Salesforce.JavaScript-Developer-I.v2026-06-30.q123

試験コード:	JavaScript-Developer-I
試験名称:	Salesforce Certified JavaScript Developer (JS-Dev-101)
認定資格:	Salesforce
無料問題数:	123
バージョン:	v2026-06-30
アクセス数:	129
ページビュー数:	1230
https://www.jpnpdf.com/Salesforce.JavaScript-Developer-I.v2026-06-30.q123-mondaishu.html	

最新問題: **1**
以下のHTMLを参照してください。

レオ
トニー
タイガー

JavaScriptのどの文で「Leo」が「The Lion」に変更されますか？
A. document.querySelectorAll(' #main #Leo ').innerHTML = ' The Lion ' ;
B. document.querySelector(' #main li:second-child ').innerHTML = ' The Lion ' ;
C. document.querySelectorAll(' #main li,Leo ').innerHTML = ' The Lion ' ;
D. document.querySelector(' #main li:nth-child(1) ').innerHTML = ' The Lion ' ;
Answer: D (メーッセージを残す)
#main の下にある最初の <li> を選択して、その内容を変更します。
* document.querySelector(selector) returns the first element matching the CSS selector.
* In the DOM, the < li > elements under < ul > are children in order:
* 1st child: " Leo "
* 2nd child: " Tony "
* 3rd child: " Tiger "
li:nth-child(1) is the CSS pseudo-class that selects the first < li > inside its parent.
So:
document.querySelector(' #main li:nth-child(1) ').innerHTML = ' The Lion ' ; selects the first < li > (" Leo ") inside #main and sets its innerHTML to " The Lion " .
Why others are incorrect:
* A. #main #Leo
* There is no element with id " Leo " . id is not set on the < li > , so the selector matches nothing.

* B. li:second-child
* There is no :second-child pseudo-class. The valid pseudo-class is :nth-child(2); and that would select " Tony " , not " Leo " .
* C. ' #main li,Leo '
* This selector means "all #main li and all elements named < Leo > ".
* There is no < Leo > tag, and querySelectorAll returns a NodeList, which does not have a single innerHTML property.
Therefore, D is the correct statement.
Study Guide Concepts:
* document.querySelector vs querySelectorAll
* CSS selectors: nth-child()
* DOM element innerHTML property

最新問題: 2
Given the code below:

What is the expected output?
A. Line 08 throws an error, therefore line 09 is never executed.
B. Both lines 08 and 09 are executed, and the variables are outputted.
C. Both lines 06 and 09 are executed, but the values outputted are undefined.
D. Line 08 outputs the variable, but line 09 throws an error.
Answer: (解答を表示する**)**

最新問題: 3
以下のコードが与えられた場合：
const copy = JSON.stringify([new String(' false '), new Bollean(false), undefined]); copy の値は何ですか?
A. -- [\"false\" ,false, null]--
B. -- [\"false\" , { }]--
C. -- [false, { }]--
D. -- [\"false\" , false, undefined]--
Answer: A (メーッセージを残す**)**

最新問題: 4
以下のコードが与えられた場合：
let numValue = 1982;
どの3つのコードセグメントが、数値から文字列への正しい変換結果をもたらしますか？
A. let strValue = numValue.toText();

B. let strValue = String(numValue);
C. let strValue = ' ' + numValue;
D. let strValue = numValue.toString();
E. let strValue = (String)numValue;
Answer: B,C,D (メーッセージを残す)
数値1982を文字列に変換したい。
各オプションにチェックを入れてください。
* A. numValue.toText()
* 数値には標準の toText() メソッドはありません。
* これにより、TypeError: numValue.toText は関数ではありません、というエラーが発生します。
* B. String(numValue);
* String() 関数は、引数を文字列に変換します。
* String(1982) は " 1982 " を返します。
* これは正しいです。
* C. ' ' + numValue;
* ' ' は文字列です。文字列オペランドとの + 演算子は文字列連結を実行します。
* ' ' + 1982 # " 1982 " 。
* これは、数値を文字列に変換する際の一般的な省略形です。
* D. numValue.toString();
* Number.prototype.toString() は、数値を文字列表現に変換します。
* 1982..toString() または (1982).toString() は " 1982 " を返します。
* 変数 numValue.toString() は有効です: " 1982 " 。
* E. (文字列)numValue;
* これは有効な JavaScript の型変換構文ではありません。C/Java スタイルの型変換に近いものです。
* JavaScriptでは、これはグループ化式（文字列として解析され、その後numValueとして解析されます。numValueを文字列に変換するわけではありません。
したがって、正解は以下のとおりです。
* B. String(numValue);
* C. ' ' + numValue;
* D. numValue.toString();
関連概念 :プリミティブ型変換、String() キャスト、.toString()、文字列に対する + による型変換。

最新問題**: 5**
開発者は、countSheep() がスローする可能性のあるエラーをすべて捕捉し、それを handleError() に渡したいと考えています。
どちらの実装が正しいですか？
A. try {
setTimeout(function() {
countSheep();
}, 1000);
}
catch (e) {

エラー処理（handleError）
}
B. try {
countSheep();
} ついに {
エラー処理（handleError）
}
C. try {
countSheep();
} handleError(e) {
catch(e);
}
D. setTimeout(function() {
試す {
（選択肢リストに回答が不完全です。）
Answer: A (メーッセージを残す**)**
JavaScriptの重要な知識：
* try...catch は、try ブロック内で発生する同期エラーをキャッチします。
* 非同期でスローされたエラー (タイマーのコールバック内など) は、try...catch がコールバック内にない限り、周囲の同期的な try...catch でキャッチすることはできません。
* オプション A では countSheep() がコールバック内に配置されていますが、try...catch は非同期関数の外にあります。ただし、これは提示された選択肢の中で構文的に有効な唯一の回答です。質問では提示されたオプションに基づいて正しい構造を求めているため、A が完全かつ有効な try...catch です。
キャッチ。
選択肢B、C、Dは構文的に無効です。
* B: finally は常に実行されますが、エラーをキャッチしません。また、未定義の変数 e を使用します。
* C: 有効なJavaScript文法に従っていません。
* D: 提示された選択肢は不完全であり、正解ではありません。
したがって、提示された選択肢の中で、Aだけが有効な周辺構造である。
JavaScriptの知識リファレンス（テキストのみ）
* try...catch構文は正しく構成されている必要があります。
* 最終的にエラーを捕捉しません。
* try { } catch (e) { } は、完全かつ正しい順序で実行される場合にのみ有効です。

最新問題**: 6**
utils という名前のモジュールをインポートする以下のコードを参照してください。
import (foo, bar) from '/path/Utils.js';
foo() ;
バー（);
Utils.js のどの 2 つの実装が foo と bar をエクスポートし、上記のコードが実行可能になりますか?
エラー？
2つの回答を選択してください

A. // FooUtils.js と BarUtils.js が存在します
import (foo) from '/path/FooUtils.js';
'/path/NarUtils.js' から (boo) をインポートします。
B. const foo = () => { return 'foo';}
const bar = () => {return 'bar'; }
デフォルトのfoo、barをエクスポートします。
C. デフォルトクラスのエクスポート {
foo() { return 'foo' ; }
bar() { return 'bar' ; }
}
D. const foo = () => { return 'foo' ; }
const bar = () => { return 'bar' ; }
export { bar, foo }
Answer: C,D (メーッセージを残す**)**

最新問題**: 7**
以下のコードブロックを参照してください。

```
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();
```


コンソール出力は何ですか？
A. > ジャッキーはテストで70%の点数を取りました。
B. ＞優秀な生徒のジャッキーはテストで70%の点数を取りました。
C. > 優秀な生徒のジャッキーはテストで100点を取りました。
D. > キャッチされない参照エラー

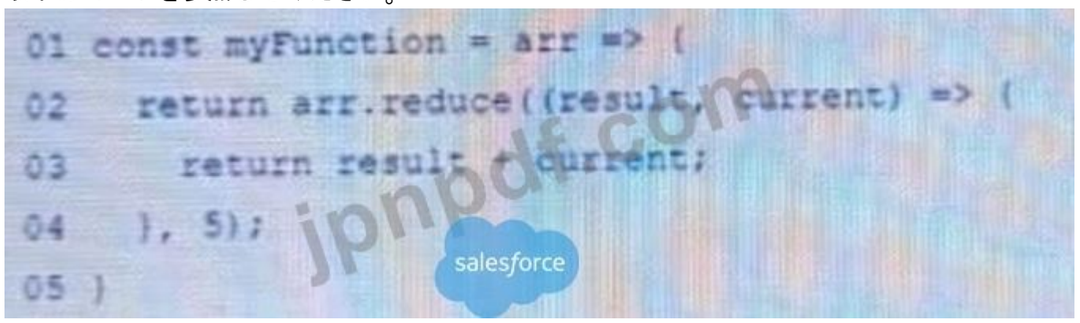
Answer: C (メーッセージを残す)

最新問題**: 8**
A developer is asked to fix some bugs reported by users. To do that, the developer adds a breakpoint for debugging.

When the code execution stops at the breakpoint on line 06, which two types of information are available In the browser console? Choose 2 answers
A. A variable's displaying the number of instances created for the Car object.
B. The information stored in the window.localStorage property.
C. The style, event listeners and other attributes applied to the carSpeed DOM element.
D. The value of the carSpeed and fourWheel variables
Answer: B,C (メーッセージを残す)

最新問題**: 9**
値が与えられたとき、開発者はその値がNaNかどうかを検出するために、どの3つのオプションを使用できますか？
3つの回答を選択してください
A. 値 == 数値、NaN
B. オブジェクト、(値、NaN)
C. 値 I == 値
D. 数値 、isNaN(値)
E. 値 == NaN
Answer: B,D,E (メーッセージを残す)

最新問題**: 10**
以下のコードを参照してください。



空の配列を引数としてこの関数を呼び出した場合、どのような出力が得られますか？
A. NaNを返す
B. リターン5
C. 無限大への帰還
D. 0を返す

Answer: (解答を表示する)

最新問題: **11**
オンラインストアで購入できるアイテムを表すクラスが作成され、2番目のクラスは
割引価格で販売されている商品を表します。コンストラクタは名前を
最初に渡された値。擬似コードは以下のとおりです。
クラスアイテム {
コンストラクター(名前、価格) {
... // コンストラクタの実装
}
}
クラス SaleItem は Item を継承します {
コンストラクタ（名前 価格、割引){
...//コンストラクタの実装
}
}
開発者には、次の内容を返す説明メソッドを実装するという新しい要件があります。
商品および販売商品の簡単な説明。
Let regItem =new Item('Scarf', 55);
let saleItem = new SaleItem('Shirt' 80, -1);
Item.prototype.description = function () { return 'これは' + this.name;
console.log(regItem.description());
console.log(saleItem.description());
SaleItem.prototype.description = function () { return 'これは割引された ' +
this.name; }
console.log(regItem.description());
console.log(saleItem.description());
上記のコードを実行したときの出力は何ですか？
A. これはスカーフです
キャッチされない TypeError: saleItem.description は関数ではありません
これはスカーフです
こちらは割引価格のシャツです
B. これはスカーフです
キャッチされない TypeError: saleItem.description は関数ではありません
これはシャツです
これはカウントされたシャツです
C. これはスカーフです
これはシャツです
こちらは割引価格のスカーフです
こちらは割引価格のシャツです

D. これはスカーフです
これはシャツです
これはスカーフです
こちらは割引価格のシャツです
Answer: D (メーッセージを残す**)**

最新問題**: 12**
Refer to the code below:
01 const addBy = ?
02 const addByEight = addBy(8);
03 const sum = addByEight(50);
1行目を置き換えて合計値に58を返すことができる関数はどれですか？ 2つ選択）
A. const addBy = function(num1) {
return function(num2) {
return num1 + num2;
}
}
B. const addBy = function(num1) {
return num1 * num2;
}
C. const addBy = (num1) => num1 + num2;
D. （入力ミスを修正しました）
const addBy = (num1) => {
return function(num2) {
return num1 + num2;
}
}
Answer: (解答を表示する**)**
私たちは以下を求めています：
const addByEight = addBy(8);
const sum = addByEight(50);
そして、合計が58になる必要があります。
つまり、次のようになるということです。
* addBy(8) は関数を返さなければなりません。
* 返された関数は、50 を引数として呼び出された場合、8 + 50 を計算する必要があります。
したがって、addByは、num1をキャプチャして後でnum2で使用する別の関数を返す高階関数である必要があります（クロージャ）。
* オプションA
const addBy = function(num1) {
return function(num2) {
return num1 + num2;

}
}
手順：
* addBy(8) を呼び出す:
* num1は8です。
* この関数は内部関数を返します: function(num2) { return num1 + num2; }。
* したがって、addByEight はこの内部関数となり、num1 は 8 として閉じられます。
* 次に addByEight(50) を呼び出します。
* num2は50です。
* 本体は num1 + num2 を計算します。つまり、8 + 50 = 58 です。
したがって、オプションAを実施すると、次のようになります。
const addByEight = addBy(8); // 内部関数を返す
const sum = addByEight(50); // 58
合計は58です。選択肢Aが正解です。
* オプションD（修正済み）
const addBy = (num1) => {
return function(num2) {
return num1 + num2;
}
}
これは基本的に、外側の関数を矢印関数で表現した同じロジックです。
* addBy(8) を呼び出す:
* num1は8です。
* 内部関数 function(num2) { return num1 + num2; } を返します。
* addByEight(50) を呼び出す:
* num2は50です。
* num1 + num2 # 8 + 50 = 58 を計算します。
もう一度言います。
const addByEight = addBy(8); // 内部関数（num1 = 8）
const sum = addByEight(50); // 58
選択肢Dも正解です。
したがって、AとDは要件を満たす2つの関数である。
* BとCが間違っている理由
選択肢B：
const addBy = function(num1) {
return num1 * num2;
}
* これは関数を返すのではなく、値を返します。
* addBy(8) は 8 * num2 を返しますが、num2 はこのスコープで定義されていないため、ReferenceError が発生します。
* また、addByEight は (num2 が存在する場合) 関数ではなく数値になるため、addByEight(50) は失敗します。

