プログラミング演習I
– 第2回　プログラムに慣れる その2–

1　Rubyをとにかく使おう その2

第1回の演習で、簡単なプログラムの入力と実行を繰り返しを行ったので、だいぶプログラミングに慣れてきたと思う。そこで、今回の演習からは、プログラムを単に実行するだけではなく、プログラム1行ごとの意味を理解することを目標とする。

どのようなプログラムを記述すると、下記に示したことを行うことが出来るのだろうか?
まずは、演習 2-1 から点線内に示されているプログラムを実行し、また教科書を使って調べながらすすめていこう。

プログラミング演習I 前半で行う内容の一部

■　単語のつづりをチェックするには？
　□　正しい場合には OK と表示し、間違っている場合は NG と表示するには？

■　同じ処理を何度も繰り返すには？
　□　こんにちはＲｕｂｙさん！　を　画面上に 100 行表示するには？
　□　0 から 100 までの数を画面上に表示し、最後に総和を表示するには？

■　変数って何？
　□　文字列や数値を記憶させておきたい場合はどうする？

■　5 人分の名前を記憶させておくには？
　□　その 5 人の名前を表示するには？

■　果物の種類と価格を記憶させておくには？
　□　りんご 130 円、オレンジ 80 円、バナナ 30 円　というデータを記憶させておくには？
　□　記憶させておいた果物の種類と価格を表示するには？

教科書

p.25- 条件判断 if

演習 2-1

```
-----------------------------------------------
print("本を英語で書くと？入力してください\n")
name = gets.chomp
if name == "book"
  print("正解！\n")
else
  print("不正解! 正解は、 book だよ。\n")
end
-----------------------------------------------
```

演習 2-2

```
-----------------------------------------------
print("本を英語で書くと？入力してください\n")
name = gets.chomp
if name != "book"
  print("不正解! 正解は、 book だよ。\n")
else
  print("正解！\n")
end
-----------------------------------------------
```

教科書

p.27 繰り返し while

演習 2-3　You　> と表示されたら文章を入力し、Enter キーを押す。(bye と入力するまで入力と出力を繰り返すプログラム)

```
-----------------------------------------------
command = ""
print("Ruby > ", "こんにちは Ruby です", "\n")
while command != "bye"
  print("You  > ")
  command = gets.chomp
  print("Ruby > ", command, "\n")
end
-----------------------------------------------
```

教科書

p.21- 変数

演習 2-4

```
-----------------------------------------------
name = "ベートーベン"
birth = 1770
print(name, "は、", birth, "年に生まれた\n")
-----------------------------------------------
```

演習 2-5

```
-----------------------------------------------
name = "シューベルト"
birth = 1797
print(name, "は、", birth, "年に生まれた\n")
-----------------------------------------------
```

演習 2-6

```
-----------------------------------------------
name_a = "ベートーベン"
birth_a = 1770
print(name_a, "は、", birth_a, "年に生まれた\n")
name_b = "シューベルト"
birth_b = 1797
print(name_b, "は、", birth_b, "年に生まれた\n")
print(name_a, "は、", name_b, "の", birth_b - birth_a, "年前に生まれた\n")
-----------------------------------------------
```

教科書

p.34- 配列

演習 2-7　配列とは何か、また要素、インデックスとは何か調べよう。

演習 2-8

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

print(name[0], "\n")
print(name[1], "\n")

print(name[0], "は、", birth[0], "年に生まれた\n")
print(name[1], "は、", birth[1], "年に生まれた\n")
print(name[0], "は、", name[1], "の", birth[1] - birth[0], "年前に生まれた\n")
-----------------------------------------------
```

演習 2-9

```
-----------------------------------------------
a = [1, 3, 5, 7]
total = a[0] + a[1] + a[2] + a[3]
print(total, "\n")
-----------------------------------------------
```

演習 2-10

```
-----------------------------------------------
a = [1, 3, 5, 7, 9, 11]
total = a[0] + a[1] + a[2] + a[3] + a[4] + a[5]
print(total, "\n")
-----------------------------------------------
```

教科書

p.38- 配列の大きさ

演習 2-11　配列の大きさを表示しよう。

```
-----------------------------------------------
name = ["ベートーベン", "シューベルト"]
print(name.size, "\n")
-----------------------------------------------
```

演習 2-12　配列の大きさを表示しよう。

```
-----------------------------------------------
name = ["ベートーベン", "シューベルト", "ショパン"]
print(name.size, "\n")
-----------------------------------------------
```

演習 2-13　配列の要素を表示してみよう。

```
-----------------------------------------------
name = ["ベートーベン", "シューベルト", "ショパン"]
print(name[0], "\n")
print(name[1], "\n")
print(name[2], "\n")
-----------------------------------------------
```

演習 2-14

```
-----------------------------------------------
name = ["ベートーベン", "シューベルト", "ショパン"]
i = 0
while i < 3
  print(i, ":", name[i], "\n")
  i = i + 1
end
-----------------------------------------------
```

演習 2-15

