**let** menus = ['Pizza Salami', 'Pizza Margharita'];

**let** prices = [10.99, 8.99];

**let** amounts = [2, 3];

**function** **getValueFromInput**(input) {

**return** document.getElementById(input).value;

}

**function** **getMenuFromInput**() {

// Finde das Input Feld mit der ID menu im Dokument

**let** menuElement = document.getElementById("menu");

// Überprüfe, ob das Input Feld existiert

**if** (menuElement) {

// Entferne alle leeren Zeichen am Anfang und am Ende des Wertes

**let** menuValue = menuElement.value.trim();

// Überprüfe, ob der Wert gültig ist

**if** (menuValue) {

// Gib den Wert zurück

**return** menuValue;

} **else** {

// Gib eine Fehlermeldung zurück, wenn der Wert leer ist

**return** "Bitte wähle ein Menü aus.";

}

} **else** {

// Gib eine Fehlermeldung zurück, wenn das Input Feld nicht gefunden wurde

**return** "Input Feld mit der ID menu wurde nicht gefunden.";

}

}

// Eine Funktion, die den Wert des Input Felds mit der ID price validiert und zurückgibt

**function** **getPriceFromInput**() {

// Finde das Input Feld mit der ID price im Dokument

**let** priceElement = document.getElementById("price");

// Überprüfe, ob das Input Feld existiert

**if** (priceElement) {

// Konvertiere den Wert des Input Felds in eine Zahl

**let** priceValue = Number(priceElement.value);

// Überprüfe, ob der Wert eine gültige Zahl ist

**if** (!isNaN(priceValue)) {

// Gib den Wert zurück

**return** priceValue;

} **else** {

// Gib eine Fehlermeldung zurück, wenn der Wert keine Zahl ist

**return** "Bitte gib einen gültigen Preis ein.";

}

} **else** {

// Gib eine Fehlermeldung zurück, wenn das Input Feld nicht gefunden wurde

**return** "Input Feld mit der ID price wurde nicht gefunden.";

}

}

**function** **onAddMenu**() {

// Die Werte der Input Felder erhalten

**let** menu = getMenuFromInput();

**let** price = getPriceFromInput();

// Die Werte überprüfen, ob sie gültig sind

**if** (**typeof** menu === "string" && **typeof** price === "number") {

// Die Werte in ein Objekt speichern

**let** menuObject = {

menu: menu,

price: price

};

// Das Objekt in ein Array einfügen

**let** menuArray = JSON.parse(localStorage.getItem("menuArray")) || [];

menuArray.push(menuObject);

// Das Array in den lokalen Speicher des Browsers speichern

localStorage.setItem("menuArray", JSON.stringify(menuArray));

// Die Werte am Ende der Arrays menus und prices hinzufügen

menus.push(menu);

prices.push(price);

// Eine 1 am Ende des Arrays amounts hinzufügen

amounts.push(1);

// Eine Bestätigungsnachricht anzeigen

alert("Menü hinzugefügt!");

} **else** {

// Fehlermeldungen anzeigen

alert(menus);

alert(prices);

