1. 実習環境の導入

1-1. Monaca の登録

アシアル株式会社が提供するハイブリッドアプリ(Android と iOS の両方で動作する HTML5 と JavaScript をベースとしたアプリ)の開発環境である Monaca(フリープラン)を利用する。

https://ja.monaca.io/

この環境への登録法とその操作については、以下の URL の pdf ファイルに記載されているが、ここでも簡単に紹介する。

https://edu.monaca.io/wp-content/uploads/book001_typeb_1-4.pdf

≪実習 1.1≫ 以下の手順に従って、自分のアカウントを作成せよ。〔pdf ファイル 8～9 ページ〕
ここで、アカウントには Google や Facebook のアカウントが利用できる。必要ならば、Google アカウントを取得しておくと良い。

① **ブラウザ Google Chrome で**公式サイト URL を開き、次の手順で登録する。
https://ja.monaca.io/
【注】 **Monaca はブラウザ Google Chrome に対応しており、他のブラウザでは正しく機能しない事に注意する。**

② 画面中央または右上隅の無料トライアルをクリックして、メールアドレスとパスワードを入力して、アカウント新規作成をクリックする。この段階では「仮登録」である。

③ 仮登録したメールアドレスに届くメールに記載された URL のサイトにて本登録する。無料トライアルを始めて開始する時に表示されるウインドウにおいて、無料トライアル期間(14 日間)終了後の利用プランを問われる個所があるが、そこでは「Free プラン」を選択する。他のプラン(有料プラン)を選択すると、余計な情報提供が求められる。次に、次へ進むをクリックする。

④ 続いて表示される「Monaca Dashboard」にてアプリを開発していく。「Monaca Dashboard」ウインドウの右上隅のマーク をクリックすると、アカウント情報とともに現在登録の料金プランを確認できる。また、ここからログアウトすることもできる。

⑤ 登録後に再度 Monaca にログインする時は、公式サイト(https://ja.monaca.io/)からログインを選択して、登録したメールアドレスとパスワードを入力すると、「Monaca Dashboard」に移動する。

【注】 無料トライアル期間ではすべての機能が利用できるが、その後の Free プランでは『**同時に開発できるアプリは最大 3 個まで**』のように、利用できる機能に制限がある。このような制限は実習を進めていく上で足かせにもなるが、必要な対処策はここで紹介する。

1-2. プロジェクトの作成

Monaca では、アプリを「プロジェクト」と呼ばれる単位で扱う。

≪実習 1.2≫ 以下の手順に従って、アプリを作成せよ。〔pdf ファイル 10～16 ページ〕

① Monaca Dashboard にて、アプリを新しく作成するために、新しいプロジェクトをつくるをクリックする。

② Monaca では、用意されたテンプレート(ひな形)を用いてアプリを作成する方法なので、プロジェクトを新規に作成するときには、どれかのテンプレートを選択することになる。この実習では、必要

最小限の環境だけが揃った「最小限のテンプレート」を用いる。

③ プロジェクト名を付け、必要に応じてそのアプリの説明も付けて、作　成をクリックする。pdf ファイルでは、「はじめてのプログラム」というプロジェクト名で、説明は特につけていない。これらのプロジェクト名や説明はいつでも変更できる。

④ Monaca Dashboard の左側のプロジェクト一覧の上部に新たに作成したアプリプロジェクトが表示されるので、その箇所をクリックする。右側に現れるウインドウ内のクラウド IDE で開くをクリックすると、Monaca のクラウド IDE(Monaca IDE)が開く。

⑤ pdf ファイル 13～16 ページの指示に従って、Monaca IDE のコードエディタ内でコードを編集する。

【注】 Monaca IDE では、ファイルを保存したときにプレビュー画面にて動作を確認できるので、ファイルを変更したら必ず保存すること。コードエディタ内のファイルの保存には、Monaca IDE のメニューの「ファイル」の[保存]または[すべて保存]を用いる。正しく保存されると、Monaca IDE 画面の上部に、ファイルが保存されましたという表示の後に、スマホ画面のプレビューの箇所に Browsersync: connected と表示され、動作に反映されます。間違いなどを修正した後には**ファイルの保存を忘れないように**。また、間違いなどの修正箇所が定かでないような時は、Monaca IDE のメニューの「編集」の[元に戻す]を利用して、これまでの修正を元に戻すことで、動作していた状態に戻ることがある。このように、プログラムを作成するときには、**動かない状態から作るのではなく、動く状態に戻してから改めて変更する**方法も有効である。

⑥ Monaca IDE のプレビュー画面にて、正常動作を確認する。正常に動作していない時は、必要箇所を修正する。

【注】 Monaca はクラウド上のサービスなので、作成したファイルなどはすべてクラウド上にある。そのため、アプリプログラムを手元に置くなど、**作成したアプリを保管するには後述の処理が必要**になる。

≪実習 1.3≫ 以下の手順に従って、アプリを作成せよ。

≪実習 1.2≫で作成したアプリにおいて、次のように変更せよ。

① 11 行目の JavaScript 関数 alert()の引数(ひきすう)の文字列を、自分の名前に変更する。

② 15 行目の「はじめてのプログラム」の箇所を、自分が属する大学名に変更する。

1-3. Monaca デバッガーの利用

Monaca では、アプリをブラウザ上のクラウド IDE(Monaca IDE)で作成するだけでなく、手持ちのスマートフォンでその動作を確認することもできる。この動作確認のためのアプリが「Monaca デバッガー」である。

≪実習 1.4≫ 以下の手順に従って、Monaca デバッガーのインストールと動作確認をせよ。〔pdf ファイル 17～18 ページ〕

① pdf ファイル 17 ページの指示に従って、「Monaca デバッガー」を自分のスマートフォンにインストールする。

② 「Monaca デバッガー」を起動して、先に Monaca に登録したメールアドレスとパスワードでログインし、上で作成したアプリ「はじめてのプログラム」を実行する。

【注】 クラウドIDE(Monaca IDE)とMonacaデバッガーは同期しているが、手動で同期させたい時はMonacaデバッガーのメニューの更新をクリックする。

1-4. 授業用フォルダの準備

授業において配布される「AppDevPrac.zip」を授業用USBやローカルPCの任意の場所に展開する。このzipファイルを展開すると、AppDevPracフォルダの中にsampleサブフォルダがある。AppDevPracフォルダを授業用フォルダとし、授業で用いるアプリのサンプルコードは s サブフォルダ内に保存し、AppDevPracフォルダ直下には、授業で作成したアプリの中核フォルダのzip形式ファイルを保管する。

1-5. アプリ(プロジェクト)の管理

Monacaにおけるアプリ(プロジェクト)の管理には、以下の手法が用意されている。

(a) アプリ(プロジェクト)全体を、新規に作成する。
(b) アプリ(プロジェクト)全体を、アーカイブ(保管庫)に移動する。アーカイブ先はクラウドになる。
(c) アプリ(プロジェクト)全体を、アーカイブ(保管庫)から取り出して編集する。
(d) アプリ(プロジェクト)全体を、zip 形式でエクスポート(クラウドからダウンロード)する。ダウンロード先は、ローカルのPCになる。
(e) アプリ(プロジェクト)全体を、zip 形式でインポート(クラウドにアップロード)する。アップロード元は、ローカルのPCでもネット上の任意のサイトでも構わない。
(f) アプリ(プロジェクト)の一部のフォルダを、zip 形式でエクスポート(クラウドからダウンロード)する。ダウンロード先は、ローカルのPCになる。
(g) アプリ(プロジェクト)の一部のフォルダに、ファイルをアップロードする。アップロード元は、ローカルのPCになる。
(h) アプリ(プロジェクト)全体を、削除する。

有料プランではこれらの機能のすべてを利用できるが、Free **プランでは(b),(c)と(d)が利用できない**。これに加えて、Free **プランで同時開発できるアプリ(プロジェクト)が3個に制限されている**ので、作成したアプリ(プロジェクト)を管理し、再利用する手法が必要になる。そこで、**この実習では(a),(e),(f),(g)と(h)の機能を用いて**アプリ(プロジェクト)を管理**する**。

【注】 本来のアプリ(プロジェクト)の保管と再利用では、手法(b)と(c)または手法(d)と(e)を用いる。アプリ(プロジェクト)の保管と再利用に、手法(f)と(g)を用いるのはこの実習だけであることに注意する。

(a) アプリ(プロジェクト)全体を、新規に作成する。
Monaca Dashboard から新しいプロジェクトをつくるをクリックして、使用するテンプレートを選択する。この実習では、Freeプランでのアプリの保管と再利用に対応するために、「最小限のテンプレート」を利用する。

(e) アプリ(プロジェクト)全体を、インポートする。
ローカルのPCやネット上に保管されたzip形式アプリ(プロジェクト)全体を、Monaca Dashboardにアップロードすることができる。1.1節で示したpdfファイルには、ネット上のzip形式のアプリをイン

ポートする方法が紹介されているが、この実習では毎回の課題の最初に、**授業用フォルダ AppDevPrac 内のサンプルコードサブフォルダ sample に保管された zip 形式のアプリ全体をインポートして実習を始める。**

≪実習 1.5≫ 以下の手順に従って、zip 形式アプリ(プロジェクト)をインポートせよ。

① sample フォルダ内に、「import_test.zip」が存在することを確認する。
② Monaca Dashboard 左側上部の[インポート]をクリックする。
③ 切り替わる画面において、「[1]インポート方法」では[ZIP ファイル]をクリックする。
④ 次の「[2]ZIP ファイル」では、[Choose a file]をクリックして、sample フォルダ内の「import_test.zip」を選択して、[開く]をクリックする。
⑤ 次の「3 プロジェクトの情報」においては、プロジェクト名を「importTest」とし、説明には入力せずに、[プロジェクトのインポート]をクリックする。
⑥ Monaca Dashboard において、インポートしたプロジェクトを選択して、[クラウド IDE で開く]をクリックし、Monaca IDE でその動作を確認する。

(f) アプリ(プロジェクト)の一部のフォルダを、zip 形式でエクスポートする。

Free プランでは、アプリ(プロジェクト)の全体のエクスポートが利用できないので、アプリ(プロジェクト)の中核フォルダ(www)を次の手順でエクスポート(zip 形式)する。

≪実習 1.6≫ 以下の手順に従って、zip 形式アプリ(プロジェクト)をインポートせよ。

① Monaca IDE にて、保管したいアプリを表示しておく。ここでは、≪実習 1.3≫で修正した「はじめてのプログラム」を利用する。
② Monaca IDE の左側のプロジェクトパネルにおいて、www フォルダを選択して、右クリックする。
③ 現れたプルダウンメニューの最下行の「フォルダーをエクスポート」をクリックすると、ダウンロードが開始される。ブラウザ Chrome では、ブラウザ画面の一番下にダウンロードされた zip 形式のファイルが表示され、ファイル名の右側に表示されるマーク[^]をクリックして「フォルダを開く」をクリックすると、ダウンロード先(Windows10 では通常、ユーザフォルダ内の「ダウンロード」フォルダ)が開く。
④ ダウンロードしたファイル(「dir」で始まる zip ファイル)を、授業用フォルダ AppDevPrac 内に移動させ、その場所で zip ファイルを展開する。
⑤ 展開すると、「www」フォルダが生成されるので、そのフォルダ名を変更する。例えば、プロジェクト名とする。ここでは、「はじめてのプログラム」としておく。その後、ダウンロードされた zip ファイルは削除して構わない。

