プログラミング演習I
– 第3回　変数、配列、ハッシュ –

1　はじめに

第 3 回の演習の目標は、

- 変数、定数、予約語、オブジェクトを理解する。
- 四則演算と Math モジュールを利用した計算を理解する。
- 変数の型と型変換を理解する。
- 配列とハッシュの基本的な使い方を理解する。

ことである。

2　オブジェクト、変数

教科書

p.57(p.59)- オブジェクト

演習 3-1　数値オブジェクトとは何か、また、文字列オブジェクトとは何か調べよう。

演習 3-2　変数とは何か、また、変数の種類を調べよう。

演習 3-3　定数とは何か調べよう。

演習 3-4　予約語とは何か調べよう。

演習 3-5　(ex3-5.rb) なぜ実行できないのか考えよう。また、間違っている箇所を修正しよう。

```
-----------------------------------------------
case = "Ruby"
if case == "Ruby"
  print(case, "\n")
end
-----------------------------------------------
```

演習 3-6　(ex3-6.rb) なぜ実行できないのか考えよう。また、間違っている箇所を修正しよう。

```
-----------------------------------------------
class = "Ruby"
if class == "Ruby"
  print(class, "\n")
end
-----------------------------------------------
```

教科書

p.220(p.221)- 数値クラス

演習 3-7　数値クラスの関係図を参照し、数値クラスの種類とそれらの関係について調べよう。

演習 3-8　整数と浮動小数点数では計算結果がどのように異なるのか考えよう。

```
-----------------------------------------------
print(3 + 2, "\n")
print(3 + 2.0, "\n")

print(3 - 2, "\n")
print(3 - 2.0, "\n")

print(3 * 2, "\n")
print(3 * 2.0, "\n")

print(3 / 2, "\n")
print(3 / 2.0, "\n")

print(3 ** 2, "\n")
print(3 ** 0.5, "\n")
-----------------------------------------------
```

演習 3-9　四則演算以外の演算子の種類を調べてみよう。また、次の式の計算結果を表示するプログラムを作成しよう。
(ヒント（実行結果） 2 番目の式の実行結果は、523.333333333333　）

$$2^8$$

$$4 \div 3 \times 3.14 \times 5^3$$

$$(12 + 20) \times 4 \div 3$$

教科書

p.226(p.228)- Math モジュール

演習 3-10　Math モジュールとは何か、また、どのようなメソッドがあるのか調べよう。

演習 3-11

```
------------------------------------------------
a = 2
print(Math.sqrt(a), "\n")
------------------------------------------------
```

演習 3-12　演習 3-10 で調べた、メソッドを使用して算術演算を行うプログラムを作成しよう。

教科書

p.228(p.229)- 数値型の変換

演習 3-13

```
------------------------------------------------
print("数を入力してください\n")
x = gets.chomp

print("数を入力してください\n")
y = gets.chomp

z = x + y
print(x, "+", y, "=", z, "\n")
------------------------------------------------
```

演習 3-14　入力した 2 つの整数に対する演算結果が異なる理由を考えよう。

```
-----------------------------------------------
print("整数を入力してください\n")
x = gets.chomp
print("整数を入力してください\n")
y = gets.chomp

z = x + y
print(x, "+", y, "=", z, "\n")
z = x.to_i + y.to_i
print(x, "+", y, "=", z, "\n")
-----------------------------------------------
```

演習 3-15　入力した 2 つの整数に対する演算結果が異なる理由を考えよう。

```
-----------------------------------------------
print("整数を入力してください\n")
x = gets.chomp.to_i
print("整数を入力してください\n")
y = gets.chomp.to_i

z = x / y
print(x, "/", y, "=", z, "\n")

z = x.to_f / y.to_f
print(x, "/", y, "=", z, "\n")
-----------------------------------------------
```

演習 3-16　`to_f` メソッドとは何か、また　`to_i` メソッドとは何か調べよう。この 2 つ以外に、型変換を行うためのメソッドがあるか調べてみよう。

演習 3-17　3 つの数を入力すると、その 3 つの数の和と積を計算し、表示するプログラムを作成しよう。

演習 3-18　3 つの数を入力すると、その 3 つの数の平均を計算し、表示するプログラムを作成しよう。

3　配列

教科書

p.241(p.241)- 配列

演習 3-19

```
------------------------------------------------
num = [1, 2, 3]
strs = ["a", "b", "c"]

print(num[0], "\n")
print(num[1], "\n")
print(num[2], "\n")

print(strs[0], "\n")
print(strs[1], "\n")
print(strs[2], "\n")
------------------------------------------------
```

