The background features a grayscale profile of a man's head and shoulders, facing left. Overlaid on his hair is a circular arrangement of code snippets in various programming languages, including Python, JavaScript, and Java. The main title is written in large, bold, black Japanese characters across the center of the image.

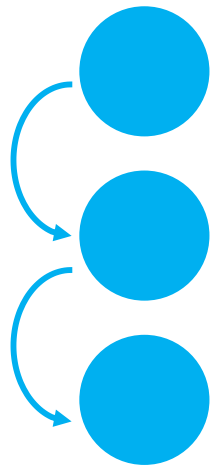
ウズウズカレツジ プログラマーコース

条件分岐～if～

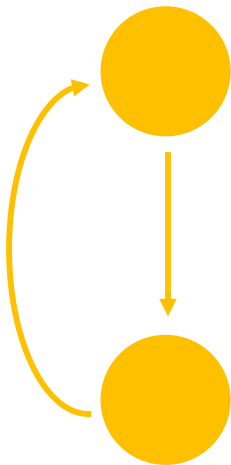
構造化プログラミング

最適に組み合わせることで
品質の高いプログラムを書くことができる

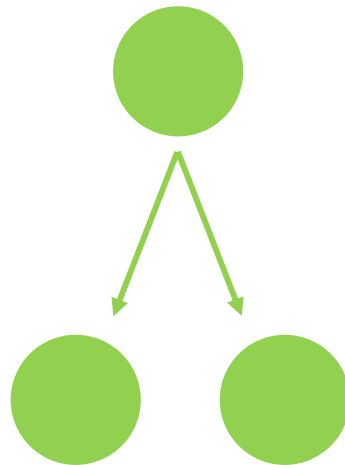
順次



繰り返し



条件分岐



《構造化プログラミング》

- これまでプログラムはソースコード上の命令文を上から順番に処理していくものだと説明してきましたが、**制御構文**というものを使うと処理の流れを自在に組むことが可能になります。
- ソースコード上の命令文を上から順に実行していく処理方法のことを「**順次**」と言います。制御構文を扱えるようになると順次以外にも「**繰り返し**」「**条件分岐**」といったプログラムの流れを作ることができるようになります。
- 「順次」「繰り返し」「条件分岐」の3つを最適に組み合わせることで誰が見ても理解のしやすい、シンプルでロジカルなアルゴリズムが書けると言われており、このような考え方でプログラムを組み立てる手法のことを**構造化プログラミング**と言います。

条件

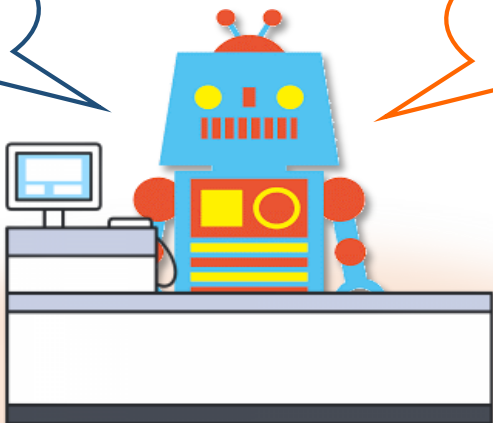
顧客の年齢が20歳未満である

false

お会計220円
になります

true

未成年の方に
お酒は売れません



《条件分岐》

□構造化プログラミングにおける「条件分岐」とは「**条件に合致する、合致しないで実行する処理を振り分ける**」ことを意味します。

「xxxの場合はaaaを、yyyの場合はbbbを実行せよ」とプログラムを組むことで、状況に応じた判断をコンピュータにさせることが可能になります。

□条件分岐の制御構文として主要なものは以下の2つです。

- if文
- switch文

▼Sample1_12_1.java (4~12行目)

```
//ifのしくみ  
int customerAge = Integer.parseInt(args[0]); //顧客の年齢  
if( customerAge < 20){  
    System.out.println("「未成年の方にお酒は売れません」");  
}
```

C:\¥Workspace>java Sample1_12_1 19
「未成年の方にお酒は売れません」

C:\¥Workspace>java Sample1_12_1 20

C:\¥Workspace>_

▼Sample1_12_2.java (4~16行目)

```
//if-elseのしくみ  
int customerAge = Integer.parseInt(args[0]); //顧客の年齢  
if( customerAge < 20){  
    System.out.println("「未成年の方にお酒は売れません」");  
}else{  
    System.out.println("「お会計220円になります」");  
}
```

