

Integration Of System

株式会社アイオス
インターンシップ



～プログラム体験ワーク～

プログラム基礎学習編

1. はじめに
 - ・ プログラム基礎学習編の目的
 - ・ プログラムを学ぶ前に
2. プログラム演習①
3. プログラム (JavaScript) の基本
4. プログラムの基本構文
5. 環境準備
6. プログラム演習②
7. さいごに

はじめに

はじめに（プログラム基礎学習編の目的）



インターンシップの目的

本コンテンツでは、「プログラムとは何かを知ること」のみではなく、分からないことが発生した際に「周囲の方と協力しながら答えを見つけること」の2つの経験を目的とします。

はじめに（プログラムを学ぶ前に）

プログラム言語とは？

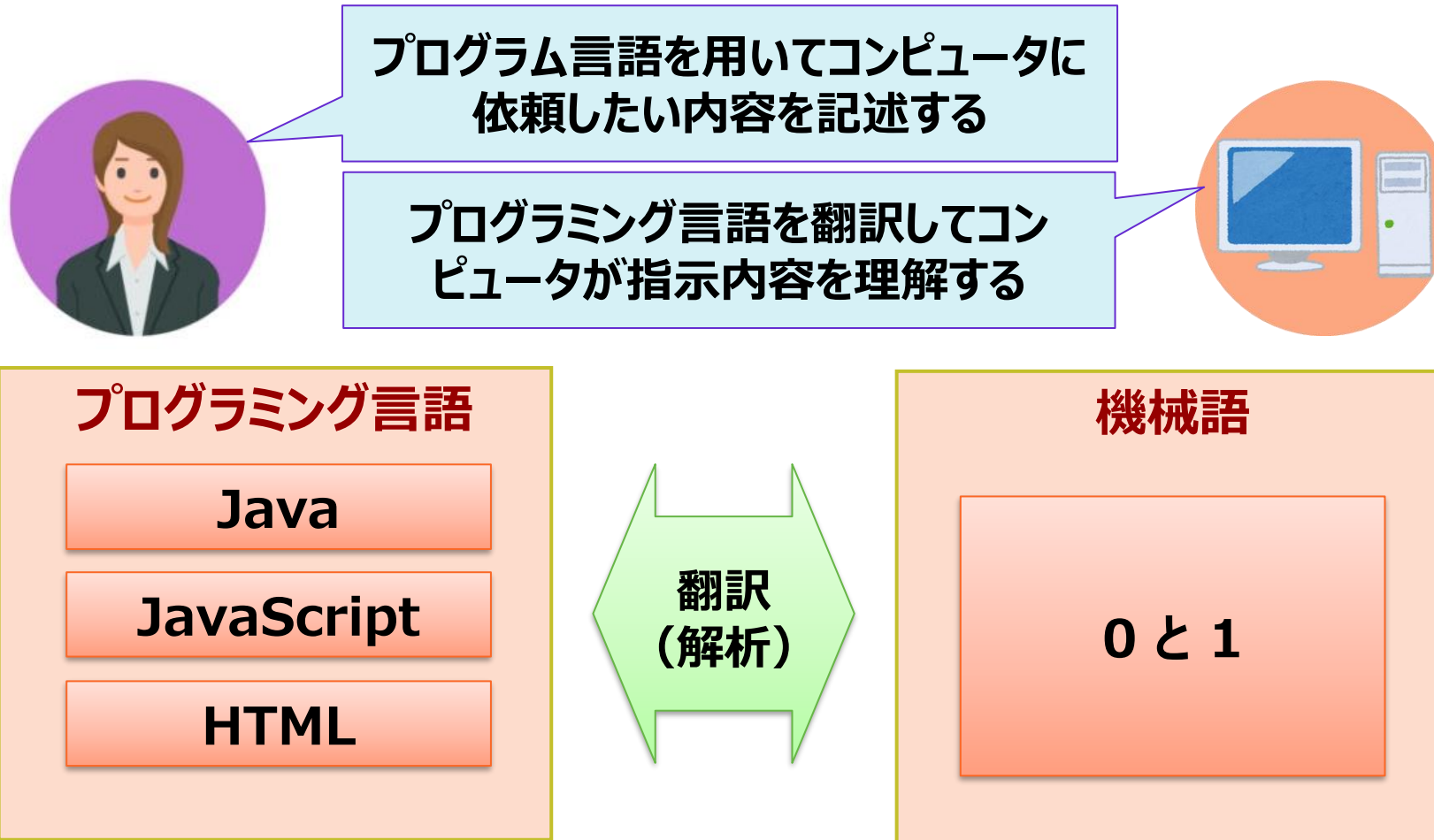
コンピュータに対して「こうやって動いて欲しい！」と伝えるための行動のことを「プログラミング」といいます。日本語や外国語と殆ど同じ意味合いです。

コンピュータは、すべての情報を0(ゼロ)と1(イチ)で扱っています。コンピュータが直接理解できるこの0(ゼロ)と1(イチ)が並ぶ情報を「機械語」といいます。



はじめに（プログラムを学ぶ前に）

プログラム言語とは？



HTML

HTMLとは、Hyper Text Markup Language（ハイパーテキスト・マークアップ・ランゲージ）の略、「**WEBページの言葉の部分を作成するためのプログラミング言語**」と読み解きます。

JavaScript

最も一般的な使われ方が、「**WEBページに動きを付ける**」ことです。写真をスライドさせて表示したり、WEBページにあるボタンなどに動きを付ける動作は、JavaScriptなどで制作されています。Webページに動きを付けることによって、検索している人の目を引いたり、その人の行動を促す効果があります。動きをつけて質の高いWEBサイト作るときには、欠かせないプログラミング言語になります。

プログラム演習①

Step1 事前準備

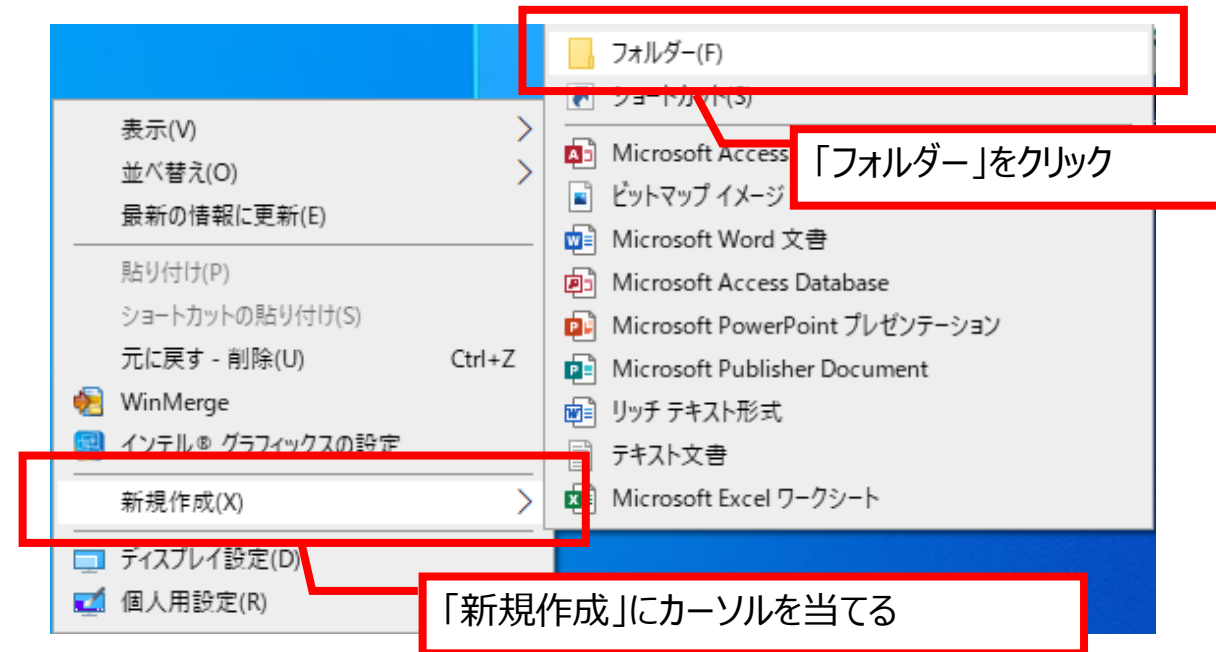
プログラム演習に備えて、以下手順を実施する。

手順 1 デスクトップ上新規フォルダ作成

①デスクトップ上で、右クリックを押下し左記画面を参考に
新規フォルダを作成する。

②フォルダの名前を isjs として設定する。

※名前の変更方法が分からない場合には
講師に確認しましょう！



Step1 事前準備

手順2 フォルダ拡張子設定

- ① 手順1で作成したフォルダを開き、左記画面を参考に
“ファイル名拡張子”にチェックが入っていることを確認する。
※チェックが入っていない場合、チェックを付ける

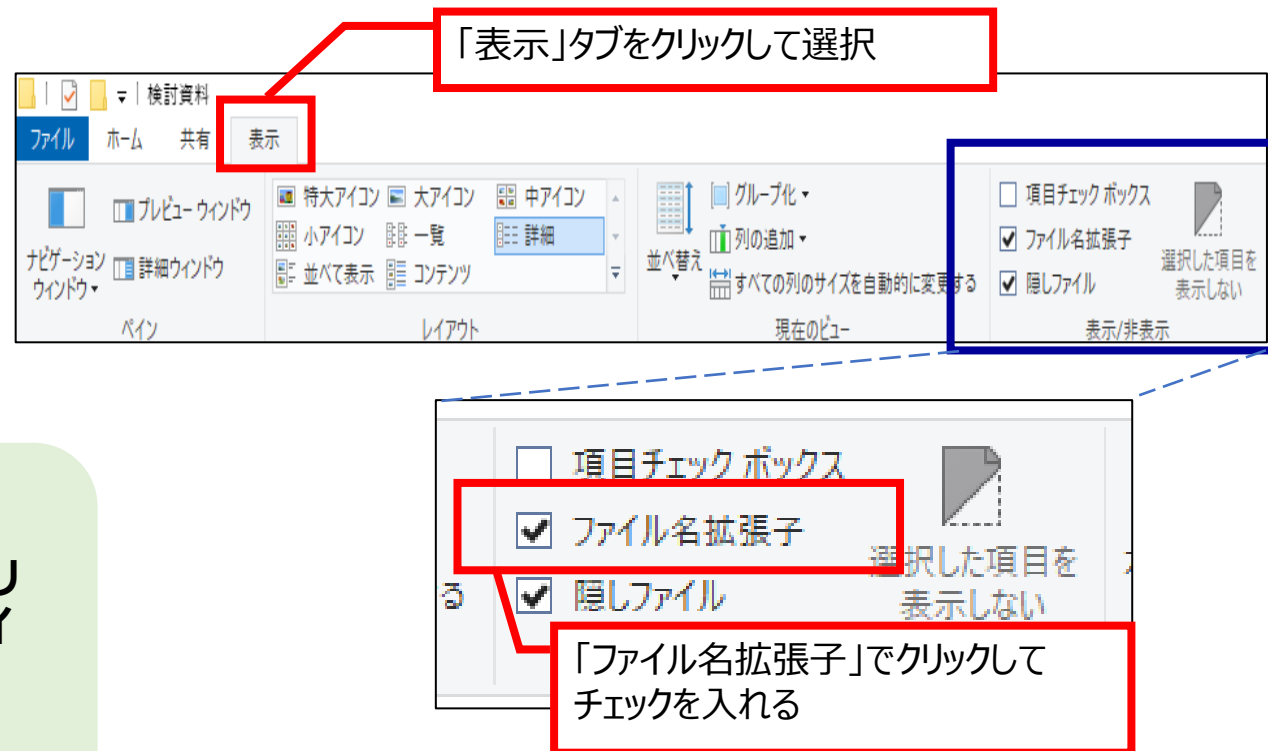
【拡張子について】

拡張子とはファイルの種類を示します。拡張子名で、どのアプリケーションで利用するファイルであるのか概ね分かります。ファイル名と拡張子の間は、ドット（ピリオド）で区切られています。

