

HTML

と

JavaScript

を学ぶ

“ブロック”プログラミング教材

Javloc

ジャブロック

解き方BOOK

目次

本教材のねらい	2
Javloc (ジャブロック) 使用方法	3
Javloc (ジャブロック) 機能一覧	4
HTML を学ぶ + ブロック操作練習	5
教材サイトから問題文・アプリを開く	6
1. 条件分岐 if 初級 解答手順	15
教材サイトからアプリを開く	16
ページを構成 (HTML 部分の作成)	17
プログラムを組み立てる (JavaScript 部分の作成)	22
2. 条件分岐 if 中級 解答手順	29
教材サイトからアプリを開く	30
ページを構成 (HTML 部分の作成)	30
プログラムを組み立てる (JavaScript 部分の作成)	39
3. 条件分岐 if 上級 解答手順	47
教材サイトからアプリを開く	47
ページを構成 (HTML 部分の作成)	48
プログラムを組み立てる (JavaScript 部分の作成)	48
<巻末資料>作成上の基本ルール	56

本教材のねらい

本教材は、コンピュータで情報が処理される仕組みを体感して学ぶことができる教材です。

「アルゴリズム」や「プログラミング」の学習には「実践」が不可欠です。

「体感」・「実践」・「経験」を繰り返すことで、コンピュータで情報が処理される仕組みを理解します。

<各級段階イメージ>

	アルゴリズム	ブロック	問題文表現
初級	処理手順どおりに ブロックを組み立てること で経験できる	使うブロックしか 準備されていない コーディングルールを 意識しなくてもOK	専門的な言葉なし 動きを文章で表現
中級	処理手順をヒントに ブロックを組み立てること で仕組みを理解	エラーを含むブロックあり 基本的な コーディングルール が身に付く	変数や宣言といった プログラミングで使う キーワードを使用
上級	処理手順の記載なし 自ら考え組み立てる力が身 につく	エラーを含むブロックあり 基本的な コーディングルール が身に付く	変数や宣言といった プログラミングで使う キーワードを使用 中級より端的に表現される

Javloc(ジャブロック)使用方法

利用申請後、当協会より送信されてくる「Javloc」のサイトへアクセスします。

4つの題材（条件分岐2種類・繰り返し2種類）が3つの級（初級・中級・上級）に分けられており、合計12問の問題が準備されています。初級から順に解答を進めていくと理解が深まります。

Javloc ジャブロック

HTML + JavaScript で
プログラミングの基本を学ぶ

まずはブロック操作
の練習として
HTML 練習問題を
ご活用ください。

これも①と同様に問題 PDF
を表示または印刷してから
ブロック練習を開始してく
ださい。

HTMLブロック練習 HTML練習問題PDF

初級プロジェクト

条件分岐「if」初級	条件分岐「switch」初級	繰り返し「for」初級	繰り返し「while」初級
問題文PDF	問題文PDF	問題文PDF	問題文PDF

中級プロジェクト

条件分岐「if」中級	条件分岐「switch」中級	繰り返し「for」中級	繰り返し「while」中級
問題文PDF	問題文PDF	問題文PDF	問題文PDF

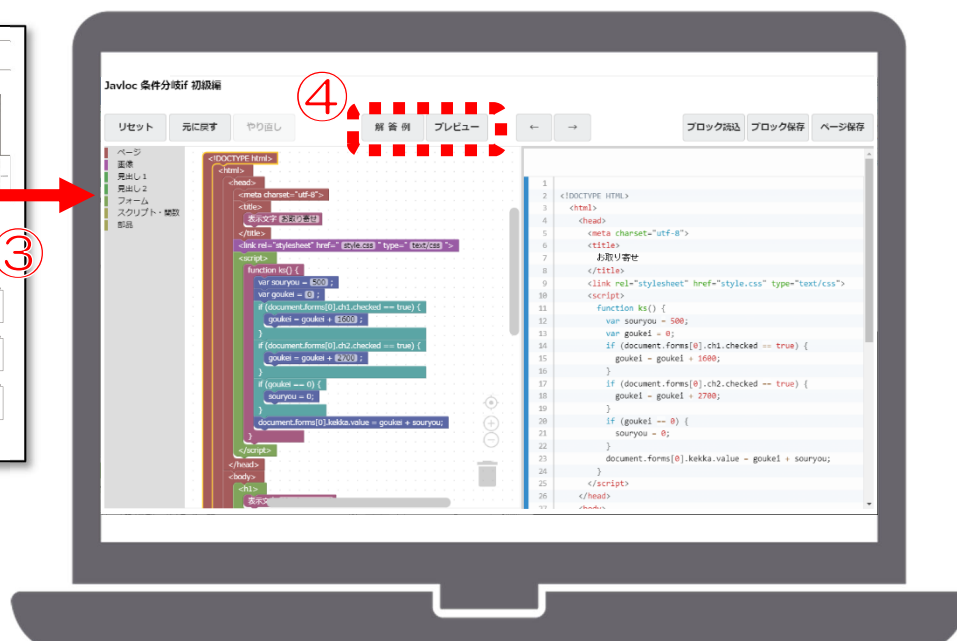
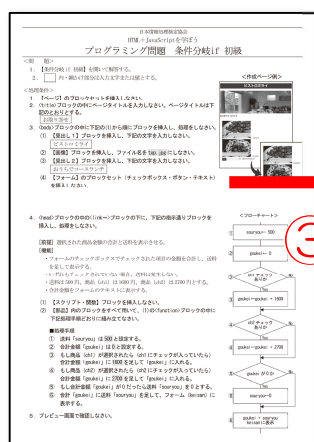
上級プロジェクト

条件分岐「if」上級	条件分岐「switch」上級	繰り返し「for」上級	繰り返し「while」上級
問題文PDF	問題文PDF	問題文PDF	問題文PDF

- ① 解答する問題の「問題文 PDF」をクリックし表示または印刷をします。
- ② 解答する問題のボタン「+」をクリックします。
- ③ 問題文に従ってブロックを操作し、文字や値を入力します。
- ④ 右側に表示されるソースや「プレビュー」機能をクリックして動きを確認しながら解答します。「解答例」機能をクリックすると解答例の WEB ページを確認することができます。

問題文

操作画面



Javloc(ジャブロック)機能一覧

Javloc 条件分岐if 初級編

The screenshot displays the Javloc editor with two main panels. On the left, the 'Block Edit Area' shows a visual representation of HTML blocks and JavaScript code. A context menu is open, listing actions like '複製' (Copy), 'コメントを追加' (Add comment), 'ブロックを折りたたむ' (Fold block), 'ブロックを無効にする' (Disable block), '7個のブロックを削除' (Delete 7 blocks), and 'ヘルプ' (Help). On the right, the 'ソース表示エリア' (Source Display Area) shows the raw HTML and JavaScript code corresponding to the blocks. The code includes a function `ks()` that calculates a total based on form inputs.

機能名		機能
①	リセット	ブロック編集エリアに配置したブロックをすべて削除する
②	ブロックリスト	HTML・JavaScriptのブロックが準備されているエリア ここからブロックを挿入する（問題文に従って挿入・配置）
③	解答例	解答ページを表示する（完成しているページで動きを確認）
④	プレビュー	作成した結果を表示する
⑤	エリア範囲切り替え	ブロック編集エリアとソース表示エリアの幅を変更する
⑥	ブロック読込	ブロックファイル（xml形式）を読み込む ※⑦の機能で保存したブロックを読み込んで再開することができます。 ※違う問題・級のブロックを読み込むとエラーになります。 ※ファイル名が保存時のものでない場合はエラーになります。
⑦	ブロック保存	作成中のブロックの状態をそのまま保存（xml形式） ※作業を中断した際や結果を残しておきたいときに活用してください。 ※保存時のファイル名は変更しないでください。変更するとブロック読み込みができなくなります。複数保存するとファイル名の後ろに「(1)」などの番号が付きますが、それはそのままでも問題ありません。
⑧	ページ保存	ブロックで生成されたページソースを保存（html形式）
⑨	位置合わせ	ブロックのある位置に合わせる
⑩	拡大・縮小	ブロック編集エリア画面の拡大・縮小
⑪	ゴミ箱	ブロックを捨てる ※ブロックを選択して「Delete」キーまたはドラッグ&ドロップでゴミ箱に入ります。 ※ゴミ箱をクリックすると捨てたブロックが表示され、それをクリックすると、またブロック編集エリアに戻すことができます。
⑫	ブロックを右クリック 編集メニュー	ブロックの上で右クリックすると編集メニューが表示されます。 ※「元に戻す」ボタンはありませんが、キーボードの「Ctl」＋「Z」キー押下でひとつ前の状態に戻すことができます。
⑬	元に戻す	操作を元に戻す
⑭	やり直し	元に戻した操作をやり直す

HTML を学ぶ + ブロック操作練習

普段よく目にする Web ページや Web アプリですが、これらを制作する際に必要となるのが **HTML** と **CSS** です。表示する文字などの内容・構成は **HTML** に、見た目(装飾)は **CSS** に書き込みます。これから「HTML ブロック練習」の問題文を見ながら、Javloc でブロックを組み立てて HTML を記述してみます。最後に、あらかじめ準備された CSS ファイルにリンクをして装飾がどのように行われるのか確認します。

HTML ブロック練習

【HTML ブロック練習】を開き、HTML をブロックで組み立て、WEB ページを完成させましょう。

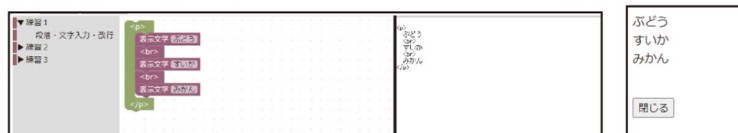
<練習 1>段落・改行を学び、HTML に文字が表示されることを体験する。

▼練習 1 をクリックし、下記の処理をこなさい。

1. 【段落・文字入力・改行】ブロックセットを選択し、編集エリアに配置する。
2. <p>ブロックの中にある表示文字、
タグを並べ替え、下記の順番に組み替える。

ぶどう
すいか
みかん

[処理結果参考 (編集エリア・ソースエリア・プレビュー画面)]

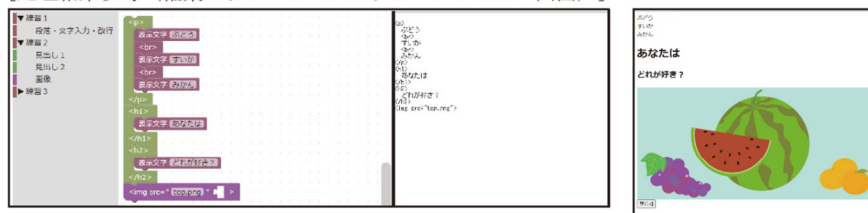


<練習 2>見出し・画像を学ぶ。

▼練習 2 をクリックし、下記の処理をこなさい。

1. <練習 1>で作成したブロックの下に【見出し 1】ブロックを選択して配置し、下記の文字を入力する。
あなたは
2. 1. のブロックの下に【見出し 2】ブロックを選択して配置し、下記の文字を入力する。
どれが好き？
3. 2. のブロックの下に【画像】ブロックを選択して配置する。
4. 「プレビュー」をクリックし、<作成例 1>のようになっているか確認する。

[処理結果参考 (編集エリア・ソースエリア・プレビュー画面)]



<練習 3>WEB ページの基本構造を知る。

▼練習 3 をクリックし、下記の処理をこなさい。

1. 【ページ】ブロックセットを選択し、編集エリアの空いている部分に配置する。
2. <練習 2>で作成したブロックすべてを 1. ブロックセット内の<body>タグの中へ配置する。
3. 「プレビュー」をクリックし、<作成例 1>のようになっているか確認する。

[処理結果参考 (編集エリア・ソースエリア・プレビュー画面)]



教材サイトから問題文・アプリを開く

まず、Javloc 教材サイトから[HTML 練習問題 PDF]を開き、印刷または画面で確認しましょう。

[HTML ブロック練習]をクリック



<練習 1>段落・改行を学び、HTML に文字が表示されることを体験する。

▼練習 1 をクリックし、下記の処理を下さい。

1. 【段落・文字入力・改行】ブロックセットを選択し、編集エリアに配置する。

リスト内[練習 1]をクリック。



リスト内[段落・文字入力・改行]をクリック。<p>ブロックセットをドラッグし、編集エリアにドロップ。ソース表示エリアで設定されていることを確認。



HTML では、コンピュータに文書の意味を理解してもらうために、**タグ**と呼ばれる印を組み合わせで使います。基本的には、開始タグ<>と終了タグ</>で一組になります。要素を開始タグと終了タグで挟み込んで表現します。

例えば、今回の<p>みかん~すいか</p>は「みかん~すいか」までは「**段落**」ですよ！という指示がコンピュータに伝わります。

中には、終了タグ</>がなく、開始タグ<>のみで記述される要素もあります。

2. <p>ブロックの中にある表示文字、
タグを並べ替え、下記の順番に組み替える。

ぶどう
すいか
みかん

<p>ブロックの中にある「表示文字 みかん」ブロックをドラッグし、編集エリアの広いところにドロップ。<p>ブロック内の全ブロックが外れる。

Javloc html練習編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

▼練習1 段落・文字入力・改行
▶練習2
▶練習3

<p>
</p>

表示文字 みかん

表示文字 ぶどう

表示文字 すいか

1
2 <p>
3
4 </p>
5
6 みかん
7

8 ぶどう
9

10 すいか

こんなイメージ？

※本教材「Javloc」では、ブロックをドラッグすると、その下に付いているブロックも一緒に付いてきます。

「表示文字 ぶどう」ブロックをドラッグし、上のブロックから離れた場所にドロップ。

Javloc html練習編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

▼練習1 段落・文字入力・改行
▶練習2
▶練習3

<p>
</p>

表示文字 みかん

表示文字 ぶどう

表示文字 すいか

1
2 <p>
3
4 </p>
5
6 みかん
7

8
9 ぶどう
10

11 すいか

「表示文字 みかん」ブロックの下にある
ブロックをドラッグし「表示文字 すいか」ブロックの下にドロップ

Javloc html練習編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

▼練習1 段落・文字入力・改行
▶練習2
▶練習3

<p>
</p>

表示文字 みかん

表示文字 ぶどう

表示文字 すいか

1
2 <p>
3
4 </p>
5
6 みかん
7

8
9

▼練習1 段落・文字入力・改行
▶練習2
▶練習3

<p>
</p>

表示文字 みかん

表示文字 ぶどう

表示文字 すいか

1
2 <p>
3
4 </p>
5
6 みかん
7
8 ぶどう
9

10 すいか

「表示文字 みかん」ブロックをドラッグし、一番下にドロップ。

「表示文字 ぶどう」ブロックをドラッグし、<p>ブロックの中にドロップ。

問題文通りぶどう・すいか・みかんの順番になっていることを確認。

[プレビュー]ボタンをクリックし、縦に順番通りに並んでいるか確認。

Try

は、「改行」を指示するタグです。試しに、下図のように
タグの上で右クリックをしてメニューから[ブロックを削除]を選択し、
タグを削除してみましょう。タッチスクリーンの場合は長押し

ソースを見ると、改行されているため表示もそのようになるかと思いきや、Web ページでは改行されずに並んで表示されてしまいます。

<練習 2> 見出し・画像を学ぶ。

▼練習 2 をクリックし、下記の処理を下さい。

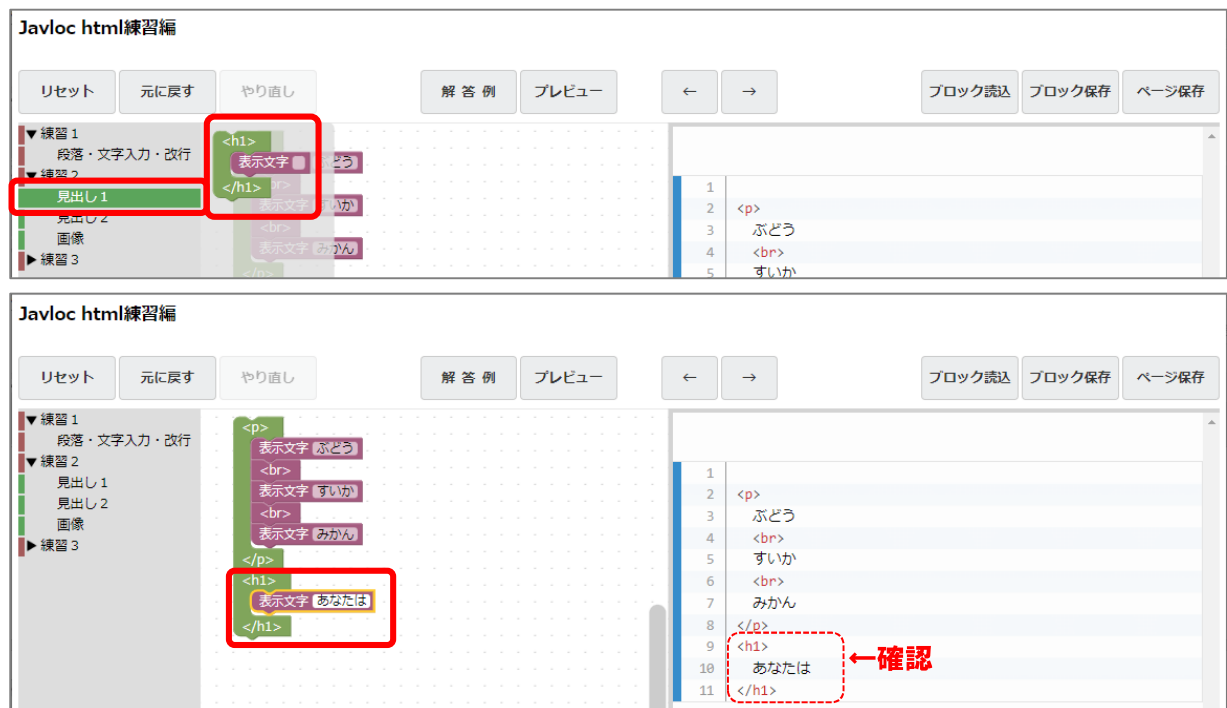
1. <練習 1> で作成したブロックの下に【見出し 1】ブロックを選択して配置し、下記の文字を入力する。

あなたは

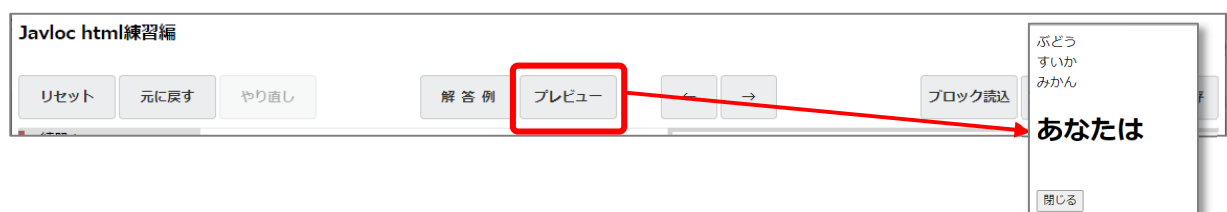
リスト内[練習 2]をクリック。



リスト内[見出し 1]をクリック。<h1>ブロックをドラッグし、<p>ブロックの下にドロップ。表示文字ブロックの空欄に「あなたは」と入力し、ソース表示エリアで見出し 1 が設定されていることを確認。



