

プログラミング演習I

– 第3回 条件判断、繰り返し –

1 はじめに

第3回の演習の目標は、

- 条件判断を理解する。(if文とcase文)
- 論理演算子を理解する。
- 繰り返しを理解する。(while文)

2 条件判断

教科書

p.71- 条件判断

演習 3-1

```
-----  
print("西暦を入力してください \n")  
ad = gets.chomp.to_i  
heisei = ad - 1988  
print(heisei, "\n")  
-----
```

演習 3-2

演習 3-1 のプログラムにおいて、2009 と入力すると正しい結果は得られるか、また 1980 と入力した場合は正しい結果が得られるかどうか確かめよう。

平成元年は、西暦何年か？もし、答えとして負の値を表示しないようにするためには、下記のようなプログラムを記述する必要がある。

```
-----  
「もし、入力された西暦が *** より大きい場合は、平成に変換する計算を行う。  
それ以外は、変換できませんというエラーメッセージを表示する。」  
-----
```

このようなプログラムを作成するためには、条件判断を行う必要がある。条件判断とは何か、教科書 p.71-p.72 から調べよう。

演習 3-3

条件判断と出力結果を見比べながら、比較演算子の意味を理解しよう。また、name や num をいろいろ変えて、試してみよう。

```
----- [ 表示結果をメモしておこう]  
name = "Ruby"  
print(name == "Ruby", "\n")  
print(name == "Perl", "\n")  
  
num = 100  
print(num == 99, "\n")  
print(num == 100, "\n")  
print(num < 100, "\n")  
print(num <= 100, "\n")  
print(num >= 100, "\n")  
print(num > 100, "\n")  
-----
```

演習 3-4

条件判断と出力結果を見比べながら、== と != の違いを理解しよう。また、name や num をいろいろ変えて、試してみよう。

```
----- [ 表示結果をメモしておこう]  
name = "Ruby"  
print(name == "Ruby", "\n")  
print(name == "Perl", "\n")  
print(name != "Ruby", "\n")  
print(name != "Perl", "\n")  
  
num = 100  
print(num == 100, "\n")  
print(num == 99, "\n")  
print(num != 100, "\n")  
print(num != 99, "\n")  
-----
```

演習 3-5

```

-----
name = "ベートーベン"
birth = 1770

if name == "ベートーベン" && birth == 1770
  print("Hello\n")
end
-----

```

演習 3-6

演習 3-5 のプログラムの 2 行目の birth の値を 2009 に変更し、実行してみよう。なぜ、Hello と表示されなかったのか理由を考えよう。

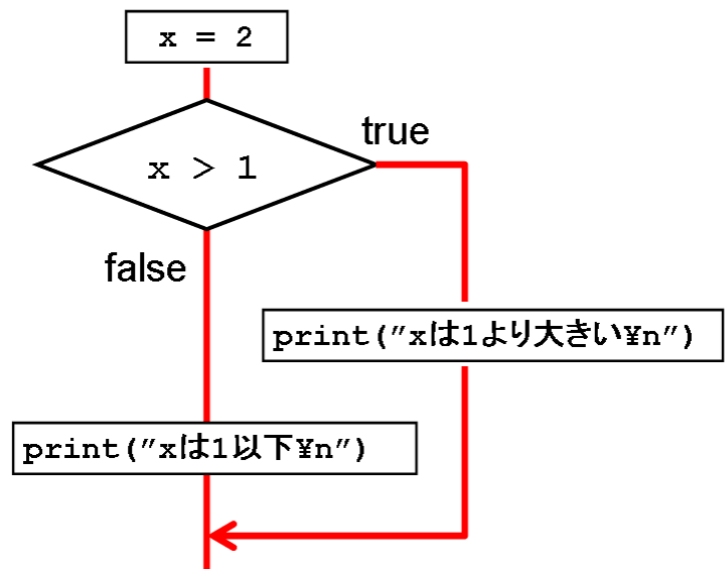
演習 3-7

条件判断の記述方法と その条件判断文が true と false の場合に何を実行するのかをプログラムと図を見ながら理解しよう。

```

-----
x = 2
if x > 1
  print("x は 1 より大きい\n")
else
  print("x は 1 以下\n")
end
-----