【注】 この手法でエクスポートすると、すべて「dir」で始まるファイル名の zip ファイルになり、それを展開すると「www」フォルダになる。ここで、フォルダ名を変更しなければ、次に「フォルダーをエクスポート」したり展開したりしたときに、上書きになってしまう。そのために、展開して生成されたフォルダ名を変更しておく。

(g) アプリ(プロジェクト)の一部のフォルダに、ファイルをアップロードする。

この機能は本来、アプリに必要な画像や音声ファイルなどをアップロードするためにあるが、実習で用いる Free プランでは、アプリ(プロジェクト)全体のエクスポートができないので、それのインポートもできない。そこで、(f)の手法でエクスポートしたアプリ(プロジェクト)の中核フォルダ(www)のアップロードに、この機能を流用する。この機能により、以前作成したアプリプログラムを再利用することができる。

≪実習 1.7≫ 以下の手順に従って、≪実習 1.6≫でエクスポートしたフォルダ内をアップロードせよ。

この手法でアプリを復元できるのは、**単純な機能だけのアプリに限られる**。特に、「最小限のテンプレート」を用いて作成するこの実習のアプリでは、この手法でもアプリを復元できる。

① AppDevPrac フォルダ内の復元したいアプリの中核フォルダ(アプリプロジェクト名に改名された www フォルダ)をエクスプローラなどで開く。ここでは、≪実習 1.6≫で作成した「はじめてのプログラム」フォルダを開く。このときフォルダ内に components フォルダ、css フォルダと index.html ファイルなどが存在することを確認する。

② Monaca Dashboard において、最小限のテンプレートを用いて、任意の名前の新しいプロジェクトを作成する。ここでは、プロジェクト名を「firstApp」とする。

③ 作成したプロジェクトを Monaca IDE で開き、左側のプロジェクトパネルにおいて、www フォルダの前が▼マークになって、www フォルダ内が表示されていることを確認する。www フォルダ内には、components フォルダ、css フォルダと index.html ファイルがある。

【注】 index.html の下に表示される.gitignore から package.json までのファイルは、このプロジェクトのルートフォルダに存在するファイルで、www フォルダ内ではない事に注意する。

④ www フォルダ内の components フォルダ、css フォルダと index.html ファイルをすべて削除する。そのために、削除するフォルダやファイルを右クリックで選択して、現れるプルダウンメニューから[削除]を選択する。その後表示される確認ダイアログでは OK をクリックする。

⑤ www フォルダ内の２つのフォルダと１つのファイルを削除したら、www フォルダを右クリックで選択して、現れたプルダウンメニューで[ファイルをアップロード]を選択する。

⑥ ここで現れる「ファイルをアップロード」ダイアログ内の[ファイルをドロップしてください]の箇所に、①で開いていたフォルダ内のすべてのフォルダとファイルをドラッグ＆ドロップする。「ファイルをアップロード」ダイアログでは、アップロードしたフォルダやファイルは確認できないが、Monaca IDE のプロジェクトパネル内ではアップロードされたフォルダやファイルが表示される。すべてのフォルダとファイルをアップロードされたら、「ファイルをアップロード」ダイアログの右上隅の×をクリックして、ダイアログを閉じる。

⑦ この時点で、Monaca IDE のプレビュー画面にはアップロードされたアプリの動作結果が表示されるハズ。また、アプリの index.html をコードエディタに表示する時は、プロジェクトパネル内の index.html の箇所をダブルクリックする。

(h) アプリ(プロジェクト)全体を、削除する。

Free プランでのアプリ開発実習を続けるために、中核フォルダを保存するなどして不要になったアプリ(プロジェクト)は削除する。アプリ(プロジェクト)全体の削除は、Monaca Dashboard において、アプリ(プロジェクト)前の□をチェックして、削除をクリックする。

≪実習 1.8≫ 以下の手順に従って、プロジェクト一覧内のアプリ(プロジェクト)を削除せよ。

① Monaca Dashboard のプロジェクト一覧内の削除したいアプリ(プロジェクト)名の先頭にある□をチェックする。ここでは、「はじめてのプログラム」プロジェクトを選択する。

② このときに、プロジェクト一覧の上に表示される項目から削除をクリックする。

③ 現れる確認ダイアログで OK をクリックすると、目的のアプリ(プロジェクト)が削除される。

≪実習 1.9≫ 現時点で、Monaca Dashboard のプロジェクト一覧に残っているすべてのアプリ(プロジェクト)を削除せよ。

1-6. 参照サイト

この実習テキストでは、HTML5、CSS や JavaScript の詳細な説明を省いているが、実習内の課題を解く上で、HTML5、CSS や JavaScript に対する理解が必須となる。そこで、これらに対する知識が必要になったときに、参照できるサイトを以下に紹介する。

(a) HTML5

HTML 入門：http://html5.imedia-web.net/

(b) CSS (CSS3)

CSS 入門：https://techacademy.jp/magazine/4872

(c) JavaScript

JavaScript 入門：https://www.pazru.net/js/

2. カレンダー①

2-1. 本実習でのポイント

本実習では、以下の項目に特に注意する。

(a) HTML5

- HTML の基本的な記述スタイル
- HTML タグ内での id プロパティ、style プロパティ
- <body>タグとその中での onLoad プロパティ
- <button>タグとその中での onClick プロパティ、size プロパティ

(b) CSS (CSS3)

- text-align プロパティとその値
- font-size プロパティとその値
- color プロパティとその値(特に、16 進での RGB 指定)

(c) JavaScript

- 変数や関数の宣言
- 関数の呼び出しと引数(ひきすう)
- JavaScript における日付(Date オブジェクト)の取り扱い
- document オブジェクトと getElementById 関数
- switch-case 文
- 文字列の連結

2-2. サンプルコードの解説と修正

◎サンプルアプリ名：calendar1.zip

スマホ画面に日めくりカレンダーを表示する

≪実習 2.1≫ サンプルアプリを calendar1 プロジェクトとしてインポートし、以下のプログラムと解説を参考に、インポートしたプログラム内の★の箇所を埋めよ。

行番号	calendar1//index.html
1:A	<!DOCTYPE HTML>
2:	<html>
3:	<head>
4:B	<meta charset="utf-8">
5:	<meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no">
6:	<meta http-equiv="Content-Security-Policy" content="default-src * data: gap: content: https://ssl.gstatic.com; style-src * 'unsafe-inline'; script-src * 'unsafe-inline' 'unsafe-eval'">
7:	<script src="components/loader.js"></script>
8:	<link rel="stylesheet" href="components/loader.css">
9:	<link rel="stylesheet" href="css/style.css">
10:C	<style>
11:	
12:	</style>

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 1：HTML5 の指定書式

B 行 4-9：アプリ開発環境 Monaca での自動挿入部分

C 行 10-12：HTML 内でのスタイル指定をまとめて記述する場合は、この場所に記述する。

```
13:A    <script>
14:B      var year;     //  表示年
15:       var month;    //  表示月
16:       var day;      //  表示日
17:       var day_week; //  表示曜日(0：日曜日、・・・、6：土曜日)
18:
19:C      function getToday() {
20:D        var today =new Date();       //  今日の日付を today に取得
21:E        year=today.getFullYear();    //  year に西暦年を設定
22:         month=today.getMonth()+1;    //  month に月(1～12)を設定
23:         day=today.getDate();         //  day に日を設定
24:         day_week=today.getDay();     //  day_week に曜日を設定
25:F        dispDate();          //  日付を画面に表示する関数の呼び出し
26:       }
27:
28:G      function dispDate() {   //  日付を画面に表示する関数
29:H        var style_sizeM="font-size: 200%;";   //  Midle サイズの大きさ
30:         var style_sizeH="font-size: 700%;";   //  Huge サイズの大きさ
31:         var style_color0="color: #FF0000;";   //  赤色
32:         var style_color4="color: #0000FF;";   //  青色
33:         var style_color7="color: #000000;";   //  黒色
34:         var textcolor="";                     //  テキスト色変数
35:I        document.getElementById("disp_year").style=style_sizeM;
36:J        document.getElementById("disp_year").innerHTML=year;
37:
38:K        document.getElementById("disp_month").style=style_sizeH;
39:         document.getElementById("disp_month").innerHTML=month;
40:
41:L        switch (day_week) {
42:           case 0: textcolor=style_color0; break;
43:           case 6: textcolor=style_color4; break;
44:           default: textcolor=style_color7;
45:         }
46:M        document.getElementById("disp_day").style=style_sizeH+textcolor;
```

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 13-80：HTML 内での JavaScript プログラムをまとめて記述する場合は、この場所に記述する。

B 行 14-17：アプリ画面に表示する年、月、日、曜日を保存する変数を準備する。

C 行 19-26：画面が表示された当日の日付を取得する関数 getToday()の具体的な処理コードを記述。

D 行 20：今日の日付を取得する。Date オブジェクト(インスタンス)を引数なしで生成すると、今日の日付が取得される。

E 行 21-24：日付変数 today の西暦年、月、日、曜日を対応する変数に設定する。ここで、月は 1～12 としている所に注意する。

【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

F 行 25：アプリ画面に日付を表示する関数 dispDate()を呼び出す。

G 行 28-50：関数 dispDate()が呼び出されたときに実行するコードの記述。

H 行 29-34：テキストのスタイル(文字サイズ、文字色)を指定する変数の設定。

I 行 35：HTML ドキュメント内で id が disp_year の要素の style プロパティを変数 style_large の値に設定する。

J 行 36：HTML ドキュメント内で id が disp_year の要素の innerHTML プロパティ(タグに挟まれた記述内容)を、変数 year の値に設定する。

K 行 38-39：HTML ドキュメント内で id が disp_month の要素の style プロパティと innerHTML プロパティ(タグに挟まれた記述内容)を、それぞれ変数 style_huge と変数 year の値に設定する。

【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

L 行 41-45：曜日に対応する変数 day_week の値(0～6)に応じて文字色スタイル変数の値を変える。

M 行 46-47：HTML ドキュメント内で id が disp_day の要素の style プロパティを、文字サイズスタイル style_huge と文字色スタイル textcolor を合わせた設定とし、innerHTML プロパティ(タグに挟まれた記述内容)を変数 day の値に設定する。

【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

```
47:          document.getElementById("disp_day").innerHTML=day;
48:A         document.getElementById("disp_day_week").style=style_sizeM+textcolor;
49:          document.getElementById("disp_day_week").innerHTML=yobi(day_week);
50:        }
51:
52:B       function yobi(dw) {
53:        var yb="";
54:C         switch (dw) {
55:          case 0: yb="日"; break;
56:          case 1: yb="月"; break;
57:          case 2: yb="火"; break;
58:          case 3: yb="水"; break;
59:          case 4: yb="木"; break;
60:          case 5: yb="金"; break;
61:          case 6: yb="土"; break;
62:          default: yb="※";
63:         }
64:D         return yb;
65:        }
66:
67:E       function udDate(udd) {
68:F         var pdate=new Date(year,month-1,day);   //  現在表示されている日付を取得
69:G         pdate.setDate(pdate.getDate()+udd);     //  表示されている日付に udd を加減して設定
70:H         year=pdate.getFullYear();   //  year に更新日付の西暦年を設定
71:          month=pdate.getMonth()+1;   //  month に更新日付の月(1～12)を設定
72:          day=pdate.getDate();        //  day に更新日付の日を設定
73:          day_week=pdate.getDay();    //  day_week に更新日付の曜日を設定
74:I         dispDate();          //  日付を画面に表示する関数の呼び出し
75:        }
76:     </script>
77:  </head>
78:
79:J <body onLoad="getToday()">
80:K <!-- 日付表示エリア -->
81:     今日は、<br />
82:L    <div style="text-align: center;">
```