演習 3-20

```
------------------------------------------------
num = [1, 2, 3]
strs = ["a", "b", "c"]

print(num, "\n")
print(strs, "\n")
------------------------------------------------
```

演習 3-21

```
------------------------------------------------
name= ["ベートーベン", "シューベルト"]
birth = [1770, 1797]

print(name[0], "は", birth[0], "年に生まれた。\n")
------------------------------------------------
```

演習 3-22　演習 3-21 を「1797 年に生まれたのは、シューベルトだ。」と表示できるように変更してみよう。さらに数名の有名人の名前と生まれた年を配列に追加し、表示できるように変更してみよう。

演習 3-23

```
-----------------------------------------------
a = [1, 2, 3]
b = [10, 20, 30]
c = [100, 200, 300]

print(a[0] + b[0] + c[0], "\n")
print(a[1] + b[1] + c[1], "\n")
-----------------------------------------------
```

演習 3-24　演習 3-23 のプログラムに追加し、a[2] と b[2] と c[2] の値を足した結果も表示できるようにしよう。

演習 3-25　配列を作成しよう。(配列の大きさの設定と初期化について調べること)

```
-----------------------------------------------
a = Array.new(3, 0)
print(a[0], "\n")
a[0] = 2
a[1] = 4
a[2] = 6
print(a[0], "\n")
-----------------------------------------------
```

演習 3-26　配列を作成しよう。(配列の大きさは？初期値は？)

```
-----------------------------------------------
a = Array.new(3, "")
print(a[0], "\n")
a[0] = "a"
a[1] = "b"
a[2] = "c"
print(a[0], "\n")
-----------------------------------------------
```

演習 3-27

```
-----------------------------------------------
name = Array.new(3, "")
print("名前を入力してください\n")
name[0] = gets.chomp

print("名前を入力してください\n")
name[1] = gets.chomp
```

```
print("名前を入力してください\n")
name[2] = gets.chomp

print(name[0], "さん、おはようございます！\n")
print(name[1], "さん、こんにちは！\n")
print(name[2], "さん、こんばんは！\n")
-----------------------------------------------
```

演習 3-28　演習 3-27 のプログラムを下記のように変更しよう。

4 名の名前を保存するための配列を用意し、値の初期化を行う。
4 名の名前をキーボードから入力する。
4 名の名前を表示する

演習 3-29　3 つの整数を入力し、その和を計算し表示するプログラムを作成しよう。プログラムの構成は以下のとおりである。

1. 3 つの整数を保存するための配列を用意し、値の初期化を行う。（初期値は 0)
2. 3 つの整数をキーボードから入力する。
3. 3 つの整数の和を計算し、表示する。

4　ハッシュ

教科書

p.303(p.299)- ハッシュ

演習 3-30

```
-----------------------------------------------
color = {"blue" => "青", "red" => "赤", "green" => "緑"}

print("色を入力してください (blue, red, green) \n")
str = gets.chomp
print(color[str], "\n")
-----------------------------------------------
```

演習 3-31

```
-----------------------------------------------
jpn = {"frog" => "かえる", "bee" => "はち", "duck" => "かも", "cicada" => "せみ"}

print("選んで入力してください（frog, bee, duck, cicada）\n")
str = gets.chomp
print(jpn[str], "\n")
-----------------------------------------------
```

演習 3-32　キー、値　はそれぞれ何を意味するのか調べよう。また、Hash.new はどんな処理を行うメソッドか調べよう。

演習 3-33　（ハッシュの初期化について調べること。）

```
------------------------------------------------
h1 = Hash.new
h2 = Hash.new("")
print(h1["not_key"], "\n")
print(h2["not_key"], "\n")

h2["R"] = "Ruby"
print(h2["R"], "\n")
------------------------------------------------
```

演習 3-34

```
------------------------------------------------
eng = {"かえる" => "frog", "はち" => "bee", "かも" => "duck", "せみ" => "cicada"}
print(eng["かえる"], "\n")
print(eng["はち"], "\n")
print(eng["かも"], "\n")
print(eng["せみ"], "\n")
------------------------------------------------
```

演習 3-35　演習 3-34 のプログラムに、キーと値の一覧を表示する下記のプログラムを追加してみよう。(each メソッドを使う)

```
------------------------------------------------
eng.each{|key, value|
  print(key, ":", value, "\n")
}
------------------------------------------------
```

