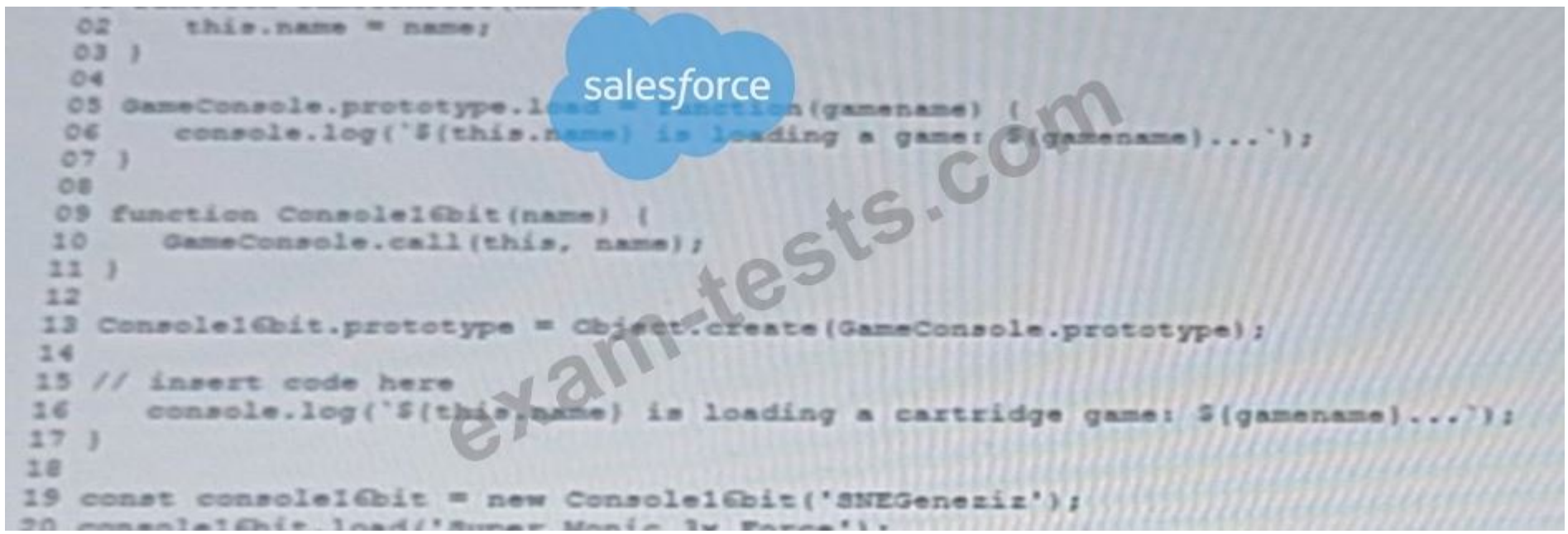
NEW QUESTION: 28
Refer to following code:
class Vehicle {
constructor(plate) {
This.plate =plate;
}
}
Class Truck extends Vehicle {
constructor(plate, weight) {
//Missing code

This.weight = weight;
}
displayWeight() {
console.log('The truck ${this.plate} has a weight of ${this.weight} lb.');}} Let myTruck = new Truck('123AB', 5000);
myTruck.displayWeight(); Which statement should be added to line 09 for the code to display 'The truck 123AB has a weight of 5000lb.'?
A. Vehicle.plate = plate;
B. super(plate);
C. Super.plate =plate;
D. This.plate =plate;
Answer: (SHOW ANSWER)

NEW QUESTION: 29
Refer to the code below:
01 let first = ' Who ' ;
02 let second = ' What ' ;
03 try {
04 try {
05 throw new Error(' Sad trombone ');
06 } catch (err) {
07 first = ' Why ' ;
08 throw err;
09 } finally {
10 second = ' When ' ;
11 }
12 } catch (err) {
13 second = ' Where ' ;
14 }
What are the values for first and second once the code executes?
A. first is Who and second is Where.
B. first is Why and second is Where.
C. first is Who and second is When.
D. first is Why and second is When.
Answer: (SHOW ANSWER)
Initial values:
* first = ' Who '
* second = ' What '
Execution:
Inner try/catch/finally:
* Line 05: throw new Error(' Sad trombone ');
* Control goes to the inner catch.

* Inner catch (lines 06-08):
catch (err) {
first = ' Why ' ;
throw err;
}
* first is set to ' Why ' .
* The error is rethrown.
* Inner finally (lines 09-11):
finally {
second = ' When ' ;
}
* finally runs whether or not there was an error.
* second becomes ' When ' .
After inner finally, the rethrown error continues to propagate to the outer catch.
Outer catch (lines 12-14):
} catch (err) {
second = ' Where ' ;
}
* Because the inner try rethrew, the outer catch runs.
* It sets second = ' Where ' .
Final values:
* first was changed to ' Why ' in the inner catch and never changed again.
* second became ' When ' in the inner finally, and then ' Where ' in the outer catch.
So:
* first is ' Why '
* second is ' Where '
Option B is correct.
Concepts: nested try/catch/finally, rethrowing errors, order of execution for catch vs finally, variable mutation through error propagation.

NEW QUESTION: 30
Given the code block below:



What should a developer insert line 15 to output the following message using the load method?

SNENneziz is loading a cartridge game: super Monic 3x Force...

A. Console16bit = object. Create (GameConsole. Prototype). Load _ function (gamename) (

B. Console116bit. Prototype. Load (gamename) = function () (

C. Console16bit. Prototype. Load = function (gamename) (

D. Console16bit. Prototype. Load (gamename) (

Answer: C (LEAVE A REPLY)

NEW QUESTION: 31

Corrected code:

function Person() {

this.firstName = " John " ;

}

Person.prototype = {

job: x = > " Developer "

};

const myFather = new Person();

const result = myFather.firstName + " " + myFather.job();

What is the value of result after line 10 executes?

A. " John Developer "

B. " John undefined "

C. Error: myFather.job is not a function

D. " undefined Developer "

Answer: A (LEAVE A REPLY)

The verified answer is A .

This constructor function runs when new Person() is called:

function Person() {

this.firstName = " John " ;

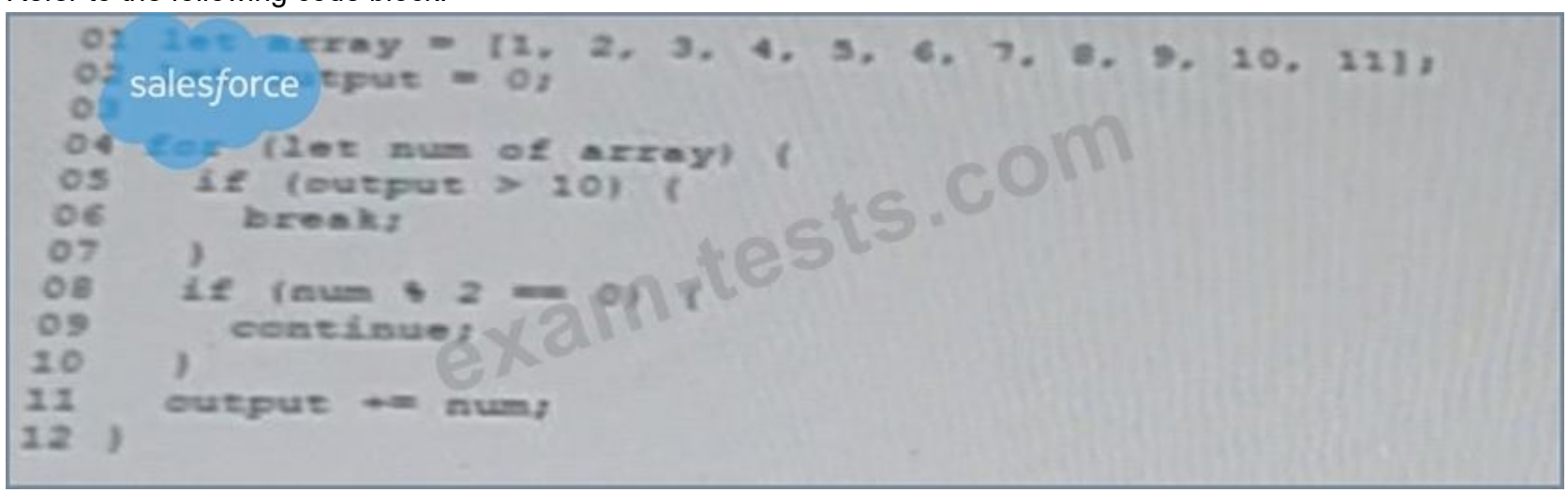
}
So this line:
const myFather = new Person();
creates an object like this:
{
firstName: " John "
}
Then the prototype is assigned a method named job:
Person.prototype = {
job: x = > " Developer "
};
Because myFather is created from Person, it can access properties and methods from Person.prototype.
So:
myFather.firstName
returns:
" John "
And:
myFather.job()
returns:
" Developer "
Therefore:
const result = myFather.firstName + " " + myFather.job();
becomes:
const result = " John " + " " + " Developer " ;
Final value:
" John Developer "
Option B is incorrect because job() returns " Developer " , not undefined.
Option C is incorrect because job exists on Person.prototype, so it is callable.
Option D is incorrect because firstName is assigned inside the constructor.
Therefore, the verified answer is A .

NEW QUESTION: 32
A developer is leading the creation of a new web server for their team that will fulfill API requests from an existing client.

