# 応用情報処理 No. 7

## JavaScriptの制御文

プログラムは、上から順番に実行されるのが基本である。しかし、条件によって、処理をとばしたり、前に戻ったりしたい場合がある。この場合に、実行する順番を制御するための文が制御文である。プログラムの命令文[[1]](#footnote-1)の実行の順番を制御するには、基本的な制御構造を組み合わせることでそれが可能になる。構造化プログラム定理[[2]](#footnote-2)（structured program theorem）によれば、

1. 連続（または、順次）　　Π(*a*, *b*)、
2. 選択（または、条件分岐）Δ(α, *a*, *b*)、
3. 反復（または、繰り返し）Ω(α, *a*)

の三種類の基本制御構造の組み合わせで、あらゆるプログラムを作成できる。

これらの基本制御構造は、フローチャート（流れ図）で描くと、以下のようである。ここで、四角形　　　　はデータを加工する一連の機能を表し、ひし形　　　　　は条件判断に基づいた流れの分岐を表す。

*α*

*a*

*b*

*a*

*α*

*a*

*b*

Π(*a*, *b*)

Δ(α, *a*, *b*)

Ω(α, *a*)

*a*

*α*

真

真

偽

偽

JavaScriptでは、連続の制御構造をつくるには、命令文をセミコロン(;)で区切って実行したい順番に並べるだけである。プログラムを見やすくするには、セミコロンで改行すると良い。すると、上の行から順次、実行することになる。

　選択の制御構造をつくるには、if文、if～else～文、switch文などを使用する。

　反復の制御構造をつくるには、for文、while文、do/while文などを使用する。

## JavaScriptのif文

選択、または条件判断には、JavaScriptではif文を使う。書き方には、いくつかのバリエーションがある。

条件

JavaScript文

真

偽

if ( 条件 ) JavaScript文;

条件

JavaScript文1;

JavaScript文2;

 ．．．

JavaScript文n;

真

偽

if ( 条件 ) {

JavaScript文1;

JavaScript文2;

 …

JavaScript文n-1;

JavaScript文n;

}

if～else～文

条件

JavaScript文1;

JavaScript文2;

 ．．．

JavaScript文n;

真

偽

JavaScript文n+1;

JavaScript文n+2;

 ．．．

JavaScript文n+m;

if ( 条件 ) {

JavaScript文1;

JavaScript文2;

 …

JavaScript文n;

}else{

JavaScript文n+1;

JavaScript文n+2;

 …

JavaScript文n+m;

}

条件1

JavaScript文1;

 ．．．．．．

JavaScript文n;

真

偽

JavaScript文n+1;

 ．．．．．．

JavaScript文n+m;

条件2

条件3

JavaScript文n+m+1;

 ．．．．．．

JavaScript文n+m+k;

真

真

偽

偽

JavaScript文n+m+k+1;

 ．．．．．．

JavaScript文n+m+k+h;

If ( 条件1 ) {

JavaScript文1;

JavaScript文2;

 …

JavaScript文n;

}else if( 条件2 ){

JavaScript文n+1;

JavaScript文n+2;

 …

JavaScript文n+m;

}else if( 条件3 ){

JavaScript文n+m+1;

JavaScript文n+m+2;

 …

JavaScript文n+m+k;

}else{

JavaScript文n+m+k+1;

JavaScript文n+m+k+2;

 …

JavaScript文n+m+k+h;

}

## JavaScriptの比較演算子

二つの変数の値や、値そのものを比較して、真（true）か偽(false)を返す演算子を比較演算子という。

書き方は、比較演算子を☆という記号で代表させると、X ☆ Yというように左辺Xと右辺Yを真ん中で繋ぐ位置に書く。☆(X, Y)とは書かないことに注意すること。

|  |  |  |
| --- | --- | --- |
| 名前 | 記号（X ☆ Y） | 意味 |
| 等値演算子 | X == Y | XとYは等しい |
| 不等値演算子 | X != Y | XとYは等しくない |
| 小なり演算子 | X < Y | XはYより小さい |
| 大なり演算子 | X > Y | XはYより大きい |
| 小なりイコール演算子 | X <= Y | XはY以下である |
| 大なりイコール演算子 | X >= Y | XはY以上である |

## JavaScriptの論理演算子

複数の条件判断が必要な複雑な条件を表現するために、AND、OR、NOTという論理演算子（ブーリアン（Boolean）演算子とも言う）が使用できる。

X、Yで条件を表すとして、論理演算子の記号は以下のようである。

|  |  |  |  |
| --- | --- | --- | --- |
| 名前 | 記号 | 意味 | ブーリアンの意味 |
| 論理積演算子 | X && Y | X AND Y | XとYが同時に真のときのみ、真。それ以外は偽。 |
| 論理和演算子 | X || Y | X OR Y | XかYのどちらか一方でも真であれば、真。それ以外は偽。 |
| 論理否定演算子 | ! X | NOT X | Xが真なら偽、Xが偽なら真。 |