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 48-49：HTML ドキュメント内で id が disp_day_week の要素の style プロパティを、文字サイズスタイル style_large と文字色スタイル textcolor を合わせた設定とし、innerHTML プロパティ(タグに挟まれた記述内容)には曜日に対応する変数 day_week の値を引数とした関数 yobi()の戻り値を設定する。

【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

B 行 52-65：一つの引数を持つ関数 yobi()が呼び出されたときに実行するコードの記述。

C 行 54-63：引数 dw の値に応じて、変数 yb の値を変更する。

D 行 64：関数 yobi()の戻り値に変数 yb の値を設定する。

E 行 67-75：日付を更新する関数 udDate()の具体的な処理コードを記述。

F 行 68：現在表示されている日付(year, month, day)を基に、日付オブジェクト(インスタンス)を生成する。

G 行 69：上で設定された日付に、更新したい値を格納した変数 udd を加算する。これによって、日付が更新される。

H 行 70-73：更新された日付から西暦年、月、日、曜日を対応する変数に設定する。ここで、月は 1～12 としている所に注意する。

【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

I 行 74：アプリ画面に更新された日付を表示する関数 dispDate()を呼び出す。

J 行 79-98：アプリ画面に実際に表示する内容を記述する。79 行目の body タグにおいて、onLoad プロパティに getToday()を設定しているので、この body タグ内が読み込まれたときに、同時に画面表示当日の日付を取得する関数 getToday()が呼び出される。

K 行 80-88：日付表示エリアの記述として、コメントで挟んでいる。

L 行 82-87：具体的な日付を表示する箇所を div タグで一要素としてまとめ、その要素に style プロパティとし

```
83:A        <span id="disp_year" style="">YYYY</span> 年<br />
84:         <span id="disp_month" style="">MM</span> 月
85:         <span id="disp_day" style="">DD</span> 日<br />
86:         <span id="disp_day_week" style="">WD</span> 曜日<br />
87:       </div>
88:   <!-- ／日付表示エリア -->
89:     <br />
90:B  <!-- 日付更新ボタン -->
91:C      <div style="text-align: center;">
92:D        <button id="prev_btn" onClick="udDate(-1)" size="4">＜前日</button>
93:         <button id="today_btn" onClick="getToday()" size="4">今日</button>
94:         <button id="next_btn" onClick="udDate(1)" size="4">翌日＞</button>
95:       </div>
96:   <!-- ／日付更新ボタン -->
97:
98:   </body>
99:   </html>
```

【注】 どうしても正常動作に至らない場合は、Ans_www フォルダ内の Ans_index.html を参考にせよ。

≪実習 2.2≫ サンプルアプリの www フォルダをエクスポートして、授業用フォルダに保管し、フォルダ名を calendar1 とせよ。

≪実習 2.3≫ サンプルアプリの index.html の 31-35 行目で CSS のプロパティ値をそれぞれ設定しているが、各プロパティの解釈を復習して、それぞれの値を好きなように変更し、その影響を確認せよ。

≪実習 2.4≫ サンプルアプリの index.html の 44-45 行目は switch-case 文での case の 2 種類の値に対する処理を記述している。これらの行で指定される case 値を変更して、その違いを確認せよ。この確認には、95-97 行目で表示されるボタンが役に立つ。

≪実習 2.5≫ サンプルアプリの index.html の 43-47 行目の switch-case 文を、同じ挙動をするように、else-if 文を用いた構文で書き換えよ。挙動の確認には、95-97 行目で表示されるボタンが役に立つ。

≪実習 2.6≫ これまでに変更してきたアプリの www フォルダをエクスポートして、授業用フォルダに保管し、フォルダ名を calendar1ex とせよ。

※※※※※　プログラムの解説・修正問題　※※※※※

て、「text-align: center;」と設定して、要素全体を画面の中央に配置している。

A 行 83-86：年、月、日、曜日を表示する箇所を span タグで区分けし、それぞれに id を設定して、style プロパティを用意している。それぞれの style プロパティの内容は、JavaScript によって、変更される。また、YYYY、MM、DD、WD なども各 id で抽出された要素の innerHTML プロパティの書き換えによって変更される。

B 行 90-96：日付更新ボタンの表示エリアの記述として、コメントで挟んでいる。

C 行 91-95：具体的な日付移動(更新)ボタンを表示する箇所を div タグで一要素としてまとめ、その要素に style プロパティとして、「text-align: center;」と設定して、要素全体を画面の中央に配置している。

D 行 92-94：日付更新ボタンのクリック(タップ)に応じて、JavaScript の関数の呼び出し方を変える。

【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

ヒント：行 92,94 は日付を更新する関数 udDate()を呼び出すが、その時の引数は「-1」または「1」とする。また、行 93 は画面表示当日の日付を取得する関数 getToday()を引数なしで呼び出す。

3. カレンダー②

3-1. 本実習でのポイント

本実習では、以下の項目に特に注意する。

(d) HTML5

- HTML の基本的な記述スタイル
- HTML タグ内での id プロパティ、class プロパティ、style プロパティ
- <body>タグとその中での onLoad プロパティ
- <button>タグとその中での onClick プロパティ、size プロパティ

(e) CSS (CSS3)

- text-align プロパティとその値
- font-size プロパティとその値
- color プロパティとその値(特に、16 進での RGB 指定)

(f) JavaScript

- 変数や関数の宣言
- 関数の呼び出しと引数(ひきすう)
- JavaScript における日付(Date オブジェクト)の取り扱い
- document オブジェクトと getElementById 関数
- switch-case 文
- 文字列の連結

3-2. サンプルコードの解説と修正

◎サンプルアプリ名：calendar2.zip

スマホ画面に月めくりカレンダーを表示する

≪実習 3.1≫ サンプルアプリを calendar2 プロジェクトとしてインポートし、以下のプログラムと解説を参考に、インポートしたプログラム内の★の箇所を埋めよ。

行番号	calendar2//index.html
1:	<!DOCTYPE HTML>
2:	<html>
3:	<head>
4:	<meta charset="utf-8">
5:	<meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no">
6:	<meta http-equiv="Content-Security-Policy" content="default-src * data: gap: content: https://ssl.gstatic.com; style-src * 'unsafe-inline'; script-src * 'unsafe-inline' 'unsafe-eval'">
7:	<script src="components/loader.js"></script>
8:	<link rel="stylesheet" href="components/loader.css">
9:	<link rel="stylesheet" href="css/style.css">
10: A	<style>
11:	.sizeM {
12: B	font-size: 200%;

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 10-30：この HTML ファイル内のスタイル設定。セレクタの先頭に「.」(ドット)があるので、ここでのセレクタはすべて class 属性で指定される箇所のスタイル指定である。

B 行 12：class 属性「sizeM」で指定される個所のフォントサイズの指定(200%)である。

【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

```
13:         }
14:
15:         .sun {
16:           width: 35px;
17:A          color: red;
18:         }
19:
20:         .week_day {
21:           width: 35px;
22:B          color: black;
23:         }
24:
25:         .sat {
26:           width: 35px;
27:C          color: blue;
28:         }
29:
30:       </style>
31:D      <script>
32:E        var year;     //  表示年
33:         var month;    //  表示月
34:         var day_week; //  表示曜日(0：日曜日、・・・、6：土曜日)
35:
36:F        function getThisMonth() {
37:           var today =new Date();       //  今日の日付を today に取得
38:           year=today.getFullYear();     //  year に今年の西暦年を仮設定
39:           month=today.getMonth()+1;     //  month に今月の月(1～12)を仮設定
40:G          var pdate =new Date(year,month-1,1);   //  今月の初日の日付を取得
41:           year=pdate.getFullYear();    //  year に今年の西暦年を設定
42:           month=pdate.getMonth()+1;    //  month に今月の月(1～12)を設定
43:           day_week=pdate.getDay();     //  day_week に今月の初日の曜日を設定
44:H          dispMonthCalender();         //  月カレンダーを画面に表示する関数の呼び出し
45:         }
46:
47:         function setMonth() {
48:           var y=parseInt(document.getElementById("set_year").value);
49:           var m=parseInt(document.getElementById("set_month").value);
50:
51:I          var pdate =new Date(y,m-1,1);
```

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 17：class 属性「sun」で指定される個所の文字色の指定(赤：red または #FF0000)である。
　　【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

B 行 22：class 属性「week_day」で指定される個所の文字色の指定(黒：brack または #000000)である。
　　【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

C 行 27：class 属性「sat」で指定される個所の文字色の指定(青：blue または #0000FF)である。
　　【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

D 行 31-102：この HTML ファイル内で利用する JavaScript のコードをまとめて記述している。

E 行 32-34：アプリ画面に表示するカレンダーの年、月、および月初めの曜日を保存する変数を準備する。

F 行 36-45：画面が表示された日の当月を取得し、その月の初日の日付を取得する関数 getThisMonth()の具体的な処理コードを記述。

G 行 40-43：当月の初日(1 日)の日付を pdate に取得し、その西暦年、月、初日の曜日をそれぞれ year、month、day_week に設定する。[参考：検索キーワード「Javascript　日付」]
　　【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

H 行 44：月カレンダーを画面に表示する関数 dispMonthCalendar()を呼び出す。
　　【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

I 行 51-54：指定月の初日(1 日)の日付を pdate に取得し、その西暦年、月、初日の曜日をそれぞれ year、month、day_week に設定する。[参考：検索キーワード「Javascript　日付」]
　　【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

```
52:          year=pdate.getFullYear();    //  year に設定月の西暦年を設定
53:          month=pdate.getMonth()+1;    //  month に設定月の月(1～12)を設定
54:          day_week=pdate.getDay();     //  day_week に設定月の初日の曜日を設定
55:A         dispMonthCalender();         //  月カレンダーを画面に表示する関数の呼び出し
56:        }
57:
58:B       function dispMonthCalender() {
59:C         clearCalendar();
60:D         document.getElementById("disp_year").innerHTML=year;
61:          document.getElementById("disp_month").innerHTML=month;
62:E         var posid="";
63:F         var daysM=ndM();
64:G         for (var day=1; day<=daysM; day++) {
65:H           posid="d"+(100+day+day_week);
66:            document.getElementById(posid).innerHTML=day;
67:          }
68:        }
69:
70:I       function clearCalendar() {
71:          var posid="";
72:          for (var i=101; i<=142; i++) {
73:            posid="d"+i;
74:            document.getElementById(posid).innerHTML="";
75:          }
76:        }
77:
78:J       function ndM() {
79:K         var ds;
80:L         switch(month) {
81:M           case 2: ds=28;
82:N                   if ((year%400==0)||(year%4==0 && year%100!=0))
83:                      ds=29;
84:                   break;
85:O           case 4: case 6: case 9: case 11:
```

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 55：月カレンダーを画面に表示する関数 dispMonthCalendar()を呼び出す。
【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