The team wants a web server that runs on Node.Js, and they want to use thenew web framework Minimalist.Js.
The lead developer wants to advocate for a more seasoned back-end framework that already has a community around it.
Which two frameworks could the lead developer advocate for?
Choose 2 answers
A. Angular
B. Gatsby
C. Express
D. Koa
Answer: (SHOW ANSWER)

NEW QUESTION: 33
A developer receives a comment from the Tech Lead that the code given below has
error:
```
const monthName = 'July';
const year = 2019;
if(year === 2019) {
monthName = 'June';
}
```
Which line edit should be made to make this code run?
A. 02 let year =2019;
B. 03 if (year == 2019) {
C. 02 const year = 2020;
D. 01 let monthName ='July';
Answer: D (LEAVE A REPLY)

NEW QUESTION: 34
Refer to the following code block:

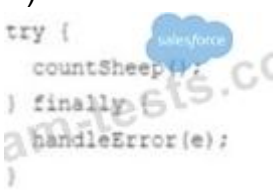


What is the value of output after the code executes?
A. 25
B. 16

C. 11
D. 36
Answer: B (LEAVE A REPLY)

NEW QUESTION: 35
A developer wants to use a try...catch statement to catch any error that countSheep () may throw and pass it to a handleError () function.
What is the correct implementation of the try...catch?
A)

B)



C)

D)

A. Option
B. Option
C. Option
D. Option
Answer: B (LEAVE A REPLY)

NEW QUESTION: 36
A developer wrote the following code to test a sum3 function that takes in an array of numbers and returns the sum of the first three numbers in the array, and the test passes.
A different developer made changes to the behavior of sum3 to instead sum only the first two numbers present in the array.

Which two results occur when running this test on the updated sum3 function?

Choose 2 answers

A. The line 05 assertion fails.

B. The line 02 assertion passes.

C. The line 05 assertion passes.

D. Theline 02 assertion fails.

Answer: (SHOW ANSWER)

NEW QUESTION: 37

Given two expressions var1 and var2. What are two valid ways to return the logical AND of the two expressions and ensure it is data type Boolean ?

Choose 2 answers:

A. var1 && var2

B. Boolean(var1 && var2)

C. Boolean(var1) && Boolean(var2)

D. var1.toBoolean() && var2toBoolean()

Answer: B,C (LEAVE A REPLY)

NEW QUESTION: 38

Given the code below:

Which three code segments result in a correct conversion from number to string? Choose 3 answers

A. let strValue = * * 4 numValue;

B. let strValue = numValue. toString();

C. let strValue = numValue.toText ();

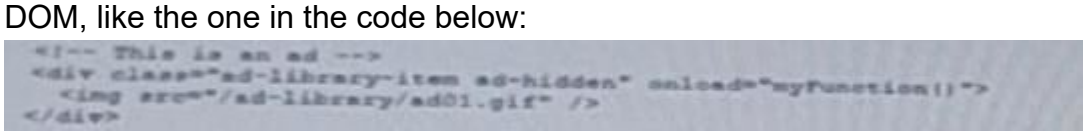
D. let scrValue = String(numValue);

E. let strValue = (String)numValue;

Answer: A,B,D (LEAVE A REPLY)

NEW QUESTION: 39

Universal Containers recently launched its new landing page to host a crowd-funding campaign. The page uses an external library to display some third-party ads. Once the page is fully loaded, it creates more than 50 new HTML items placed randomly inside the DOM, like the one in the code below:



All the elements includes the same ad-library-item class, They are hidden by default, and they are randomly displayed while the user navigates through the page.

A. Use the DOM inspector to remove all the elements containing the class ad-library-item.

B. Use the browser to execute a script that removes all the element containing the class ad-library-item.
C. Use the DOM inspector to prevent the load event to be fired.
D. Use the browser console to execute a script that prevents the load event to be fired.
Answer: B (LEAVE A REPLY)

NEW QUESTION: 40
A developer at Universal Containers is creating their new landing page based on HTML, CSS, and JavaScript. The website includes multiple external resources that are loaded when the page is opened.
To ensure that visitors have a good experience, a script named personalizeWebsiteContent needs to be executed when the webpage Is loaded and there Is no need to wait for the resources to be available.
Which statement should be used to call personalizeWebsiteContent based on the above business requirement?
A. windows,addEventListener('onDOMCContentLoaded', personalizeWebsiteContent);
B. windows,addEventListener('DOMContent Loaded ', personalizeWebsiteContent);
C. windows,addEventListener('onload', personalizeWebsiteContent);
D. windows,addEventListener('load', personalizeWebsiteContent);
Answer: D (LEAVE A REPLY)

NEW QUESTION: 41
A developer wants to set up a secure web server with Node.js. The developer creates a directory locally called app-server, and the first file is app-server/index.js Without using any third-party libraries, what should the developer add to index.js to create the secure web server?
A. const https =require('https');
B. const http =require('http');
C. const server =require('secure-server');
D. const tls = require('tls');
Answer: A (LEAVE A REPLY)

NEW QUESTION: 42
developer is trying to convince management that their team will benefit from using Node.js for a backend server that they are going to create. The server will be a web server that handles API requests from a website that the team has already built using HTML, CSS, and JavaScript.
Which three benefits of Node.js can the developer use to persuade their manager?
Choose 3 answers:
A. Executes server-side JavaScript code to avoid learning a new language.
B. Installs with its own package manager to install and manage third-party libraries.
C. Uses non-blocking functionality for performant request handling .
D. Ensures stability with one major release every few years.
E. Performs a static analysis on code before execution to look for runtime errors.
Answer: B,C,E (LEAVE A REPLY)

NEW QUESTION: 43

A developer creates a simple webpage with an input field. When a user enters text in the input field and clicks the button, the actual value of the field must be displayed in the console.
Here is the HTML file content:
<input type =" text" value="Hello" name ="input">
<button type ="button" >Display </button>
The developer wrote the javascript code below:
Const button = document.querySelector('button');
button.addEvenListener('click', () => (
Const input = document.querySelector('input');
console.log(input.getAttribute('value'));
When the user clicks the button, the output is always "Hello".
What needs to be done make this code work as expected?
A. Replace line 04 with console.log(input .value);
B. Replace line 02 with button.addCallback("click", function() {
C. Replace line 02 with button.addEventListener("onclick", function() {
D. Replace line 03 with const input = document.getElementByName('input');
Answer: A (LEAVE A REPLY)

NEW QUESTION: 44
Which javascript methods can be used to serialize an object into a string and deserialize a JSON string into an object, respectively?
A. JSON.stringify and JSON.parse
B. JSON.encode and JSON.decode
C. JSON.serialize and JSON.deserialize
D. JSON.parse and JSON.deserialize
Answer: (SHOW ANSWER)

NEW QUESTION: 45
myArraym can have one level, two levels, or more levels.
Which statement flattens myArray when it can be arbitrarily nested?
A. myArray. join (","). split (",");
B. [ ] .concat {. . .myArray) ;
C. myArray.flat(Infinity);
D. myArray. reduce ((prev, curr) => prev.concat(curr) [ ]);
Answer: D (LEAVE A REPLY)

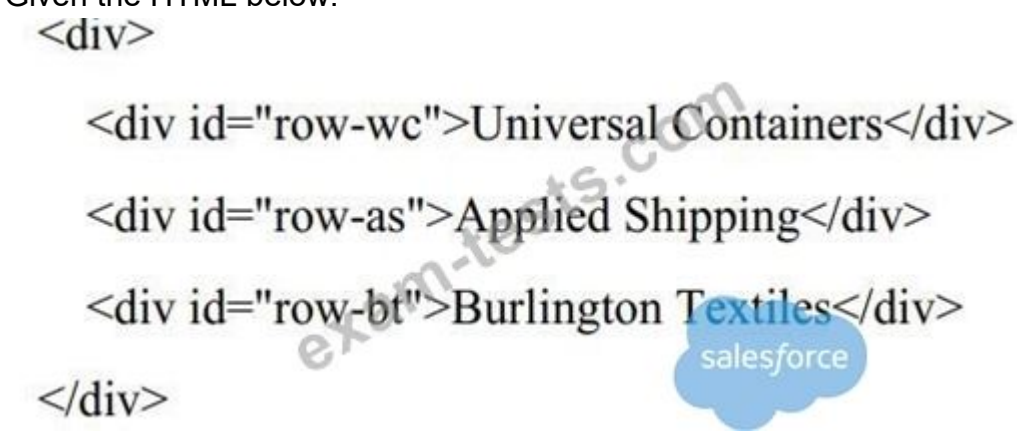
NEW QUESTION: 46
A developer has a web server running with Node.js. The command to start the web server is node server.js.
The web server started having
latency issues. Instead of a one second turnaround for web requests, the developer now sees a five second turnaround.
Which command can the web developer run to see what the module is doing during the latency period?

A. DEBUG=http, https node server.js
B. NODE_DEBUG=http,https node server.js
C. DEBUG=true node server.js
D. NODE_DEBUG=true node server.js
Answer: C (LEAVE A REPLY)

NEW QUESTION: 47
Refer to the code below:

What is the value of result when the code executes?
A. 5-10
B. 10-5
C. 10-10
D. 5-5
Answer: C (LEAVE A REPLY)

NEW QUESTION: 48
Given the HTML below:



Which statement adds the priority-account css class to the Applied Shipping row?

Answer:
document.querySelector('#row-as').classList.add('priority-account');

NEW QUESTION: 49
Given the code below:
01 setCurrentUrl();
02 console.log(" The current URL is: " + url);
03
04 function setCurrentUrl() {
05 url = window.location.href;
06 }
What happens when the code executes?
A. The url variable has global scope and line 02 throws an error.
B. The url variable has global scope and line 02 executes correctly.
C. The url variable has local scope and line 02 executes correctly.
D. The url variable has local scope and line 02 throws an error.
Answer: B (LEAVE A REPLY)
* Inside setCurrentUrl, url is assigned without var, let, or const :
url = window.location.href;
* In non-strict mode, this implicitly creates a global variable url on window.
Execution order:
* setCurrentUrl(); creates/sets global url to window.location.href.
* console.log(" The current URL is: " + url); has access to url in global scope and prints correctly.
Thus:
* url has global scope.
* Line 02 runs without error and logs a valid string.
So B is correct.