C:\¥Workspace>java Sample1_12_2 19
「未成年の方にお酒は売れません」

C:\¥Workspace>java Sample1_12_2 20
「お会計220円になります」

C:\¥Workspace>

《if文 / if-else文》

□if文は条件分岐の構文の1つです。

() 内の条件がtrueであれば { } 内の処理を実行し、falseであれば何もせずに構文を抜けます。書き方は以下のとおりです。

if(条件) { 条件がtrueで実行したい処理 }


□通常のif文は条件がfalseであれば何もしますが、if-else文ではfalseだった場合の処理も定義することができます。

これは条件がtrueかfalseかの2分岐であることを意味します。書き方は以下のとおりです。

**if(条件) { 条件がtrueで実行したい処理 }
else { 条件がfalseで実行したい処理 }**

▼Sample1_12_1.java (4~12行目)

```
//if-elseのしくみ
int customerAge = Integer.parseInt(args[0]) ; //顧客の年齢
if( customerAge < 12 ){
    System.out.println("「お父さんかお母さんは？」");
}
else if( customerAge < 20 ){
    System.out.println("「未成年の方にお酒は売れません」");
}
else{
    System.out.println("「お会計220円になります」");
}
```



```
C:\¥Workspace>java Sample1_12_3 11
「お父さんかお母さんは？」
C:\¥Workspace>java Sample1_12_3 12
「未成年の方にお酒は売れません」
C:\¥Workspace>java Sample1_12_3 19
「未成年の方にお酒は売れません」
C:\¥Workspace>java Sample1_12_3 20
「お会計220円になります」
C:\¥Workspace>_
```

≪ifの連鎖 (else-if文) ≫

- if-else文では2分岐しかできませんが、**else-if文**を用いることで多分岐の制御構文を組むことが可能です。
else-ifを繋げていくことで「条件xxxがtrueならaaaする、そうでなくて条件yyyがtrueならbbbする、そうでなくて・・・」という多分岐を実現します。
3分岐の場合の書き方は以下のとおりです。

```
if( 条件① ) {
    条件①がtrueで実行したい処理
}
else if( 条件② ) {
    条件①がfalse 且つ
    条件②がtrueで実行したい処理
}
else {
    条件①も条件②もfalseで
    実行したい処理
}
```

<演習：Ex1_12_1>

コマンドライン引数から数値を1つ受け取り、これをテストの点数とします。

(1)以下のプログラムを作成してください。

- 点数が0～100以外の数字だった場合「不正な点数です！」と表示する

(2)以下の機能を(1)のプログラムに追加してください。

- 点数が0～59の数字だった場合「赤点です！」と表示する
- 点数が60～79の数字だった場合「普通です！」と表示する
- 点数が80～100の数字だった場合「優秀です！」と表示する

(3)以下の機能を(1)(2)のプログラムに追加してください。

- 点数の数字が100だった場合のみ
「満点だったので宿題免除です！！」と最後に表示する

```
C:¥Workspace>java Ex1_12_1 101  
不正な点数です！
```

```
C:¥Workspace>java Ex1_12_1 59  
赤点です！
```

```
C:¥Workspace>java Ex1_12_1 60  
普通です！
```

```
C:¥Workspace>java Ex1_12_1 80  
優秀です！
```

```
C:¥Workspace>java Ex1_12_1 100  
優秀です！  
満点だったので宿題免除です！
```

```
C:¥Workspace>_
```

▲実行例

<演習：Ex1_12_2>

```
0 1 2 3 4 5 6 7 8 9 10 11 12
1  /*-< 演習：Ex1_12_2 >-----<
2  以下は以前に講座で扱った Sample1_11_1.java と同じ内容の処理です。<
3  このプログラムは 0 や 1 などをコマンドライン引数で受け取ると無限ループに陥ってしまいます。<
4  この対策としてreceiveNumberが以下のNGパターンに該当する場合はwhile文を実行せず、代わりに<
5  「適切な値を入力してください」というメッセージを表示するプログラムに書き換えてください。<
6  <
7  [receiveNumberのNGパターン]<
8  |_ receiveNumberの値がマイナス<
9  |_ receiveNumberの値が0<
10 |_ receiveNumberの値が1<
11 <
12 -----*/<
13 class Ex1_12_1 {<
14 ^   public static void main (String[] args) {<
15 ^   ^   <
16 ^   ^   /*コマンドライン引数で受け取った数字の累乗の数のうち、100未満のもののみを表示するプログラムを作る。<
17 ^   ^   **【例】コマンドライン引数での入力：3 → 表示される数：3 , 9 , 27 , 81<
18 ^   ^   */<
19 ^   ^   <
20 ^   ^   int receiveNumber = Integer.parseInt(args[0]); //ループ毎にcalcNumberに掛ける数（コマンドライン引数で受け取った値）<
21 ^   ^   int calcNumber    = receiveNumber ;           //表示する数（初期値：コマンドライン引数で受け取った値）<
22 ^   ^   <
23 ^   ^   while( calcNumber < 100 ){<
24 ^   ^   ^   <
25 ^   ^   ^   System.out.println("calcNumber : " + calcNumber);<
26 ^   ^   ^   <
27 ^   ^   ^   calcNumber *= receiveNumber ;<
28 ^   ^   ^   <
29 ^   ^   }<
30 ^   ^   <
31 ^   }<
32 }
```