

# 1. 繰り返し while 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

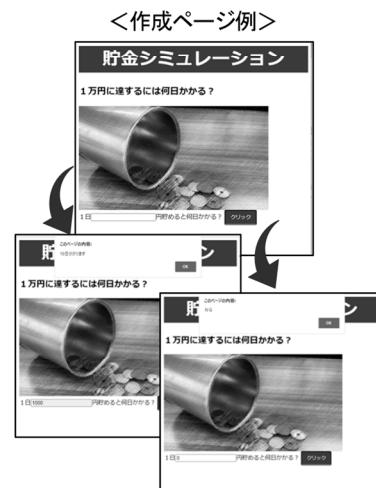
## プログラミング問題 繰り返しwhile 初級

<問題題>

1. 【繰り返し while 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。  
**貯金をしよう**
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。  
**貯金シミュレーション**
  - (2) 【見出し2】ブロックを挿入し、下記の文字を入力しなさい。  
**1万円に達するには何日かかる？**
  - (3) 【画像】ブロックを挿入し、ファイル名を top.jpg にしなさい。
  - (4) 【フォーム】のブロックセット（ナンバー・ボタン）を挿入しなさい。



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] 入力された金額が1万円に達するまでの数（日数）をアラートで表示させる。

[機能]

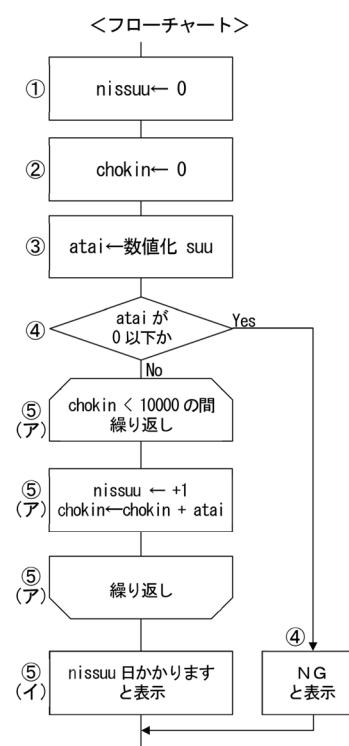
- ・10000を超えるまで指定された値（フォームのテキストに入力された値）の加算を繰り返し、その回数をカウントする。
- ・カウントした回数を表示する。
- ・フォームに0以下の値が入力された場合、「NG」と表示する。

- (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 日数「nissuu」を0と設定する。
- ② 貯金額「chokin」を0と設定する。
- ③ フォームのナンバー（suum）の値を数値にして「atai」に設定する。
- ④ もし「atai」が0以下の場合、「NG」の文字を表示する。
- ⑤ それ以外の場合
  - (ア) 貯金額「chokin」が10000未満の間、下記の処理を繰り返す。  
    日数「nissuu」に1を足す。  
    貯金額「chokin」と「atai」を足して貯金額「chokin」に入れる。
  - (イ) 日数「nissuu」の値と「日かかります」と表示する。

5. プレビュー画面で確認しなさい。

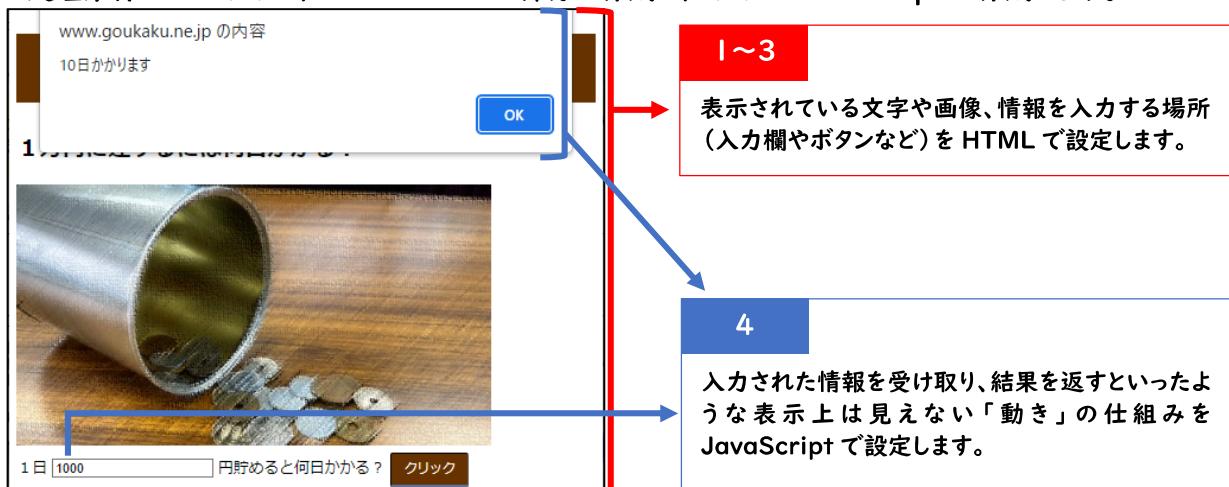


## 教材サイトからアプリを開く

Javloc 教材サイトから[初級プロジェクト]の[繰り返し「while」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。

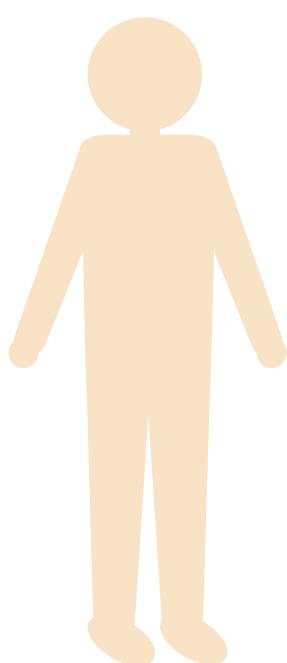


## HTML と CSS と JavaScript の役割 イメージ図

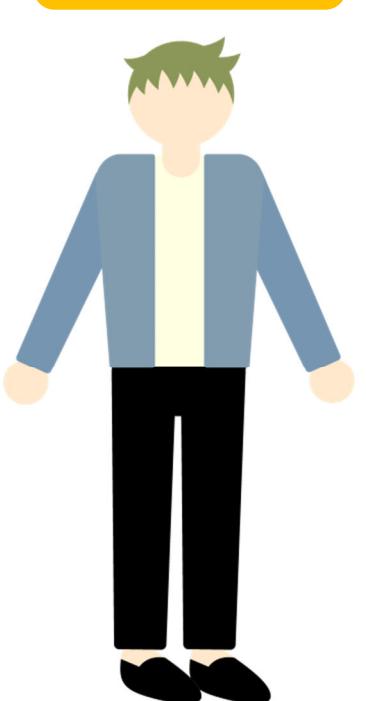
HTML のみ

HTML + CSS

HTML + CSS  
+ JavaScript



構造



デザイン



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。

The screenshot shows the Javloc visual editor interface. On the left, there is a sidebar with a tree view containing categories like ページ (Page), 文字入力 (Text Input), 画像 (Image), etc. A red box highlights the 'ページ' node. In the center, a code editor displays an HTML template with a title block and a link to a stylesheet. A large gray arrow points downwards from the sidebar towards the code editor, indicating the action of dragging the block.

Javloc 繰り返しwhile 初級編

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

ページ

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>
      表示文字
    </title>
    <link rel="stylesheet" href="style.css" type="text/css" >
  </head>
  <body>
  </body>
</html>
```

### 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。

貯金をしよう

<title>ブロックの表示文字ブロックの空欄に「貯金をしよう」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。

The screenshot shows the Javloc visual editor. The 'title' block's content is being edited, with the text '貯金をしよう' highlighted. A red arrow labeled '→入力' points to the input field. To the right, the source code editor shows the updated HTML with the title set to '貯金をしよう'. Another red arrow labeled '→確認' points to the title element in the source code.

Javloc 繰り返しwhile 初級編

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

ページ