例) **Sample.xlsx** ← 拡張子
ファイル名 ↑ ↑ 区切り文字のドット

拡張子の種類

txt:テキスト文書、pdf:PDFファイル、doc:Wordファイル
js:JavaScriptファイル、html:HTMLファイル など



Step1 事前準備

手順3 ファイルダウンロード

①ファイル共有サイトにアクセスする。

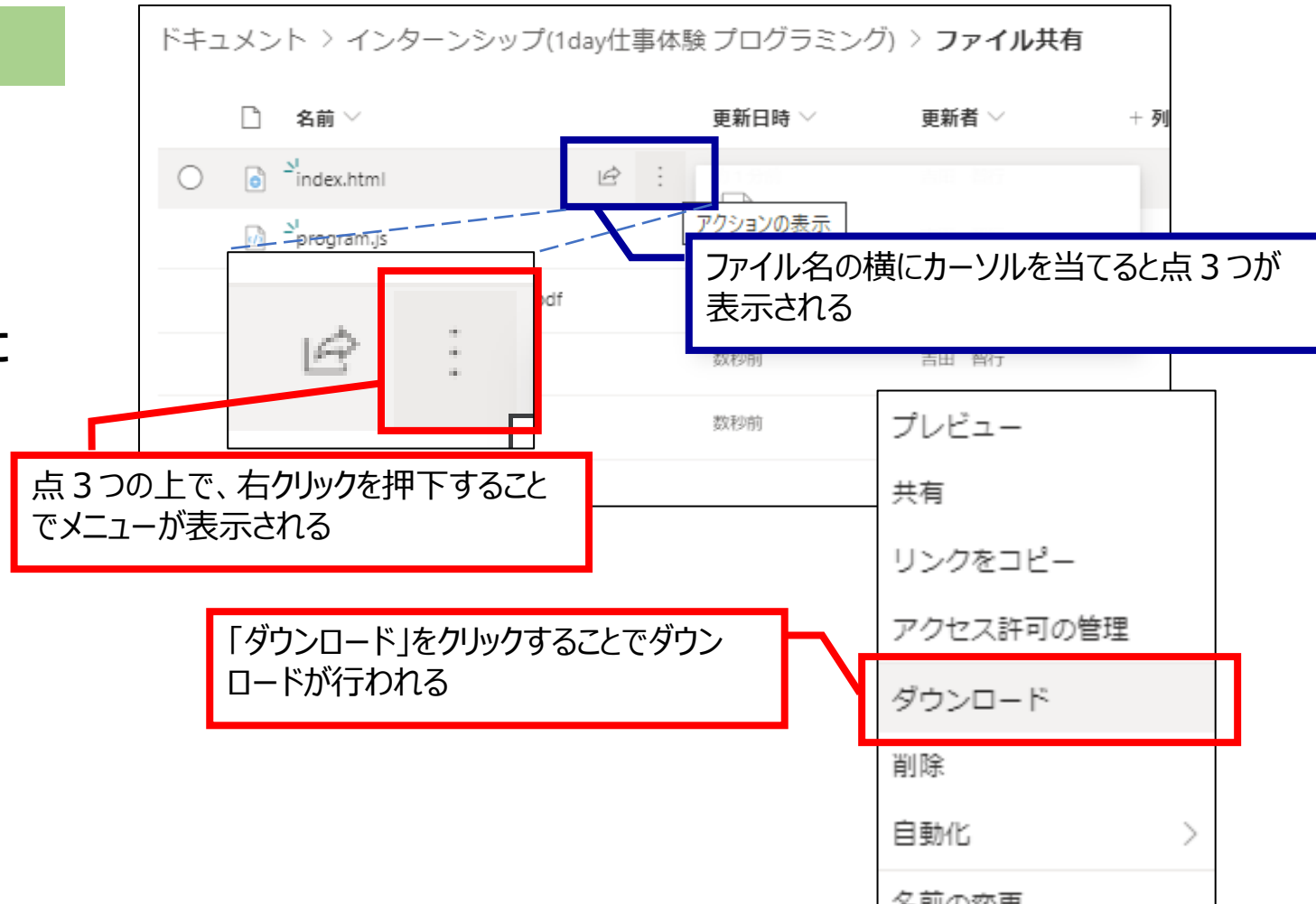
※入力パスワードはチャットにて共有

②ファイル共有サイト上に存在するファイルを全て

手順2で作成したフォルダ配下に左記画面を参考にダウンロードする。

※ダウンロード対象ファイルは以下5ファイル

- 1) index.html
- 2) program.js
- 3) プログラム基礎学習編資料.pdf
- 4) 画面設計書.pdf
- 5) 機能仕様書.pdf



Step2 サンプルプログラム作成

手順4 新規ファイル作成

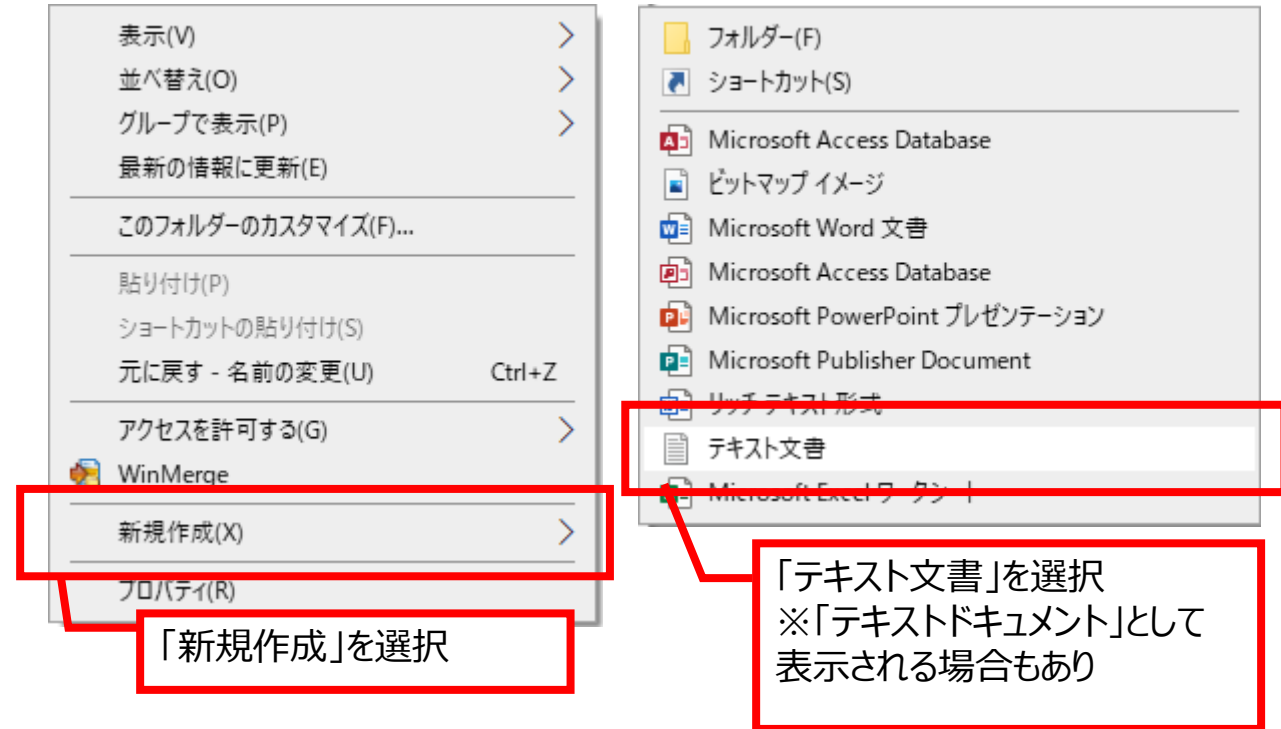
①手順2で作成したフォルダを開き、フォルダ上で右クリック
押下し左記画面を参考に新規ファイル（テキストファイル）
を2つ作成する。

※この時はファイル名称を変更しないため以下のような
名前になって2つ作成されていれば良い

「新しいテキスト ドキュメント.txt」

「新しいテキスト ドキュメント (2).txt」

「新しいテキスト ドキュメント - コピー.txt」



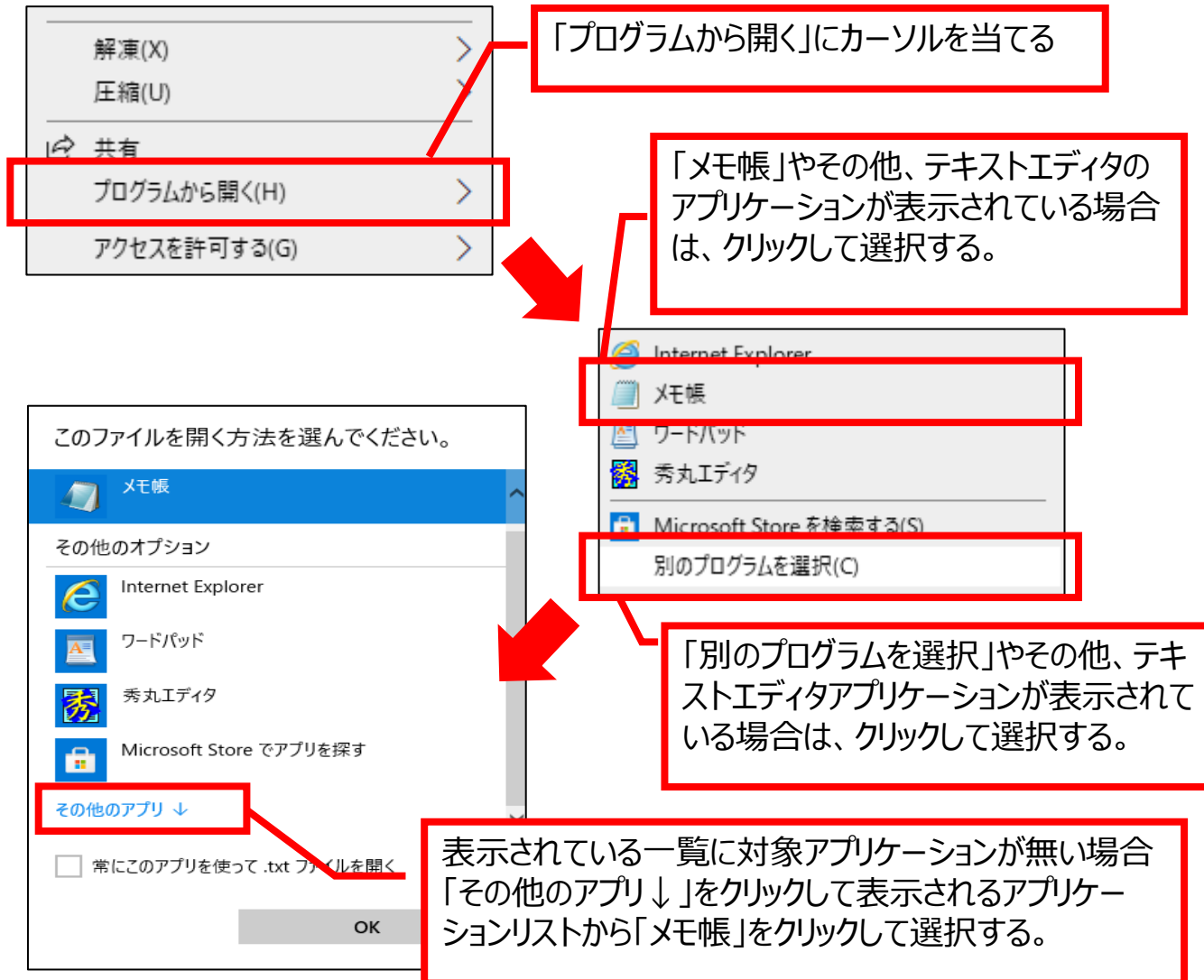
Step2 サンプルプログラム作成

手順5 テキストファイルを開く

①手順4で作成したファイルをテキストエディタで開く。
この際、テキストエディタのアプリケーションを既に使っている方はご自身の使い慣れたもので開く。テキストエディタアプリケーションを使っていない方は「メモ帳」で開く。

