プログラミング演習I 夏休みの自由課題

概要

- 夏休みの自由課題は【堅実コース】と【冒険コース】があります。どちらを選んでも構いませんし、やらなくても構いません。ただし、両方は提出できません。
- 【堅実コース】は中テストに加点します。最大 20 点です。ただし、100 点（満点）以上には加点しません。
- 【冒険コース】は中テストが 85 点以上だった場合に最大 40 点を加点しますが、85 点未満だった場合は無効になります。
- 締切：中テスト終了後（木曜クラス：10/6，金曜クラス：10/7）

課題【堅実コース】

- この PDF ファイルを表紙も含めて【両面】印刷して、解答を書き込んだものを提出してください。
- コンピュータで実行する前に考えて書いてみましょう。
- ☆☆の問題は、少し難しいのでがんばって下さい。

課題【冒険コース】

- 「おおっ！」と唸らせるプログラムを作成してください。
- 全学計算機システム（Windows）で教員と TA が実行可能な ruby プログラムであればなんでも O K です。
- 審査員は、TA6 名と教員 2 名の 8 名です。
- 提出物は、以下の電子的なファイルを電子メールで提出してください。宛先は授業中に指示します。
  1. プログラムファイル（そのまま実行できるもの。行番号がついていないもの）
  2. プログラムの説明ファイル（PDF ファイル）
  3. 実行結果（PDF ファイル）

曜日クラス　学籍番号　　氏名

1　変数

Ex. 1
キーボードから 2 つの整数を入力すると掛け算と割り算を行い、その結果を表示するプログラムを作成しよう。下の説明にしたがって、プログラムを書いてみよう。

プログラム	説明
	キーボードから入力された 2 つの整数を変数に代入する 掛け算の結果を変数に代入する 割り算の結果を変数に代入する 掛け算の結果を表示する 割り算の結果を表示する

Ex. 2
以下の指示にしたがって用語の説明をまとめよう。

- 変数とは何か。

  説明：

- 予約語とは何か。予約語リストを調べよう。

  説明：

  予約語リスト：

2　配列とハッシュ

Ex. 3

以下のプログラムは配列を使った計算プログラムである。プログラムの説明を書いてみよう。

プログラム	説明

`seisu = Array.new(3, 0)`
 	
`seisu[0] = gets.chomp.to_i`
`seisu[1] = gets.chomp.to_i`
`seisu[2] = gets.chomp.to_i`
 	