```
<!DOCTYPE HTML>
<html>
  <head>
    <meta charset="utf-8">
    <title>
      貯金をしよう
    </title>
    <link rel="stylesheet" href="style.css" type="text/css" >
  </head>
  <body>
  </body>
</html>
```

[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。

The screenshot shows the Javloc visual editor with a preview window open. The 'Preview' button is highlighted with a red box. Below the preview window, a red arrow labeled '↑確認' points to the 'Confirm' button. A red box also highlights the 'Close' button at the bottom of the preview window.

Javloc 繰り返しwhile 初級編

リセット 元に戻す やり直し 解答例 プレビュー ← → C ① about:blank ↑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。

貯金シミュレーション

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「貯金シミュレーション」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>

貯金シミュレーション

1万円に達するには何日かかる？



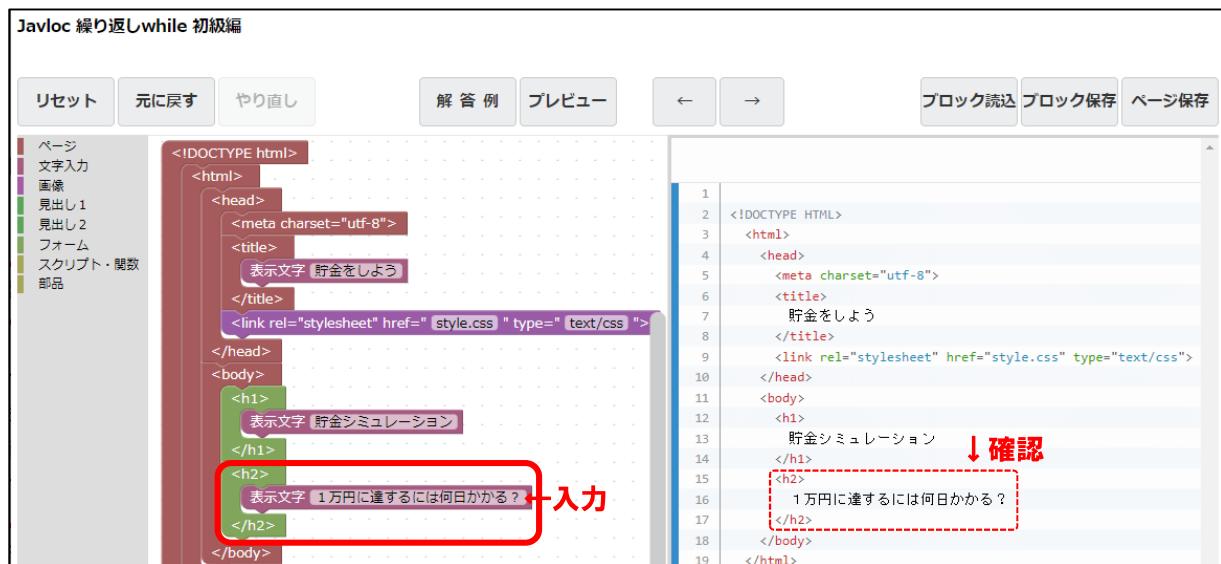
見出し1の背景色が茶色で、文字色が白色になっているのは、CSSで既に設定しているためです。

(2) 【見出し2】ブロックを挿入し、下記の文字を入力しなさい。

1万円に達するには何日かかる？

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、<h1>ブロックの下にドロップ。

表示文字ブロックの空欄に「1万円に達するには何日かかる？」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。



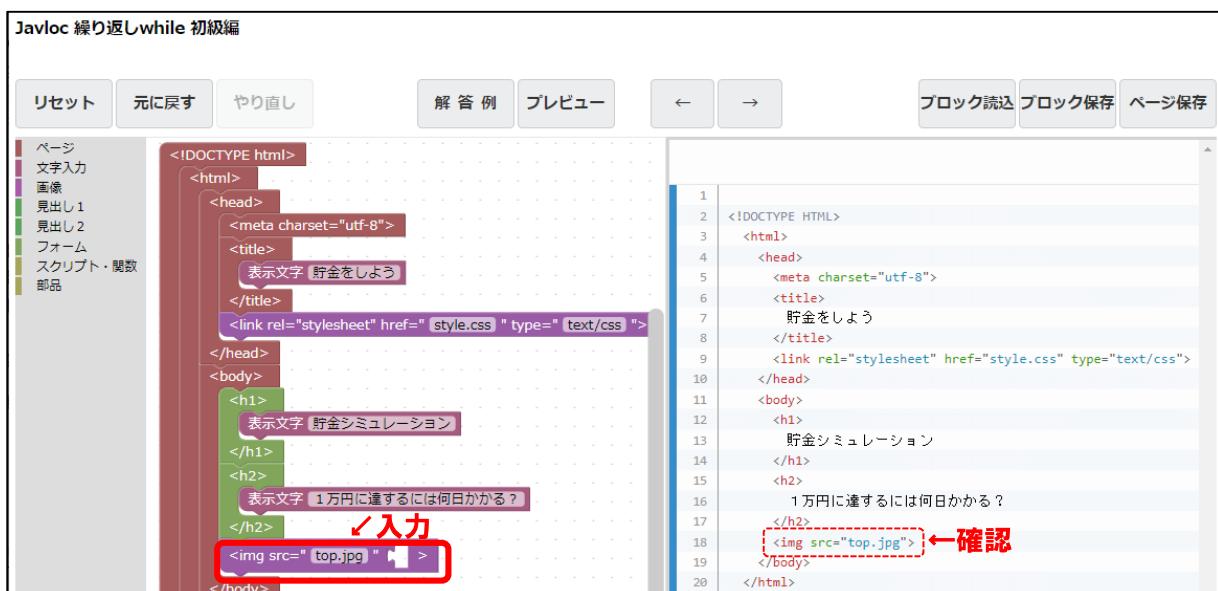
[プレビュー]ボタンをクリックし、正しく設定されているか確認。

[閉じる]ボタンをクリックし、ブロック操作画面へ戻る。

(3) 【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

## ブロックの下にドロップ。 “”の空欄に「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。 ※必ず半角で入力しましょう。



[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

#### (4) 【フォーム】のブロックセット（ナンバー・ボタン）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<img>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 繰り返しwhile 初級編

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

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

<form> E html>  
表示文字 1日  
<input type="number" name=" suu ">  
表示文字 円貯めると何日かかる?  
<input type="button" value=" クリック " onclick=" bt ()">  
</form>  
</title>

Javloc 繰り返しwhile 初級編

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

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

<h1>  
<h2>  
表示文字 1万円に達するには何日かかる?  
</h2>  
  
<form>  
表示文字 1日  
<input type="number" name=" suu ">  
表示文字 円貯めると何日かかる?  
<input type="button" value=" クリック " onclick=" bt ()">  
</form>  
</body>  
</html>

↓ 確認

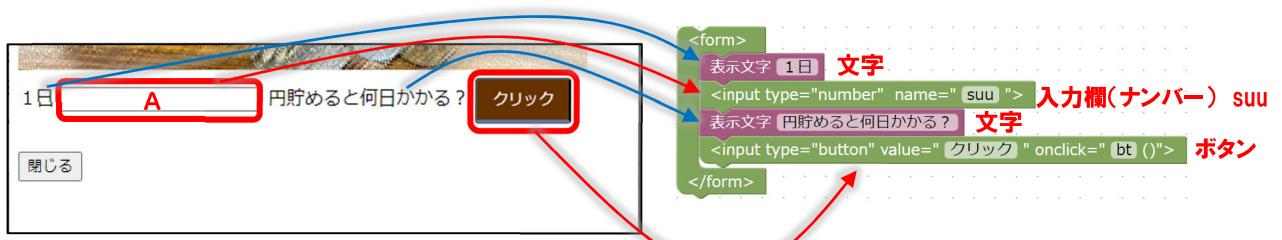
1 !DOCTYPE HTML  
2 <html>  
3 <head>  
4 <meta charset="utf-8">  
5 <title>  
6 貯金をしよう  
7 </title>  
8 <link rel="stylesheet" href="style.css" type="text/css">  
9 </head>  
10 <body>  
11 <h1>  
12 貯金シミュレーション  
13 </h1>  
14 <h2>  
15 1万円に達するには何日かかる?  
16 </h2>  
17   
18 <form>  
19 1日  
20 <input type="number" name=" suu ">  
21 円貯めると何日かかる?  
22 <input type="button" value=" クリック " onclick=" bt ()">  
23 </form>  
24 </body>  
25

[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 繰り返しwhile 初級編

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読込 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、A の入力欄(ナンバー)に数を入力し「クリック」ボタンを押しても、何も表示されません。この後設定する JavaScript によって、「クリック」ボタンを押すと入力された数を使って処理が実行され、結果が表示されるようになります。



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] 入力された金額が 1 万円に達するまでの数（日数）をアラートで表示させる。

[機能]

- 10000 を超えるまで指定された値（フォームのテキストに入力された値）の加算を繰り返し、その回数をカウントする。
- カウントした回数を表示する。
- フォームに 0 以下の値が入力された場合、「NG」と表示する。



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 繰り返しwhile 初級編

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

**貯金シミュレーション**

1万円に達するには何日かかる？

このフォームの部分を操作してみましょう。  
入力欄に値を入力し、「クリック」ボタンを押してみます。  
右図の2パターンを試してみましょう。

1 日  円貯めると何日かかる？ **クリック**

**0 と入力**

www.goukaku.ne.jp の内容  
N G

1

OK

1 日  円貯めると何日かかる？ **クリック**

<入力した値が 0 の場合>

アラートメッセージに「NG」と出力する。

<それ以外の場合>

10000 を超えるまで

入力した値の加算を繰り返して

その回数をカウントし

アラートメッセージにカウントした回数を出力する。

**1000 と入力**

www.goukaku.ne.jp の内容  
10日かかります

1

OK

1 日  1000 円貯めると何日かかる？ **クリック**

## (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 screenshot shows two instances of the Javloc editor. In the top instance, a red box highlights the 'Script' block (a green rectangle containing 'function bt() { }') in the code area. In the bottom instance, the same 'Script' block has been selected and is being moved, with a red dashed box indicating its path from the code area to the 'Link' block's position in the 'head' section of the HTML structure.

<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 □( ) とは、ユーザーが作成する関数を定義するJavaScriptです。□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ます。

(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 日数「nissuu」を0と設定する。

リスト内【部品】をクリック、枠内にある `var nissuu = 0;` ブロックを選択し、ドラッグして `function bt (){~}` ブロックの中にドロップ。



Javloc 繰り返しwhile 初級編

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

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

`var nissuu = 0;`

```
1 <!DOCTYPE HTML>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title> 表示文字 賞金をしよう </title>
6   </head>
7   <body>
8     <script>
9       function bt() {
10         var nissuu = 0;
11       }
12     </script>
13   </body>
14 </html>
```

Javloc 繰り返しwhile 初級編

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

ページ 文字入力 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 賞金をしよう </title>`

`</head>`

`<body>`

`<script>`

`function bt() {`

`var nissuu = 0;` ←確認

`}`

`</script>`

```
1 <!DOCTYPE HTML>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title> 賞金をしよう </title>
6   </head>
7   <body>
8     <script>
9       function bt() {
10         var nissuu = 0; ←確認
11       }
12     </script>
13   </body>
14 </html>
```

「var」というのは、変数を新規に作りますという意味です。

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。

「=」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。

**比較演算子「==」と間違えやすいので注意しましょう。**

ここでは、変数「nissuu」を新規に作成し、そこに「0」という値を入れています。

## ② 賞金額「chokin」を0と設定する。

リスト内[部品]をクリック、枠内にある `var chokin = 0;` 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 繰り返しwhile 初級編

```
リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック説明 ブロック保存 ページ保存  
ページ 文字入力 画像 見出し1 見出し2 フォーム フォーマット・関数 部品  
var nissuu = 0;  
var chokin = 0;  
<meta charset="utf-8">  
var atai = Number(document.forms[0].suu.value);  
表示文字 賞金をしよう  
1 <!DOCTYPE HTML>  
2 <html>  
3 <head>  
4 <meta charset="utf-8">  
5 <title>  
6 賞金をしよう  
7 </title>  
8 <link rel="stylesheet" href="style.css" type="text/css">  
9 <script>  
10 function bt() {  
11     var nissuu = 0;  
12     var chokin = 0;  
13 }  
14 </script>  
15 "
```

Javloc 繰り返しwhile 初級編

```
リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック説明 ブロック保存 ページ保存  
ページ 文字入力 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 賞金をしよう  
</title>  
<link rel="stylesheet" href="style.css" type="text/css">  
<script>  
function bt() {  
    var nissuu = 0;  
    var chokin = 0;  
}  
</script>  
1 <!DOCTYPE HTML>  
2 <html>  
3 <head>  
4 <meta charset="utf-8">  
5 <title>  
6 賞金をしよう  
7 </title>  
8 <link rel="stylesheet" href="style.css" type="text/css">  
9 <script>  
10 function bt() {  
11     var nissuu = 0;  
12     var chokin = 0;  
13 }  
14 </script>  
15 "
```

### ③ フォームのナンバー (suu) の値を数値にして「atai」に設定する。

リスト内[部品]をクリック、枠内にある `var atai = Number(document.forms[0].suu.value);` ブロックを選択し、ドラッグして②で設定したブロックの下にドロップ。

```

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読込 ブロック保存 ページ保存
ページ 文字入力 画像 見出し1 見出し2 フォーム スクリプト・関数 部品 表示文字 賀金をしよう
var nissuu = 0;
var chokin = 0;
var atai = Number(document.forms[0].suu.value);

```

```

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読込 ブロック保存 ページ保存
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title> 表示文字 賀金をしよう </title>
<link rel="stylesheet" href="style.css" type="text/css" >
<script>
function bt() {
    var nissuu = 0;
    var chokin = 0;
    var atai = Number(document.forms[0].suu.value);
}
</script>

```

```

1 <!DOCTYPE HTML>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <title> 賀金をしよう </title>
6 <link rel="stylesheet" href="style.css" type="text/css">
7 <script>
8 function bt() {
9     var nissuu = 0;
10    var chokin = 0;
11    var atai = Number(document.forms[0].suu.value);
12 }
13 </script>
14
15
16

```

↑ 確認

「Number(~)」は、(~) 内にある値を**数値に変換**します。

「`document.forms[0]`」は、今作成している Web ページに最初に登場するフォームのことを指しています。(この問題では、処理条件 3 – (4)で挿入した`<form>`ブロックセットのことです。)

ブロックの中身は

```
var atai = Number(document.forms[0].suu.value);
```

なので、これは何をしているかというと、

フォームの「suu」の値を、数値に変換して、「atai」という箱（変数）に入れておく。

ということになります。

#### ④ もし「atai」が0以下の場合、「NG」の文字を表示する。

リスト内[部品]をクリック、枠内にある `if (atai <= 0) {~}` ブロックを選択し、ドラッグして③で設定したブロックの下にドロップ。

Javloc 繰り返しwhile 初級編

```

var nissuu = 0;
<link rel="stylesheet" href="style.css" type="text/css">
var chokin = 0;
function bt() {
    var atai = Number(document.forms[0].suu.value);
    var nissuu = 0;
    var chokin = 0;
    var atai = Number(document.forms[0].suu.value);
    if(atai <= 0){
        alert('NG');
    }
}

```

Javloc 繰り返しwhile 初級編

```

</title>
<link rel="stylesheet" href="style.css" type="text/css">
<script>
function bt() {
    var nissuu = 0;
    var chokin = 0;
    var atai = Number(document.forms[0].suu.value);
    if(atai <= 0){
        alert('NG');
    }
}

```

ブロックの中身を見てみましょう。