```
-----------------------------------------------
name = ["ベートーベン", "シューベルト", "ショパン"]
i = 0
while i < name.size
  print(i, ":", name[i], "\n")
  i = i + 1
end
-----------------------------------------------
```

Shoes を使って、絵を描いてみよう

Shoes の使い方
* Ruby プログラムを作成する
* Windows のスタートメニュー [すべてのプログラム]-[Shoes]-[Shoes] を選択する
* "Welcome to SHOES" というメッセージのウィンドウが表示される
* [Open an App.] をクリックし、作成したプログラムを選択する。

```
---- プログラムリスト sample.rb ------------------------------------------
Shoes.app(:width => 700, :height => 700, :resizable => false) {

  # 配列の宣言 R(赤) r_color, G(緑) g_color, B(青) b_color
  r_color = [1.0, 0.0, 0.0]
  g_color = [0.0, 1.0, 0.0]
  b_color = [0.0, 0.0, 1.0]

  button("push", width => 200, :height => 50){
    stroke "white"
    i = 0
    x = 0
    y = 0
    while i < 3
      x = x + 50 * i
      y = y + 100 * i
      # x 座標， y 座標, 丸の横の長さ，丸の縦の大きさ，:fill => rgb(R の値，G の値，B の値)
      oval x, y, 80, 30,:fill => rgb(r_color[i], g_color[i], b_color[i])
      i = i + 1
    end
  }
}
-------------------------------------------------------------------------
```

* 上のプログラムでは 3 個の丸が表示される。さらに丸を 1 個加えてみよう。
  r_olor, g_color b_color に値を追加
  他にも変更する箇所あり！さて、どこ？

* 表示位置を変えてみよう
* 円にしてみよう

教科書

p.40- ハッシュ

演習 2-16　ハッシュとは何か調べよう。

演習 2-17

```
-----------------------------------------------
birth_table ={"ベートーベン" => 1770, "シューベルト" => 1797}

print("ベートーベンの誕生年：", birth_table["ベートーベン"], "\n")
print("シューベルトの誕生年：", birth_table["シューベルト"], "\n")
-----------------------------------------------
```

演習 2-18

```
-----------------------------------------------
font_table = {"normal" => "+0", "small" => "-1", "big" => "+1"}
print(font_table["small"], "\n")
print(font_table["normal"], "\n")
print(font_table["big"], "\n")
-----------------------------------------------
```

演習 2-19

```
-----------------------------------------------
book_table = {"たのしい Ruby" => 2730, "プログラミング Ruby" => 3990}
print(book_table["たのしい Ruby"], "円\n")
print(book_table["プログラミング Ruby"], "円\n")
-----------------------------------------------
```

演習 2-20

```
-----------------------------------------------
font_table = {"normal"=> "+0", "small" => "-1", "big" => "+1"}
font_table.each { |key, value|
   print(key, value, "\n")
}
-----------------------------------------------
```

演習 2-21

```
-----------------------------------------------
book_table = {"たのしい Ruby" => 2730, "プログラミング Ruby" => 3990}
book_table.each { |key, value|
   print(key, value, "\n")
}
-----------------------------------------------
```

演習 2-22　教科書 p.43 List2.1 のプログラムを実行しよう。

演習 2-23　教科書 p.43 List2.1 のプログラムのハッシュに格納する部分を下記のように変更し、実行しよう。

```
-----------------------------------------------
font_table = {"normal" => "+0", "small" => "-1", "big" => "+1", "verybig" => "+2"}
-----------------------------------------------
```

演習 2-24　プログラムを実行しよう。(実行方法は、演習 2-22 と同じ 。 `ruby プログラム名 > color1.html`)

```
-----------------------------------------------
print("<html>\n<title>color list</title>\n")
print("<body>\n<p>\n")
color_table ={"赤" => "red", "青" => "blue", "黄" => "yellow"}
color_table.each{ |key, value|
  print("<font color=\"", value, "\">", key, "</font><br>\n")
}
print("</p>\n</body>\n</html>\n")
-----------------------------------------------
```

演習 2-25　プログラムを実行しよう。(実行方法は、演習 2-22 と同じ。 `ruby ファイル名 > color2.html`)

```
-----------------------------------------------
print("<html>\n<title>color</title>\n")
print("<body bgcolor = \"blue\">\n")
print("<p>\n")
print("背景の色は、何色かな？\n")
print("</p>\n</body>\n</html>\n")
-----------------------------------------------
```

演習 2-26　背景色（赤色、黄色、青色）をコンピュータが決定するプログラムを実行しよう。(実行方法は、演習 2-22 と同じ 。 `ruby ファイル名 > color3.html`) 演習 2-25 のプログラムと、どの部分が異なるのか確認しよう。また、プログラム中の rand について調べよう。

```
-----------------------------------------------
color = ["red", "yellow", "blue"]
no = rand(3)

print("<html>\n<title>color</title>\n")
print("<body bgcolor = \"", color[no], "\">\n")
print("<p>\n")
print("背景の色は、何色かな?\n")
print("</p>\n</body>\n</html>\n")
-----------------------------------------------
```