Step6 / JavaScript

JavaScript（ジャバスクリプト）とは、主にウェブサイトに動きや対話性を持たせるために使われるプログラミング言語です。HTML がウェブページの構造を、CSS がデザインを定義するのに対し、JavaScript はユーザーの操作に応じて要素を動的に変化させたり、計算を行ったりします。

JavaScript の記述：

```
<!DOCTYPE html>
<html lang="ja">
<head>
  <meta charset="UTF-8">
  <title>Demo</title>
</head>
<body>
  <h1>Hello world</h1>

  <script>
    // ここに JavaScript の記述します
    alert("ケロケロ");
  </script>
</body>
</html>
```

解説：

- <script></script> のように、ブラウザに対して「ここに JavaScript 等のプログラムが記述されている」と認識させるために、プログラム部分をこれらのタグ（開始タグと終了タグ）で囲んで記述します。
- alert(“◯◯◯”); という関数は、◯◯◯という文字列をブラウザにダイアログ出力します。
- JavaScript のコメントは 一行の場合、// と記述します。
  複数行の書くこともできます。その場合は CSS コメントと同じ /* ◯◯◯ */ で囲います。

Step7 / 入力・出力

JavaScript からの入力・出力方法にはいくつかの方法があります。
・ダイアログ：「alert()」「confirm()」「prompt()」
・コンソール：「console.log()」　※ 開発中に利用
・HTML（DOM 操作）：「document.getElementById("msg").textContent」 等

alert()：

```
<script>
  alert( "ケロケロ" );
</script>
```

alert() は""の文字をダイアログ表示します。「OK」のみのボタンを表示します。

confirm()：

```
<script>
  // 確認ダイアログを表示
  confirm("カエルが好きですか？");
</script>
```

- confirm() は「OK」「キャンセル」の 2 択ボタンを表示します。
- OK を押すと true、キャンセルを押すと false が返ります。

prompt()：

```
<script>
  // 入力ダイアログを表示
  prompt("あなたの名前を教えてください:");
</script>
```

- prompt() は「入力欄」と「OK」「キャンセル」のボタンを持つダイアログです。

console.log()：

```
<script>
  console.log( "こんにちは" );
</script>
```

alert()のように JavaScript に書いた文字を表示します。
ただし、alert()はダイアログ表示することに対して、console.log()は Dev Tools に出力します。

DOM 操作：

```
<p id="note"></p>

<script>
  document.getElementById("note").textContent = "いい天気ですね！ ";
</script>
```

alert()はダイアログ表示することに対して、html タグを指定して操作し内容を置き換えます。
これを DOM 操作といいます。
DOM (Document Object Model)は、HTML などのマークアップ言語で書かれた文書を、JavaScript などのプログラミング言語から操作・変更できるようにするための API（Application Programming Interface）です。

なお、基本的に、プログラムの行末には「;」の記号を書きます(省略することもできますが、習慣として付けるようにしましょう)。

Step8 / 変数

const： 定数 ＝ 再代入が不可。

```
<script>
  const b = 3;
  alert( b );
</script>
```

let： 変数 ＝ 再代入が可能。

```
<script>
  let a;
     a = 3;
  alert( a );
</script>
```

2 行を 1 行にまとめて

```
<script>
  let a = 3;
  alert( a );
</script>
```

以下に箇条書きで「変数」の説明を行います。
・変数は、値の入れ物、あるいは値を入れる箱と考えることができる
・変数は、英数字で（文字数は自由）で表される（例：a x b1 b2 time sokudo 等々）
・ユーザの入力や、プログラムの進行によって、中身の値は変化することがある
（変化させることができる）
・「＝」の記号を使って、変数に値を代入できる（「＝」の右側から左側へ代入する）
・変数は自由に作れるが、「if」や「alert」等、プログラム自体が機能を提供する
ために使う「予約語」は変数として使うことはできない
・変数は、なるべくわかりやすい名前をつけることで、他の人（あるいは時間が経った後の自分）がプログラムを見たときに理解しやすくなります。
・JavaScript では、プログラム中で使用する変数は、プログラムの冒頭で、let という書式を使って「この文字(列)を、変数として使用する」ことを宣言しておくのが一般的な書き方になります。（これらの記述は省略することもできます）。