[プレビュー]ボタンをクリックし、「あなたは」の文字が太く大きく表示されていることを確認。



<h1>は、文章の中で目立たせたい「見出し」を指示するタグです。これは、その名のとおり見出しを示すためのタグで、<h1>、<h2>、<h3>～と合計で 6 種類あります。数字が小さいほど大きな見出しを表します。

見出し 1 <h1>～</h1> → 1 番目に大きな見出し
見出し 2 <h2>～</h2> → 2 番目に大きな見出し
見出し 3 <h3>～</h3> → 3 番目に大きな見出し
}

2. 1. のブロックの下に【見出し2】ブロックを選択して配置し、下記の文字を入力する。

どれが好き？

リスト内[見出し2]をクリック。<h2>ブロックをドラッグし、<h1>ブロックの下にドロップ。
表示文字ブロックの空欄に「どれが好き？」と入力し、ソース表示エリア・プレビューで見出し
2が設定されていることを確認。

Javloc html練習編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

▼練習1 段落・文字入力・改行
▼練習2 見出し1
見出し2
▼練習3 画像

<h2>
表示文字 どれが好き？
</h2>

1
2
3
4
5
6
7
8
9
10
11
12
13
14

```
<p>  
ぶどう  
<br>  
すいか  
<br>  
みかん  
</p>  
<h1>  
あなたは  
</h1>  
<h2>  
どれが好き？  
</h2>
```

ぶどう
すいか
みかん
あなたは
どれが好き？
閉じる

←確認→

3. 2. のブロックの下に【画像】ブロックを選択して配置する。

リスト内[画像]をクリック。ブロックをドラッグし、<h2>ブロックの下にドロップ。
ソース表示エリアで画像が設定されていることを確認。

Javloc html練習編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

▼練習1 段落・文字入力・改行
▼練習2 見出し1
見出し2
▼練習3 画像

1
2
3
4
5

```
<p>  
表示文字 ぶどう  
<br>  
表示文字 すいか  
<br>  
表示文字 みかん  
</p>  
<h1>  
あなたは  
</h1>  
<h2>  
表示文字 どれが好き？  
</h2>  

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

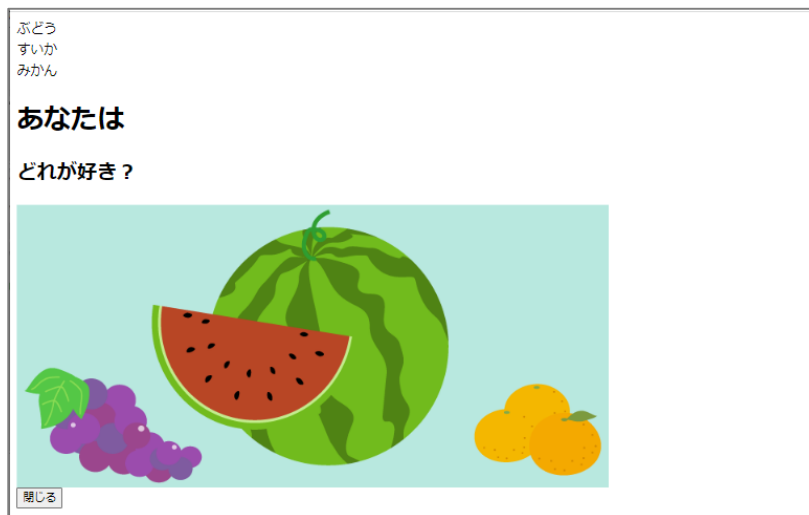
```
<p>  
ぶどう  
<br>  
すいか  
<br>  
みかん  
</p>  
<h1>  
あなたは  
</h1>  
<h2>  
どれが好き？  
</h2>  

```

←確認→

4. 「プレビュー」をクリックし、＜作成例1＞のようになっているか確認する。

プレビューで確認をすると、下図のように画像が表示される。



＜img＞は、「画像」の表示を指示するタグです。後ろにある「src = “□”」で、表示する画像ファイルを指定します。□部分に画像のファイル名を記述します。

＜練習3＞WEB ページの基本構造を知る。

▼練習3をクリックし、下記の処理をこなさい。

1. 【ページ】ブロックセットを選択し、編集エリアの空いている部分に配置する。

リスト内[練習3]をクリック。

リスト内[ページ]をクリック。枠内のブロックセットをドラッグし、編集エリアの開いている部分にドロップ。



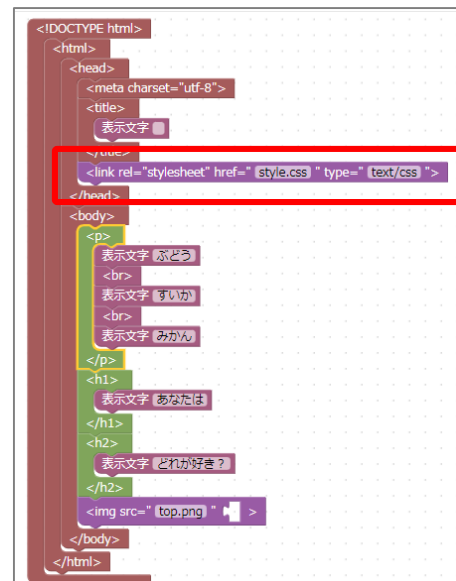
<p>ブロックをドラッグすると、その下のすべてのブロックが付いてくるので、そのまま今挿入したブロックセット内の<body>ブロックの中にドロップ。

The screenshot shows the Javloc HTML editor interface. On the left, a sidebar lists exercises: 練習 1 (段落・文字入力・改行), 練習 2 (見出し 1, 見出し 2, 画像), and 練習 3 (ページ). The main workspace displays a block-based HTML structure. A red arrow points from a block containing text and image elements (a paragraph block) towards the <body> tag in the central structure. The right pane shows the resulting HTML code, with lines 1-13 visible, including <p>,
, <h1>, <h2>, and tags.

ソース表示エリアを確認。

This screenshot shows the same Javloc HTML editor interface, but with a red dashed box highlighting the source code area on the right. A red arrow points to the box with the text "確認" (Check/Confirm). The source code area displays the full HTML document structure, including the <body> section which contains the paragraph and image blocks. The central workspace also shows the block-based structure, with the paragraph block highlighted by a red box.

プレビューで確認すると、文字や文字の後ろに色が付いていたり、文字が中央に配置されていたりします。これは、先ほど1. で挿入したブロックセットにある `<link rel="stylesheet" href="style.css" type="text/css">` で設定された「CSS」ファイルの設定が反映された結果。



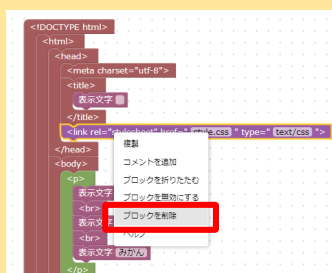
CSS では、主に装飾を設定します。今回は下記のような設定がされています。

本教材では、CSS のリンク設定のみ行い、ファイル内の設定は行いません。

<pre>h1{ font-size : 300%; text-align : center; color : #faebd7; background-color : #66cdad; }</pre>	見出し 1 に対しての装飾 ・ フォントサイズ：300% ・ 文字の水平方向の配置：中央揃え ・ 文字の色：#faebd7 ・ 文字の背景色：#66cdad
<pre>h2{ font-size : 200%; text-align : center; }</pre>	見出し 2 に対しての装飾 ・ フォントサイズ：200% ・ 文字の水平方向の配置：中央揃え
<pre>p { font-size : 150%; text-align : center; color : #66cdad; }</pre>	段落に対しての装飾 ・ フォントサイズ：150% ・ 文字の水平方向の配置：中央揃え ・ 文字の色：#66cdad

Try

試しに、下図のように `<link rel="stylesheet" href="style.css" type="text/css">` の上で右クリックをしてメニューから[ブロックを削除]を選択して CSS ファイルを指定する `<link>` タグを削除し、プレビューで確認してみましょう。



【プレビュー】



色はなく、配置も左揃えの状態の文字と画像が表示されます。これは、装飾の情報が何もない状態です。

プログラミング問題

解答手順

1. 条件分岐 if 初級 解答手順

<問題文>

日本情報処理検定協会
HTML+JavaScriptを学ぼう

プログラミング問題 条件分岐if 初級

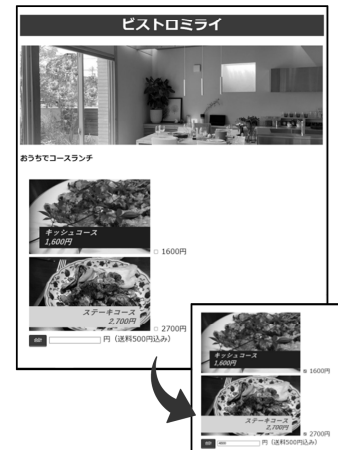
<問題>

1. 【条件分岐 if 初級】を開いて解答する。
2. 内・網かけ部分は入力文字または値とする。

<処理条件>

1. 【ページ】のブロックセットを挿入しなさい。
2. <title>ブロックの中にページタイトルを入力しなさい。ページタイトルは下記のとおりとする。
3. <body>ブロックの中に下記の(1)から順にブロックを挿入し、処理をしなさい。
 - (1) 【見出し1】ブロックを挿入し、下記の文字を入力しなさい。
 - (2) 【画像】ブロックを挿入し、ファイル名を top.jpg にしなさい。
 - (3) 【見出し2】ブロックを挿入し、下記の文字を入力しなさい。
 - (4) 【フォーム】のブロックセット (チェックボックス・ボタン・テキスト) を挿入しなさい。

<作成ページ例>



4. <head>ブロックの中の<link>ブロックの下に、下記の指示通りブロックを挿入し、処理をしなさい。

【前提】 選択された商品金額の合計と送料を表示させる。

【機能】

- ・ フォームのチェックボックスでチェックされた項目の金額を合計し、送料を足して表示する。
- ・ いずれもチェックされていない場合、送料は発生しない。
- ・ 送料は500円、商品 (ch1) は1600円、商品 (ch2) は2700円とする。
- ・ 合計金額をフォームのテキストに表示する。

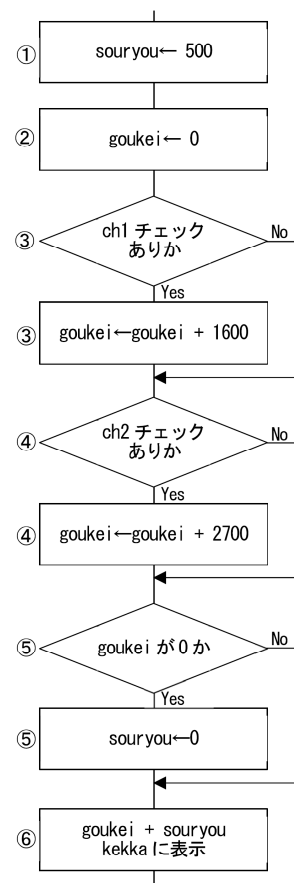
- (1) 【スクリプト・関数】ブロックを挿入しなさい。
- (2) 【部品】内のブロックをすべて用いて、(1)の<function>ブロックの中に下記処理手順どおりに組み立てなさい。

■処理手順

- ① 送料「souryou」は500と設定する。
- ② 合計金額「goukei」は0と設定する。
- ③ もし商品 (ch1) が選択されたら (ch1 にチェックが入っていたら) 合計金額「goukei」に1600を足して「goukei」に入れる。
- ④ もし商品 (ch2) が選択されたら (ch2 にチェックが入っていたら) 合計金額「goukei」に2700を足して「goukei」に入れる。
- ⑤ もし合計金額「goukei」が0だったら送料「souryou」を0とする。
- ⑥ 合計「goukei」に送料「souryou」を足して、テキスト (kekka) に表示する。

5. プレビュー画面で確認しなさい。

<フローチャート>



教材サイトからアプリを開く

Javloc 教材サイトから[初級プロジェクト]の[条件分岐「if」初級]をクリック。



<処理条件> 1～3. では、ページの HTML 部分を作成し、4. では JavaScript を作成します。



1～3

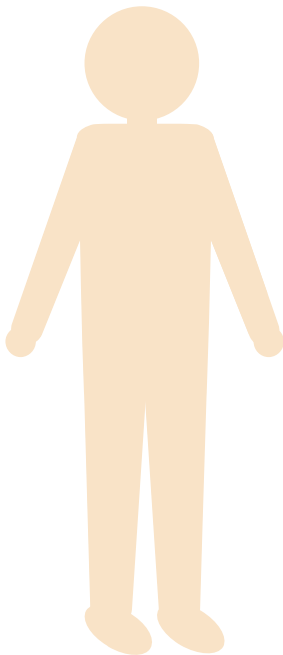
表示されている文字や画像、情報を入力する場所（チェックボックスやボタンなど）や結果を表示する場所などを HTML で設定します。

4

入力された情報を受け取り、結果を返すといったような表示上は見えない「動き」の仕組みを JavaScript で設定します。

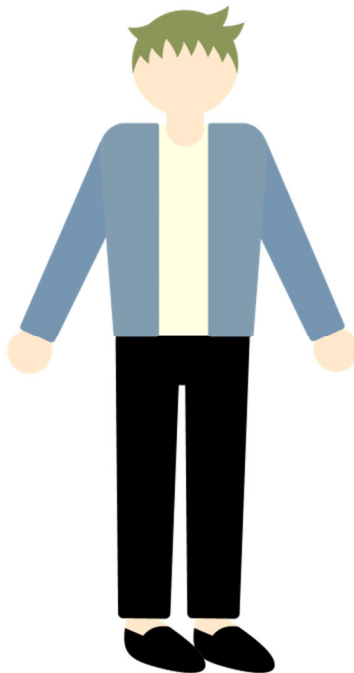
HTML と CSS と JavaScript の役割 イメージ図

HTML のみ



構造

HTML + CSS



デザイン

HTML + CSS
+ JavaScript



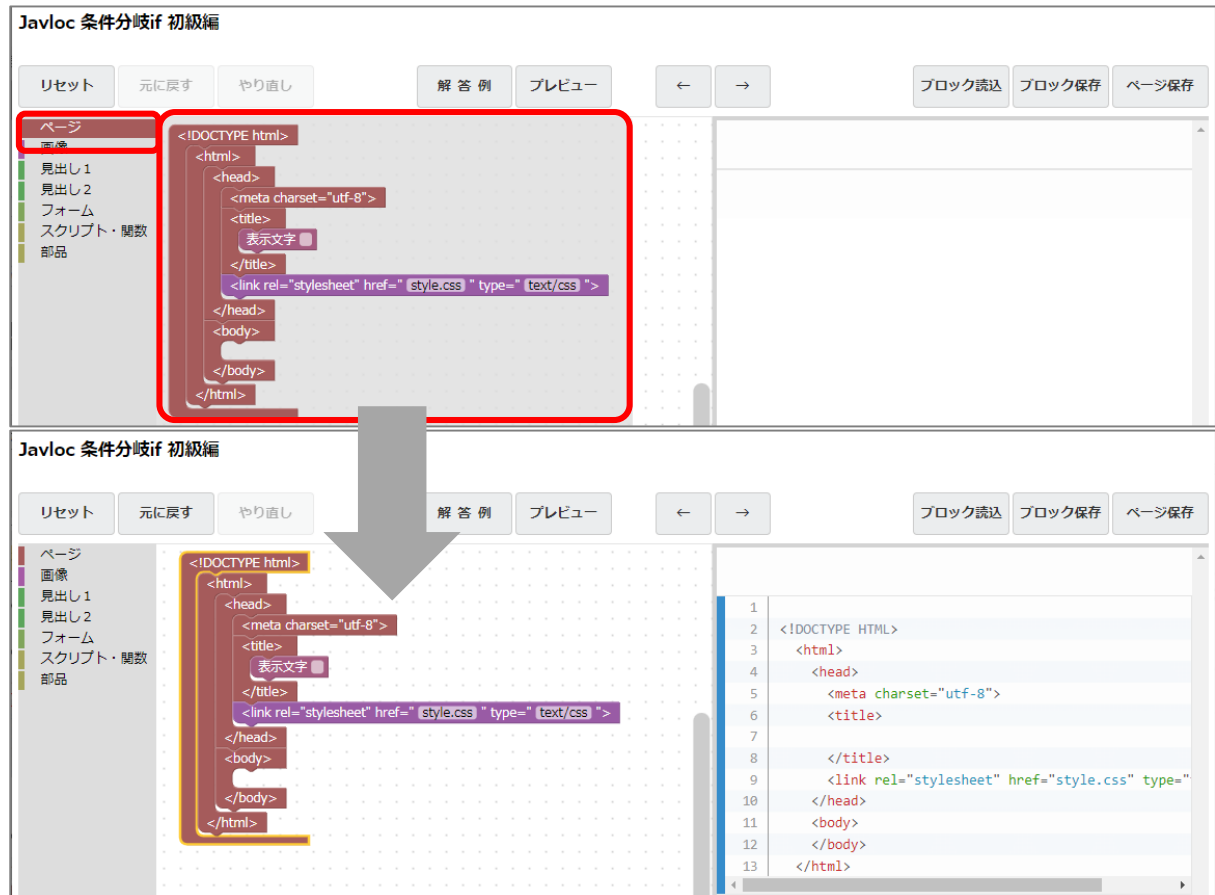
動き

ページを構成（HTML 部分の作成）

1. 【ページ】のブロックセットを挿入しなさい。

リスト内[ページ]をクリック。

枠内にあるブロックセットをドラッグし、編集エリアにドロップ。



2. <title>ブロックの中にページタイトルを入力しなさい。ページタイトルは下記のとおりとする。

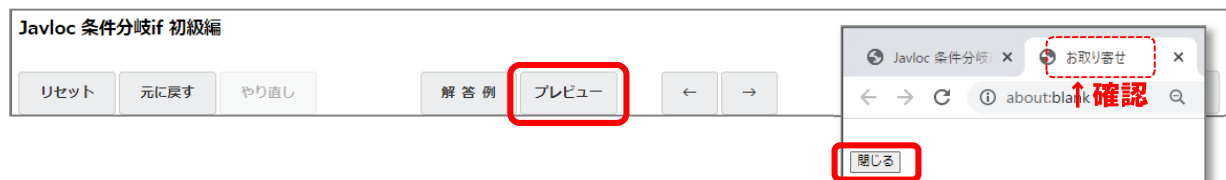
お取り寄せ

<title>ブロックの表示文字ブロックの空欄に「お取り寄せ」と入力。

入力後、ソース表示エリアでページタイトルが設定されていることを確認。



[プレビュー]ボタンをクリックして確認。[閉じる]ボタンをクリックしブロック操作画面へ戻る。



3. <body>ブロックの中に下記の(1)から順にブロックを挿入し、処理をしない。

(1) 【見出し1】ブロックを挿入し、下記の文字を入力しなさい。

ビストロミライ

リスト内[見出し1]をクリック、枠内にある<h1>ブロックをドラッグし、<body>ブロックの中にドロップ。表示文字ブロックの空欄に「ビストロミライ」と入力し、ソース表示エリアで見出し1が設定されていることを確認。



[プレビュー]ボタンをクリックし、正しく設定されているか確認。

[閉じる]ボタンをクリックし、ブロック操作画面へ戻る。



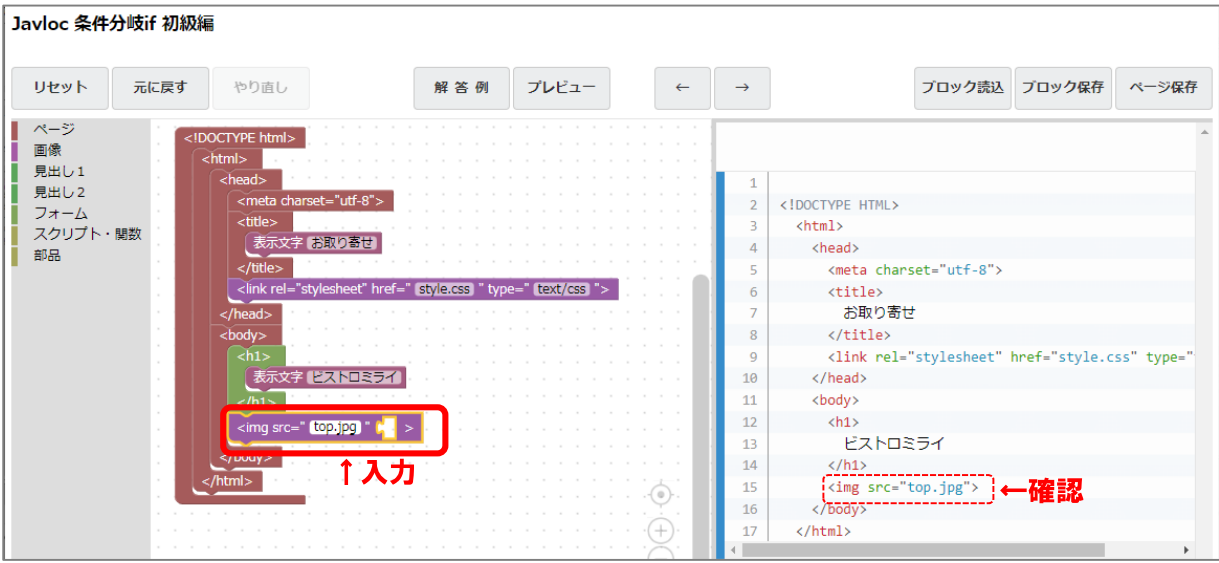
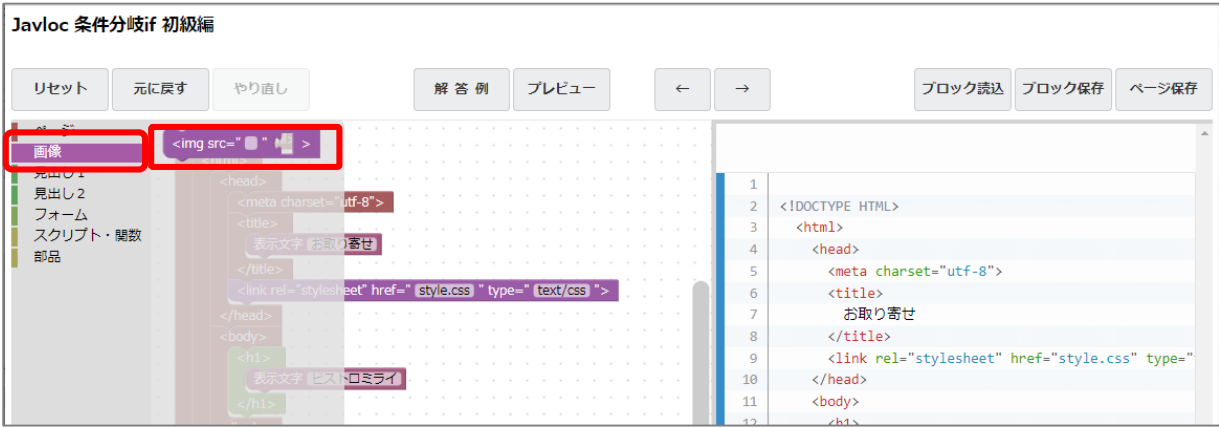
↑ <作成ページ例>を参考に
「見出し1」の設定が正
しく行われているか確認

見出し1の背景色が茶色で、文字色が白色になっているのは、CSSで既に設定しているためです。詳しくはP.13の説明をご確認ください。



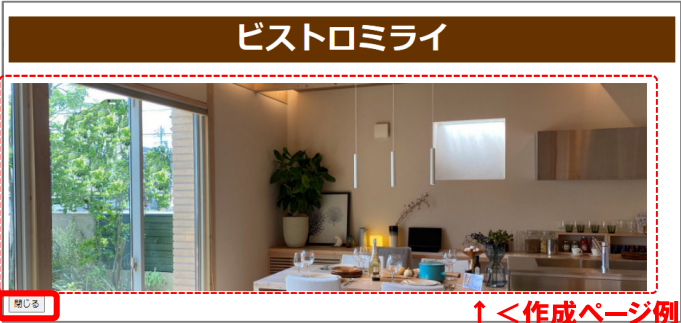
(2) 【画像】ブロックを挿入し、ファイル名を `top.jpg` にしなさい。

リスト内[画像]をクリック、枠内にあるブロックをドラッグし<h1>ブロックの下にドロップ。 “ ”の空欄の中に「top.jpg」と入力し、ソース表示エリアで画像が設定されていることを確認。



[プレビュー]ボタンをクリックし、正しく設定されているか確認。

[閉じる]ボタンをクリックし、ブロック操作画面へ戻る。



↑＜作成ページ例＞を参考に
「画像」の設定が正しく
行われているか確認



(3) 【見出し2】ブロックを挿入し、下記の文字を入力しなさい。

おうちでコースランチ

リスト内[見出し2]をクリック、枠内にある<h2>ブロックをドラッグしブロックの下にドロップ。

表示文字ブロックの空欄に「おうちでコースランチ」と入力し、ソース表示エリアで見出し2が設定されていることを確認。

Javloc 条件分岐if 初級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ページ 画像 見出し1 見出し2 フォーム スクリプト・関数 部品