NEW QUESTION: 50
Refer to the following code block:

```
01 class Student {
02   constructor(name) {
03     this.name = name;
04   }
05
06   takeTest() {
07     console.log('${this.name} got 70% on test.');
08   }
09 }
10
11 class BetterStudent extends Student {
12   constructor(name) {
13       super(name);
14       this.name = 'Better student ' + name;
15   }
16   takeTest() {
17     console.log('${this.name} got 100% on test.');
18   }
19 }
20
21 let student = new BetterStudent('Jackie');
22 student.takeTest();
```


What is the console output?

A. > Uncaught Reference Error

B. > Better student Jackie got 100% on test.

C. > Jackie got 70% on test.

D. > Better student Jackie got 70% on test.

Answer: B (LEAVE A REPLY)

NEW QUESTION: 51

Which statement parses successfully?

A. JSON.parse ("foo");

B. JSON.parse ("foo");

C. JSON.parse (""foo'");

D. JSON. parse (""foo'");

Answer: D (LEAVE A REPLY)

NEW QUESTION: 52

A developer is creating a simple webpage with a button. When a user clicks this button for the first time, a message is displayed. The developer wrote the JavaScript code below, but something is missing. The message gets displayed every time a user clicks the button, instead of just the first time.

```
01 function listen(event) {
02
03 alert( ' Hey! I am John Doe ' );
04
```

05 }
06 button.addEventListener(' click ' , listen);
Which two code lines make this code work as required?
A. On line 04, use event.stopPropagation();
B. On line 02, use event.first to test if it is the first execution.
C. On line 06, add an option called once to button.addEventListener().
D. On line 04, use button.removeEventListener(' click ' , listen);
Answer: C,D (LEAVE A REPLY)
Requirement:
* The message should be displayed only on the first click of the button.
Original behavior:
* The handler listen is attached with button.addEventListener(' click ' , listen);.
* That means listen is called on every click until the listener is removed or configured otherwise.
Two correct ways to ensure the listener only fires once:
* Use the once option in addEventListener
* Remove the event listener after the first run inside the handler
Option C:
On line 06, add an option called once to button.addEventListener().
This means modifying line 06 to:
button.addEventListener(' click ' , listen, { once: true });
The once: true option tells the browser:
* Call the listen function at most once.
* After it is called the first time, automatically remove the listener.
So:
* First click: alert shows, listener is removed automatically.
* Subsequent clicks: no further calls to listen, no alerts.
This satisfies the requirement.
Option D:
On line 04, use button.removeEventListener(' click ' , listen);
This means updating the handler:
function listen(event) {
alert(' Hey! I am John Doe ');
button.removeEventListener(' click ' , listen);
}
Now:
* On the first click, listen is executed:
* It shows the alert.
* It removes itself from the button's click listeners.
* On subsequent clicks, listen is no longer registered, so nothing happens.
This also satisfies the requirement.
Why A and B are incorrect:

Option A:
On line 04, use event.stopPropagation();
* event.stopPropagation() stops the event from bubbling up the DOM tree.
* It does not prevent the current listener from being called again in the future.
* The click handler will still run on every click; it just prevents other listeners higher in the DOM from receiving the event.
Option B:
On line 02, use event.first to test if it is the first execution.
* There is no built-in event.first property in standard DOM events.
* This property does not exist and will be undefined.
* You would need your own external flag (let hasRun = false;) to track first execution, but that is not what B describes.
Therefore, the two correct modifications are:
The answer: C, D
Study Guide / Concept References (no links):
* addEventListener options object: { once: true }
* removeEventListener to manually deregister event handlers
* Event propagation (event.stopPropagation) vs handler lifecycle
* DOM event model and listener registration

NEW QUESTION: 53
A developer has an ErrorHandler module that contains multiple functions.
What kind of export be leverages so that multiple functions can be used?
A. Multi
B. All
C. Named
D. Default
Answer: C (LEAVE A REPLY)

NEW QUESTION: 54
function myFunction() {
a = a + b;
var b = 1;
}
myFunction();
console.log(a);
console.log(b);
Which statement is correct?
A. Line 02 throws a reference error, therefore line 03 is never executed.
B. Both line 02 and 03 are executed, but the values printed are undefined.
C. Both line 02 and 03 are executed, and the variables are hoisted.
D. Line 08 outputs the variable, but line 09 throws an error.
Answer: A (LEAVE A REPLY)

The correct answer is A , not D.
Inside myFunction(), the declaration:
var b = 1;
is hoisted to the top of the function scope, but only the declaration is hoisted, not the assignment. So internally, JavaScript treats the function approximately like this:
function myFunction() {
var b;
a = a + b;
b = 1;
}
Now examine this line:
a = a + b;
Before JavaScript can assign a value to a, it must first evaluate the right-hand side:
a + b
At this point:
b
exists locally because var b was hoisted, so its value is:
undefined
However:
a
has not been declared anywhere. Reading an undeclared variable causes a:
ReferenceError
So this line throws an error before assignment happens:
a = a + b;
Because the error occurs on that line, the next line inside the function is never executed:
var b = 1;
Also, because the error is not handled with try...catch, the program stops before reaching:
console.log(a);
console.log(b);
Therefore, the accurate statement is:
Line 02 throws a reference error, therefore line 03 is never executed.
So the verified answer is A .

NEW QUESTION: 55
Refer to the code below:
Function changeValue(obj) {
Obj.value = obj.value/2;
}
Const objA = (value: 10);
Const objB = objA;
changeValue(objB);

Const result = objA.value;
What is the value of result after the code executes?
A. Undefined
B. 10
C. Nan
D. 5
Answer: D (LEAVE A REPLY)

NEW QUESTION: 56
A developer executes:
document.cookie;
document.cookie = ' key=John Smith ' ;
What is the behavior?
A. Cookies are read and the key value is set, the remaining cookies are unaffected.
B. Cookies are read, but the key value is not set because the value is not URL encoded.
C. Cookies are not read because line 01 should be document.cookies, but the key value is set and all cookies are wiped.
D. Cookies are read and the key value is set, and all cookies are wiped.
Answer: A (LEAVE A REPLY)
* document.cookie retrieves the cookie string.
* Setting document.cookie = ' key=John Smith ' adds or updates only that one cookie.
Important details:
* Writing to document.cookie does not overwrite all cookies .
* The browser does not require URL encoding , though it is recommended. Non-encoded spaces are allowed and automatically handled.
* document.cookies does not exist; the correct property is document.cookie.
Therefore, the correct behavior:
* Cookies are read.
* A new cookie key=John Smith is set.
* Existing cookies remain.
JavaScript Knowledge References (text-only)
* Writing document.cookie appends or updates cookies, not replaces them.
* document.cookie is both getter and setter for cookie strings.

NEW QUESTION: 57
Cloud Kicks has a class to represent items for sale in an online store, as shown below:
Class Item{
constructor (name, price){
this.name = name;
this.price = price;
}
formattedPrice(){

return 's' + String(this.price);}}
A new business requirement comes in that requests a ClothingItem class that should have all of the properties and methods of the Item class but will also have properties that are specific to clothes.
Which line of code properly declares the clothingItem class such that it inherits from Item?
A. Class ClothingItem implements Item{
B. Class ClothingItem extends Item {
C. Class ClothingItem {
D. Class ClothingItem super Item {
Answer: B (LEAVE A REPLY)

NEW QUESTION: 58
Given the code below:
```
const delay = async delay =>{
return new Promise((resolve,reject)=>{
console.log(1);
setTimeout(resolve,deleay);
});
};
const callDelay = async ()=>{
console.log(2);
const yup = await delay(1000);
console.log(3);
}
console.log(4);
callDelay();
console.log(5);
```
What is logged to the console?
A. 4 2 1 5 3
Answer: A (LEAVE A REPLY)

NEW QUESTION: 59
Refer to the following code block (with corrected template literals using backticks):
```
01 class Animal {
02 constructor(name) {
03 this.name = name;
04 }
05
06 makeSound() {
07 console.log(`${this.name} is making a sound.`);
08 }
09 }
```

```
10
11 class Dog extends Animal {
12 constructor(name) {
13 super(name);
14 this.name = name;
15 }
16 makeSound() {
17 console.log(`${this.name} is barking.`);
18 }
19 }
20
21 let myDog = new Dog( ' Puppy ' );
22 myDog.makeSound();
```

What is the console output?

A. > Uncaught ReferenceError

B. > Undefined

C. > Puppy is barking.

Answer: (SHOW ANSWER)

* Animal class:
* Has a constructor that sets this.name.
* Has a makeSound method that logs a message using this.name.
* Dog class:
* Extends Animal.
* Its constructor calls super(name) to run the parent constructor, which sets this.name.
* It overrides makeSound() to log:
* console.log(`${this.name} is barking.`);
* Instantiation:

```
let myDog = new Dog( ' Puppy ' );
myDog.makeSound();
```

* new Dog(' Puppy '):
* Calls Dog constructor with name = ' Puppy ' .
* Inside, super(name) sets this.name = ' Puppy ' in the Animal constructor.
* Then this.name = name; in the Dog constructor keeps it as ' Puppy ' .
* myDog.makeSound():
* Calls the overridden makeSound() in Dog.
* Logs: " Puppy is barking. "

No reference errors, no undefined; the output is:

Puppy is barking.