```



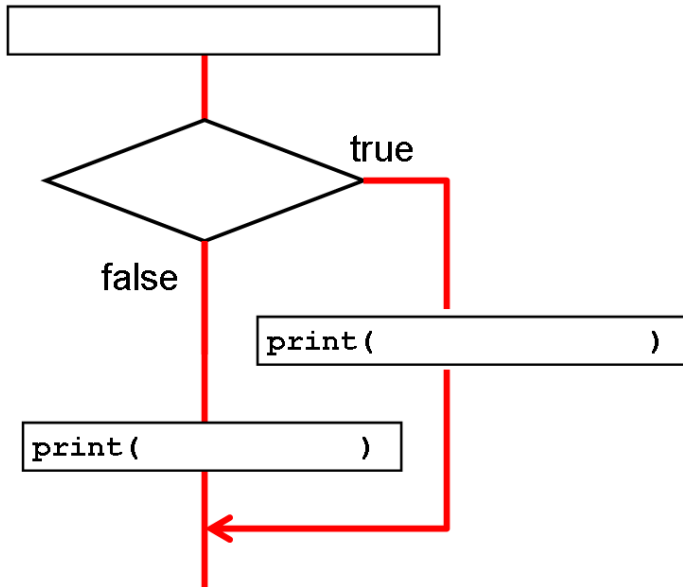
演習 3-8

演習 3-7 のプログラムの 1 行目の x の値を変更した場合、どのような結果が得られるのか考えよう。

- x = 0 とした場合は、得られる結果は？
- x = 1 とした場合は、得られる結果は？
- x = 2 とした場合は、得られる結果は？

演習 3-9

年齢を入力すると、日本の法律で飲酒可能かどうか判定するプログラムを作成しよう。図の中に条件判断文と命令文を記述した後、プログラムを作成してみよう。

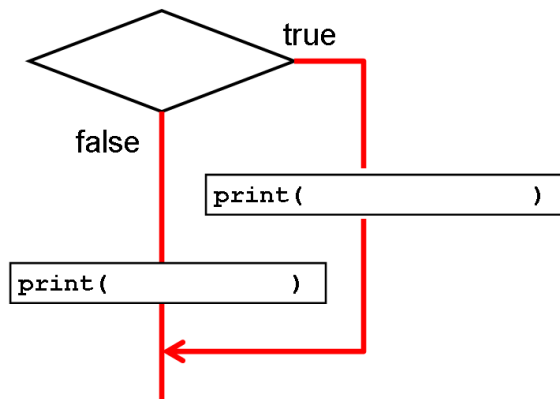


教科書
p.75- if 文

演習 3-10

文字列をいろいろ変えて、試してみよう。また図の中に条件判断文と命令文を書いてみよう。

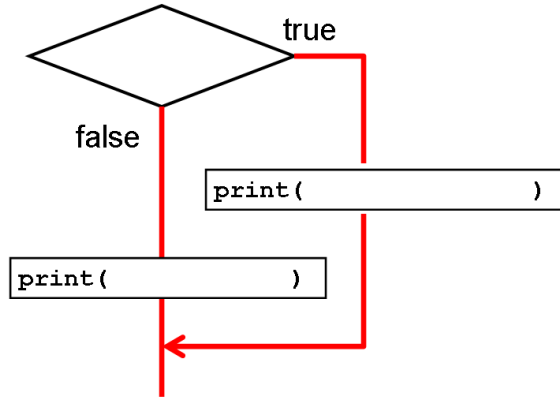
```
-----  
a = "Ruby"  
b = "Perl"  
  
if a == b  
  print("a と b は等しい\n")  
else  
  print("a と b は等しくない\n")  
end  
-----
```