```
<h2>
表示文字
</h2>

<h1>
表示文字 ビストロミライ
</h1>
<head>
<meta charset="utf-8">
<title>
表示文字 お取り寄せ
</title>
<link rel="stylesheet" href="style.css" type="text/css">
</head>
<body>
<h2>
表示文字 おうちでコースランチ
</h2>
</body>
</html>
```

1 <!DOCTYPE HTML>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <title>
6 </title>

Javloc 条件分岐if 初級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ページ 画像 見出し1 見出し2 フォーム スクリプト・関数 部品

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>
表示文字 お取り寄せ
</title>
<link rel="stylesheet" href="style.css" type="text/css">
</head>
<body>
<h1>
表示文字 ビストロミライ
</h1>

<h2>
表示文字 おうちでコースランチ
</h2>
</body>
</html>
```

1 <!DOCTYPE HTML>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <title>
6 </title>
7 お取り寄せ
8 </title>
9 <link rel="stylesheet" href="style.css" type="text/css">
10 </head>
11 <body>
12 <h1>
13 ビストロミライ
14 </h1>
15
16 <h2>
17 おうちでコースランチ
18 </h2>
19 </body>
20 </html>

[プレビュー]ボタンをクリックし、正しく設定されているか確認。

[閉じる]ボタンをクリックし、ブロック操作画面へ戻る。

Javloc 条件分岐if 初級編

リセット 元に戻す やり直し 解答例 **プレビュー** ← → ブロック読み込み ブロック保存 ページ保存



(4) 【フォーム】のブロックセット（チェックボックス・ボタン・テキスト）を挿入しなさい。

リスト内[フォーム]をクリック、枠内にある<form>ブロックセットをドラッグし<h2>ブロックの下にドロップ。ソース表示エリアでフォームが設定されていることを確認。

Javloc 条件分岐if 初級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ページ 画像 見出し1 見出し2 フォーム スクリプト・関数 部品

```
<form>
  
  <input type="checkbox" name="ch1" >
  表示文字 1600円
  <br>
  
  <input type="checkbox" name="ch2" >
  表示文字 2700円
  <br>
  <input type="button" value="合計" onclick="ks()" >
  <input type="text" name="kekka" >
  表示文字 円 (送料500円込み)
</form>
```

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8">
<title>
お取り寄せ
</title>
<link rel="stylesheet" href="style.css" type="text/css">
</head>
<body>
<h1>
ビストロミライ
</h1>

<h2>
おうちでコースランチ
</h2>
<form>
  
  <input type="checkbox" name="ch1" >
  1600円
  <br>
  
  <input type="checkbox" name="ch2" >
  2700円
  <br>
  <input type="button" value="合計" onclick="ks()" >
  <input type="text" name="kekka" >
  円 (送料500円込み)
</form>
</body>
</html>
```

↓ 確認

[プレビュー]ボタンをクリックし、<作成ページ例>を参考に「フォーム」の設定が正しく行われているか確認。[閉じる]ボタンをクリックし、ブロック操作画面へ戻る。

Javloc 条件分岐if 初級編

リセット 元に戻す やり直し 解答例 **プレビュー** ← → ブロック読み込み ブロック保存 ページ保存

まだJavaScriptが設定されていないため、「合計」ボタンをクリックしても、Aのテキスト欄には何も表示されません。この後設定するJavaScriptによって、「合計」ボタンをクリックすると計算が行われ、Aのテキスト欄に結果が表示されるようになります。

キッシュコース 1,600円

ステーキコース 2,700円

合計 A 円 (送料500円込み)

```
<form>
   画像
  <input type="checkbox" name="ch1" > チェックボックス ch1
  表示文字 1600円 文字
  <br> 改行
   画像
  <input type="checkbox" name="ch2" > チェックボックス ch2
  表示文字 2700円 文字
  <br> 改行
  <input type="button" value="合計" onclick="ks()" > ボタン
  <input type="text" name="kekka" > テキスト
  表示文字 円 (送料500円込み) 文字
</form>
```

4. <head>ブロックの中の<link>ブロックの下に、下記の指示通りブロックを挿入し、処理をなさい。

【前提】 選択された商品金額の合計と送料を表示させる。

【機能】

- ・フォームのチェックボックスでチェックされた項目の金額を合計し、送料を足して表示する。
- ・いずれもチェックされていない場合、送料は発生しない。
- ・送料は 500 円、商品（ch1）は 1600 円、商品（ch2）は 2700 円とする。
- ・合計金額をフォームのテキストに表示する。



まだこの時点では前提や機能の意味が分からなくても、解答例のページを確認し、「こんな動きのプログラムを作成するんだな」と想像するくらいで問題ありません。

まず、[前提]と[機能]を読み、「解答例」ボタンをクリックし、解答例ページの動きを確認します。

Javloc 条件分岐if 初級編

リセット 元に戻す やり直し **解答例** プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ビストロミライ

このフォームの部分を操作してみましょう。
チェックボックスにチェックを入れ「合計」ボタンを押してみます。
右図の4パターンを試してみましょう。

おうちでコースランチ

☐ キッシュコース 1,600円 □ 1600円

☐ ステーキコース 2,700円 □ 2700円

合計 円 (送料500円込み)

閉じる

1600 + 送料 500 = 2100

☐ キッシュコース 1,600円 □ 1600円

☒ ステーキコース 2,700円 □ 2700円

合計 2100 円 (送料500円込み)

2700 + 送料 500 = 3200

☐ キッシュコース 1,600円 □ 1600円

☒ ステーキコース 2,700円 □ 2700円

合計 3200 円 (送料500円込み)

チェックボックスで選択された商品の金額を足して、さらに送料 500 円を足した合計金額がテキスト内に表示される。

商品が何も選択されていないときは、送料は足されず、0 円となる。

1600 + 2700 + 送料 500 = 4800

☒ キッシュコース 1,600円 □ 1600円

☒ ステーキコース 2,700円 □ 2700円

合計 4800 円 (送料500円込み)

両方チェックがなければ
送料も発生しない→0

☐ キッシュコース 1,600円 □ 1600円

☐ ステーキコース 2,700円 □ 2700円

合計 0 円 (送料500円込み)

(1) 【スクリプト・関数】ブロックを挿入しなさい。

リスト内[スクリプト・関数]をクリック、枠内にある<script>ブロックセットをドラッグし、<link>ブロックの下にドロップ。ソース表示エリアでスクリプトが設定されていることを確認。

The first screenshot shows the Javloc editor interface. On the left sidebar, the 'Script' block is highlighted. In the main workspace, a red box highlights the script block being dragged. The second screenshot shows the script block inserted into the code editor. A red box highlights the function definition, and a red arrow points to it with the text '確認' (Check).

<script> ~ </script> とは、JavaScript などのスクリプトを文書内に埋め込んだり外部スクリプトを読み込んだりするために使用するものです。

function 〇 () とは、関数を定義する JavaScript です。〇部分に関数名を記述します。これは、さまざまな処理を一つにまとめて、名前を付けることができるものです。

(2) 【部品】内のブロックをすべて用いて、(1)の<function>ブロックの中に下記処理手順どおりに組み立てなさい。

■ 処理手順

- ① 送料「souryou」は 500 と設定する。

リスト内[部品]をクリック、枠内にある var souryou = ; ブロックを選択してドラッグ。

The screenshot shows the Javloc editor interface. On the left sidebar, the 'Parts' block is highlighted. In the main workspace, a red box highlights the 'Parts' block being dragged. The second screenshot shows the 'Parts' block inserted into the code editor. A red box highlights the variable declaration, and a red arrow points to it with the text '確認' (Check).

function ks () { ~ } ブロックの中にドロップ。

空欄に「500」と入力。

※必ず半角で入力しましょう。

Javloc 条件分岐if 初級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ページ 画像 見出し1 見出し2 フォーム スクリプト・関数 部品

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>
表示文字 お取り寄せ
</title>
<link rel="stylesheet" href="style.css" type="text/css">
<script>
function ks() {
var souryou = 500;
}
</script>
</head>
<body>
<h1>
表示文字 ビストロミライ
</h1>
</body>
</html>
```

↑ 入力

```
1 <!DOCTYPE HTML>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <title>
6 お取り寄せ
7 </title>
8 <link rel="stylesheet" href="style.css" type="text/css">
9 <script>
10 function ks() {
11 var souryou = 500; ← 確認
12 }
13 </script>
14 </body>
```

プログラムはすべて「半角」で入力します！そして、大文字と小文字も区別されます。

ここで行ったのは、「変数」に名前を付け、500 という値を入れるということです。

「変数」とは、値や文字列を保管しておく箱のようなものです。

「var」というのは、変数を新規に作りますという意味で、「=」は代入演算子と呼ばれ、右側に書かれる値や文字列をこの箱に入れるという意味になります。

ここでは、変数「souryou」を新規に作成し、そこに「500」という値を入れています。

② 合計金額「goukei」は0と設定する。

リスト内[部品]をクリック、枠内にある var goukei = ; ブロックを選択し、ドラッグして①で設定したブロックの下にドロップ。

Javloc 条件分岐if 初級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ページ 画像 見出し1 見出し2 フォーム スクリプト・関数 部品

```
var souryou = 500;
var goukei = 0;
<meta charset="utf-8">
if (document.forms[0].ch1_checked == true) {
goukei = goukei + 500;
}
<link rel="stylesheet" href="style.css" type="text/css">
if (document.forms[0].ch2_checked == true) {
goukei = goukei + 500;
}
if (goukei == 0) {
souryou = 0;
}
document.forms[0].kekka.value = goukei + souryou;
```

```
1 <!DOCTYPE HTML>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <title>
6 お取り寄せ
7 </title>
8 <link rel="stylesheet" href="style.css" type="text/css">
9 <script>
10 function ks() {
11 }
12 </script>
13 </body>
```

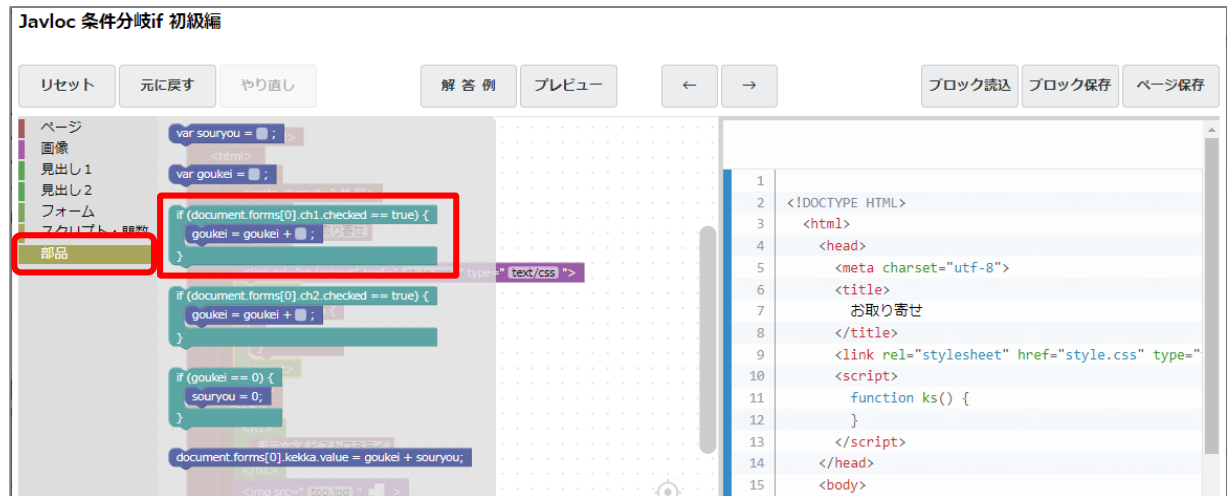
空欄に「0」と入力。※必ず半角で入力しましょう。

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>
表示文字 お取り寄せ
</title>
<link rel="stylesheet" href="style.css" type="text/css">
<script>
function ks() {
var souryou = 500;
var goukei = 0;
}
</script>
</head>
<body>
<h1>
表示文字 ビストロミライ
</h1>
</body>
</html>
```

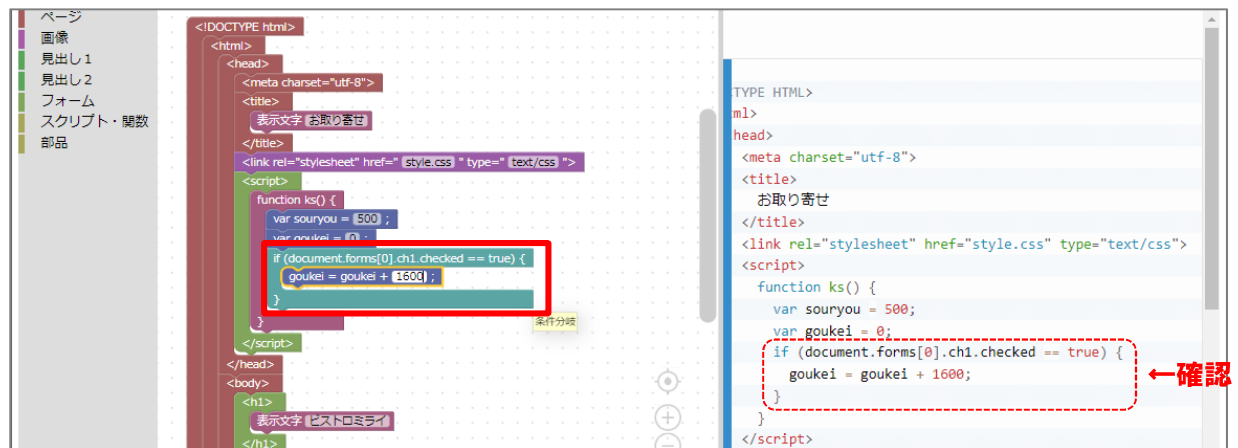
```
1 <!DOCTYPE HTML>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <title>
6 お取り寄せ
7 </title>
8 <link rel="stylesheet" href="style.css" type="text/css">
9 <script>
10 function ks() {
11 var souryou = 500;
12 var goukei = 0; ← 確認
13 }
14 </script>
15 </body>
```


- ③ もし商品 (ch1) が選択されたら (ch1 にチェックが入っていたら) 合計金額「goukei」に 1600 を足して「goukei」に入れる。

リスト内[部品]をクリック、枠内にある `if (document.forms[0].ch1.checked == true) { ~ }` ブロックセットを選択し、ドラッグして②で設定したブロックの下にドロップ。



`goukei = goukei +` の後ろの空欄に「1600」と入力。※必ず半角で入力しましょう。



ブロックの中身を見てみましょう。