```

if (atai <= 0) {
    alert('NG');
}

```

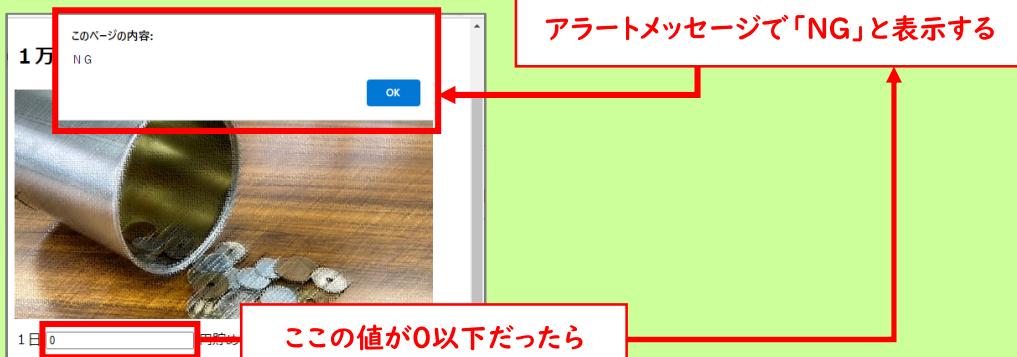
これを日本語で読み下すと

```

もし ( 変数 atai が 0 以下だったら ) {
    アラートメッセージで表示 ('NG')
}

```

<作成ページ例 抜粋>



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。

## ⑤ それ以外の場合

(ア) 貯金額「chokin」が10000未満の間、下記の処理を繰り返す。

日数「nissuu」に1を足す。

貯金額「chokin」と「atai」を足して貯金額「chokin」に入れる。

リスト内[部品]をクリック、枠内にある `else {~}` ブロックセットを選択し、ドラッグして④で設定したブロックの下にドロップ。

Javloc 繰り返しwhile 初級編

The screenshot shows the Javloc interface. On the left, there's a sidebar with categories like ページ, 文字入力, 画像, etc., and a '部品' (Components) section highlighted with a red box. In the center, a Scratch-like stage has a green 'while' loop block selected. This block contains an 'if' condition and a 'repeat' loop. A red box highlights the 'else' part of the 'if' block. On the right, the script editor shows the corresponding JavaScript code:

```

1 <!DOCTYPE HTML>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>
6       貯金をしよう
7     </title>
8     <link rel="stylesheet" href="style.css" type="text/css">
9     <script>
10    function bt() {
11      var nissuu = 0;
12      var chokin = 0;
13      var atai = Number(document.forms[0].suu.value);
14      if (atai <= 0) {
15        alert('N G');
16      }
17    }
18  </script>
19 </head>
20 <body>

```

Javloc 繰り返しwhile 初級編

The screenshot shows the Javloc interface after the 'else' block has been moved. The Scratch stage now has a complete 'while' loop with an 'if' condition and a 'repeat' loop. A dashed red box highlights the 'else' block in the Scratch script. On the right, the script editor shows the updated JavaScript code:

```

5 <meta charset="utf-8">
6 <title>
7 貯金をしよう
8 </title>
9 <link rel="stylesheet" href="style.css" type="text/css">
10 <script>
11 function bt() {
12   var nissuu = 0;
13   var chokin = 0;
14   var atai = Number(document.forms[0].suu.value);
15   if (atai <= 0) {
16     alert('N G');
17   }
18   else {
19     while (chokin < 10000) {
20       nissuu = nissuu + 1;
21       chokin = chokin + atai;
22     }
23   }
24 } </script>
25 </head>
26 <body>

```

A red arrow points to the 'else' block in the Scratch script with the text '確認' (Confirm).

(イ) 日数「nissuu」の値と「日かかります」の文字を表示する。

リスト内[部品]をクリック、枠内にある `alert(nissuu + '日かかります');` ブロックを選択し、ドラッグ。

Javloc 繰り返しwhile 初級編

The screenshot shows the Javloc interface. The '部品' (Components) section is highlighted with a red box. On the Scratch stage, there's a green 'repeat' loop block with a 'say' block inside. A red box highlights the 'say' block. On the right, the script editor shows the corresponding JavaScript code:

```

15 if (atai <= 0) {
16   alert('N G');
17 }
18 else {
19   while (chokin < 10000) {
20     nissuu = nissuu + 1;
21     chokin = chokin + atai;
22   }
23 }

```

(ア) で設定した<else>ブロックの中、<while>ブロックの下にドロップ。

Javloc 繰り返しwhile 初級編

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

```
function bt() {
    var nissuu = 0;
    var chokin = 0;
    var atai = Number(document.forms[0].suu.value);
    if(atai <= 0){
        alert('N G');
    } else {
        while(chokin < 10000){
            nissuu = nissuu + 1;
            chokin = chokin + atai;
        }
        alert(nissuu + '日かかります');
    }
}
```

12 var nissuu = 0;  
13 var chokin = 0;  
14 var atai = Number(document.forms[0].suu.value);  
15 if (atai <= 0) {  
16 alert('N G');  
17 }  
18 else {  
19 while (chokin < 10000) {  
20 nissuu = nissuu + 1;  
21 chokin = chokin + atai;  
22 }  
23 alert(nissuu + '日かかります');  
24 }  
25 }  
26 </script>  
27 </head>

←確認

ブロックの中身を見てみましょう。

```
else {
    while (chokin < 10000) {
        nissuu = nissuu + 1;
        chokin = chokin + atai;
    }
    alert (nissuu + '日かかります');
}
```

これを日本語で読み下すと

それ以外の場合 {  
繰り返す(変数 chokin が 10000 より小さい間) {  
 変数 nissuu に 1 を足す ← 日数をカウントする箱  
 変数 chokin に変数 atai を足す ← 貯金箱  
}  
アラートメッセージで表示する(変数 nissuu の値と「日かかります」の文字)  
}

今回は if と else を分解して説明していますが、これはひとかたまりの条件分岐です。

もし 0 以下が入力されたら

→ 「N G」とアラートメッセージを表示する。

それ以外だったら

→貯金箱の中身が 10000 円より小さい間

- ・日数を数える箱に 1 日足す
- ・貯金箱に 1 日分のお金を入れる

を繰り返し、10000 円貯まるまでに何日かかるか (変数 nissuu の中身) を求める。

そして、その変数 nissuu の値と「日かかります」をアラートメッセージで表示する。



問題文のフローチャートと一緒に確認してみましょう。

## 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. 繰り返し while 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 繰り返しwhile 中級

<問題題>

1. 【繰り返し while 中級】を開いて解答する。
2.  内・網掛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。  
**貯金をしよう**
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。  
**貯金シミュレーション**
  - (2) 【見出し2】ブロックを挿入し、下記の文字を入力しなさい。  
**1万円に達するには何日かかる？**
  - (3) 【画像】ブロックを挿入し、ファイル名を top.jpg にしなさい。
  - (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 【文字入力】ブロックを挿入し、下記の文字を入力しなさい。  
**1日**
    - ② 【フォーム】からナンバーのブロックを選択・挿入し、name は suu。
    - 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。  
**円貯めると何日かかる？**
    - ④ 【フォーム】からボタンのブロックを選択・挿入しなさい。value はクリック、onclick は関数名 bt。

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] 入力された金額が1万円に達するまでの数（日数）をアラートで表示させる。

[機能]

- ・10000 を超えるまで指定された値（フォームのテキストに入力された値）の加算を繰り返し、その回数をカウントする。
- ・カウントした回数を表示する。
- ・フォームのナンバーに0以下の値が入力された場合、「NG」と表示する。

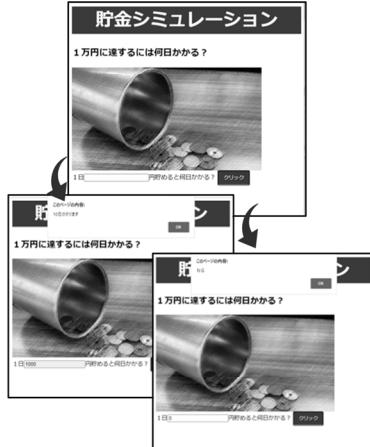
- (1) 【スクリプト・関数】ブロックを挿入し、3-(4)-④で指定した関数名を入力しなさい。
- (2) 【部品】内から正しいブロックを選択し、(1)の関数ブロックの中に下記処理手順どおりに組み立てなさい。

■処理手順

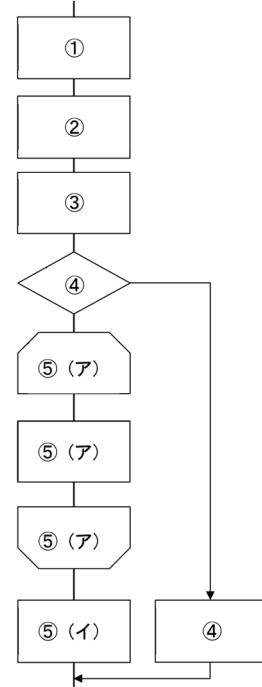
- ① 変数 nissuu を宣言し、0 を代入
- ② 変数 chokin を宣言し、0 を代入
- ③ 変数 atai を宣言し、フォームのナンバー (suu) の値を数値に変換し代入
- ④ 変数 atai が0以下の場合、「NG」と表示
- ⑤ それ以外の場合
  - (ア) chokin が10000未満の間、下記の処理を繰り返す
    - nissuu に1を加算
    - chokin に atai を加算し、chokin に代入
  - (イ) 「〇日かかります」と表示 ※〇はnissuu の値

5. プレビュー画面で確認しなさい。

<作成ページ例>



<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[繰り返し「while」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。

This screenshot shows the Javloc editor interface for the 'Looping with while' intermediate project. On the left, there's a sidebar with a 'Page' category selected. The main area shows an HTML code structure with various blocks like 'DOCTYPE html', 'head', 'meta charset="utf-8"', 'title', 'body', and 'link'. A large red arrow points from the sidebar towards the main editing area, indicating the action of dragging the block.

This screenshot shows the Javloc editor interface after the 'Page' block has been dropped into the main editing area. The sidebar still shows the 'Page' category, but the main area now displays the full HTML code with the block integrated. A large red arrow points from the sidebar towards the main editing area, indicating the completed action.

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。

貯金をしよう

<title>ブロックの表示文字ブロックの空欄に「貯金をしよう」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。

Javloc 繰り返しwhile 中級編

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

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

表示文字 貯金をしよう ←入力

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>貯金をしよう</title>
    <link rel="stylesheet" href="style.css" type="text/css">
  </head>
  <body>
  </body>
</html>
```

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

表示文字 貯金をしよう ←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。

貯金シミュレーション

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「貯金シミュレーション」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。

Javloc 繰り返しwhile 中級編

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

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

表示文字 貯金をしよう

見出し1

表示文字 貯金シミュレーション

```
<!DOCTYPE HTML>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <h1>貯金をしよう</h1>
  </body>
</html>
```

1 2 3

Javloc 繰り返しwhile 中級編

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

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

表示文字 貯金をしよう

表示文字 貯金シミュレーション ←入力

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>貯金をしよう</title>
    <link rel="stylesheet" href="style.css" type="text/css">
  </head>
  <body>
    <h1>貯金シミュレーション</h1>
  </body>
</html>
```

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

表示文字 貯金シミュレーション ←確認

(2) 【見出し2】ブロックを挿入し、下記の文字を入力しなさい。

1万円に達するには何日かかる？

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<h1>ブロックの下にドロップ。

表示文字ブロックの空欄に「1万円に達するには何日かかる？」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 繰り返しwhile 中級編

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

ページ 文字入力 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 1万円に達するには何日かかる? </h2>

ソース表示エリア

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8">
<title>[title]</title>
<link rel="stylesheet" href="style.css" type="text/css">
</head>
<body>
<h1> 表示文字 賞金をしよう </h1>
<h2> 表示文字 1万円に達するには何日かかる? </h2>
</body>
</html>
```