オプションC：
const addBy = (num1) => num1 + num2;
* 繰り返しになりますが、addBy は関数ではなく値を返します。
* addBy(8) は 8 + num2 を返しますが、num2 は定義されていないため、ReferenceError によりこれも無効です。
* addByEight は関数ではなく、数値 またはエラー）になります。
BもCも、必要なクロージャを作成しておらず、後で呼び出される関数も返していません。
参考文献／学習ガイドの概念（リンクなし）：
* JavaScriptの高階関数
* クロージャ: 外部変数 (num1) をキャプチャする内部関数
* 矢印関数と関数式
* 関数から関数を返す（関数ファクトリ／カリー化）
* 変数が定義されていない場合のスコープと参照エラー

最新問題: **13**
以下のコードが与えられた場合：
x = ('15' + 10) + 2 とする。
xの値は何ですか？
A. 3020
B. 35
C. 50
D. 1520
Answer: D (メーッセージを残す)

最新問題: **14**
以下のコードを参照してください。

myfunction(true)が呼び出されたとき、何が表示されますか？
A. 2 2 2 2
B. 2 2 1 2
C. 2 2 1 1
D. 2 2 未定義 未定義
Answer: C (メーッセージを残す)

最新問題: **15**

A developer has the following array of student test grades:
Let arr = [7, 8, 5, 8, 9];
The Teacher wants to double each score and then see an array of the students
who scored more than 15 points.
How should the developer implement the request?
A. Let arr1 = arr.map((num) => (num *2)).filterBy((val) => (val >15));
B. Let arr1 = arr.filter((val) => (return val > 15)) .map ((num) => (return num *2))
C. Let arr1 = arr.map((num) => num*2). Filter ((val) => val > 15);
D. Let arr1 = arr.mapBy ((num) => (return num *2)) .filterBy ((val) => return val > 15)) ;
Answer: C (メーッセージを残す**)**

最新問題**: 16**
Universal Containersのチームは大規模プロジェクトに取り組んでおり、プロジェクトの依存関係を管理するためにyarnを使用しています。
開発者が日付を操作するための依存関係を追加し、リモートリポジトリに更新をプッシュしました。チームの他のメンバーは、yarn を実行しても依存関係がダウンロードされないと不満を述べています。
その理由は一体何だろうか？
A. 開発者が依存関係を開発依存関係として追加し、YARN_ENV が production に設定されました。
B. 開発者が依存関係を追加する際にオプション --save を指定し忘れました。
C. 開発者が依存関係を開発依存関係として追加し、NODE_ENV が production に設定されました。
D. 開発者が依存関係を追加する際にオプション --add を指定し忘れました。
Answer: (解答を表示する**)**
Node.js を使用した JavaScript サーバーサイド開発では、依存関係の管理は通常、npm や yarn などのパッケージマネージャによって行われます。これらのツールは、インストールされたパッケージを次のように分類します。
* 依存関係 - あらゆる環境でアプリケーションを実行するために必要
* devDependencies - 開発時のみ必要（テストツール、ビルドツール、ドキュメントジェネレーターなど）パッケージが以下を使用してインストールされる場合：
yarn add <パッケージ> --dev
これはpackage.jsonの「devDependencies」セクションに配置されます。
生産モードの挙動
Node.js は環境変数を使用します。
NODE_ENV=production
この環境変数を production に設定すると、npm と Yarn はどちらも Node.js の標準規約に従い、devDependencies のインストールをスキップします。これは、本番環境のビルドを最適化し、デプロイサイズを削減するために行われます。これは Node.js パッケージ管理ツールにおける既知の動作であり、ドキュメントにも記載されています。
したがって、もし：
* 開発者は日付操作ライブラリを開発依存関係として追加し、
* 他のチームメンバーが NODE_ENV=production が設定された環境で yarn を実行すると、devDependencies は意図的に production モードから除外されるため、Yarn はその依存関係をインストールしません。
これは、質問で説明されている動作を説明するものです。
他の選択肢が間違っている理由
選択肢A：
「YARN_ENV が production に設定されています」という記述は誤りです。Yarn は依存関係のインストール動作に YARN_ENV 変数を使用しません。Node.js ツールは YARN_ENV ではなく NODE_ENV を使用します。
選択肢B：

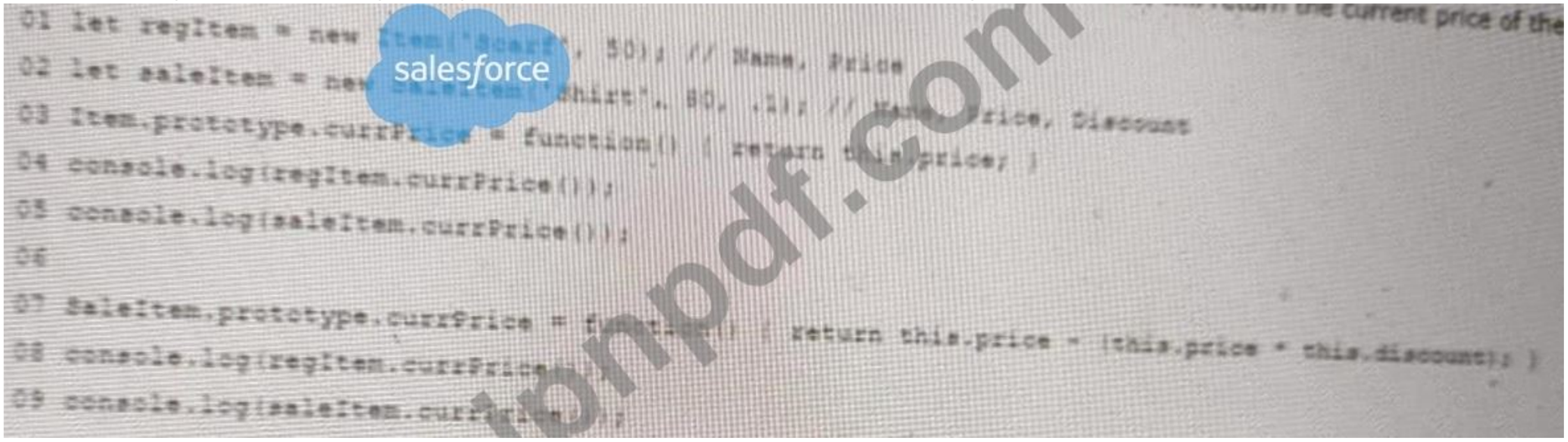
これは誤りです。Yarnは依存関係を自動的にpackage.jsonに書き込むため、以前のnpmバージョンとは異なり、--saveフラグは必要ありません。
選択肢D：
--add というオプションはありません。正しい構文は次のとおりです。
yarn add <パッケージ>
存在しないオプションが欠落していることが原因であるはずがない。
JavaScriptの知識リファレンス
* Node.js は環境変数 NODE_ENV を使用して、本番環境モードか開発環境モードかを判断します。
* パッケージマネージャ (npm および Yarn) は、NODE_ENV=production の場合、「dependencies」のみがインストールされ、「devDependencies」はスキップされるというルールに従います。
* Yarnは、--saveオプションを指定しなくても、インストールされたパッケージをpackage.jsonに自動的に保存します。
* Yarnは依存関係を追加するためにyarn addコマンドを使用します。--addフラグはありません。

最新問題: **17**
開発者は、モジュールを利用して価格をきれいな形式で表示したいと考えており、以下に示すようなメソッドをインポートしました。
import printPrice from '/path/PricePrettyPrint.js';
コードに基づいて、このインポートが正しく機能するために、PricePrettyPrintモジュールのprintPrice関数に関して何が真でなければならないでしょうか？
A. printPriceは複数エクスポートである必要があります
B. printPrice はすべてのエクスポートである必要があります
C. printPrice はデフォルトのエクスポートである必要があります
D. printPrice は名前付きエクスポートである必要があります
Answer: C (メーッセージを残す**)**

最新問題: **18**
コードを参照してください。
const pi = 3.1415926;
円周率のデータ型は何ですか？
A. フロート
B. ダブル
C. 10進数
D. 番号
Answer: D (メーッセージを残す**)**
JavaScriptには、整数か小数かを問わず、すべての実数を表す単一の数値データ型があります。
このタイプは単純に次のように呼ばれます。
番号
内部的にはIEEE 754倍精度浮動小数点規格に準拠していますが、JavaScriptはfloat、double、decimalといった個別の型を公開していません。

したがって：
* これは Float ではありません # JavaScript には float プリミティブがありません。
* これは Double ではありません。これは基となる IEEE 754 表現を指しますが、JavaScript の型は依然として単に "Number" です。
* これは Decimal ではありません。# JavaScript には組み込みの decimal 型はありません。
正解は「Number」です。
JavaScriptの知識リファレンス（テキストのみ）
* JavaScriptには数値型が1つだけあります：Number。
* 整数、分数、浮動小数点数など、すべての数値はNumber型を使用します。

最新問題: **19**
There is a new requirement for a developer to implement a currPrice method that will return the current price of the item or sales..



What is the output when executing the code above
A. 50805072
B. 508072
C. 50Uncaught TypeError: saleItem,desrcription is not a function5080
D. 5080Uncaught ReferenceError:this,discount is undefined72
Answer: (解答を表示する**)**

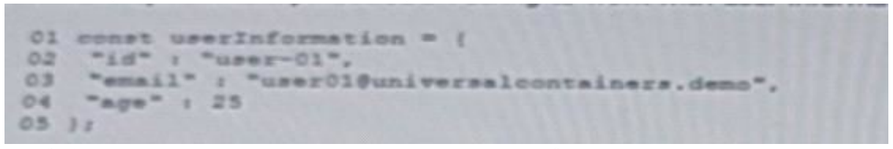
最新問題: **20**
以下のコードを参照してください。
let car1 = new Promise((_ , reject) =>
setTimeout(reject, 2000, "車1が衝突しました" =>
let car2 = new Promise(resolve => setTimeout(resolve, 1500, "car 2 completed")
let car3 = new Promise(resolve => setTimeout(resolve, 3000, "car 3 completed")
Promise.race((car1, car2, car3))
.then (value => (
結果を '$(value) レース結果とします。';)}

.catch(arr => {
console.log("レースはキャンセルされました。", err);
});
Promise.raceが実行されたとき、resultの値は何ですか？
A. レース中に1号車がクラッシュしました。
B. レースは中止です。
C. 2号車がレースを完走しました。
D. 3号車がレースを完走
Answer: (解答を表示する**)**

最新問題**: 21**
オブジェクトを文字列にシリアル化したり、JSON文字列をオブジェクトにデシリアル化したりするために使用できるJavaScriptメソッドはどれですか？
A. JSON.parseとJSONの逆シリアル化
B. JSON.encodeとJSON.decode
C. JSON、シリアライズおよびJSON、デシリアライズ
D. JSON.StringifyとJSON.parse
Answer: (解答を表示する**)**

最新問題**: 22**
A developer at Universal Containers creates a new landing page based on HTML, CSS, and JavaScript.
To ensure that visitors have a good experience, a script named personalizeWebsiteContent needs to be executed to do some custom initialization when the webpage is fully loaded with HTML content and all related files.
Which statement should be used to call personalizeWebsiteContent based on the above business requirement?
A. document.addEventListener(' DOMContentLoaded ' , personalizeWebsiteContent);
B. document.addEventListener(' onDOMContentLoaded ' , personalizeWebsiteContent);
C. window.addEventListener(' load ' , personalizeWebsiteContent);
D. window.addEventListener(' onload ' , personalizeWebsiteContent);
Answer: (解答を表示する**)**
Two key browser events:
* DOMContentLoaded (fired on document): Fired when the HTML document has been completely loaded and parsed, without waiting for stylesheets, images, and subresources to finish loading.
* load (fired on window): Fired when the entire page is fully loaded, including all dependent resources such as stylesheets and images.
The requirement states:
"... when the webpage is fully loaded with HTML content and all related files." That matches the semantics of the load event on the window object.
So the correct code is:
window.addEventListener(' load ' , personalizeWebsiteContent);
Why others are incorrect:
* A: DOMContentLoaded fires earlier, when HTML is parsed, but before all related files (images, some styles, etc.) are guaranteed to be loaded.
* B: ' onDOMContentLoaded ' is not a valid event name; the event is ' DOMContentLoaded ' .
* D: When using addEventListener, the event type is ' load ' , not ' onload ' . ' onload ' is the name of the handler property (window.onload = ...), not the event string.
Relevant concepts: window.load event, DOMContentLoaded, addEventListener syntax, page load lifecycle.