## 条件分岐の例題

JavaScriptの入力ダイアログを表示するprompt()関数を使って、誕生月を入力してもらうWebページを作成してみよう。ただし、誕生月であるから、入力された数値が、1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12か、それら以外かを判断して、正しくない数値の場合は「誕生月が正しくありません！」と表示されるようにしたい。

## Do! まずは、prompt()関数の使用例として、以下のソースコードの内容のHTMLファイルをif.htmlという名前で作成しなさい。

[if.html]

<html>

<head><title>if文</title></head>

<body>

<script type="text/javascript">

document.write('<h2>あなたの誕生月</h2>');

var month;

month = prompt( "あなたの誕生月を入力してください。(半角で)", "0" );

//処理a

document.write( 'あなたは' + month + '月生まれです。<br>' );

</script>

</body>

</html>

## 要求： 入力された値が、１～１２のどれか以外のときは、「誕生月が正しくありません！」と赤で表示されるようにしたい。

入力された値は変数monthに格納される。この値が、１～１２である条件（「条件１」）は数学では1≦ month ≦１２というように表現するが、これは、（1≦ month） AND （month ≦１２）という意味である。従って、「条件１」は

条件１

処理a

真

偽

処理b

( 1 <= month ) && (month <= 12 )

あるいは、

1 <= month && month <= 12 　（&&は優先順位が低いため、括弧を省略可）

と書くことができる。

処理aは、この条件が満たされたときのみに行い、そうではないときは、「誕生月が正しくありません！」と表示する（これを処理bとする）ようにすればよい。この部分の流れ図は前頁の図のようになる。従って、if～else文を使うと良い。

## Do! if.htmlの処理aの部分を、if～else文を使って以下のように変更し、上の要求を満たすようにしよう。

if.htmlの変更点

if( 1<= month && month <= 12){

 //処理a

 document.write('あなたは' + month + '月生まれです。<br>');

}else{

 //処理b

 document.write('<span style="color:red;">');

 document.write('誕生月が正しくありません！');

 document.write('</span><br>');

}

【重要！】プログラムの書き方の注意：

処理a、処理ｂのブロックは、下のように字下げ（Tabを挿入）すると、始まりの”{”と終わりの”}”の対応が見やすくなり、ブロック構造がわかりやすくなる。バグを見つけやすくなるように、わかり易すいプログラムを書くことが大事である。



先頭を揃える

Tabで字下げして同じブロック内で先頭を揃える

## 条件分岐の例題－バグの修正

上のやりかたでは、要求を完全には満たしていない。従って、バグのあるプログラムである。そのバグとは、たとえば、prompt()関数の入力データとして、小数3.14を入力すると、3.14月生まれになってしまうことである（実際に試してみよ！）。すなわち、変数monthが整数であることのチェックが必要である。これを「条件2」としよう。

変数の値が整数かどうかをチェックする方法で、我々のここまでの知識の範囲で行うとすれば、次のようにすれば良い。まず、monthを1で割ったときの余りを求める。もしこの余りが0でなければ、小数点以下が0ではないので、整数ではないことになる。（ただし、9.99999999999999999のように桁が多いと、丸め誤差のため、10と見なされてしまう！）

「条件2」をJavaScriptでどう表現すれば良いか、案を考えて検討してみよう。

案１．

 var amari;

 amari = month % 1;

として、「条件２」を amari == 0 とする。こうすると、「条件２」が真のときは変数monthの値は整数、偽のときは整数ではないので、if文の条件は、「条件1」 && 「条件2」となり、これは、

( 1 <= month ) && (month <= 12 ) && ( amari == 0 )

と書ける。

案2．

 新しい変数amariを導入しなくても、剰余計算の結果をそのまま「条件2」に使うことができる。変数amariを使わないで、if文の条件は

( 1 <= month ) && (month <= 12 ) && ( (month % 1) == 0 )

とできる。案２の方がシンプルで良い。演算の優先順位があるので、括弧を省略して

 1 <= month && month <= 12 && month % 1 == 0

と書いても良い。

## Do! 案２に従って、if.htmlのバグを修正しよう。そして、試しに3.14を入力して、「誕生月が正しくありません！」と表示されるかをチェックしなさい。

## 条件分岐の例題－バグの修正（最終）

残念ながら9.99999999999999999のように桁が多い数値が入力された場合、上の「条件2」だけでは不十分である。変数monthにprompt()関数の返り値として代入されるのは文字列である。9.99999999999999999のように桁数が多くなると、もともとコンピュータは有限の桁の数値しか扱えないので、数値としては10.0として計算されることになる。すると、9.99999999999999999を1で割った余りは、10.0を1で割ったときの余りと計算されてしまう。9.99999999999999999を10.0という数値としてしまう誤差は、丸め誤差と呼ばれていて、有限桁の2進数計算しかできないコンピュータでは避けることのできない誤差である。