Javloc 繰り返しwhile 中級編

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

ページ 文字入力 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 1万円に達するには何日かかる? </h2>

ソース表示エリア

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8">
<title> 賞金をしよう </title>
<link rel="stylesheet" href="style.css" type="text/css">
</head>
<body>
<h1> 賞金シミュレーション </h1>
<h2> 表示文字 1万円に達するには何日かかる? </h2>
</body>
</html>
```

(3) 【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<img>ブロックをドラッグし<h2>ブロックの下にドロップ。 “”の空欄に「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。※必ず半角で入力しましょう。

Javloc 繰り返しwhile 中級編

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

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

<img src="" >

ソース表示エリア

```
<!DOCTYPE HTML>
```

スクリプト・関数 部品

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

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

表示文字 賞金をしよう

表示文字 賞金シミュレーション

表示文字 1万円に達するには何日かかる?



ソース表示エリア

```
<head>
<meta charset="utf-8">
<title> 賞金をしよう </title>
<link rel="stylesheet" href="style.css" type="text/css">
</head>
<body>
<h1> 賞金シミュレーション </h1>
<h2> 表示文字 1万円に達するには何日かかる? </h2>
 ←確認
</body>
</html>
```

(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。

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<img>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 繰り返しwhile 中級編

```

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読込 ブロック保存 ページ保存
ページ 文字入力 画像 見出し1 見出し2 フォーム スクリプト・関数 部品
<form>
</form>
<input type="number" name=" " />

```

Javloc 繰り返しwhile 中級編

```

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読込 ブロック保存 ページ保存
ページ 文字入力 画像 見出し1 見出し2 フォーム スクリプト・関数 部品
表示文字 賀金をしよう
</title>
<link rel="stylesheet" href="style.css" type="text/css">
</head>
<body>
<h1> 表示文字 賀金シミュレーション
</h1>
<h2> 表示文字 1万円に達するには何日かかる？
</h2>

<form>
</form>
</body>
</html>

```

<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」になるというイメージを持っていれば大丈夫です。

① 【文字入力】ブロックを挿入し、下記の文字を入力しなさい。

1日

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ロップし、<form>ブロックの中にドロップ。空欄に「枚」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。

Javloc 繰り返しwhile 中級編



Javloc 繰り返しwhile 中級編

This screenshot shows the completed code in the Javloc editor. The code is as follows:

```
<!DOCTYPE HTML>
<html>
  <head>
    <meta charset="utf-8">
    <title>貯金をしよう</title>
    <link rel="stylesheet" href="style.css" type="text/css">
  </head>
  <body>
    <h1>貯金シミュレーション</h1>
    <h2>1万円に達するには何日かかる？</h2>
    
    <form>
      <input type="text" value="1日" /> ← 入力
    </form>
  </body>
</html>
```

The 'Text Input' block from the previous screenshot has been successfully inserted into the 'Form' block. The value '1日' is now displayed in the input field. The code editor also highlights the '1日' value with a red box and the text '←確認' (← confirmation).



## ② 【フォーム】からナンバーのブロックを選択・挿入しなさい。name は suu。

リスト内【フォーム】をクリック、枠内にある`<input type="number" ~>`ブロックをドラッグし、表示文字ブロックの下にドロップ。空欄に「suum」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。**※必ず半角で入力しましょう。**

Javloc 繰り返しwhile 中級編

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

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

表示文字 貯金をしよう

`<form> 表示文字 貯金をしよう`

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

`<head>`

`<body>`

`<h1> 表示文字 貯金シミュレーション`

`</h1>`

`<h2> 表示文字 1万円に達するには何日かかる？`

`</h2>`

``

`<form> 表示文字 1日`

`<input type="number" name="suu" >`

`</form>`

`</body>`

`</html>`

1 <!DOCTYPE HTML>  
2 <html>  
3 <head>  
4 <meta charset="utf-8">  
5 <title>  
6 貯金をしよう  
7 </title>  
8 <link rel="stylesheet" href="style.css" type="text/css" >  
9 </head>  
10 <body>  
11 <h1>  
12 貯金シミュレーション  
13 </h1>  
14 <h2>  
15 1万円に達するには何日かかる？  
16 </h2>  
17   
18 <form>  
19 1日  
20 <input type="number" name="suu" > ←確認  
21 </form>

Javloc 繰り返しwhile 中級編

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

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

表示文字 貯金をしよう

`</title>`

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

`</head>`

`<body>`

`<h1> 表示文字 貯金シミュレーション`

`</h1>`

`<h2> 表示文字 1万円に達するには何日かかる？`

`</h2>`

``

`<form> 表示文字 1日`

`<input type="number" name="suu" >` ↓入力

`</form>`

`</body>`

`</html>`

1 <!DOCTYPE HTML>  
2 <html>  
3 <head>  
4 <meta charset="utf-8">  
5 <title>  
6 貯金をしよう  
7 </title>  
8 <link rel="stylesheet" href="style.css" type="text/css" >  
9 </head>  
10 <body>  
11 <h1>  
12 貯金シミュレーション  
13 </h1>  
14 <h2>  
15 1万円に達するには何日かかる？  
16 </h2>  
17   
18 <form>  
19 1日  
20 <input type="number" name="suu" > ←確認  
21 </form>

## 貯金シミュレーション

1万円に達するには何日かかる？



1日

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。  
[円貯めると何日かかる？]

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し、  
 <input type="number" ~> ブロックの下にドロップ。空欄に「円貯めると何日かかる？」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。



The screenshot shows the Javloc editor interface. On the left, there's a sidebar with categories like ページ, 文字入力, 画像, 見出し1, 見出し2, フォーム, スクリプト・関数, 部品. The main area shows an HTML structure with several '表示文字' blocks (like '貯金をしよう', '貯金シミュレーション', '1円に達するには何日かかる?') and one '文字入力' block (with '1円' typed into it). To the right is the code editor with the same structure. A red box highlights the input field in the code editor, and another red box highlights the question '円貯めると何日かかる?' in the preview area.



- ④ 【フォーム】からボタンのブロックを選択・挿入しなさい。value はクリック、onclick は関数名 bt。

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し、表示文字ブロックの下にドロップ。value の“”空欄に「クリック」、onclick の“”空欄に「bt」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。

※bt は必ず半角で入力しましょう。

Javloc 繰り返しwhile 中級編

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

ページ 文字入力 画像 見出し1 見出し2 フォーム ベクターフォント・実装 部品

```

1 <!DOCTYPE HTML>
2 <html>
3   <head>
4     <meta charset="utf-8">
5   <title>
6     貯金をしよう
7   </title>
8 </html>

```

Javloc 繰り返しwhile 中級編

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

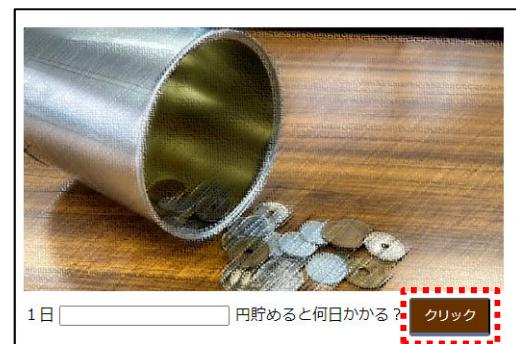
ページ 文字入力 画像 見出し1 見出し2 フォーム スクリプト・関数 部品

```

1 <!DOCTYPE HTML>
2 <html>
3   <head>
4     <meta charset="utf-8">
5   <title>
6     貯金をしよう
7   </title>
8   <link rel="stylesheet" href="style.css" type="text/css">
9 </head>
10 <body>
11   <h1> 貯金シミュレーション </h1>
12   <h2> 1万円に達するには何日かかる？ </h2>
13   
14   <form>
15     表示文字 1日
16     <input type="number" name="suu" value="1" style="width: 100px; margin-bottom: 10px;"/>
17     表示文字 円貯めると何日かかる？
18     <input type="button" value="クリック" onclick="bt()" style="width: 100px; background-color: #007bff; color: white; border: none; padding: 5px; font-weight: bold;"/>
19   </form>
20 </body>
21 </html>

```

↑ 入力 ↑ ↓ 確認



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] 入力された金額が 1 万円に達するまでの数（日数）をアラートで表示させる。

[機能]

- 10000 を超えるまで指定された値（フォームのテキストに入力された値）の加算を繰り返し、その回数をカウントする。
- カウントした回数を表示する。
- フォームに 0 以下の値が入力された場合、「NG」と表示する。



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 繰り返しwhile 初級編

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

**貯金シミュレーション**

1万円に達するには何日かかる？

このフォームの部分を操作してみましょう。  
入力欄に値を入力し、「クリック」ボタンを押してみます。  
右図の2パターンを試してみましょう。

1 日  円貯めると何日かかる？ **クリック**

**0 と入力**

www.goukaku.ne.jp の内容  
N G

1

OK

1 日  円貯めると何日かかる？ **クリック**

<入力した値が 0 の場合>

アラートメッセージに「NG」と出力する。

<それ以外の場合>

10000 を超えるまで

入力した値の加算を繰り返して

その回数をカウントし

アラートメッセージにカウントした回数を出力する。

**1000 と入力**

www.goukaku.ne.jp の内容  
10日かかります

1

OK

1 日  1000 円貯めると何日かかる？ **クリック**

(1) 【スクリプト・関数】ブロックを挿入し、3-(4)-④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。

Javloc 繰り返しwhile 中級編

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

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

<script> html> function () { <head> } </script> e> 表示文字 賞金をしよう

```

1 <!DOCTYPE HTML>
2 <html>
3   <head>
4     <meta charset="utf-8">
5   <title> 賞金をしよう
6   <link rel="stylesheet" href="style.css" type="text/css" >
7   <script>
8     function () {
9       //処理
10    }
11   </script>
12 </head>
13 <body>
14
15

```

Javloc 繰り返しwhile 中級編

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

ページ 文字入力 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 賞金をしよう </title> <link rel="stylesheet" href="style.css" type="text/css" > <script> function () { </script> </head> <body>

```

1 <!DOCTYPE HTML>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title> 賞金をしよう
6     <link rel="stylesheet" href="style.css" type="text/css">
7     <script>
8       function () {
9         //処理
10      }
11   </script>
12 </head>
13 <body>
14
15

```

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

関数名「bt」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。

Javloc 繰り返しwhile 中級編

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

ページ 文字入力 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 賞金をしよう </title> <link rel="stylesheet" href="style.css" type="text/css" > <script> function bt() { </script> </head> <body>