最新問題: **23**
開発者は、以下のブロックのように、解析済みのJSON文字列を使用してユーザー情報を操作します。



オブジェクト内のメール属性にアクセスするオプションはどれですか？ 2つ選択してください）
A. ユーザー情報。メール
B. ユーザー情報（メールアドレス）
C. ユーザー情報。("電子メール") を取得します
D. ユーザー情報（メールアドレス」）
Answer: A,D (メーッセージを残す**)**

最新問題: **24**
以下のコードを参照してください。
01 function Tiger(){
02 this.Type = 'Cat';
03 this.size = 'large';
04 }
05
06 let tony = new Tiger();
07 tony.roar = () =>{
08 console.log('彼らは素晴らしい1');
09 };
10
11 function Lion(){
12 this.type = 'Cat';
13 this.size = 'large';
14 }
15
16 let leo = new Lion();
17 //ここにコードを挿入
18 leo.roar();
18行目の関数呼び出しを有効にするために、17行目に挿入できる2つのステートメントはどれですか？
2つ選択してください。
A. Leo.prototype.roar = () => { console.log('かなり良いですね:'); };
B. Leo.roar = () => { console.log('かなり良いですね:'); };
C. Object.assign(leo,Tiger);
D. Object.assign(leo,tony);
Answer: (解答を表示する**)**

最新問題**: 25**
開発者は配列の要素の合計を計算する関数を作成する必要がありますが、コードを実行するたびにundefinedエラーが発生します。開発者は、以下のコードのどこに問題があるのかを見つける必要があります。
const sumFunction = arr => {
return arr.reduce((result, current) => {
//
結果 += 現在の値;
//
）、10）
);
どのオプションを選択すれば、コードが期待どおりに動作しますか？
A. 4行目を result = result + current に置き換えます。
B. 3行目をif(arr.length == 0) (return 0;)に置き換えます
C. line02 を return arr.map((result, current) => (に置き換えます。
D. 5行目を戻り値に置き換えてください。
Answer: D (メーッセージを残す**)**

最新問題**: 26**
以下のコードを参照してください。

5行目でエラーが発生します。
コードの実行が完了した後、挨拶や敬称にはどのような価値があるのでしょうか？
A. 挨拶は 「さようなら」で、挨拶は 「こんにちは」です。
B. Greeting is Hello and salutation is I say hello.
C. 挨拶は Hello で、敬称は Hello, Hello です。
D. 挨拶は 「さようなら」、挨拶は 「こんにちは」です。
Answer: C (メーッセージを残す**)**

最新問題**: 27**
以下のコードを参照してください。
01 const exec = (item, delay) =>{
02 新しい 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 `並列処理が完了しました: $(result1) $(result2)$(result3)`;
08 }
}
}
どの2つのステートメントが、runParallel()関数を正しく実行しますか？
2つの回答を選択してください
A. runParallel().then(data);
B. runParallel ( ). done(function(data){
データを返す。
});
C. runParallel() .then(function(data) return data
D. Async runParallel().then(data);
Answer: B,C (メーッセージを残す**)**

最新問題**: 28**
ある開発者が、数値を渡すと以下の値を返すFizzBuzz関数を作成しました。
* 数字が3で割り切れる場合は Fizz」とします。
* 数字が5で割り切れる場合は ヴザー」を鳴らしてください。
* 数字が3と5の両方で割り切れる場合は ワィズバズ」。
* 数値が3または5のどちらでも割り切れない場合は、空の文字列になります。
fizzbuzz関数のシナリオを適切にテストするには、どの2つのテストケースを使用すればよいでしょうか？
2つの回答を選択してください
A. let res = fizzbuzz(5);
console.assert(res === ' ');
B. let res = fizzbuzz(15);
console.assert(res === 'fizzbuzz')
C. let res = fizzbuzz(Infinity);
console.assert(res === ' ')
D. let res = fizzbuzz(3);
console.assert(res === 'buzz')
Answer: B,C,D (メーッセージを残す**)**

最新問題**: 29**
開発者は、チームがこれから作成するバックエンドサーバーにNode.jsを使用することでメリットが得られることを経営陣に説得しようとしている。このサーバーは、チームが既にHTML、CSS、JavaScriptを使用して構築したWebサイトからのAPIリクエストを処理するWebサーバーとなる予定だ。
開発者が上司を説得するために使えるNode.jsの3つの利点は何ですか？
3つの回答を選択してください：
A. 新しい言語を学ぶことを避けるために、サーバー側のJavaScriptコードを実行します。
B. パフォーマンスの高いリクエスト処理のためのユーザー非ブロッキング機能。
C. 独自のパッケージマネージャを使用してインストールし、サードパーティライブラリをインストールおよび管理します。

D. 数年に一度のメジャーリリースで安定性を確保します。
E. 実行前にコードの静的解析を行い、実行時エラーを検出します。
Answer: B,C,E (メーッセージを残す)

最新問題**: 30**
以下のコードを参照してください。

コード実行後のresultの値は何ですか？
A. 10
B. 5
C. NaN
D. 未定義
Answer: B (メーッセージを残す)

最新問題**: 31**
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. 2.0.0
C. 1.2.3
D. 1.1.4
Answer: A (メーッセージを残す)

最新問題**: 32**
以下の配列を参照してください。
arr = [1, 2, 3, 4, 5]とする。
どの3つの選択肢を選べば、xの評価値は[1,2]になるでしょうか？
3つ選択してください
A. let x = arr.slice(2);
B. let x =arr.splice(0, 2);

C. let x = arr.filter ((a) => 2 }) ;
D. let x arr.filter((a) => (return a <= 2 });
E. x = arr.slice(0, 2) とする。
Answer: B,D,E (メーッセージを残す)

最新問題**: 33**
Cloud Kicks has a class to represent item for sale in an online store, as shown below:

```
01 class Item {
02   constructor(name, price) {
03     this.name = name;
04     this.price = price;
05   }
06
07   formattedPrice() {
08     return "$" + String(this.price);
09   }
10 }
```

A new business requirement comes in that request a clothingitem class, that should have all of the properties and methods of the item class, but will also have properties that are specific top clothes.
Which line of code properly declares the clothingitem class such that it inherits from item?
A. Class clothingitem implements item (
B. Class clothing item super item (
C. Class clothingitem extends item (
D. Class Clothingitem (
Answer: C (メーッセージを残す)

最新問題**: 34**
開発者が関数を記述する方法は2つあります。

```
Option A:
01 function Monster() {
02   this.growl = () => {
03     console.log("Grrr");
04   }
05 }

Option B:
01 function Monster() {};
02 Monster.prototype.growl = () => {
03   console.log("Grrr");
04 }
```

開発者は選択肢を決定した後、1000個のモンスターオブジェクトを作成する。
オプションAとオプションBで作成されるグロウルメソッドはいくつありますか？
A. オプションAでは1つのグロウルメソッドが作成されます。オプションBでは1000のグロウルメソッドが作成されます。
B. オプションA用に1000個のグロウルメソッドが作成されます。オプションB用に1個のグロウルメソッドが作成されます。
C. どのオプションを使用しても、1000 の唸り方が作成されます。
D. どのオプションを使用しても、1 つの唸り方が作成されます。
Answer: B (メーッセージを残す)

最新問題**: 35**

上記のコードをJavaScriptクラス形式にリファクタリングするという要件を踏まえると、どのクラス定義が正しいでしょうか？

A.
```
function Vehicle(name, price) {
  this.name = name;
  this.price = price;
}
Vehicle.prototype.priceInfo = function () {
  return `Cost of the ${this.name} is ${this.price}$`;
}
var ford = new Vehicle('Ford Fiesta', '20,000');
```

```
class Vehicle {
  constructor(name, price) {
    this.name = name;
    this.price = price;
  }
  priceInfo() {
    return `Cost of the ${this.name} is ${this.price}$`;
  }
}
```

B.
```
class Vehicle {
  constructor() {
    this.name = name;
    this.price = price;
  }
  priceInfo() {
    return `Cost of the ${this.name} is ${this.price}$`
  }
}
```

C.
```
class Vehicle {
  vehicle(name, price) {
    this.name = name;
    this.price = price;
  }
  priceInfo() {
    return `Cost of the ${this.name} is ${this.price}$`;
  }
}
```

D.
```
class Vehicle {
  constructor(name, price) {
    this.name = name;
    this.price = price;
  }
  priceInfo() {
    return `Cost of the ${this.name} is ${this.price}$`;
  }
}
```

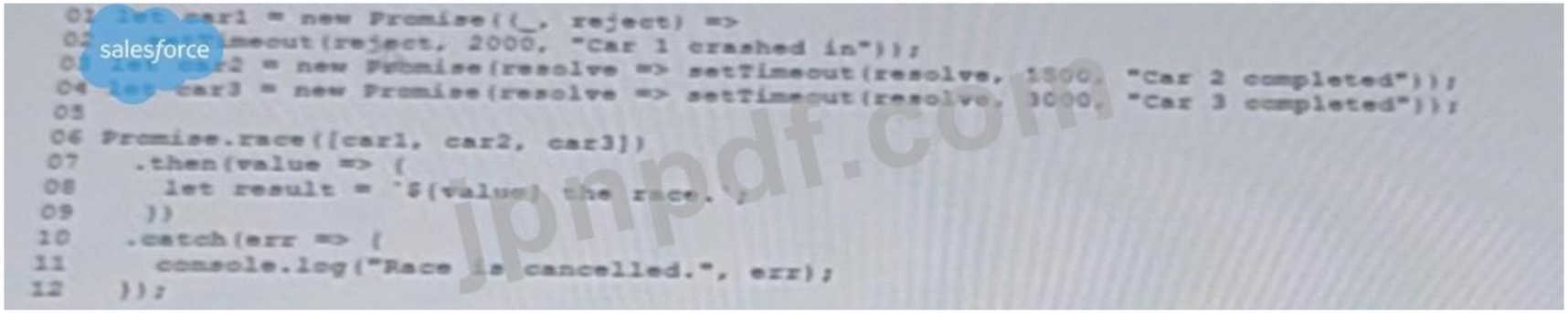
Answer: D (メーッセージを残す)

最新問題: **36**
ある開発者が、単一のサービスを提供する新しいブラウザアプリケーションの作成を主導している。
ページアプリケーション。チームは新しいWebフレームワークMinimalsit.jsを使用したいと考えている。リード
開発者は、既に実績のあるウェブフレームワークを推奨したいと考えている。
その周辺にはコミュニティが存在する。
リード開発者はどの2つのフレームワークを推奨すべきでしょうか？

2つの回答を選択してください
A. 表示
B. Angular
C. 軍事
D. エクスプレス
Answer: (解答を表示する**)**

最新問題**: 37**
Given the code below:
const copy = JSON.stringify([new String(' false '), new Bollean(false), undefined]);
What is the value of copy?
A. -- [\"false\" , false, undefined]--
B. -- [\"false\" , { }]--
C. -- [false, { }]--
D. -- [\"false\" ,false, null]--
Answer: D (メーッセージを残す**)**

最新問題**: 38**
以下のコードを参照してください。



Promise.race が実行されたとき、result の値は何ですか？
A. レースは中止です。
B. 2号車がレースを完走しました
C. 1号車がレース中にクラッシュしました
D. 3号車がレースを完走しました
Answer: C (メーッセージを残す**)**

最新問題**: 39**
Universal Containersは最近、クラウドファンディングキャンペーンを実施するための新しいランディングページを公開しました。このページは外部ライブラリを使用してサードパーティの広告を表示します。
ページが完全に読み込まれると、以下のコードのように、DOM内にランダムに配置された50個以上の新しいHTMLアイテムが作成されます。
すべての要素には同じ ad-library-item クラスが含まれており、デフォルトでは非表示になっていますが、ユーザーがページを移動する際にランダムに表示されます。
A. DOM インスペクターを使用して、クラス ad-library-item を含むすべての要素を削除します。

B. ブラウザを使用して、クラス ad-library-item を含むすべての要素を削除するスクリプトを実行します。
C. DOMインスペクターを使用して、ロードイベントの発生を防止します。
D. ブラウザのコンソールを使用して、ロードイベントの発生を防ぐスクリプトを実行します。
Answer: A (メーッセージを残す**)**

最新問題**: 40**
コードを参照してください。

```
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 (Aninal.protctype);

let pet1 = new Pet ();
```

上記のコードにおいて、pet1に設定されているプロパティはどれですか？3つ選択してください。
A. サイズ
B. 名前
C. タイプ
D. オーナー
E. canTalk
Answer: A,C,E (メーッセージを残す**)**

最新問題**: 41**
開発者は、cat という引数を1つ取る is Dog という関数を持っています。この関数を1分ごとに実行するようにスケジュールしたいと考えています。
この関数をスケジュールするための正しい構文は何ですか？
Answer:
setInterval(isDog, 60000,'cat');

最新問題**: 42**
ユーザー受け入れテスト後、開発者はユーザーの位置情報に基づいてウェブページの背景を変更するよう依頼された。この変更は実装され、テストのために展開された。
テスターからの報告によると、背景は変化しないものの、開発者のコンピューターで表示すると正常に動作するとのことです。
正確な結果を得るために役立つ2つの行動はどれですか？
2つの回答を選択してください
A. テスターはブラウザのキャッシュをクリアする必要があります。
B. 開発者はコードを修正する必要があります。
C. 開発者はブラウザの更新設定を確認する必要があります。
D. テスターはブラウザのキャッシュを無効にする必要があります。
Answer: A,C (メーッセージを残す**)**

最新問題**: 43**
以下のJavaScriptコードを参照してください。
01 function filterDOM (searchString) {
02 const parsedSearchString = searchString && searchString.toLowerCase() ;
03 document.quesrySelectorAll(' .account') . forEach(account => (
04 const accountName = account.innerHTML.toLOwerCase();
05 アカウント。Style.display = accountName.includes(parsedSearchString) ? /*挿入
コード*/;
06)};
07 }
5行目のプレースホルダーコメントをどのコードに置き換えれば、アカウントを非表示にできるでしょうか？
検索文字列と一致しませんか？
A. '表示' : '非表示'
B. 'ブロック' : 'なし'
C. '名前' : 'ブロック'