※テキストエディタを開く方法

- 1) 拡張子“.txt”のファイルをダブルクリックして開く
- 2) 手順4で作成したファイル上（名前の上）で、
右クリックを押下し左記画面を参考にアプリケーションを選んで開く



「プログラムから開く」にカーソルを当てる

「メモ帳」やその他、テキストエディタのアプリケーションが表示されている場合は、クリックして選択する。

「別のプログラムを選択」やその他、テキストエディタアプリケーションが表示されている場合は、クリックして選択する。

表示されている一覧に対象アプリケーションが無い場合「その他のアプリ ↓」をクリックして表示されるアプリケーションリストから「メモ帳」をクリックして選択する。

Step2 サンプルプログラム作成

手順6 テキストファイルを編集する

- ① 2つのファイルについて、それぞれ以下内容（1行）を入力し保存する。

【ファイル1】

```
<script src="sample.js"></script>
```

【ファイル2】

```
alert("Hello World!!");
```

※全て半角文字で記載することに注意して入力する

Step2 サンプルプログラム作成

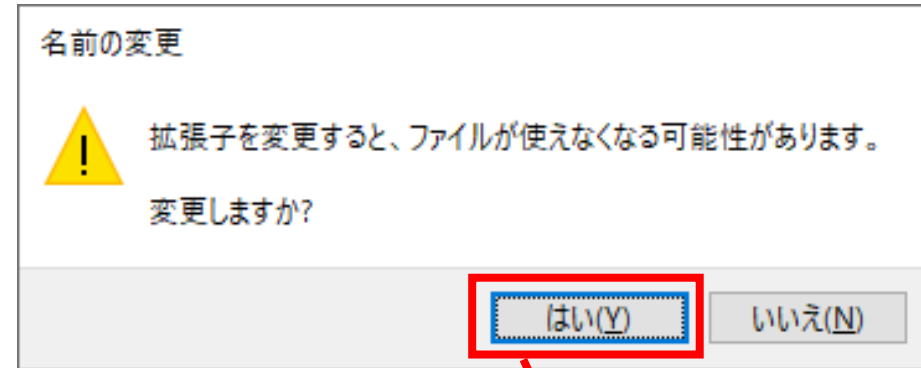
手順7 ファイル名を編集する

①手順6で作成した2つのファイルについて、ファイル名称をそれぞれ以下の通り編集する。ファイル名を変更すると、右図のような 確認ダイアログが表示されるため「はい(Y)」をクリックして選択する。

ファイル1 test.html

ファイル2 sample.js

②ファイル名を変更すると、ファイルアイコンが変更されることを確認する。



「はい(Y)」をクリックして選択



Step2 サンプルプログラム作成

手順8 ファイルを実行する

- ①手順7までの作業で作成した“test.html”をダブルクリックして実行する。
エクスプローラーが起動し、“Hello World!!”が表示されることを確認する。

※ダブルクリックして実行してもエクスプローラーが起動しない。または“Hello World!!”が表示されない場合には、手順を戻って確認する。

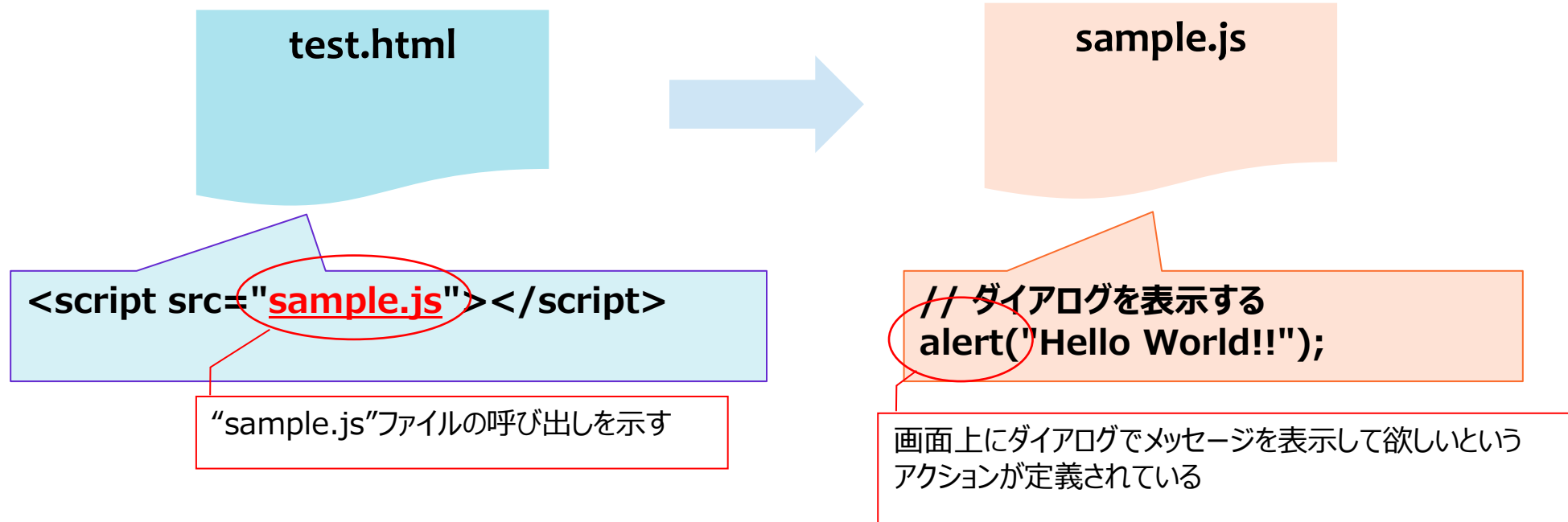


Step3 サンプルプログラム理解

作成したプログラム内容を理解する

前述の通り拡張子が「js」となっているものが、JavaScriptのプログラムとなります。

HTMLのファイルから、JavaScriptファイルを読み出し実行しているプログラムとなっています。



プログラム(JavaScript)の基本

基本となる用語を覚えてみましょう！

変数

“変数”とは値を入れておく箱のようなもの。変数を使うための準備を行う必要がありますが、プログラムでは以下のように記述します。

```
var name = "Ios Tarou";  
alert(name);
```

alert関数で変数**name**の内容を確認する

変数 **name** という変数に、
“Ios Tarou” という文字列を
登録する。

基本となる用語を覚えてみましょう！

関数

ある処理をまとめたプログラムのこと。まとめた処理を何度も使う際に、簡単に使い回しができるため有効に使えます。

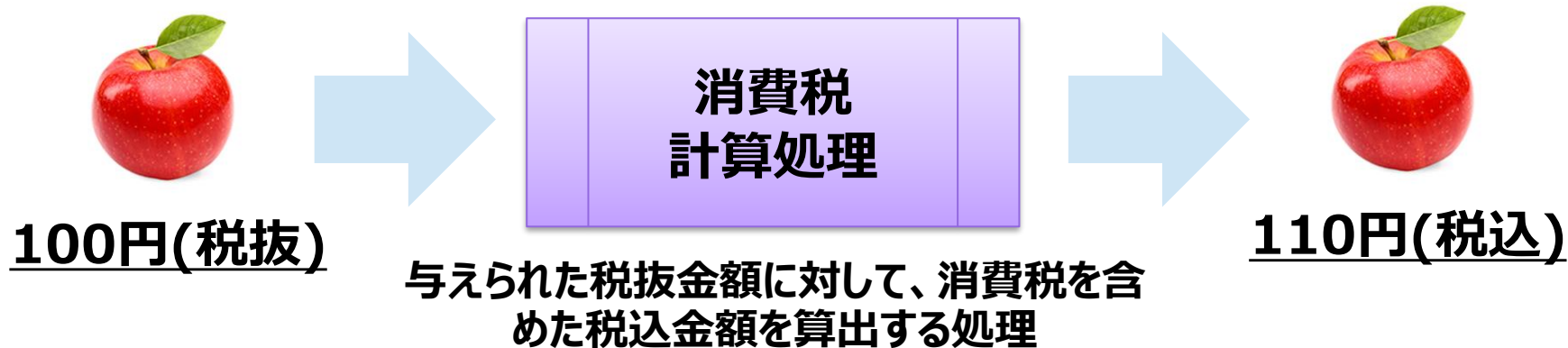
引数

関数を呼び出すときに一緒に渡す値のことで、関数の中でその値を利用することができます。

戻り値

関数から値（処理結果など）を呼び出し元に返すことができます。

プログラム（JavaScript）の基本



引数

関数

戻り値

```
// 関数を定義
function tax(price) {
  var taxout = price * 110/100;
  return taxout;
}
// 関数の呼び出し
alert(tax(100)); // 100円に対する税込価格を表示
```

プログラムの基本構文

基本となるプログラム構文を覚えてみましょう！

条件分岐① if文

「雨だったら傘を持っていき、雨じゃなかったら手ぶらで出かけよう」などの様に、もしも〇〇だったら“処理A”を、違ったら“処理B”を実行する場合など、条件により処理を分けるときに使用します。

```
// 条件分岐プログラムの例
if(今日の天気 == 雨) {
    傘を持っていく      //条件式が 真 (true) のとき実行
} else {
    手ぶらで出かける    //条件式が 偽 (false) のとき実行
}
```

条件分岐② switch文

もしも値が、

○だったら “処理A”

△だったら “処理B”

□だったら “処理C”

○△□でもなかったら “処理D”

のように、値によって処理を変えるとき
に使用します。

※if文で記載することも可能

breakはswitch処理を
抜ける事を意味する

```
// 条件分岐プログラムの例
switch(今日のおやつ){
  case ケーキ:
    とても喜んで食べる
    break;

  case ミカン:
    喜んで食べる
    break;

  case なし:
    喜びがない
    break;

  default :
    何もしない
}
```


繰り返し① for文

決められた回数だけ処理を繰り返します。

// 繰り返し処理のプログラム例

```
for ( var count = 0; count < 10; count++ ) {
```

// 変数countの内容を表示する

alert(count);