`wa = seisu[0] + seisu[1] + seisu[2]`
`print("3つの整数の和は", wa, "です。\n")`
 	
`heikin = wa / 3`
`print("3つの整数の平均は", heikin, "です。\n")`	

キーボードから 3，5，8 と入力したときの出力を書いてみよう。

Ex. 4

以下のプログラムはハッシュを使った英語の辞書である。プログラムの説明を書いてみよう。

プログラム	説明

`ja = {"cat" => "猫", "book" => "本"}`
 	
`ja.each {|key, value|`
`  print(key, " ==> ", value, "\n")`
`}`	

Ex. 5

以下の指示にしたがって用語の説明をまとめよう。

- 配列とは何か。「要素」、「インデックス」という用語を使って説明しよう。

  説明：

- ハッシュとは何か。「キー」、「値」という用語を使って説明しよう。

  説明：

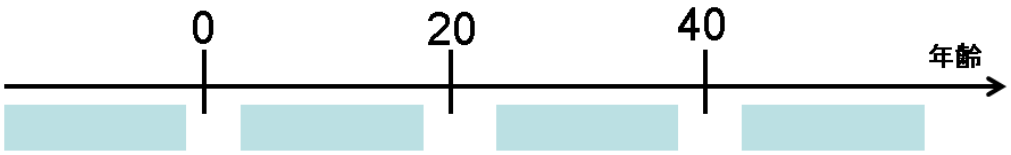
3　条件判断

Ex. 6

年齢を入力すると、日本の法律で飲酒可能かどうか判定し、下記に示すようなメッセージを表示するプログラムを作成しよう。

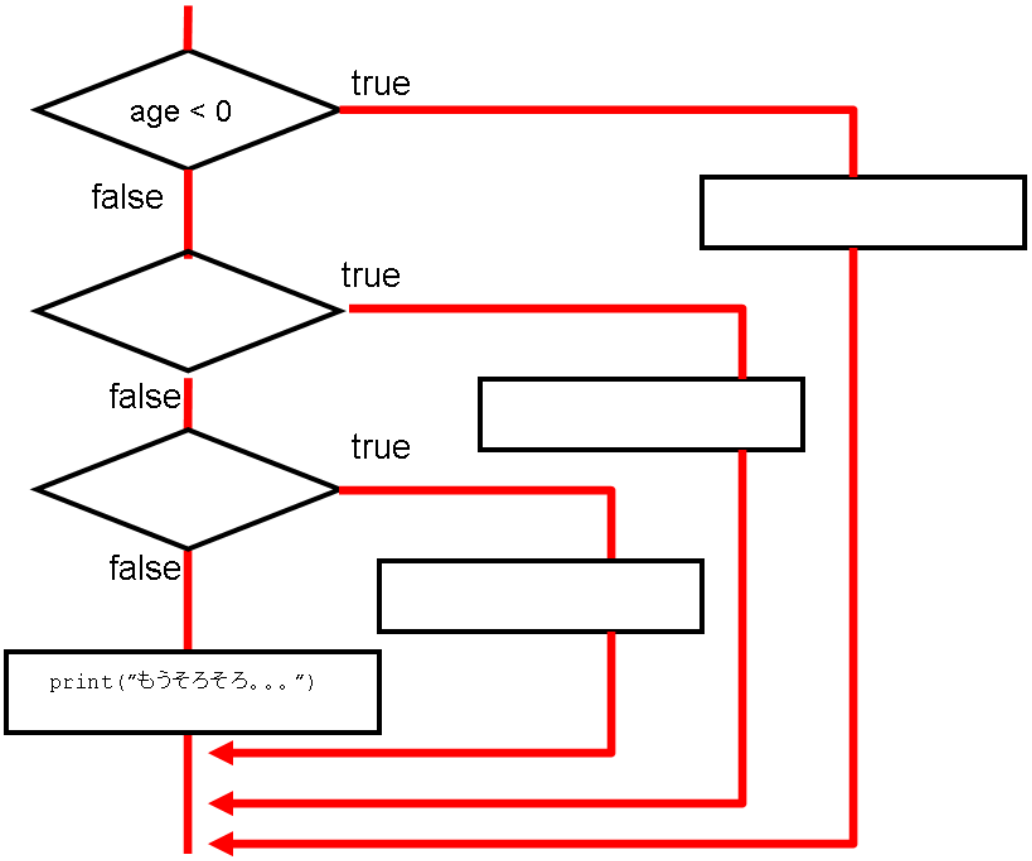
条件	メッセージ
年齢が 0 歳未満	0 以上の値を入力してください。
年齢が 0 歳以上 20 歳未満	未成年です。まだ飲酒できません。
年齢が 20 歳以上 40 歳未満	飲酒できますが、飲みすぎに注意しましょう。
それ以外	もうそろそろ飲酒をひかえたほうがいいですよ。

下図に年齢とメッセージ（省略形でよい）の関係を書いてみよう。



表示するメッセージは？

下図の流れ図の空欄を埋めてみよう。



実際のプログラムを書いてみよう。

```
print("年齢を入力してください\n")
age =

if
  print("                                                    \n")
elsif
  print("                                                    \n")
elsif
  print("                                                    \n")
else
  print("                                                    \n")
end
```

4 繰り返し while

Ex. 7
以下は配列の要素を表示するプログラムである。(kazu[0], kazu[1]… の順に表示する。) ただし、一部しか表示されない。正しいプログラムに修正しよう。さらに、プログラムの説明を書いて理解しよう。