Study Guide Concepts:

* ES6 classes and inheritance (extends)
* super() in subclass constructors

* Method overriding in subclasses
* Template literals: `${this.name} ...`

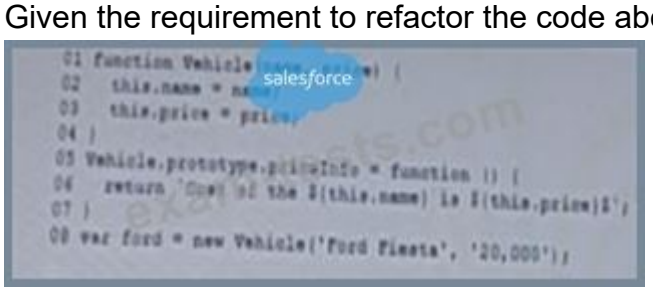
NEW QUESTION: 60
A developer needs to debug a Node.js web server because a runtime error keeps occurring at one of the endpoints.
The developer wants to test the endpoint on a local machine and make the request against a local server to look at the behavior. In the source code, the server, js file will start the server. the developer wants to debug the Node.js server only using the terminal.
Which command can the developer use to open the CLI debugger in their current terminal window?
A. node -i server.js
B. node start inspect server,js
C. node inspect server,js
D. node server,js inspect
Answer: C (LEAVE A REPLY)

NEW QUESTION: 61
Refer to the code below:
console.log("start);
Promise.resolve('Success') .then(function(value){
console.log('Success');
});
console.log('End');
What is the output after the code executes successfully?
A. Success
Start
End
B. Start
End
Success
C. Start
Success
End
D. End
Start
Success
Answer: (SHOW ANSWER)

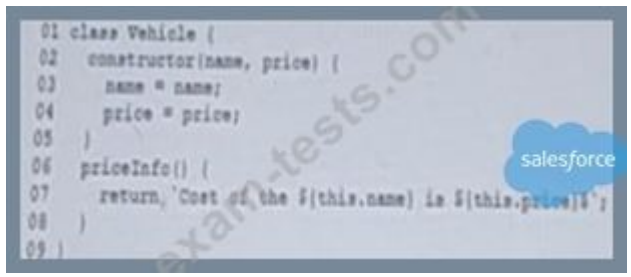
NEW QUESTION: 62
Given the code below:
Setcurrent URL ();
console.log('The current URL is: ' +url);
function setCurrentUrl() {
Url = window.location.href:
What happens when the code executes?
A. The url variable has local scope and line 02 executes correctly.
B. The url variable has local scope and line 02 throws an error.
C. The url variable has global scope and line 02 throws an error.
D. The url variable has global scope and line 02 executes correctly.
Answer: D (LEAVE A REPLY)

NEW QUESTION: 63
Which code statement below correctly persists an object in localStorage?
A. const setLocalstorage = (storageKey , jsobject) => (
windows. LocalStorage,setitem(storagekey. jSON,string
)
B. Const setLocalstorage = (jsobject) => (
Window .localstorage .setitem (jsobject) ;
C. Const setlocalstorage = (storagekey, jsObject) => (
Window. Localstorage.persist (storagekey, jsObject);
)
D. const setLocalstorage = (jsobject) =>
Windows.localstorage.connectobject(jsObject);
Answer: D (LEAVE A REPLY)

NEW QUESTION: 64
Given the requirement to refactor the code above to JavaScript class format, which class definition is correct?

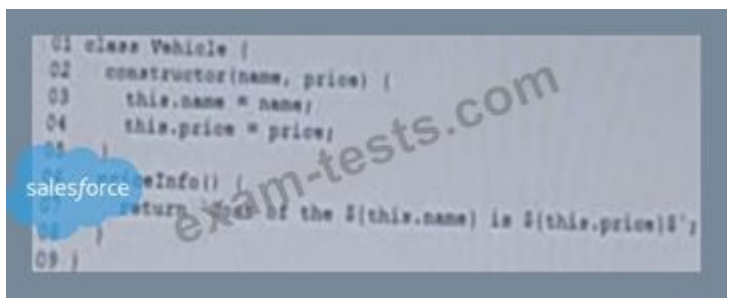


A.



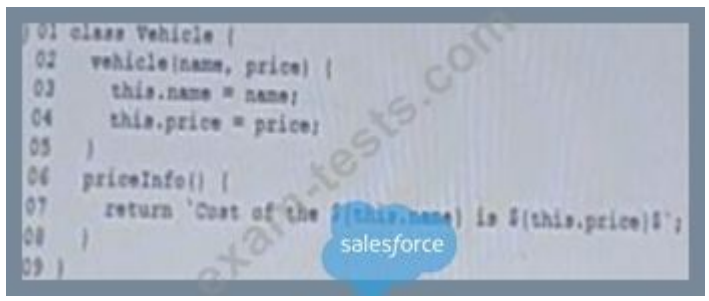
```
01 class Vehicle {
02   constructor(name, price) {
03     name = name;
04     price = price;
05   }
06   priceInfo() {
07     return `Cost of the ${this.name} is ${this.price}$`;
08   }
09 }
```

B.



```
01 class Vehicle {
02   constructor(name, price) {
03     this.name = name;
04     this.price = price;
05   }
06   priceInfo() {
07     return `Cost of the ${this.name} is ${this.price}$`;
08   }
09 }
```

C.



```
01 class Vehicle {
02   vehicle(name, price) {
03     this.name = name;
04     this.price = price;
05   }
06   priceInfo() {
07     return `Cost of the ${this.name} is ${this.price}$`;
08   }
09 }
```

D.

```
01 class Vehicle {
02   constructor(name, price) {
03     this.name = name;
04     this.price = price;
05   }
06   priceInfo() {
07     return `Cost of the ${this.name} is ${this.price}$`;
08   }
09 }
```

Answer: B (LEAVE A REPLY)

NEW QUESTION: 65

Refer to the code below:

Let car1 = new Promise((_ , reject) =>

setTimeout(reject, 2000, "car 1 crashed in" =>

Let car2 =new Promise(resolve => setTimeout(resolve, 1500, "car 2 completed") Let car3 =new Promise(resolve =>

setTimeout(resolve, 3000, "car 3 completed") Promise.race((car1, car2, car3))

.then (value => (

Let result = '$(value) the race.';)}

.catch(arr => {

console.log("Race is cancelled.", err);

});

What is the value of result when Promise.race executes?

A. Car 3 completes the race

B. Race is cancelled.

C. Car 1 crashed in the race.

D. Car 2 completed the race.

Answer: (SHOW ANSWER)

NEW QUESTION: 66

A developer writes the code below to calculate the factorial of a given number.

```
01 function factorial(number) {
02   return number * factorial(number - 1);
03 }
04 factorial(3);
```

What is the result of executing line 04?

A. 0

B. RuntimeError

C. 6

D. -Infinity

Answer: B (LEAVE A REPLY)

NEW QUESTION: 67

Refer to the code below:

01 const exec = (item, delay) =>{

02 new Promise(resolve => setTimeout(() => resolve(item), delay)),

03 async function runParallel() {

04 Const (result1, result2, result3) = await Promise.all{

05 [exec ('x', '100') , exec('y', 500), exec('z', '100')]

06);

07 return `parallel is done: $(result1) $(result2)$(result3)`;

08 }

}

}

Which two statements correctly execute the runParallel () function?

Choose 2 answers

A. runParallel () .then(function(data)

return da

B. Async runParallel () .then(data);

C. runParallel ( ). done(function(data){

return data;

});

D. runParallel () .then(data);

Answer: (SHOW ANSWER)

NEW QUESTION: 68

A developer wants to define a function log to be used a few times on a single-file JavaScript script.

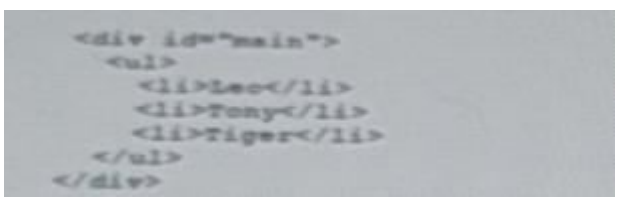
01 // Line 1 replacement
02 console.log('"LOG:', logInput);
03 }
Which two options can correctly replace line 01 and declare the function for use?
Choose 2 answers
A. const log(loginInput) {
B. const log = (logInput) => {
C. function log = (logInput) {
D. function leg(logInput) {
Answer: B,D (LEAVE A REPLY)

NEW QUESTION: 69
Given the JavaScript below:
function onLoad() {
console.log(" Page has loaded! ");
}
Where can the developer see the log statement after loading the page in the browser?
A. On the browser JavaScript console
B. On the terminal console running the web server
C. In the browser performance tools log
D. On the webpage console log
Answer: A (LEAVE A REPLY)
console.log() in browser-side JavaScript writes output to the browser's JavaScript console , which is available in the browser's Developer Tools.
* In a typical browser (Chrome, Firefox, Edge, etc.), you open DevTools and go to the Console tab.
* Any console.log(" Page has loaded! "); executed in page JavaScript will appear there.
Why A is correct:
* The function onLoad is a client-side function (runs in the browser).
* When it executes, the console.log call goes to the browser's JavaScript console , not the server.
Why the others are incorrect:
* B. On the terminal console running the web server
* That console shows logs from the server-side runtime (e.g., Node.js logs).
* This code is clearly browser-side JavaScript (no Node.js or server context shown).
* Therefore, the output does not appear in the terminal.
* C. In the browser performance tools log
* Performance tools (Timeline, Performance tab, etc.) show metrics about rendering, CPU time, network, etc., not generic console.log messages.
* console.log messages appear in the Console tab, not in performance logs.
* D. On the webpage console log
* There is no built-in concept of a "webpage console log" UI rendered on the page itself by default.