演習 3-11

a と b の値をいろいろ変えて、試してみよう。また図の中に条件判断文と命令文を書いてみよう。

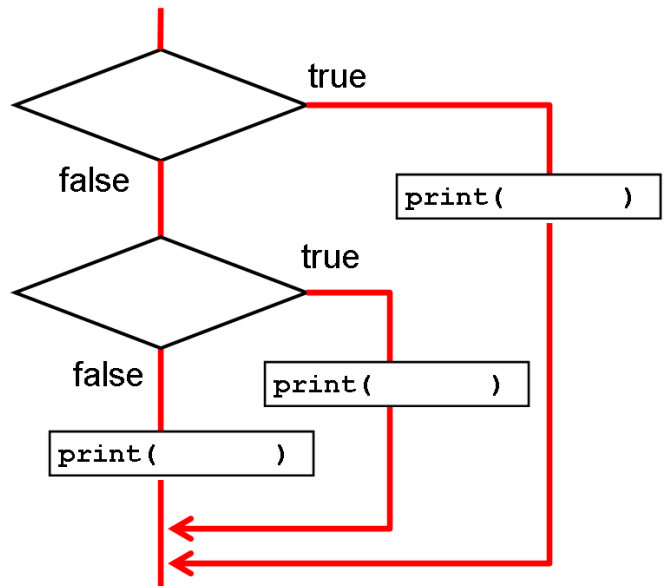
```
-----  
a = 10  
b = 20  
  
if b > a  
    print("b は a より大きい\n")  
else  
    print("b は a と等しい、または、a よ  
り小さい\n")  
end  
-----
```



演習 3-12

a と b の値をいろいろ変えて、試してみよう。また、図の中に、条件判断文と命令文を書いてみよう。

```
-----  
a = 10  
b = 10  
  
if b > a  
    print("b は a より大きい\n")  
elsif b == a  
    print("b は a と等しい\n")  
else  
    print("b は a より小さい\n")  
end  
-----
```



演習 3-13

演習 3-12 のプログラムをキーボードから 2 つの整数を入力し、その大小を判定するプログラムに書き換えてみよう。

(ヒント) a = 10 と b = 10 の部分を gets を使って書き換える。

演習 3-14

下記のプログラムを実行してみよう。また、5行目を `menu = gets.chomp` だけに変更した場合は、他にどこを変更する必要があるか考えてみよう。

```
print("1 こんにちは\n")
print("2 おはよう\n")
print("3 Hi!\n")
print("1,2,3 いずれかの整数を入力してください\n")
menu = gets.chomp.to_i

if menu == 1
  print("こんにちは\n")
elsif menu == 2
  print("おはよう\n")
elsif menu == 3
  print("Hi!\n")
else
  print("1,2,3の中から選んでください\n")
end
```

演習 3-15

下記のプログラムを実行してみよう。また、教科書 p.78 の図を参考にして、プログラムの流れ図を書いてみよう。また、5行目を `menu = gets.chomp` と変更した場合は、他にどこを変更する必要があるか考えてみよう。

```
print("1 こんにちは\n")
print("2 おはよう\n")
print("3 Hi!\n")
print("1,2,3 いずれかの整数を入力してください\n")
menu = gets.chomp.to_i

case menu
when 1
  print("こんにちは\n")
when 2
  print("おはよう\n")
when 3
  print("Hi!\n")
else
  print("1,2,3の中から選んでください\n")
end
```

演習 3-16

四則演算子 (+、-、*、/) を入力するとその演算子を用いた演算を行い、演算結果を表示するプログラムを以下につづけて作成しよう。

```
-----  
x = 10  
y = 20  
print("四則演算子を入力してください\n")  
enzan = gets.chomp  
  
case enzan  
when "+"  
  print("x + y = ", x + y, "\n")  
  
-----
```