プログラム	説明

`kazu = [1, 3, 10, 2, 9, 7, 8, 4, 6, 5]`
 	
`i = 0`
`while i < 5`
`  print(kazu[i], "\n")`
`  i = i + 1`
`end`	

Ex. 8
配列の要素を後ろから順に全て表示するように以下のプログラムを修正してみよう。さらに、プログラムの説明を書いて理解しよう。

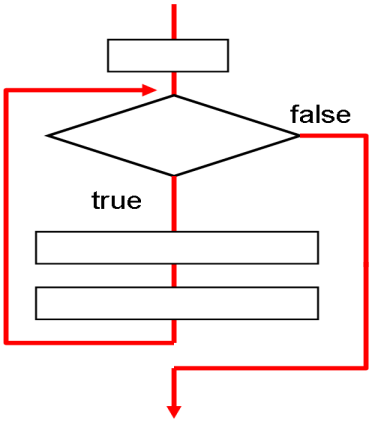
プログラム	説明

`kazu = [1, 3, 10, 2, 9, 7, 8, 4, 6, 5]`
 	
`i = 0`
`while i < 5`
`  print(kazu[i], "\n")`
`  i = i + 1`
`end`	

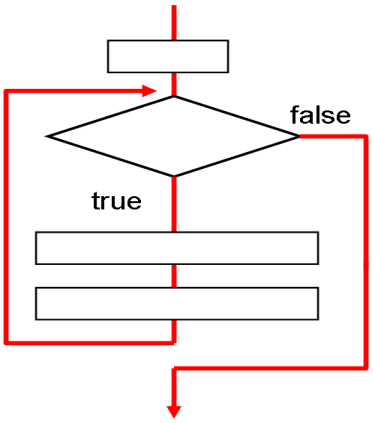
Ex. 9

下記の 2 つのプログラム A と B は、表示結果が異なる。流れ図を完成させて、その理由を考えよう。

```
# プログラム A
week = ["日", "月", "火", "水", "木", "金", "土"]

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




```
# プログラム B
week = ["日", "月", "火", "水", "木", "金", "土"]

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



プログラム A とプログラム B の結果が違う理由

Ex. 10

要素数 4 の配列を用意し、4 個の整数をキーボードから入力し配列に保存する。その配列の要素を全て表示するプログラムをコメントにしたがって完成させよう。

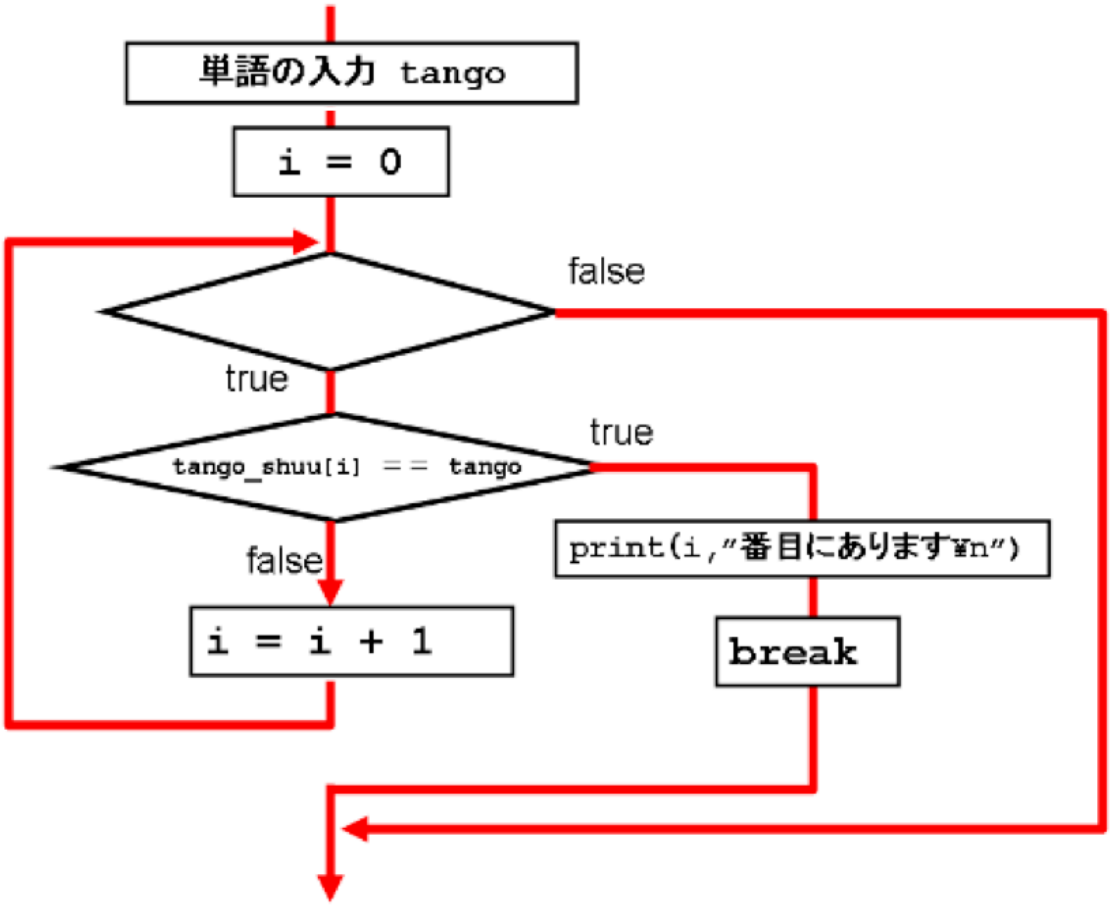
```
kazu =                                  # 配列の宣言　要素数 4 初期値 0
i = 0
while                                   # 配列の要素数だけ繰り返す（4 は使わない）
  print("整数を入力してください \n")
  kazu[i] = gets.chomp.to_i
                                        # ここは繰り返しで重要なポイント！
end
# 配列の要素を全て表示する
i = 0
while



end
```