B 行 58-68：月のカレンダーを表示する関数 dispMonthCalendar()の具体的な処理コードを記述。

C 行 59：月カレンダー表示エリアを初期化する関数 clearCalendar()を呼び出す。

D 行 60-61：HTML ドキュメント内で id が disp_year と disp_month の要素の innerHTML プロパティ(タグに挟まれた記述内容)をそれぞれ変数 year と変数 month の値に設定する。
【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

E 行 62：月カレンダー内の日の表示場所を特定するための変数を用意する。

F 行 63：月の日数を戻り値とする関数 ndM()を呼び出し、その戻り値(日数)を daysM に設定する。

G 行 64-67：月カレンダー内の適切な曜日の場所に、月内の日をすべて表示する。

H 行 65：月内の日と表示すべき適切な曜日の場所を設定する。例えば、初日(1 日)の曜日は day_week にある。ここで、「String()」は()の数を文字列に変換する関数である。
【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

I 行 70-78：月カレンダー表示エリアの初期化関数 clearCalendar()の具体的な処理コードの記述。

J 行 78-90：表示月の日数を戻り値とする関数 ndM()の具体的な処理コードの記述。

K 行 79：戻り値の月の日数を保存する変数 ds を用意する。

L 行 80-88：表示月(変数 month)の値(1～12)によって、月の日数 ds を変更するための switch-case 文。

M 行 81-84：2 月の場合の日数 ds を設定する処理。原則として 28 だが、うるう年の場合には、29 にしなければならない。

N 行 82-83：うるう年の判定に対応する if 文。うるう年は、「400 で割り切れる年、または 4 で割り切れても 100 で割り切れない年」となる。
【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

O 行 85-86：4,6.9.11 月の日数を設定する。

```
 86:                     ds=30; break;
 87:A          default: ds=31;
 88:         }
 89:B        return ds;
 90:       }
 91:
 92:C       function udMonth(udd) {
 93:         var pdate=new Date(year,month-1,1);      //  現在の表示月の初日の日付を取得
 94:         pdate.setMonth(pdate.getMonth()+udd);    //  月に udd を加減して日付を設定
 95:D        year=pdate.getFullYear();    //  year に更新月の西暦年を設定
 96:         month=pdate.getMonth()+1;    //  month に更新月の月(1～12)を設定
 97:         day_week=pdate.getDay();     //  day_week に更新月の初日の曜日を設定
 98:E        dispMonthCalender();         //  月カレンダーを画面に表示する関数の呼び出し
 99:       }
100:
101:    </script>
102:  </head>
103:F <body onLoad="getThisMonth()">
104:G <!-- 日付指定エリア：動作時の取得日付の変更用 -->
105:    <input id="set_year" type="text" min="2000" max="2050" size="3">年
106:    <input id="set_month" type="text" min="1" max="12" size="2">月
107:H   <button id="set_date" onClick="setMonth()" size="4">表示月設定</button>
108:  <!-- ／日付指定エリア：動作時の取得日付の変更用 -->
109:  <hr>
110:  <!-- 月カレンダー表示エリア -->
111:     <div style="text-align: center;">
112:       <span id="disp_year" class="sizeM" style="color: brown;">YYYY</span> 年
113:       <span id="disp_month" class="sizeM" style="color: violet;">MM</span> 月<br />
114:       <table style="display:table; margin: 0 auto;">
115:         <tr> <!-- 曜日行 -->
116:           <th class="sun">日</th>
117:           <th class="week_day">月</th>
118:           <th class="week_day">火</th>
119:           <th class="week_day">水</th>
120:           <th class="week_day">木</th>
121:           <th class="week_day">金</th>
122:           <th class="sat">土</th>
123:         </tr>
124:         <tr> <!-- 第１週 -->
125:I          <td id="d101" class="sun">1</td>
126:           <td id="d102" class="week_day">2</td>
127:           <td id="d103" class="week_day">3</td>
```

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 87：それ以外の月の日数を設定する。

B 行 89：変数 ds の値をこの関数の戻り値として、呼び出し元に返す。

C 行 92-99：カレンダーに表示する月を更新する関数 udMonth()の具体的な処理コードを記述。

D 行 95-97：pdate の西暦年、月、初日の曜日をそれぞれ year、month、day_week に設定する。

【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

E 行 98：月カレンダーを画面に表示する関数 dispMonthCalendar()を呼び出す。

【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

F 行 103-197：アプリ画面に実際に表示する内容を記述する。103 行目の body タグにおいて、onLoad プロパティに getThisMonth()を設定しているので、この body タグ内が読み込まれたときに、同時に画面表示当日を含む月を取得する関数 getThisMonth()が呼び出される。

G 行 104-108：表示月を指定するためのテキストボックスを表示する箇所。

H 行 107：表示月設定のボタンのクリック(タップ)に応じて、JavaScript の関数を呼び出す。

【修正問題】　インポートプログラム内の★の箇所に適切なコードを記述せよ。

ヒント：表示月設定ボタンのクリックでは、表示月を設定する関数 setMonth()を引数なしで呼び出す。

I 行 125-131：表内のセルを意味する td タグ内に、id プロパティと class プロパティがある。id プロパティと class プロパティの使い分けに注意する。

```
128:            <td id="d104" class="week_day">4</td>
129:            <td id="d105" class="week_day">5</td>
130:            <td id="d106" class="week_day">6</td>
131:            <td id="d107" class="sat">7</td>
132:          </tr>
133:          <tr> <!-- 第2週 -->
134:            <td id="d108" class="sun">8</td>
135:            <td id="d109" class="week_day">9</td>
136:            <td id="d110" class="week_day">10</td>
137:            <td id="d111" class="week_day">11</td>
138:            <td id="d112" class="week_day">12</td>
139:            <td id="d113" class="week_day">13</td>
140:            <td id="d114" class="sat">14</td>
141:          </tr>
142:          <tr> <!-- 第3週 -->
143:            <td id="d115" class="sun">15</td>
144:            <td id="d116" class="week_day">16</td>
145:            <td id="d117" class="week_day">17</td>
146:            <td id="d118" class="week_day">18</td>
147:            <td id="d119" class="week_day">19</td>
148:            <td id="d120" class="week_day">20</td>
149:            <td id="d121" class="sat">21</td>
150:          </tr>
151:          <tr> <!-- 第4週 -->
152:            <td id="d122" class="sun">22</td>
153:            <td id="d123" class="week_day">23</td>
154:            <td id="d124" class="week_day">24</td>
155:            <td id="d125" class="week_day">25</td>
156:            <td id="d126" class="week_day">26</td>
157:            <td id="d127" class="week_day">27</td>
158:            <td id="d128" class="sat">28</td>
159:          </tr>
160:          <tr> <!-- 第5週 -->
161:            <td id="d129" class="sun">29</td>
162:            <td id="d130" class="week_day">30</td>
163:            <td id="d131" class="week_day">31</td>
164:            <td id="d132" class="week_day">32</td>
165:            <td id="d133" class="week_day">33</td>
166:            <td id="d134" class="week_day">34</td>
167:            <td id="d135" class="sat">35</td>
168:          </tr>
169:A         <tr> <!-- 第7週 -->
170:            <td id="d136" class="sun">36</td>
171:            <td id="d137" class="week_day">37</td>
172:            <td id="d138" class="week_day">38</td>
173:            <td id="d139" class="week_day">39</td>
174:            <td id="d140" class="week_day">40</td>
175:            <td id="d141" class="week_day">41</td>
176:            <td id="d142" class="sat">42</td>
177:          </tr>
178:        </table>
179:      </div>
180:  <!-- ／月カレンダー表示エリア -->
181:  <!-- 表示月更新ボタン -->
182:      <table style="display:table; margin: 0 auto;">
183:        <tr>
184:          <td style="width: 70px; text-align: left;">
              <button id="prev_btn" onClick="udMonth(-1)" size="4">< 前月</button>
```

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 169-177：月の初日が週末の時に、月末が第6週になることもあるので、そのために用意する。

```
185:A          </td>
186:           <td style="width: 105px; text-align: center;">
187:             <button id="today_btn" onClick="getThisMonth()" size="4">今月</button>
188:           </td>
189:B          <td style="width: 70px; text-align: right;">
190:             <button id="next_btn" onClick="udMonth(1)" size="4">翌月 ></button>
191:C          </td>
192:         </tr>
193:       </table>
194:   <!-- ／表示月更新ボタン -->
195:   </body>
196:   </html>
197:
```

【注】 どうしても正常動作に至らない場合は、Ans_www フォルダ内の Ans_index.html を参考にせよ。

≪実習 3.2≫ サンプルアプリの www フォルダをエクスポートして、授業用フォルダに保管し、フォルダ名を calendar2 とせよ。

≪実習 3.3≫ サンプルアプリの index.html の 15-28 行目で CSS のプロパティ値をそれぞれ設定しているが、プロパティ(幅、色)の解釈を復習して、それぞれの値を好きなように変更し、その影響を確認せよ。

≪実習 3.4≫ サンプルアプリの index.html の 186, 192 行目の OnClick で関数を呼び出すときに引数をそれぞれ設定しているが、その引数の値を変更して、その影響を確認せよ。

≪実習 3.5≫ これまでに変更してきたアプリの www フォルダをエクスポートして、授業用フォルダに保管し、フォルダ名を calendar2ex とせよ。

≪実習 3.6≫ Monaca Dashboard より、calendar1 と calendar2 の 2 つのプロジェクトを削除せよ。

※※※※※ プログラムの解説・修正問題 ※※※※※

A 行 185：< 前月のボタンのクリック(タップ)に応じて、JavaScript の関数を呼び出す。
【修正問題】 インポートプログラム内の★の箇所に適切なコードを記述せよ。
ヒント：< 前月のボタンのクリックでは、表示月を更新して表示する関数 udMonth()を引数「-1」で呼び出す。

B 行 189：今月のボタンのクリック(タップ)に応じて、JavaScript の関数を呼び出す。
【修正問題】 インポートプログラム内の★の箇所に適切なコードを記述せよ。
ヒント：今月のボタンのクリックでは、現在の月を取得して表示する関数 getThisMonth()を引数なしで呼び出す。

C 行 191：翌月 >のボタンのクリック(タップ)に応じて、JavaScript の関数を呼び出す。
【修正問題】 インポートプログラム内の★の箇所に適切なコードを記述せよ。
ヒント：翌月 >のボタンのクリックでは、表示月を更新して表示する関数 udMonth()を引数「+1」で呼び出す。

4. サイコロ実験

4-1. 本実習でのポイント

本実習では、以下の項目に特に注意する。

(g) HTML5

- HTML の基本的な記述スタイル
- HTML タグ内での id プロパティ、style プロパティ
- <button>タグとその中での onClick プロパティ、size プロパティ

(h) CSS (CSS3)

- color プロパティとその値(特に、16 進での RGB 指定)
- background-color プロパティとその値

(i) JavaScript

- 文字列の整数値化
- 数ではない(Not a Number)かどうかの判定
- 乱数の生成とその整数化(小数点以下切り捨て)
- switch-case 文、文字列の連結
- 四捨五入関数と実数の小数点以下のけた指定
- JavaScript における日付(Date オブジェクト)の取り扱い
- document オブジェクトと getElementById 関数