```
if (document.forms[0].ch1.checked == true) {
  goukei = goukei + 1600;
}
```

「==」は「等しい」を意味する比較演算子です。代入演算子「=」と間違えやすいので注意しましょう。

これを日本語で読み下すと

もし (フォーム内の ch1 のチェックボックスにチェックが入ったら) {
変数 goukei に 1600 を足して変数 goukei に入れる処理を行う
}

<作成ページ ch1 のチェックボックス部分抜粋>



ここにチェックが入っていれば上記の処理が行われ、④へ
ここにチェックが入っていなければ上記の処理は行われず、④へ

上記のように「もし (A) だったら、{B} する」という条件式は「if(A) {B}」と記述します。

- ④ もし商品 (ch2) が選択されたら (ch2 にチェックが入っていたら) 合計金額「goukei」に 2700 を足して「goukei」に入れる。

リスト内[部品]をクリック、枠内にある `if (document.forms[0].ch2.checked == true) { ~ }` ブロックセットを選択し、ドラッグして③で設定したブロックの下にドロップ。

Javloc 条件分岐if 初級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ページ 画像 見出し1 見出し2 フォーム スクリプト・関数 部品

```

var souryou = 0;
var goukei = 0;
<meta charset="utf-8">
if (document.forms[0].ch1.checked == true) {
  goukei = goukei + 1500;
}
if (document.forms[0].ch2.checked == true) {
  goukei = goukei + 2700;
}
if (goukei == 0) {
  souryou = 0;
}
document.forms[0].kekka.value = goukei + souryou;

```

```

1 <!DOCTYPE HTML>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <title>
6     お取り寄せ
7   </title>
8   <link rel="stylesheet" href="style.css" type="text/css">
9   <script>
10     function ks() {
11       }
12   </script>
13 </head>
14 <body>

```

`goukei = goukei +` の後ろの空欄に「2700」と入力。※必ず半角で入力しましょう。

Javloc 条件分岐if 初級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ページ 画像 見出し1 見出し2 フォーム スクリプト・関数 部品

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>
  表示文字 (お取り寄せ)
</title>
<link rel="stylesheet" href="style.css" type="text/css">
<script>
  function ks() {
    var souryou = 500;
    var goukei = 0;
    if (document.forms[0].ch1.checked == true) {
      goukei = goukei + 1500;
    }
    if (document.forms[0].ch2.checked == true) {
      goukei = goukei + 2700;
    }
  }
</script>
</head>
<body>
<h1>
  表示文字 (リストロミイ)
</h1>

```

```

TYPE HTML
ml>
head>
<meta charset="utf-8">
<title>
  お取り寄せ
</title>
<link rel="stylesheet" href="style.css" type="text/css">
<script>
  function ks() {
    var souryou = 500;
    var goukei = 0;
    if (document.forms[0].ch1.checked == true) {
      goukei = goukei + 1500;
    }
    if (document.forms[0].ch2.checked == true) {
      goukei = goukei + 2700;
    }
  }
</script>

```

←確認

- ⑤ もし合計金額「goukei」が0だったら送料「souryou」を0とする。

リスト内[部品]をクリック、枠内にある `if (goukei == 0) { ~ }` ブロックセットを選択し、ドラッグ。

Javloc 条件分岐if 初級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ページ 画像 見出し1 見出し2 フォーム スクリプト・関数 部品

```

var souryou = 0;
var goukei = 0;
<meta charset="utf-8">
if (document.forms[0].ch1.checked == true) {
  goukei = goukei + 1500;
}
if (document.forms[0].ch2.checked == true) {
  goukei = goukei + 2700;
}
if (goukei == 0) {
  souryou = 0;
}
document.forms[0].kekka.value = goukei + souryou;

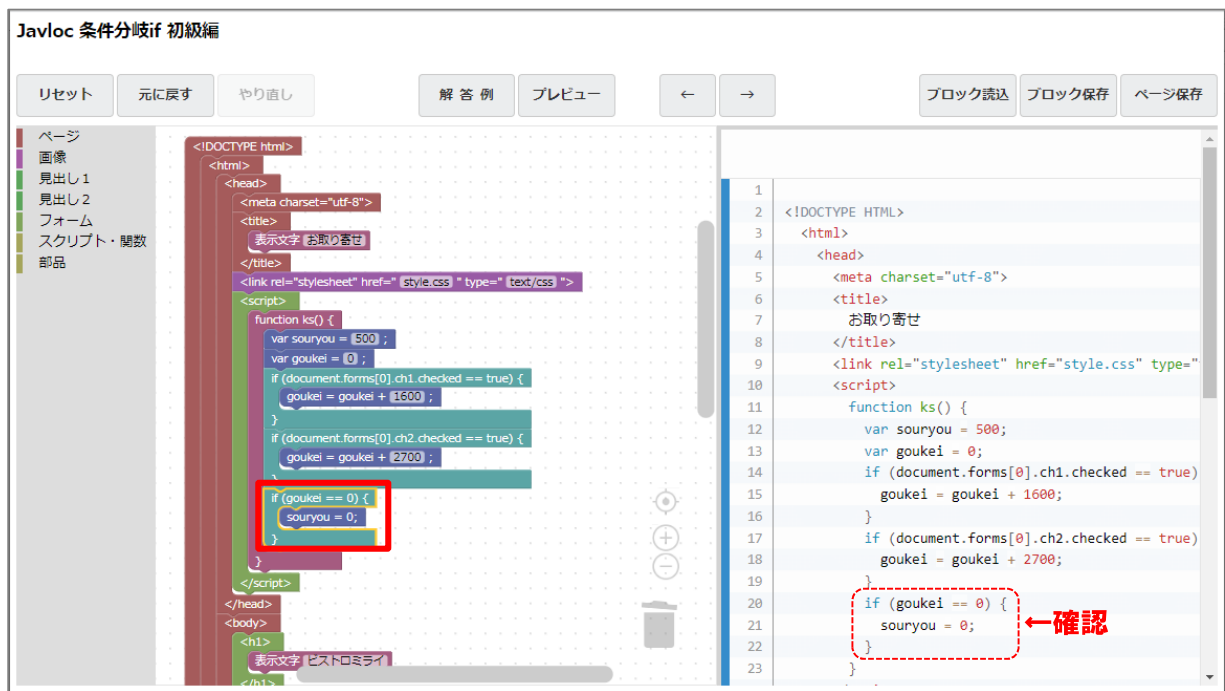
```

```

1 <!DOCTYPE HTML>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <title>
6     お取り寄せ
7   </title>
8   <link rel="stylesheet" href="style.css" type="text/css">
9   <script>
10     function ks() {
11       }
12   </script>
13 </head>
14 <body>

```

④で設定したブロックの下にドロップ。

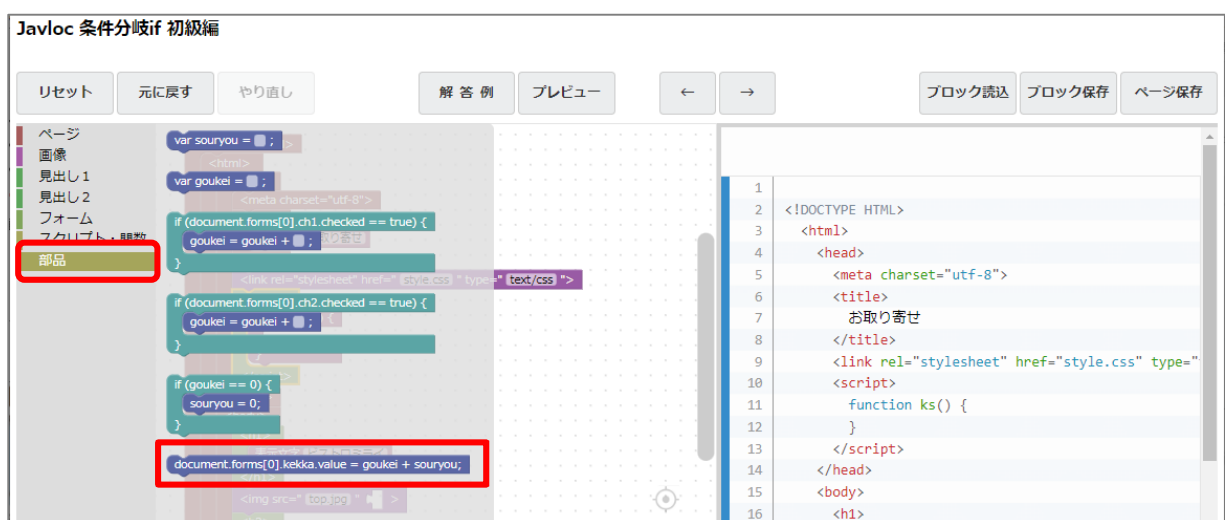


③と④では、1600 円の商品が選択された場合と 2700 円の商品が選択された場合の処理を設定しました。さらに⑤では、変数「goukei」が 0 だった場合（両方とも選択されなかった場合）、送料の変数「souryou」を 0 にするという処理を設定しました。

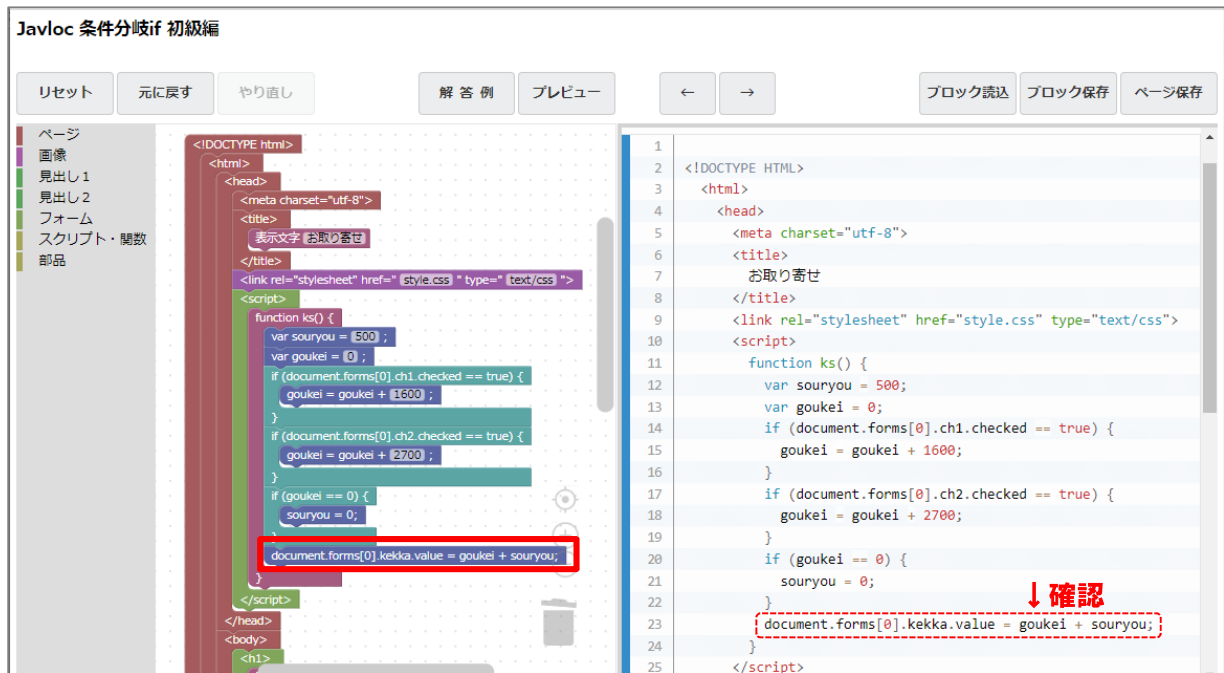
⑤を設定していなければ、両方のチェックが入っていないときに結果が「500」となってしまいます。何も選んでいないのに送料の 500 円だけ発生するという状態になってしまうのです。それを防ぐために⑤を設定します。

⑥ 合計「goukei」に送料「souryou」を足して、テキスト（kekka）に表示する。

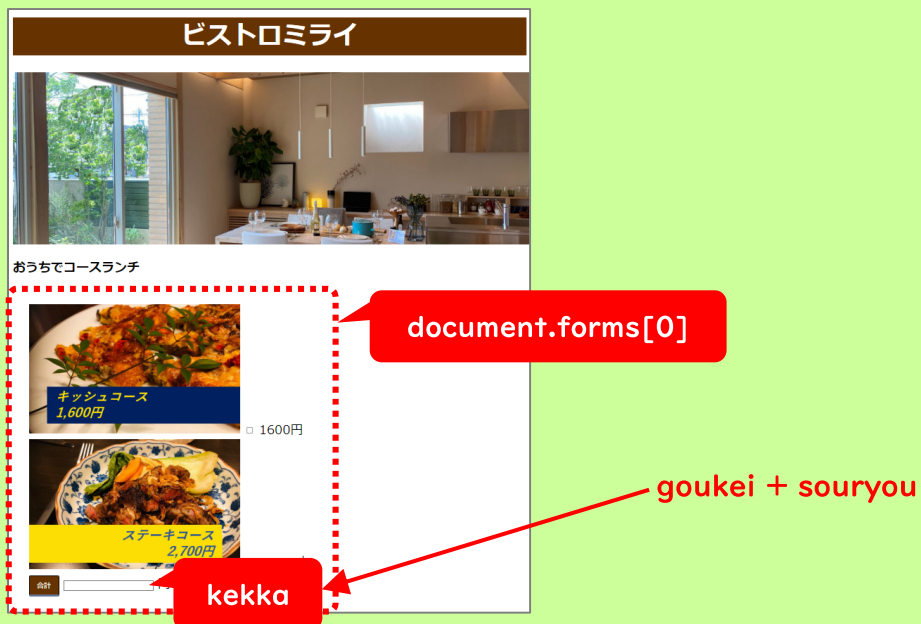
リスト内[部品]をクリック、枠内にある `document.forms[0].kekka.value = goukei + souryou;` ブロックを選択し、ドラッグ。



⑤で設定したブロックの下にドロップ。



フォーム（`document.forms[0]`）内のテキスト（名前：`kekka`）に商品の合計金額が入った変数「`goukei`」と送料が入った変数「`souryou`」を足した値を返します。



5. プレビュー画面で確認しなさい。

[プレビュー]ボタンをクリックし、動作を確認。



4. で解答例を基に確認した内容と同様の動作になっているか見てみましょう。

2. 条件分岐 if 中級 解答手順

<問題文>

日本情報処理検定協会

HTML+JavaScriptを学ぼう

プログラミング問題 条件分岐if 中級

<問題>

1. 【条件分岐 if 中級】を開いて解答する。
2. 内・網かけ部分は入力文字または値とする。

<処理条件>

1. 【ページ】のブロックセットを挿入しなさい。
2. <title>ブロックの中にページタイトルを入力しなさい。ページタイトルは下記のとおりとする。

お取り寄せ
3. <body>ブロックの中に下記の(1)から順にブロックを挿入し、処理をしなさい。
 - (1) 【見出し1】ブロックを挿入し、下記の文字を入力しなさい。

ビストロミライ
 - (2) 【画像】ブロックを挿入し、ファイル名を top.jpg にしなさい。
 - (3) 【見出し2】ブロックを挿入し、下記の文字を入力しなさい。

おうちでコースランチ
 - (4) 【フォーム】から<form>ブロックを選択・挿入し、その中に下記のブロックを挿入し、処理をしなさい。
 - ① 【画像】ブロックを挿入し、ファイル名を 01.jpg にしなさい。
 - ② 【フォーム】からチェックボックスのブロックを選択・挿入しなさい。name は ch1、value は 1。
 - ③ 【文字入力】ブロックを挿入し、下記の文字を入力しなさい。

1600 円
 - ④ 【改行】ブロックを挿入しなさい。
 - ⑤ 【画像】ブロックを挿入し、ファイル名を 02.jpg にしなさい。
 - ⑥ 【フォーム】からチェックボックスのブロックを選択・挿入しなさい。name は ch2、value は 2。
 - ⑦ 【文字入力】ブロックを挿入し、下記の文字を入力しなさい。

2700 円
 - ⑧ 【改行】ブロックを挿入しなさい。
 - ⑨ 【フォーム】からボタンのブロックを選択・挿入しなさい。value は合計、onclick は関数名 ks。
 - ⑩ 【フォーム】からテキストのブロックを選択・挿入しなさい。name は kekka。
 - ⑪ 【文字入力】ブロックを挿入し、下記の文字を入力しなさい。

円 (送料 500 円込み)

<作成ページ例>



4. <head>ブロックの中の<link>ブロックの下に、下記の指示通りブロックを挿入し、処理をしなさい。

[前提]

選択された商品金額の合計と送料を表示させる。

[機能]

- ・ フォームのチェックボックスでチェックされた項目の金額を合計し、送料を足して表示する。
- ・ いずれもチェックされていない場合、送料は発生しない。
- ・ 送料は 500 円、商品 (ch1) は 1600 円、商品 (ch2) は 2700 円とする。
- ・ 合計金額をフォームのテキストに表示する。

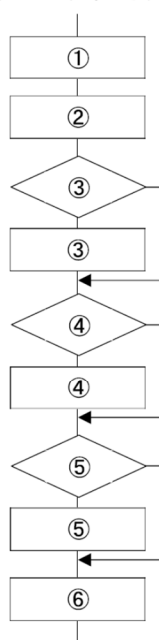
- (1) 【スクリプト・関数】ブロックを挿入し、3-(4)-⑨で指定した関数名を入力しなさい。
- (2) 【部品】内から正しいブロックを選択し、(1)の関数ブロックの中に下記処理手順どおりに組み立てなさい。

■処理手順

- ① 変数 souryou を宣言し、500 を代入
- ② 変数 goukei を宣言し、0 を代入
- ③ フォームのチェックボックス (ch1) が true の場合、goukei に 1600 を加算して代入
- ④ フォームのチェックボックス (ch2) が true の場合、goukei に 2700 を加算して代入
- ⑤ goukei が 0 の場合、souryou に 0 を代入
- ⑥ goukei に souryou を加算し、フォームのテキスト (kekka) に代入