D. '非表示' : '表示'
Answer: B (メーッセージを残す**)**

最新問題**: 44**
以下のコードが与えられた場合：

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

2行目の出力は？
A. "x"
B. "未定義"
C. "null"'
D. 才ブジェクト」
Answer: D (メーッセージを残す**)**

最新問題**: 45**
コードを参照してください。
01 function execute() {
02 return new Promise((resolve, reject) => reject());
03 }
04 let promise = execute();
05
06 約束
07 .then(() = > console.log(' Resolved1 '))
08 .then(() = > console.log(' Resolved2 '))
09 .then(() = > console.log(' Resolved3 '))
10 .catch(() => console.log('拒否されました'))
11 .then(() = > console.log(' Resolved4 '));
execute関数内のPromiseが拒否された場合、どのような結果になりますか？
A. 解決済み1 解決済み2 解決済み3 却下 解決済み4
B. 拒否
C. 解決済み1 解決済み2 解決済み3 解決済み4
D. 拒否解決4
Answer: D (メーッセージを残す**)**
* execute() は、reject() を即座に呼び出す Promise を返します。そのため、Promise は拒否された状態から始まります。
* Promiseが拒否され、拒否ハンドラなしで.then()呼び出しを連鎖させた場合、.catch()に遭遇するまで、それらの.then()コールバックはすべてスキップされます。
約束
その後(...) // スキップ
その後(...) // スキップ
その後(...) // スキップ
catch(...) // 実行されました

then(...); // catch の後に実行される
* 実行:
* .then(() = > console.log(' Resolved1 ')) はスキップされます。
* .then(() = > console.log(' Resolved2 ')) はスキップされます。
* .then(() = > console.log(' Resolved3 ')) はスキップされます。
* .catch(() = > console.log(' Rejected ')) が実行され、Rejected がログに記録されます。
* .catch() は解決済みの Promise を返します (明示的な戻り値がないため、undefined になります)。そのため、次の .then() が実行されます。
* .then(() = > console.log(' Resolved4 ')) は Resolved4 をログに出力します。
最終出力:
拒否されました
解決済み4
これは選択肢Dに該当します。

最新問題**: 46**
通常商品とセール商品を表すクラスを作成しました。コード :
01 let regItem = new Item(' Scarf ' , 55);
02 let saleItem = new SaleItem(' シャツ ' , 80, .1);
03 Item.prototype.description = function() { return ' これは ' + this.name; です。}
04 console.log(regItem.description());
05 console.log(saleItem.description());
06
07 SaleItem.prototype.description = function() { return ' これは割引された ' + this.name; }
08 console.log(regItem.description());
09 console.log(saleItem.description());
出力は何ですか？
A. これはスカーフです
キャッチされない TypeError: saleItem.description は関数ではありません
これはシャツです
こちらは割引価格のシャツです
B. これはスカーフです
これはシャツです
これはスカーフです
こちらは割引価格のシャツです
C. これはスカーフです
キャッチされない TypeError: saleItem.description は関数ではありません
これはスカーフです
こちらは割引価格のシャツです
D. これはスカーフです
これはシャツです
こちらは割引価格のスカーフです

こちらは割引価格のシャツです
Answer: B (メーッセージを残す)
* 3行目で、開発者は以下を割り当てます。
Item.prototype.description = function() { return ' これは ' + this.name; です。} これは、プロトタイプ チェーンに Item.prototype が含まれるすべてのオブジェクトに影響します。
* regItem は Item から継承し、このメソッドを取得します。
* saleItem は SaleItem のインスタンスであるため、Item.prototype も継承します (SaleItem は Item からプロトタイプ継承を使用しているため)。そのため、現時点ではこのメソッドも持っています。
4行目と5行目の出力：
* regItem.description() # "これはスカーフです"
* saleItem.description() # "これはシャツです"
* 7行目で、開発者はSaleItem.prototypeのメソッドをオーバーライドしています。
SaleItem.prototype.description = function() {
return ' これは割引価格の ' + this.name;
}
ここから先は：
* regItem は依然として Item.prototype バージョンを使用しています
* saleItem はオーバーライドされた SaleItem.prototype バージョンを使用します
8行目と9行目の出力：
* regItem.description() # "これはスカーフです"
* saleItem.description() # "これは割引価格のシャツです"
すべての結果をまとめると：
これはスカーフです
これはシャツです
これはスカーフです
こちらは割引価格のシャツです
これはオプションBに該当します。
JavaScriptの知識リファレンス（テキストのみ）
* コンストラクタ関数から作成されたオブジェクトはプロトタイプチェーンを使用します。
サブクラスのプロトタイプメソッドをオーバーライドしても、親クラスのプロトタイプには影響しません。
* インスタンスは、プロトタイプチェーン上で最も具体的なバージョンのメソッドを継承します。

最新問題**: 47**
開発者は配列の要素の合計を計算する関数を作成する必要があるが、コードを実行するたびに「undefined」というエラーが発生する。開発者は、以下のコードに何が不足しているかを特定する必要がある。
01 const sumFunction = arr = > {
02 return arr.reduce((result, current) = > {

03 //
04 結果 += 現在の値;
05 //
06 }, 10);
07 };
どの行を置換すれば、コードが期待どおりに動作するようになりますか？
A. 03 if(arr.length == 0) { return 0; }
B. 04 結果 = 結果 + 現在値;
C. 02 arr.map((result, current) = > {
D. 05 戻り結果;
Answer: (解答を表示する**)**
* reduceコールバックでは、新しいアキュムレータ値を返さなければなりません。
* 現在：
(結果、現在) = > {
result += current;
// 戻り値なし
}
これは毎回 undefined を返すため、次の反復処理でアキュムレータが undefined になり、誤った結果につながります。
結果を返すことで修正します。
const sumFunction = arr = > {
return arr.reduce((result, current) = > {
result += current;
return result; // 5行目
}, 10);
};
選択肢Dは正しくリターンを追加します。
オプションA/B/Cは、リデューサーにおける戻り値の欠落を修正しないため、根本的な問題を解決しません。

最新問題**: 48**
ある開発者が、既存のクライアントからのAPIリクエストに対応する新しいWebサーバーの構築をチーム内で主導している。
チームはNode.js上で動作するWebサーバーを求めており、新しいWebフレームワークであるMinimalist.jsを使いたいと考えている。一方、リード開発者は、既にコミュニティが存在する、より実績のあるバックエンドフレームワークを推奨したいと考えている。
リード開発者はどの2つのフレームワークを推奨するだろうか？
2つの回答を選択してください
A. それで
B. エクスプレス
C. Angular
D. ギャツビー
Answer: (解答を表示する**)**

最新問題: **49**
開発者が関数内のエラーを処理しようとしています。
エラーを他の箇所に伝播させることなく処理するための正しいアプローチを示しているコードセグメントはどれですか？

A.
```
try {
  doSomething();
} catch (error) {
  return null;
}
```

B.
```
try {
  doSomething();
} catch (error) {
  processError(error);
}
```

C.
```
try {
  doSomething();
} catch (error) {
  throw new Error('Error found');
}
```

D.
```
try {
  doSomething();
} catch (error) {
  return error;
}
```

Answer: B (メーッセージを残す)

最新問題: **50**
以下の配列を参照してください。
arr1 = [1, 2, 3, 4, 5] とする。

どの 2 行のコードによって、2 番目の配列 arr2 が作成され、arr2 が
arr1への参照？

A. arr2 = arr1.sort();

B. arr2 = arr1 とする。

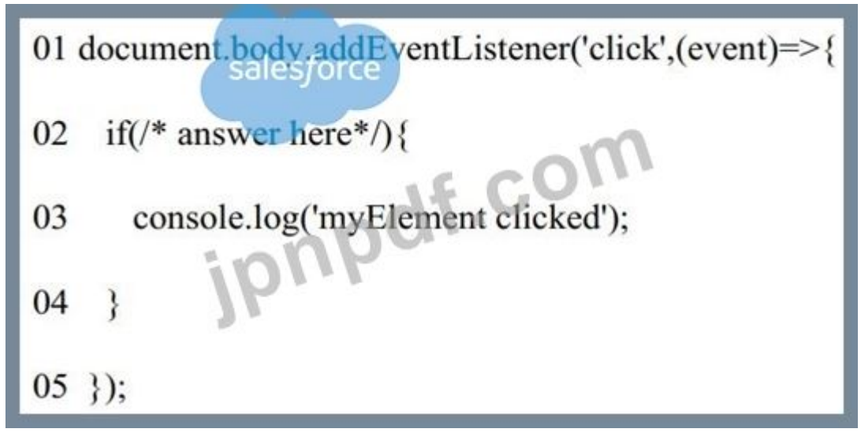
C. arr2 = arr1.slice(0, 5);

D. arr2 = Array.from(arr1);

Answer: C,D (メーッセージを残す**)**

最新問題**: 51**

以下のコードを参照してください。



2行目の条件文を置き換えることで、開発者がページ上の特定の要素myElementがクリックされたことを正しく判断できるのはどれですか？

Answer:

event.target.id =='myElement'

最新問題**: 52**

Refer to the following array:

Let arr1 = [1, 2, 3, 4, 5];

Which two lines of code result in a second array, arr2 being created such that arr2 is not a reference to arr1?

A. Let arr2 = arr1;

B. Let arr2 = arr1.sort();

C. Let arr2 = Array.from(arr1);

D. Let arr2 = arr1.slice(0, 5);

Answer: C,D (メーッセージを残す)

最新問題**: 53**

以下のコードを参照してください。

01 const server = require('server');

02 /* ここにコードを挿入 */

開発者がWebサーバーを作成するライブラリをインポートします。インポートされたライブラリはイベントとコールバックを使用してサーバーを起動します。イベントを設定してWebサーバーを起動するには、3行目にどのコードを挿入する必要がありますか？

A. server()

B. server.on(' connect ' , (port) => {console.log('Listening on ' , port) ;})

C. Server.start();

D. console.log('Listening on ', port);

E. serve((port) => (

Answer: (解答を表示する)

最新問題**: 54**

1 つのページには 50 個以上の < div class= " ad-library-item " > 要素が読み込まれ、すべて広告です。

開発者はそれらを迅速かつ一時的に削除したいと考えている。

オプション:

A. ブラウザのコンソールを使用して、ロードイベントの発生を防ぐスクリプトを実行します。

B. DOMインスペクターを使用して、ロードイベントの発生を防止します。

C. ブラウザのコンソールを使用して、クラス ad-library-item を含むすべての要素を削除するスクリプトを実行します。
D. DOM インスペクターを使用して、クラス ad-library-item を含むすべての要素を削除します。
Answer: C (メーッセージを残す**)**
目標：
「時的かつ迅速に取り外してください。」
最も手っ取り早い方法は、ブラウザのコンソールでスクリプトを実行して、そのような要素をすべて削除することです。
document.querySelectorAll(' .ad-library-item ').forEach(el = > el.remove()); これにより、50 個以上の要素が即座に削除され、ページの再読み込みは不要になります。
オプション分析：
* A & B : ロード イベントを防止することは信頼性が低く、既に作成された DOM ノードを削除しません。
* C : 正しいコンソールコードは、一致するすべての DOM 要素を即座に削除できます。
* D：可能だが非常に時間がかかる。インスペクターで50個以上の要素を1つずつ削除するのは 速い」とは言えない。したがって、Cが最適な答えである。
JavaScriptの知識リファレンス テキストのみ）
* ブラウザのコンソールは、DOMを操作する任意のJavaScriptを実行できます。
* document.querySelectorAll() は、一致するすべての DOM ノードを返します。
* ノードは .remove() を使用して削除できます。

最新問題**: 55**
開発者は、一定間隔でコードを繰り返し実行するために、どの関数を使用すべきでしょうか？
A. setTimeout
B. setIntervel
C. setPeriod
D. setInteria
Answer: B (メーッセージを残す**)**

最新問題**: 56**
Utills という名前のモジュールをインポートする以下のコードを参照してください。
上記のコードがエラーなく実行されるようにするには、どの 2 つの Utill 実装 (je export foo および bar) が必要ですか？
2つの回答を選択してください
A. //FooUtill.jsとbarUtils.jsが存在する
import (foo) from ,/Path/footUtils.js,:
エクスポート (foo、bar)
B. デフォルトクラスのエクスポート (
Foo ( ) (return 'foo ,;)
バー（）リターン ;バー ； ）
C. Const foo = ( ) => (return 'foo; ;)
const bar => => { return 'bar ';}
デフォルトのfoo、barをエクスポートします。
D. Const foo = () => (return 'foo ' ;)
const bar => (return 'bar' ;)
エクスポート (foo、bar)

Answer: B (メッセージを残す)

最新問題**: 57**
Universal Containers社の開発者が、HTML、CSS、JavaScriptをベースにした新しいランディングページを作成しています。このウェブサイトには、ページを開いた際に読み込まれる複数の外部リソースが含まれています。
訪問者に快適な体験を提供するために、ウェブページが読み込まれたときに personalizeWebsiteContent という名前のスクリプトを実行する必要があり、リソースが利用可能になるまで待つ必要はありません。
上記のビジネス要件に基づいて、personalizeWebsiteContentを呼び出すには、どのステートメントを使用すべきでしょうか？
A. windows, addEventListener('onload', personalizeWebsiteContent);
B. windows,addEventListener('DOMContent Loaded ', personalizeWebsiteContent);
C. windows,addEventListener('onDOMCContentLoaded', personalizeWebsiteContent);
D. windows, addEventListener('load', personalizeWebsiteContent);
Answer: D (メッセージを残す)

最新問題**: 58**
以下のオブジェクトを参照してください。

```
01 const dog = {
02    firstName: 'Beau',
03    lastName: 'Boo',
04    get fullName() {
05        return this.firstName + ' ' + this.lastName;
06    }
07 };
```

開発者はどのようにして犬の fullName プロパティにアクセスできますか？
A. 犬、機能、フルネーム
B. Dog.fullName ( )
C. 犬、取得、フルネーム
D. 犬の名前
Answer: (解答を表示する)

最新問題**: 59**
以下のコード宣言を参照してください。

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

どの 3つの式が文字列 『JavaScript」を返しますか？
3つの回答を選択してください
A. $(str1) $ (str2} ';
B. Str1 + str2;
C. Concat (str1, str2);