4-2. サンプルコードの解説と修正

◎サンプルアプリ名：saikoro.zip

スマホ画面にサイコロを振って、出た目の回数や確率を表示する。

≪実習 4.1≫ サンプルアプリを saikoro プロジェクトとしてインポートし、以下のプログラムと解説を参考に、インポートしたプログラム内の★の箇所を埋めよ。

行番号	saikoro//index.html
1:	<!DOCTYPE HTML>
2:	<html>
3:	<head>
4:	<meta charset="utf-8">
5:	<meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no">
6:	<meta http-equiv="Content-Security-Policy" content="default-src * data: gap: content: https://ssl.gstatic.com; style-src * 'unsafe-inline'; script-src * 'unsafe-inline' 'unsafe-eval'">
7:	<script src="components/loader.js"></script>
8:	<link rel="stylesheet" href="components/loader.css">
9:	<link rel="stylesheet" href="css/style.css">
10:	<script>
11:A	function startTry() {
12:	var tryN; // 設定試行回数
13:	var try_n; // 試行回数の記録変数
14:	var c1,c2,c3,c4,c5,c6; // 出現回数の記録変数
15:	var dice; // 試行でのサイコロの目
16:	

※※※※※ プログラムの解説・修正問題 ※※※※※

A 行 11-54：サイコロを振る試行を指定回数繰り返し、出た目の回数とそれぞれの確率を計算し、それらの値を表示する関数 startTry()のコードを記述。

```
17:A          tryN=parseInt(document.getElementById("tryN").value);
18:B          if (isNaN(tryN)) {
19:C            document.getElementById("tryN").style="background-color: red;";
20:D            return;
21:           }
22:E          document.getElementById("tryN").style="background-color: white;";
23:F          document.getElementById("try_num").innerHTML=tryN;
24:
25:           try_n=0;
26:           c1=c2=c3=c4=c5=c6=0;
27:
28:           for (var i=1; i<=tryN; i++) {
29:G            dice=Math.floor(Math.random()*6)+1;
30:H            switch (dice) {
31:I              case 1: c1++; break;
32:               case 2: c2++; break;
33:               case 3: c3++; break;
34:               case 4: c4++; break;
35:               case 5: c5++; break;
36:               case 6: c6++; break;
37:               default:
38:             }
39:           }
40:
41:         //  それぞれの目の出現回数を表示する
42:J          document.getElementById("ap1").innerHTML=c1;
43:          document.getElementById("ap2").innerHTML=c2;
44:          document.getElementById("ap3").innerHTML=c3;
45:          document.getElementById("ap4").innerHTML=c4;
46:          document.getElementById("ap5").innerHTML=c5;
47:          document.getElementById("ap6").innerHTML=c6;
```

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 17：id「tryN」で指定されたテキストボックスの値(文字列)を整数値として変数 tryN に取得する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

B 行 18：変数 tryN に取得された値が「数でない(Not a Number)」かどうかを判定する。この判定で『true』になったときは、19-21 行に記述された処理を行う。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

C 行 19：id「tryN」で指定されたテキストボックスの style プロパティで、背景色を赤色に指定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

D 行 20：「return」文によって、この関数の処理を終了する。

E 行 22：id「tryN」で指定されたテキストボックスの style プロパティで、背景色を白色に指定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

F 行 23：id「try_num」で指定されたタグで囲まれた箇所の内側(inner)の記述(HTML 文)を変数 tryN の値に設定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

G 行 29：まず、Math ライブラリ内の random()関数によって、0 以上 1 未満の乱数を生成し、それを？倍して 0 以上 6 未満の乱数とする。次に、Math ライブラリ内の floor()関数を用いて、その乱数の小数点以下を切り捨てて、0～5 の整数にする。最後に、？を加えて、1～6 の整数にする。最終的に変数 dice に代入される値を、試行したときのサイコロの目と考える。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

H 行 30-38：サイコロの目に応じた多分処理。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

I 行 31-36：それぞれのサイコロの目に応じて、その目の記録変数を 1 ずつ更新する。ここで、演算子「++」はその変数の値を 1 更新して代入するもので、例えば、「a++」は「a=a+1」と同じ働きをする。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

J 行 42-47：サイコロのそれぞれの目の出現回数を画面上に表示する。該当する id を持つタグの箇所の内側(inner)の記述(HTML 文)を対応する回数を記録した変数の値に設定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

```
48:
49:         //  それぞれの目の出現確率を表示する
50:A          document.getElementById("prob1").innerHTML=Math.round((c1/tryN)*1000)/10;
51:          document.getElementById("prob2").innerHTML=Math.round((c2/tryN)*1000)/10;
52:          document.getElementById("prob3").innerHTML=Math.round((c3/tryN)*1000)/10;
53:          document.getElementById("prob4").innerHTML=Math.round((c4/tryN)*1000)/10;
54:          document.getElementById("prob5").innerHTML=Math.round((c5/tryN)*1000)/10;
55:          document.getElementById("prob6").innerHTML=Math.round((c6/tryN)*1000)/10;
56:        }
57:      </script>
58:  </head>
59:  <body>
60:    <h1 style="color: blue;">サイコロ試行実験</h1>
61:    試行回数：<input id="tryN" type="text" min="1"size="3">回
62:B   <button id="try_start" onClick="startTry()" size="4">試行開始</button><br>
63:  <hr>
64:    <span id="try_num">??</span>回までの<br>
65:    目の出現回数と出現確率(参考：1/6≒16.7%)<br>
66:  <!-- 出現結果と確率 -->
67:C   1の目：<span id="ap1">0</span>回  出現確率 <span id="prob1">0</span>%<br>
68:    2の目：<span id="ap2">0</span>回  出現確率 <span id="prob2">0</span>%<br>
69:    3の目：<span id="ap3">0</span>回  出現確率 <span id="prob3">0</span>%<br>
70:    4の目：<span id="ap4">0</span>回  出現確率 <span id="prob4">0</span>%<br>
71:    5の目：<span id="ap5">0</span>回  出現確率 <span id="prob5">0</span>%<br>
72:    6の目：<span id="ap6">0</span>回  出現確率 <span id="prob6">0</span>%<br>
73:  <!-- ／出現結果と確率 -->
74:  </body>
75:  </html>
```

【注】 どうしても正常動作に至らない場合は、Ans_www フォルダ内の Ans_index.html を参考にせよ。

≪実習 4.2≫ サンプルアプリの www フォルダをエクスポートして、授業用フォルダに保管し、フォルダ名を saikoro とせよ。

≪実習 4.3≫ サンプルアプリでは、正 6 面体のサイコロをイメージしているので、index.html の 29 行目で、1～6 の整数乱数を生成している。この一部を変更して、正 8 面体のサイコロ状のものを想定して、1～8 の整数乱数を生成するように変更せよ。さらに、これに対応して、目の出現回数と出現確率の表示箇所を整え、必要な変数も用意せよ。

≪実習 4.4≫ これまでに変更してきたアプリの www フォルダをエクスポートして、授業用フォルダに保管し、フォルダ名を saikoro_ex とせよ。

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 50-55：サイコロのそれぞれの目の出現確率を画面上に表示する。このとき、確率は%表示とし、その表示も小数点 1 位(小数点 2 位を四捨五入)とする。四捨五入には Math ライブラリの round()関数を用いるが、『Math.round((c1/tryN)*★★)/★』の「★★」と「★」に設定する数を工夫する。例えば、「0.3456789」に対しては、まず「345.6789」とし、四捨五入で「346」となり、最後に「34.6」となるように「★★」と「★」を工夫する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

B 行 62：[試行開始]ボタンがクリックされたときに、startTry()関数を起動させる。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

C 行 67-72：サイコロの出た目ごとの出現回数と出現確率の表示場所を設定。特に、出現確率の表示箇所の id プロパティを正しく設定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

5. おみくじ

5-1. 本実習でのポイント

本実習では、以下の項目に特に注意する。

(a) HTML5

- HTML の基本的な記述スタイル
- HTML タグ内での id プロパティ、style プロパティ
- <body>タグとその中での onLoad プロパティ
- <button>タグとその中での onClick プロパティ、size プロパティ

(b) CSS (CSS3)

- text-align プロパティとその値
- font-size プロパティとその値
- color プロパティとその値(特に、16 進での RGB 指定)

(c) JavaScript

- 乱数の生成とその整数化
- else if 文の使い方、switch-case 文、文字列の連結
- 関数の呼び出しと引数(ひきすう)
- JavaScript における日付(Date オブジェクト)の取り扱い
- document オブジェクトと getElementById 関数

5-2. サンプルコードの解説と修正

◎サンプルアプリ名：omikuji.zip

スマホ画面におみくじを表示する。

≪実習 5.1≫ サンプルアプリを omikuji プロジェクトとしてインポートし、以下のプログラムと解説を参考に、インポートしたプログラム内の★の箇所を埋めよ。

行番号	omikuji//index.html
1:	<!DOCTYPE HTML>
2:	<html>
3:	<head>
4:	<meta charset="utf-8">
5:	<meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no">
6:	<meta http-equiv="Content-Security-Policy" content="default-src * data: gap: content: https://ssl.gstatic.com; style-src * 'unsafe-inline'; script-src * 'unsafe-inline' 'unsafe-eval'">
7:	<script src="components/loader.js"></script>
8:	<link rel="stylesheet" href="components/loader.css">
9:	<link rel="stylesheet" href="style">
10:	<script>
11:	function play() {
12: A	// 0～4 の範囲のランダムな値(整数)を得る
13:	var no = Math.floor(Math.random() * 5);
14:	// ランダムな値に応じて表示する画像を変える
15:	var image_name;

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 12-13：0～4 の整数の乱数を取得する。13 行目において、まず 0 以上 1 未満の実数の乱数を生成する関数を呼び出し、その値を 5 倍する。次に、その結果の小数点以下を切り捨てて整数(0～4)にする。

【修正問題】インポートプログラム内の★の箇所に適切なコードを記述せよ。

```
16:A          switch (no) {
17:             case 0: image_name = "daikichi.png"; break;
18:             case 1: image_name = "chuukichi.png"; break;
19:             case 2: image_name = "shoukichi.png"; break;
20:             case 3: image_name = "suekichi.png"; break;
21:             default: image_name = "kyou.png";
22:           }
23:B          // 画像と文字列の差し替え
24:           document.getElementById("omikuji").src = "images/" + image_name;
25:           document.getElementById("playBtn").innerHTML = "引きなおす";
26:         }
27:       </script>
28:       <style>
29:         body {
30:           background-image : url("../images/omikuji-bg.png");
31:           background-size : cover;
32:           background-repeat: no-repeat;
33:           margin : 0;
34:           padding : 0;
35:           height: 100%;
36:           width: 100%;
37:           text-align: center;
38:         }
39:         #omikuji{
40:           margin: 20px;
41:           width : 60%;
42:         }
43:         #playBtn {
44:           width: 60%;
45:           padding: 10px;
46:           font-size: 22px;
47:C          border-radius: 10px;
48:           background-color: #444444;
49:           color: white;
50:           outline: none;
51:         }
52:       </style>
53:   </head>
54:   <body>
55:D    <img id="omikuji" src="images/omikuji-box.png">
56:E      <button id="playBtn" onclick="play()">おみくじをひく</button>
57:   </body>
58:   </html>
```

【注】 どうしても正常動作に至らない場合は、Ans_www フォルダ内の Ans_index.html を参考にせよ。

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 16-22：変数 no の値(0～4)に応じて、画像ファイル名を変更する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、結果を振り分けよ。