```

1 <!DOCTYPE HTML>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title> 賞金をしよう
6     <link rel="stylesheet" href="style.css" type="text/css">
7     <script>
8       function bt() {
9         //処理
10      }
11   </script>
12 </head>
13 <body>
14
15

```

**function □()**とは、関数を定義するJavaScriptです。□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 bt (){～}」となるように入力します。

これで、クリックボタンが押されたら onclick、関数(bt)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 nissuu 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 nissuu = ;` ブロックを選択し、ドラッグして `function bt (){~}` 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 繰り返しwhile 中級編

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

if(atai <= 0){  
 alert('N G');  
}  
else {  
 while(chokin < 0){  
 nissuu = nissuu + 1;  
 chokin = chokin + 1;  
 }  
 else {  
 while(chokin <= 0){  
 nissuu = nissuu + 1;  
 chokin = chokin + 1;  
 }  
 }  
 <h1> 表示文字 資金シミュレーション  
 var atai = document.forms[0].suu.value;  
 var atai = Number(document.forms[0].suu.value);  
  
 var nissuu = 0;

1 <!DOCTYPE HTML>  
2 <html>  
3 <head>  
4 <meta charset="utf-8">  
5 <title> 資金をしよう  
6 </title>  
7 <link rel="stylesheet" href="style.css" type="text/css">  
8 <script>  
9 function bt() {  
10 //  
11 }  
12 </script>  
13 </head>  
14 <body>  
15 <h1> 資金シミュレーション  
16 <h2> 1万円に達するには何日かかる？  
17 </h2>  
18   
19 <form>  
20 1日  
21 <input type="number" name="suu">  
22 円貯めると何日かかる？  
23 <input type="button" value="クリック" onclick="bt()">  
24 </form>

空欄に「0」と入力。※必ず半角で入力しましょう。

Javloc 繰り返しwhile 中級編

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

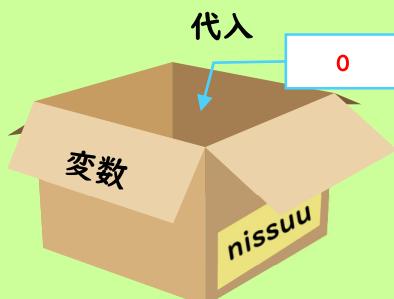
<!DOCTYPE html>  
<html>  
 <head>  
 <meta charset="utf-8">  
 <title> 表示文字 資金をしよう  
 </title>  
 <link rel="stylesheet" href="style.css" type="text/css">  
 <script>  
 function bt() {  
 var nissuu = 0;  
 }  
 </script>

1 <!DOCTYPE HTML>  
2 <html>  
3 <head>  
4 <meta charset="utf-8">  
5 <title> 資金をしよう  
6 </title>  
7 <link rel="stylesheet" href="style.css" type="text/css">  
8 <script>  
9 function bt(){  
10 var nissuu = 0;  
11 }  
12 </script>  
13 </head>

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 chokin 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 chokin = ; ブロックを選択し、ドラッグして var nissuu = 0; ブロックの下にドロップ。

**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 繰り返しwhile 中級編

```

1 <!DOCTYPE HTML>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>
6       賃金をしよう
7     </title>
8     <link rel="stylesheet" href="style.css" type="text/css">
9     <script>
10    function bt() {
11      var nissuu = 0;
12    }
13  </script>
14 </head>
15 <body>
16   <h1>
17     賃金シミュレーション
18   </h1>
19   <h2>1万円に達するには何日かかる？</h2>
20   
21   <form>
22     1日
23     <input type="number" name="suu">
24     円貯めると何日かかる？
25     <input type="button" value="クリック" onclick="bt()">
26   </form>
27 </body>
28 </html>
29

```

空欄に「0」と入力。※必ず半角で入力しましょう。

Javloc 繰り返しwhile 中級編

```

1 <!DOCTYPE HTML>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>
6       賃金をしよう
7     </title>
8     <link rel="stylesheet" href="style.css" type="text/css">
9     <script>
10    function bt() {
11      var nissuu = 0;
12      var chokin = 0; ←確認
13    }
14  </script>
15 </head>
16 <body>
17 </body>
18 </html>

```

### ③ 変数 atai を宣言し、フォームのナンバー (suu) の値を数値に変換し代入

リスト内[部品]をクリック、枠内にある `var atai = Number(~);` ブロックを選択し、ドラッグして `var chokin = 0;` ブロックの下にドロップ。

**ブロック選択のヒント** 問題文に「数値に変換」とあったら「Number」を含むブロックを選びましょう。

Javloc 繰り返しwhile 中級編

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

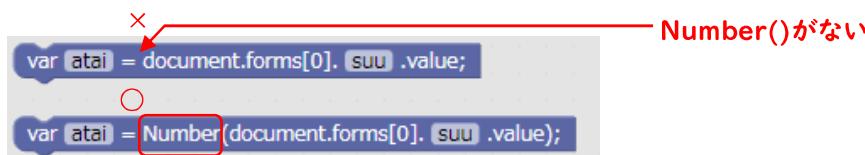
ページ 文字入力 画像 見出し1 見出し2 フォーム スクリプト・関数 部品

```

1 <!DOCTYPE HTML>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>
6       貯金をしよう
7     </title>
8     <link rel="stylesheet" href="style.css" type="text/css">
9     <script>
10    function bt () {
11      var nissuu = 0;
12      var chokin = 0;
13    }
14  </script>
15 </head>
16 <body>
17   <h1>
18     貯金シミュレーション
19   </h1>
20   <h2>
21     1万円に達するには何日かかる？
22   </h2>
23   
24

```

[部品]の中には、誤ったブロックが混ざっています。



```

1 <!DOCTYPE HTML>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>
6       貯金をしよう
7     </title>
8     <link rel="stylesheet" href="style.css" type="text/css">
9     <script>
10    function bt () {
11      var nissuu = 0;
12      var chokin = 0;
13      var atai = Number(document.forms[0]. suu .value);
14    }
15  </script>
16 </head>
17 <body>
18

```

「Number(~)」は、(~) 内にある値を数値に変換します。

「`document.forms[0]`」は、今作成している Web ページに最初に登場するフォームのことを指しています。(この問題では、処理条件 3 – (4)で作成した`<form>`のことです。)

`.suu.value` は、右図で示されている `suu` という名前の

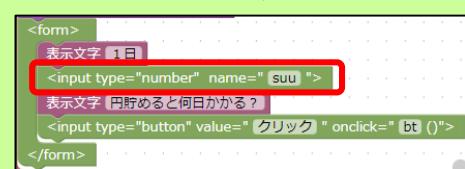
付いた数値入力欄 (number) に入力された値を指します。

フォームで入力された値は文字列として扱われるため、

数値に変換していないと、この値を用いて計算をするこ

とができません。そのため、`Number(~)`を用いて数値に変換する必要があるのです。

このように、型の処理をしていないと、正しく動作しません。



#### ④ 変数 atai が 0 以下の場合、「NG」と表示

リスト内[部品]をクリック、枠内にある `if (atai <= 0) {~}` ブロックを選択し、ドラッグして  
`var atai = Number(~);` ブロックの下にドロップ。

**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。

Javloc 繰り返しwhile 中級編



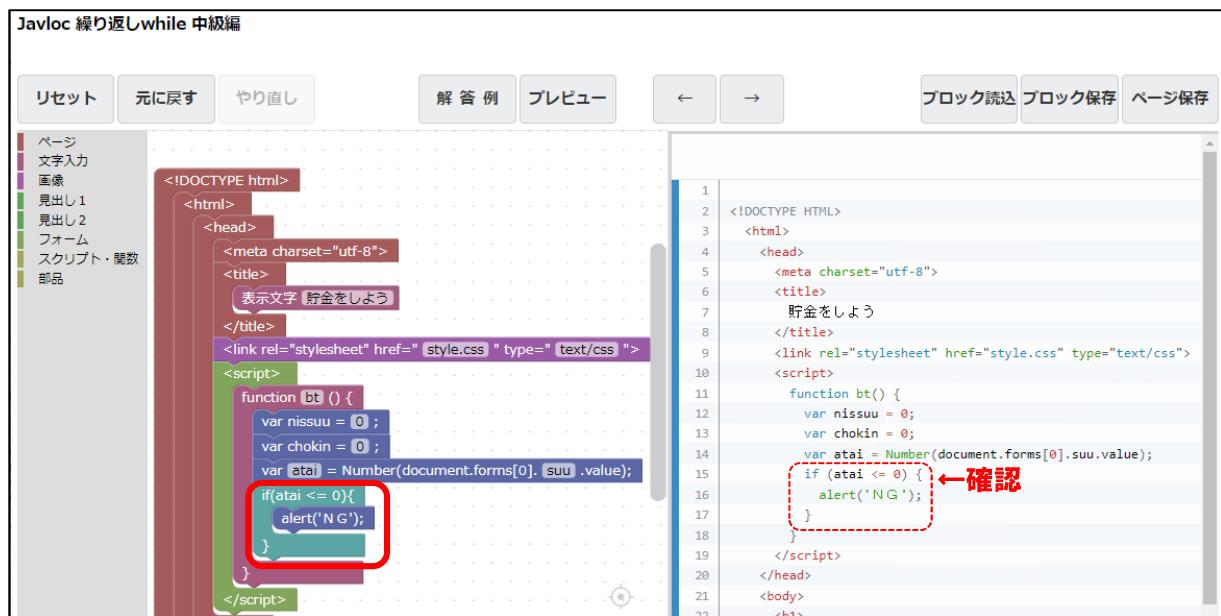
The screenshot shows the Javloc interface. On the left, there's a sidebar with categories like ページ, 文字入力, 画像, 見出し1, 見出し2, フォーム, スクリプト・関数, and 部品. The '部品' category is highlighted with a red box. In the center, a block-based script is being edited. A blue 'if' block with the condition `if(atai <= 0){` and the body `alert('NG');` is selected and being moved. To its right, the corresponding traditional programming code is shown:

```

1  <!DOCTYPE HTML>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>
6  表示文字 貯金をしよう
7  </title>

```

Javloc 繰り返しwhile 中級編



The screenshot shows the Javloc interface again. The sidebar now includes 'スクリプト・関数'. The main area shows the script with the 'if' block integrated. A red box highlights the condition `if(atai <= 0){` in the code. A red dashed box highlights the same condition in the visual editor. A red arrow points from the visual editor to the code editor with the text '確認' (confirmation).

```

1  <!DOCTYPE HTML>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <title>
6  貯金をしよう
7  </title>
8  <link rel="stylesheet" href="style.css" type="text/css">
9
10 <script>
11   function bt () {
12     var nissuu = 0;
13     var chokin = 0;
14     var atai = Number(document.forms[0].suu.value);
15     if(atai <= 0){
16       alert('NG');
17     }
18   }
19 </script>
20 </head>
21 <body>
22 <h1>

```

## ⑤ それ以外の場合

(ア) chokin が 10000 未満の間、下記の処理を繰り返す

nissuu に 1 を加算

chokin に atai を加算し、chokin に代入

リスト内[部品]をクリック、枠内にある下図のブロックセットを選択。

Javloc 繰り返しwhile 中級編

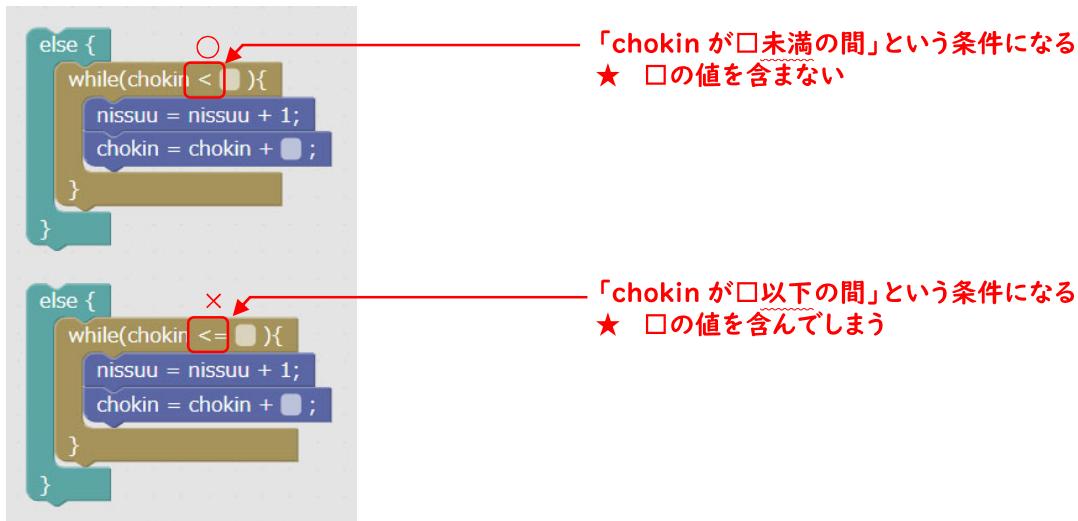
リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読込 ブロック保存 ページ保存

ページ 文字入力 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> 貯金をしよう </title>
</head>
<link rel="stylesheet" href="style.css" type="text/css">
<script>
function bt() {
 var nissuu = 0;
 var chokin = 0;
 var atai = Number(document.forms[0].suu.value);
 if (atai <= 0) {
 alert('NG');
 } else {
 while(chokin < 10000) {
 nissuu = nissuu + 1;
 chokin = chokin + atai;
 }
 }
}
</script>
</body>`

[部品]の中には、誤ったブロックが混ざっています。

問題文では、「chokin が 10000 未満の間、下記の処理を繰り返す」とあるので



ドラッグして if (atai <= 0) { } ブロックの下にドロップ。

while (chokin < □); の空欄には、「10000」と入力。chokin = chokin + □; の空欄には、「atai」と入力。

※必ず半角で入力しましょう。

Javloc 繰り返しwhile 中級編

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

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

`var nissuu = 0;
var chokin = 0;
var atai = Number(document.forms[0].suu.value);
if (atai <= 0) {
 alert('NG');
} else {
 while(chokin < 10000) {
 nissuu = nissuu + 1;
 chokin = chokin + atai;
 }
}
</script>
</head>`

入力 確認

## (イ) 「〇日かかります」と表示 ※〇はnissuu の値

リスト内[部品]をクリック、枠内にある `alert(nissuu + '日かかります');` ブロックを選択し、ドラッグ。

