

## 試験に出るアルゴリズム

### §1. 算術演算子

#### Essence 01

数  $x$  の小数点以下を切り捨てた整数を、算術演算子を用いて求めなさい。

答)  $x \div 1$  または  $x - x \% 1$

**類題** 算術演算子を用いて、数  $x$  を次のような数に変換しなさい。

- ① 小数点以下のみの数
- ② 小数第3位以下を切り捨てた数
- ③ 小数点以下を四捨五入した数
- ④ 100の位を四捨五入した数

答) ①  $x - x \div 1$   
②  $x * 100 \div 1 / 100$   
③  $(x + 0.5) \div 1$   
④  $(x + 50) / 100 \div 1 * 100$

☞  $-7/4, -7 \div 4, -7 \% 4$  はそれぞれいくつか。  
[割られる数] = [割る数] × [商] + [余り]  
ただし  $0 \leq [\text{余り}] < [\text{割る数}]$   
この場合は、 $-7 = 4 \times (-2) + 1$  である。  
答)  $-1.75, -2, 1$

#### Essence 02

500円未満の金額  $x$  円を支払うために必要な硬貨の最低枚数を、硬貨種ごとに求めなさい。

答)  
 $x = \text{【入力(500未満の整数)】}$   
 $\text{Kinsyu} = [100, 50, 10, 5, 1]$   
 $\text{nokori} = x$   
 $\text{maisukei} = 0$   
 $i$  を 0 から 4 まで 1 ずつ増やしながら繰り返す:  
|  $\text{maisukei} = \text{nokori} \div \text{Kinsyu}[i]$   
|  $\text{nokori} = \text{nokori} \% \text{Kinsyu}[i]$   
| 表示する( $\text{Kinsyu}[i]$ , "円:",  $\text{maisukei}$ , "枚")  
|  $\text{maisukei} = \text{maisukei} + \text{maisukei}$   
表示する("計:",  $\text{maisukei}$ , "枚")

**類題** 1g, 2g, 5g, 10g のおもりがそれぞれ十分にある。mgの物を測るための最低のおもりの個数を求めなさい。

答) 略

☞ 国によっては、さらに 2円, 20円, 200円, ... と硬貨の種類を増やしている。これはなぜだろうか。この疑問に応えるシミュレーションを考えて実行してみよう。ところで日本では、2000円札は根付かきませんでした。なぜ日本ではこの試みは失敗したのでしょうか。

#### Essence 03

関数を以下のように定義する。

関数: 乱数( ) ... 0 以上 1 未満の無作為な数  
関数: 整数(数) ... 小数点以下を切り捨てた数  
このとき、次の式の意味することは何ですか。  
 $\text{atai} = \text{整数}(\text{乱数}() * 6) + 1$

答)  $n = \text{乱数}()$   $0 \leq n < 1$  の乱数  
 $n = \text{乱数}() * 6$   $0 \leq n < 6$  の乱数  
 $n = \text{整数}(\text{乱数}() * 6)$   $0, 1, \dots, 5$  の乱数  
 $n = \text{整数}(\text{乱数}() * 6) + 1$   $1, 2, \dots, 6$  の乱数  
従って、  
 $\text{atai}$  はサイコロの目の出方を表している。

**類題** 1,000以上10,000未満の整数の乱数を発生させたい。これらの関数を用いて作成せよ。

答)  $\text{整数}(\text{乱数}() * 9000) + 1000$

☞ 優秀な諸君だから、 $N$ 以上 $M$ 未満の乱数を得るための式はすぐに思いつくだろう。

#### Essence 04

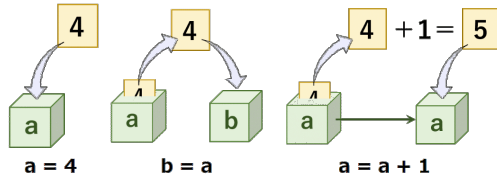
次の各式について、各問いに答えなさい。

- ①  $2 * 3 * 2$  と  $2 ** 3 ** 2$  を求めよ。
- ②  $((4 * x + 1) * x - 2) * x + 3$  を簡単にせよ。

解) ①  $2 * 3 * 2 = 2 \times 3 \times 2 = 6$   
 $2 ** 3 ** 2 = 2^3 = 2^9 = 512$   
②  $((4 * x + 1) * x - 2) * x + 3$   
 $= (4 * x + 1) * x * x - 2 * x + 3$   
 $= 4 * x * x * x + 1 * x * x - 2 * x + 3$   
 $= 4 ** 3 + 1 * x ** 2 - 2 * x + 3$   
すなわち、 $x$  の 3 次式を意味している。  
最後の式より②の式の方が計算が速い。