Ex. 11

説明にしたがってプログラムを作成しよう。繰り返しには while 文を使うこと。

プログラム	説明
tango_shuu = ["library", "book", ,] print("単語を入力してください\n") tango = gets.chomp	4 個の単語を保存した配列 tango_shuu を用意する。 キーボードから入力された単語を tango に代入する。 入力された単語 tango と配列 tango_shuu の要素を順に比較し、一致した場合は「＊番目にあります」と表示し、単語の比較を終了する。

まずはプログラムの流れを確認しよう。流れ図の空欄を埋めてみよう。



Ex. 12

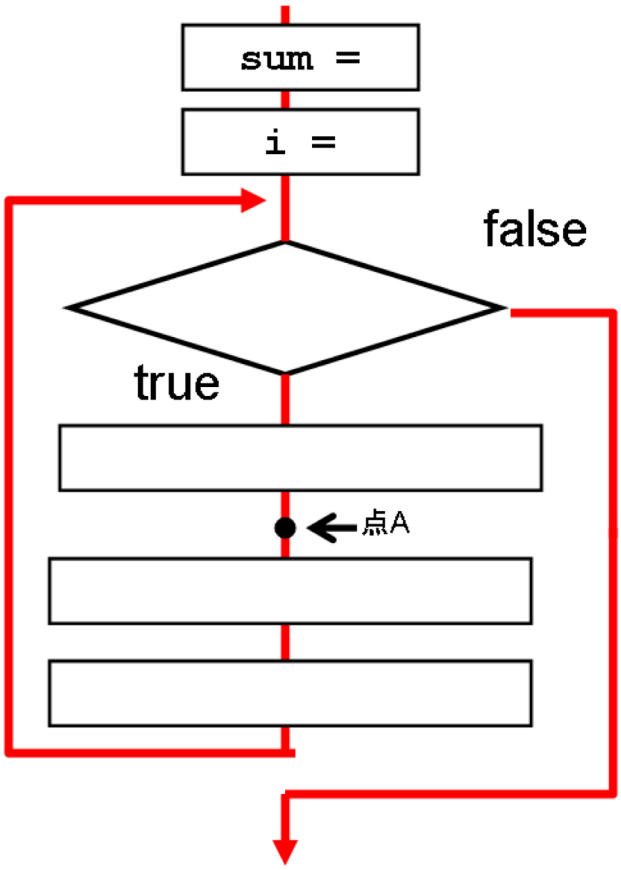
途中で while の繰り返しを終了する時には（＿＿＿＿＿＿＿＿）を使う。

Ex. 13

以下のプログラムの説明を書いて、流れ図を完成させよう。また図の点 A での i の値と sum の値を表にまとめてみよう。さらに、このプログラムによって何が求められるのか考えよう。