```

function bt() {
    var nissuu = 0;
    var chokin = 0;
    var atai = Number(document.forms[0].suu.value);
    if (atai <= 0) {
        alert('NG');
    } else {
        while (chokin < 10000) {
            nissuu = nissuu + 1;
            chokin = chokin + atai;
        }
        alert(nissuu + '日かかります');
    }
}

```

[部品]の中には、誤ったブロックが混ざっています。

問題文では、「〇日かかります」と表示 ※〇はnissuu の値』とあるので、



(ア) で設定した<else>ブロックの中、<while>ブロックの下にドロップ。

```

function bt() {
    var nissuu = 0;
    var chokin = 0;
    var atai = Number(document.forms[0].suu.value);
    if (atai <= 0) {
        alert('NG');
    } else {
        while (chokin < 10000) {
            nissuu = nissuu + 1;
            chokin = chokin + atai;
        }
        alert(nissuu + '日かかります'); ←確認
    }
}

```

条件分岐のブロックの中身を日本語に読み下して見てみましょう。

```

if (atai <= 0) {
    alert('NG');
}
else {
    while (chokin < 10000) {
        nissuu = nissuu + 1;
        chokin = chokin + atai;
    }
    alert (nissuu + '日かかります');
}

```

もし (変数 atai が 0 以下だったら) {  
    アラートメッセージで表示 ('NG')  
}  
それ以外の場合 {  
    繰り返す (変数 chokin が 10000 より小さい間) {  
        変数 nissuu に 1 を足す ← 日数をカウントする箱  
        変数 chokin に変数 atai を足す ← 賞金箱  
    }  
    アラートメッセージで表示 ('nissuu 日かかります')  
}

今回は、「目標額に到達するまで何回貯金を繰り返したか」を知りたいため、while 文を使用しています。

WEB ページ上では、結果しか確認することができません。

そのため次ページでは、具体的に while 文ではどのような処理を繰り返して結果を導き出したのかを見ていきます。

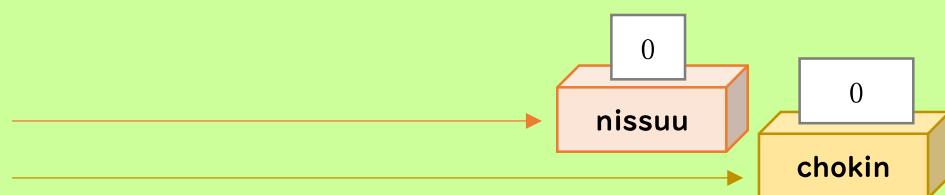
例)

数値入力欄 (suu) に「5000」と入力した場合を考えてみましょう

最初の状態

nissuu = 0

chokin = 0

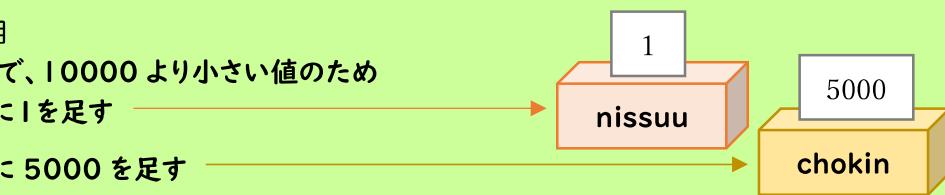


繰り返し処理1回目

①chokin が「0」で、10000 より小さい値のため

nissuu(0)に1を足す

chokin(0)に5000を足す



繰り返し処理2回目

②chokin が「5000」で、10000 より小さい値のため

nissuu(1)に1を足す

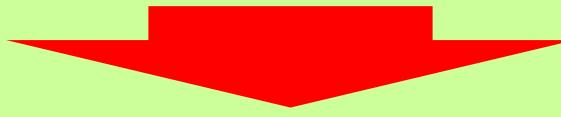
chokin(5000)に5000を足す



繰り返し処理3回目

③chokin が「10000」で、10000 より小さい値でないため

繰り返し処理を終了し、次の処理(アラートメッセージ表示)へ移行する



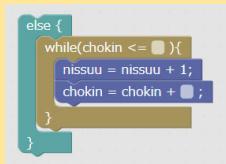
nissuu には「2」が入っているため  
アラートメッセージには「2日かかります」と表示されます



Try

次の設定も終え、正しく動作することを確認したら、

試しに、誤っている方のブロック (chokin<=10000 となる条件) に置き換えてみましょう。



すると、繰り返し処理3回目が行われ、結果は「3日かかります」となってしまいます。

求めたいのは1万円貯めるために必要な日数をカウントすることなので、chokin<10000 と設定する必要があることが体感できるでしょう。

## 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. 繰り返し while 上級 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 繰り返しwhile 上級

<問題>

- <body>のプロックセトを挿入しなさい。
- <title>プロックの中に入力フィールドを入力しなさい。ページタイトルは下記のとおりとする。  
貯金しよう
- <body>プロックの中に下記の(1)から(4)にプロックを挿入し、処理をしなさい。
  - 【見出し】1 プロックを挿入し、下記の文字を入力しなさい。  
貯金シミュレーション
  - 【見出し】2 プロックを挿入し、下記の文字を入力しなさい。  
1万円に達するには何日かかる？
  - 【画像】 ブロックを挿入し、ファイル名を top.jpg にしなさい。
  - 【フォーム】から【文書入力】プロックを選択・挿入し、その中に下記のプロックを挿入し、処理をしなさい。
    - 【文書入力】プロックを挿入し、下記の文字を入力しなさい。  
1日
    - 【フォーム】からナナーのプロックを選択・挿入し、下記の文字を入力しなさい。  
貯める
    - 【文書入力】プロックを挿入し、下記の文字を入力しなさい。  
1日かかる
    - 【フォーム】からボタンのプロックを選択・挿入しなさい、value はクリック、onclick は関数名 bt。

4. <head>プロックの中の<link>~</link>の下に【JavaScript】から必要なプロックを選択し<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]**  
1日あたりの貯金額を入力し、それが1万円に達するまでの日数をカウントし、表示するページ。

**[仕様]**

- 画面に入力ボックス（ナンバー）に値を入力して「クリック」ボタンを押すと、その入力値の回数をカウントして、合計貯金額が1万円に達したらその回数をアラートで表示する。
- 入力値が0以下の場合、別のアラートメッセージを表示する。
- 各アラートメッセージは<画面イメージ>を参照する。

**[使用定義]**

関数

```
bt : 「クリック」ボタンで動作する処理の関数名
```

変数

```
nissuu : 日数をカウントする（初期設定値：0）
chokin : 合計貯金額を保持する（初期設定値：0）
atai : フォーム要素（ナンバー）suu の情報を取得・数値化
フォーム要素（ナンバー）の名前
suu : 1日の貯金額が入力される場所
```

5. プレビュー画面で確認なさい。

<画面イメージ>

(1) ホーム画面

貯金シミュレーション

1万円に達するには何日かかる？

1日 円貯めると何日かかる？ クリック

A

(2) フォームに「1000」と入力・ボタン押下

1万円に達するには何日かかる？

1000 円貯めると何日かかる？ クリック

10日かかります クリック

B

(3) フォームに「10」と入力・ボタン押下

1万円に達するには何日かかる？

10 円貯めると何日かかる？ クリック

1日かかります クリック

C

(4) 「NG」とアラートが表示されている。

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(( )) --> Input1[ ]
    Input1 --> Input2[ ]
    Input2 --> Input3[ ]
    Input3 --> Decision{ }
    Decision --> Count1[ ]
    Count1 --> Count2[ ]
    Count2 --> Count3[ ]
    Count3 --> Count4[ ]
    Count4 --> Decision
    Decision --> Alert1[ ]
    Alert1 --> Alert2[ ]
    Alert2 --> Alert3[ ]
    Alert3 --> End(( ))
  
```

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[繰り返し「while」上級]をクリック。



## ページを構成（HTML 部分の作成）

1～3 は、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]

1 日あたりの貯金額を入力し、それが 1 万円に達するまでの日数をカウントし、表示するページ。

### インプット部分の確認

#### 3. の(4)入力フォーム

<ブロック>

```
<form>
    表示文字 1日
    <input type="number" name="suu">
    表示文字 円貯めると何日かかる？
    <input type="button" value="クリック" onclick="bt()"/>
</form>
```

<作成ページ例>



### 仕様部分を読み解く

[仕様]

- 画面の入力ボックス（ナンバー）に値を入力して「クリック」ボタンを押すと、その入力値の回数をカウントして、合計貯金額が 1 万円に達したらその回数をアラートで表示する。