5. プレビュー画面で確認しなさい。

<フローチャート>



教材サイトからアプリを開く

Javloc 教材サイトから[中級プロジェクト]の[条件分岐「if」 中級]をクリック。

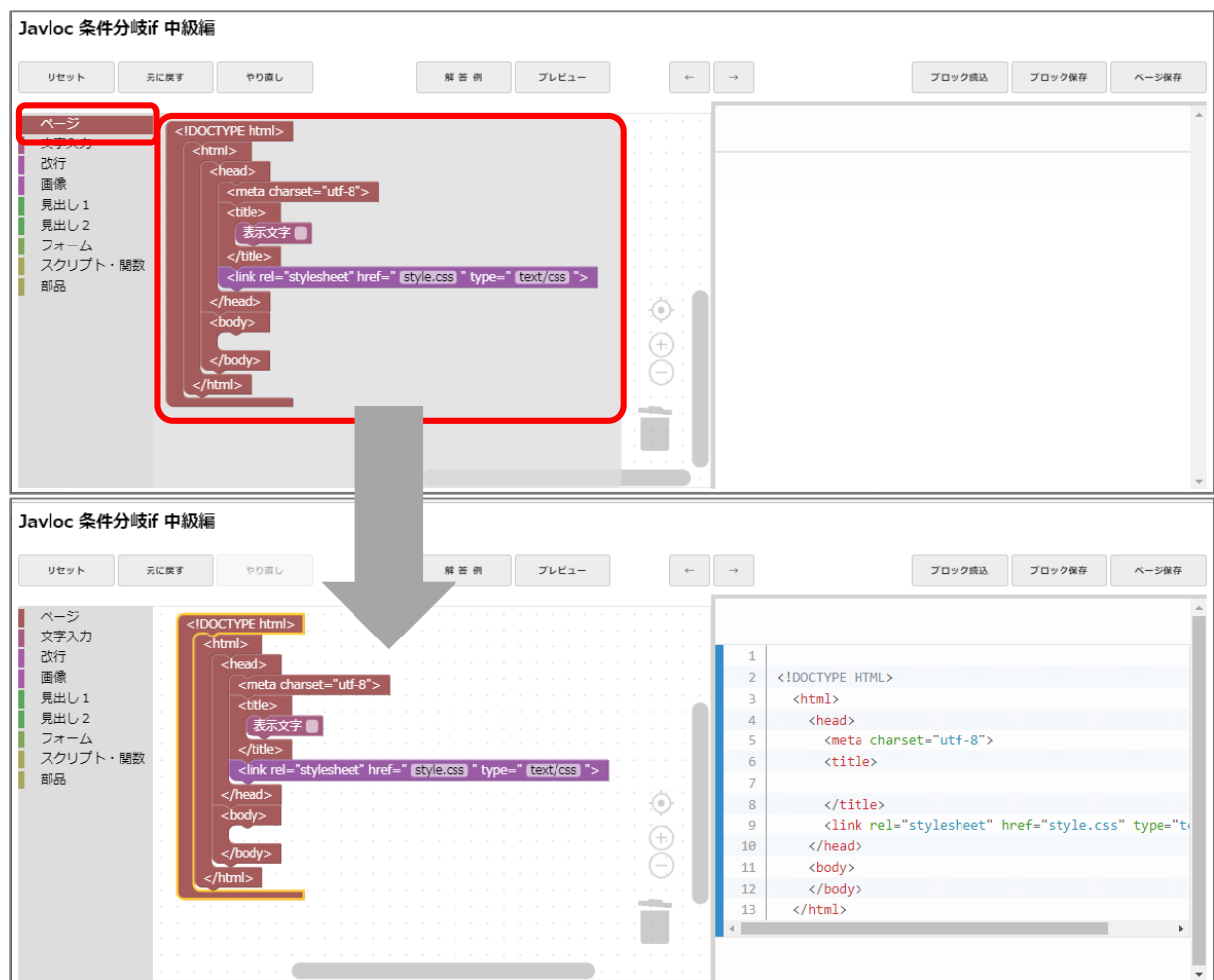


ページを構成（HTML 部分の作成）

1. 【ページ】のブロックセットを挿入しなさい。

リスト内[ページ]をクリック。

枠内にあるブロックセットをドラッグし、編集エリアにドロップ。



2. <title>ブロックの中にページタイトルを入力しなさい。ページタイトルは下記のとおりとする。

お取り寄せ

<title>ブロックの表示文字ブロックの空欄に「お取り寄せ」と入力。

入力後、ソース表示エリア・プレビューでページタイトルが設定されていることを確認。



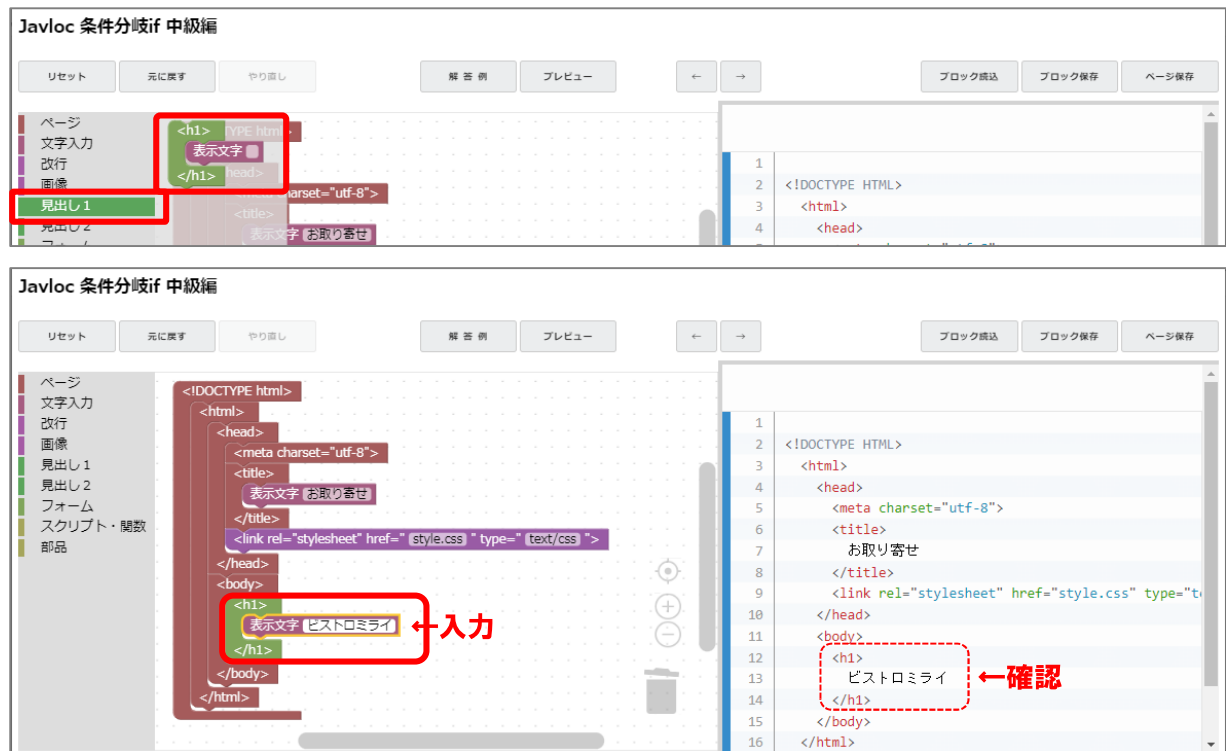
3. <body>ブロックの中に下記の(1)から順にブロックを挿入し、処理をしなさい。

(1) 【見出し 1】ブロックを挿入し、下記の文字を入力しなさい。

ビストロミライ

リスト内[見出し 1]をクリック、枠内にある<h1>ブロックをドラッグし、<body>ブロックの中にドロップ。

表示文字ブロックの空欄に「ビストロミライ」と入力し、ソース表示エリア・プレビューで見出し 1 が設定されていることを確認。



(2) 【画像】ブロックを挿入し、ファイル名を top.jpg にしなさい。

リスト内[画像]をクリック、枠内にあるブロックをドラッグし<h1>ブロックの下にドロップ。
“”の空欄の中に「top.jpg」と入力し、ソース表示エリア・プレビューで画像が設定されていることを確認。

The first screenshot shows the 'Javloc 条件分岐if 中級編' interface. On the left, the '画像' (Image) block is selected in the sidebar. In the main workspace, an '' block is being dragged. The right pane shows the HTML source code with the following structure:

```
1 <!DOCTYPE HTML>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <title>
6     お取り寄せ
7   </title>
8   <link rel="stylesheet" href="style.css" type="text/css">
9 </head>
10 <body>
11   <h1>
12     ビストロミライ
13   </h1>
14   <img src="">
15 </body>
16 </html>
```

The second screenshot shows the same interface after the image block has been inserted. The '' block is now in the workspace. The right pane shows the updated HTML source code:

```
1 <!DOCTYPE HTML>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <title>
6     お取り寄せ
7   </title>
8   <link rel="stylesheet" href="style.css" type="text/css">
9 </head>
10 <body>
11   <h1>
12     ビストロミライ
13   </h1>
14   
15 </body>
16 </html>
```

Red annotations in the second screenshot include '↑入力' (input) pointing to the 'top.jpg' text and '←確認' (confirm) pointing to the '' block in the preview pane.

(3) 【見出し2】ブロックを挿入し、下記の文字を入力しなさい。

おうちでコースランチ

リスト内[見出し2]をクリック、枠内にある<h2>ブロックをドラッグ。ブロックの下にドロップ。

表示文字ブロックの空欄に「おうちでコースランチ」と入力し、ソース表示エリア・プレビューで見出し2が設定されていることを確認。

The first screenshot shows the 'Javloc 条件分岐if 中級編' interface. On the left, the '見出し2' (Heading 2) block is selected in the sidebar. In the main workspace, an '<h2>' block is being dragged. The right pane shows the HTML source code with the following structure:

```
1 <!DOCTYPE HTML>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <title>
6     お取り寄せ
7   </title>
8   <link rel="stylesheet" href="style.css" type="text/css">
9 </head>
10 <body>
11   <h1>
12     ビストロミライ
13   </h1>
14   
15   <h2>
16     おうちでコースランチ
17   </h2>
18 </body>
19 </html>
```

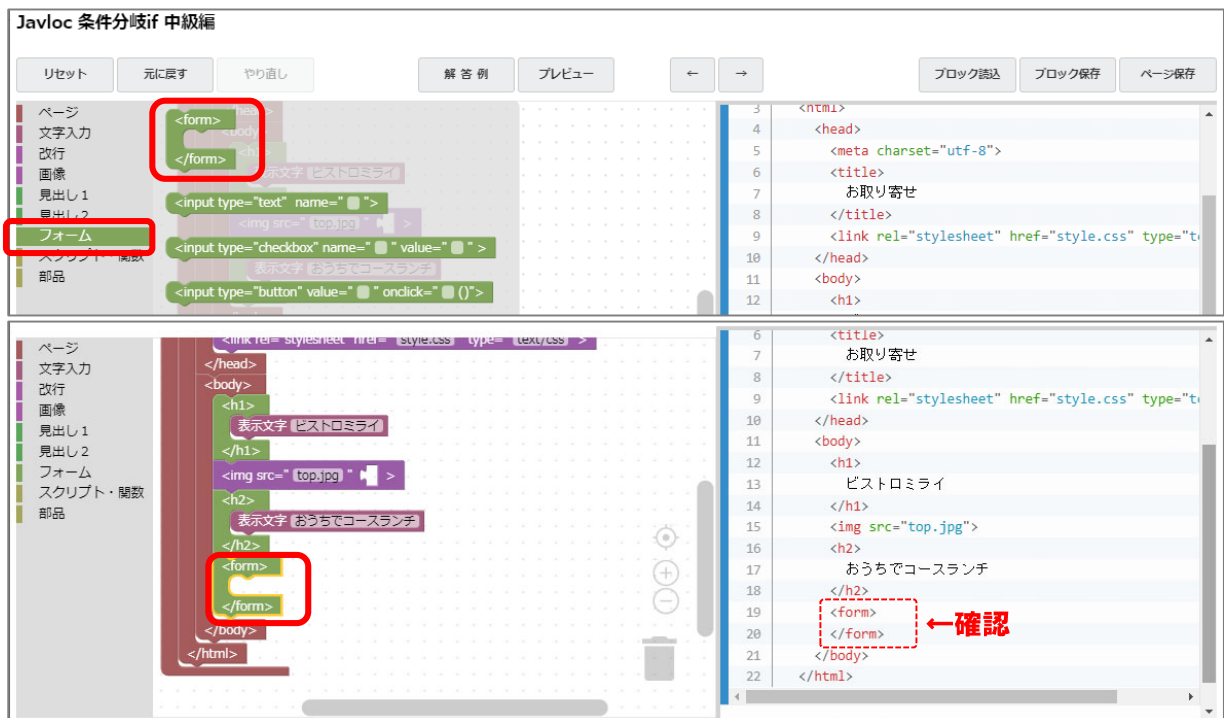
The second screenshot shows the same interface after the heading 2 block has been inserted. The '<h2>' block is now in the workspace. The right pane shows the updated HTML source code:

```
1 <!DOCTYPE HTML>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <title>
6     お取り寄せ
7   </title>
8   <link rel="stylesheet" href="style.css" type="text/css">
9 </head>
10 <body>
11   <h1>
12     ビストロミライ
13   </h1>
14   
15   <h2>
16     おうちでコースランチ
17   </h2>
18 </body>
19 </html>
```

Red annotations in the second screenshot include '↑入力' (input) pointing to the 'おうちでコースランチ' text and '←確認' (confirm) pointing to the '<h2>' block in the preview pane.

- (4) 【フォーム】から<form>ブロックを選択・挿入し、その中に下記のブロックを挿入し、処理を下さい。

リスト内[フォーム]をクリック、枠内にある<form>ブロックをドラッグし<h2>ブロックの下にドロップ。ソース表示エリアフォームが設定されていることを確認。



<form>タグは、入力欄や送信ボタンを作成する際に使用する要素です。ここで入力された情報はプログラムを使用して Web サーバーへ送信することができるので、アンケートや会員登録などのページで利用されています。

本教材では、ここに入力された情報を JavaScript でコントロールします。

手順通りに完成した後に動きを確認すると分かりやすいので、今の時点では、この(4)で組み立てたブロックが「情報を入力する場所」となるというイメージを持っていれば大丈夫です。

- ① 【画像】ブロックを挿入し、ファイル名を 01.jpg に下さい。

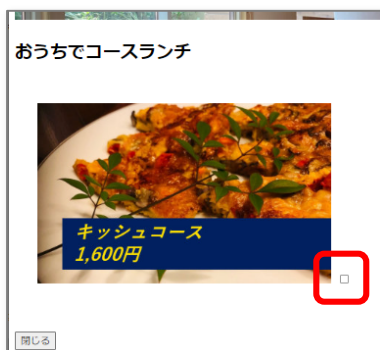
リスト内[画像]をクリック、枠内にあるブロックをドラッグし、<form>ブロックの中にドロップ。""の空欄に「01.jpg」と入力し、ソース表示エリア・プレビューで画像が設定されていることを確認。※必ず半角で入力しましょう。



② 【フォーム】からチェックボックスのブロックを選択・挿入しなさい。name は ch1、value は 1 。

リスト内[フォーム]をクリック、枠内にある<input type="checkbox" ~>ブロックをドラッグし、ブロックの下にドロップ。name の""空欄に「ch1」、value の""空欄に「1」と入力し、ソース表示エリア・プレビューでチェックボックスが設定されていることを確認。

※必ず半角で入力しましょう。



③ 【文字入力】ブロックを挿入し、下記の文字を入力しなさい。

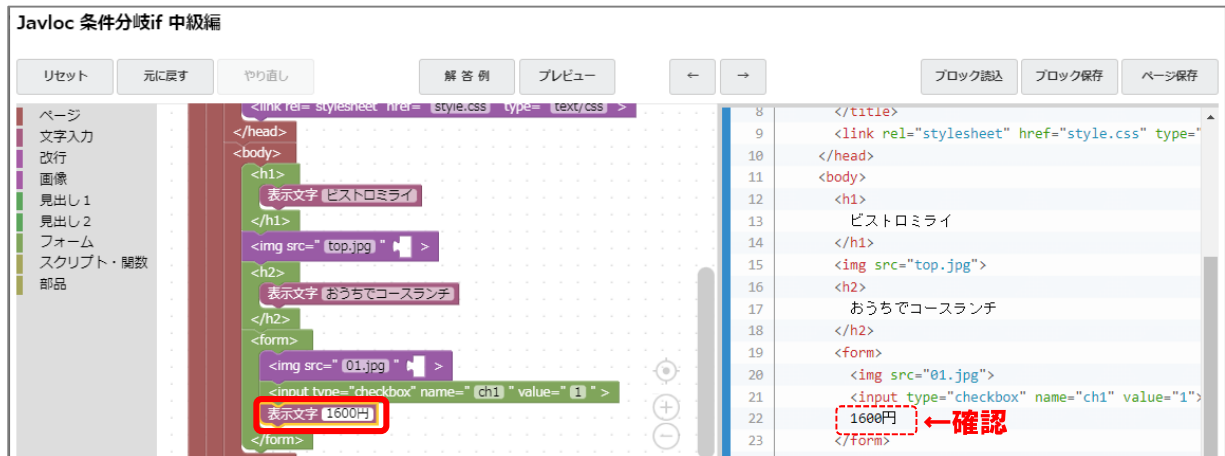
1600 円

リスト内[文字入力]をクリック、枠内にある<表示文字>ブロックをドラッグ。



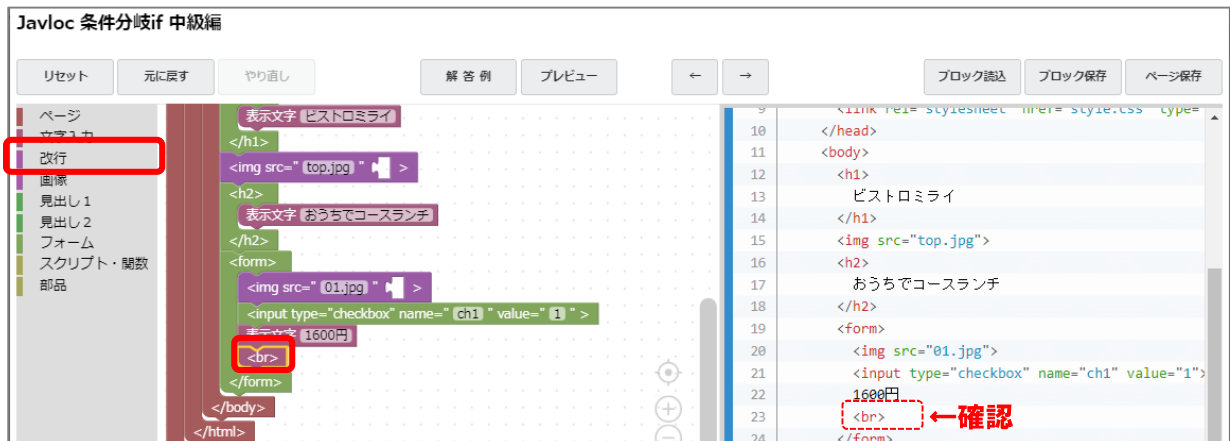
②で設定したブロックの下にドロップ。

空欄に「1600 円」と入力し、ソース表示エリア・プレビューで文字が設定されていることを確認。 ※表示される文字の設定なので全角入力でも問題ありません。



④ 【改行】ブロックを挿入しなさい。

リスト内[改行]をクリック、枠内にある
ブロックをドラッグ。③で設定したブロックの下にドロップ。ソース表示エリアで改行が設定されていることを確認。



※この改行がないと、次の⑤で設定する画像が「1600 円」の右側に配置されてしまいます。

- ⑤ 【画像】ブロックを挿入し、ファイル名を 02.jpg にしなさい。
- ⑥ 【フォーム】からチェックボックスのブロックを選択・挿入しなさい。name は ch2、value は 2。
- ⑦ 【文字入力】ブロックを挿入し、下記の文字を入力しなさい。