B 行 23-25：DOM による画像と文字列を差し替える。25 行目では、ボタンのトップテキストを書き換えている。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

C 行 47：ボタンの四角形の角を丸くする。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、角のカーブを 10px にせよ。

D 行 55：おみくじ画像を表示するタグを id で指定する。また、最初のおみくじ画像を指定しておく。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

E 行 56：[おみくじをひく]ボタンタグを id で指定し、クリック操作に対応する JavaScript 関数を指定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

≪実習 5.2≫ サンプルアプリの www フォルダをエクスポートして、授業用フォルダに保管し、フォルダ名を omikuji とせよ。

≪実習 5.3≫ サンプルアプリの index.html の 16-22 行目では、switch-case 文を用いて、変数 no の値に応じた処理を記述している。この部分を else-if 文を用いた記述に変更せよ。else-if 文を用いて書き換えた際に、当初の行数から加減があっても構わない。
ここで、この実習が正しく終了した後は、元(16-22 行目)のように switch-case 文を用いた**記述を戻しておくこと**。

≪実習 5.4≫ サンプルアプリの index.html の 12-13 行目で、0～4 の整数乱数を生成しているが、13 行目の一部を変更して、0～9 の整数乱数を生成するように変更せよ。また、これによる、おみくじの各種類の出現頻度を確認せよ。

≪実習 5.5≫ 実習 5.4 を実施した後に、変数 no の 0～9 の値に対して右表のように当たり種類が対応するように index.html の 16-22 行目を変更せよ。また、この変更によって各種類の出現頻度を確認せよ。この変更の際に変更前の行数と変更後の行数が違っていても構わない。
ヒント：switch-case 文において、複数の値に処理を対応させる記述法を調べよ。

当たり種類	変数 no の値
大吉	0
中吉	1,2,3
小吉	4,5,6
末吉	7,8
凶	9

≪実習 5.6≫ 実習 5.5 で変更した switch-case 文による処理多分岐の部分を、else-if 文を用いた記述に変更せよ。この変更の際に変更前の行数と変更後の行数が違っていても構わない。
ヒント：else-if 文の条件式は、変数 no の値に対する等号による条件ではなく、不等号で表わす範囲の条件にするとよい。

≪実習 5.7≫ 上の実習 5.6 を実施すると、index.html の 12-13 行目で、0～9 の整数の乱数を生成している。これを整数ではなく、0 以上 1 未満の**実数**乱数となるように変更せよ。さらに、右表のように当たり種類が対応するように、上の実習 5.6 にて変更した箇所(16-22 行目相当)を再度修正せよ。

当たり種類	変数 no の値
大吉	0≦no＜0.1
中吉	0.1≦no＜0.4
小吉	0.4≦no＜0.7
末吉	0.7≦no＜0.9
凶	0.9≦no＜1.0

ヒント：0 以上 1 未満の**実数**乱数の生成には、元々の random()関数で良い。「0.4≦no＜0.7」のような変数 no の値の範囲をそのまま条件式で表すと、「0.4≦no かつ no＜0.7」のように複合条件としなければならない。しかし、else-if 文を正しく利用すれば、変数 no の範囲条件において、「0.4≦no」か「no＜0.7」のどちらか一方の単純条件として表現できる。

≪実習 5.8≫ これまでに変更してきたアプリの www フォルダをエクスポートして、授業用フォルダに保管し、フォルダ名を omikuji_ex とせよ。

≪実習 5.9≫ Monaca Dashboard より、sikoro と omikuji の 2 つのプロジェクトを削除せよ。

6. 数あてゲーム

6-1. 本実習でのポイント

本実習では、以下の項目に特に注意する。

(d) HTML5

- HTML の基本的な記述スタイル
- HTML タグ内での id プロパティ、class プロパティ、style プロパティ
- <input>タグとその中での size プロパティ
- <button>タグとその中での onClick プロパティ

(e) CSS (CSS3)

- text-align プロパティとその値
- font-size プロパティとその値
- padding プロパティとその値

(f) JavaScript

- 文字列の連結
- 関数の呼び出し
- 乱数の活用。isNaN の利用。
- JavaScript における文字列の表現「"」と「'」の使い方。
- document オブジェクトと getElementById 関数
- DOM における innerHTML の使い方。

6-2. サンプルコードの解説と修正

◎サンプルアプリ名：kazuate.zip

ランダムに設定された 0～99 の数を、ヒントを参考に当てる。

≪実習 6.1≫ サンプルアプリを kazuate プロジェクトとしてインポートし、以下のプログラムと解説を参考に、インポートしたプログラム内の★の箇所を埋めよ。

行番号	kazuate//index.html
1:	<!DOCTYPE HTML>
2:	<html>
3:	<head>
4:	<meta charset="utf-8">
5:	<meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no">
6:	<meta http-equiv="Content-Security-Policy" content="default-src * data: gap: content: https://ssl.gstatic.com; style-src * 'unsafe-inline'; script-src * 'unsafe-inline' 'unsafe-eval'">
7:	<script src="components/loader.js"></script>
8:	<link rel="stylesheet" href="components/loader.css">
9:	<link rel="stylesheet" href="css/style.css">
10:	<style>
11:A	.middle-size {
12:	font-size: 18px;
13:	}
14:	

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 11-13　HTML のタグ内の class プロパティで、「middle-size」と指定された箇所のスタイルを指定する。
ここでは、文字サイズ属性を「18px」に指定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

```
15:A        #challenge {
16:           padding-left: 20px;
17:         }
18:       </style>
19:
20:B      <script>
21:         var ans;      //  正解
22:         var ans_c;    //  解答回数
23:         var ch_msg;   //  チャレンジメッセージ
24:         var ans_col;  //  解答欄(テキストボックス)
25:         var kaito_id; //  解答欄の id プロパティ値
26:
27:C        function gameStart() {
28:D          ans=Math.floor(Math.random()*100);
29:E          document.getElementById("ch").innerHTML="チャレンジ ";
30:F          ans_c=1;
31:           ch_msg='<button ';
32:G          ch_msg+='class="middle-size" onClick="check()">'+ans_c+'回目 ';
33:           ch_msg+='</button>';
34:H          kaito_id='kaito'+ans_c;
35:I          ans_col='<input type="text" '+'class="middle-size" id="'+kaito_id+'"
      size="3">';
36:J          document.getElementById("challenge").innerHTML=ch_msg+ans_col;
37:K          document.getElementById("start").innerHTML="ゲームリセット";
38:         }
39:
40:L        function check() {
```

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 15-17　HTML のタグ内の id プロパティで、「challenge」と指定された箇所のスタイルを指定する。ここでは、余白(左側)属性を「20px」に指定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

B 行 20-65　HTML 内で、JavaScript のコードを記述するために必要なタグを指定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

C 行 27　JavaScript 内で、関数を宣言する際に必要な予約語を指定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

D 行 28　0～99 の整数乱数を生成して、変数 ans に保存する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、結果を振り分けよ。

E 行 29　id プロパティが「ch」と指定された要素(タグ)の箇所の HTML 記述を文字列『チャレンジ』に書き換える。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

F 行 30　ゲームスタート時やゲームリセット時に解答回数変数 ans_c を 1 に設定し直して、初期化する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

G 行 32　button タグ内において、ボタンがクリックされたときに JavaScript の関数 check()を呼び出すように設定する。また、button タグの HTML 記述箇所(ボタンのトップテキスト)を、解答回数に応じて、「1 回目」、「2 回目」、「3 回目」のように、解答回数と文字列「回目」を連結した表示に設定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

H 行 34　解答欄の id を表す変数 kaito _id を、文字列「kaito」と解答回数の連結として設定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

I 行 35　input タグ内の、class プロパティを「middle-size」に、size プロパティを「3」に設定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

J 行 36　id プロパティが「challenge」で指定されている要素の HTML 記述を、2 つの文字列変数 ch_msg と ans_col の連結で書き換える。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

K 行 37　id プロパティが「start」で指定されている要素(ボタンのトップテキスト)を、文字列「ゲームリセット」と書き換える。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

L 行 40　JavaScript 内で、関数を宣言する際に必要な予約語を指定する。

```
41:A          var kaito=parseInt(document.getElementById("kaito"+ans_c).value);
42:B          if (isNaN(kaito)) {
43:             return;
44:           }
45:           ch_msg+=" "+kaito+" : ";
46:
47:C          if (kaito==ans) {
48:D            ch_msg+='<span style="color: red;">○ 正解です！！</span>';
49:E            document.getElementById("challenge").innerHTML=ch_msg;
50:           } else {
51:F            if (kaito<ans) {
52:               ch_msg+='<span>↑ 上です！！</span><br />';
53:             } else {
54:               ch_msg+='<span>↓ 下です！！</span><br />';
55:             }
56:G            ans_c++;
57:             ch_msg+='<button ';
58:H            ch_msg+='class="middle-size" onClick="check()">'+ans_c+'回目 ';
59:             ch_msg+='</button>';
60:I            kaito_id='kaito'+ans_c;
61:J            ans_col='<input '+'class="middle-size" id="'+kaito_id+'" size="3">';
62:K            document.getElementById("challenge").innerHTML=ch_msg+ans_col;
63:           }
64:         }
65:       </script>
66:   </head>
67:
68:   <body>
69:L    <div style="text-align: center;">
70:       <h1>数あてゲーム</h1>
71:
```

※※※※※　プログラムの解説・修正問題　※※※※※

【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

A 行 41　id プロパティが文字列変数 kaito_id の値として指定される要素(解答欄：テキストボックス)の値を取り出し、整数値に変換する。

【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

B 行 42　41 行で変数 kaito に取得した値が「数でない(Not a Number)」かどうかを判定する。

【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

C 行 47　変数 kaito と変数 ans の値が一致する(等しい)かどうかを判定する。

【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

D 行 48　span タグ内の文字列の文字色を赤(red または#ff0000)に指定する。

【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

E 行 49　id プロパティが challenge のタグ内 HTML を変数 ch_msg の値に設定する。

【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

F 行 51　変数 kaito と変数 ans の値が一致しなかった場合で、変数 ans の方が大きいかどうかを判定する。

【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

G 行 56　解答回数を 1 だけ更新する。

【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

H 行 58　class プロパティが middle-size のボタンをクリックしたときに、check()関数を呼び出す。

【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

I 行 60　解答欄の id を表す変数 kaito _id を、文字列「kaito」と解答回数の連結として設定する。

【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

J 行 61　input タグ内の、class プロパティを「middle-size」に、size プロパティを「3」に設定する。

【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

K 行 62　id プロパティが challenge のタグ内 HTML を変数 ch_msg と ans_col の値どうしの連結に設定する。

【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

L 行 69　この div タグ内のテキスト等の表示を中央に配置する。

【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

```
72:       <p>0～99 の数を当ててください！</p>
73:A      <button id="start" onClick="gameStart()">ゲームスタート</button>
74:     </div>
75:B    <h2 id="ch"></h2>
76:C    <div id="challenge">
77:     </div>
78:
79:   </body>
80:   </html>
```