Step7 では　alert("ケロケロ"); の書式で直接的に表示内容を指定しましたが、alert(a); と記述することで、変数 a の中身を表示することができます(上述のプログラムを実行すると、メッセージとして「３」が表示されます。

Step9 / データの種類

変数に格納できるデータの種類：

```
<script>
  let name = "ケロケロ";     // 文字列(String)
  let age = 10;              // 数値(Number)
  let isFrog = true;         // 真偽値(Boolean)
  let colors = ["赤", "青"]; // 配列(Array)
  let info = { name: "カエル", color: "緑" }; // オブジェクト(Object)

  alert( name );
</script>
```

文字列は “” で囲み、数値は 囲み不要です。
真偽値は特殊で true/false の文字ですが囲み不要です。
配列、オブジェクトは 別途説明します。ここでは異なる記法になることを理解してください。
また、未定義（undefined）や null といったデータの種類がありますが、割愛します。

Step10 / 演算子（四則演算等）

```
<script>
  // 変数の準備
  let a = 5;
  let b = 2;

  // 各演算
  let add = a + b;     // 足し算
  let sub = a - b;     // 引き算
  let mul = a * b;     // 掛け算
  let div = a / b;     // 割り算
  let mod = a % b;     // 余り
  let pow = a ** b;    // べき乗(a の b 乗)

  alert( add );
</script>
```

注意点：

データ型の違いにより演算結果が変わります。以下の変数 a と 変数 b の値は何になるでしょうか。

```
<script>
  let x = "60"; // 文字列
  let y = 60;   // 数値

  let a = x + x;
  let b = y + y;

  alert( a );
  alert( b );
</script>
```

Step11 / ランダム生成

```
<script>
  let num = Math.random();
  alert( num );
</script>
```

以下に箇条書きで「**Math.random( )**」の説明を行います。
・ゲーム等に不可欠な「ランダムさ」を導入するための方法です。
・プログラム中でランダム性が必要になる局面は様々な場面が考えられますが、JavaScript の場合は、すべてこの命令を基にして作ります。
・「Math.random（ ）」は呼び出すたびに（使用するたびに）異なる値を返します。
・「Math.random（ ）」は 0 以上 1 未満の範囲の小数を、一つ選んでくれる（ゲットできる）関数です。「Math.random（ ）」の Math は数学（mathematics）の意味で、Math.random（ ）の他に、サイン関数の Math.sin（ ）や、絶対値を求める関数の Math.abs( )、小数点以下を切り捨てる関数 Math.floor( )等の、数学的な機能が用意されています。（これらの詳細は別途説明します）
・上述のプログラムは、変数 num を使用せずに、

```
alert( Math.random( ) );
```

という一行を書くことで、同じ動作が実現できます（複数の括弧が組み合わされていますが、基本的に括弧の内側から処理が行われます）。ただし、上述のプログラムのように、一旦変数にランダムな値を代入して(確保して)、以降の処理を、その変数を使って行う方法が一般的です。

とりあえず、0 以上 1 未満のランダムな小数がそのまま表示されるだけのプログラムですが、これはもっとも原始的な「おみくじ」（あるいは「占い」）と言えると思います。例えば値が 1 に近ければめでたく、 0 に近いとめでたくない、というルールで考えることができます。

Step12 / 条件分岐（if）

```
<script>
  let num = Math.random();

  if( num > 0.5 ){
    alert( "吉" );
  }

  if( num <= 0.5 ){
    alert( "凶" );
  }
</script>
```