→インプット部分の設定を確認。

→合計貯金額を計算して保管しておくための箱（変数）を準備。

→日数をカウントするための箱（変数）を準備。

→合計貯金額が 1 万円に達するまで、入力された値の加算・回数のカウントを繰り返す。

→カウントした回数はアラートメッセージに表示。

- 入力値が 0 以下の場合、別のアラートメッセージを表示する。

→入力された値が 0 以下の場合、別のアラートメッセージを表示。

- 各アラートメッセージは<画面イメージ>を参照する。

→入力値 0 以下の場合:<画面イメージ> (3) を参照。アラートメッセージは「NG」と確認。

→それ以外の場合:<画面イメージ> (2) を参照。アラートメッセージは「○日かかります」と確認。

## 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

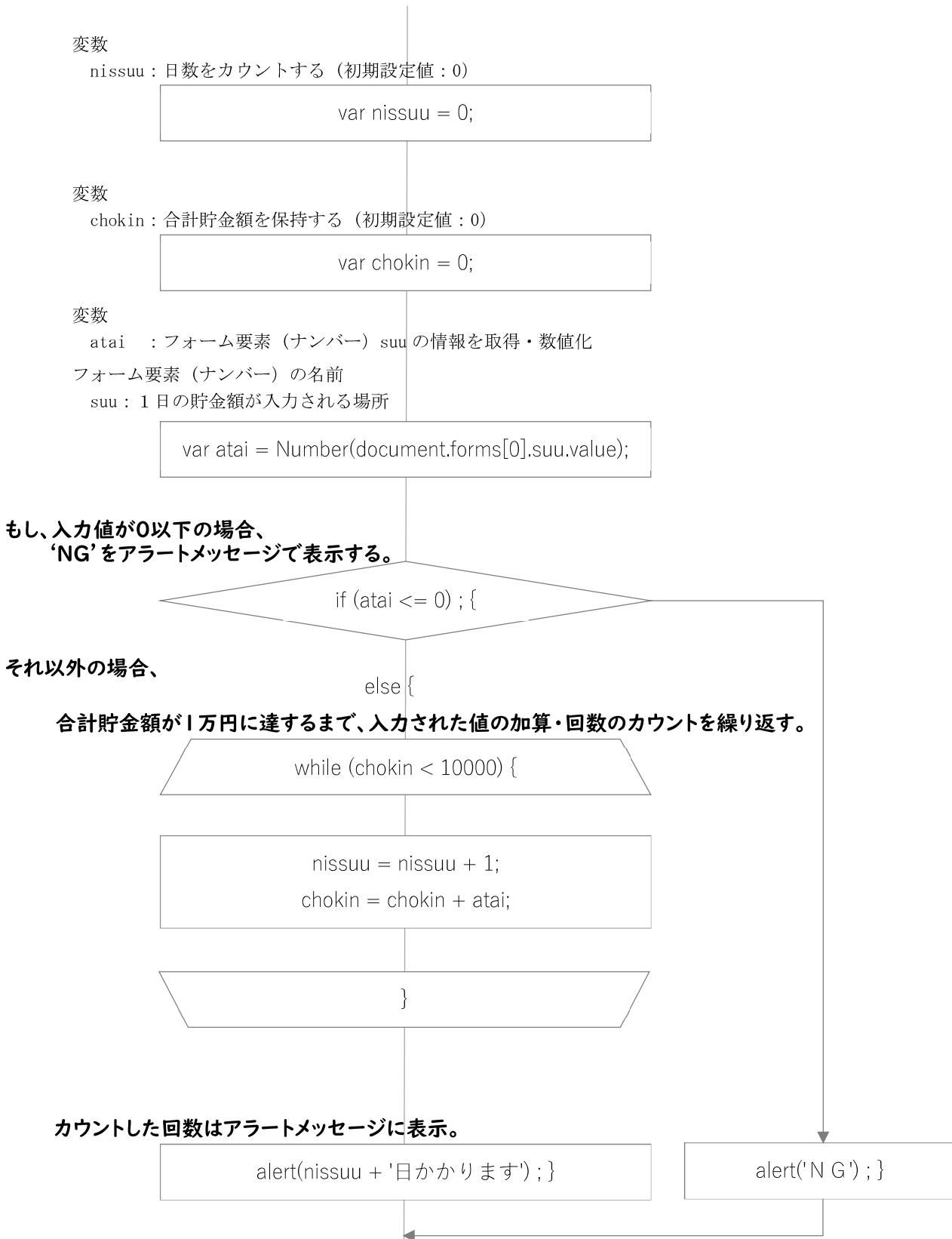
bt: 「クリック」ボタンで動作する処理の関数名

→処理はすべて関数「bt」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。



<フローチャート> 自分で考えて記入してみましょう。



## 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。

Javloc 繰り返しwhile 上級編

The screenshot shows the Javloc editor interface. In the top navigation bar, there are buttons for 'リセット', '元に戻す', 'やり直し', '解答例', 'プレビュー', and 'ページ保存'. Below the toolbar is a palette on the left containing categories like 'ページ', '文字入力', '画像', etc., with 'JavaScript' highlighted. A green box highlights the 'JavaScript' block in the palette. A red box highlights the 'link' block in the code area. The code area shows an HTML template with a script block in the head section. The source code is as follows:

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8">
<title>貯金をしよう</title>
<link rel="stylesheet" href="style.css" type="text/css">
<script></script>
</head>
```

Javloc 繰り返しwhile 上級編

The screenshot shows the Javloc editor interface. The palette on the left has 'JavaScript' selected. A green box highlights the 'function' block in the palette. A red box highlights the 'script' block in the code area. The code area shows the same HTML template as before, but now it includes a function definition within the script block. The source code is as follows:

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8">
<title>貯金をしよう</title>
<link rel="stylesheet" href="style.css" type="text/css">
<script>
<function () {}>
</script>
</head>
```

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「bt」を設定。ソース表示エリアで設定を確認。

※必ず半角で入力しましょう。

Javloc 繰り返しwhile 上級編

The screenshot shows the Javloc editor interface. The palette on the left has 'JavaScript' selected. A green box highlights the 'function' block in the palette. A red box highlights the 'script' block in the code area. The code area shows the same HTML template as before, but now it includes a function definition within the script block, with the function name set to 'bt'. The source code is as follows:

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8">
<title>貯金をしよう</title>
<link rel="stylesheet" href="style.css" type="text/css">
<script>
<function bt () {}>
</script>
</head>
```

Javloc 繰り返しwhile 上級編

The screenshot shows the Javloc editor interface. The palette on the left has 'JavaScript' selected. A green box highlights the 'function' block in the palette. A red box highlights the 'script' block in the code area. The code area shows the same HTML template as before, but now it includes a function definition within the script block, with the function name set to 'bt'. The source code is as follows:

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8">
<title>貯金をしよう</title>
<link rel="stylesheet" href="style.css" type="text/css">
<script>
<function bt () {}>
</script>
</head>
<body>
<h1>
```

リスト内[JavaScript]をクリック、枠内にある var nissuu = ; 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「0」を設定。ソース表示エリアで設定を確認。

Javloc 繰り返しwhile 上級編

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

ページ 文字入力 画像 見出し1 見出し2 フォーム JavaScript

```

<head>
  <meta charset="utf-8">
  <title>貯金をしよう</title>
  <link rel="stylesheet" href="style.css" type="text/css">
</head>
<body>
  <h1>貯金シミュレーション</h1>
  <h2>1万円に達するには何日かかる？</h2>
  <form>
    <input type="number" name="suu">
    円貯めると何日かかる？
    <input type="button" value="クリック" onclick="bt()">
  </form>
</body>
<html>
  <head>
    <meta charset="utf-8">
    <title>貯金をしよう</title>
    <link rel="stylesheet" href="style.css" type="text/css">
  </head>
  <body>
    <h1>貯金シミュレーション</h1>
    <h2>1万円に達するには何日かかる？</h2>
    <form>
      <input type="number" name="suu">
      円貯めると何日かかる？
      <input type="button" value="クリック" onclick="bt()">
    </form>
  </body>
</html>

```

Javloc 繰り返しwhile 上級編

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

ページ 文字入力 画像 見出し1 見出し2 フォーム JavaScript

```

<head>
  <meta charset="utf-8">
  <title>貯金をしよう</title>
  <link rel="stylesheet" href="style.css" type="text/css">
</head>
<body>
  <h1>貯金シミュレーション</h1>
  <h2>1万円に達するには何日かかる？</h2>
  <form>
    <input type="number" name="suu">
    円貯めると何日かかる？
    <input type="button" value="クリック" onclick="bt()">
  </form>
</body>
<html>
  <head>
    <meta charset="utf-8">
    <title>貯金をしよう</title>
    <link rel="stylesheet" href="style.css" type="text/css">
  </head>
  <body>
    <h1>貯金シミュレーション</h1>
    <h2>1万円に達するには何日かかる？</h2>
    <form>
      <input type="number" name="suu">
      円貯めると何日かかる？
      <input type="button" value="クリック" onclick="bt()">
    </form>
  </body>
</html>

```

リスト内[JavaScript]をクリック、枠内にある var chokin = ; ブロックを選択し、ドラッグして var nissuu = 0; 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。

Javloc 繰り返しwhile 上級編

```

<meta charset="utf-8">
<title>貯金をしよう</title>
<link rel="stylesheet" href="style.css" type="text/css">
<script>
  function bt() {
    var nissuu = 0;
    var chokin = 0;
    if(atai <= 0){
      表示文字 貯金をしよう
    }
    else {
      <script>
        function bt () {
          var nissuu = 0;
          while(chokin < 10000){
            nissuu = nissuu + 1;
            chokin = chokin + 1;
            if(nissuu == 10000){
              表示文字 1万円に達するには何日かかる?
            }
          }
        }
      </script>
    }
  }
</script>
<body>
  <h1>貯金シミュレーション</h1>
  <h2>1万円に達するには何日かかる?</h2>
  <form>
    <input type="button" value="クリック" onclick="bt()">
  </form>
</body>
<html>

```

Javloc 繰り返しwhile 上級編

```

<meta charset="utf-8">
<title>貯金をしよう</title>
<link rel="stylesheet" href="style.css" type="text/css">
<script>
  function bt() {
    var nissuu = 0;
    var chokin = 0;
    if(atai <= 0){
      表示文字 貯金をしよう
    }
    else {
      <script>
        function bt () {
          var nissuu = 0;
          while(chokin < 10000){
            nissuu = nissuu + 1;
            chokin = chokin + 1;
            if(nissuu == 10000){
              表示文字 1万円に達するには何日かかる?
            }
          }
        }
      </script>
    }
  }
</script>
<body>
  <h1>貯金シミュレーション</h1>
  <h2>1万円に達するには何日かかる?</h2>
  <form>
    <input type="button" value="クリック" onclick="bt()">
  </form>
</body>
<html>

```

リスト内[JavaScript]をクリック、枠内にある var atai = Number(~) ; ブロックを選択し、ドラッグして var chokin = 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 繰り返しwhile 上級編

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

ページ 文字入力 画像 見出し1 見出し2 フォーム JavaScript

```
<script>    document.charset="utf-8">
<title>    貯金をしよう
</script>    <body>    貯金をしよう
</title>
</body>
<function bt() {
    <script>
        <link rel="stylesheet" href="style.css" type="text/css">
        <script>
            function bt() {
                var nissuu = 0;
                var chokin = 0;
            }
        </script>
    </head>
    <body>
        <h1>貯金をしよう</h1>
        <form>
            <input type="text" name="atai" value="0" />
            <input type="button" value="貯金" onclick="bt()" />
            <input type="button" value="残高" />
        </form>
        <div>    貯金をしよう
        </div>
    </body>
}</function>
<var atai = Number(document.forms[0].suu.value);>
<var atai = document.forms[0].suu.value;*>
```

Javloc 繰り返しwhile 上級編

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

ページ 文字入力 画像 見出し1 見出し2 フォーム JavaScript

```
<meta charset="utf-8">
<title>
    表示文字 貯金をしよう
</title>
<link rel="stylesheet" href="style.css" type="text/css">
<script>
    function bt () {
        var nissuu = 0 ;
        var chokin = 0 ;
        var atai = Number(document.forms[0].suu.value);
    }
</script>
```

6 <title>
7 貯金をしよう
8 </title>
9 <link rel="stylesheet" href="style.css" type="text/css">
10 <script>
11 function bt () {
12 var nissuu = 0 ;
13 var chokin = 0 ;
14 var atai = Number(document.forms[0].suu.value);
15 }
16 </script>
17 </head>
18 <body>
19 <h1>
20 貯金シミュレーション

リスト内[JavaScript]をクリック、枠内にある if (atai <= 0) {～} ブロックを選択し、ドラッグして var atai = Number(～); 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 繰り返しwhile 上級編

The screenshot shows the Javloc editor interface. The top menu bar includes 'リセット', '元に戻す', 'やり直し', '解答例', 'プレビュー', '←', '→', 'ブロック読込', 'ブロック保存', and 'ページ保存'. On the left, there's a sidebar with categories: ページ, 文字入力, 画像, 見出し1, 見出し2, フォーム, and a red-highlighted 'JavaScript' category. The main workspace displays a Scratch script with a red box around the 'if' block. To the right is the generated JavaScript code:

```
<title>
貯金をしよう
</title>
<link rel="stylesheet" href="style.css" type="text/css">
<script>
function bt() {
    var nissuu = 0;
    var chokin = 0;
    var atai = Number(document.forms[0].suu.value);
    if(atai <= 0){
        bt();
        nissuu = 0;
        chokin = 0;
    }
}
```

