基礎プログラミングI

第2回　変数・制御構造

値と変数

値と変数

計算機は何らかのデータを受け取って、それに対して加工や計算などをした結果を吐き出す。

動作中のプログラムの中でやり取りされるデータは全て**値**として存在している。

数値と文字列

値は大きく分けて**数値**と**文字列**に分類される。

数値：加減乗除算のできる値
文字列：文字が並んだ値

数値と文字列の例

数値：5, 123, 3.5, −2

文字列：“Hello,” “123,” “太郎,” “1+1”

Rubyでは文字列を**ダブルクォート(“”)**か**シングルクォート(‘’)**で括る。

数値と文字列の例

数値の123と文字列の “123”は足し算の結果が異なる。

数値：123+123→246

文字列：“123”+“123” → “123123”

真偽値

「ある条件が成り立つか否か」を表す値のことを**真偽値**という。

「条件が成立する」→ true
「条件が成立しない」→ false

真偽値の例

- trueとなる条件式
  5 > 3
  2 < 9
- falseとなる条件式
  5 < 3
  2 > 9

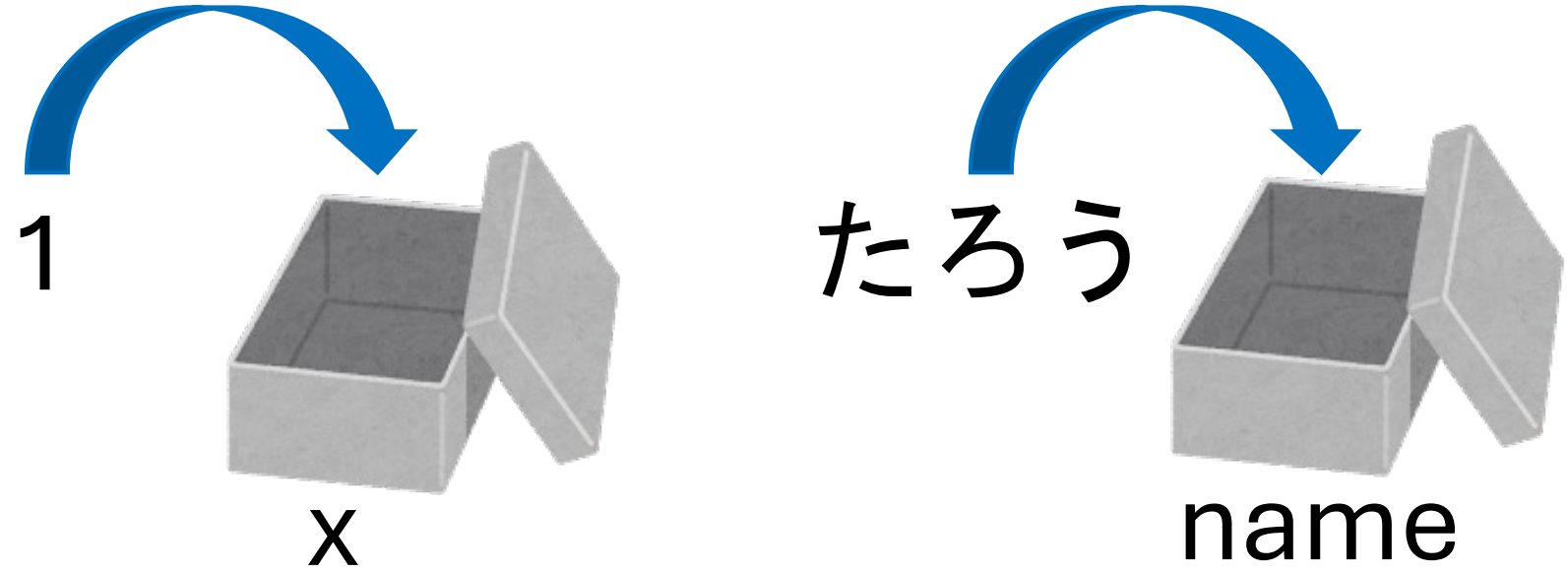
真偽値

Rubyの場合「空っぽ」を意味する値として**nil**がある。

これは偽を意味する。

変数

変数とは「値」を入れる
「箱」のようなもの。



代入

変数に値を入れること。

例：

x = 5

name = “Taro”

参照

変数に代入されている値を取り出すこと。

例：

x = 5

y = x +3

制御構造

制御構造

こんな時はどうするか？

- 計算結果がプラスならx、ゼロかマイナスならyを実行
- 同じ処理を5回繰り返す

制御構造

制御構造でプログラムの実行順序などを変えたり、一部を実行させなくしたりする。

条件判断: if文

条件によって挙動が変わるプログラムを作る場合

```
if 条件 then
    条件が成り立ったとき実行したい処理
end
```

条件判断: if文

```
if 条件1 then
    処理1
elsif 条件2
    処理2
else 条件3
    処理3
end
```

繰り返し：while文

繰り返しを行う構文

```
while 繰り返し続ける条件
    繰り返したい処理
end
```

まとめ

値：
プログラムの中でやり取りされるデータ
変数：
値を入れる箱のようなもの
制御構造：
プログラムの実行順序などを変える仕組み