この問題の解決方法は、monthに格納されるデータを文字列ではなく数値にしてしまうことである。そのために、JavaScriptの組み込み関数のNumber()関数を使うことにしよう。

Number(str)

文字列strを数値に変換する。アルファベットなど数値に変換できない文字列の場合は、NaNを返す。

var month;

month = prompt( "あなたの誕生月を入力してください。(半角で)", "0" );

month = Number(month);

のように、prompt()関数の次の行に1行追加するか、或いはもっと短くして、

var month = Number( prompt( "あなたの誕生月を入力してください。(半角で)", "0" ) );

のようにprompt()関数の返り値をNumber()関数の引数にし、更に、その返り値を変数宣言のvarの初期値とすれば、上のように3行を1行にまとめて書くことができる。

##  Do! Number()関数で、monthに数値データが格納されるように変更して、if.htmlのバグを修正しよう。そして、試しに9.99999999999999999を入力して、「誕生月が正しくありません！」、あるいは「あなたは10月生まれです。」と表示されるかをチェックしなさい。

## Do! 問題：日の入力

prompt()関数を使って、誕生日を変数dayに格納し、ページ表示するコードをif.htmlに追加しなさい。

尚、dayに対するif文の条件は、1≦ day ≦31の範囲の整数とする。

## Do! 問題：年の入力

prompt()関数を使って、誕生年を変数yearに格納し、ページ表示するコードをif.htmlに追加しなさい。

尚、yearに対するif文の条件は、1900≦ year ≦2017の範囲の整数とする。

## 西暦から曜日を求める（Zellerの公式）

西暦の年がyear、月がmonth、日がdayという変数に格納されているとき、この日付の曜日は、次のZellerの公式で計算できることが知られている。

// Zellerの公式

if( month == 1 || month == 2 ){

 year = year - 1;

 month = month + 12;

}

var wday = ( year + Math.floor(year / 4) - Math.floor( year / 100) + Math.floor(year / 400) + Math.floor((13 \* month + 8)/5) + day ) % 7;

ここで、次の関数（メソッド[[3]](#footnote-3)）を使用した：

Math.floor(x)

数値xの小数部分を切り捨てて、x以下の最大の整数を返す。

計算結果のwdayは7で割った余りなので、0, 1, 2, 3, 4, 5, 6のいずれかの値である。0は日曜日、1は月曜日、2は火曜日、3は水曜日、4は木曜日、5は金曜日、6は土曜日を意味する。

## Do! 以下のようにif.htmlにZellerの公式で曜日を計算できるようにコードを追加した。黒塗りの部分をコードに修正して、正しく動作するようにせよ。

[if.html]

<html>

<head>

<title>if文</title>

</head>

<body>

<h2>あなたな何曜日生まれ？</h2>

<script type="text/javascript">

document.write('<h2>あなたの誕生年</h2>');

var year = Number( prompt( "あなたの誕生年を入力してください。(西暦を半角で)", "0" ));

if( 1900<= year && year <= 2014 && year % 1 == 0 ){

 //処理a

 document.write('あなたは' + year + '年生まれです。<br>');

}else{

 //処理b

 document.write('<span style="color:red;">');

 document.write('誕生年が正しくありません！');

 document.write('</span><br>');

}

document.write('<h2>あなたの誕生月</h2>');

var month = Number( prompt( "あなたの誕生月を入力してください。(半角で)", "0" ) );

if( 1<= month && month <= 12 && month % 1 == 0 ){

 //処理a

 document.write('あなたは' + month + '月生まれです。<br>');

}else{

 //処理b

 document.write('<span style="color:red;">');

 document.write('誕生月が正しくありません！');

 document.write('</span><br>');

}

document.write('<h2>あなたの誕生日</h2>');

var day = Number( prompt( "あなたの誕生日を入力してください。(半角で)", "0" ) );

if( 1<= day && day <= 31 && day % 1 == 0 ){

 //処理a

 document.write('あなたは' + day + '日生まれです。<br>');

}else{

 //処理b

 document.write('<span style="color:red;">');

 document.write('誕生日が正しくありません！');

 document.write('</span><br>');

}

document.write('<p>' + year + '年' + month + '月' + day + '日は、');

// Zellerの公式

if( month == 1 || month == 2){

 year = year - 1;

 month = month + 12;

}

var wday = ( year + Math.floor(year / 4) - Math.floor( year / 100) + Math.floor(year / 400) + Math.floor((13 \* month + 8)/5) + day ) % 7;

var youbi = '×';

if( wday == 0 ){

 youbi = '日';

}else if( wday == 1 ){

 youbi = '月';