* Unless you explicitly code something to display logs in the DOM, console.log output is not visible on the page, only in Developer Tools.
Therefore, the correct place to see that log is:
The answer: A
JavaScript knowledge / Study Guide references (concept names only, no links):
* Browser Developer Tools - Console tab
* console.log() in client-side JavaScript
* Difference between client-side logs and server-side logs

NEW QUESTION: 70
Universal Containers (UC) notices that its application that allows users to search for accounts makes a network request each time a key is pressed. This results in too many requests for the server to handle.
* Address this problem, UC decides to implement a debounce function on string change handler.
What are three key steps to implement this debounce function?
Choose 3 answers:
A. Ensure that the network request has the property debounce set to true.
B. When the search string changes, enqueue the request within a setTimeout.
C. If there is an existing setTimeout and the search string changes, cancel the existing setTimeout using the persisted timerId and replace it with a new setTimeout.
D. If there is an existing setTimeout and the search string change, allow the existing setTimeout to finish, and do not enqueue a new setTimeout.
E. Store the timeId of the setTimeout last enqueued by the search string change handle.
Answer: A,B,D (LEAVE A REPLY)

NEW QUESTION: 71
Refer to the HTML below:



Which JavaScript statement results in changing "Tony" to "Mr. T. "?
A. Document.querySelectorAll ('#main # TONY ') , innerHTML = Mr, T , ' ;
B. Document.querySelectorAll ('#main li, Tony ') innerHTNL = Mr, T , ' ;
C. Document.querySelector (' '#main li:second-child') innerHTML, = Mr, T, ';
D. Document.querySelector ("#main li:nth-child (2) ') , innerHTML = 'Mr . T ';
Answer: A (LEAVE A REPLY)

NEW QUESTION: 72
A developer creates a generic function to log custom messages in the console. To do this, the function below is implemented.
01 function logStatus(status){
02 console./*Answer goes here*/{'Item status is: %s', status};

03 }
Which three console logging methods allow the use of string substitution in line 02?
A. Message
B. Assert
C. Info
D. Error
E. Log
Answer: (SHOW ANSWER)

NEW QUESTION: 73
Refer to the following code:
< html lang= " en " >
< body >
< button class= " secondary " > Save draft < /button >
< button class= " primary " > Save and close < /button >
< /body >
< script >
function displaySaveMessage(event) {
console.log(' Save message. ');
}
function displaySuccessMessage(event) {
console.log(' Success message. ');
}
window.onload = function() {
document.querySelector(' .secondary ')
.addEventListener(' click ' , displaySaveMessage, true);
document.querySelector(' .primary ')
.addEventListener(' click ' , displaySuccessMessage, true);
}
< /script >
< /html >
A. > Outer message
B. > Outer message
Inner message
C. > Inner message
D. > Inner message
Outer message
Answer: B (LEAVE A REPLY)
The answer choices appear to belong to an event propagation question using nested elements such as an outer element and an inner element.
The important clue is this third argument:

addEventListener(' click ' , handlerFunction, true);
When the third argument is true, the event listener runs during the capturing phase .
In event capturing, the browser handles the event from the outside toward the inside:
Outer element # Inner element
So, if an inner element is clicked and both the outer and inner elements have click listeners registered with capture mode set to true, the outer listener runs first, then the inner listener runs.
That produces:
Outer message
Inner message
So the matching option is B .
Important correction:
In the exact code shown, the buttons are sibling elements:
< button class= " secondary " > Save draft < /button >
< button class= " primary " > Save and close < /button >
They are not nested inside one another. Therefore, with the code exactly as written:
Clicking the secondary button logs:
Save message.
Clicking the primary button logs:
Success message.
However, because the provided answer choices refer to Outer message and Inner message , the intended concept is clearly event capturing. In a capturing-phase question with nested outer and inner elements, the correct order is:
Outer message
Inner message
Therefore, the verified answer for the intended event-capturing question is B .

NEW QUESTION: 74
Refer to the following array:
Let arr = [1, 2, 3, 4, 5];
Which three options result in x evaluating as (3, 4, 5)?
Choose 3 answers
A. Let x = arr.filter((a) => (a < 2)) ;
B. Let x = arr.alice (2) ;
C. Let x = arr.filter((a) =>)return a > 2)) ;
D. Let x = arr. Slince (2, 3);
E. Let x = arr.aplice (2, 3);
Answer: B,C,E (LEAVE A REPLY)

NEW QUESTION: 75
developer publishes a new version of a package with new features that do not break backward compatibility. The previous version number was 1.1.3.
Following semantic versioning format, what should the new package version number

be?
A. 1.2.0
B. 1.1.4
C. 2.0.0
D. 1.2.3
Answer: A (LEAVE A REPLY)

NEW QUESTION: 76
Refer to the code below:
Const searchTest = 'Yay! Salesforce is amazing!" ;
Let result1 = searchText.search(/sales/i);
Let result 21 = searchText.search(/sales/i);
console.log(result1);
console.log(result2);
After running this code, which result is displayed on the console?
A. > true > false
B. > 5 > -1
C. > 5 > 0
D. > 5 >undefined
Answer: D (LEAVE A REPLY)

NEW QUESTION: 77
JavaScript:
01 function Tiger() {
02 this.type = ' Cat ' ;
03 this.size = ' large ' ;
04 }
05
06 let tony = new Tiger();
07 tony.roar = () = > {
08 console.log(' They\ ' re great! ');
09 };
10

11 function Lion() {
12 this.type = ' Cat ' ;
13 this.size = ' large ' ;
14 }
15
16 let leo = new Lion();
17 // Insert code here
18 leo.roar();
Which two statements could be inserted at line 17 to enable line 18?
A. leo.roar = () = > { console.log(' They\ ' re pretty good! '); };
B. Object.assign(leo, tony);
C. Object.assign(leo, Tiger);
D. leo.prototype.roar = () = > { console.log(' They\ ' re pretty good! '); };
Answer: A,B (LEAVE A REPLY)
There are two valid ways to ensure leo.roar() exists:
* Directly assign a roar function to leo (Option A). Assigning a property to an instance object creates a new method on that instance.
* leo.roar = () = > { console.log(' They\ ' re pretty good! '); };
After this, calling leo.roar() is valid.
* Copy tony's properties into leo using Object.assign (Option B). Object.assign(target, source) copies enumerable own properties from the source object into the target object. Since tony.roar exists, executing:
* Object.assign(leo, tony);
copies roar into leo, making leo.roar() valid.
Why the other answers are incorrect:
* Option C: Object.assign(leo, Tiger) copies properties from the function object Tiger , not from Tiger.
prototype and not from a Tiger instance. Tiger (the function) has no roar property, so nothing useful is copied.
* Option D: leo.prototype is undefined because leo is an instance, not a constructor function. Only constructor functions have a .prototype property. This line would cause an error.
JavaScript Knowledge References (text-only)
* Instances have their own properties and do not contain a .prototype property.
* Object.assign(target, source) copies own enumerable properties of the source object.
* Assigning a function as a property of an object creates a callable method.

NEW QUESTION: 78
Refer to the code below:
let timeFunction =() => {
console.log('Timer called.");
};
let timerId = setTimeout (timedFunction, 1000);
Which statement allows a developer to cancel the scheduled timed function?
A. clearTimeout(timedFunction);
B. removeTimeout(timerId);

C. clearTimeout(timerId);
D. removeTimeout(timedFunction);
Answer: (SHOW ANSWER)

NEW QUESTION: 79
Refer to the code declarations below:

```
let str1 = 'Java';
let str2 = 'Script';
```

Which three expressions return the string JavaScript?
Choose 3 answers
A. Str1.concat (str2);
B. $(str1) $ (str2} ';
C. Str1.join (str2);
D. Str1 + str2;
E. Concat (str1, str2);
Answer: (SHOW ANSWER)

NEW QUESTION: 80
Refer to the code snippet below:

```
Let array = [1, 2, 3, 4, 4, 5, 4, 4];
For (let i =0; i < array.length; i++){
if (array[i] === 4) {
array.splice(i, 1);
}
}
```

What is the value of the array after the code executes?
A. [1, 2, 3, 5]
B. [1, 2, 3, 4, 4, 5, 4]
C. [1, 2, 3, 4, 5, 4]
D. [1, 2, 3, 4, 5, 4, 4]
Answer: C (LEAVE A REPLY)

NEW QUESTION: 81
Refer to the following code:

```
01 let obj = {
02   foo: 1,
03   bar: 2
04 }
05 let output = [];
06
07 for (let something in obj) {
08   output.push(something);
09 }
10
11 console.log(output);
```


What is the output of line 11?

A. ["foo", "bar"]

B. ["bar", "foo"]

C. [1,2]

D. ["foo:1", "bar:2"]

Answer: A (LEAVE A REPLY)

NEW QUESTION: 82

Which statement accurately describes the behaviour of the async/ await keyworks ?

A. The associated sometimes returns a promise.

B. The associated function will always return a promise

C. The associated class contains some asynchronous functions.

D. The associated function can only be called via asynchronous methods

Answer: A (LEAVE A REPLY)

NEW QUESTION: 83

Given the following code:

01 let x = null;

02 console.log(typeof x);

What is the output of line 02?

A. " object "

B. " undefined "

C. " x "

D. " null "

Answer: A (LEAVE A REPLY)

This question focuses on the typeof operator and how JavaScript represents the type of the value null.

* Declaration on line 1:

let x = null;
Here:
* x is explicitly assigned the value null.
* In JavaScript, null is a special primitive value that represents "no value" or "empty value".
* Line 2:
console.log(typeof x);
The typeof operator returns a string indicating the type of the operand. For example:
* typeof 1 returns " number "
* typeof " hello " returns " string "
* typeof true returns " boolean "
* typeof undefined returns " undefined "
* typeof {} returns " object "
* typeof [] returns " object " (arrays are objects)
* typeof function() {} returns " function "
For null, due to an early design decision in JavaScript, the result is:
typeof null === " object "
This is a known historical quirk of the language:
* Although null is a primitive and conceptually means "no object", typeof null returns the string " object "
.
Therefore, typeof x when x is null evaluates to " object " .
So on line 2, the value printed is:
" object "
This matches option A.
The incorrect options are:
* B. " undefined " : would be typeof x if x were undefined, not null.
* C. " x " : typeof returns a string describing the type of the value, not the variable name.
* D. " null " : although intuitive, JavaScript does not return " null " for typeof null; it returns " object " instead.
Therefore, the correct answer is:
The answer: A
JavaScript knowledge / study guide reference concepts:
* typeof operator and its return values
* Primitive types in JavaScript (null, undefined, number, string, boolean, symbol, bigint)
* Historical behavior: typeof null returning " object "
* Difference between null and undefined in JavaScript

NEW QUESTION: 84
Refer to the HTML below:

```
<div id="main">
  <ul>
    <li>Leo</li>
    <li>Tony</li>
    <li>Tiger</li>
  </ul>
</div>
```

Which JavaScript statement results in changing " The Lion."?