}

}

**function** **getMenuIndex**(menu) {

// Durchlaufe das Array menus von Anfang bis Ende

**for** (**let** i = 0; i < menus.length; i++) {

// Überprüfe, ob das aktuelle Element gleich dem Parameter menu ist

**if** (menus[i] === menu) {

// Gib den Index i zurück

**return** i;

}

}

// Gib -1 zurück, wenn das Menü nicht gefunden wurde

**return** -1;

}

Nach der änderung der letzten Funktion des onAddMenu

**let** menus = ['Pizza Salami', 'Pizza Margharita'];

**let** prices = [10.99, 8.99];

**let** amounts = [2, 3];

**function** **getValueFromInput**(input) {

**return** document.getElementById(input).value;

}

**function** **getMenuFromInput**() {

// Finde das Input Feld mit der ID menu im Dokument

**let** menuElement = document.getElementById("menu");

// Überprüfe, ob das Input Feld existiert

**if** (menuElement) {

// Entferne alle leeren Zeichen am Anfang und am Ende des Wertes

**let** menuValue = menuElement.value.trim();

// Überprüfe, ob der Wert gültig ist

**if** (menuValue) {

// Gib den Wert zurück

**return** menuValue;

} **else** {

// Gib eine Fehlermeldung zurück, wenn der Wert leer ist

**return** "Bitte wähle ein Menü aus.";

}

} **else** {

// Gib eine Fehlermeldung zurück, wenn das Input Feld nicht gefunden wurde

**return** "Input Feld mit der ID menu wurde nicht gefunden.";

}

}

// Eine Funktion, die den Wert des Input Felds mit der ID price validiert und zurückgibt

**function** **getPriceFromInput**() {

// Finde das Input Feld mit der ID price im Dokument

**let** priceElement = document.getElementById("price");

// Überprüfe, ob das Input Feld existiert

**if** (priceElement) {

// Konvertiere den Wert des Input Felds in eine Zahl

**let** priceValue = Number(priceElement.value);

// Überprüfe, ob der Wert eine gültige Zahl ist

**if** (!isNaN(priceValue)) {

// Gib den Wert zurück

**return** priceValue;

} **else** {

// Gib eine Fehlermeldung zurück, wenn der Wert keine Zahl ist

**return** "Bitte gib einen gültigen Preis ein.";

}

} **else** {

// Gib eine Fehlermeldung zurück, wenn das Input Feld nicht gefunden wurde

**return** "Input Feld mit der ID price wurde nicht gefunden.";

}

}

**function** **onAddMenu**() {

// Die Werte der Input Felder erhalten

**let** menu = getMenuFromInput();

**let** price = getPriceFromInput();

// Die Werte überprüfen, ob sie gültig sind

**if** (**typeof** menu === "string" && **typeof** price === "number") {

// Die Werte in ein Objekt speichern

**let** menuObject = {

menu: menu,

price: price

};

// Das Objekt in ein Array einfügen

**let** menuArray = JSON.parse(localStorage.getItem("menuArray")) || [];

menuArray.push(menuObject);

// Das Array in den lokalen Speicher des Browsers speichern

localStorage.setItem("menuArray", JSON.stringify(menuArray));

// Den Index des Menüs in dem Array menus finden

**let** menuIndex = getMenuIndex(menu);

// Überprüfen, ob das Menü schon in dem Array menus existiert

**if** (menuIndex === -1) {

// Die Werte am Ende der Arrays menus und prices hinzufügen

menus.push(menu);

prices.push(price);

// Eine 1 am Ende des Arrays amounts hinzufügen

amounts.push(1);

} **else** {

// Den Wert in dem Array amounts an der Index-Stelle um 1 erhöhen

amounts[menuIndex]++;

}

// Eine Bestätigungsnachricht anzeigen

alert("Menü hinzugefügt!");

} **else** {

// Fehlermeldungen anzeigen

alert(menus);

alert(prices);

}

}

**function** **getMenuIndex**(menu) {

// Durchlaufe das Array menus von Anfang bis Ende

**for** (**let** i = 0; i < menus.length; i++) {

// Überprüfe, ob das aktuelle Element gleich dem Parameter menu ist

**if** (menus[i] === menu) {

// Gib den Index i zurück

**return** i;

}

}

// Gib -1 zurück, wenn das Menü nicht gefunden wurde

**return** -1;

}

Menü hinzufügen

**Warenkorb Menü hinzufügen**

In dieser Challenge lernst du den einfachsten Weg, um Waren zu einem Warenkorb hinzuzufügen. Hierfür erstellen wir 3 verschiedene Container für den Namen, den Preis und die Menge des der Ware. Diese Container werden nun mithilfe unseres Codes befüllt.



Erstelle ein Array namens menus.

Array menus wird die Essennamen von Menüs in unseren Warenkorb enthalten.

**Tipp:**

Bitte definiere Variablen immer ganz oben in deinem Script. Die Definitionsreihenfolge ist hier egal.



Erstelle ein Array namens prices.

Array prices wird die Prise von Menüs in unseren Warenkorb enthalten.