2700 円
- ⑧ 【改行】ブロックを挿入しなさい。

①～④と同様に、⑤～⑧を設定。ソース表示エリア・プレビューで設定を確認。

Javloc 条件分岐if 中級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ブロック一覧: ページ, 文字入力, 改行, 画像, 見出し1, 見出し2, フォーム, スクリプト・関数, 部品

ブロック内容:

```


<h2>
  表示文字 おうちでコースランチ
</h2>
<form>
  
  <input type="checkbox" name="ch1" value="1" />
  表示文字 1600円
   ←02.jpg ※必ず半角
  <input type="checkbox" name="ch2" value="2" /> ch2 ↑ ※必ず半角
  表示文字 2700円 ↑ 2 ※必ず半角
</form>
</body>
</html>

```

ソース表示エリア:

```

<body>
<h1>
  ビストロミライ
</h1>

<h2>
  おうちでコースランチ
</h2>
<form>
  
  <input type="checkbox" name="ch1" value="1" />
  1600円
  <br>
  
  <input type="checkbox" name="ch2" value="2" />
  2700円
  <br>
</form>
</body>
</html>

```

プレビュー:

確認

- ⑨ 【フォーム】からボタンのブロックを選択・挿入しなさい。value は合計、onclick は関数名 ks。

リスト内[フォーム]をクリック、枠内にある `<input type="button" ~>` ブロックをドラッグ。

Javloc 条件分岐if 中級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ブロック一覧: ページ, 文字入力, 改行, 画像, 見出し1, 見出し2, フォーム, スクリプト・関数, 部品

ブロック内容:

```

<form>
  <body>
  <h1>
    表示文字 ビストロミライ
  </h1>
  
  <input type="checkbox" name=" " value=" " />
  <input type="button" value=" " onclick="ks()" />
</form>
</body>
</html>

```

ソース表示エリア:

```

<html>
<head>
  <meta charset="utf-8">
  <title>
    お取り寄せ
  </title>
  <link rel="stylesheet" href="style.css" type="text/css" />
</head>
<body>
  <h1>

```


⑧で設定したブロックの下にドロップ。value の""空欄に「合計」、onclick の""空欄に「ks」と入力し、ソース表示エリア・プレビューでボタンが設定されていることを確認。

※「ks」は必ず半角で入力しましょう。

Javloc 条件分岐if 中級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ページ 文字入力 改行 画像 見出し1 見出し2 フォーム スクリプト・関数 部品

```
</h2>
</form>

<input type="checkbox" name="ch1" value="1" * >
<br>

<input type="checkbox" name="ch2" value="2" * >
表示文字 1600円
<br>
<input type="button" value="合計" onclick="ks()" * >
</form>
</body>
</html>
```

```
13 ビストロミライ
14 </h1>
15 
16 <h2>
17 おうちでコースランチ
18 </h2>
19 <form>
20 
21 <input type="checkbox" name="ch1" value="1">
22 1600円
23 <br>
24 
25 <input type="checkbox" name="ch2" value="2">
26 2700円
27 <br>
28 <input type="button" value="合計" onclick="ks()" * >
29 </form>
30 </body>
31 </html>
```

↑ 確認



⑩ 【フォーム】からテキストのブロックを選択・挿入しなさい。name は kekka。

リスト内[フォーム]をクリック、枠内にある<input type="text" ~>ブロックをドラッグし、⑨で設定したブロックの下にドロップ。name の""空欄に「kekka」と入力し、ソース表示エリア・プレビューでテキストが設定されていることを確認。

Javloc 条件分岐if 中級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ページ 文字入力 改行 画像 見出し1 見出し2 フォーム スクリプト・関数 部品

```
<form>
<head>
<body>
</form>
<h1>
<input type="text" name="" * >
<input type="checkbox" name="" * value="" * >
<input type="button" value="" * onclick="" * >
```

```
3 <html>
4 <head>
5 <meta charset="utf-8">
6 <title>
7 お取り寄せ
8 </title>
9 <link rel="stylesheet" href="style.css" type="t
10 </head>
11 <body>
12 <h1>
```

Javloc 条件分岐if 中級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ページ 文字入力 改行 画像 見出し1 見出し2 フォーム スクリプト・関数 部品

```
</h2>
</form>

<input type="checkbox" name="ch1" value="1" * >
表示文字 1600円
<br>

<input type="checkbox" name="ch2" value="2" * >
表示文字 2700円
<br>
<input type="button" value="合計" onclick="ks()" * >
<input type="text" name="kekka" * >
</form>
</body>
</html>
```

```
14 </h1>
15 
16 <h2>
17 おうちでコースランチ
18 </h2>
19 <form>
20 
21 <input type="checkbox" name="ch1" value="1">
22 1600円
23 <br>
24 
25 <input type="checkbox" name="ch2" value="2">
26 2700円
27 <br>
28 <input type="button" value="合計" onclick="ks()" * >
29 <input type="text" name="kekka" * >
30 </form>
31 </body>
32 </html>
```

↑ 確認



プログラムを組み立てる（JavaScript 部分の作成）

4. <head>ブロックの中の<link>ブロックの下に、下記の指示通りブロックを挿入し、処理をなさい。

【前提】

選択された商品金額の合計と送料を表示させる。

【機能】

- ・フォームのチェックボックスでチェックされた項目の金額を合計し、送料を足して表示する。
- ・いずれもチェックされていない場合、送料は発生しない。
- ・送料は 500 円、商品（ch1）は 1600 円、商品（ch2）は 2700 円とする。
- ・合計金額をフォームのテキストに表示する。



まだこの時点では前提や機能の意味が分からなくても、解答例のページを確認し、「こんな動きのプログラムを作成するんだな」と想像するくらいで問題ありません。

まず、[前提]と[機能]を読み、「解答例」ボタンをクリックし、解答例ページの動きを確認します。

Javloc 条件分岐if 初級編

リセット 元に戻す やり直し **解答例** プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ピストロミライ

このフォームの部分を操作してみましょう。
チェックボックスにチェックを入れ「合計」ボタンを押してみます。
右図の4パターンを試してみましょう。

おうちてコースランチ

☐ キッシュコース 1,600円

☐ ステーキコース 2,700円

合計 円（送料500円込み）

閉じる

1600 + 送料 500 = 2100

☒ キッシュコース 1,600円

☐ ステーキコース 2,700円

合計 円（送料500円込み）

2,700 + 送料 500 = 3200

☐ キッシュコース 1,600円

☒ ステーキコース 2,700円

合計 円（送料500円込み）

チェックボックスで選択された商品の金額を足して、さらに送料 500 円を足した合計金額がテキスト内に表示される。

商品が何も選択されていないときは、送料は足されず、0 円となる。

1600 + 2700 + 送料 500 = 4800

☒ キッシュコース 1,600円

☒ ステーキコース 2,700円

合計 円（送料500円込み）

両方チェックがなければ
送料も発生しない→0

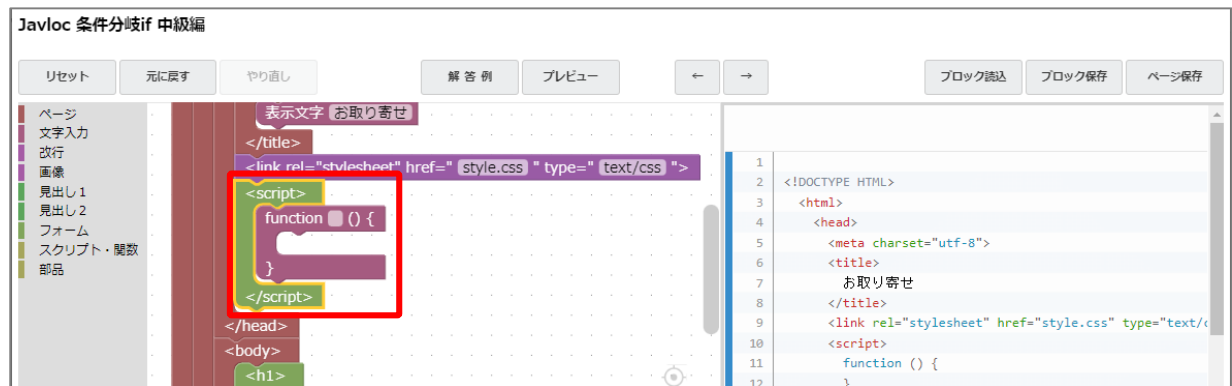
☐ キッシュコース 1,600円

☐ ステーキコース 2,700円

合計 円（送料500円込み）

(1) 【スクリプト・関数】ブロックを挿入し、3-(4)-⑨で指定した関数名を入力しなさい。

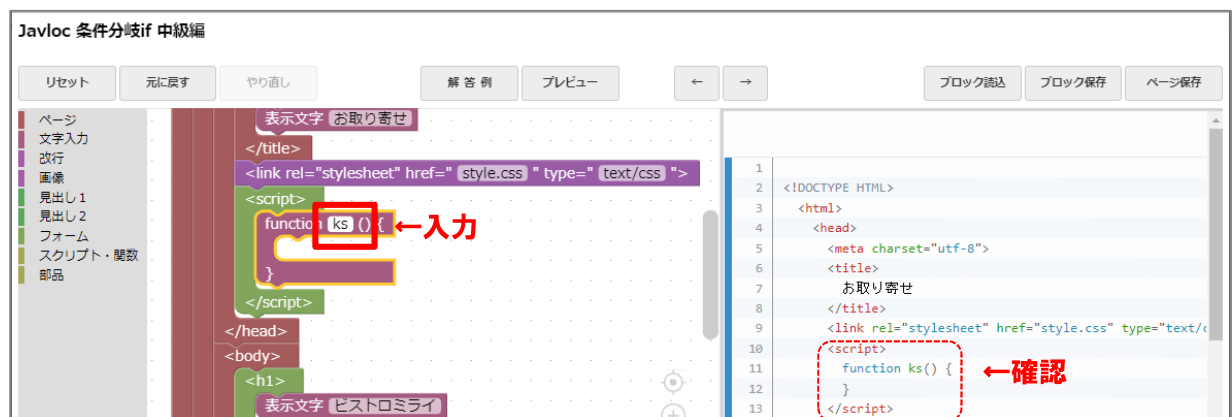
リスト内[スクリプト・関数]をクリック、枠内にある<script>ブロックセットをドラッグし、<link>ブロックの下にドロップ。



3-(4)-⑨で挿入・入力したボタンのブロック `<input type="button" value=" 合計 " onclick=" ks ()"` にある `onclick="ks()"` を参照し、関数名が「ks」だと確認します。

関数名「ks」を空欄に入力。※必ず半角で入力しましょう。

ソース表示エリアでスクリプト・関数が設定されていることを確認。



function 〇 () とは、関数を定義する JavaScript です。〇部分に関数名を記述します。これは、さまざまな処理を一つにまとめて、名前を付けることができるものです。

「function ks () { ~ }」となるように入力します。

これで、合計ボタンが押されたら (onclick)、関数 (ks) が処理されるという仕組みになります。

これから、その関数内の処理を組み立てます。

(2) 【部品】内から正しいブロックを選択し、(1)の関数ブロックの中に下記処理手順どおりに組み立てなさい。

■処理手順

① 変数 `souryou` を宣言し、500 を代入

リスト内【部品】をクリック、枠内にある `var souryou =` ; ブロックを選択し、ドラッグして `function ks () { ~ }` ブロックの中にドロップ。

ブロック選択のヒント 問題文に「変数」とあったら「var」を含むブロックを選びましょう。

Javloc 条件分岐if 中級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ページ 文字入力 改行 画像 見出し1 見出し2 フォーム スクリプト、関数

部品

document.forms[0].kekka.value = goukei + souryou;

if (document.forms[0].ch2.checked == true) {
goukei = goukei + ;
}

if (document.forms[0].ch2.checked == true) {
goukei = goukei + ;
function ks() {
}
}

if (document.forms[0].ch1.checked == true) {
goukei = goukei + ;
}

if (document.forms[0].ch1.checked == true) {
goukei = goukei + ;
}

if (goukei == 0) {
souryou = 0;
}

if (goukei == 0) {
souryou = 0;
}

var goukei = ;
var souryou = ;

```
1 <!DOCTYPE HTML>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <title>
6 お取り寄せ
7 </title>
8 <link rel="stylesheet" href="style.css" type="text/css">
9 <script>
10 function ks() {
11 }
12 </script>
13 </head>
14 <body>
15 <h1>
16 ビストロミライ
17 </h1>
18 
19 <h2>
20 おうちでコースランチ
21 </h2>
22 <form>
23 
24 <input type="checkbox" name="ch1" value="1">
25 <input type="checkbox" name="ch2" value="2">
26 1600円
27 <input type="button" value="注文" onclick="ks()">
```

空欄に「500」と入力。※必ず半角で入力しましょう。

Javloc 条件分岐if 中級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ページ 文字入力 改行 画像 見出し1 見出し2 フォーム

部品

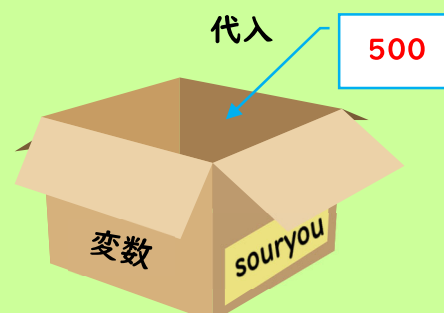
var souryou = 500;

```
9 <link rel="stylesheet" href="style.css" type="text/css">
10 <script>
11 function ks() {
12 var souryou = 500; ←確認
13 }
14 </script>
```

変数 (var) : 値を入れる箱のようなもの

宣言 : 変数を用意すること

代入 (=) : 変数に値を入れること



② 変数 goukei を宣言し、0 を代入

リスト内[部品]をクリック、枠内にある `var goukei =` ; ブロックを選択してドラッグ。

ブロック選択のヒント 問題文に「変数」とあったら「var」を含むブロックを選びましょう。

Javloc 条件分岐if 中級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ページ 文字入力 改行 画像 見出し1 見出し2 フォーム **部品**

`document.forms[0].kekka.value = goukei + souryou;`

`</doctype html>`

`if (document.forms[0].ch2.checked == true) {`
`goukei = goukei + 1000;`
`}`

`</body>`

`if (document.forms[0].ch2.checked == true) {`
`goukei = goukei + 1000;`
`}`

`</script>`

`if (document.forms[0].ch1.checked == true) {`
`goukei = goukei + 1600;`
`souryou = 500;`
`}`

`</script>`

`if (document.forms[0].ch1.checked == true) {`
`goukei = goukei + 1600;`
`}`

`if (goukei == 0) {`
`souryou = 0;`
`src="top.jpg";`
`}`

`if (goukei == 0) {`
`souryou = 0;`
`src="01.jpg";`
`}`

`if (goukei == 0) {`
`souryou = 0;`
`src="02.jpg";`
`}`

`var goukei = 0;`

`var souryou = 0;`

`src="02.jpg";`

```
8 </title>
9 <link rel="stylesheet" href="style.css" type="text/css">
10 <script>
11     function ks() {
12         var souryou = 500;
13     }
14 </script>
15 </head>
16 <body>
17 <h1>
18     ビストロミライ
19 </h1>
20 
21 <h2>
22     おうちでコースランチ
23 </h2>
24 <form>
25     
26     <input type="checkbox" name="ch1" value="1">
27     1600円
28 <br>
29     
30     <input type="checkbox" name="ch2" value="2">
31     2700円
32 <br>
33     <input type="button" value="合計" onclick="ks()">
34     <input type="text" name="kekka">
35     円 (送料500円込み)
```

①で設定したブロックの下にドロップ。

空欄に「0」と入力。※必ず半角で入力しましょう。

Javloc 条件分岐if 中級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ページ 文字入力 改行 画像 見出し1 見出し2 フォーム **部品**

`<link rel="stylesheet" href="style.css" type="text/css">`

`<script>`

`function ks() {`

`var souryou = 500;`

`var goukei = 0;` ←確認

`</script>`

```
10 <script>
11     function ks() {
12         var souryou = 500;
13         var goukei = 0;
14     }
15 </script>
16 </head>
```

③ フォームのチェックボックス (ch1) が true の場合、goukei に 1600 を加算して代入

リスト内[部品]をクリック、枠内にある `if (document.forms[0].ch1.checked == true) { ~ }` ブロックセットを選択。

ブロック選択のヒント 問題文に「場合」とあったら条件式。「if」を含むブロックを選びましょう。

ページ 文字入力 改行 画像 見出し1 見出し2 フォーム **部品**

`document.forms[0].kekka.value = goukei + souryou;`

`</doctype html>`

`if (document.forms[0].ch2.checked == true) {`
`goukei = goukei + 1000;`
`}`

`</body>`

`if (document.forms[0].ch2.checked == true) {`
`goukei = goukei + 1000;`
`}`

`</script>`

`if (document.forms[0].ch1.checked == true) {`
`goukei = goukei + 1600;`
`souryou = 500;`
`}`

`</script>`

`if (document.forms[0].ch1.checked == true) {`
`goukei = goukei + 1600;`
`souryou = 500;`
`}`

`if (goukei == 0) {`
`souryou = 0;`
`src="top.jpg";`
`}`

`if (goukei == 0) {`
`souryou = 0;`
`src="01.jpg";`
`}`

`if (goukei == 0) {`
`souryou = 0;`
`src="02.jpg";`
`}`

← 選択理由は次ページに！

```
8 </title>
9 <link rel="stylesheet" href="style.css" type="text/css">
10 <script>
11     function ks() {
12         var souryou = 500;
13         var goukei = 0;
14     }
15 </script>
16 </head>
17 <body>
18 <h1>
19     ビストロミライ
20 </h1>
21 
22 <h2>
23     おうちでコースランチ
24 </h2>
25 <form>
26     
```


[部品]の中には、誤ったブロックが混ざっています。

まず、条件式を表す「if」を含むブロックに絞り、問題文のキーワードを確認して正しいものを選択しましょう。

[部品]の中の「if」を含むブロックは下図の6種類。その中から問題文にある 部分に注目。

③ フォームのチェックボックス (ch1) が true の場合、goukei に 1600 を加算して代入

`document.forms[0].ch1.checked == true` `if (~ == true){` `goukei + 1600` `=` `}`