【注】 どうしても正常動作に至らない場合は、www_Ans フォルダ内の index_Ans.html を参考にせよ。

≪実習 6.2≫ サンプルアプリの www フォルダをエクスポートして、授業用フォルダに保管し、フォルダ名を kazuate とせよ。

≪実習 6.3≫ サンプルアプリでは、正解にたどり着くまで正解が分からないが、78 行目の部分にボタンを作成し、そのクリックによって、正解をアラートで表示するような関数を呼び出す工夫をせよ。

ヒント：正解をアラートで表示する関数を JavaScript で作成し、ボタンをクリックしたときにその関数を呼び出すように設定する。

≪実習 6.4≫ サンプルアプリでは、0～99 の数あてになっているが、どんな数であっても最大何回の解答で正解にたどり着けるか検討し、その方法を、実習 6.3 で表示させたアラートの中に正解と併せて表示せよ。

≪実習 6.5≫ サンプルアプリでは、0～99 の数あてゲームになっているが、これを 0～999 に変更せよ。

≪実習 6.6≫ サンプルアプリでは、当てる数の最大値が固定されている。そこで、71 行目にこの最大値を変更するためのテキストボックスを用意し、そこで設定された値に対応した数あてゲームとなるように変更せよ。

≪実習 6.7≫ これまでに変更してきたアプリの www フォルダをエクスポートして、授業用フォルダに保管し、フォルダ名を kazuate_ex とせよ。

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 73　この button タグの id プロパティを start に設定し、クリックしたときに、JavaScript の関数 gameStart()を呼び出す。
　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

B 行 75　この h2 タグの id プロパティを ch に設定する。
　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

C 行 76-77　この div タグの id プロパティを challenge に設定する。
　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

7. クイズ

7-1. 本実習でのポイント

本実習では、以下の項目に特に注意する。

(g) HTML5

- HTML の基本的な記述スタイル
- HTML タグ内での id プロパティ、style プロパティ
- <body>タグとその中での onLoad プロパティ
- <input>タグとその中での placeholder プロパティ
- <button>タグとその中での onClick プロパティ

(h) CSS (CSS3)

- HTML ファイルとは別の外部ファイルとしての利用
- text-align プロパティとその値
- font-size プロパティとその値
- color プロパティとその値(特に、16 進での RGB 指定)
- width プロパティとその値
- margin プロパティとその値
- id セレクタの表現

(i) JavaScript

- 連想配列の配列、配列の長さ(要素数)
- else if 文の使い方、switch-case 文、文字列の連結
- 関数の呼び出し
- document オブジェクトと getElementById 関数

7-2. サンプルコードの解説と修正

◎サンプルアプリ名：quiz.zip

スマホ画面に英単語(小文字)問題を表示する。

≪実習 7.1≫ サンプルアプリを quiz プロジェクトとしてインポートし、以下のプログラムと解説を参考に、インポートしたプログラム内の★の箇所を埋めよ。

ここで、HTML ファイルは www フォルダの直下にあるが、**スタイルシートファイル style.css は www¥css の中にある**ことに注意する。

行番号	quiz //css¥style.css
1:	body {
2:A	text-align: center;
3:	}
4:	
5:	h1 {
6:B	color: #979797;
7:	}
8:	

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 2：body セレクタに対して、テキストの横方向の配置を中央揃えに設定する。

【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

B 行 6：h1 セレクタに対して、文字色を 16 進で「979797」に設定する。

【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

```
 9:A #question {
10:B   width: 100%;
11:C   margin: 40px 0px;
12:D   font-size: 60px;
13:  }
14:E #answerForm {
15:F   font-size: 20px;
16:  }
17:G #answer {
18:H   width: 60%;
19:  }
```

行番号	quiz//index.html
1:	<!DOCTYPE HTML>
2:	<html>
3:	<head>
4:	<meta charset="utf-8">
5:	<meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no">
6:	<meta http-equiv="Content-Security-Policy" content="default-src * data: gap: content: https://ssl.gstatic.com; style-src * 'unsafe-inline'; script-src * 'unsafe-inline' 'unsafe-eval'">
7:	<script src="components/loader.js"></script>
8:	<link rel="stylesheet" href="components/loader.css">
9:	<link rel="stylesheet" href="css/style.css">
10:	<script>
11:	// 問題番号
12:I	var no = 0;
13:	// 正解数
14:J	var score = 0;
15:	// 単語リスト
16:K	var wordList = [
	{

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 9：id 名「question」に対する id セレクタに対するスタイル指定の記述であることを示す。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

B 行 10：横幅(width)をスマホ画面(横幅)いっぱい(100%)に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

C 行 11：マージンを左右 40px、上下 0px に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

D 行 12：フォントサイズを 60px に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

E 行 14：id 名「answerForm」に対する id セレクタに対するスタイル指定の記述であることを示す。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

F 行 12：フォントサイズを 20px に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

G 行 17：id 名「answer」に対する id セレクタに対するスタイル指定の記述であることを示す。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

H 行 18：横幅(width)をスマホ画面(横幅)の 60%に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

I 行 12：変数 no は、現在表示している問題の番号を表している。ここで初期化して、次の問題に進むたびに 1 ずつ増加する(81 行目)。

J 行 14：変数 score は、正解数をカウントしている。ここで初期化して、正解するたびに 1 ずつ増加する(74 行目)。

K 行 16-37： 設問と正解の組を連想配列で表し、その配列として問題全体を表している。問題全体を表す配列は角かっこの『[』と『]』で囲み、各設問と解答の組の連想配列は波かっこの『{』と『}』で囲まれることに注意する。

```
17:A          japanese: "電話",
18:           english: "phone"
19:         },
20:         {
21:           japanese: "歴史",
22:           english: "history"
23:         },
24:         {
25:           japanese: "社会",
26:           english: "society"
27:         },
28:         {
29:           japanese: "世代",
30:           english: "generation"
31:         },
32:         {
33:           japanese: "知識",
34:           english: "knowledge"
35:         }
36:       ];
37:
38:       // 問題を表示する
39:       function showQuestion() {
40:B          if (no<wordList.length) {
41:C            //  次の問題がある場合は、表示する
42:             document.getElementById("question").innerHTML=wordList[no].japanese;
43:D          } else {
44:             //  全問終了したら、成績を発表する
45:               document.getElementById("question").innerHTML=score+"/"+wordList.length;
46:E            document.getElementById("answerForm").style.display="none";
47:F            var msg;
48:             var img;
49:             if (score==wordList.length) {
50:G              //  全問正解の場合
51:               msg="全問正解！よくできました！";
52:               img="gold.png";
53:             } else if (score>=wordList.length*0.6) {
```

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 17-20： 一つの設問と正解の組を波かっこの『{』と『}』で囲まれる一つの連想配列で表している。連想配列の要素は、キーと値の組で『キー：値』のように表される。ここでは、ここでは、設問のキーが「japanese」、解答のキーが「english」であり、それぞれの値(文字列)が設問と解答に対応している。

B 行 40-67： 関数 showQuestion()を記述している。ここでは次の問題の表示だけでなく、すべての問題が終了したときに、成績に応じた評価も表示する。

C 行 41-66：表示する問題番号が wordList に用意した要素数より小さければその問題を表示し、そうでなければ全問終了したとして成績を評価する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

D 行 43：id 名 question を持つタグ内に、wordList の(変数 no の値で示される順番の連想配列としての)要素の設問キーを指定する。このとき、そのキーに対応する値が設定される。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

E 行 46：id 名 question を持つタグ内に、「正解数/問題数」のように表示する。ここで、正解数は変数 score の値であり、問題数は WordList の要素数である。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

F 行 47：全問終了したときには、解答の入力欄と解答ボタンを表示していた id 名 answerForm のタグ内は表示すべきではないので、ここの style の display プロパティを非表示に指定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

G 行 50：全問正解したかどうかを判定する。ここで、全問正解とは正解数が問題数(WordList の要素数)に一致する時である。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

```
54:A                msg="惜しい！あともう一歩でした！";
55:                 img="silver.png";
56:               } else if (score>=wordList.length*0.4) {
57:B                msg="もう少しです！";
58:                 img="bronze.png";
59:               } else {
60:                 msg="もうちょっとがんばりましょう！";
61:                 img="partic.png";
62:               }
63:               document.getElementById("resultMessage").innerHTML=msg;
64:C              document.getElementById("resultImage").src=img;
65:D          }
66:       }
67:
68:       // 入力された回答の正誤判定を行う
69:       function judge() {
70:E          var answer=document.getElementById("answer").value;
71:F          if (answer==wordList[no].english) {
72:G            alert("正解です");
73:H            score++;
74:           } else {
75:             alert("残念! 不正解です");
76:           }
77:           document.getElementById("answer").value="";
78:I
79:           //  次の問題を表示
80:           no++;
81:           showQuestion();
82:J      }
83:      </script>
84:   </head>
85:
86:   <body onload="showQuestion()">
```

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 54：正解数が問題数の 6 割以上かどうかを判定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

B 行 57：正解数が問題数の 4 割以上かどうかを判定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

C 行 64：id 名 resultMessage を持つタグ内に、変数 msg の値を表示するように設定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

D 行 64：id 名 resultImage を持つ img タグ内で表示する画像ファイルを変数 img に設定されているファイル名に設定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

E 行 70-83： 関数 judge()を記述している。ここでは各設問に対する採点を行い、正解であれば変数 score の値を 1 増加させる。また、次の問題に備えて変数 no も 1 増加させて、showWuestion() 関数を呼び出す。

F 行 71：id 名 answer を持つ input タグ内で表示されたテキストボックス内の値をそのまま変数 answer に取り込む。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

G 行 72：変数 answer の値が、現在の問題番号の問題の正解に一致するかどうか判定する。ここで、正解は問題の解答キーを指定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

H 行 73,76：解答の正解/不正解に応じたメッセージを警告(アラート)として表示する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

I 行 78：id 名 answer を持つ input タグ内で表示されたテキストボックス内の値を初期化(空の文字列)にする。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

J 行 82：関数 showQuestioon()を呼び出す。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

```
87:A   <h1 id="title">英単語(小文字)クイズ</h1>
88:    <div id="question"></div>
89:B   <div id="answerForm">
90:C     <input type="text" id="answer" placeholder="英単語を入力">
91:D     <button onclick="judge()">解答</button>
92:E   </div>
93:    <p id="resultMessage"></p>
94:F   <img id="resultImage" src="">
95:G </body>
96:  </html>
97:
```

【注】 どうしても正常動作に至らない場合は、www_Ans フォルダ内の index_Ans.html と css サブフォルダ内の style_Ans.css を参考にせよ。

≪実習 7.2≫ サンプルアプリの www フォルダをエクスポートして、授業用フォルダに保管し、フォルダ名を quiz とせよ。

≪実習 7.3≫ サンプルアプリでは、成績の評価を正解率 10 割、6 割以上、4 割以上、それ未満としているが、この基準を矛盾の無い範囲で変更してみよ。

≪実習 7.4≫ サンプルアプリでは成績の評価が 4 段階であるが、これを矛盾の無い範囲で 5 段階に変更せよ。その際、必要なメッセージや画像は適当に準備せよ。

≪実習 7.5≫ サンプルアプリは 5 問のクイズであるが、他の 5 個の問題と解答を適切に追加して、10 問のクイズとせよ。

≪実習 7.5≫ サンプルアプリは英単語のクイズであるが、これを別のジャンルのクイズに変更してみよ。ここで、正解が一つに特定できる問題のみが対象となる事に注意せよ。