**Tipp:**

Bitte definiere Variablen immer ganz oben in deinem Script. Die Definitionsreihenfolge ist hier egal.



Erstelle ein Array namens amounts.

Array amounts wird die Menge von Menüs in unseren Warenkorb enthalten.

**Tipp:**

Bitte definiere Variablen immer ganz oben in deinem Script. Die Definitionsreihenfolge ist hier egal.



Füge den Arrays Werte hinzu.

* Füge dem Array menus Essennamen hinzu. Die Namen sollen vom Datentyp string sein und dürfen keine leeren Zeichen am Anfang und Schluss des Names enthalten. Z.B.:"Pizza Salami" und nicht " Pizza Salami ".
* Füge dem Array prices Nummern hinzu. Die Nummern sollen vom Datentyp number und dürfen Kommazahlen sein. Z.B.: 1.99 oder 2.0 und nicht "1.99".
* Füge dem Array amounts ganze Zahlen hinzu. Die Zahlen sollen vom Datentyp number sein (Keine Kommazahlen!). Z.B.: 1 oder 10 und nicht "1" oder 1.5.

**Tipp:**

Achte darauf, dass alle diese drei Arrays sollen gleiche Länge haben.



Erstelle eine Hilfsfunktion namens getValueFromInput, welche dir den Wert von einem **beliebigen** Input Feld zurückgibt.

Die Funktion getValueFromInput sollte einen Parameter haben, welcher der ID von einem Input Feld entspricht.



Erstelle eine Hilfsfunktion namens getMenuFromInput, welche den Wert von dem Input Feld mit der ID menu validiert und zurückgibt.

Dem Wert sollten **alle leeren Zeichen** am Anfang und am Ende entfernt werden.

**Tipp:**

- Nutze die Hilfsfunktion getValueFromInput, um die Werte von den Input-Feldern auszulesen.

- Versuche auch das Internet für die Lösung des Problems zu verwenden.

- Nutze Stichworte beim Suchen, um genau dein Problem zu beschreiben.

- Beispiel: "Javascript, String, Remove empty spaces from both sides"



Erstelle eine Hilfsfunktion namens getPriceFromInput, welche den Wert von dem Input Feld mit der ID price validiert und zurückgibt.

Der Wert sollte von Datentyp number sein.

**Tipp:**

- Nutze die Hilfsfunktion getValueFromInput, um die Werte des Input Felds herauszulesen.

- Versuche im Internet Lösungen zu finden für deine Probleme.

- Nutze Stichworte, um genau dein Problem zu beschreiben.

- Beispiel: "javascript string convert to number"



Erstelle eine Funktion, welche vom Button Menü hinzufügen per Klick aufgerufen wird.

**Tipp:**

Untersuche den Button mit den Developer Tools vom Browser, um den Funktionsnamen herauszufinden.



Erweitere die Funktion onAddMenu, sodass der Inhalt der Input-Felder am Ende der Arrays menus und prices hinzugefügt werden. Auch am Ende des Arrays amounts sollte eine 1 hinzugefügt werden.

**Tipp:**

Nutze die Hilfsfunktionen getMenuFromInput und getPriceFromInput, um die Werte von Input-Feldern herauszulesen.



Erstelle eine Hilfsfunktion namens getMenuIndex, welche einen Parameter namens menu hat. Das Ergebnis der Funktion ist der Index, wo sich der Parameter menu in dem Array menus befindet. Gebe -1 zurück, wenn das Ergebnis nicht gefunden werden kann.

**Tipp:**

- Versuche im Internet Lösungen zu finden für deine Probleme.

- Nutze Stichworte beim Suchen, um genau dein Problem zu beschreiben.

- Beispiel: "Javascript index of element in array"



Ändere die Funktion onAddMenu so, dass die Werte nur hizugefügt werden, wenn der Wert des Input Felds mit der ID menu sich noch nicht in dem Array menus befindet.

Andernfalls, wenn der Wert sich in Array menus befindet, dann finde die Index-Stelle und erhohe der Wert in Array amounst in dieser Stelle um 1.