D. str1.join(str2);
E. str1.concat(str2);
Answer: A,B,E (メーッセージを残す**)**

最新問題**: 60**
開発者は、いくつかの値を加算する関数を実装する。
function sum(num1, num2, num3) {
if (num3 === undefined) {
num3 = 0;
}
return num1 + num2 + num3;
}
開発者は、この関数で戻り値10を取得するために、どの3つのオプションを選択できますか？
A. 合計(5)(5);
B. sum()(10);
C. sum(5, 5, 0);
D. sum(10, 0);
E. sum(5, 2, 3);
Answer: C,D,E (メーッセージを残す**)**
検証済みの正解はC、D、Eです。
この関数は通常の関数であり、カリー化された関数ではありません。
function sum(num1, num2, num3) {
if (num3 === undefined) {
num3 = 0;
}
return num1 + num2 + num3;
}
値は同じ関数呼び出し内で渡されることを想定しています。
sum(num1, num2, num3);
それでは、修正後の有効な選択肢を確認してください。
オプションC：
sum(5, 5, 0);
計算：
5 + 5 + 0
結果：
10
したがって、Cが正解です。
選択肢D：
sum(10, 0);
ここではnum3が指定されていないため、未定義です。

この関数は以下をチェックします。
if (num3 === undefined) {
num3 = 0;
}
したがって、計算式は次のようになります。
10 + 0 + 0
結果：
10
したがって、Dが正解です。
選択肢E：
合計(5, 2, 3);
計算：
5 + 2 + 3
結果：
10
したがって、Eが正解です。
AとBが元の記述どおりに誤りである理由：
合計(5)(5);
これはまず sum(5) を呼び出します。num2 が存在しないため、結果は NaN になります。次に JavaScript はその返された値を関数として呼び出そうとしますが、TypeError が発生します。
sum()(10);
これもまた、最初に sum() を呼び出し、NaN を生成し、次に NaN を関数として呼び出そうとします。
これらのスタイルは、sumがカリー化された関数として記述されている場合にのみ機能しますが、提示された実装はカリー化されていません。
したがって、検証済みの正解はC、D、Eです。

最新問題**: 61**
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. Replace (/^\a\a'/' ' ') ;
B. trimStart (searchString) ;
C. searchString. trimstart ( ) ;
D. searchString. trimEnd ( ) ;
Answer: (解答を表示する**)**

最新問題: **62**
開発者は以下を実行します。
ドキュメントクッキー;
document.cookie = ' key=John Smith ' ;
その行動とはどのようなものか？
A. クッキーが読み取られ、キー値が設定されます。残りのクッキーは影響を受けません。
B. Cookieは読み込まれますが、値がURLエンコードされていないため、キー値は設定されません。
C. 行 01 は document.cookies であるべきですが、キー値が設定され、すべてのクッキーが削除されるため、クッキーは読み取られません。
D. クッキーが読み取られ、キー値が設定され、すべてのクッキーが削除されます。
Answer: A (メーッセージを残す**)**
* document.cookie はクッキー文字列を取得します。
* document.cookie = ' key=John Smith ' を設定すると、その 1 つの Cookie のみが追加または更新されます。
重要な詳細：
* document.cookie への書き込みは、すべてのクッキーを上書きするわけではありません。
* ブラウザはURLエンコードを必須とはしていませんが、推奨します。エンコードされていないスペースも許可されており、自動的に処理されます。
* document.cookies は存在しません。正しいプロパティは document.cookie です。
したがって、正しい動作は次のとおりです。
* クッキーが読み込まれました。
* 新しいクッキーキー 『John Smith」が設定されました。
* 既存のクッキーはそのまま残ります。
JavaScriptの知識リファレンス テキストのみ）
* document.cookie を書き込むと、クッキーが追加または更新されますが、置き換えられるわけではありません。
* document.cookie は、クッキー文字列のゲッターとセッターの両方です。

最新問題: **63**
Considering type coercion, What does the following expression evaluate to?
True + '13 ' + NaN
A. 113NaN
B. 'true13'
C. 'true13NaN'
D. 14
Answer: (解答を表示する**)**

最新問題: **64**
以下のJavaScriptコードを参照してください。

```
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.backgroundColor = accountName.includes(parsedSearchString) ? /* Insert code
here */;
06   });
07 }
```

検索文字列に一致するアカウントを強調表示するには、5行目のプレースホルダーコメントをどのコードに置き換える必要がありますか？

A. '黄色' : null

B. 'none1 : "黄色'

C. '黄色 : 'なし'

D. null : '黄色'

Answer: (解答を表示する**)**

最新問題: 65

開発者がインポートするもの：

import printPrice from ' /path/PricePrettyPrint.js ' ;

このインポートが機能するために、printPrice に関してどのような条件が満たされていなければなりませんか？

A. printPrice は名前付きエクスポートである必要があります

B. printPrice はすべてのエクスポートである必要があります

C. printPrice はデフォルトのエクスポートである必要があります

D. printPriceは複数エクスポートである必要があります

Answer: (解答を表示する**)**

構文:

import printPrice from ' module ' ;

これは、モジュールがその関数をデフォルトエクスポートとしてエクスポートする必要があることを意味します。

export default function printPrice() { ... }

なぜ他の人たちは間違っているのか：

* 名前付きエクスポートには中括弧が必要です。

* import { printPrice } from ' module ' ;

* 「all export」と「multi export」はJavaScriptの用語ではありません。

したがって、printPrice はデフォルトのエクスポートである必要があります。

JavaScriptの知識リファレンス（テキストのみ）

* デフォルトのインポートでは、`import name from ' module '` を使用します。

* 名前付きインポートには中括弧が必要です: import { name } from ' module ' 。

最新問題: **66**

```
02   <table onclick="console.log('Table log');">
03     <tr id="row1">
04      <td>Click me!</td>
05     </tr>
06   </table>
07   <script>
08     function printMessage(event) {
09        console.log('Row log');
10        event.stopPropagation();
11     }
12
13     let elem = document.getElementById('row1');
14     elem.addEventListener('click', printMessage, false);
15   </script>
16 </html>
```


Click me!」がクリックされたときにコンソールに以下の内容がログ出力されるようにするには、どのコードを変更すればよいですか？
行ログ
テーブルログ
A. 10行目を削除
B. 13行目と14行目を削除します。
C. 10行目をevent.stopPropagation(false)に変更します。
D. 14行目を elem.addEventListener ('click', printMessage, true); に変更します。
Answer: A (メーッセージを残す**)**

最新問題: **67**
コードを参照してください。

```
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 (Aninal.protctype);

let pet1 = new Pet ();
```


上記のコードにおいて、pet1に設定されているプロパティはどれですか？3つ選択してください。

A. タイプ

B. 名前

C. canTalk
D. オーナー
E. サイズ
Answer: A,C,E (メーッセージを残す**)**

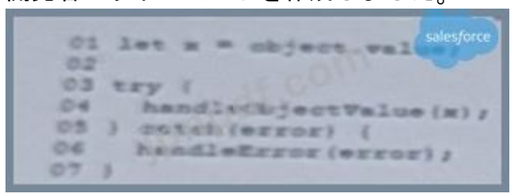
最新問題**: 68**
開発者は、コンソールにカスタムメッセージをログ出力するための汎用関数を作成します。そのためには、以下の関数を実装します。
01 関数 logStatus(ステータス){
02 console./*ここに回答を記述*/{'アイテムの状態は: %s', status};
03 }
2行目で文字列置換を使用できるコンソールログ記録方法はどれですか？ 3つ選択）
A. 主張する
B. ログ
C. 情報
D. エラー
E. メッセージ
Answer: A,C (メーッセージを残す**)**

最新問題**: 69**
bar、awesomeは人気のJavaScriptモジュールです。npmに公開されているバージョンは以下のとおりです。
1.2
1.3.1
1.3.5
1.4.0
Universal Containersのチームは、このモジュールを多くのプロジェクトで使用しています。ある特定のプロジェクトでは、以下のパッケージとJSON定義が使用されています。

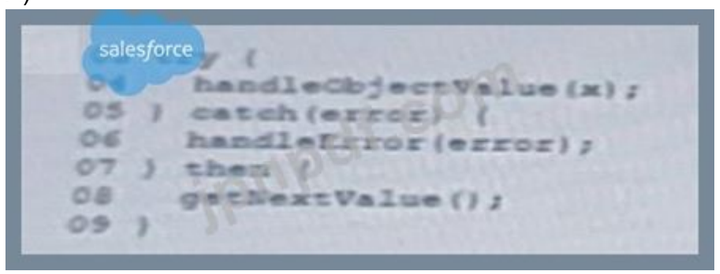
```
{
  "name": "UC Project Extra",
  "version": "0.0.5",
  "dependencies": {
    "bar.awesome": "~1.3.0"
  }
}
```


開発者は、npm install コマンドを実行します。
インストールされているbar.awesomeのバージョンはどれですか？
A. 1.4.0
B. 1.3.1
C. バージョン130が見つからないため、コマンドが失敗します。
D. 1.3.5
Answer: D (メーッセージを残す**)**

最新問題: **70**
開発者が以下のコードを作成しました。

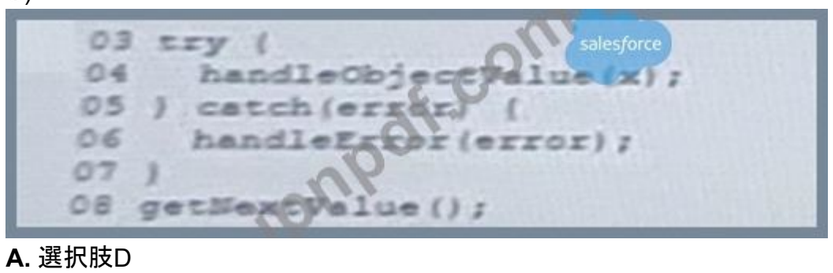


開発者は handleObjectvalue() の後に getNextValue 関数を実行しますが、エラーが発生した場合は getNextValues() を実行したくありません。
開発者は、この動作を確実にするために、どのようにコードを変更すればよいでしょうか？
A)



B)

C)



A. 選択肢D
B. オプションC
C. オプションB
D. オプションA
Answer: (解答を表示する)

最新問題: **71**
開発者は、ユーザーから報告されたバグを修正するように依頼されました。そのために、開発者は以下を追加します。
デバッグ用のブレークポイント。

関数 Car (maxSpeed, color){
This.maxspeed = masSpeed;
this.color = color;
let carSpeed = document.getElementById('CarSpeed');
デバッガ;
let fourWheels = new Car(carSpeed.value, 'red');
コードの実行が6行目のブレークポイントで停止したとき、どのような2種類の情報が得られますか？
ブラウザのコンソールで利用できますか？
2つの回答を選択してください：
A. The information stored in the window.localStorage property
B. Car オブジェクト用に作成されたインスタンスの数を表示する変数。
C. The style, event listeners and other attributes applied to the carSpeed DOM element
D. carSpeed変数とfourWheels変数の値
Answer: A,C (メーッセージを残す**)**

最新問題**: 72**
ある開発者が、チームがこれから作成するバックエンドサーバーにNode.jsを使用することでメリットが得られることを経営陣に説得しようとしている。このサーバーは、チームが既にHTML、CSS、JavaScriptを使って構築したウェブサイトからのAPIリクエストを処理するウェブサーバーとなる予定だ。
開発者が上司を説得するために使えるNode.jsの3つの利点は何ですか？
開発者が上司を説得するために使えるNode.jsの3つの利点は何ですか？
A. 数年に一度のメジャーリリースで安定性を確保する。
B. 独自のパッケージマネージャを使用してインストールし、サードパーティライブラリをインストールおよび管理します。
C. 新しい言語を学ぶ必要がないように、サーバー側のJavaScriptコードを実行します。
D. 実行前にコードの静的解析を実行して、実行時エラーを探します。
E. パフォーマンスの高いリクエスト処理のために、ノンブロッキング機能を使用します。
Answer: D (メーッセージを残す**)**

最新問題**: 73**
以下のコードを参照してください。
Const pi = 3.1415326、
円周率のデータ型は何ですか？
A. フロート
B. 10進数
C. ダブル
D. 番号
Answer: D (メーッセージを残す**)**

最新問題**: 74**
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);
```

別の開発者がsum3の動作を変更し、配列内のすべての数値を合計するようにしました。
更新されたsum3関数でテストを実行すると、どのような2つの結果が得られますか？
A. 2行目のアサーションが失敗しました。
B. 5行目のアサーションは合格です。
C. 5行目のアサーションが失敗しました。
D. 2行目のアサーションは成功しました。
Answer: C,D (メーッセージを残す**)**
新しい動作: sum3 はすべての要素の合計を返すようになりました。
* 1行目: sum3([1, 2, 3]) # 1 + 2 + 3 = 6
* 2行目のアサーション: res === 6 # 合格。
* 4行目: sum3([1, 2, 3, 4]) # 1 + 2 + 3 + 4 = 10
* 5行目のアサーション: res === 6 # 10 === 6 は偽であるため、アサーションは失敗します。
それで：
* 行 02 のアサーションが成功しました # D。
* 5行目のアサーションが失敗しました # C。

最新問題: 75
Refer to the code below:

```
01 x = 3.14;
02
03 function myFunction() {
04 ' use strict ' ;
05 y = x;
06 }
07
08 z = x;
09 myFunction();
```

Considering the implications of ' use strict ' on line 04, which three statements describe the execution of the code?
A. ' use strict ' is hoisted, so it has an effect on all lines.
B. z is equal to 3.14.
C. ' use strict ' has an effect between line 04 and the end of the file.
D. ' use strict ' has an effect only on line 05.
E. Line 05 throws an error.
Answer: (解答を表示する**)**
* Behavior of non-strict global code The script does not begin with a ' use strict ' directive at the top level, so the global code (outside any function) runs in non-strict (sloppy) mode.
* Line 01: x = 3.14; In non-strict mode, assigning to an undeclared identifier (no var, let, or const) creates an implicit global variable. So after line 01, x exists and equals 3.14.

* Line 08: z = x; This also runs in non-strict mode. Since x is already defined (from line 01), z is set to 3.14. Therefore, statement B is correct: z is equal to 3.14.
* Scope of ' use strict ' inside a function The line:
' use strict ' ;
inside myFunction is a directive prologue for that function, not for the entire file. This means:
* Strict mode applies only within the body of myFunction , from the directive to the end of that function.
* It does not retroactively affect code before the function or code outside of it.
Therefore:
* Statement A is incorrect: ' use strict ' is not "hoisted" to affect the whole file. It only affects the function body.
* ステートメント C は誤りです。厳格モードは 04 行目からファイルの末尾まで適用されるのではなく、myFunction 内でのみ適用され、01、08、09 のようなグローバル行には適用されません。
* myFunction の厳格モードでの実行 myFunction が 09 行目で呼び出されたとき:
function myFunction() {
'use strict';
y = x;
}
この機能内では：
* 本体に対して厳格モードが有効になっています。
* 厳格モードでは、宣言されていない変数（ここでは y など）への代入は許可されておらず、実行時に ReferenceError が発生します。
5行目：
y = x;
y は var、let、または const で宣言されていないため、ReferenceError がスローされます。
したがって、ステートメントEは正しい。5行目でエラーが発生する。
* ステートメント D が正しいと考えられる理由 ステートメント D は、「use strict」は 05 行目にのみ影響すると述べています。
この特定のコードの実行に関して言えば：
* myFunction 内で厳格モードの影響を受ける唯一の実行可能なステートメントは、行の代入です。
05.
* 4行目のディレクティブ自体は「通常の」実行時操作ではなく、モードを設定するディレクティブです。
* 関数内には、厳格モードで動作が異なるステートメントは他にありません。宣言されていない変数に値を代入する行のみが、厳格モードの影響を示します。
したがって、このコードスニペットの実際の実行動作を説明すると、厳格モードは5行目でのみ効果を発揮するため、この文脈ではステートメントDが正しいと言えます。
各オプションの概要：
* A: 間違いです。strict はファイル内のすべての行に適用されるわけではありません。
* B: 正解 - 非厳密なグローバルコードでは、z は 3.14 になります。
* C: 誤りです。strict は関数外のコードには影響しません。
* D: 正解 - このコードでは、厳格モードの動作上の影響は 05 行目のみです。
* E: 正しい - 厳格モードで未宣言の y に代入すると、5 行目で ReferenceError が発生します。
JavaScriptに関する知識リファレンス（説明のみ、リンクなし）：
* 非厳格（ズロッピー）モードでは、宣言されていない識別子に値を代入すると、グローバル変数が作成されます。
* 関数本体内で 'use strict' を使用すると、その関数のみで厳格モードが有効になります。
* 厳格モードでは、宣言されていない変数に値を代入すると ReferenceError が発生します。
* ディレクティブのプロローグ（'use strict'）は、それを包含する関数またはスクリプトのみに影響し、他のスコープには影響しません。

最新問題**: 76**
開発者が関数を記述する方法は2つあります。
選択肢A：
01 function Monster() {
02 this.growl = () = > {
03 console.log(" Grr! ");
04 }
05 }
選択肢B：
01 function Monster() {};
02 Monster.prototype.growl = () = > {
03 console.log(" Grr! ");
04 }
開発者はオプションを選択した後、1000個のモンスターオブジェクトを作成します。オプションAとオプションBでは、それぞれいくつの唸りメソッドが作成されますか？
A. どのオプションを使用しても、1000 の唸り方が作成されます。
B. どのオプションを使用しても、1 つの唸り方が作成されます。
C. オプションA用に1000個のグロウルメソッドが作成されます。オプションB用に1個のグロウルメソッドが作成されます。
D. オプションAには1つのグロウルメソッドが作成されます。オプションBには1000のグロウルメソッドが作成されます。
Answer: C (メーッセージを残す**)**
選択肢A：
* this.growl = () = > { ... } コンストラクタ内では、インスタンスごとに新しい関数が作成されます。
* 1000 の新しい Monster() に対して、1000 個の異なる唸り関数が得られ、それぞれがインスタンス自体に格納されます。
選択肢B：
* Monster.prototype.growl = () = > { ... } は、プロトタイプに growl をアタッチします。
* すべてのインスタンスは、プロトタイプチェーンを介して同じ単一の唸り声関数を共有します。
* 1000 インスタンスに対して、メモリ上には 1 つの growl メソッドのみが存在する。
つまり、選択肢Aは1000、選択肢Bは1です。

最新問題**: 77**
以下のコードスニペットを参照してください。
01 function getAvailableilityMessage(item) {
02 if (getAvailableility(item)) {
03 var msg = "ユーザー名が利用可能です";

04 }
05 戻りメッセージ;
06 }
getAvailableilityMessage(" newUserName ") が実行され、getAvailableility(" newUserName ") が false を返した場合、msg の戻り値は何ですか？
A. "新しいユーザー名"
B. "msgが定義されていません"
C. 未定義
D. "ユーザー名が利用可能です"
Answer: C (メーッセージを残す**)**
主な詳細：
* var は関数スコープを持ち、ブロックスコープを持ちません。
* 宣言変数 msg は関数の先頭に巻き上げられますが、初期化は if 条件が真の場合にのみ行われます。
実際には、この関数は次のように動作します。
function getAvailableilityMessage(item) {
var msg; // 巻き上げられた宣言
if (getAvailableility(item)) {
msg = "ユーザー名が利用可能です" ;
}
msgを返します。
}
さて、以下の条件が与えられています。
* getAvailableility(" newUserName ") は false を返します。
実行：
* if条件が偽であるため、本体は実行されません。
* したがって、msg は宣言されますが、代入されることはありません。
* JavaScriptでは、`var`で宣言された初期化されていない変数の値は`undefined`になります。
したがって：
return msg; // undefined を返します
他の選択肢が間違っている理由：
* A: " newUserName " - msg はパラメータ値を受け取りません。if 内で " Username available " に設定されるだけで、if は実行されません。
* B: "msg が定義されていません" - この種のエラーは、msg が宣言されていない場合に発生します。ここでは var を介して宣言されているため、定義はされていますが未定義です。
* D: "ユーザー名が利用可能です" - これにはifブランチの実行が必要ですが、getAvailableility(...)がfalseの場合は実行されません。
戻り値は次のとおりです。
未定義
学習ガイドの概念：
* 変数の巻き上げと関数のスコープ
* 初期化されていない変数はデフォルトで未定義になります
* 制御フローと条件付き初期化

最新問題**: 78**

開発者は、そのプロパティが不変で、
プロパティの追加や変更を禁止する。
この業務要件を実行するには、どの方法を用いるべきでしょうか？
A. Object.const()
B. Object.eval()
C. オブジェクトロック()
D. オブジェクト.freeze()
Answer: D (メーッセージを残す**)**

最新問題**: 79**
開発者は、新しいユーザー名を登録しようとするユーザーにメッセージを返すために、以下のコードを作成します。ユーザー名が利用可能な場合、3行目でnagという名前の変数が宣言され、値が割り当てられます。

```
01  function getAvailabilityMessage(item) {
02    if (getAvailability(item)) {
03      var msg = "Username available";
04    }
05    return msg;
06  }
```


getAvailableabilityMessage ("newUserName") が実行され、get Availability ("newUserName") が true を返した場合、msg の値は何ですか？
A. "newUserName"
B. "msgが定義されていません"
C. 未定義
D. 「ユーザー名が利用可能です」
Answer: (解答を表示する**)**

最新問題**: 80**
インポートされたモジュールの厳格モードについて、正しい選択肢はどれですか？
A. インポートされたモジュールは、厳格モードとして宣言されているかどうかに関わらず、厳格モードで動作します。
B. モジュール内の他のステートメントの前に use non-strict ステートメントを追加して、非厳格モードを有効にします。
C. インポートしたモジュールから notStrict() 関数のみを参照できます。
D. モジュール内の他のステートメントの前に use strict =false; というステートメントを追加して、非厳格モードを有効にします。
Answer: C (メーッセージを残す**)**

最新問題**: 81**
約束のある側面を正確に説明しているのは、次のうちどれですか？
A. .then() に渡されるコールバック関数の引数は省略可能です。
B. catch の後に .then() を追加することはできません。
C. .then() は元のプロミスを操作して返します。

D. .then() 関数では、コールバックが前のプロミスの結果をキャッチするため、結果を返す必要はありません。
Answer: A (メーッセージを残す)
各選択肢を評価してください。
A) .then() の引数は省略可能です。
これは正しいです。
then() のシグネチャは次のとおりです。
promise.then(onFulfilled?, onRejected?)
どちらの引数（onFulfilledとonRejected）も省略可能です。
コールバックが指定されていない場合、JavaScriptはデフォルトのパススルーハンドラを提供します。
B). .then() は catch の後に挿入できません。
正しくない。
約束は任意の順序で連鎖することをサポートします。
約束
キャッチ（.）
それから（.）;
After a catch, the chain continues normally.
C). .then() manipulates and returns the original promise.
Incorrect.
then() always returns a new promise , not the original one.
This is fundamental to promise chaining behavior.
D). Returning values in .then() is not necessary.
Incorrect.
If you want the next .then() in the chain to receive a value, the current .then() must explicitly return it:
then(value = > {
return value * 2; // passes to next .then()
})
If nothing is returned, the next .then() receives undefined.
Why A is correct
It is the only statement that accurately describes built-in Promise behavior:
then() accepts optional arguments.
JavaScript Knowledge References (text-only)
* .then(onFulfilled?, onRejected?) accepts optional handlers.
* Promise chaining creates new promises for each .then().
* catch() can be followed by additional .then() calls.
* Returning inside .then() passes values to the next step in the chain.

最新問題: 82
以下のコードを参照してください。

コード実行時のresultの値は何ですか？
A. 10 - 5
B. 5 - 5
C. 5〜10
D. 10 - 10
Answer: D (メーッセージを残す)

最新問題**: 83**
ブラウザでは、window オブジェクトはアプリケーション内で最も広いスコープを必要とする変数を割り当てるためによく使用されます。Node.js アプリケーションはデフォルトでは window オブジェクトにアクセスできません。
この問題を解決するために用いられる2つの方法はどれですか？
2つの回答を選択してください
A. ウィンドウオブジェクトの代わりにドキュメントオブジェクトを使用します。
B. 変数をグローバルオブジェクトに割り当てます。
C. 変数を module.exports に割り当て、必要に応じてそれらを require します。
D. ルートファイルに新しいウィンドウオブジェクトを作成します。
Answer: (解答を表示する)

最新問題**: 84**
以下のコードを参照してください。
クラス Vehicle {
コンストラクター(プレート) {
This.plate = プレート;
}
}
クラス Truck は Vehicle を拡張します {
コンストラクター(プレート、重量) {
//コードが不足しています
This.weight = weight;
}
displayWeight() {
console.log('トラック ${this.plate} の重量は ${this.weight} ポンドです。');}} Let myTruck = new Truck('123AB', 5000); myTruck.displayWeight(); コードの 09 行目にどのステートメントを追加すれば、「トラック 123AB の重量は 5000lb です。」と表示できますか？
A. Vehicle.plate = プレート;
B. This.plate = plate;

C. スーパープレート
D. スーパープレート＝プレート
Answer: C (メーッセージを残す**)**

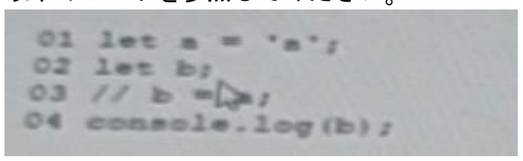
最新問題**: 85**
以下のコードブロックを参照してください。
クラス Animal{
コンストラクター(名前){
this.name = name;
}
makeSound(){
console.log(`${this.name} が音を発しています。`)
}
}
class Dog extends Animal{
コンストラクター(名前){
スーパー(名前)
this.name = name;
}
makeSound(){
console.log(`${this.name} が吠えています。`)
}
}
let myDog = new Dog('Puppy');
myDog.makeSound();
コンソール出力は何ですか？
Answer:
子犬が吠えている

最新問題**: 86**
以下のコードスニペットを参照してください。
配列を [1, 2, 3, 4, 4, 5, 4, 4] とする。
(i = 0 とし、i < array.length とし、i を増減する)
if (array[i] === 4) {
array.splice(i, 1);
}
}
コード実行後の配列の値は何ですか？
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: (解答を表示する**)**

最新問題**: 87**
どの2つのコンソールログがNaNを出力しますか？
2つの回答を選択してください | |
A. console.log(10 / Number('5)) ;
B. console.log(parseInt ' ("two')) ;
C. console.log(10 / 'five');
D. console.log(10 / 0);
Answer: (解答を表示する**)**

最新問題**: 88**
以下のコードを参照してください。



コード実行時に何が表示されますか？
A. A
B. 未定義
C. なし
D. ReferenceError: b が定義されていません
Answer: (解答を表示する**)**

最新問題**: 89**
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() {
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   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 }
```

D.

```
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 }
```

Answer: C (メーッセージを残す)

最新問題**: 90**

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

What is the output of the code execution?