A. document.querySelector('$main li.Tony').innerHTML = '" The Lion ';

B. document.querySelectorAll('$main $TONY').innerHTML = '" The Lion

C. document.querySelector('$main li:second-child').innerHTML = " The Lion ';

D. document.querySelector('$main li:nth-child(2)'),innerHTML = " The Lion. ';

Answer: B (LEAVE A REPLY)

NEW QUESTION: 85

Correct implementation of try...catch for countsDeep():

A. try {
countsDeep();
} handleError (e){
catch(e);
}

B. setTimeout(function() {
try {
countsDeep();
} catch (e) {
handleError(e);
}
}, 1000);

C. try {
setTimeout(function() {
countsDeep();
}, 1000);
} catch (e) {
handleError(e);
}

D. try {
setTimeout(function() {

countsDeep();
}, 1000);
} catch (e) {
handleError(e);
}

Answer: B (LEAVE A REPLY)

The correct answer is B because countsDeep() is executed inside the callback function passed to setTimeout(), and the try...catch block is also placed inside that same callback.

In JavaScript, setTimeout() schedules a function to run later. The outer code finishes first, and the callback runs asynchronously after the delay. Because of this, a try...catch block placed outside setTimeout() cannot catch errors thrown later inside the callback.

Correct logic:

setTimeout(function() {
try {
countsDeep();
} catch (e) {
handleError(e);
}
}, 1000);

Here, when countsDeep() runs after 1000 milliseconds, any error thrown by countsDeep() happens inside the try block. Therefore, the catch (e) block can catch that error and pass it to handleError(e).

Why the other options are incorrect:

A is incorrect because the syntax is invalid JavaScript. A valid try...catch structure must be:

try {
// code
} catch (e) {
// handle error
}

Option A incorrectly writes:

} handleError (e){
catch(e);
}

That is not valid try...catch syntax.

C is incorrect because the try...catch surrounds only the call to setTimeout(), not the later execution of countsDeep(). If countsDeep() throws an error after the timer expires, the outer catch block will not catch it.

D is incorrect for the same reason as C. In the original question, D also had a typing error: it used countSheep() instead of countsDeep(). Even after correcting that typing error, D is still incorrect because the try...catch is outside the asynchronous callback.

NEW QUESTION: 86

Given the code below:

```
01 function Person(name, email) {
02   this.name = name;
03   this.email = email;
04 }
05
06 const john = new Person('John', 'john@email.com');
07 const jane = new Person('Jane', 'jane@email.com');
08 const emily = new Person('Emily', 'emily@email.com');
09
10 let usersList = [john, jane, emily];
```


Which method can be used to provide a visual representation of the list of users and to allow sorting by the name or email attribute?

A. console.table(usersList) ;

B. console.info(usersList) ;

C. console.group(usersList) ;

D. console.groupCol lapsed (usersList) ;

Answer: C (LEAVE A REPLY)

NEW QUESTION: 87

developer removes the HTML class attribute from the checkout button, so now it is simply:

<button>Checkout</button>.

There is a test to verify the existence of the checkout button, however it looks for a button with class= "blue". The test fails because no such button is found.

Which type of test category describes this test?

A. False negative

B. True positive

C. True negative

D. False positive

Answer: (SHOW ANSWER)

NEW QUESTION: 88

Refer to the code snippet below:

```
Let array = [1, 2, 3, 4, 4, 5, 4, 4];
For (let i =0; i < array.length; i++)
if (array[i] === 4) {
array.splice(i, 1);
}
}
```

What is the value of array after the code executes?

A. [1, 2, 3, 4, 5, 4, 4]

B. [1, 2, 3, 4, 4, 5, 4]

C. [1, 2, 3, 5]

D. [1, 2, 3, 4, 5, 4]

Answer: (SHOW ANSWER)

```
let array = [1, 2, 3, 4, 4, 5, 4, 4];
for (let i =0; i < array.length; i++){
if (array[i] === 4) {
array.splice(i, 1);
}
}
console.log(array)
▸ (6) [1, 2, 3, 4, 5, 4]                    VM1963:7
undefined
```


NEW QUESTION: 89

Refer to the code:

```
function Animal(size, type) {
this.size = size || "small";
this.type = type || "Animal";
this.canTalk =false;}
let Pet = function (size, type, name, owner){
Animal.call(this, size, type);
this.name = name;
this.owner = owner;}
Pet.prototype = Object.create(Animal.prototype);
let pet1 = new Pet();
console.log(pet1);
```

Given the code above, which three properties are set pet1?

Choose 3 answers:

A. canTalk
B. Type
C. Owner
D. Name
E. Size
Answer: (SHOW ANSWER)

NEW QUESTION: 90
Refer to the code below:
Function Person(firstName, lastName, eyecolor) {
this.firstName =firstName;
this.lastName = lastName;
this.eyeColor = eyeColor;
}
Person.job = 'Developer';
const myFather = new Person('John', 'Doe');
console.log(myFather.job);
What is the output after the code executes?
A. ReferenceError: eyeColor is not defined
B. Undefined
C. Developer
D. ReferenceError: assignment to undeclared variable "Person"
Answer: (SHOW ANSWER)

NEW QUESTION: 91
A developer wrote the following code:
01 let X = object.value;
02
03 try {
04 handleObjectValue(X);
05 } catch (error) {
06 handleError(error);
07 }
The developer has a getNextValue function to execute after handleObjectValue(), but does not want to execute getNextValue() if an error occurs.
How can the developer change the code to ensure this behavior?
A. 03 try{
04 handleObjectValue(x);
05 } catch(error){
06 handleError(error);
07 }

08 getNextValue();
B. 03 try {
04 handleObjectValue(x)
05 .....
C. 03 try{
04 handleObjectValue(x);
05 } catch(error){
06 handleError(error);
07 } then {
08 getNextValue();
09 }
D. 03 try{
04 handleObjectValue(x);
05 } catch(error){
06 handleError(error);
07 } finally {
08 getNextValue();
10 }
Answer: B (LEAVE A REPLY)

NEW QUESTION: 92
Refer to the code below?
Let searchString = ' look for this ';
Which two options remove the whitespace from the beginning of searchString?
Choose 2 answers
A. searchString.trimEnd();
B. searchString.replace(/*\s\s*/, '');
C. trimStart(searchString);
D. searchString.trimStart();
Answer: B,D (LEAVE A REPLY)

NEW QUESTION: 93

At Universal Containers, every team has its own way of copying JavaScript objects. The code snippet shows an Implementation from one team:

```
01 function Person() {
02   this.firstName = "John";
03   this.lastName = "Doe";
04   this.name = () => {
05     console.log(`Hello ${this.firstName} ${this.lastName}`);
06   }
07 }
08
09 const john = new Person();
10 const dan = JSON.stringify(JSON.parse(john));
11 dan.firstName = 'Dan';
12 dan.name();
```

What is the output of the code execution?

A. Hello Dan

B. SyntaxError: Unexpected token in JSON

C. Hello John Doe

D. Hello Dan Doe

Answer: B (LEAVE A REPLY)

NEW QUESTION: 94

bar, awesome is a popular JavaScript module. the versions publish to npm are:

1.2

1.3.1

1.3.5

1.4.0

Teams at Universal Containers use this module in a number of projects. A particular project has thepackage, json definition below.

```
{
  "name": "UC Project Extra",
  "version": "0.0.5",
  "dependencies":
    "bar.awesome": "~1.3.0"
  }
}
```

A developer runs this command: npm install.

Which version of bar .awesome is installed?

A. 1.4.0

B. 1.3.1
C. 1.3.5
D. The command fails, because version 130 is not found
Answer: C (LEAVE A REPLY)

NEW QUESTION: 95
Refer to the code:
01 let car1 = new Promise((_, reject) = >
02 setTimeout(reject, 2000, " Car 1 crashed in "));
03 let car2 = new Promise(resolve = >
04 setTimeout(resolve, 1500, " Car 2 completed "));
05 let car3 = new Promise(resolve = >
06 setTimeout(resolve, 3000, " Car 3 completed "));
07
08 Promise.race([car1, car2, car3])
09 .then(value = > {
10 let result = ' $(value) the race. ' ;
11 })
12 .catch(err = > {
13 console.log(" Race is cancelled. " , err);
14 });
What is the value of result when Promise.race executes?
A. Car 2 completed the race.
B. Car 3 completed the race.
C. Race is cancelled.
D. Car 1 crashed in the race.
Answer: (SHOW ANSWER)
Promise.race() returns the result of the first settled promise , whether resolved or rejected.
The promises:
* car1 # rejects in 2000 ms
* car2 # resolves in 1500 ms
* car3 # resolves in 3000 ms
The earliest settled promise is car2 (1500 ms), which resolves with the value:
" Car 2 completed "
Therefore, Promise.race enters the .then() branch with:
value = " Car 2 completed "
Inside the .then():
let result = ' $(value) the race. ' ;
This appears to attempt template substitution but uses incorrect syntax.
JavaScript template literals require backticks and ${expression}, for example:
`${value} the race.`

Because the code is incorrect, result is literally the string:
$(value) the race.
However, the question asks:
"What is the value of result when Promise.race executes ?"
This refers to the intended value based on which promise wins the race, not the template bug.
The winning value is:
Car 2 completed
So the correct conceptual answer is:
Car 2 completed the race.
JavaScript Knowledge References (text-only)
* Promise.race() returns the first settled (resolved or rejected) promise.
* The earliest resolve here is car2 (1500 ms).
* Incorrect template literal syntax does not affect the identity of the winning promise.

NEW QUESTION: 96
A developer wants to create an object from a function in the browser using the code below.

```
01 function Monster(){ this.name = 'hello' };
02 const m = Monster();
```

What happens due to the lack of the mm keyword on line 02?
A. window.name is assigned to 'hello' and the variable = remains undefined.
B. The m variable is assigned the correct object but this.name remains undefined.
C. The m variable is assigned the correct object.
D. window.m Is assigned the correct object.
Answer: A (LEAVE A REPLY)

NEW QUESTION: 97
Refer to the code below:
01 function changeValue(param) {
02 param = 5;
03 }
04 let a = 10;
05 let b = a;
06
07 changeValue(b);
08 const result = a + ' - ' + b;
What is the value of result when the code executes?
A. 5-5
B. 5-10
C. 10-5
D. 10-10

Answer: D (LEAVE A REPLY)

We must understand pass-by-value for primitives in JavaScript.