**Tipp:**

Nutze dir das Hilfsfunktion getMenuIndex, um zu überprufen, ob das Menü sich bereits in dem Array menus befindet

function renderBasket(){ let basket = document.getElementById('basket'); basket.innerHTML=``; for (let i = 0; i < dishesBasket.length; i++) { const dish = dishesBasket[i]; //console.log(dish); const price = priceBasket[i]; //console.log(price); const amount = amountBasket[i]; //console.log(amount); basket.innerHTML +=` <div class="stylefood"> ${amount}x ${dish} ${price}€ <div class="symbole"> <button class= "hinzufügen" onclick='add\_amount'>+</button </div> <button class="löschen" onclick='remove\_amount'>-</button </div> </div>`; } } erklerre mir bitte diese Function

Kopieren

Suchen nach: function renderBasket erklärung

Antworten für Sie werden generiert…

Diese Funktion renderBasket ist ein Beispiel für eine Funktion in der Programmiersprache JavaScript. [Eine Funktion ist eine Beziehung zwischen zwei Mengen, die jedem Element aus der Definitionsmenge genau ein Element aus der Wertemenge zuordnet1](https://studyflix.de/mathematik/was-ist-eine-funktion-5167). Die Definitionsmenge ist die Menge der möglichen Eingaben für die Funktion, und die Wertemenge ist die Menge der möglichen Ausgaben.

Die Funktion renderBasket hat keine explizite Eingabe, aber sie verwendet einige globale Variablen, die außerhalb der Funktion definiert sind. Diese Variablen sind:

* basket: ein HTML-Element, das den Inhalt des Warenkorbs darstellt
* dishesBasket: ein Array, das die Namen der Gerichte enthält, die im Warenkorb sind
* priceBasket: ein Array, das die Preise der Gerichte enthält, die im Warenkorb sind
* amountBasket: ein Array, das die Anzahl der Gerichte enthält, die im Warenkorb sind

Die Funktion renderBasket hat auch keine explizite Ausgabe, aber sie verändert das HTML-Element basket, indem sie ihm einen neuen Inhalt zuweist. Dieser Inhalt besteht aus einer Reihe von div-Elementen, die jeweils ein Gericht, seinen Preis und seine Anzahl anzeigen. Außerdem haben diese div-Elementen zwei Schaltflächen, die es dem Benutzer ermöglichen, die Anzahl der Gerichte zu erhöhen oder zu verringern.

Die Funktion renderBasket verwendet eine for-Schleife, um über die Elemente der Arrays dishesBasket, priceBasket und amountBasket zu iterieren. Für jedes Element wird ein neues div-Element erstellt und an das HTML-Element basket angehängt. Die Funktion renderBasket wird typischerweise aufgerufen, wenn sich der Inhalt des Warenkorbs ändert, um das HTML-Element basket zu aktualisieren.

Calculator

Dieser Code ist eine Funktion in JavaScript, die den Gesamtpreis eines Warenkorbs berechnet und anzeigt. Hier ist eine Erklärung, was jeder Teil des Codes macht:

* function calculator() { ... } definiert eine Funktion mit dem Namen calculator, die keine Parameter hat und einen Codeblock ausführt, wenn sie aufgerufen wird.
* let sum = 0; deklariert eine Variable mit dem Namen sum und weist ihr den Wert 0 zu. Diese Variable wird verwendet, um die Summe der Preise aller Artikel im Warenkorb zu speichern.
* berechnen = document.getElementById('berechnen'); weist der Variablen berechnen das HTML-Element mit der ID berechnen zu. Dieses Element wird verwendet, um die Summe der Preise ohne Lieferkosten anzuzeigen.
* lieferK = document.getElementById('lieferK'); weist der Variablen lieferK das HTML-Element mit der ID lieferK zu. Dieses Element wird verwendet, um die Summe der Preise mit Lieferkosten anzuzeigen.
* totalSum = document.getElementById('totalSum') weist der Variablen totalSum das HTML-Element mit der ID totalSum zu. Dieses Element wird verwendet, um die Lieferkosten anzuzeigen.
* for (i = 0; i < priceBasket.length; i++) { ... } ist eine Schleife, die von 0 bis zur Länge des Arrays priceBasket läuft. Das Array priceBasket enthält die Preise der einzelnen Artikel im Warenkorb. Die Variable i wird als Index verwendet, um auf jedes Element des Arrays zuzugreifen.
* sum += priceBasket[i] \* amountBasket[i]; erhöht die Variable sum um das Produkt des Preises und der Anzahl des Artikels an der Position i im Warenkorb. Das Array amountBasket enthält die Anzahl der einzelnen Artikel im Warenkorb.
* berechnen.innerHTML = ${(sum).toFixed(2)} €; setzt den Inhalt des Elements berechnen auf die Variable sum mit zwei Nachkommastellen und einem Euro-Symbol.
* if (sum < 99) { ... } else { ... } ist eine bedingte Anweisung, die überprüft, ob die Variable sum kleiner als 99 ist. Wenn ja, wird der Codeblock im if-Zweig ausgeführt, sonst der Codeblock im else-Zweig.
* lieferK.innerHTML = ${(sum + 1.50).toFixed(2)}€; setzt den Inhalt des Elements lieferK auf die Variable sum plus 1.50 mit zwei Nachkommastellen und einem Euro-Symbol. Dies entspricht den Gesamtkosten mit Lieferkosten, wenn die Summe kleiner als 99 ist.
* totalSum.innerHTML = 1.50€; setzt den Inhalt des Elements totalSum auf 1.50 Euro. Dies entspricht den Lieferkosten, wenn die Summe kleiner als 99 ist.
* lieferK.innerHTML = ${(sum).toFixed(2)}; setzt den Inhalt des Elements lieferK auf die Variable sum mit zwei Nachkommastellen. Dies entspricht den Gesamtkosten ohne Lieferkosten, wenn die Summe größer oder gleich 99 ist.
* totalSum.innerHTML = 0.00€; setzt den Inhalt des Elements totalSum auf 0.00 Euro. Dies entspricht den Lieferkosten, wenn die Summe größer oder gleich 99 ist.

Ich hoffe, das hat dir geholfen, den Code zu verstehen. Wenn du mehr über JavaScript lernen möchtest, kannst du dir diese [Einführung](https://www.geeksforgeeks.org/introduction-to-javascript/) oder dieses [Tutorial](https://www.w3schools.com/Js/) ansehen. Du kannst auch einen [kostenlosen KI-Code-Erklärer](https://zzzcode.ai/code-explain) verwenden, um Code in jeder Programmiersprache online zu erklären.

Lieferando Erklärung

Hallo, das ist Bing. Ich kann versuchen, dir den Code zu erklären. 😊

[Der Code ist in JavaScript geschrieben, einer Programmiersprache, die verwendet wird, um Webseiten interaktiv zu gestalten1](https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript). Der Code erstellt ein Menü mit zwei Gerichten (Cheeseburger und Salami Pizza), die jeweils einen Namen, eine Beschreibung und einen Preis haben. Der Code ermöglicht es dem Benutzer, die Gerichte zum Warenkorb hinzuzufügen oder zu entfernen, und berechnet die Gesamtsumme der Bestellung.

Hier ist eine kurze Erklärung der wichtigsten Funktionen des Codes:

* render() ist eine Funktion, die das Menü auf der Webseite anzeigt. Sie verwendet eine Schleife, um über das Array menue zu iterieren, das die Informationen über die Gerichte enthält. Für jedes Gericht wird ein HTML-Element mit dem Namen, der Beschreibung, dem Preis und einem Button zum Hinzufügen zum Warenkorb erstellt. Die Funktion fügt diese Elemente dem Element mit der ID menu hinzu, das im HTML-Dokument definiert sein muss.
* addOrder(i) ist eine Funktion, die aufgerufen wird, wenn der Benutzer auf den Button zum Hinzufügen zum Warenkorb klickt. Sie nimmt einen Parameter i, der den Index des gewählten Gerichts im Array menue angibt. Die Funktion speichert den Namen und den Preis des Gerichts in den Arrays dishesBasket und priceBasket, die den Inhalt des Warenkorbs darstellen. Außerdem speichert sie die Anzahl der bestellten Gerichte in dem Array amountBasket. Wenn das Gericht bereits im Warenkorb vorhanden ist, erhöht die Funktion die Anzahl um eins. Andernfalls fügt sie das Gericht und die Anzahl zum Warenkorb hinzu. Die Funktion ruft dann die Funktion renderBasket() auf, um den Warenkorb zu aktualisieren.
* trashRender() ist eine Funktion, die den Inhalt des Warenkorbs in einem Element mit der ID tresh anzeigt. Sie verwendet eine Schleife, um über die Arrays dishesBasket und priceBasket zu iterieren, die den Namen und den Preis der Gerichte im Warenkorb enthalten. Für jedes Gericht wird ein HTML-Element mit dem Namen, dem Preis und einem Button zum Löschen erstellt. Die Funktion fügt diese Elemente dem Element mit der ID tresh hinzu, das im HTML-Dokument definiert sein muss.
* renderBasket() ist eine Funktion, die den Inhalt des Warenkorbs in einem Element mit der ID basket anzeigt. Sie verwendet eine Schleife, um über die Arrays dishesBasket, priceBasket und amountBasket zu iterieren, die den Namen, den Preis und die Anzahl der Gerichte im Warenkorb enthalten. Für jedes Gericht wird ein HTML-Element mit der Anzahl, dem Namen, dem Preis und zwei Buttons zum Hinzufügen oder Abziehen erstellt. Die Funktion fügt diese Elemente dem Element mit der ID basket hinzu, das im HTML-Dokument definiert sein muss. Die Funktion ruft dann die Funktion calculator() auf, um die Gesamtsumme der Bestellung zu berechnen.
* calculator() ist eine Funktion, die die Gesamtsumme der Bestellung berechnet. Sie verwendet eine Schleife, um über das Array priceBasket zu iterieren, das den Preis der Gerichte im Warenkorb enthält. Sie multipliziert den Preis mit der Anzahl der Gerichte, die im Array amountBasket gespeichert ist, und addiert die Ergebnisse zu einer Variablen sum. Die Funktion zeigt die Summe in einem Element mit der ID berechnen an, das im HTML-Dokument definiert sein muss. Die Funktion überprüft auch, ob die Summe kleiner als 20 ist, und fügt gegebenenfalls eine Liefergebühr von 1,50 € hinzu. Die Funktion zeigt die Liefergebühr in einem Element mit der ID lieferK und die Gesamtsumme in einem Element mit der ID totalSum an, die im HTML-Dokument definiert sein müssen.
* add\_amount(i) ist eine Funktion, die aufgerufen wird, wenn der Benutzer auf den Button zum Hinzufügen klickt. Sie nimmt einen Parameter i, der den Index des gewählten Gerichts im Array dishesBasket angibt. Die Funktion erhöht die Anzahl des Gerichts im Array amountBasket um eins und ruft dann die Funktion renderBasket() auf, um den Warenkorb zu aktualisieren.
* remove\_amount(i) ist eine Funktion, die aufgerufen wird, wenn der Benutzer auf den Button zum Abziehen klickt. Sie nimmt einen Parameter i, der den Index des gewählten Gerichts im Array dishesBasket angibt. Die Funktion überprüft, ob der Warenkorb leer ist oder ob die Anzahl des Gerichts gleich oder kleiner als eins ist. Wenn ja, zeigt sie eine Warnmeldung an oder entfernt das Gericht aus den Arrays dishesBasket, priceBasket und amountBasket. Wenn nicht, verringert sie die Anzahl des Gerichts im Array amountBasket um eins und ruft dann die Funktion renderBasket() auf, um den Warenkorb zu aktualisieren.

Ich hoffe, das hilft dir, den Code zu verstehen. Wenn du mehr über JavaScript lernen möchtest, kannst du diese Tutorials besuchen: [JavaScript Tutorial - W3Schools](https://www.w3schools.com/Js/) oder [What is JavaScript? - Learn web development | MDN](https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript" \t "_blank). 😊