Javloc 繰り返しwhile 上級編

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

ページ 文字入力 画像 見出し1 見出し2 フォーム JavaScript

```
<meta charset="utf-8">
<title> 表示文字 貯金をしよう </title>
<link rel="stylesheet" href="style.css" type="text/css" >
<script>
function bt () {
    var nissuu = 0 ;
    var chokin = 0 ;
    var atai = Number(document.forms[0].suu.value);
    if(atai <= 0){
        }
}
</script>
</head>
<body>
<h1> 貯金シミュレーション </h1>
<h2>
```

リスト内[JavaScript]をクリック、枠内にある alert('NG'); ブロックを選択し、ドラッグして if (atai <= 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。

Javloc 繰り返しwhile 上級編

```

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読込 ブロック保存 ページ保存
ページ 文字入力 画像 見出し1 見出し2 フォーム JavaScript
<meta charset="utf-8">
var atai = document.forms[0].suu.value;
alert(nissuu + '貯金をしよう');
<link rel="stylesheet" href="style.css" type="text/css">
<script>
function bt() {
    var nissuu = 0;
    var chokin = 0;
    var atai = Number(document.forms[0].suu.value);
    if(atai <= 0) {
        alert('NG');
    }
}

```

Javloc 繰り返しwhile 上級編

```

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読込 ブロック保存 ページ保存
ページ 文字入力 画像 見出し1 見出し2 フォーム JavaScript
<meta charset="utf-8">
<title>
    表示文字 貯金をしよう
</title>
<link rel="stylesheet" href="style.css" type="text/css">
<script>
function bt() {
    var nissuu = 0;
    var chokin = 0;
    var atai = Number(document.forms[0].suu.value);
    if(atai <= 0) {
        alert('NG');
    }
}

```

リスト内[JavaScript]をクリック、枠内にある else { } ブロックを選択し、ドラッグして if (atai <= 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 繰り返しwhile 上級編

```

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読込 ブロック保存 ページ保存
ページ 文字入力 画像 見出し1 見出し2 フォーム JavaScript
function bt() {
    alert('NG');
    var nissuu = 0;
    var chokin = 0;
    if(atai <= 0) {
        var atai = Number(document.forms[0].suu.value);
        if(atai <= 0) {
            alert('NG');
        }
    }
}

```

Javloc 繰り返しwhile 上級編

```

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読込 ブロック保存 ページ保存
ページ 文字入力 画像 見出し1 見出し2 フォーム JavaScript
var chokin = 0;
var atai = Number(document.forms[0].suu.value);
if(atai <= 0) {
    alert('NG');
}
else {
}

```

リスト内[JavaScript]をクリック、枠内にある while (chokin < □) {～} ブロックを選択し、ドラッグして else {～} ブロックの中にドロップ。数値「10000」を設定。ソース表示エリアで設定を確認。

Javloc 繰り返しwhile 上級編

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

ページ 文字入力 画像 見出し1 見出し2 フォーム JavaScript

var chokin = 0;  
var atai = Number(document.forms[0].suu.value);  
if(atai <= 0){  
 alert('N G');  
}  
else {  
 while(chokin < □){  
 ...  
 }  
}

var nissuu = 0;  
var chokin = 0;  
var atai = Number(document.forms[0].suu.value);  
if (atai <= 0) {  
 alert('N G');  
}  
else {  
 ...  
}  
else {  
 while (chokin < 10000) {  
 ...  
 }  
}  
...  
1万円に達するには何日かかる?

Javloc 繰り返しwhile 上級編

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

ページ 文字入力 画像 見出し1 見出し2 フォーム JavaScript

var chokin = 0;  
var atai = Number(document.forms[0].suu.value);  
if(atai <= 0){  
 alert('N G');  
}  
else {  
 while(chokin < 10000){  
 ...  
 }  
}

var chokin = 0;  
var atai = Number(document.forms[0].suu.value);  
if (atai <= 0) {  
 alert('N G');  
}  
else {  
 while (chokin < 10000) {  
 ...  
 }  
}  
...  
1万円に達するには何日かかる?

リスト内[JavaScript]をクリック、枠内にある nissuu = nissuu + 1; ブロックを選択し、ドラッグして while (chokin < 10000) {～} 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。

Javloc 繰り返しwhile 上級編

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

ページ 文字入力 画像 見出し1 見出し2 フォーム JavaScript

var chokin = 0;  
var atai = Number(document.forms[0].suu.value);  
if(atai <= 0){  
 alert('N G');  
}  
else {  
 while(chokin < 10000){  
 nissuu = nissuu + 1;  
 chokin = chokin + 1;  
 }  
}  
var nissuu = 0;

13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38

var chokin = 0;  
var atai = Number(document.forms[0].suu.value);  
if (atai <= 0) {  
 alert('N G');  
}  
else {  
 while (chokin < 10000) {  
 }  
}  
}/>  
</script>  
</head>  
<body>  
<h1>  
貯金シミュレーション  
</h1>  
<h2>  
1万円に達するには何日かかる？  
</h2>  
  
<form>  
1日  
<input type="number" name="suu">  
円貯める日何日かかる？  
<input type="button" value="クリック" onclick="bt()">  
</form>

Javloc 繰り返しwhile 上級編

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

ページ 文字入力 画像 見出し1 見出し2 フォーム JavaScript

var chokin = 0;  
var atai = Number(document.forms[0].suu.value);  
if(atai <= 0){  
 alert('N G');  
}  
else {  
 while(chokin < 10000){  
 nissuu = nissuu + 1;  
 chokin = chokin + 1;  
 }  
}  
</script>

function bt() {  
 var nissuu = 0;  
 var chokin = 0;  
 var atai = Number(document.forms[0].suu.value);  
 if (atai <= 0) {  
 alert('N G');  
 }  
 else {  
 while (chokin < 10000) {  
 nissuu = nissuu + 1;  
 }  
 }  
}

リスト内[JavaScript]をクリック、枠内にある chokin = chokin + □; ブロックを選択し、ドラッグして nissuu = nissuu + 1; ブロックの下にドロップ。変数名「atai」を設定。ソース表示エリアで設定を確認。

Javloc 繰り返しwhile 上級編

```

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読込 ブロック保存 ページ保存
ページ 文字入力 画像 見出し1 見出し2 フォーム JavaScript
var chokin = 0;
var atai = Number(document.forms[0].suu.value);
if(atai <= 0){
    alert('NG');
}
else {
    while(chokin < 10000) {
        nissuu = nissuu + 1;
        while(chokin <= 10000) {
            nissuu = nissuu + 1;
        }
    }
}
</head>
nissuu = nissuu + 1;
</html>
var nissuu = 0;

function bt() {
    var nissuu = 0;
    var chokin = 0;
    var atai = Number(document.forms[0].suu.value);
    if (atai <= 0) {
        alert('NG');
    }
    else {
        while (chokin < 10000) {
            nissuu = nissuu + 1;
        }
    }
}
</script>
</head>
<body>
<h1>貯金シミュレーション</h1>
<h2>1万円に達するには何日かかる？</h2>

<form>
    1日
</form>

```

Javloc 繰り返しwhile 上級編

```

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読込 ブロック保存 ページ保存
ページ 文字入力 画像 見出し1 見出し2 フォーム JavaScript
var chokin = 0;
var atai = Number(document.forms[0].suu.value);
if(atai <= 0){
    alert('NG');
}
else {
    while(chokin < 10000) {
        nissuu = nissuu + 1;
        chokin = chokin + atai;
    }
}
</script>
var nissuu = 0;
var chokin = 0;
var atai = Number(document.forms[0].suu.value);
if (atai <= 0) {
    alert('NG');
}
else {
    while (chokin < 10000) {
        nissuu = nissuu + 1;
        chokin = chokin + atai;
    }
}
</script>
</head>
<body>
<h1>貯金シミュレーション</h1>

```

リスト内[JavaScript]をクリック、枠内にある alert(nissuu + '日かかります'); ブロックを選択し、ドラッグして while (chokin < 10000) {～} 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 繰り返しwhile 上級編

```

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読込 ブロック保存 ページ保存
ページ 文字入力 画像 見出し1 見出し2 フォーム JavaScript
var chokin = 0;
var atai = Number(document.forms[0].suu.value);
if(atai <= 0){
    alert('N G');
}
else {
    while(chokin < 10000){
        nissuu = nissuu + 1;
        chokin = chokin + atai;
    }
    alert(nissuu + '日かかります');
}
alert('N G');

```

```

FUNCTION DL() {
    var nissuu = 0;
    var chokin = 0;
    var atai = Number(document.forms[0].suu.value);
    if (atai <= 0) {
        alert('N G');
    }
    else {
        while (chokin < 10000) {
            nissuu = nissuu + 1;
            chokin = chokin + atai;
        }
    }
    alert(nissuu + '日かかります');
}
</script>
</head>
</body>
<h1>貯金シミュレーション</h1>
<h2>

```

Javloc 繰り返しwhile 上級編

```

リセット 元に戻す やり直し 解答例 プレビュー ← → ブロック読込 ブロック保存 ページ保存
ページ 文字入力 画像 見出し1 見出し2 フォーム JavaScript
var chokin = 0;
var atai = Number(document.forms[0].suu.value);
if(atai <= 0){
    alert('N G');
}
else {
    while(chokin < 10000){
        nissuu = nissuu + 1;
        chokin = chokin + atai;
    }
    alert(nissuu + '日かかります');
}

```

```

var nissuu = 0;
var chokin = 0;
var atai = Number(document.forms[0].suu.value);
if (atai <= 0) {
    alert('N G');
}
else {
    while (chokin < 10000) {
        nissuu = nissuu + 1;
        chokin = chokin + atai;
    }
    alert(nissuu + '日かかります');
}
</script>
</head>
</body>
<h1>貯金シミュレーション</h1>
<h2>

```

## 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 繰り返しwhile 上級編

<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。

# プログラム全体の流れ

- ① 数値入力欄から値が入力され、クリックボタンが押されると、関数が実行される。
- ② 変数 nissuu に 0 を代入する。
- ③ 変数 chokin に 0 を代入する。
- ④ 変数 atai に①の数値入力欄「suu」を数値化した値を代入する。
- もし変数 atai が 0 以下だった場合、「NG」というアラートメッセージを表示する。
- ⑤ ④の条件でない場合、(ア)・(イ)の処理を行う。
- (ア) while 文を使用し、変数 atai が 10000 未満の間、以下の処理を繰り返す。
- 変数 nissuu に 1 を足す・変数 chokin に変数 atai を足す
- (イ) 変数 nissuu の値と「日かかります」を文字列としてアラートメッセージに表示する。

## 【HTML 部分・作成ページ例】

①

```
<form>
  表示文字 1 日
  <input type="number" name="suu">
  表示文字 円貯めると何日かかる?
  <input type="button" value=" クリック " onclick=" bt ()">
</form>
```

0と入力した場合



1000と入力した場合



## 【JavaScript 関数 bt が実行】

①

```
var nissuu = 0 ;
var chokin = 0 ;
```

③

```
var atai = Number(document.forms[0].suu.value);
```

atai = 0

④

```
if(atai <= 0){
  alert('NG');
}
```

atai = 1000

⑤

```
else {
  (ア) while(chokin < 10000 ){
    nissuu = nissuu + 1;
    chokin = chokin + atai ;
  }
  (イ) alert(nissuu + '日かかります');
}
```

## <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「」(シングルクオーテーション)または「」(ダブルクオーテーション)で囲む。  
他に「``」(バッククオート)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。