* Initial values:

let a = 10;

let b = a; // b gets a copy of the value 10

So:

* a is 10
* b is 10 (independent copy)
* Function call:

changeValue(b);

Function definition:

function changeValue(param) {

param = 5;

}

* param receives the value of b, which is 10.
* Inside the function, param is a local variable .
* param = 5; changes only this local copy.
* It does not affect b outside the function.

After the function call:

* a is still 10.
* b is still 10.
* Result:

const result = a + ' - ' + b;

* a is 10.
* b is 10.
* String concatenation: ' 10 - 10 ' .

So result is:

" 10 - 10 "

Therefore, the correct option is:

The answer: D

Study Guide Concepts:

* Primitive values (numbers, strings, booleans) are passed by value
* Function parameters as local variables
* String concatenation with +
* Difference between mutating references vs primitives

NEW QUESTION: 98

A developer is required to write a function that calculates the sum of elements in an array but is getting undefinedevery time the code is executed. The developer needs to find what is missing in the code below.

Const sumFunction = arr => {

Return arr.reduce((result, current) => {

//
Result += current;
//
), 10);
);
Which option makes the code work as expected?
A. Replace line02 with return arr.map((result, current) => (
B. Replace line 03 with if(arr.length == 0) (return 0;)
C. Replace line 04 with result = result +current;
D. Replace line 05 with return result;
Answer: (SHOW ANSWER)

NEW QUESTION: 99
bar, awesome is a popular JavaScript module. the versions publish to npm are:

1.2
1.3.1
1.3.5
1.4.0

Teams at Universal Containers use this module in a number of projects. A particular project has the package, json definition below.

```
{
  "name": "UC Project Extra",
  "version": "0.0.5",
  "dependencies":
    "bar.awesome": "~1.3.0"
  }
}
```


A developer runs this command: npm install.
Which version of bar .awesome is installed?
A. The command fails, because version 130 is not found
B. 1.3.1
C. 1.3.5
D. 1.4.0
Answer: C (LEAVE A REPLY)

NEW QUESTION: 100
A developer has an ErrorMandler module that contains multiple functions.
What kind of export should be leveraged so that multiple function can be used?

A. Multi
B. Named
C. default
D. All
Answer: (SHOW ANSWER)

NEW QUESTION: 101
Refer to the string below.
Const str='Salesforce';
Which two statements results in the word 'Sales'?
Answer:
Str.substring(0,5);
Str.substr(0,5);

NEW QUESTION: 102
is below:
<input type="file" onchange="previewFile()">

The JavaScript portion is:
01 function previewFile(){
02 const preview = document.querySelector('img');
03 const file = document.querySelector('input[type=file]').files[0];
04 //line 4 code
05 reader.addEventListener("load", () => {
06 preview.src = reader.result;
07 },false);
08 //line 8 code
09 }
In lines 04 and 08, which code allows the user to select an image from their local computer , and to display the image in the browser?
A. 04 const reader = new File();
08 if (file) reader.readAsDataURL(file);
B. 04 const reader = new FileReader();
08 if (file) URL.createObjectURL(file);
C. 04 const reader = new FileReader();
08 if (file) reader.readAsDataURL(file);
D. 04 const reader = new File();
08 if (file) URL.createObjectURL(file);
Answer: C (LEAVE A REPLY)

NEW QUESTION: 103

A developer has two ways to write a function:
Option A:
function Monster(){
this.growl = ()=>{
console.log('Grr!');
}
}
Option B:
function Monster(){};
Monster.prototype.growl = ()=>{
console.log('Grr!');
}
After deciding on an option, the developer creates 1000 monster objects.
How many growl methods are created with Option A and Option B?
A. 1 methods for both
B. 1000 for Option A, 1 for Option B
C. 1000 for both
D. 1 for Option A, 1000 for Option B
Answer: A (LEAVE A REPLY)

NEW QUESTION: 104
A developer is creating a simple webpage with a button. When a user clicks this button for the first time, a message is displayed.
The developer wrote the JavaScript code below, but something is missing. The message gets displayed every time a user clicks the button, instead of just the first time.
01 function listen(event) {
02 alert ('Hey! I am John Doe') ;
03 button.addEventListener ('click', listen);
Which two code lines make this code work as required?
Choose 2 answers
A. On line 04, use event.stopPropagation ( ),
B. On line 02, use event.first to test if it is the first execution.
C. On line 04, use button.removeEventListener(' click" , listen);
D. On line 06, add an option called once to button.addEventListener().
Answer: C,D (LEAVE A REPLY)

NEW QUESTION: 105
Refer to the code below:
01 const exec = (item, delay) =>{
02newPromise(resolve => setTimeout(() => resolve(item), delay)),
03 async function runParallel() {

04 Const (result1, result2, result3) = await Promise.all{
05 [exec ('x', '100') , exec('y', 500), exec('z', '100')]
06);
07 return `parallel is done: $(result1)$(result2)$(result3)`;
08 }
}
}
Which two statements correctly execute the runParallel () function?
Choose 2 answers
A. runParallel () .then(data);
B. Async runParallel () .then(data);
C. runParallel ( ). done(function(data){return data;});
D. runParallel () .then(function(data)return data
Answer: (SHOW ANSWER)

NEW QUESTION: 106
Refer to the code below:
const searchText = ' Yay! Salesforce is amazing! ' ;
let result1 = searchText.search(/sales/i);
let result2 = searchText.search(/sales/);
console.log(result1);
console.log(result2);
After running this code, which result is displayed on the console?
A. 5
undefined
B. 5
0
C. true
false
D. 5
-1
Answer: D (LEAVE A REPLY)
String: " Yay! Salesforce is amazing! "
Index positions:
* ' Y ' at 0, ' a ' at 1, ' y ' at 2, ' ! ' at 3, space at 4, ' S ' at 5, ' a ' at 6, ' l ' at 7, ' e ' at 8, ' s ' at 9, etc.
Substring " Sales " starts at index 5.
String.prototype.search with a regex returns the index of the first match or -1 if there is no match.
* searchText.search(/sales/i);
* /sales/i is case-insensitive because of the i flag.
* It matches " Sales " beginning at index 5.
* So result1 is 5.

* searchText.search(/sales/);
* /sales/ is case-sensitive.
* It requires lowercase " sales " .
* The text has " Sales " with uppercase S, so this does not match.
* search returns -1 when there is no match.
* So result2 is -1.
Console output:
* First log: 5
* Second log: -1
Option D matches this.
Concepts: regex search, case sensitivity vs i flag, String.prototype.search return values.

NEW QUESTION: 107
Refer to the code below:

```
01 let foodMenu1 = ['Pizza', 'Burger', 'French fries'];
02 let finalMenu = foodMenu1;
03 finalMenu.push('Garlic bread');
```

What is the value of foobDienu1 after the code executes?
A. {'Pizza' , Burger' , French fries '}
B. {'Garlic bread', Pizza' , Burger' , French fries'}
C. {'Pizza' , Burger' , French fries ' , garlic bread; }
D. {'Garlic bread'}
Answer: C (LEAVE A REPLY)

NEW QUESTION: 108
Refer to the string below:
const str = 'Salesforce';
Which two statements result in the word 'Sales'?
Choose 2 answers
A. str.substring (1, 5);
B. str.substring (0, 5);
C. str.substr (0, 5);
D. str.substr(1, 5);

Answer: (SHOW ANSWER)

NEW QUESTION: 109
A developer wants to set up a secure web server with Node.js. The developer creates a directory locally called app-server, and the first file is app-server/index.js.
Without using any third-party libraries, what should the developer add to index.js to create the secure web server?
A. const server = require(' secure-server ');
B. const tls = require(' tls ');
C. const http = require(' http ');
D. const https = require(' https ');
Answer: D (LEAVE A REPLY)
* Node.js provides two built-in modules for creating servers:
* http # creates non-secure HTTP servers
* https # creates secure HTTPS servers
* A secure server requires:
* SSL/TLS certificates (key and cert)
* The built-in Node.js module designed to handle these: https
* Syntax for a secure server:
const https = require(' https ');
const fs = require(' fs ');
const options = {
key: fs.readFileSync(' key.pem '),
cert: fs.readFileSync(' cert.pem ')
};
https.createServer(options, (req, res) = > {
res.write(' Secure Server ');
res.end();
}).listen(443);
* Evaluation of options:
* A is incorrect: No built-in module named secure-server exists.
* B is incorrect: tls provides low-level TLS sockets, not HTTP/HTTPS servers.
* C is incorrect: http only creates an insecure server.
* D is correct: https is the built-in module required for secure web servers.
JavaScript Knowledge References (text-only)
* Node.js exposes built-in modules http and https for server creation.
* Secure servers require the https module, which supports TLS/SSL.
* The tls module is lower-level and not used for full HTTP servers.

NEW QUESTION: 110
GIven a value, which three options can a developer use to detect if the value is NaN?
Choose 3 answers !

A. value == NaN
B. Number.isNaN(value)
C. value === Number.NaN
D. Object.is(value, NaN)
E. value ! == value
Answer: (SHOW ANSWER)

NEW QUESTION: 111
Refer to the following code block:
01 let array = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11];
02 let output = 0;
03
04 for (let num of array) {
05 if (output > 10) {
06 break;
07 }
08 if (num % 2 == 0) {
09 continue;
10 }
11 output += num;
12 }
What is the value of output after the code executes?
A. 16
B. 25
C. 11
D. 36
Answer: A (LEAVE A REPLY)
This code uses:
* A for...of loop to iterate over values in array.
* break to exit the loop entirely when output > 10.
* continue to skip even numbers.
* It sums only certain numbers into output.
Let's walk through the loop step by step.
Initial values:
* array = [1,2,3,4,5,6,7,8,9,10,11]
* output = 0
Loop: for (let num of array) { ... }
* First iteration: num = 1
* Line 05: if (output > 10) # 0 > 10 is false # no break.
* Line 08: if (num % 2 == 0) # 1 % 2 == 1, not 0, so false # no continue.
* Line 11: output += num # output = 0 + 1 = 1.