比較演算子 == になっていない
代入 = になっている

比較演算子 == になっていない
代入 = になっている

問題文のキーワードと一致するのはこのブロック

ドラッグして②で設定したブロックの下にドロップ。

`goukei = goukei +` の後ろの空欄に「1600」と入力。※必ず半角で入力しましょう。

Javloc 条件分岐if 中級編

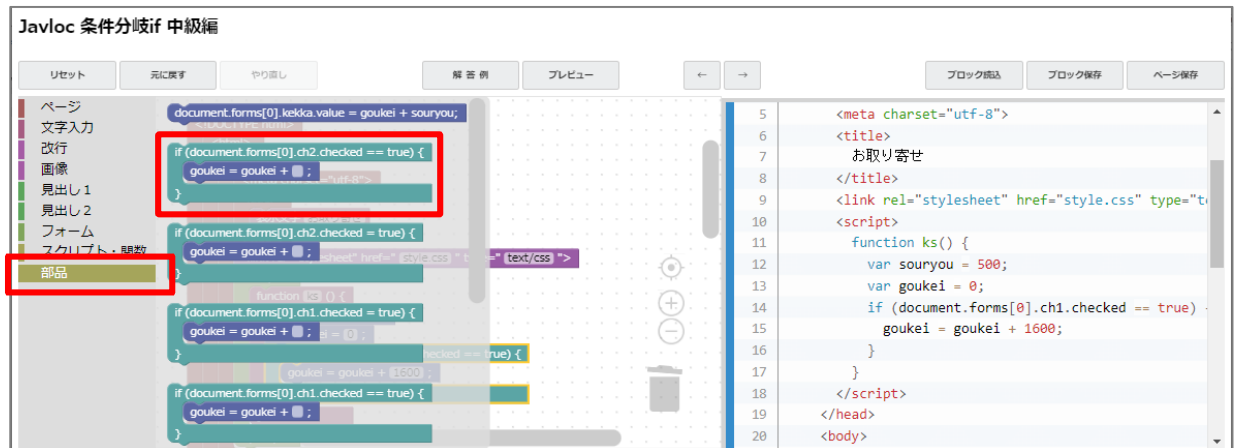
↑ 確認

ここにチェックが入っていれば上記の処理が行われ、④へ

ここにチェックが入っていなければ上記の処理は行われず、④へ

④ フォームのチェックボックス (ch2) が true の場合、goukei に 2700 を加算して代入

リスト内[部品]をクリック、枠内にある `if (document.forms[0].ch2.checked == true) { ~ }` ブロックセットを選択。

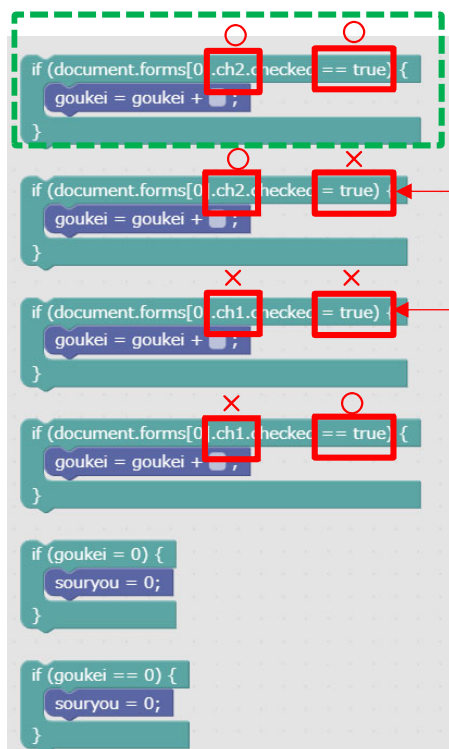


③と同様に[部品]の中から「if」を含むブロックに絞り、その中から正しいものを選びましょう。

[部品]の中の「if」を含むブロックは下図の6種類。その中から問題文にある 部分に注目。

④ フォームのチェックボックス (ch2) が true の場合、goukei に 2700 を加算して代入

`document.forms[0].checked` `if(~ == true){` `+` `=` `}`



問題文のキーワードと一致するのはこのブロック

比較演算子 == になっていない
代入 = になっている

比較演算子 == になっていない
代入 = になっている

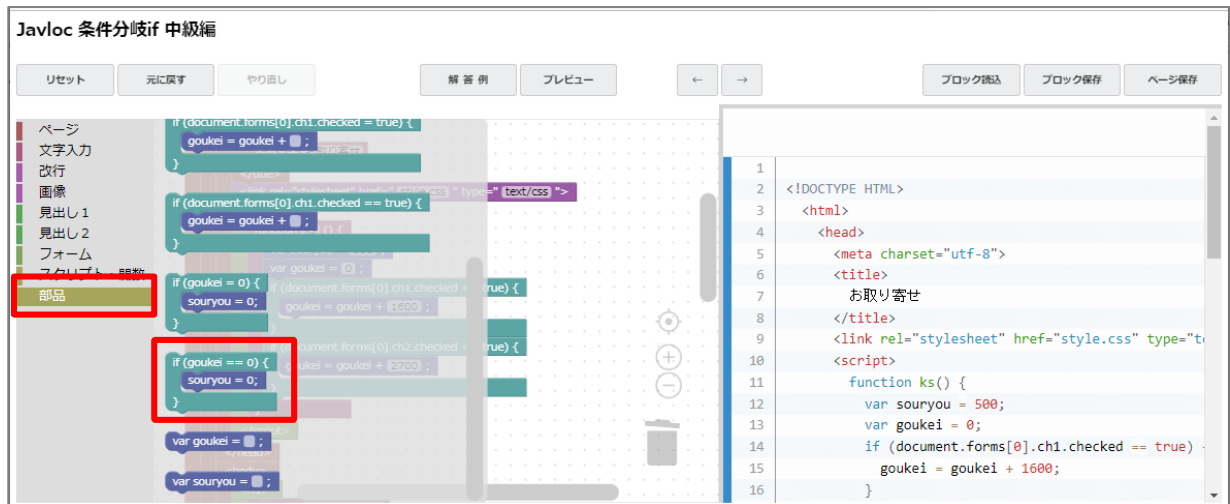
ドラッグして③で設定したブロックの下にドロップ。

goukei = goukei + の後ろの空欄に「2700」と入力。※必ず半角で入力しましょう。



⑤ goukei が 0 の場合、souryou に 0 を代入

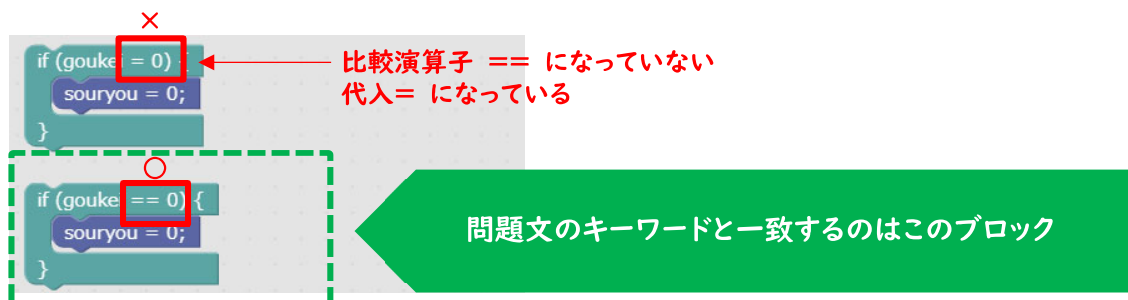
リスト内[部品]をクリック、枠内にある if (goukei == 0) { ~ } ブロックセットを選択。



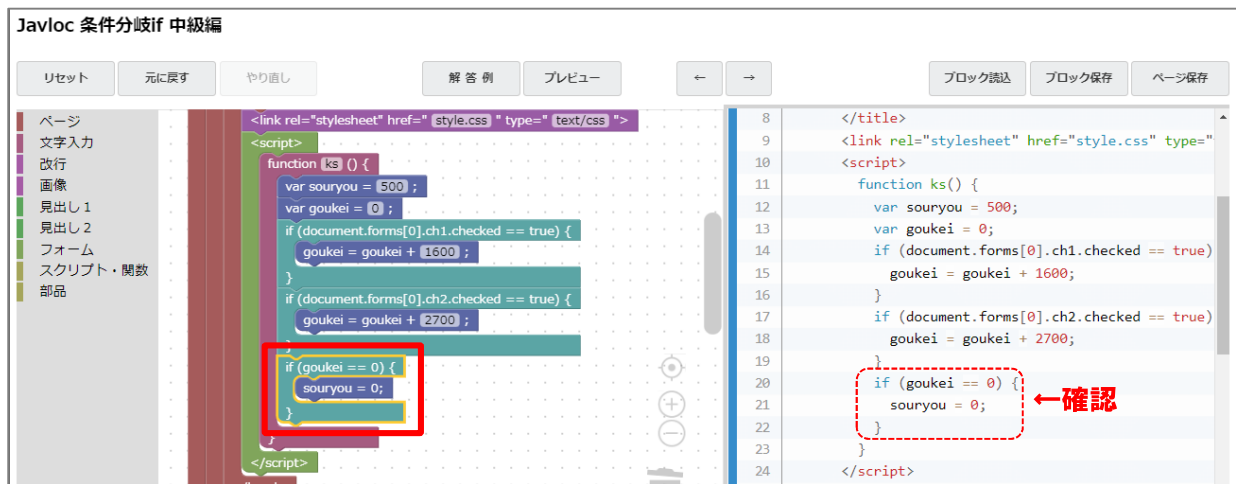
④と同様に[部品]の中から「if」を含むブロックに絞り、その中から正しいものを選びましょう。

[部品]の中の「if」を含むブロックのうち、③・④で選択肢として挙げた4つのブロックを除くと下図の2種類となる。その中から問題文にある 部分に注目。

⑤ goukei が 0 の場合 souryou に 0 を代入
if (~ == 0){ = 0 }



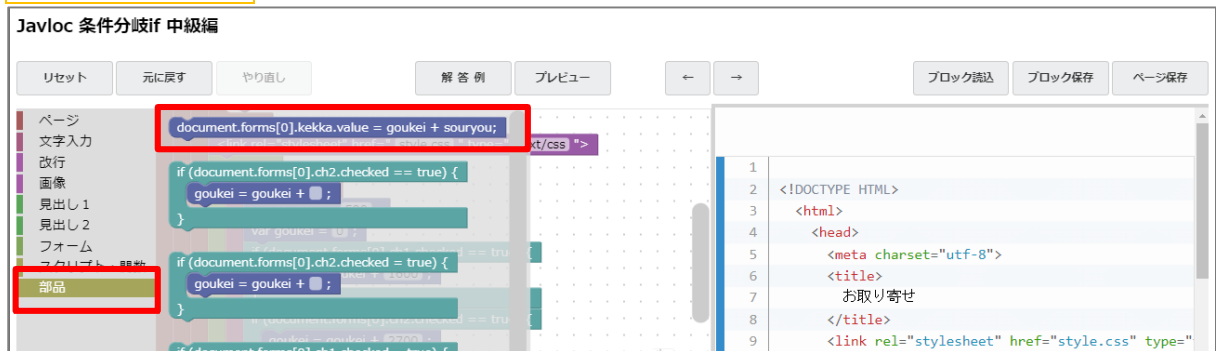
ドラッグして④で設定したブロックの下にドロップ。



⑥ goukei に souryou を加算し、フォームのテキスト (kekka) に代入

リスト内[部品]をクリック、枠内にある `document.forms[0].kekka.value = goukei + souryou;` ブロックを選択。

ブロック選択のヒント 問題文「テキスト (kekka)」を目印に「kekka」を含むブロックを選択しましょう。



ドラッグして⑤で設定したブロックの下にドロップ。



3-(4)-⑩で挿入・入力したテキストのブロック `<input type="text" name="kekka">` を参照してみましょう。このテキストの場所に `goukei + souryou` の計算結果を返し、表示します。

5. プレビュー画面で確認しなさい。

[プレビュー]ボタンをクリックし、動作を確認。



4. で解答例を基に確認した内容と同様の動作になっているか見てみましょう。

3. 条件分岐 if 上級 解答手順

<問題文>

日本情報処理検定協会
HTML+JavaScriptを学ぶ
プログラミング問題 条件分岐if 上級

<問題>

- 【条件分岐 if 上級】を開いて解答する。
- ☐ 内、制がけ部分は入力文字または値とする。

A

1. 【ページ】のブロックセットを挿入しない。

2. <title>のブロックの中にページタイトルを入力しない。ページタイトルは下記のとおりとする。

【お取り寄せ】

3. <body>ブロックの中に下記の(1)から順にブロックを挿入し、処理をしない。

(1) 【見出し1】ブロックを挿入し、下記の文字を入力しない。

【ビストロミライ】

(2) 【画像】ブロックを挿入し、ファイル名を top.jpg にしない。

(3) 【見出し2】ブロックを挿入し、下記の文字を入力しない。

【おうちでコースランチ】

(4) 【フォーム】から<form>ブロックを選択・挿入し、その中に下記のブロックを挿入し、処理をしない。

① 【画像】ブロックを挿入し、ファイル名を 01.jpg にしない。

② 【フォーム】からチェックボックスのブロックを選択・挿入しない。name は ch1、value は 1。

③ 【文字入力】ブロックを挿入し、下記の文字を入力しない。

1600 円

④ 【改行】ブロックを挿入しない。

⑤ 【画像】ブロックを挿入し、ファイル名を 02.jpg にしない。

⑥ 【フォーム】からチェックボックスのブロックを選択・挿入しない。name は ch2、value は 2。

⑦ 【文字入力】ブロックを挿入し、下記の文字を入力しない。

2700 円

⑧ 【改行】ブロックを挿入しない。

⑨ 【フォーム】からボタンのブロックを選択・挿入しない。value は合計、onclick は関数名 ke。

⑩ 【フォーム】からテキストのブロックを選択・挿入しない。name は kekka。

⑪ 【文字入力】ブロックを挿入し、下記の文字を入力しない。

0 (送料 500 円込み)

C

4. <head>ブロックの中の<link>ブロックの下に【JavaScript】から必要なブロックを選択して画面イメージ・<フローチャート>どおりの動作になるよう、【前提】・【仕様】・【使用定義】を基に構文を完成させない。

【前提】

店のランチを通販で購入する場合の金額を算出し、表示するページ。

商品の種類は2種類。購入時送料が500円必要。

【仕様】

- 画面のチェックボックスにチェックを入れて「合計」ボタンを押すと、チェックされた商品の合計金額に送料（500円）が追加された金額を表示ボックス（テキスト）に表示する。
- 送料は、商品がいくつ選択されても一律500円になるように処理する。
- チェックボックス「ch1」は1600円、チェックボックス「ch2」は2700円で処理されるように設定する。
- 何も選択されずに「合計」ボタンが押された場合は0円とする。

【使用定義】

関数

ke: 「合計」ボタンで動作する処理の関数名

変数

souryou: 送料を設定する（初期設定値: 500）

gotokai: 合計金額を保持する（初期設定値: 0）

フォーム要素（チェックボックス）の名前

ch1: 商品1チェックボックス（設定値: 1600）

ch2: 商品2チェックボックス（設定値: 2700）

フォーム要素（テキスト）の名前

kekka: 合計値を表示する

5. プレビュー画面で確認しない。

B

<画面イメージ>

(1) 初期画面

(2) 一つ目のみチェック・ボタン押下

(3) 二つ目のみチェック・ボタン押下

テキスト内には何も表示されていない。

テキスト内に合計金額が表示されている。

テキスト内に合計金額が表示されている。

(4) 両方にチェック・ボタン押下

(5) チェックなし・ボタン押下

テキスト内に合計金額が表示されている。

テキスト内の合計金額は0と表示されている。

D

<フローチャート> 自分で考えて記入してみましょう。

```

graph TD
    Start([スタート]) --> Ch1{ch1チェック}
    Ch1 -- あり --> Sum1[合計金額に1600円を追加]
    Ch1 -- なし --> Ch2{ch2チェック}
    Ch2 -- あり --> Sum2[合計金額に2700円を追加]
    Ch2 -- なし --> Total{合計ボタン押下}
    Sum1 --> Total
    Sum2 --> Total
    Total --> Kekka[kekkaに合計値を表示]
    Kekka --> End([終了])
    
```

A=HTML 部分の指示

C=JavaScript 部分の指示

まず、**AとB-(1)の画面イメージを参照し**、ページ全体の構成を**処理条件のとおり**に**組み立て**ていきます。

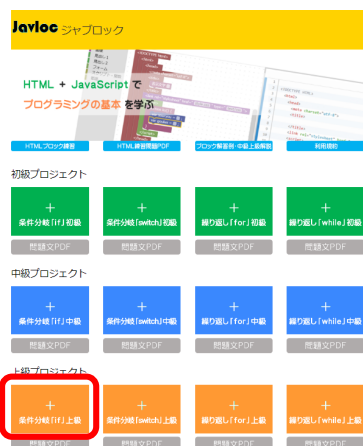
次に、**Cの[前提]・[仕様]・[使用定義]**、そして**Bの画面イメージで動作の確認**をします。

実際にブロックを組み立てる前に、**Dのフローチャートを用いて処理手順の整理**をしてみましょう。

その後、**フローチャートどおりにブロックを組み立て**ていきます。

教材サイトからアプリを開く

Javloc 教材サイトから[上級プロジェクト]の[条件分岐「if」上級]をクリック。



ページを構成（HTML 部分の作成）

1～3 は、P.30～P.38 の中級編解答方法をご参照ください。

プログラムを組み立てる（JavaScript 部分の作成）

4. <head>ブロックの中の<link>ブロックの下に【JavaScript】から必要なブロックを選択し<画面イメージ>・<フローチャート>どおりの動作になるよう、[前提]・[仕様]・[使用定義]を基に構文を完成させなさい。

[前提]

店のランチを通販で購入する場合の金額を算出し、表示するページ。

商品の種類は2種類。購入時送料が500円必要。

インプット部分の確認

3. の(4)入力フォーム

<ブロック>

<作成ページ例>

```
<form>
  
  <input type="checkbox" name=" ch1 " value=" 1 " >
  表示文字 1600円
  <br>
  
  <input type="checkbox" name=" ch2 " value=" 2 " >
  表示文字 2700円
  <br>
  <input type="button" value=" 合計 " onclick=" ks ()">
  <input type="text" name=" kekka " >
  表示文字 円（送料500円込み）
</form>
```

仕様部分を読み解く

[仕様]

- 画面のチェックボックスにチェックを入れて「合計」ボタンを押すと、チェックされた商品の合計金額に送料（500円）が付加された金額を表示ボックス（テキスト）に表示する。

→インプット部分の各設定値を確認。

→チェックされた商品の合計+送料500円を求める。

→結果はフォームのテキスト内(kekka)に返し、表示。

- 送料は、商品がいくつ選択されても一律500円になるように処理する。

→送料の箱(変数)を準備。500円と設定。

- チェックボックス「ch1」は1600円、チェックボックス「ch2」は2700円で処理されるように設定する。

→加算していくための(金額を置いておく)箱(変数)を準備。

→もし、チェックボックス ch1 にチェックが入ったら、1600円を加算。

→もし、チェックボックス ch2 にチェックが入ったら、2700円を加算。

- 何も選択されずに「合計」ボタンが押された場合は0円とする。

→もし、2つのチェックボックスにチェックが入らなかった場合、0とする。

(変数 goukei の中身が0だった場合、送料 souryou を0とする)

これに[使用定義]の内容を照らし合わせてフローチャートを作成

【使用定義】

関数

ks: 「合計」ボタンで動作する処理の関数名

→処理はすべて関数「ks」の中で処理されるため、
フローチャートに関数は含みません。

```
<script>  
  ↓ ks  
  function () {  
    }  
</script>
```

←フローチャートで作成した
処理がこの中に入る

＜フローチャート＞ 自分で考えて記入してみましょう。

変数

souryou: 送料を設定する (初期設定値: 500)

変数

goukei: 合計金額を保持する (初期設定値: 0)

もし、チェックボックス ch1にチェックが入ったら、
1600 円を加算。

フォーム要素 (チェックボックス) の名前

ch1: 商品1 チェックボックス (設定値: 1600)