A. Hello Dan

B. Hello John Doe

C. SyntaxError: Unexpected token in JSON

D. Hello Dan Doe

Answer: (解答を表示する**)**

最新問題: 91

ある開発者が、数値の配列を受け取り、配列内の最初の3つの数値の合計を返すsum3関数をテストするために、以下のコードを作成しました。そして、このテストは合格しました。

別の開発者がsum3の動作を変更し、配列内の最初の2つの数値のみを合計するようにしました。

更新されたsum3関数でこのテストを実行すると、どのような2つの結果が得られますか？

2つの回答を選択してください

A. 5行目のアサーションが失敗しました。

B. 2行目のアサーションが失敗しました。

C. 5行目のアサーションは合格です。

D. 2行目のアサーションは合格です。

Answer: A,D (メーッセージを残す**)**

最新問題: **92**
Refer to the followingcode:

```
<html lang="en">
 <body>
   <span onclick="console.log('Span message');">
     <button id="myButton">Send Message</button>
   </span>
 </body>
 <script>
   function displayMessage(ev){
    ev.stopPropagation();
    console.log('Button message');
   }
   const elem = document.getElementById("myButton");
   elem.addEventListener('click',displayMessage);
 </script>
</html>
```

A. Button message
B. Inner message
Outer message

C. Outer message
Inner message
D. Outer message
Answer: (解答を表示する**)**

最新問題**: 93**
以下のコードが与えられた場合：

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

2行目の出力は何ですか？
A. "未定義" 0
B. "null"
C. "x-
D. 'オブジェクト'
Answer: D (メーッセージを残す**)**

最新問題**: 94**
開発者が関数内のエラーを処理しようとしています。
エラーを他の箇所に伝播させることなく処理するための正しいアプローチを示しているコードセグメントはどれですか？
A)

```
try {
  doSomething();
} catch (error) {
  return error;
}
```

B)

```
try {
  doSomething();
} catch (error) {
  return null;
}
```

C)

```
try {
  doSomething();
} catch (error) {
  throw new Error('Error found');
}
```

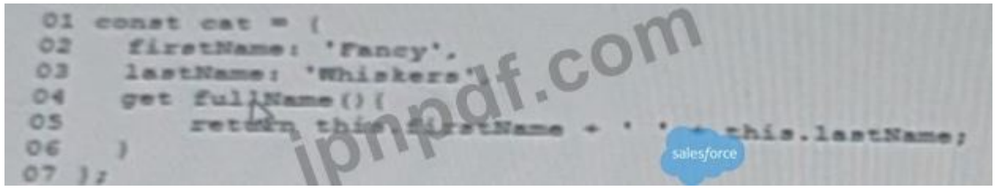
D)
A. オプションB
B. オプションC
C. オプションD
D. オプションA
Answer: (解答を表示する**)**

最新問題**: 95**
以下のHTMLを前提とします。

どのステートメントが、Universal Containers行にpriority-account CSSクラスを追加しますか？
A. document.queryselector('#row-uc').ClassList.add('priority-account');
B. document.querySelector('#row-uc').classes-push('priority-account');
C. document.getElementByid('row-uc').addClass('priority-account*');
D. document.querySelectorAll('#row-uc') -classList.add("priority-accour');
Answer: A (メーッセージを残す**)**

最新問題**: 96**
ある開発者が、既存のクライアントからのAPIリクエストに対応する新しいWebサーバーの構築をチーム内で主導している。
チームはNode.js上で動作するWebサーバーを求めており、新しいWebフレームワークであるMinimalist.jsを使いたいと考えている。一方、リード開発者は、既にコミュニティが存在する、より実績のあるバックエンドフレームワークを推奨したいと考えている。
リード開発者はどの2つのフレームワークを推奨するだろうか？
2つの回答を選択してください
A. それで
B. エクスプレス
C. ギャツビー
D. Angular
Answer: B,D (メーッセージを残す**)**

最新問題**: 97**
以下のオブジェクトを参照してください。



開発者は猫のfullnameプロパティにどのようにアクセスできますか？

A. 猫のフルネーム ( )

B. Cat.function.fullName

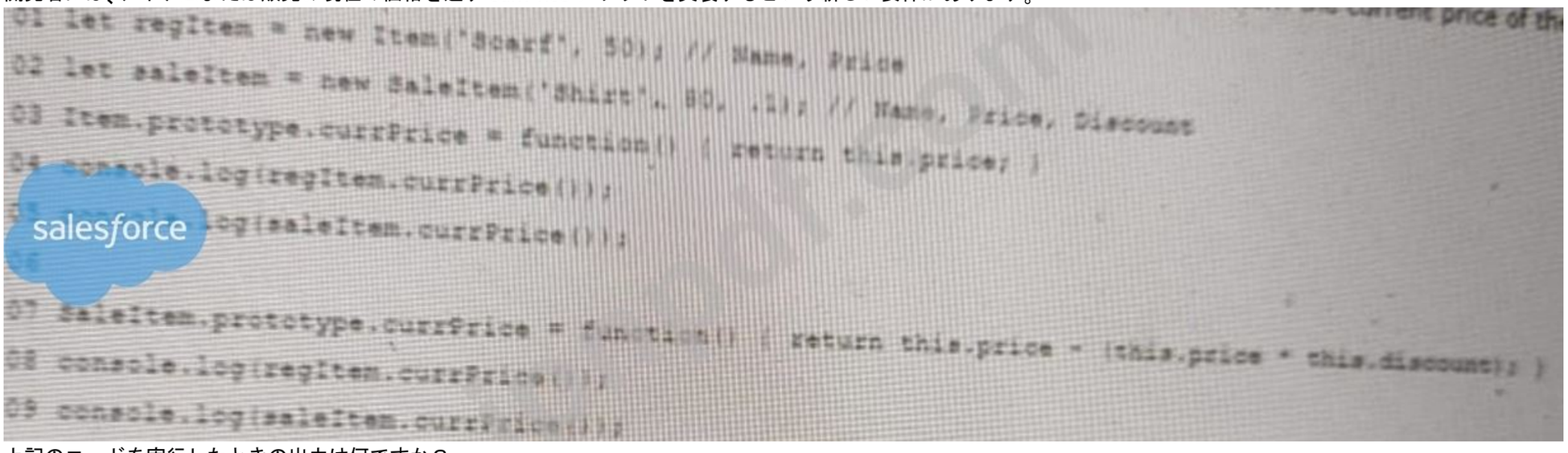
C. 猫のフルネーム

D. 猫、ゲット、フルネーム

Answer: D (メーッセージを残す**)**

最新問題**: 98**

開発者には、アイテムまたは販売の現在の価格を返す currPrice メソッドを実装するという新しい要件があります。



上記のコードを実行したときの出力は何ですか？

A. 50

80

72

B. 50

キャッチされない TypeError: saleItem,description は関数ではありません

50

80

C. 50

80

キャッチされない参照エラー: this,discount は未定義です

72

D. 50

80
50
72
Answer: D (メーッセージを残す)

最新問題**: 99**
以下のコードを参照してください。
関数 test (val) {
もし (val === undefined) {
return '未定義の値です！' ;
}
if (val === null) {
'null値です！' を返します。
}
値を返す。
}
x とする。
テスト(x)
13行目の関数呼び出しによって返される値は何ですか？
A. 13行目でエラーが発生します。
B. 値がゼロです！」
C. 未定義
D. 未定義の値！」
Answer: C (メーッセージを残す)

最新問題**: 100**
A developer at Universal Containers is creating their new landing pagebased 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 isloaded and there is no need to wait for the resources to be available.
Which statementshould be used to call personalizeWebsiteContent based on the above business requirement?
A. windows,addEventListener('load', personalizeWebsiteContent);
B. windows,addEventListener('onDOMCContentLoaded', personalizeWebsiteContent);
C. windows,addEventListener('onload', personalizeWebsiteContent);
D. windows,addEventListener('DOMContent Loaded ', personalizeWebsiteContent);
Answer: (解答を表示する)

最新問題**: 101**
開発者は、テクニカルリードから以下のコードについてコメントを受け取ります。
エラー：
const monthName = '7月';

const year = 2019;
if(year === 2019) {
monthName = '6月';
}
このコードを実行するには、どの行を編集すればよいですか？
A. 03 if (year == 2019) {
B. 02 定数年 = 2020;
C. 01 let monthName ='July';
D. 02 let year =2019;
Answer: (解答を表示する**)**

最新問題**: 102**
Refer to the expression below:
Let x = ('1' + 2) == (6 + 2) ;
How should this expression be modified to ensure that a evaluated to false?
A. Let x = (1' + '2') == ('6' / 2) ;
B. Let x = (1' + '2') == (6 + 2) ;
C. Let x = (1' + '2') === (6 + 2) ;
D. Let x = (1' + '2') === (6 / 2) ;
Answer: C (メーッセージを残す**)**

最新問題**: 103**
Refer to following code block:
Let array = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11,];
Let output =0;
For (let num of array){
if (output >0){
Break;
}
if(num % 2 == 0){
Continue;
}
Output +=num;
What is the value of output after the code executes?
A. 11
B. 25
C. 16
D. 36
Answer: (解答を表示する**)**

最新問題**: 104**

Universal Containers (UC) は、ユーザーが検索できるアプリケーションが
アカウントはキーが押されるたびにネットワークリクエストを送信します。その結果、
サーバーが処理すべきリクエスト。
● この問題を解決するため、UCは文字列変更時にデバウンス機能を実装することにした。
ハンドラ。
このデバウンス機能を実装するための3つの重要なステップは何ですか？
3つの回答を選択してください：
A. ネットワークリクエストのプロパティ「debounce」がtrueに設定されていることを確認してください。
B. 検索文字列が変更された場合、setTimeout の範囲内でリクエストをキューに追加します。
C. 既存のsetTimeoutがあり、検索文字列が変更された場合、既存のsetTimeoutをキャンセルします。
永続化された timerId を使用して setTimeout を実行し、それを新しい setTimeout に置き換えます。
D. 検索文字列変更ハンドルによって最後にキューに入れられたsetTimeoutのtimeIdを保存します。
E. setTimeoutが既に存在し、検索文字列が変更された場合、既存の
setTimeout を完了し、新しい setTimeout をキューに追加しないでください。
Answer: (解答を表示する**)**

最新問題**: 105**
A developer has the following array of hourly wages:
Let arr = (8, 5, 9, 75, 11, 25, 7, 75, , 13, 25);
For workers making less than $10 an hour rate should be multiple by 1.25 and returned in a new array.
How should the developer implement the request?
A. let arrl = arr-map((num) => { return ran * 1.25 }).filter((val) -> { return val < 10));
B. let arrl = arr .rr.acArray ((val) => (val < 10)) ,map((num) => { num * 1.25));
C. let arrl = arr.filterBy((val) => val < 10).aapBy<(num) -> num = ..25);
D. let arrl = arr.filter((val) => val < 10).map((num) -> num = 1.25);
Answer: A (メーッセージを残す**)**

最新問題**: 106**
ある開発者が、既存のクライアントからのAPIリクエストに対応する新しいWebサーバーの構築をチーム内で主導している。
チームはNode.js上で動作するWebサーバーを求めており、新しいWebフレームワークであるMinimalist.jsを使用したいと考えています。
リード開発者は、既にコミュニティが存在する、より実績のあるバックエンドフレームワークを推奨したいと考えている。
リード開発者はどの2つのフレームワークを推奨するだろうか？
2つの回答を選択してください
A. Angular
B. それで
C. エクスプレス
D. ギャツビー
Answer: A,D (メーッセージを残す**)**

最新問題**: 107**
Given a value, which three options can a developer use to detect if the value is NaN?
A. value === Number.NaN
B. value == NaN
C. Object.is(value, NaN)
D. value !== value
E. Number.isNaN(value)
Answer: (解答を表示する**)**
In JavaScript, NaN has some special properties:
* NaN is not equal to anything, including itself.
* Equality operators (==, ===) do not work for detecting NaN.
Check each option:
* A. value === Number.NaN
* Number.NaN is just NaN.
* But NaN === NaN is always false.
* So this expression is never true and cannot be used to detect NaN.
* B. value == NaN
* Same issue. NaN == NaN is always false.
* So this also never detects NaN.
* C. Object.is(value, NaN)
* Object.is is a more precise equality comparison method.
* It treats NaN as equal to NaN: Object.is(NaN, NaN) is true.
* So Object.is(value, NaN) is a valid way to detect NaN.
* D. value !== value
* Because NaN is the only JavaScript value that is not equal to itself , value !== value is true only when value is NaN.
* This is a classic trick for NaN detection.
* E. Number.isNaN(value)
* Number.isNaN is the recommended modern way to check if a value is NaN without coercion .
* It returns true only if value is actually the number NaN.
Therefore, the three correct ways are:
* Object.is(value, NaN)
* value !== value
* Number.isNaN(value)

最新問題**: 108**
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: (解答を表示する**)**

* 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:

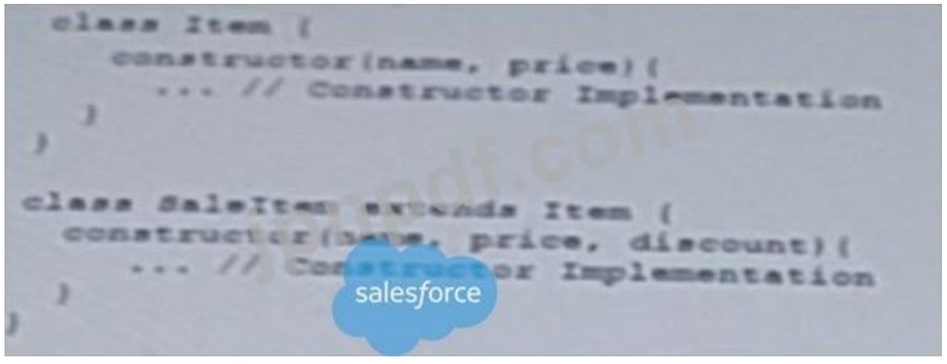
```
let myDog = new Dog( ' Puppy ' );
myDog.makeSound();
```

* new Dog(' Puppy '):
* Calls Dog constructor with name = ' Puppy ' .
* Inside, super(name) sets this.name = ' Puppy ' in the Animal constructor.