3 繰り返し

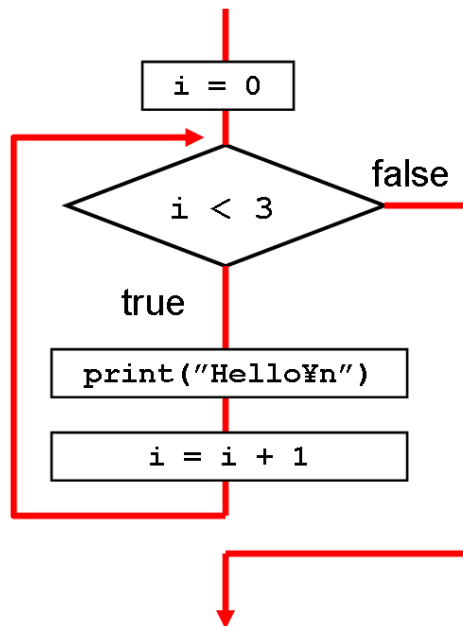
教科書

p.92- while 文

演習 3-17

Hello は何回表示されるか確かめよう。図を見ながらプログラム中の条件判断文を理解しよう。

```
-----  
i = 0  
while i < 3  
  print("Hello\n")  
  i = i + 1  
end  
-----
```



演習 3-18

i の値がどのように変化しているのか、確かめよう。

```
-----  
i = 0  
print(i, "\n")  
i = i + 1  
print(i, "\n")  
i = i + 1  
print(i, "\n")  
i = i + 1  
print(i, "\n")  
-----
```

演習 3-19

sum の値がどのように変化しているのか、確かめよう。

```
-----  
sum = 0  
print(sum, "\n")  
sum = sum + 2  
print(sum, "\n")  
sum = sum + 2  
print(sum, "\n")  
sum = sum + 2  
print(sum, "\n")  
-----
```

演習 3-20

Hello は何回表示されるか確かめよう。また、i の値がどのように変化しているのか、確かめよう。

```
-----  
i = 0  
while i < 3  
    print("i = ", i, "\n")  
    print("Hello\n")  
    i = i + 1  
    print("i = ", i, "\n")  
end  
-----
```


演習 3-21

演習 3-20 のプログラムの `while i < 3` の部分を `while i <= 3` に変更して実行してみよう。Hello が何回表示されるか確かめよう。

演習 3-22

sum の値がどのように変化しているのか確かめよう。

```
-----  
sum = 0  
while sum < 6  
  print(sum, "\n")  
  sum = sum + 2  
end  
-----
```

演習 3-23

演習 3-22 を変更し、

0 3 6 9 12 15 18 21 24 27 30

と表示するプログラムを作成しよう。

演習 3-24

break の使い方を確認しよう。i と j の値は、0 からいくつまで表示されるか確認しよう。また、`j == 3` という条件判断文をいろいろ変えて試してみよう。

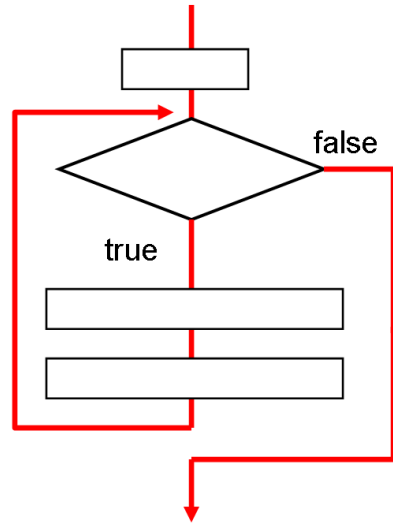
```
-----  
i = 0  
while i < 10  
  print("i = ", i, "\n")  
  i = i + 1  
end
```

```
j = 0  
while j < 10  
  print("j = ", j, "\n")  
  if j == 3  
    break  
  end  
  j = j + 1  
end  
-----
```

演習 3-25

下記のプログラムを実行しよう。また、プログラムの流れ図を完成させよう。

```
-----  
a = ["日", "月", "火", "水", "木", "金", "土"]  
i = 0  
while i < 3  
  print(a[i], "\n")  
  i = i + 1  
end  
-----
```