演習 3-36　ハッシュに、キーが同じものを入力した場合、どのような結果となるか、word の中身を増やしたり、減らしたりして確かめよう。

```
------------------------------------------------
word = {"cat" => "猫", "cat" => "chat", "cat" => "katze",
"cat" => "gatto", "cat" => "gato"}

print(word["cat"], "\n")
------------------------------------------------
```

演習 3-37　ハッシュで表現できるものを考えて定義し、キーを指定すると値を表示するプログラムを作成しよう。

演習 3-38　ハッシュに、値を設定する。

```
-----------------------------------------------
word = Hash.new
word["cat"] = "猫"
word["dog"] = "犬"

word.each{|key, value|
  print(key, ":", value, "\n")
}
-----------------------------------------------
```

演習 3-39　ハッシュの大きさを調べる。(重要： store について調べておこう)

```
-----------------------------------------------
word = Hash.new

print("キーを入力してください:")
key = gets.chomp
print("値を入力してください:")
value = gets.chomp
word.store(key,value)
print("ハッシュのサイズ = ", word.size, "\n")

print("キーを入力してください:")
key = gets.chomp
print("値を入力してください:")
value = gets.chomp
word.store(key,value)
print("ハッシュのサイズ = ", word.size, "\n")

word.each{|key, value|
  print(key, ":", value, "\n")
}
-----------------------------------------------
```

演習 3-40　ハッシュのキーと値を削除する

```
-----------------------------------------------
color = {"blue"=> "青", "red" => "赤", "green" => "緑"}

print("削除前\n")
color.each{|key, value|
  print(key, ":", value, "\n")
}

color.delete("red")
print("削除後\n")
color.each{|key, value|
  print(key, ":", value, "\n")
}
-----------------------------------------------
```

演習 3-41　ハッシュの値として「文字列」を保存した場合と「整数値」を保存した場合を比較しておこう。

プログラムＡ（果物の値段を整数値として保存：期待した結果が得られる例)

```
-----------------------------------------------
fruit = {"りんご" => 150, "みかん" => 100, "バナナ" => 70}
sum = 0

fruit.each{|key, value|
  print(key, ":", value, "円\n")
  sum = sum + value
}

print("合計:", sum, "円\n")
-----------------------------------------------
```

プログラムＢ（果物の値段を文字列として保存：期待した結果が得られない例）

```
-----------------------------------------------
fruit = {"りんご" => "150", "みかん" => "100", "バナナ" => "70"}
sum = ""

fruit.each{|key, value|
  print(key, ":", value, "円\n")
  sum = sum + value
}

print("合計:", sum, "円\n")
-----------------------------------------------
```

プログラムＣ（果物の値段を文字列として保存：期待した結果は得られるが、良くない例）

```
-----------------------------------------------
fruit = {"りんご" => "150", "みかん" => "100", "バナナ" => "70"}
sum = 0

fruit.each{|key, value|
  print(key, ":", value, "円\n")
  sum = sum + value.to_i        # 価格の合計を計算するために文字列を整数に変換
}

print("合計:", sum, "円\n")
-----------------------------------------------
```

演習 3-42　ハッシュのキーと値をキーボードから入力してみよう。
　色を英語で入力してください：と表示されたら、blue と入力し、色を日本語で入力してください：と表示されたら、青と入力する。他の色も入力しよう。

```
-----------------------------------------------
color = Hash.new

i = 0
while i < 3
  print("色を英語で入力してください:")
  key = gets.chomp
  print("色を日本語で入力してください:")
  value = gets.chomp
  color.store(key, value)
  i = i + 1
end

color.each{|key, value|
  print(key, ":", value, "\n")
}
-----------------------------------------------
```

**** プログラム中の下記の部分は繰り返しを行うために必要な記述である。次回のテキストで学ぶ。

```
i = 0
while i < 3
 ........
  i = i + 1
end
*******************************
```

演習 3-43　演習 3-42 を下記のように変更してみよう。

　* 削除するキーを入力してください　と表示する。
　* 入力されたキーと値のペアを取り除く（削除する）。
　* 登録されている値の一覧を表示し、指定したキーと値のペアが削除されていることを確認できるようにする。

5　条件判断

教科書

p.67(p.71)- 条件判断

演習 3-44　比較演算子の種類を調べよう。true と false を意味を英語辞書で調べよう。また、比較演算の結果として得られる true と false は何を意味するのか調べよう。

教科書

p.70(p.74)- 論理演算子

演習 3-45　論理演算子にはどのようなものがあるか調べよう。