ｉｆ文の役割は、

・条件が満たされている場合は、指定された処理内容を実行する
・条件が満たされていない場合は指定された処理は実行しない（スキップする）

となります。if 文の書き方は実行内容が 1 行であれば、

if（条件）｛　条件が満たされた場合に行う処理；｝

と書けますが、一般的に行う処理は複数行にわたる場合が多く、そのような場合は、下記のように書きます。なお、{ }の中に記述される実行内容は、左側にスペースを入れて記述を右側にずらすことで、それらが if 文の実行内容であることがわかりやすくなります。（これをインデントといいます）

```
if（条件）｛
  条件が満たされた場合に行う処理１；
  条件を満たされた場合に行う処理２；
  条件を満たされた場合に行う処理３；
}
```

「条件」の部分では、基本的に二つの値の比較を行います。値としては、直接「0.5」等の数値を置くことや変数を置くことができます。二つの値の比較のために、以下の記号を使うことができます。
比較演算子
＞　　：左辺の値が右辺の値より大きい場合は・・・
＜　　：右辺の値が左辺の値より大きい場合は・・・
＞＝　：左辺の値が右辺の値以上の場合は・・・

＜＝　：右辺の値が左辺の値以上の場合は・・・
＝＝　：左右の値が同じ場合は・・・
！＝　：左右の値が異なる場合は・・・

１頁の７番目の項目で、文字の間に空白を入れることで、プログラムが見やすくなると説明しましたが、上記の「＝＝」や「＞＝」等、if 文の条件指定に使う記号については、間に空白を入れてはいけないことに注意してください。また、＝＝ と書くべきところを、単に ＝ と書いてしまう間違いが多発しますので、注意して下さい。「＝」を１個だけ使うと、既に説明した右辺の値を左辺の変数に入れるための「代入」の意味になり、「＝＝」とは別の意味になります。javascript をはじめとする様々なプログラミング言語がこの「＝＝」を採用していますが、私は間違いやすい良くない仕様だと思っています。

Step13 / if 文を増やしてメッセージの種類を多様にする

```
<script>
  let num = Math.random();

  if( num > 0.5 ){
    alert( "大吉です" );
  }

  if( num > 0.25 && num <= 0.5  ){
    alert( "吉です" );
  }

  if( num <= 0.25 ){
    alert( "凶です" );
  }
</script>
```

上述のプログラムで、これまで２種類だったメッセージを３種類に増やすことができました。プログラムの記述から、大吉、吉、凶、それぞれの出現確率を考えてみて下さい。プログラム中に新しい記号「&&」が含まれています。この「&&」を使っている二つめのｉｆ文では、変数の値の上限と下限の両方が指定されています。このことは、中学、高校の数学では、

0.25 < num <= 0.5

というようなコンパクトな書き方で条件を指定できたのを憶えている人もいると思いますが、JavaScript ではこのように書くことができず、二つの不等号に分けた上で、それを組み合わせる記号を

使う必要があります。今回は、二つ不等号で示された条件の両方を満たす必要があるので「and（かつ）論理」を適用する必要がありますが、and 論理を示す記号が「&&」になります。条件の組み合わせ方には、この and（かつ）の他に or（または）もありますが、or は「||」で記述します（「|」の記号は、「¥」のキーのシフトで打つことができます）。なお、これらの && と || は、文字の間に空白を入れてはいけません。

上述のプログラムの、「メッセージ」「それぞれの出現確率」「異なるメッセージの数」を自由に変えて試してみて下さい。

Step14 / if と else の組み合わせ

```
<script>
  let num = Math.random();

  if( num > 0.5 ){
    alert( "吉です" );
  } else {
    alert( "凶です" );
  }
</script>
```

if（条件）｛
　条件が満たされた場合に行う処理；
｝else｛
　条件が満たされなかった場合に行う処理；
｝

---=

課題：

Step14 のプログラムは、else を使う if 文の別の書き方を使って同じ動作を実現することも可能です。

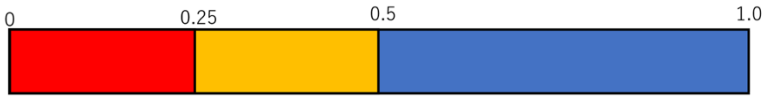
if（条件 1）｛
　条件 1 が満たされた場合に行う処理；
｝else if（条件 2）｛
　条件 1 を満たさず、条件 2 が満たされた場合に行う処理；
｝else｛
　条件 1 も 2 も満たされなかった場合に行う処理；
｝