}

繰り返し処理をする度に行われる
処理

※以下①②は同じ意味

①count++

②count = count + 1

繰り返し処理を実行する条件

繰り返し処理の前に実行される
処理（初期設定）

配列（配列変数）

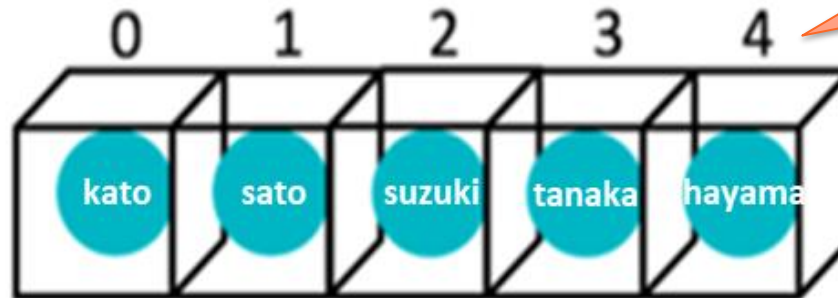
情報を記憶しておくための箱である変数の一つで、複数の箱が連なっているものをいいます。一つの変数の中に、複数の値を入れることが出来る、つまり複数の器を一つのかたまりとして扱うことができるものをいいます。

変数



変数は 1 つのデータしか
格納できない

配列変数



配列変数は、必要に応じて
複数のデータが格納できる

配列（配列変数）

配列変数の内容は以下の通りです。

```
var _3nen5kumi = ['kato','sato','suzuki','tanaka','hayama'];
```

```
// 1 番目の要素
```

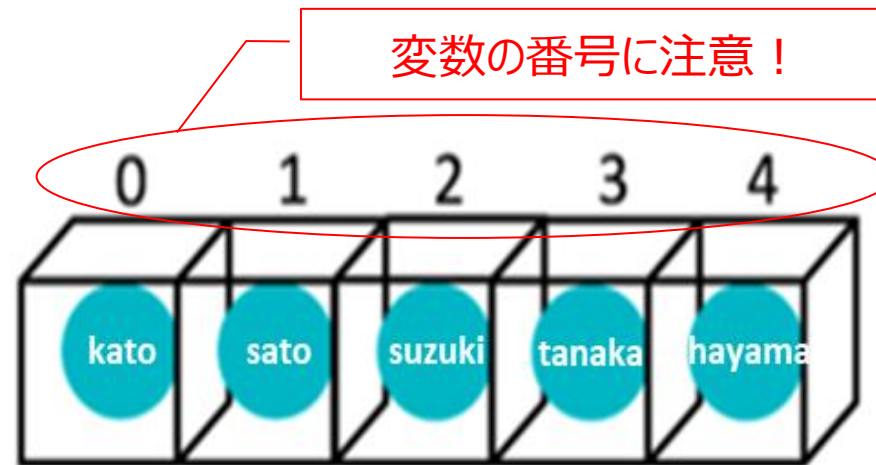
```
alert(_3nen5kumi[0]);
```

```
// 4 番目の要素
```

```
alert(_3nen5kumi[3]);
```

```
// 要素は0から始まり、[]の中の数字で取り出したり、設定したりすることができる。
```

```
_3nen5kumi[2] = 'ito';
```



配列（配列変数）

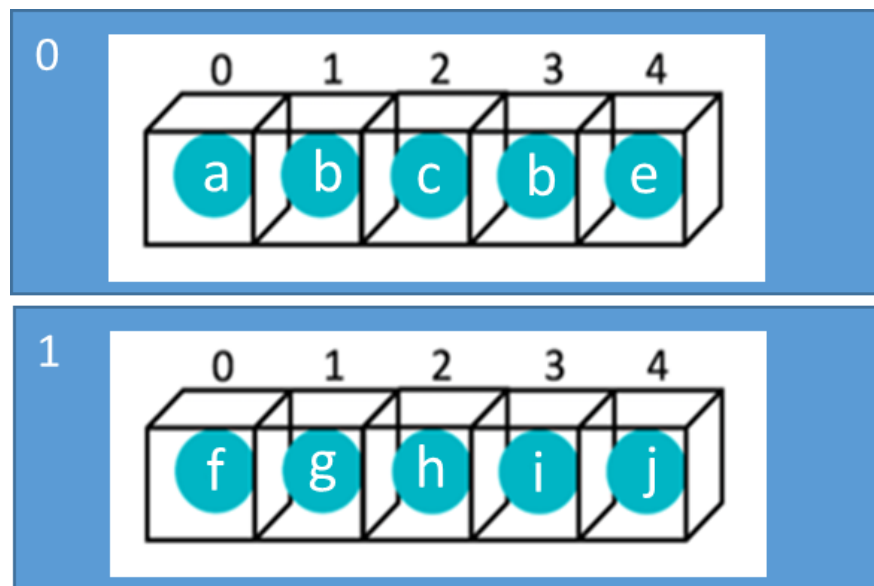
二次元配列変数の内容は以下の通りです。

```
// 二次元配列変数を宣言
```

```
var ary = [  
    [a, b, c, d, e],  
    [f, g, h, i, j],  
];
```

ary[0][2] は「 c 」を表します。

ary[1][0] は「 f 」を表します。



繰り返し② while文

繰り返し回数が決まっていない繰り返し処理に最適です。条件を満たす間処理を実行します。

```
// 繰り返し処理プログラム例
```

```
var array = ['Apple', 'Banana', 'Strawberry', 'Meron'];
```

```
var i = 0;
```

```
while(i<array.length) {
```

```
    alert(array[i++]);
```

```
}
```

“array.length” は配列に含まれる値の数 (=4) を示す。

環境準備

Visual Studio Code ソースエディタ

JavaScriptのプログラムを書くため、今回はVisual Studio Codeというソースエディターを使用します。メモ帳などを使って書くこともできますが、変数や関数など色分けされており、視覚的に見やすい利点があります。以下に Visual Studio Code インストール手順を記載します。

手順1 ダウンロード用ウェブページを表示する

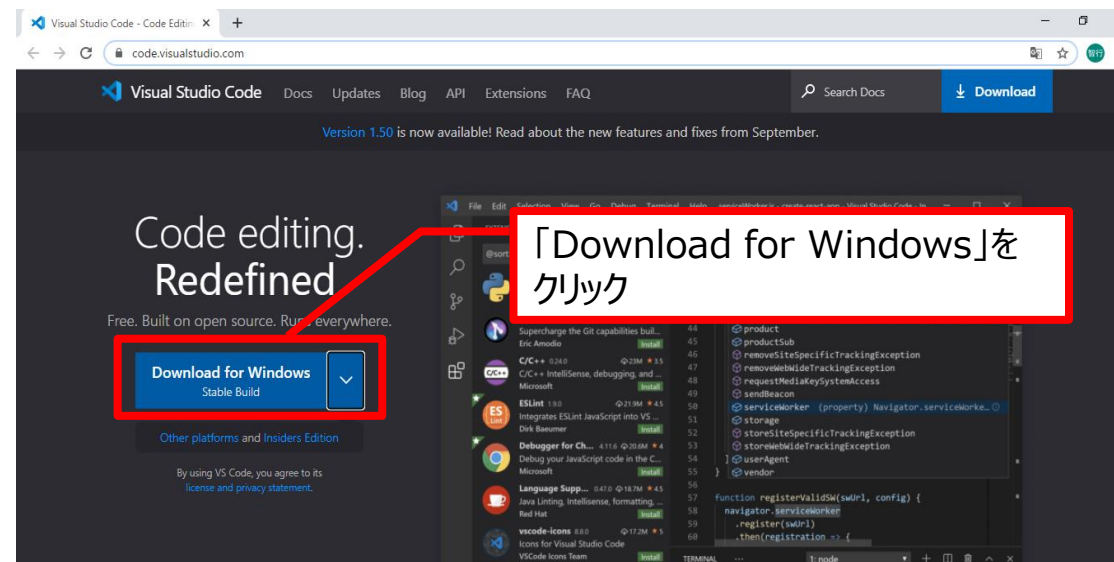
- ①Webブラウザを起動する。
- ②下記URLを入力する。

<https://code.visualstudio.com/>

手順2 Visual Studio Code をダウンロードする

- ③「Download for Windows」をクリックする。

※クリック後ファイルがダウンロードされます



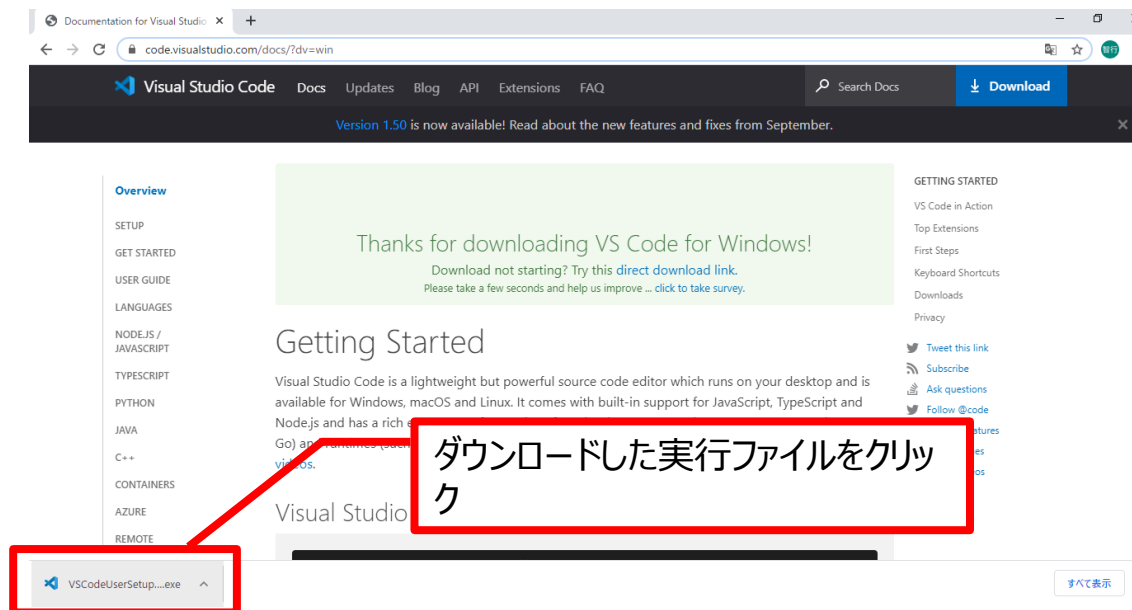
※使用するパソコンのOSにより、ダウンロードするファイルが自動的に選ばれるためMacPCの場合、「Download for Mac」となる

Visual Studio Code ソースエディタ

手順3 ダウンロードしたファイルを実行する

④ダウンロードした実行ファイルをクリックする。

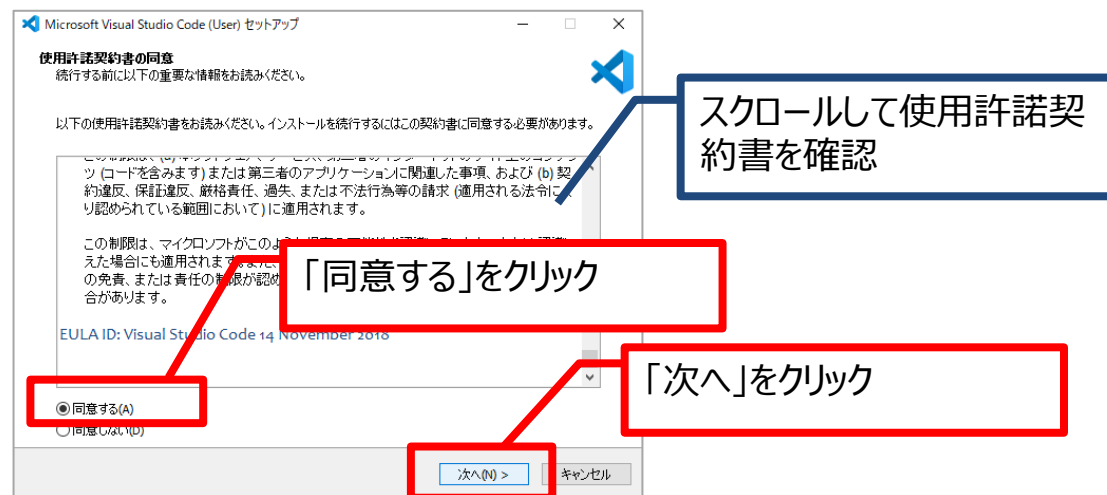
※パソコンの設定に応じて、「ユーザーアカウント制御」ダイアログボックスが表示されるため「はい」を選択