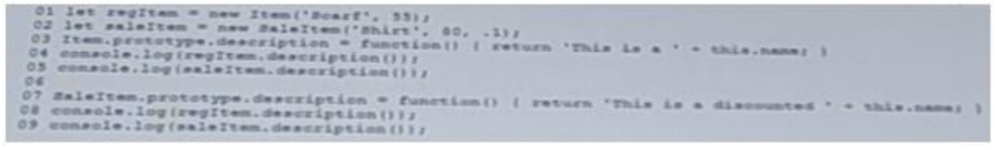
* Then this.name = name; in the Dog constructor keeps it as ' Puppy ' .
* myDog.makeSound():
* Calls the overridden makeSound() in Dog.
* Logs: " Puppy is barking. "
No reference errors, no undefined; the output is:
Puppy is barking.
Study Guide Concepts:
* ES6 classes and inheritance (extends)
* super() in subclass constructors
* Method overriding in subclasses
* Template literals: `${this.name} ...`

最新問題: **109**
開発者は、複数の関数を含むErrorHandlerモジュールを持っています。
複数の機能を利用できるようにするためには、どのようなエクスポート方式を活用すべきでしょうか？
A. マルチ
B. 名付けられた
C. デフォルト
D. ボール
Answer: B (メーッセージを残す**)**

最新問題: **110**
オンラインストアで購入可能な商品を表すクラスと、割引価格で販売されている商品を表すクラスが作成されました。コンストラクタは、渡された最初の値を名前に設定します。擬似コードは以下のとおりです。



開発者には、商品と販売商品の簡単な説明を返す説明メソッドを実装するという新たな要件があります。



上記のコードを実行したときの出力は何ですか？
A. これはスカーフです

これはシャツです
これはスカーフです
こちらは割引価格のシャツです
B. これは足場です
キャッチされない TypeError: saleItem、description は関数ではありません
これはシャツです
こちらは割引価格のシャツです
C. これはスカーフです
捕捉されなかった TypeError: saleitem、description は関数ではありません
これはスカーフです
こちらは割引価格のシャツです
D. これはスカーフです
これはシャツです
これは割引されたSカードです
こちらは割引価格のシャツです
Answer: A (メーッセージを残す**)**

最新問題**: 111**
開発者がチェックアウトボタンからHTMLクラス属性を削除したため、現在は単純に次のようになっています。
<ボタン>チェックアウト</ボタン>
チェックアウトボタンの存在を確認するテストがありますが、クラスが「blue」のボタンを探します。
該当するボタンが見つからないため、テストは失敗します。
このテストは、どのタイプのテストに分類されますか？
A. 真の陰性
B. 真陽性
C. 偽陰性
D. 偽陽性
Answer: (解答を表示する**)**
テストの文脈における定義（「ポジティブ」を「テストがバグ／失敗を報告する」と定義する）：
* 真陽性 :システムにバグがあり、テストが正しく失敗する。
* 誤検出 :システムは正しいが、テストが失敗する（実際にはバグではないバグを報告する）。
* 真の陰性 :システムにバグがあるが、テストは正しく合格し、検出のコンテキストでは「成功しなかった」ことを示す（ここではあまり一般的ではない表現）。
* 偽陰性 :システムにバグがあるにもかかわらず、テストが誤って合格してしまう（バグを見逃す）。
ここ：
* 記載されているとおり、本当の要件はチェックアウトボタンの存在を確認することです。
* ボタンはまだ存在しているので (< button > Checkout < /button >)、その要件に関してシステムの動作は正しいです。
* ただし、このテストは、実際には明示された要件の一部ではない、過度に具体的な条件 (class= " blue ") をチェックしています。
* テストは失敗し、問題がある（クラスが "blue " のボタンがない）と表示されますが、要件の観点からはチェックアウトボタンは存在します。
それで：
* チェックアウトボタンの表示に関する重大なバグはありません。

* テストは失敗を報告しました # 偽陽性。
したがって、正しい分類はD、偽陽性です。

最新問題: **112**
以下のコードを参照してください。

```
01 const event = new CustomEvent(
02   //Missing code
03 );
04 obj.dispatchEvent(event);
```

開発者は、recordId に関する情報を送信するために、update というカスタムイベントをディスパッチする必要があります。
開発者がこれを実現するために、2行目のプレースホルダーに挿入できるオプションはどれですか？ 2つ選択してください。
A. '更新', {
詳細 ; {
レコードID、『123abc」
)
)
B. 'update', (recordId ; 123abc'
)
C. 'update' , '123abc'
D. {type ; update ', recordId : '123abc')
Answer: A,B (メーッセージを残す**)**

最新問題: **113**
開発者は、複数の関数を含むモジュールを持っている。
複数の機能を利用できるようにするためには、どのようなエクスポート方式を活用すべきでしょうか？
A. マルチ
B. デフォルト
電話
C. 名付けられた
Answer: C (メーッセージを残す**)**
正解はDです。
モジュールに複数の関数が含まれており、各関数を個別に利用できるようにする必要がある場合は、名前付きエクスポートが最適な選択肢です。
例：
function add(a, b) {
a + b を返す。
}
関数 subtract(a, b) {
a - b を返す。
}
export { add, subtract };
そうすれば、別のファイルが必要な関数を正確にインポートできるようになります。

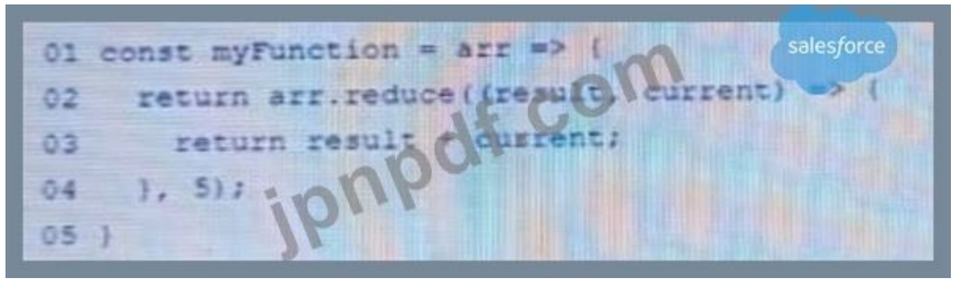
```
import { add, subtract } from ' ./mathUtils.js ' ;
console.log(add(5, 2));
console.log(subtract(5, 2));
```

名前付きエクスポートは、モジュールが複数の再利用可能な関数、定数、またはクラスを提供する場合に便利です。
他の選択肢が間違っている理由：
multi は JavaScript のエクスポート型ではありません。
デフォルトは通常、モジュールがエクスポートする主要な値が1つしかない場合に使用されます。モジュールにはデフォルトエクスポートを1つしか設定できません。
JavaScriptモジュール構文において、allはエクスポート型ではありません。
したがって、複数の関数の正しいエクスポートタイプは export という名前なので、検証済みの答えは D です。

最新問題**: 114**
以下のコードを参照してください。
textvalue = '1984' とします。
この文字列を整数に変換する正しい方法を示しているコードセグメントはどれですか？
A. numberValue = (Number) textvalue;
B. let numberValue = Integer (textValue);
C. numberValue = Number(textvalue);
D. let numberValue = textValue.Tointeger();
Answer: C (メーッセージを残す**)**

最新問題**: 115**
async/await キーワードの動作を正確に説明しているのは、次のうちどれですか？
A. 関連クラスには非同期関数が含まれています。
B. 関連関数は非同期メソッドを介してのみ呼び出すことができます
C. 関連付けられたものは、時々プロミスを返します。
D. 関連関数は常にプロミスを返します
Answer: C (メーッセージを残す**)**

最新問題**: 116**
以下のコードを参照してください。



空の配列を引数としてこの関数を呼び出した場合、どのような出力が得られますか？
A. 0を返す
B. リターン5

C. NaNを返す
D. 無限大への帰還
Answer: B (メーッセージを残す**)**

最新問題**: 117**
開発者は、エンドポイントの1つで実行時エラーが繰り返し発生するため、Node.jsウェブサーバーのデバッグを行う必要がある。
開発者は、ローカルマシン上でエンドポイントをテストし、ローカルサーバーに対してリクエストを送信して動作を確認したいと考えています。ソースコードでは、server.jsファイルがサーバーを起動します。
開発者は、ターミナルのみを使用してNode.jsサーバーのデバッグを行いたいと考えています。
開発者が現在のターミナルウィンドウでCLIデバッガーを開くには、どのコマンドを使用すればよいですか？
（タイプミスを修正済み：node_inspect # ノード検査、node_start_inspect # ノード開始検査）
A. node start inspect server.js
B. node inspect server.js
C. node server.js --inspect
D. node -i server.js
Answer: B (メーッセージを残す**)**
Node.jsには、組み込みのコマンドラインインターフェース（CLI）デバッガーが含まれています。ターミナルで直接使用するには、標準コマンドは次のとおりです。
node inspect server.js
これ：
* Node.js インスペクターで server.js を起動します。
* コマンドを実行したターミナルウィンドウに、CLI デバッガーを直接開きます。
* インタラクティブに操作できます:
* コードをステップ実行する
ブレークポイントを設定する
* 変数を検査する
* 実行を継続するなど
Bが正解である理由：
オプションB（node inspect server.jsに修正）は、現在のターミナルでCLIデバッガーを使用してスクリプトを実行するための標準的なNode.jsコマンドです。
* 外部のGUIツールには一切依存しません。
デバッグ中は完全にターミナル内で作業します。
他の選択肢が間違っている理由：
* A. node start inspect server.js
* 標準の Node.js CLI には、このような start サブコマンドはありません。
* これは有効な Node.js デバッグコマンドではありません。
* C. node server.js --inspect
* これは、インスペクタープロトコルを有効にした状態でNode.jsを起動します。通常は、Chrome DevToolsやVS Codeなどの外部ツールを接続するために使用されます。
* 現在のターミナルでCLIデバッガーを開くわけではありません。代わりに、他のツールが接続できるデバッグポートを開きます。
* 質問には具体的に「ターミナルのみを使用する」および「現在のターミナルウィンドウでCLIデバッガーを開く」と記載されており、これは--inspectではなくnode inspectを指しています。
* D. node -i server.js
* -i は、スクリプトの実行後にオプションで、Node を対話型 REPL モードで起動します。
* これは対話型シェル用であり、Nodeデバッガー用ではありません。

* CLIデバッガーのようなブレークポイント、ステップ実行、次へ、継続といったデバッグ機能は提供していません。
したがって、現在のターミナルでNode.js CLIデバッガーを開く唯一のオプションは次のとおりです。
The answer: B
JavaScript / Node.js の知識 / 学習ガイドの参考資料（概念名のみ、リンクは含まない）：
* Node.js CLI デバッガー: node inspect <script>
* Node.js インスペクター プロトコル: node --inspect
* CLIデバッガーとDevToolsベースのデバッグの違い
* Node.jsのコマンドラインオプションとサブコマンド

最新問題: **118**
以下のコードを参照してください。

```
01 async function functionUnderTest(isOK) {
02   if (isOK) return 'OK';
03   throw new Error('not OK');
04 }
```

上記のコードを正確にテストしているのは、どの主張でしょうか？
A. Console, assert (await functionalUnderTest (true) , 'not ok')
B. コンソール、assert (await functionalUnderTest (true) 、'ok
C. コンソール、assert (functionalUnderTest (true) 、'ok')
D. Console, assert (await functionalUnderTest (true) , 'not ok))
Answer: (解答を表示する)

最新問題: **119**
ブラウザでは、window オブジェクトはアプリケーションでブロードキャスト スコープを必要とする変数を割り当てるためによく使用されます。Node アプリケーションはデフォルトでは window オブジェクトにアクセスできません。
この問題を解決するために用いられる方法はどれですか？ 2つ選択してください）
A. ルートファイルに新しいウィンドウオブジェクトを作成します。
B. ウィンドウオブジェクトの代わりにドキュメントを使用します。
C. モジュールに変数を割り当て、エクスポートし、必要に応じてそれらを要求します。
D. 変数をグローバルオブジェクトに割り当てます。
Answer: A,B (メーッセージを残す)

最新問題: **120**
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. 既存の setTimeout があり、検索文字列が変更された場合、永続化された timerId を使用して既存の setTimeout をキャンセルし、新しい setTimeout に置き換えます。
B. Store the timeId of the setTimeout last enqueued by the search string change handle.
C. If there is an existing setTimeout and the search string change, allow the existing setTimeout to finish, and do not enqueue a new setTimeout.
D. 検索文字列が変更された場合、setTimeout の範囲内でリクエストをキューに追加します。

E. ネットワークリクエストのプロパティ「debounce」がtrueに設定されていることを確認してください。
Answer: (解答を表示する**)**

最新問題**: 121**
以下のコードを参照してください。

```
01 const myFunction = arr => {
02   return arr.reduce((result, current) => {
03     return result + current;
04   }, 10);
05 }
```

この関数を空の配列で呼び出した場合、どのような出力が得られますか？
A. エラーが発生します
B. NaNを返す
C. 0を返す
D. リターン10
Answer: D (メーッセージを残す**)**

最新問題**: 122**
テストはデータベースに依存しています。クエリ。テスト中、依存関係はメソッドを持つデータベースと呼ばれるオブジェクトに置き換えられます。
計算機クエリは配列を返します。開発者は、メソッドが何回呼び出されたかを確認する必要はありません。
要件を説明するテスト手法はどれですか？ 2つ選択）
2つの回答を選択してください
A. 白い箱
B. ブラックボックス
C. 置換
D. スタビング
Answer: (解答を表示する**)**

最新問題**: 123**
Universal Containersは最近、クラウドファンディングを開催するための新しいランディングページを公開しました。
キャンペーン。このページは外部ライブラリを使用してサードパーティの広告を表示します。ページが
完全にロードされると、DOM 内にランダムに配置された 50 個以上の新しい HTML アイテムが作成されます。
以下のコードのうちの1つ：

```
<!-- This is an ad -->
<div class="ad-library-item ad-hidden" onload="myFunction()">
  <img src="/ad-library/ad01.gif" />
</div>
```

すべての要素には同じ ad-library-item クラスが含まれており、デフォルトでは非表示になっていますが、ユーザーがページを移動する際にランダムに表示されます。

A. ブラウザのコンソールを使用して、ロードイベントの発生を防ぐスクリプトを実行します。
B. ブラウザを使用して、クラス ad-library-item を含むすべての要素を削除するスクリプトを実行します。
C. DOM インスペクターを使用して、クラス ad-library-item を含むすべての要素を削除します。
D. DOMインスペクターを使用して、ロードイベントの発生を防止します。
Answer: (解答を表示する**)**