演習 3-26

下記のプログラムを実行しよう。(配列week_e に文字列を格納する流れを確認しておこう。)

```
-----  
week_j = ["日", "月", "火", "水", "木", "金", "土"]  
week_e = Array.new(7, "")  
i = 0  
while i < 7  
  print(week_j[i], "曜日を英語で入力してください (例: 日なら Sunday) :")  
  week_e[i] = gets.chomp  
  i = i + 1  
end  
  
print("-----\n")  
i = 0  
while i < 7  
  print(week_j[i], " : ", week_e[i], "\n")  
  i = i + 1  
end  
print("-----\n")  
-----
```

演習 3-27

5つの整数を入力すると、最後に総和を表示するプログラムを完成させよ。

```
-----  
# 要素数 5 の配列 kazu を準備する。初期値は 0  
kazu = Array.new( ????, ????)  
i = 0  
sum = ??? # 加算した結果を保存する変数 sum の初期化  
while i < ???  
  print("整数を入力してください：")  
  kazu[i] = ???  
  sum = ???  
  i = ???  
end  
print("総和:", ???, "\n")  
-----
```

演習 3-28

整数を入力すると、最後に総和を表示するプログラムを完成させよ。 . が入力されるまで、整数を入力できるようにするにはどうする？

考えてみよう-1

プログラム A とプログラム B で表示結果が異なるのはなぜ？

-----プログラム A -----

```
i = 0  
j = 0  
while i < 3  
  while j < 3  
    print("i = ", i, " j = ", j, "\n")  
    j = j + 1  
  end  
  i = i + 1  
end
```

-----プログラム B -----

```
i = 0  
while i < 3  
  j = 0  
  while j < 3  
    print("i = ", i, " j = ", j, "\n")  
    j = j + 1  
  end  
  i = i + 1  
end  
-----
```

(応用) 演習 3-29

3つの果物の名前と価格を入力すると、ハッシュfruitに保存し、最後に価格の合計を表示するプログラムを完成させよ。

```
-----  
# 果物の名前と価格を保存するハッシュfruitの定義  
fruit = ???  
  
i = 0  
  
# 価格の合計を保存する変数の初期化  
sum = ???  
  
# 果物の名前と価格をキーボードから入力(3個入力)  
# 果物の名前をハッシュfruitのキーに保存し、果物の価格を値として保存する。  
while i < 3  
  print("果物の名前を入力してください:")  
  # 果物の名前を変数keyに保存  
  key = ???  
  print("果物の価格を入力してください:")  
  # 果物の価格を変数valueに保存(整数として保存)  
  value = ???  
  
  # ハッシュfruitに保存(変数keyの値をキーに、変数valueの値を値として保存)  
  ???  
  
  # 繰り返し回数をカウントする  
  i =  
end  
  
# ハッシュfruitのキーと値を表示し、合計金額を計算  
fruit.each{|key, value|  
  print(key, ":", value, "円\n")  
  sum = ???  
}  
  
# 合計を表示する  
print("合計:", ???, "円\n")  
-----
```

(応用) 演習 3-30

下記の辞書順で先頭となる単語を表示するプログラムを変更し、
キーボードから 4 個の単語を入力し、辞書順で先頭となる単語を表示するプログラムに変更してみよう。
さらに、while 文を使って書き換えてみよう。

```
-----  
tango = ["cat", "cook", "car"]  
  
min = tango[0]  
if min > tango[1]  
    min = tango[1]  
end  
if min > tango[2]  
    min = tango[2]  
end  
print(min, "\n")  
-----
```

(応用) 演習 3-31

キーボードから 5 つの整数を入力し、一番小さい値を表示するプログラムを作成しよう。

```
-----  
# 整数型の値を保存する配列を準備する。  
  
# 5 回入力できるようにする  
  
# 整数を入力してくださいと表示する  
  
# 入力した値を配列に保存する  
  
# 5 個の整数値を入力後、一番小さい値を判定する。  
  
# 一番小さい値を表示する  
-----
```

(応用) 演習 3-32

キーボードから 5 つの整数を入力し、一番大きい値を数字を表示するプログラムを作成しよう。