Visual Studio Code ソースエディタ

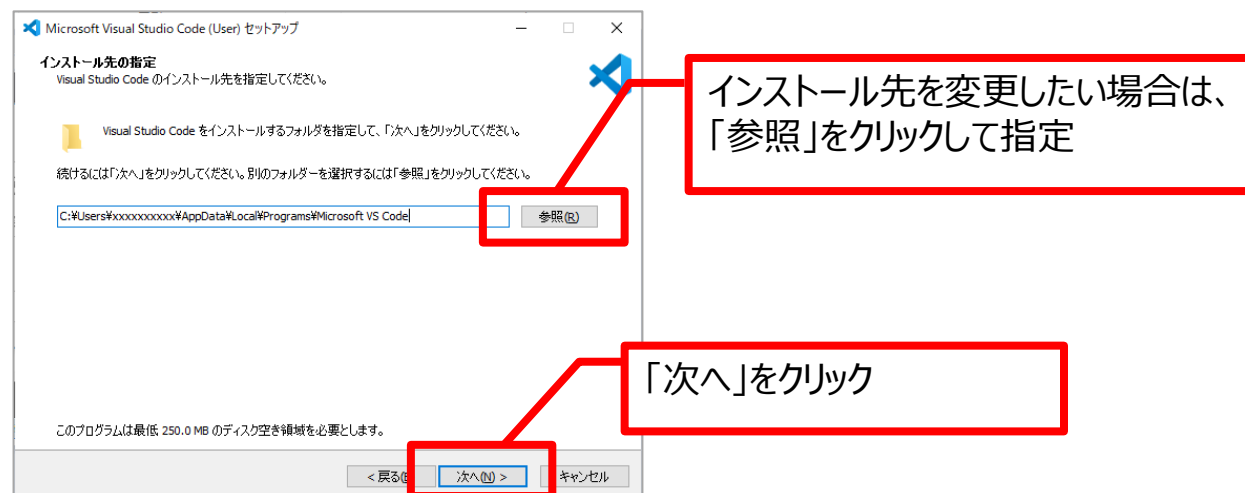
手順4 セットアップウィザードを開始する

- ⑤使用許諾契約書の内容を確認、同意の上
「次へ」をクリックする。



手順5 インストール先を指定する

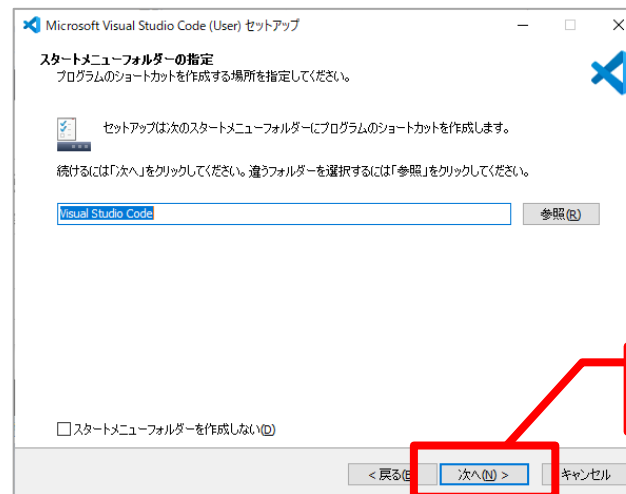
- ⑥インストール先を指定する。



Visual Studio Code ソースエディタ

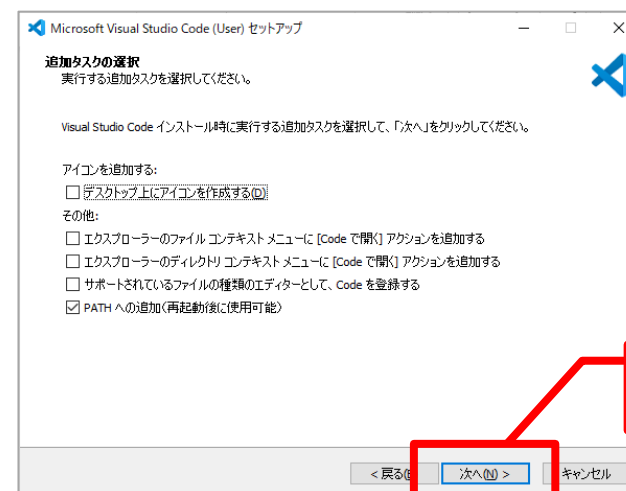
手順 6 スタートメニューフォルダーの指定する

⑦「次へ」をクリックする。 ※変更しない



手順 7 追加タスクを選択する

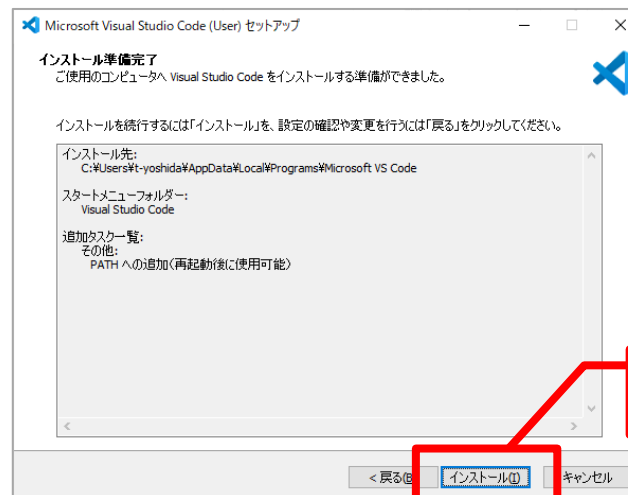
⑧「次へ」をクリックする。 ※変更しない



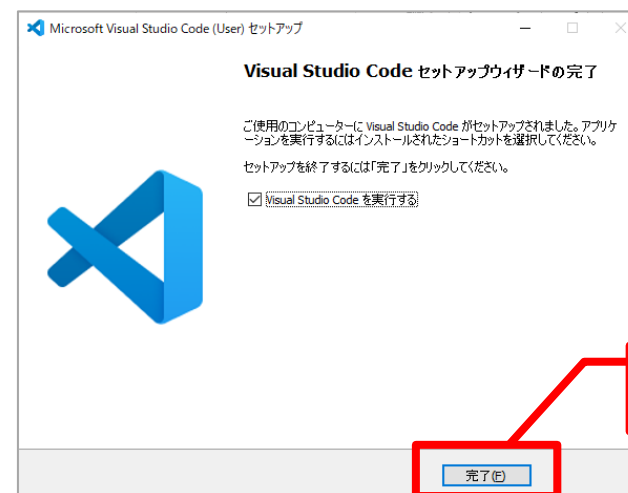
Visual Studio Code ソースエディタ

手順8 インストールを完了する

- ⑨「インストール」をクリックする。
- ⑩インストール完了後にダイアログが表示されるため「完了」をクリックする。



「インストール」をクリック

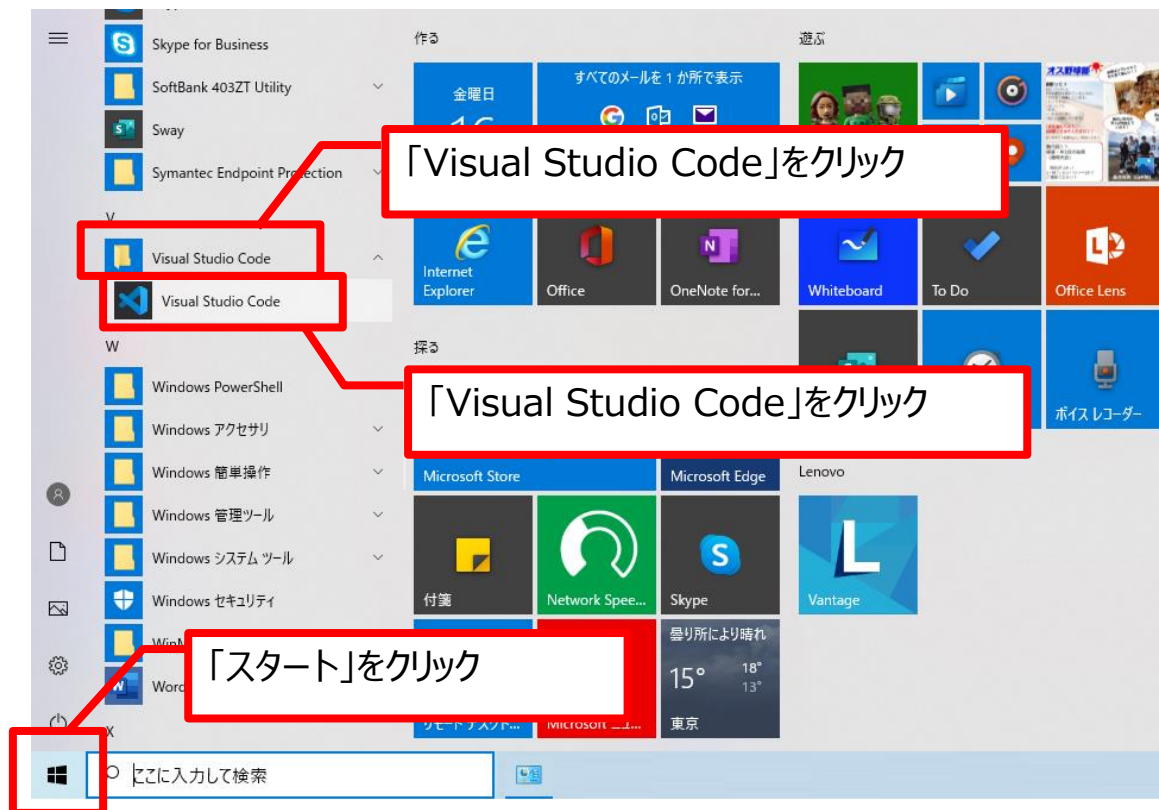


「完了」をクリック

Visual Studio Code のセットアップ

手順1 Visual Studio Code を起動する

- ①「スタート」をクリックする。
- ②「Visual Studio Code」をクリックする。
- ③「Visual Studio Code」をクリックする。