## § 2. 増分処理

**確認** 数学の変数とは扱いが異なります。



Essence 05

正の整数  $n$  を入力した後、  
1 から  $n$  までの和を求めるプログラムを示せ。

答)  $n =$  【正の整数の入力】

$sum = 0$

$i$  を 1 から  $n$  まで 1 ずつ増やしながら繰り返す

$sum = sum + i$

表示する ("計",  $sum$ )

**トレース** ※計算過程を追いかけてみること

★プログラムの解析には力を発揮します★

$n=5$  としてトレースしてみよう

| $i$ | $sum$ |                  |
|-----|-------|------------------|
| 不定  | 0     | ※ $sum$ : 初期値の設定 |
| 1   | 1     | ※ループ 1 回目終了時     |
| 2   | 3     |                  |
| 3   | 6     |                  |
| 4   | 10    |                  |
| 5   | 15    | ※ループ 5 回目終了時     |
| 5   | 15    | ※表示する            |

**類題** 配列の要素の平均を算出したい。ただし、  
要素の個数は定まっていないものとする。

関数: 要素数 (配列) ... 配列の要素の総個数を返す

$Siken = [67, 45, 78, 60, 92]$       #データ例

$wa = 0$

$i$  を 0 から 要素数( $Siken$ )-1 まで 1 ずつ増やしながら繰り返す:

$sum = sum + Siken[i]$

$heikin = sum / 要素数(siken)$

表示する ("平均",  $heikin$ )

Essence 06

0 以上の整数  $n$  を入力し  $n!$  を求めるプログラム  
を書け。  $0! = 1$  であることに注意すること。

答)  $n =$  【0 以上の整数の入力】

もし  $n == 0$  ならば:

$fact = 1$

そうでなければ:

$fact = 1$

$i$  を 1 から  $n$  まで 1 ずつ増やしながら繰り返す:

$fact = fact * i$

print( $n$ , "!",  $fact$ )

**発展** Essence 06 のプログラムを関数として  
定義する。すなわち、

関数: 階乗 (0 以上の整数  $n$ ) ...  $n!$  の値を返す

☆ この関数を用いて、順列  $nPr$  及び  
組合せ  $nCr$  を求めるプログラムを作成せよ。

・ 順列  $nPr = \frac{n!}{(n-r)!}$  だから

$n =$  【 $nPr$  の  $n$  を入力】

$r =$  【 $nPr$  の  $r$  を入力】

$nPr = \frac{\text{階乗}(n)}{\text{階乗}(n-r)}$

表示する ( $n$ , "P",  $r$ , "=",  $nPr$ )

・ 組合せ  $nCr = \frac{n!}{r!(n-r)!}$  だから

$n =$  【 $nCr$  の  $n$  を入力】

$r =$  【 $nCr$  の  $r$  を入力】

$nCr = \frac{\text{階乗}(n)}{\text{階乗}(r) * \text{階乗}(n-r)}$

表示する ( $n$ , "C",  $r$ , "=",  $nCr$ )

Essence 07

初項 3, 公差 2 の等差数列について

① 初項から第 10 項までの和を求めよ。

② 初項から 20 以下の値の項までの和を求めよ。

答) ①  $a = 3$

#初項

$d = 2$

#公差

$sum = 0$

#和の初期値

$i$  を 1 から 10 まで 1 ずつ増やしながら繰り返す:

$sum = sum + a$

$a = a + d$

表示する ( $sum$ )

②  $a = 3$

#初項

$d = 2$

#公差

$sum = 0$

#和の初期値

$a \leq 20$  の間繰り返す:

$sum = sum + a$

$a = a + d$

表示する ( $sum$ )

※注. 条件を判定する位置から、次項の  $a$  や  
和の計算  $sum = sum + a$  のステップ・位置  
が決まってくる。アルゴリズムを明確  
にすることが大切である。

**類題** 初項 3, 公比 2 の等比数列について

① 初項から第 10 項までの和を求めよ。

② 初項から 1000 未満の値の項までの和を  
求めよ。

答) hint. ・ 例題の  $a = a + d$  が  $a = a * r$  となる。

・ 「1000 未満」は  $a < 1000$  である。

### Essence 08

$a_1 = 1, a_2 = 1, a_{n+2} = a_n + a_{n+1} \ (n \geq 1)$

で定義されるフィボナッチ数列について

- ① 初項から第10項までを表示せよ
- ② 初項から順に100以下の項を表示せよ。

答) ① `a_1 = 1`  
`a_2 = 1`  
`print(a_1)`  
`print(a_2)`  
`a_n = a_1`  
`a_n1 = a_2`  
`i`を3から10まで1ずつ増やしながら繰り返す:  
`a_n2 = a_n + a_n1`  
`表示する(a_n2)`  
`a_n = a_n1`  
`a_n1 = a_n2`