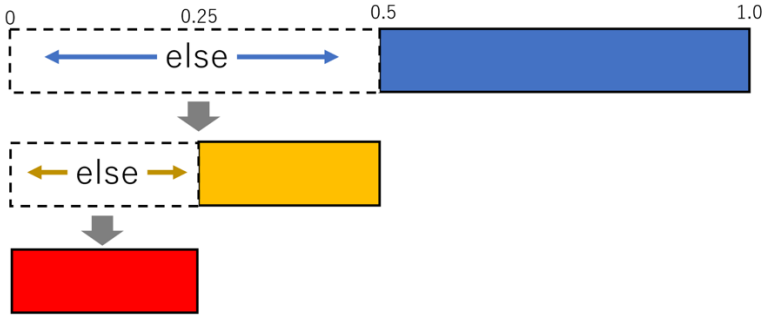
答え：

```
<script>
  let num = Math.random();

  if (num > 0.5) {
    alert("大吉です");
  } else if (num > 0.25) {
    alert("吉です");
  } else {
    alert("凶です");
  }
</script>
```

という書き方で実現できます。二つの書き方の考え方を図で示します。





課題：

上述の考え方を応用して、大凶を加えて 4 つの結果を出せるおみくじを作ります。

次ページのサンプルを見る前に、自分でプログラムを作ってみてください。

答え：

```
<script>
  let num = Math.random();

  if (num > 0.5) {
    alert("大吉です");
  } else if (num > 0.25) {
    alert("吉です");
  } else if (num > 0.125) {
    alert("凶です");
  } else {
    alert("大凶です");
  }
</script>
```

Step15 / 繰り返し（for）

おみくじを 3 回繰り返す場合、どのように記述するのでしょうか。
以下のコードは、単純に 3 回同じコードを繰り返し記述しています。
それぞれのおみくじの冒頭に「おみくじが始まること」を伝えるメッセージの表示のための alert 文を挟んでいます。

```
<script>
    let num;
    // 1 回目
    alert( "おみくじをひきましょう" );
    num = Math.random();
    if (num > 0.5) {
      alert("大吉です");
    } else if (num > 0.25) {
      alert("吉です");
    } else if (num > 0.125) {
      alert("凶です");
    } else {
      alert("大凶です");
    }

    // 2 回目
```

```
    alert( "おみくじをひきましょう" );
    num = Math.random();
    if (num > 0.5) {
      alert("大吉です");
    } else if (num > 0.25) {
      alert("吉です");
    } else if (num > 0.125) {
      alert("凶です");
    } else {
      alert("大凶です");
    }

    // 3回目
    alert( "おみくじをひきましょう" );
    num = Math.random();
    if (num > 0.5) {
      alert("大吉です");
    } else if (num > 0.25) {
      alert("吉です");
    } else if (num > 0.125) {
      alert("凶です");
    } else {
      alert("大凶です");
}
  </script>
```

同じ内容を複数回記述することで繰り返しの動作が実現できますが、これが 10 回、100 回と繰り返す場合にはプログラムの記述量が多くなり、またおみくじの挙動を少し変えたいと思ったときに改変しなくてはいけない箇所も増え手間が大変です。このような同一の処理を繰り返し行うために導入されている仕組みの代表例が「for 文」になります。for 文を使ってどのように記述すればよいか、考えてみて下さい。

```
<script>
let num;
let i;

for( i = 1; i < 4; i++ ){

   alert( "おみくじをひきましょう" );

   num = Math.random( );
   if( num > 0.5 ){
      alert( "大吉です" );
   }
   if( num > 0.25 && num <= 0.5  ){
      alert( "吉です" );
   }
   if( num <= 0.25 ){
      alert( "凶です" );
   }
}
</script>
```

for 文は、通常以下のように記述します。

```
for ( ①； ②； ③ ) {
    実行内容１；
    実行内容２；
    実行内容３；
    実行内容４；
 }
```