* Second iteration: num = 2
* output > 10 # 1 > 10 is false # no break.
* num % 2 == 0 # 2 % 2 == 0, so true # continue.
* Because of continue, line 11 is skipped.
* output remains 1.
* Third iteration: num = 3
* output > 10 # 1 > 10 is false.
* num % 2 == 0 # 3 % 2 == 1, false # no continue.
* output += num # output = 1 + 3 = 4.
* Fourth iteration: num = 4
* output > 10 # 4 > 10 is false.
* num % 2 == 0 # 4 % 2 == 0, true # continue.
* Skip sum; output remains 4.
* Fifth iteration: num = 5
* output > 10 # 4 > 10 is false.
* num % 2 == 0 # 5 % 2 == 1, false.
* output += num # output = 4 + 5 = 9.
* Sixth iteration: num = 6
* output > 10 # 9 > 10 is false.
* num % 2 == 0 # 6 % 2 == 0, true # continue.
* output remains 9.
* Seventh iteration: num = 7
* output > 10 # 9 > 10 is false.
* num % 2 == 0 # 7 % 2 == 1, false.
* output += num # output = 9 + 7 = 16.
* Eighth iteration would be num = 8, but:

At the top of the loop body, line 05 is checked again:

* if (output > 10) # 16 > 10 is true, so break; is executed.

When break runs:

* The loop terminates immediately.
* No further iterations (for num = 8, 9, 10, 11) are executed.
* Therefore, output stays at 16.

Final value of output after the loop ends is 16.

This matches option A.

Why other options do not match:

* B. 25: Would require adding more odd numbers (e.g., 9, 11) after 7, but the loop stops early due to output > 10.
* C. 11: Would be smaller; the actual sum of 1 + 3 + 5 + 7 until break is 16.
* D. 36: Would require summing many more values (e.g., most or all odd numbers up to 11), but again, the break condition stops the loop once output exceeds 10.

So:

The answer: A

JavaScript knowledge / Study Guide references (concept names only, no links):
* for...of loop over arrays
* break statement in loops (terminating a loop early)
* continue statement in loops (skipping to the next iteration)
* Modulo operator % to test even and odd numbers
* Step-by-step execution and control flow in loops

NEW QUESTION: 112
Which two code snippets showworking examples of a recursivefunction?
Choose 2 answers
A. Function factorial (numVar) {If (numVar < 0) return;If (numVar ===0) return 1;return numVar -1;
B. Const factorial =numVar => {If (numVar < 0) return;If (numVar === 0) return 1;returnnumVar * factorial (numVar - 1);};
C. Const sumToTen = numVar => {If (numVar < 0)Return;return sumToTen(numVar + 1)};
D. Let countingDown = function(startNumber) {If (startNumber >0) {console.log(startNumber) ;return countingDown(startNUmber);}
else {return startNumber;}};
Answer: (SHOW ANSWER)

NEW QUESTION: 113
Given the code below:
```
const delay = async delay =>{
return new Promise((resolve,reject)=>{
console.log(1);
setTimeout(resolve,deleay);
});
};
const callDelay = async ()=>{
console.log(2);
const yup = await delay(1000);
console.log(3);
}
console.log(4);
callDelay();
console.log(5);
```
What is logged to the console?
Answer:
4 2 1 5 3

NEW QUESTION: 114
Refer to the code below:
```
const event = new CustomEvent(
//Missing Code
```

);
obj.dispatchEvent(event);
A developer needs to dispatch a custom event called update to send information about recordId.
Which two options could a developer insert at the placeholder in line 02 to achieve this?
Choose 2 answers
A. { type : 'update', recordId : '123abc' }
B. 'Update' , '123abc'
C. 'Update' , {
Details : {
recordId : '123abc'
}
}
D. 'Update' , (
recordId : '123abc'
(
Answer: C,D (LEAVE A REPLY)

NEW QUESTION: 115
Given the code below:
01 const delay = async delay = > {
02 return new Promise((resolve, reject) = > {
03 console.log(1);
04 setTimeout(resolve, delay);
05 });
06 };
07
08 const callDelay = async () = > {
09 console.log(2);
10 const yup = await delay(1000);
11 console.log(3);
12 };
13
14 console.log(4);
15 callDelay();
16 console.log(5);
What is logged to the console?
A. 4 2 1 5 3
B. 4 2 1 5 3
C. 1 4 2 3 5
D. 4 5 1 2 3
Answer: (SHOW ANSWER)

Execution order:
* Top-level code runs synchronously:
* Line 14: console.log(4); # logs 4.
* Line 15: callDelay(); is called.
* Inside callDelay:
* Line 9: console.log(2); # logs 2.
* Line 10: await delay(1000);:
* Calls delay(1000).
* Inside delay(1000):
* Line 3: console.log(1); # logs 1.
* Line 4: setTimeout(resolve, delay); schedules resolve in 1000 ms.
* delay returns a pending Promise. await pauses callDelay here and returns control to the event loop.
* Back to top-level:
* Line 16: console.log(5); # logs 5.
So synchronous log sequence is: 4, 2, 1, 5.
* After ~1000 ms:
* The setTimeout in delay resolves the Promise.
* The await in callDelay resumes.
* Line 11: console.log(3); # logs 3.
Final log order: 4 2 1 5 3.
Both A and B show the same sequence; one must be chosen, so A is correct.
Concepts: async/await flow, Promise resolution timing, event loop, and ordering of synchronous vs timer callbacks.

NEW QUESTION: 116
Refer to the code below:
Let textValue = '1984';
Which code assignment shows a correct way to convert this string to an integer?
A. Let numberValue = Integer(textValue);
B. Let numberValue = textValue.toInteger();
C. let numberValue = Number(textValue);
D. Let numberValue = (Number)textValue;
Answer: (SHOW ANSWER)

NEW QUESTION: 117
Refer tothe code below:
const event = new CustomEvent(
//Missing Code
);
obj.dispatchEvent(event);
A developer needs to dispatch a custom event called update to send information about recordId.
Which two options could a developer insert at the placeholder in line02 to achieve this?

Choose 2 answers

A. 'Update' , '123abc'

B. 'Update' , {Details : {recordId : '123abc'}}

C. 'Update' , (recordId : '123abc'(

D. { type : 'update', recordId : '123abc' }

Answer: B,C (LEAVE A REPLY)

NEW QUESTION: 118

```
static delay = async delay = > {
return new Promise(resolve = > {
setTimeout(resolve, delay);
});
};
static asyncCall = async () = > {
await delay(1000);
console.log(1);
};
console.log(2);
asyncCall();
console.log(3);
```

Assume delay and asyncCall are in scope as functions.

What is logged to the console?

A. 1 2 3

B. 1 3 2

C. 2 1 3

D. 2 3 1

Answer: D (LEAVE A REPLY)

The correct answer is D , because JavaScript executes synchronous code first, while the code after await runs later after the Promise resolves.

The execution order is:

```
console.log(2);
```

This runs first, so JavaScript logs:

2

Then this line runs:

```
asyncCall();
```

The asyncCall() function starts executing. Inside it, JavaScript reaches:

```
await delay(1000);
```

The delay(1000) function returns a Promise that resolves after 1000 milliseconds:

```
return new Promise(resolve = > {
setTimeout(resolve, delay);
});
```

Because await pauses the rest of the asyncCall() function, this line does not run immediately:
console.log(1);
Instead, JavaScript continues executing the remaining synchronous code outside the async function:
console.log(3);
So JavaScript logs:
3
After approximately 1000 milliseconds, the Promise returned by delay(1000) resolves. Then the paused async function continues and runs:
console.log(1);
So JavaScript logs:
1
Final console output:
2
3
1
Important JavaScript concepts involved:
An async function always returns a Promise.
The await keyword pauses execution only inside the async function where it appears.
Code outside the async function continues running normally.
setTimeout() schedules its callback to run later, after the current synchronous code has finished.
Therefore, the verified answer is D. 2 3 1 .

NEW QUESTION: 119
A developer creates an object where its properties should be immutable and prevent properties from being added or modified.
Which method should be used to execute this business requirement?
A. Object. filebase ( )
B. Object. real ( )
C. Object const ( )
D. Object. Lock ( )
Answer: A (LEAVE A REPLY)

NEW QUESTION: 120
Refer to the following object:
const cat ={
firstName: 'Fancy',
lastName: ' Whiskers',
Get fullName() {
return this.firstName + ' ' + this.lastName;
}
};
How can a developer access the fullName property for cat?

A. cat.function.fullName()
B. cat.fullName
C. cat.fullName()
D. cat.get.fullName
Answer: (SHOW ANSWER)

NEW QUESTION: 121
Refer to the code snippet:
Function getAvailabilityMessage(item) {
If (getAvailability(item)){
Var msg ="Username available";
}
Return msg;
}
A developer writes this code to return a message to user attempting to register a new username. If the username is available, variable.
What is the return value of msg hen getAvailabilityMessage ("newUserName") is executed and getAvailability("newUserName") returns false?
A. undefined
B. "Msg is not defined"
C. "Username available"
D. "newUserName"
Answer: A (LEAVE A REPLY)

NEW QUESTION: 122
Which statement can a developer apply to increment the browser's navigation history without a page refesh?
A. Window.history,state,push.(newStateObject);
B. Window.history,state,push.(newStateObject, ' ' null;
C. Window.history,pushState.(newStateObject, ' ', null)) ;
D. Window.history,pushState.(newStateObject);
Answer: B (LEAVE A REPLY)