もし、チェックボックス ch2にチェックが入ったら、
2700 円を加算。

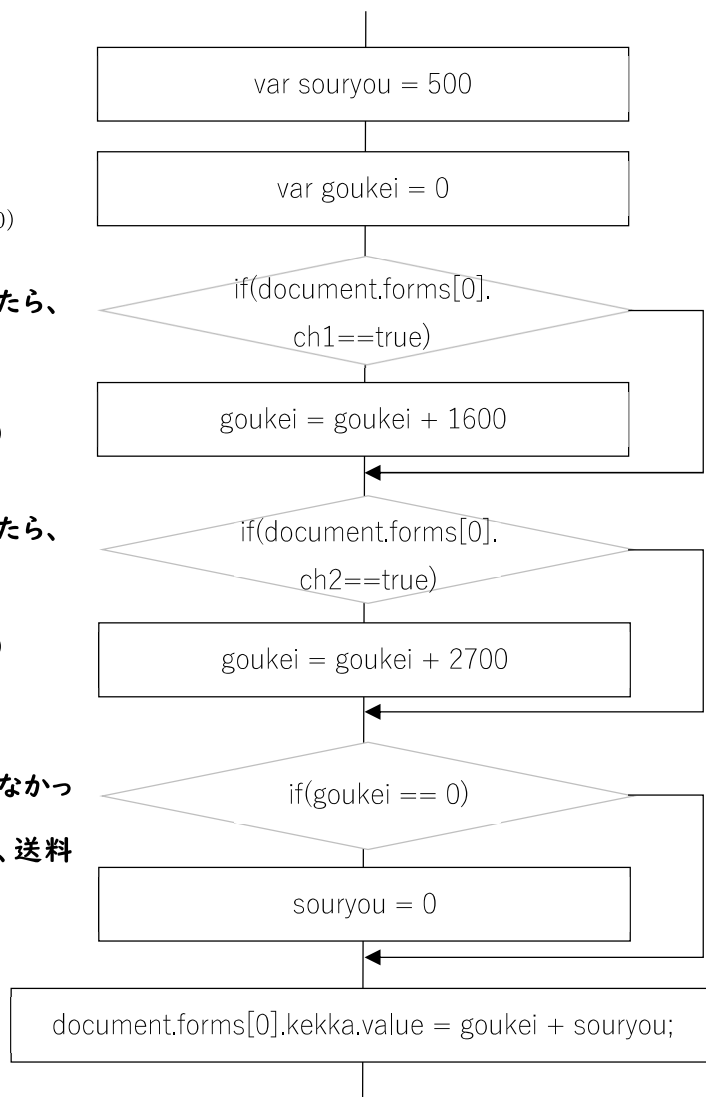
フォーム要素 (チェックボックス) の名前

ch2: 商品2 チェックボックス (設定値: 2700)

もし、2つのチェックボックスにチェックが入らなかつ
た場合、0 とする。
(変数 goukei の中身が 0 だった場合、送料
souryou を 0 とする)

フォーム要素 (テキスト) の名前

kekka: 合計値を表示する



フローチャートを基にブロックを組み立てる（プログラミング）

リスト内[JavaScript]をクリック、枠内にある<script>ブロックを選択し、ドラッグして<link>ブロックの下にドロップ。ソース表示エリアで設定を確認。

The first screenshot shows the 'Javloc 条件分岐if 上級編' interface. On the left, the 'JavaScript' block is selected in the sidebar. In the main workspace, a '<script>' block is being dragged from the sidebar. On the right, the source code area shows the current HTML structure, with the '<script>' block being added below the '<link>' block.

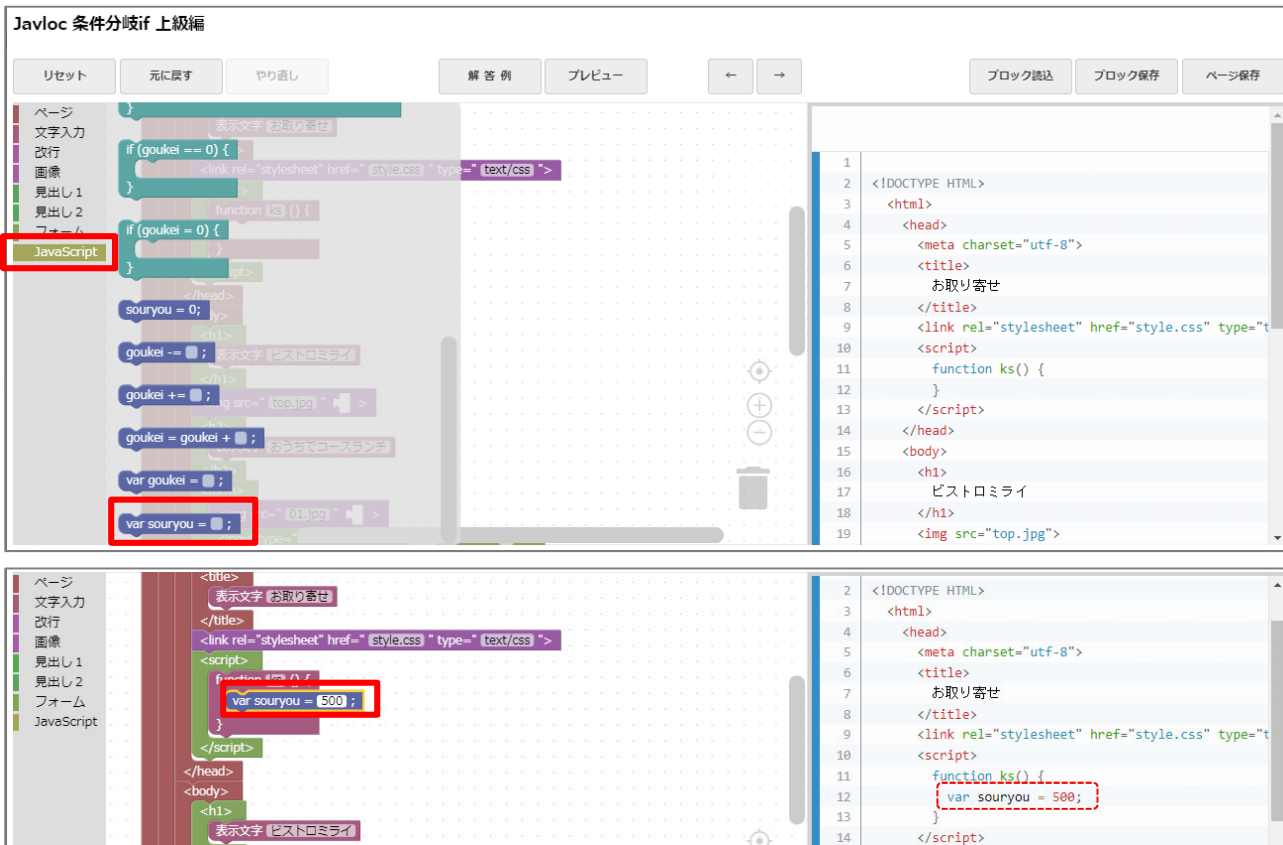
The second screenshot shows the same interface after the '<script>' block has been added. The source code area now includes the '<script>' block, and the workspace shows the updated block structure.

リスト内[JavaScript]をクリック、枠内にある<function>ブロックを選択し、ドラッグして<script>ブロックの中にドロップ。関数名「ks」を設定。ソース表示エリアで設定を確認。

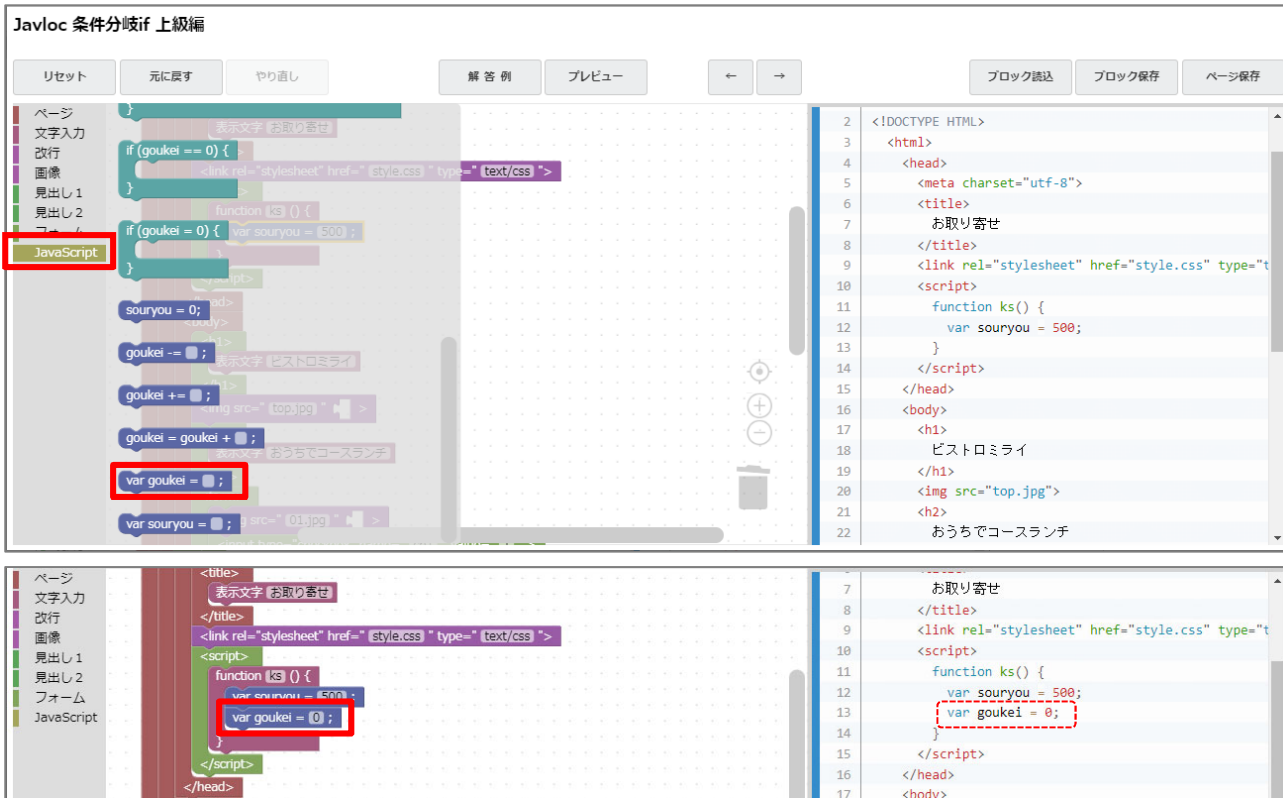
The first screenshot shows the 'Javloc 条件分岐if 上級編' interface. On the left, the 'JavaScript' block is selected in the sidebar. In the main workspace, a '<function>' block is being dragged from the sidebar. On the right, the source code area shows the current HTML structure, with the '<function>' block being added inside the '<script>' block.

The second screenshot shows the same interface after the '<function>' block has been added. The source code area now includes the '<function>' block, and the workspace shows the updated block structure.

リスト内[JavaScript]をクリック、枠内にある `var souryou =` ;ブロックを選択し、ドラッグして
 <function>ブロックの中にドロップ。数値「500」を設定。ソース表示エリアで設定を確認。



リスト内[JavaScript]をクリック、枠内にある `var goukei = ;` ブロックを選択し、ドラッグして `var souryou = ;` ブロックの下にドロップ。数値「0」を設定。ソース表示エリアで設定を確認。



リスト内[JavaScript]をクリック、枠内にある `if (document.forms[0].ch1.checked == true) {}` ブロックを選択し、ドラッグして `var goukei = ;` ブロックの下にドロップ。ソース表示エリアで設定を確認。

The screenshot shows the Javloc editor interface. On the left, the 'JavaScript' block is selected in the sidebar. The main workspace displays a code block with the following structure:

```

<title>
  表示文字 お取り寄せ
</title>
<link rel="stylesheet" href="style.css" type="text/css">
<script>
  function ks() {
    var souryou = 500;
    var goukei = 0;
    if (document.forms[0].ch1.checked == true) {
      //
    }
  }
</script>
</head>
<body>
  <h1>
    ビストロミライ
  </h1>
  
  <h2>

```

The right sidebar shows the generated HTML code, which includes the JavaScript function `ks()` and the `if` statement.

リスト内[JavaScript]をクリック、枠内にある `goukei = goukei + ;` ブロックを選択し、ドラッグして `if (document.forms[0].ch1.checked == true) {}` ブロックの中にドロップ。数値「1600」を設定。ソース表示エリアで設定を確認。

※`goukei += ;` ブロックでもOK (意味は同じ)

The screenshot shows the Javloc editor interface. On the left, the 'JavaScript' block is selected in the sidebar. The main workspace displays a code block with the following structure:

```

<title>
  表示文字 お取り寄せ
</title>
<link rel="stylesheet" href="style.css" type="text/css">
<script>
  function ks() {
    var souryou = 500;
    var goukei = 0;
    if (document.forms[0].ch1.checked == true) {
      goukei = goukei + 1600;
    }
  }
</script>
</head>
<body>
  <h1>
    ビストロミライ
  </h1>
  
  <h2>

```

The right sidebar shows the updated HTML code, which includes the `goukei = goukei + 1600;` line inside the `if` statement.

リスト内[JavaScript]をクリック、枠内にある `if (document.forms[0].ch2.checked == true) {}` ブロックを選択し、ドラッグして `if (document.forms[0].ch1.checked == true) {}` ブロックの下にドロップ。
ソース表示エリアで設定を確認。

The top screenshot shows the 'Javloc 条件分岐if 上級編' interface. On the left, the 'JavaScript' block is selected in the block list. In the main workspace, a red box highlights an existing `if (document.forms[0].ch2.checked == true) {}` block. Below it, another `if (document.forms[0].ch1.checked == true) {}` block is visible. On the right, the source code area shows the corresponding HTML and JavaScript code, with the new block being added below the first `if` statement.

The bottom screenshot shows the same workspace after the new block has been added. The new `if (document.forms[0].ch2.checked == true) {}` block is now positioned below the first `if` block. The source code area on the right reflects this change, showing the new block's code being added to the script.

リスト内[JavaScript]をクリック、枠内にある `goukei = goukei + ;` ブロックを選択し、ドラッグして `if (document.forms[0].ch2.checked == true) {}` ブロックの中にドロップ。数値「2700」を設定。ソース表示エリアで設定を確認。

※**`goukei += ;` ブロックでもOK (意味は同じ)**

The top screenshot shows the workspace with a red box highlighting the `goukei += ;` block in the block list. A red dashed box highlights the `if (document.forms[0].ch2.checked == true) {}` block in the workspace. A red arrow points to the `goukei += ;` block with the text '←これを選択してもOK' (←It's also OK to select this). The source code area on the right shows the current state of the code.

The bottom screenshot shows the workspace after the `goukei += 2700;` block has been added to the `if (document.forms[0].ch2.checked == true) {}` block. The source code area on the right shows the updated code with `goukei = goukei + 2700;` added to the script.

認心。

- ページ
- 文字入力
- 改行
- 画像
- 見出し 1
- 見出し 2
- フォーム
- JavaScript

if (goukei == 0) {} ブロックの中にドロップ。ソース表示エリアで設定を確認。

- ページ
- 文字入力
- 改行
- 画像
- 見出し 1
- 見出し 2
- フォーム
- JavaScript

リスト内[JavaScript]をクリック、枠内にある `document.forms[0].kekka.value = goukei + souryou;` ブロックを選択し、ドラッグして `if (goukei == 0) {}` ブロックの下にドロップ。ソース表示エリアで設定を確認。

Javloc 条件分岐if 上級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読み込み ブロック保存 ページ保存

ページ
文字入力
改行
画像
見出し1
見出し2
フォーム
JavaScript

```
<script>
  表示文字 (お取り寄せ)
</script>
<link rel="stylesheet" href="style.css" type="text/css">

function ks() {
  var souryou = 500;
  var goukei = 0;
  document.forms[0].kekka.value = goukei + souryou;

  if (document.forms[0].ch2.checked == true) {
    if (document.forms[0].ch2.checked == true) {
    }
  }
}
```

お取り寄せ
</title>
<link rel="stylesheet" href="style.css" type="text/css">
<script>
 function ks() {
 var souryou = 500;
 var goukei = 0;
 if (document.forms[0].ch1.checked == true)
 goukei = goukei + 1600;
 if (document.forms[0].ch2.checked == true)
 goukei = goukei + 2700;
 }
</script>
</head>
<body>

ページ
文字入力
改行
画像
見出し1
見出し2
フォーム
JavaScript

```
var goukei = 0;
if (document.forms[0].ch1.checked == true) {
  goukei = goukei + 1600;
}
if (document.forms[0].ch2.checked == true) {
  goukei = goukei + 2700;
}
if (goukei == 0) {
  souryou = 0;
  document.forms[0].kekka.value = goukei + souryou;
}
```

5. プレビュー画面で確認しなさい。

[プレビュー]ボタンをクリックし、動作を確認。

Javloc 条件分岐if 上級編

リセット 元に戻す やり直し 解答例 **プレビュー** ← → ブロック読み込み ブロック保存 ページ保存

<画面イメージ>と同様の動作になっているか見てみましょう。

＜巻末資料＞作成上の基本ルール

全般

- ・プログラムは半角英数字・半角スペースで記述する。（文字列には全角使用可）
- ・大文字と小文字は別物として扱われる。（区別される）
- ・一つの文（命令）の終わりには「;」（セミコロン）を付ける。 ; は命令文の終わりを意味する。
※ブラウザが推測して解釈してくれるため、省略が可能。ただし、思わぬ動きをする可能性もあるため付けることが推奨される。
- ・文字列は「'」（シングルクォーテーション）または「"」（ダブルクォーテーション）で囲む。
他に「`」（バッククォート）で囲む方法もある。

主な演算子

種類	意味
=	代入
==	等しい
!=	等しくない
<	未満（～より小さい）
<=	以下
>	超過（～より大きい）
>=	以上

種類	意味
+	数値の足し算（加算）
	文字列の結合
-	引き算（減算）
*	かけ算（乗算）
/	割り算（除算）
++	1 増やす
--	1 減らす

種類	意味
+=	加算代入
-=	減算代入
*=	乗算代入
/=	除算代入
,	式の区切り
.	～の
!	ではない

変数の命名ルール

- ・数字のみ、先頭が数字の名前は禁止。（NG 例：3name OK 例：name3）
- ・予約語と呼ばれる JavaScript で別の目的で既に使用することが決まっている名前は使用不可。
（NG 例：if、for、break など）
- ・大文字と小文字が厳密に区別される。（例：Box と box は別物となる）

型について

文字列や数値などのデータの種類のことを「型（Type）」と呼ぶ。

■よく使われる型

型名	種類	例
Number	数値	0、123、-4、1.5
String	文字列	‘こんにちは’、“あいうえお”
Boolean	論理値	true、false

■型の変換

+ 演算子は数値の場合は足し算（加算）、文字列の場合は結合になる。

文字列とその他の型を + 演算した場合、文字列に変換される。

例	結果	結果の型
1 + 2	3	Number 型（数値）
‘1’ + ‘2’	12	String 型（文字列）
1 + ‘2’	12	String 型（文字列）
‘1’ + 2	12	String 型（文字列）

他にも、あらかじめ関数を用いて型変換を行う方法もある。

本教材では、文字列を数値型に型変換する際、Number() ; 関数を使用。