プログラム	説明

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



i	0	1	2								
sum	0	1	3								

このプログラムによって求められた sum は（＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿）である。

Ex. 14

キーボードから整数を入力すると、1 から入力された整数までの和を計算するプログラムを作成しよう。まずは、言葉でプログラムを書いてから、実際の ruby プログラムを書いてみよう。

言葉	プログラム
キーボードから入力された整数を変数に代入する	

Ex. 15

下記のプログラムの説明を書いた後、実行せずに i と j と n の値がどう表示されるか表を埋めてみよう。

プログラム	説明

`i = 0`
`n = 0`
`while i < 3`
`  j = 0`
`  while j < 4`
`  print("i = ", i, " j = ", j, " n = ", n, "\n")`
`  n = n + 1`
`  j = j + 1`
`  end`
`  i = i + 1`
`end`	

print 文で表示される i, j, n の値

i	0	0													
j	0	1													
n	0	1													

Ex. 16

以下のように九九表を出力するプログラムを作ってみよう。数字の間は空白 1 個とする。

```
1 2 3 4 5 6 7 8 9
2 4 6 8 10 12 14 16 18
..  ..  ..
9 18 27 36 45 54 63 72 81
```

言葉	プログラム

Ex. 17

指定された数（キーボードから入力する）までの掛け算の表を表示するプログラムを作成してみよう。(12 * 12 まで、19 * 19 までなど)

言葉	プログラム

☆☆ **Ex. 18**

下記に示したように数値を右揃えにして、九九表をきれいに表示するにはどうしたらよいだろうか。前のプログラムで print メソッドの部分を変更してみよう。

```
 1  2  3  4  5  6  7  8  9
 2  4  6  8 10 12 14 16 18
..  ..  ..
 9 18 27 36 45 54 63 72 81
```

言葉	プログラム（表示を右揃えできれいにする部分だけ）

Ex. 19

以下は文字当てゲームである。何をやっているか説明してみよう。

プログラム	説明

`moji = ""`
 	
`while moji != "u"`
`  print("アルファベット 1 文字入力してください\n")`
`  moji = gets.chomp`
`end`
 	
`print("正解\n")`
`print("終了\n")`	

Ex. 20

以下は文字当てゲームである。上のプログラムとの違いを確認しながら何をやっているか説明しよう。こちらは最大何回入力することができるか？

プログラム	説明

`i = 0`
`while i < 10`
`  print("アルファベット 1 文字入力してください\n")`
`  moji = gets.chomp`
 	
`  if moji == "u"`
`  print("正解\n")`
`  break`
`  else`
`  print("残念\n")`
`  end`
 	
`  i = i + 1`
`end`
`print("終了\n")`	

このプログラムでは、アルファベットを最大（＿＿＿＿＿＿＿＿）回入力することができる。

Ex. 21
入力された野菜がお店にあるか答えてくれるプログラムを作成しよう。(STEP-1)
コメントにしたがってプログラムを完成させよう。

```
-------------------------------------------------------------------------------
# ハッシュ yaoya を作成し、野菜名と値段の組を 3 つ保存する
yaoya = {"NINJIN" => 150,                                                    }

# キーボードから入力された野菜名を変数 yasai に代入する
print("かすが商店です。どの野菜をお探しですか？\n")
yasai =

# 入力された野菜がハッシュ yaoya にあるかどうか判定する
#   ある場合は、「***はあります。***円です」と表示する
#   ない場合は、「申し訳ありません。*** は品切れです」と表示する
if yaoya.include?(yasai)
  print(yasai, "はあります。", yaoya[yasai], "円です\n")
else
  print("申し訳ありません。", yasai, "は品切れです\n")
end
-------------------------------------------------------------------------------
```