① for 文とセットで使用する変数（例えば i ）に初期値（最初の値）を代入する
② 繰り返しを行う条件式の指定（変数がどの範囲の値の場合に繰り返しを行うのか）
③ for 文では、通常一回繰り返しを行う毎に、変数の値を変化させます。
ここで、変数をどのように変化させるのかを指定します

上記のコードでは、for 文の最初の行に、

```
for（ i＝ 1;i＜4； i++ ） {
```

と書かれています。この場合は、実行内容が 3 回（ i が 1 から始まって 4 になる前まで）繰り返されます。「i++」と書かれているのは、1 回処理を行うたびに、 i の値を 1 増やす、という意味の書式です。変数の値を 1 増やす、という処理は何通りか書き方がありますが、この記法が最もシンプル(記述する文字数が少なくて済む)なので、for 文をはじめとして多くの場面で頻繁に使われます。

なお、for 文の最初の行は、

```
for（  let i＝ 1;i＜4； i++ ） {
```

このように変数「i」の初期化の部分に「let」をつけることもあります。これを付けることで、この for 文以外の場所で、変数「i」を活用していたとしても、その値に for 文が影響を及ぼさなくなります。「let」を付けておくことで、安心になる面もありますが、この資料では基本的に付けない形で説明します。

以下の記述でも「3 回繰り返す」ことは可能です。

```
for（ i＝1； i<= 3； i++ ） {
for（ i＝0； i＜3； i++ ） {
```

※ ただし、後者は i の値が変化する範囲が変わります）。

for 文は、繰り返しの条件が満たされない状態になった時に（この場合は i の値が増えて 4 になった時に）繰り返しのループから抜けて、for 文の下（for 文のループを閉じるための記号「 } 」の下）に処理が移動します。

if 文や for 文では、プログラムの中に、（ ）や { } 等の様々な括弧が現れています。これらの括弧は、必ずペアで使用します。右向きと左向き（開く方と閉じる方）の片方だけが存在することはなく、必ず相手が存在していなくてはいけません。それぞれの括弧がどのように対応しているのか（その命令の影響範囲がどこまでなのか）をよく意識して、プログラムを作成する（読む）必要があります。なお、括弧の対応関係は、VS Code を使うと、画面上で簡単に確認することができます(メモ帳にはこの機能はありません)。

Step16 / ランダムな整数を得る

おみくじのスタイルを少し変えて、その日の「ラッキナンバー」を伝えるプログラムを考えます。多くの人にとって、小数を伝えられてもピンとこないと思いますので、整数を伝える形を考えます。

```
let  num;
num = Math.floor( Math.random( ) * 10 );
alert( num );
```

Math.random( )で生成されるのは「０以上、１未満の小数」ですが、それに１０をかけることで「０以上、１０未満の小数」に変換できます。その状態にさらに Math.floor（ ）を使って小数点以下の切り捨てを行うことで、０， １， ２， ３， ４， ５， ６， ７， ８， ９の１０通りの整数のいずれかをランダムに得ることができます。

```
// 他の方法(1)
let  num = Math.floor( Math.random( ) * 10 );
alert( num );

// 他の方法(2)
alert(Math.floor( Math.random( ) * 10 ));
```

課題：

プログラムによっては、「０は必要なく、１から１０までの整数がほしい」という場合もあります。上記のプログラムを少し変えることで実現できます。どのように変えればよいか、次ページのサンプルを見る前に考えてみて下さい。

```
let  num;
num = Math.floor( Math.random( ) * 10 + 1);
alert( num );

// 他の方法
num = Math.floor( Math.random( ) * 10 ) + 1;
```

ここまでが、ランダムな整数を作る方法です。
さて、いきなり数字を提示するためぶっきらぼうな印象を与えます。もう少し「ラッキーナンバーを伝えてあげる」という親切なニュアンスを出すにはどうしたら良いか、考えてみて下さい。

Step17 / メッセージの出し方を工夫する

```
let  num;
num = Math.floor( Math.random( ) * 10 + 1);
alert( "あなたの本日のラッキナンバーを教えてあげます" );
alert( num );
```