}else if( wday == 2 ){

 youbi = '火';

}else if( wday == 3 ){

 youbi = '水';

}else if( wday == 4 ){

 youbi = '木';

}else if( wday == 5 ){

 youbi = '金';

}else if( wday == 6 ){

 youbi = '土';

}

document.write( youbi + '曜日です。</p>');

</script>

</body>

</html>

## 課題07（大の月、小の月のチェック）

1月は31日まで、2月は29日まで、3月は31日まで、4月は30日まで、5月は31日まで、6月は30日まで、7月は31日まで、8月は31日まで、9月は30日まで、10月は31日まで、11月は30日まで、12月は31日まで、として、誕生日の範囲のチェックを行うように、if.htmlを修正しなさい。

Hint: 29日までの月は2月のみ。30日までは、4月と6月と9月と11月。それら以外は31日まで。変数monthの値に従って、29か、30か、31という月末の日をgetumatuという変数に代入する。これは、if～else if ～else文を使って、

var getumatu;

if( month == 2 ){

 getumatu = 29;

}else if( month == 4 || month == 6 || month == 9 || month == 11 ){

 getumatu = 30;

}else{

 getumatu = 31;

}

とすればよい。

　そして、日付の条件を

( 1 <= day ) && (day <= getumatu ) && ( (day % 1) == 0 )

と修正すると良い。

## 12. 【参考】Dateオブジェクトの利用

課題07では、簡単のため、閏年かどうかのチェックはしていないので、2月は29日までとしている。（株）アンク著『Perlの絵本　Perlが好きになる9つの扉』（翔泳社、2005/7/14）のコラム（p. 32）に書かれている閏年の説明を応用して、閏年のチェックをできるようにしてみることにチャレンジして見ると良い。閏年のチェックもできるサンプルを、http://www.ipc.hokusei.ac.jp/~z00328/adjyou/Date\_obj\_simple.html に用意したので、同じ曜日が得られるか、動作チェックに利用すると良い。

Date\_obj\_simple.htmlは、課題07とは異なるソースコードで作成されている。JavaScriptで日付を扱うときは、JavaScriptのDateオブジェクトを利用するのが普通である。例えば var birthday = new Date(1940, 10, 9);を実行すると、birthdayは1940年10月9日0時0分0秒を表すDateオブジェクトとして定義される。すると、Zellerの公式を知らなくても、birthday.getDay()メソッドで曜日を計算してくれる。

Javascriptでは、日付と時間を1970年1月1日の0時0分0秒からミリ秒単位で記録したものとして扱う。birthday.getTime()メソッドは、この経過時間をミリ秒単位で返す。

Date\_obj\_simple.htmlでは、入力されたyear年1月1日に、Dateオブジェクトのbirthdayを初期化して、birthday.setMonth(month - 1)で入力された月のmonthに無理やり設定しなおし、birthday.setDate(day)で入力された日のdayに無理やり設定しなおしている。無理やり設定すると、1970年1月1日0時0分0秒からのミリ秒単位の経過時間に変換されるので、正しくない場合は、birthday.getFullYear()、birthday.getMonth()、birthday.getDate()の戻り値が入力された値と食い違ってしまう。この食い違いがないかあるかで、正しい日付か否かをチェックするというプログラムになっている。

## 13. Check List

* if文の書き方（分岐の条件、条件が真である場合の処理は、各々どこにどのように書くか）を理解したか？
* if～else文の書き方（分岐の条件、条件が真である場合の処理、条件が偽である場合の処理は、各々どこにどのように書くか）を理解したか？
* 比較演算子にはどのようなものがあるか？
* JavaScriptで、論理演算子（ブーリアン演算子）としてどのような記号を使うか？
* Number()は何をする関数か？
* Math.floor()は何をする関数か？
1. ここで命令文とは、str = ‘こんにちは’、c = a/b、document.write(‘Hello’)と言った、代入や演算、関数呼び出し等を意味する。 [↑](#footnote-ref-1)
2. Bohm, C. and G. Jacopini, "Flow Diagrams, Turing Machines and Languages with Only Two Formation Rules", *Communications of the ACM* **9** (1966) 366–371. [↑](#footnote-ref-2)
3. “メソッド”という言葉は、オブジェクト指向と呼ばれるプログラミングスタイルで使われる言葉である。オブジェクト指向では、あるもの（=オブジェクト）に関連する変数や関数を、そのオブジェクト名と変数名、或いは関数名をドット（.）で結んで表現する。引数xの小数点以下を切り捨てる関数の名をfloorとしているが、Javasriptでは、これはMathオブジェクトのメソッドとして定義されているので、Math.floor(x)と書かなければならない。因みに、オブジェクトと関連する変数は、そのオブジェクトのプロパティと呼ばれる。 [↑](#footnote-ref-3)