Visual Studio Code のセットアップ

手順2 Visual Studio Code の言語設定

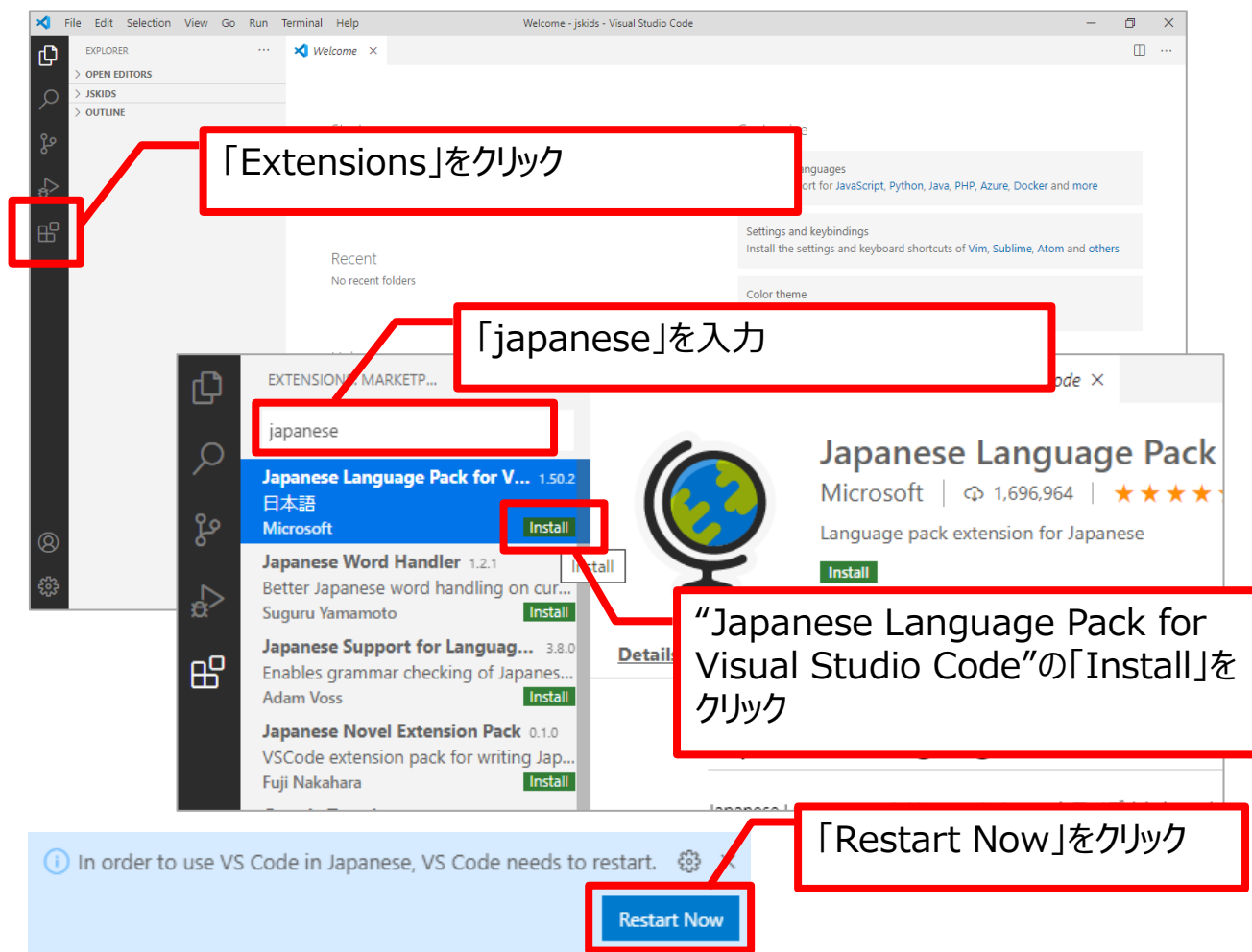
④画面右下に“表示言語を日本語に変更するには「言語パックをインストールします」というダイアログが表示されている場合、「インストールして再起動」をクリックし、⑤～⑧の手順をスキップする。ダイアログが表示されていない場合は、以下⑤～⑧を実行する。

⑤画面左メニューから「Extensions」をクリックする。

⑥検索用テキストエリアに「japanese」を入力する。

⑦“Japanese Language Pack for Visual Studio Code”の「Install」をクリックする。

⑧画面右下に表示される「Restart Now」をクリックする。

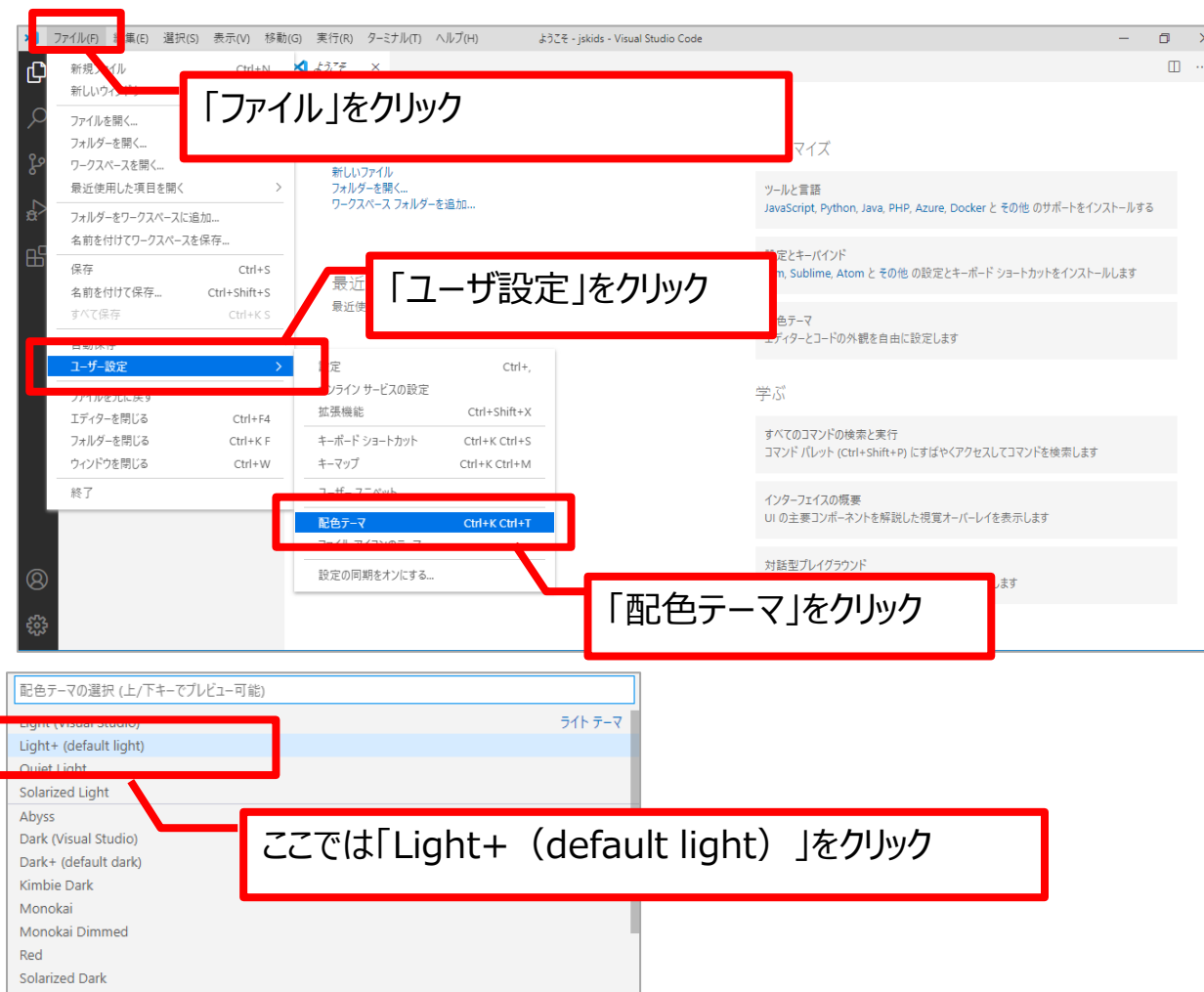


Visual Studio Code のセットアップ

手順3 Visual Studio Code の配色テーマ設定

- ⑨「ファイル」をクリックする。
- ⑩「ユーザ設定」をクリックする。
- ⑪「配色テーマ」をクリックする。
- ⑫「Light+ (default light)」をクリックする。

※配色については好みの内容を選択する



Visual Studio Code のセットアップ

手順4 Visual Studio Code のその他設定

⑬「起動時にウェルカムページを表示」をクリックしてチェックを外す。

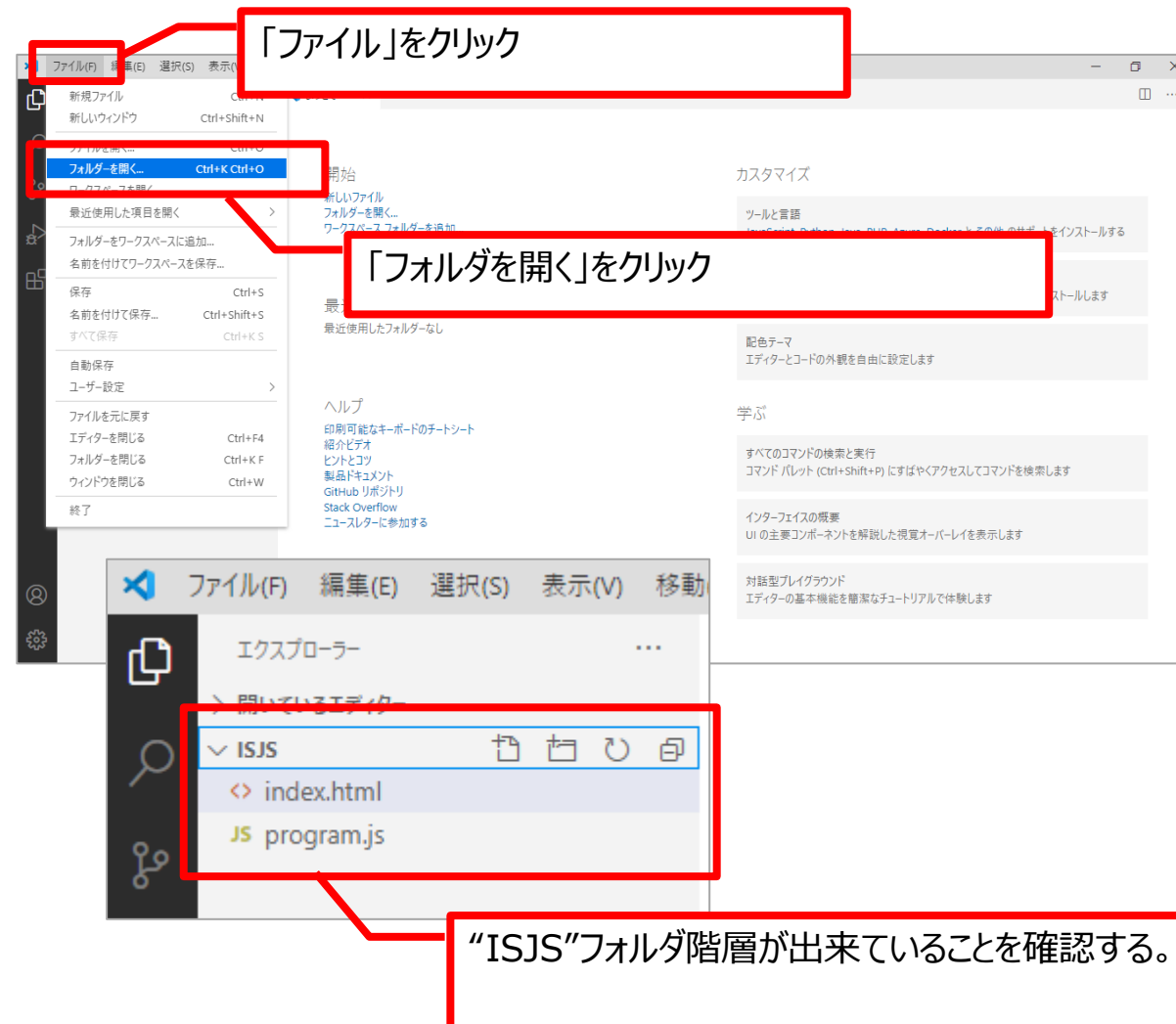


「起動時にウェルカムページを表示」をクリックしチェックを外す
※表示されない場合もあります

Visual Studio Code のセットアップ

手順5 Visual Studio Code フォルダ読み込み

- ⑭「ファイル」をクリックする。
- ⑮「フォルダを開く」をクリックする。
- ⑯デスクトップ上に作成している“isjs”フォルダを指定し、メニュー上に指定したフォルダが表示されることを確認する。