この方法では、ダイアログを 2 段階表示しています。次の step では、これらを 1 段階に省くための方法を説明します。

```
let  num;
num = Math.floor( Math.random( ) * 10 + 1);
alert ( "あなたの本日のラッキナンバーは " + num + " です" );
```

+ は、本来計算に使われる演算子ですが、文字を連結する際にも使われます。

さらに、「あなた」を入力した名前にします。
メッセージボックスには、これまで学習した alert()の他に prompt() があります。
この prompt()を使い、「あなた」の箇所を入力した名前で表示するようにします。

```
<script>
  // 入力ダイアログを表示
  let name = prompt("あなたの名前を教えてください:");

  if (name !== "") {
    let  num;
    num = Math.floor( Math.random( ) * 10 + 1);
    alert ( name + "の本日のラッキナンバーは " + num + " です" );
  }
</script>
```

ユーザが打ち込んだ文字列は、代入文の左辺に置いた変数の中に代入されます。
このとき変数に入るのは基本的に文字列になります。
今回の場合、 prompt()で入力された文字列は、変数 name に格納されます。
また、if(name !== “”) により入力がなかった場合は、実行しない条件分岐を入れています。

POINT： 戻り値

```
<script>
 // 入力ダイアログを表示
 // 戻り値 = 文字列 or null
 let name = prompt("あなたの名前を教えてください:");

 // 確認ダイアログを表示
 // 戻り値 = true または false
 let likeFrog = confirm("カエルを好きですか？");
</script>
```

Step18 / 数字当てゲーム

```
<script>
  // 変数宣言
 let randomNum;
 let inputNum;

 // 入力ダイアログを表示
 inputNum = prompt("これから 1～10 の数字を作りますので、当ててください");
 inputNum = Number(inputNum);
 // ランダムな整数を生成
 randomNum = Math.floor( Math.random( ) * 10 + 1);

 if( inputNum == randomNum ){
   alert( "すごい。あたりです。確かに " + randomNum + " でした。");
 } else {
   alert( "はずれです。私が考えたのは " + randomNum + " でした");
 }
</script>
```

課題：

上述の数字当てゲームのプログラムを以下に変更してください。

1）出題を 3 回繰り返してください。

2）prompt() のメッセージは、以下のように表示してください。？は回数の数字です。

prompt()で受け取った値は文字列です。Number(値)のように値を数字化してください。

第？問目の出題 1～10 の数字を作りますので、当ててください

3）最後（3 回の回答）が終わった後に、正解数を表示してください。

答え

```
<script>
  // 変数宣言
  let randomNum;
  let inputNum;
  let i;          // for 文のために使用する変数
  let score = 0; // 得点を示す変数(正解数は0から始まるので、初期値を0にする)

  for( i = 1; i <= 3; i++){
    // 入力ダイアログを表示
    inputNum = prompt("第 " + i + " 問目の出題 1～10 の数字を当ててください");
    inputNum = Number(inputNum);

    // ランダムな整数を生成
    randomNum = Math.floor( Math.random( ) * 10 + 1);

    if( inputNum == randomNum ){
          alert( "すごい。あたりです。確かに " + randomNum + " でした。");
          score = score + 1; // 得点を 1 点増やす
    } else {
          alert( "はずれです。私が考えたのは " + randomNum + " でした");
    }
  }

  alert( "ゲーム終了！あなたの得点は " + score + " 点です。" );
</script>
```

解説

1）出題を 3 回繰り返してください。
=> for 文で 3 回繰り返しています。

2）prompt() のメッセージは、以下のように表示してください。？は数字です。
【第？問】 1～10 の数字を作りますので、当ててください。
=> for 文の 変数 i を”第” + i + “問目・・・” に利用しています。

3）最後（3 回の回答）が終わった後に、正解数を表示してください。
=> あたり（inputNum == randomNum）の場合に、score 変数をカウントアップしています。
また、最後に alert()で score を表示することで合計数 = 正解数を表示しています。