≪実習 7.6≫ これまでに変更してきたアプリの www フォルダをエクスポートして、授業用フォルダに保管し、フォルダ名を quiz_ex とせよ。

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 87：body タグ箇所がロードされたときに関数 showQuestioon()を呼び出す。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。
B 行 89：この div タグの id 名を question に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。
C 行 90：この div タグの id 名を answerForm に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。
D 行 91：この div タグの id 名を answer に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。
E 行 87：この button がクリックされたときに関数 judge()を呼び出す。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。
F 行 94：この p タグの id 名を resultMessage に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。
G 行 95：この img タグの id 名を resultImage に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

8. 計算テスト

8-1. 本実習でのポイント

本実習では、以下の項目に特に注意する。

(a) HTML5

- HTML の基本的な記述スタイル
- HTML タグ内での id プロパティ、style プロパティ
- <button>タグとその中での onclick プロパティ
- <table>タグ内の<td>タグ内の colspan プロパティ

(b) CSS (CSS3)

- HTML ファイルとは別の外部ファイルとしての利用
- ブロックの右寄せ、左寄せ
- 以下のプロパティとその値

  display、color、width、margin、text-align、font-size
- id セレクタの表現

(c) JavaScript

- 乱数の生成、小数点以下の切り捨て
- switch case 文の使い方、文字列の連結
- 関数の呼び出しとその際の引数(ひきすう)
- document オブジェクトと getElementById 関数

8-2. サンプルコードの解説と修正

◎サンプルアプリ名：keisanTest.zip

スマホ画面に英単語(小文字)問題を表示する。

≪実習 8.1≫ サンプルアプリを keisanTest プロジェクトとしてインポートし、以下のプログラムと解説を参考に、インポートしたプログラム内の★の箇所を埋めよ。

ここで、HTML ファイルは www フォルダの直下にあるが、**スタイルシートファイル style.css は www¥css の中にある**ことに注意する。

行番号	keisanTest//css¥style.css
1:A	#prob {
2:B	font-size: 28px;
3:	}
4:	
5:C	#prob_ans {
6:D	display: none;
7:	}
8:	

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 1：id 名「prob」の id セレクタに対するスタイル指定の記述であることを示す。
　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

B 行 2：フォントサイズを 28px に設定する。
　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

C 行 5：id 名「prob_ans」の id セレクタに対するスタイル指定の記述であることを示す。
　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

D 行 6：id 名「prob_ans」の表示スタイルを「非表示」に設定する。
　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

```
 9:   input {
10:A    font-size: 28px;
11:  }
12:
13:B #eval {
14:C    font-size: 120px;
15:D    color: red;
16:  }
17:  td {
18:E    height: 40px;
19:F    width: 40px;
20:  }
21:
22:  button {
23:G    font-size: 28px;
24:H    height: 100%;
25:I    width: 100%;
26:  }
```

行番号	keisanTest//index.html
1:	<!DOCTYPE HTML>
2:	<html>
3:	<head>
4:	<meta charset="utf-8">
5:	<meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no">
6:	<meta http-equiv="Content-Security-Policy" content="default-src * data: gap: content: https://ssl.gstatic.com; style-src * 'unsafe-inline'; script-src * 'unsafe-inline' 'unsafe-eval'">
7:	<script src="components/loader.js"></script>
8:	<link rel="stylesheet" href="components/loader.css">
9:J	<link rel="stylesheet" href="css/style.css">
10:	<script>
11:K	var a, b; // 生成される 2 つの乱数(1～9)の保存変数
12:	var cor; // a*b の計算結果(正解)の保存変数

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 10：フォントサイズを 28px に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

B 行 13：id 名「eval」の id セレクタに対するスタイル指定の記述であることを示す。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

C 行 14：フォントサイズを 120px に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

D 行 15：フォント色を赤(red)に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

E 行 18：高さを 40px に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

F 行 19：幅を 40px に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

G 行 23：フォントサイズを 28px に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

H 行 24：高さを 100%に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

I 行 25：幅を 100%に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

J 行 9：サブフォルダ css 内のスタイルシートファイル style.css を利用するための link タグによる表記。

K 行 11-13：この HTML ファイルの JavaScript において、どの関数からも利用できるようなグローバル変数として宣言している。

```
13:        var ans;    //  入力される解答の保存変数
14:
15:        //  計算問題の出題関数
16:A       function start() {
17:B         a=Math.floor(Math.random()*9+1);
18:C         b=Math.floor(Math.random()*9+1);
19:D         cor=a*b;
20:
21:E         var probmsg=a+"×"+b+"＝";
22:F         document.getElementById("prob").innerHTML=probmsg;
23:G         document.getElementById("eval").innerHTML="";
24:H         document.getElementById("answer").value=""
25:I         document.getElementById("prob_ans").style.display="block";
26:        }
27:
28:        //  テンキーの押下処理関数
29:J       function push(arg) {
30:          var ansText="";
31:          switch (arg) {
32:K            case "C": document.getElementById("answer").value="";
33:                      break;
34:L            case "E": check();
35:                      break;
36:M            default: ansText=document.getElementById("answer").value;
37:N                     if (ansText=="0") ansText="";
```

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 16-26： 出題関数 start()を記述している。ここでは、乱数を用いて 2 つの数(1～9)の掛け算問題を作成して表示する。

B 行 17：1～9 の乱数を生成して、変数 a に代入する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

C 行 18：1～9 の乱数を生成して、変数 b に代入する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

D 行 19：2 変数の掛け算結果を変数 cor に代入する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

E 行 21： 変数 probmsg に「(変数 a の値)×(変数 b の値)＝」の形式の問題文を文字と変数との連結で設定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

F 行 22：id 名 prob を持つタグ内に問題文を設定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

G 行 23：id 名 eval を持つタグ内に空の文字列を設定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

H 行 24： id 名 answer を持つ<input>タグの値として空の文字列を設定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

I 行 25：id 名 prob_ans を持つ タグのスタイルの display プロパティを block 形式に設定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

J 行 29-41： テンキーのボタンを押した際の処理関数 push()を記述している。push()関数は押されたボタンに応じた値を引数として取っている。

K 行 32：関数 push()への引数が文字の「C」の場合の処理。id 名 answer を持つ<input>タグの値として空の文字列を設定する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

L 行 34：関数 push()への引数が文字の「E」の場合の処理。解答をチェックする関数 check()を引数なしで呼び出す。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

M 行 36：関数 push()への引数が文字の「C」や「E」以外の場合の処理。まず、id 名 answer を持つ<input>タグの値(文字列)を変数 ansText に代入する。
【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

N 行 37： 値の先頭が 0 になることを防ぐために、変数 ansText が「0」ならば、それを空の文字に変える。

```
38:A                    ansText+=arg;
39:B                    document.getElementById("answer").value=ansText;
40:       }
41:     }
42:
43:     //  解答のチェック関数
44:C    function check() {
45:D      ans=parseInt(document.getElementById("answer").value);
46:       if (ans==cor) {
47:E        document.getElementById("eval").innerHTML="○";
48:       } else {
49:F        document.getElementById("eval").innerHTML="×";
50:       }
51:     }
52:   </script>
53: </head>
54: <body>
55:   <h1>計算テスト</h1>
56:   <p>あなたの計算能力をテストします。</p>
57:   <div>
58:G    <div style="float:left;">
59:H      <button id="start" onclick="start()">出題</button><br>
60:       <div id="prob_ans">
61:I        <span id="prob"></span>
62:J        <input type="text" id="answer" size="2" value="">
63:       </div>
64:K      <span id="eval"></span>
65:     </div>
66:L    <table style="float:right;">
67:       <tr>
68:M      <td colspan="3"><button onclick="push('C')">Clear</button></td>
```

※※※※※　プログラムの解説・修正問題　※※※※※

A 行 38：変数 ansText に、push 関数の引数として渡された数を連結する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

B 行 39：id 名 answer を持つ<input>タグの値(文字列)として変数 ansText を設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

C 行 44-51： 解答をチェックする関数 check()を記述している。ここでは正解かどうかに応じて、その評価を画面に表示している。

D 行 45：id 名 answer を持つ<input>タグの値(文字列)を数(整数値)に変換して、変数 ans に代入する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

E 行 23：id 名 eval を持つタグ内に文字「○」を設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

F 行 23：id 名 eval を持つタグ内に文字「×」を設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

G 行 58：この タグが示すブロックを左寄せにするために、タグ内で style プロパティを指定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

H 行 59：この<button>タグで表示されるボタンをクリックした際に、JavaScript 関数 start()を呼び出す。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

I 行 61：id 名 prob を持つこのタグ内には何も表示されるものが設定されていないが、ここには JavaScript の関数内の処理によって、表示内容が設定される。

J 行 62：この<input>タグで表示されるテキストボックスの幅を「2」に設定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

K 行 64：id 名 eval を持つこのタグ内には何も表示されるものが設定されていないが、ここには JavaScript の関数内の処理によって、表示内容が設定される。

L 行 66：この<table>タグが示すブロックを右寄せにするために、タグ内で style プロパティを指定する。
　　【修正問題】インポートプログラム内の★の個所に適切なコードを記述して、書き換えよ。

M 行 68： push 関数の引数として文字「C」を「'」(シングルクォテーション)を用いて指定している。これは、HTML タグ内で「"」(ダブルクォテーション)が別の使われ方をしているからである。

```
69:         </tr>
70:         <tr>
71:           <td><button onclick="push(7)">7</button></td>
72:           <td><button onclick="push(8)">8</button></td>
73:           <td><button onclick="push(9)">9</button></td>
74:         </tr>
75:         <tr>
76:           <td><button onclick="push(4)">4</button></td>
77:           <td><button onclick="push(5)">5</button></td>
78:           <td><button onclick="push(6)">6</button></td>
79:         <tr>
80:           <td><button onclick="push(1)">1</button></td>
81:           <td><button onclick="push(2)">2</button></td>
82:           <td><button onclick="push(3)">3</button></td>
83:         <tr>
84:           <td><button onclick="push(0)">0</button></td>
85:           <td colspan="2"><button onclick="push('E')">Enter</button></td>
86:         </tr>
87:       </table>
88:     </div>
89:   </body>
90:   </html>
```

【注】 どうしても正常動作に至らない場合は、www_Ans フォルダ内の index_Ans.html と css サブフォルダ内の style_Ans.css を参考にせよ。

≪実習 8.2≫ サンプルアプリの www フォルダをエクスポートして、授業用フォルダに保管し、フォルダ名を keisanTest とせよ。

≪実習 8.3≫ サンプルアプリの 31-40 行の間の switch-case 文を、else-if 文で表現し直せ。

≪実習 8.4≫ サンプルアプリは掛け算のテストだが、加算(足し算)のテストに変更せよ。

≪実習 8.5≫ サンプルアプリは一種類の計算テストだが、足し算と掛け算がランダムに出題されるように変更せよ。
ヒント：2 種類の計算方法(足し算と掛け算)も乱数で決めるようにすると良い。

≪実習 8.6≫ サンプルアプリでは、1 回の解答に対する評価が出るようになっているが、何回も解答を繰り返した際に、「正解数／回答数」のような累積成績が表示できるように改良せよ。

≪実習 8.7≫ これまでに変更してきたアプリの www フォルダをエクスポートして、授業用フォルダに保管し、フォルダ名を keisanTest_ex とせよ。