Ex. 22
上のプログラムに使われているメソッド include? は何か？調べてまとめよう。

Ex. 23

入力された野菜がお店にあるか答えてくれるプログラムを作成しよう。(STEP-2)
STEP-1 で作成したプログラムに下記の機能を追加しよう。

- 野菜名を最大 100 回入力できるようにする。
- . (ピリオド) が入力されたら終了する。

```
-------------------------------------------------------------------------
# ハッシュ yaoya を作成し、野菜名と値段の組を 3 つ保存する
yaoya = {"NINJIN" => 150,                                          }

                                                  # <== 繰り返しを最大 100 回に
while i <                                         # <== するために必要な部分
  # キーボードから入力された野菜名を変数 yasai に代入する
  print("かすが商店です。どの野菜をお探しですか？\n")
  print("終了する場合は、.(ピリオド) を入力してください\n")
  yasai =

  # もし、 .（ピリオド）が入力されたら while 文を抜ける
  if yasai ==
                                                # <== 繰り返しを抜けるには？
  else
    if yaoya.include?(yasai)
      print(yasai, "はあります。", yaoya[yasai], "円です\n")
    else
      print("申し訳ありません。", yasai, "は品切れです\n")
    end
  end
                                            # <== ここに繰り返しに必要な文が入る
end
-------------------------------------------------------------------------
```

Ex. 24

入力された野菜がお店にあるか答えてくれるプログラムを作成しよう。(STEP-3)

STEP-2 で作成したプログラムに下記の機能を追加しよう。

- 野菜がお店 (ハッシュ yaoya) にある場合は、「***はあります。」「他のお店では、***はいくらでしたか？」と聞く。
- 入力された値段が、ハッシュに保存されている値段よりも安い場合は、お客が入力した値段をハッシュに保存し、「じゃ、***円にしておきます」と表示する。高い場合は、「かすが商店は、***円だよ。安いでしょ」と表示する。

```
--------------------------------------------------------------------------
# ハッシュ yaoya を作成し、野菜名と値段の組を 3 つ保存する
yaoya = {"NINJIN" => 150,                                                }



while i <
  # キーボードから入力された野菜名を変数 yasai に保存する
  print("かすが商店です。どの野菜をお探しですか？\n")
  print("終了する場合は、.(ピリオド) を入力してください\n")
  yasai =

  # もし、 .（ピリオド）が入力されたら繰り返しの中から抜ける
  if yasai ==

  else
    if yaoya.include?(yasai)
      print(yasai, "はあります。\n")
      print("他のお店では、", yasai, "はいくらでしたか？")
      # キーボードから入力された野菜の値段を変数 yasune に保存する
      yasune =

      # ハッシュに保存されている値と yasune を比較する
      if yasune <                              # 入力された値段の方が安い場合
        # yasune の値をハッシュに保存する

        print("じゃ、", yasune, "円にしておきます!\n")
      else                                     # 入力された値段の方が高い場合
        print("かすが商店は、", yaoya[yasai], "円だよ。安いでしょ\n")
      end
    else
      print("申し訳ありません。", yasai, "は品切れです\n")
    end
  end
                              # ここに繰り返しに必要な文が入る
end
--------------------------------------------------------------------------
```

☆☆ **Ex. 25**

入力された単語が用意した辞書に登録されているか確かめるプログラムを作成しよう。

- 単語を保存した配列 tango_shuu を作成する。
- 単語をキーボードから入力してもらう。
- 入力された単語が、配列 tango_shuu にあるかどうかを判定し、ある場合は、「辞書の＊＊番目に登録されています」と表示する。
- 単語は 3 回まで繰り返し入力できる。

プログラムの流れを図で書いてから、プログラムを完成させよう。

流れ図

```
# 配列単語 tango_shuu を用意し、英単語を保存する
tango_shuu = [                                                    ]

while i <
  print("単語を入力してください\n")
  tango = gets.chomp
  j = 0
  while
    if                == tango_shuu[j]
      print("辞書の",             , "番目に登録されています\n")
      break
    end

  end

end
```