プログラム演習②

プログラム演習を始める前に！

プログラム言語の種類により異なりますが、プログラム言語には細かい記載ルールが存在します。
本日体験するJavaScriptプログラムでは、以下に掲げる内容に注意して進めていきましょう。

- ・全角空白は使わないこと
- ・大文字小文字を意識すること
- ・カッコ文字は、開始と終了が対になっていること
- ・カッコ文字は、種類により意味合いがことなるため { ([]) } の違いを意識すること
- ・シングルクォーテーション(')やダブルクォーテーション(")の違いを意識すること

演習課題

・配布した落ち物パズルのプログラムには以下の機能が実装されていません。

①ブロックの横移動機能

②ブロックの回転機能

これらの機能を作成し、開発作業を経験してみましょう。

(1) 右方向キー (→) を押したときに“alert”関数を利用してメッセージを表示しましょう。
メッセージの内容は何でも構いません。

【ヒント】

方向キーを判別する方法については
右表が参考になります。次ページの
説明も読んで考えてみてください。

方向キー	判別方法
→	event.keyCodeが39
←	event.keyCodeが37
↑	event.keyCodeが38
↓	event.keyCodeが40

演習課題

```
function ugokasu(e) {
```

```
    // (省略)
```

```
    // 現在の座標と向きを保存
```

```
    maenoix = ix;
```

```
    maenoiy = iy;
```

```
    maenoimuki = imuki;
```

```
    // いまのブロックを消す
```

```
    kesu(cg, ix, iy, imuki, ishurui);
```

※ ※ ※ ※ ※ **ここにプログラムを追加** ※ ※ ※ ※ ※

```
    if (e.keyCode == 40) {  
        shिताidou();  
    }
```

次ページに“答え”があります

332～334行目の間に
プログラムを追加する

“メッセージを表示する”は
alert関数 を利用する

下方方向キーを押下した際に“shिताidou()”
関数を呼び出す処理を示す

演習課題

```
function ugokasu(e) {  
    // (省略)  
  
    // いまのブロックを消す  
    kesu(cg, ix, iy, imuki, ishurui);  
  
    // [→] キーが押されたかどうか  
    if (e.keyCode == 39) {  
        // メッセージ表示  
        alert("右が押されました。");  
    }  
  
    if (e.keyCode == 40) {  
        shिताidou();  
    }  
}
```

If文の判定処理はイコール記号が
2つ必要であることに注意 (“==”)

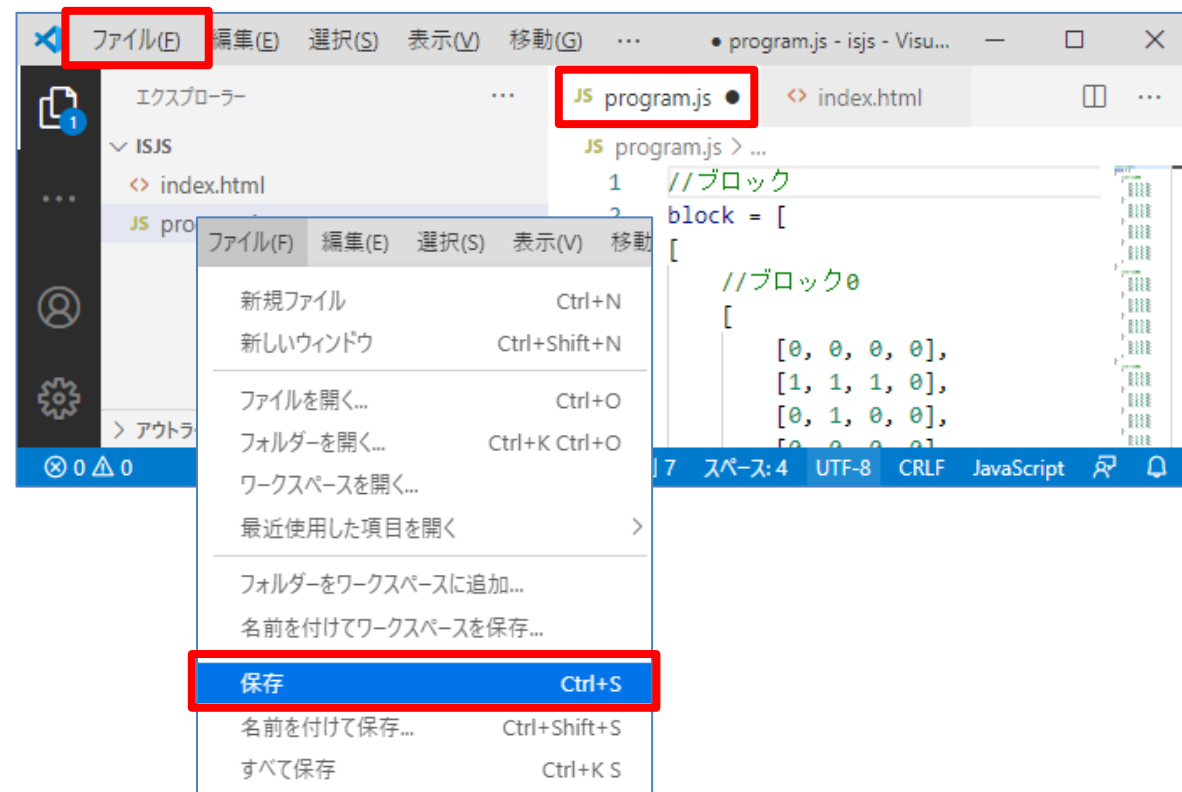
プログラム言語によって異なりますがJavascript言語では、
大文字小文字・全角半角を区別します。
また、実行するプログラム行の末尾にはセミicolon[;]が
必要になるため入力忘れに注意が必要です。

※重要 作業の効率化について

新しい業務を行う中で、業務効率化・工夫を考えられるかはとても重要です。システムエンジニアの業務においてもとても重要な要素になります。ここでは本日の作業を進める上で「知っておくと良いこと」をお伝えします。

ポイント1 ファイル保存

テキストを更新するとタブに“●”が表示されます。この状態は更新内容が反映されていないことを示します。ファイルを保存する際、「ファイル」メニューを開き「保存」をクリックすることでファイルへの保存が出来ますが、この操作はキーボードで“Cntr + S”を入力することでも同様の処理が行われます。



※重要 作業の効率化について

新しい業務を行う中で、業務効率化・工夫を考えられるかはとても重要です。システムエンジニアの業務においてもとても重要な要素になります。ここでは本日の作業を進める上で「知っておくと良いこと」をお伝えします。

ポイント2 ウェブブラウザの更新

プログラムファイルの更新を、ウェブブラウザ（ゲーム画面）に反映したい場合、右図の通り更新ボタンをクリックすることで反映することが可能になります。またこの操作はキーボードで “F5” を入力することでも同様の処理が行われます。



※重要 作業の効率化について

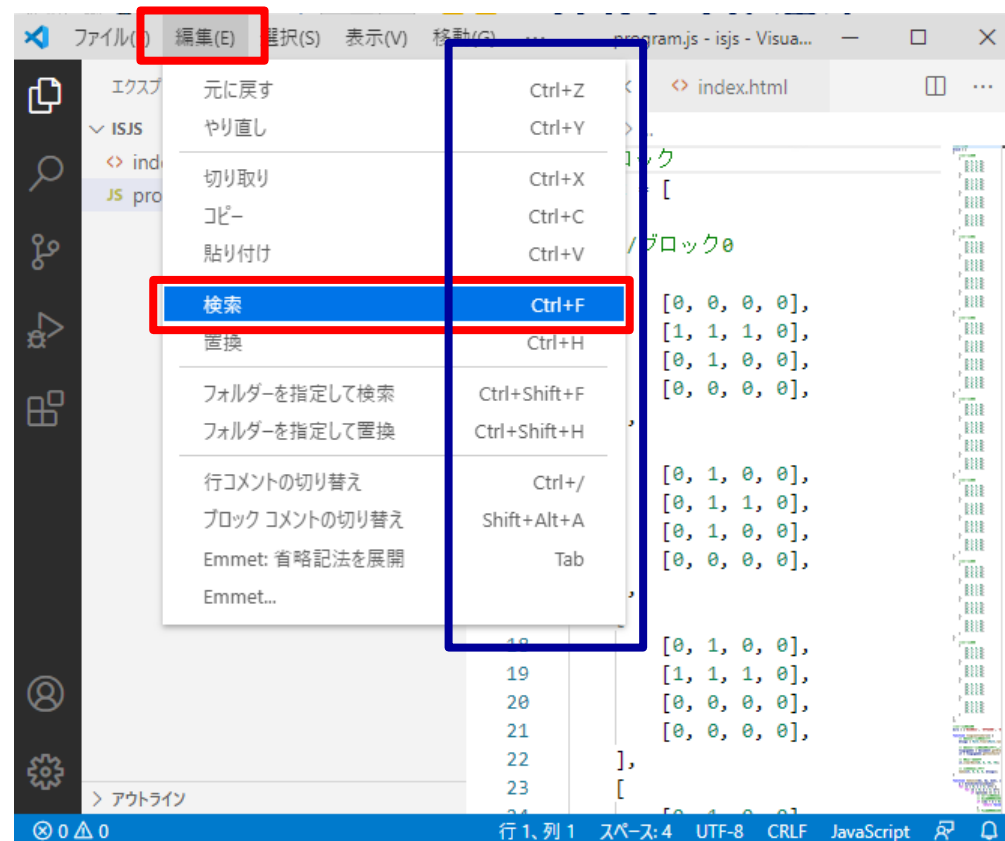
新しい業務を行う中で、業務効率化・工夫を考えられるかはとても重要です。システムエンジニアの業務においてもとても重要な要素になります。ここでは本日の作業を進める上で「知っておくと良いこと」をお伝えします。

ポイント3 ファイル検索

プログラムファイルの何れかのキーワードで検索する際、「編集」メニューを開き「検索」をクリックすることで、検索ダイアログが表示されますが、この操作はキーボードで“Cntr + F”を入力することでも同様の処理が行われます。

ポイント4 ショートカットキー

殆どのアプリケーションで“ショートカットキー”というものが登録されています。メニューバーに書かれているショートカットキーを意識できるようになると良いかと思います。



演習課題

・配布した落ち物パズルのプログラムには以下の機能が実装されていません。

①ブロックの横移動機能

②ブロックの回転機能

これらの機能を作成し、開発作業を経験してみましょう。

(2) 「→」を押したときは「右が押されました。」、「←」を押したときは「左が押されました。」というメッセージを表示しましょう。

演習課題

・配布した落ち物パズルのプログラムには以下の機能が実装されていません。

- ①ブロックの横移動機能
- ②ブロックの回転機能

これらの機能を作成し、開発作業を経験してみましょう。

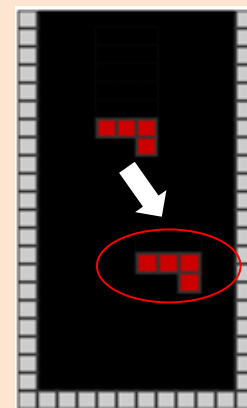
(3) 「→」を押したときはブロックを一つ右に、
「←」を押したときはブロックを一つ左に移動させましょう。

【ヒント】

場所の情報は、“ix” という変数で管理しています。

ixの値を +1 すると右に1つ、-1 すると左に1つ移動します。

次ページに答えが書いてあります。答えを見ずに、前述の
“プログラムの基本構文”内容を参考にしながら考えてみましょう。



ixの値を +1 した場合
の移動

演習課題

```
function ugokasu(e) {  
    // (省略)  
  
    // いまのブロックを消す  
    kesu(cg, ix, iy, imuki, ishurui);  
  
    // [→] キーが押されたかどうか  
    if (e.keyCode == 39) {  
        // 右に移動  
        ix = ix + 1;  
    }  
  
    // [←] キーが押されたかどうか  
    if (e.keyCode == 37) {  
        // 左に移動  
        ix = ix - 1;  
    }  
}
```

演習課題

・配布した落ち物パズルのプログラムには以下の機能が実装されていません。

①ブロックの横移動機能

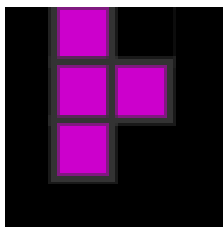
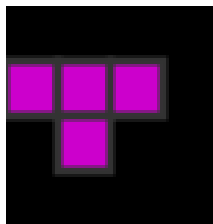
②ブロックの回転機能

これらの機能を作成し、開発作業を経験してみましょう。

(4) 「↑」を押したときにブロックを左回転させましょう。

演習課題

```
block = [  
  [  
    //ブロック0  
    [  
      [0, 0, 0, 0],  
      [1, 1, 1, 0],  
      [0, 1, 0, 0],  
      [0, 0, 0, 0],  
    ],  
    [  
      [0, 1, 0, 0],  
      [0, 1, 1, 0],  
      [0, 1, 0, 0],  
      [0, 0, 0, 0],  
    ],  
  ],  
  // (省略)
```



JavaScriptファイル（program.js）の1～192行目で配列blockの内容が左記のように定義されています。

ブロックの向き（配列番号）がどの変数で定義されているかを探してみましょう！ 見つけれない場合には、次ページを参照してみましょう。

演習課題

・配布した落ち物パズルのプログラムには以下の機能が実装されていません。

①ブロックの横移動機能

②ブロックの回転機能

これらの機能を作成し、開発作業を経験してみましょう。

(4) 「↑」を押したときにブロックを左回転させましょう。

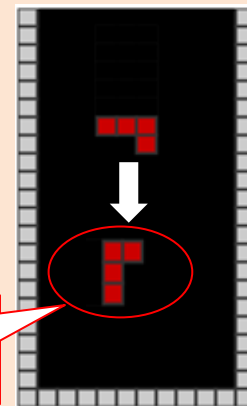
【ヒント】

ブロックの向きは、“imuki” という変数で管理しています。

“imuki”の値を +1 すると左周りに90度回転します。

回転する仕組みについては次ページで説明します。

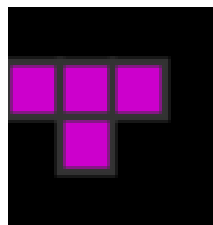
imukiの値を +1 した場合の回転



演習課題

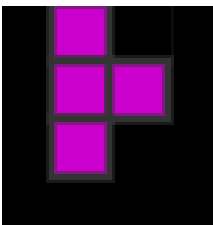
imuki が 0 の場合

```
[0, 0, 0, 0]  
[1, 1, 1, 0]  
[0, 1, 0, 0]  
[0, 0, 0, 0]
```



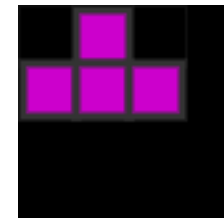
imuki が 1 の場合

```
[0, 1, 0, 0]  
[0, 1, 1, 0]  
[0, 1, 0, 0]  
[0, 0, 0, 0]
```



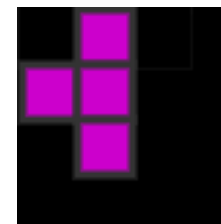
imuki が 2 の場合

```
[0, 1, 0, 0]  
[1, 1, 1, 0]  
[0, 0, 0, 0]  
[0, 0, 0, 0]
```



imuki が 3 の場合

```
[0, 1, 0, 0]  
[1, 1, 0, 0]  
[0, 1, 0, 0]  
[0, 0, 0, 0]
```



変数“imuki” が 4 になると定義されていないため・・・！？

次ページに半分答えが書いてあります。

演習課題

```
function ugokasu(e) {
```

```
  // (省略)
```

```
  // [↑] キーが押されたかどうか  
  if (e.keyCode == 38) {  
    // 回転する  
    imuki = imuki + 1;
```

ブロックの向きを示す
変数 "imuki" の値を変更する

※ ※ ※ ※ ※ ここにプログラムを追加 ※ ※ ※ ※ ※

```
}
```

```
  // (省略)
```

次ページに"答え"があります

ブロックの向きは0～3までしか
定義がないため更に工夫が必要

演習課題

```
function ugokasu(e) {
```

```
    // (省略)
```

```
    // [↑] キーが押されたかどうか
```

```
    if (e.keyCode === 38) {
```

```
        // 回転する
```

```
        imuki = imuki + 1;
```

```
        if (imuki > 3) {
```

```
            imuki = 0;
```

```
        }
```

```
    }
```

```
    // (省略)
```

ブロックの向きは0～3までしか
定義がないため3から0に戻る

追加課題

・落ち物パズルのプログラムには以下の機能を更に実装してみましょう。

①ブロックの一気下移動機能

(5) ブロックが回転するキーを「↑」から「z」に変更しましょう。

(6) 「↑」を押したときにブロックを一番下まで移動させましょう。

【ヒント】

「z」キーを判別する方法については
右表が参考になります。

(6)については、次ページで説明します。

アルファベットキー	判別方法
z	event.keyCodeが90

追加課題

```
function shिताidou_ikki() {  
  // 一番下まで移動する  
  // 描く先のCanvasを取得  
  gamegamen = document.getElementById('game');  
  cg = gamegamen.getContext('2d');  
  
  // 操作対象のブロックを消す  
  kesu(cg, ix, iy, imuki, ishurui);  
  
  // どこまで下に移動できるかどうかを確認する  
  while(true){  
  
    // どこまで下に移動できるかどうかを確認する  
    kekka = kakunin(ix, iy, imuki, ishurui);
```

ブロックを一番下まで移動する関数を
新規作成する

画面表示の事前準備

“kakunin”関数の結果で、下に移動できる・できないを
“kekka”変数に登録する

※次ページに続く

追加課題

```
// 移動できる場合、できない場合で処理を分ける
if (kekka) {
    // 移動できる場合は、位置を下げて更に確認する
    ※ ※ ※ ※ ※ ここにプログラムを追加 ※ ※ ※ ※ ※

} else {
    // 移動できない場合、ひとつ手前の繰り返し位置にブロックを描く
    ※ ※ ※ ※ ※ ここにプログラムを追加 ※ ※ ※ ※ ※
    // 移動できる位置の最下部に描く
    kaku(cg, ix, iy, imuki, ishurui);
    break;
}

// 下移動関数を実行し、当たり判定と画面の更新を行う
shिताidou()
}
```

If 関数で下に移動できる・できないを
判断して処理を分ける

1行プログラムを追加する

1行プログラムを追加する

“break”でループ処理を抜ける

追加課題

```
function ugokasu(e) {  
    // (省略)  
  
    // いまのブロックを消す  
    kesu(cg, ix, iy, imuki, ishurui);  
  
    // [↑] キーが押されたかどうか  
    if (e.keyCode == 38) {  
        // 一番下まで移動させる  
        shिताidou_ikki();  
    }  
  
    if (e.keyCode == 40) {  
        shिताidou();  
    }  
}
```

ブロックを一番下まで移動する
“shिताidou_ikki”関数を呼び出す

プログラム演習、お疲れ様でした！本日覚えてもらいたかった内容は、プログラムにおける重要要素である以下 3 つとなりますが、ご理解いただけたでしょうか！？

①変数の使い方

```
ix = 3;  
ix = ix + 1;  
ix の値は！？
```

②条件分岐

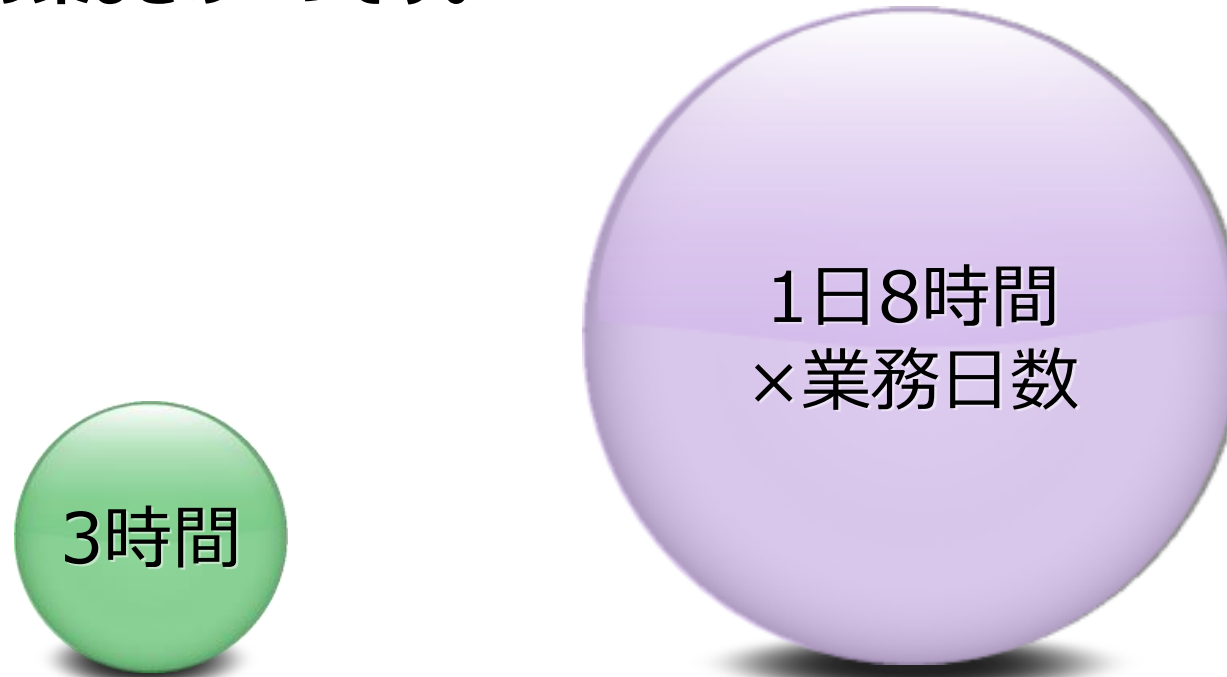
```
if ( 今日の天気 == 雨 ) {}
```

③繰り返し

```
for ( count = 0; count < 10; count++ ) {}
```

さいごに

僅か3時間程度の時間ではありましたが、プログラムについて何か覚えられたという実感はありますでしょうか。参加した方同士で会話は出来ましたでしょうか。最初は難しいと感じられた方も多いかと思いますが、日々の業務の中で知識と経験は必ず増えます。対応できることが増えていくこともまたこの職業の楽しさの一つです。



※本日はツール等利用しませんでした。ツールを使うと簡単にプログラム作成やエラー解析が可能となります。