#### 変数テーブル

`a_1, a_1` → `a_1, a_2`  
`a_n` → `a_n`  
`a_n1, a_n2` → `a_{n+1}, a_{n+2}`

#### トレース

|        | i  | a_n | a_n1 | a_n2 |
|--------|----|-----|------|------|
| ループ開始前 | 不定 | 0   | 1    | 不定   |
| ループ1回終 | 3  | 1   | 1    | 1    |
| 2回終    | 4  | 1   | 2    | 2    |
| 3回終    | 5  | 2   | 3    | 3    |
| 4回終    | 6  | 3   | 5    | 5    |
| 5回終    | 7  | 5   | 8    | 8    |
| ⋮      | ⋮  | ⋮   | ⋮    | ⋮    |

② `a_1 = 1`  
`a_2 = 1`  
`print(a_1), print(a_2)`  
`a_n = a_1, a_n1 = a_2`  
`a_n + a_n1` ≤ 100 の間繰り返す  
`a_n2 = a_n + a_n1`  
`表示する(a_n2)`  
`a_n = a_n1`  
`a_n1 = a_n2`

☞ 階差数列やその和, 群数列やその和, ……  
 など数列は自由に考えられますね。

### Essence 09

下のプログラムの実行結果を書きなさい。

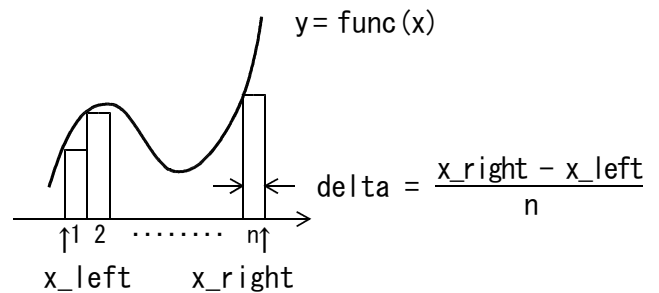
Data = [2, 6, 8, 4, 1]

`i`を0から4まで1ずつ増やしながら繰り返す  
`graph = ""`  
`j`を0からData[i]-1まで1ずつ増やしながら繰り返す  
`graph = "*" + graph`  
`表示する(i, graph)`

答) 0 \*\*  
 1 \*\*\*\*\*  
 2 \*\*\*\*\*  
 3 \*\*\*\*  
 4 \*

### Essence 10

積分区間を決めて関数の定積分を求めよう。



関数: 関数(数値)…x座標からy座標を計算

答) `x_left = 0` #下端  
`x_right = 1` #上端  
`n = 100000` #分割個数  
`delta = (x_right - x_left) / n`  
`sum = 0`  
`k`を1からnまで1ずつ増やしながら繰り返す  
`x = x_left + (k - 1) * delta`  
`sum = sum + 関数(x) * delta`  
`表示する(sum)`

☞ Python で `func = x**2, x_left=0, x_right=1`  
 として実行してみると,  
 • n=10 …… sum=0. 28500000000000001  
 • n=1000 …… sum=0. 332833500000000034  
 • n=100000 …… sum=0. 333328333334999745  
 • n=10000000 …… sum=0. 33333328333333545

### § 3. 初期値の設定

Essence 11

- ① 1から10までの和  $1+2+3+\dots+10$  及び
- ② 1から10までの積  $1\times 2\times 3\times \dots\times 10$  を求めよ。

答) ①  $n = 10$

$sum = 0$

mを1からnまで1ずつ増やしながら繰り返す:

└  $sum = sum + m$

表示する(sum)

②  $n = 10$

$product = 1$

mを1からnまで1ずつ増やしながら繰り返す:

└  $product = product * m$

表示する(product)

#### 研究

【1】①で繰り返しの条件が変わればどうなるか。

$n = 10$

$sum = 1$

mを2からnまで1ずつ増やしながら繰り返す:

└  $sum = sum + m$

表示する(sum)

【2】①を条件による繰り返しに変更する。

特に  $sum = sum + m$  と  $m = m + 1$  の順, 条件式の違いに留意し下線部に適当な数字を入れよ。

a)  $n = 10$

$sum = 0$

$m = 1$

$m \leq n$  の間繰り返す:

$sum = sum + m$

$m = m + 1$

表示する(sum)

b)  $n = 10$

$sum = 0$

$m = 0$

$m < n$  の間繰り返す:

$m = m + 1$

$sum = sum + m$

表示する(sum)

Essence 12

3数を与えて最大値を表示するプログラム

答) ①  $a = 3, b = 7, c = 6$

$m = a$

もし  $b > m$  ならば:

└  $m = b$

もし  $c > m$  ならば:

└  $m = c$

表示("最大値", m)

②  $a = 3, b = 7, c = 6$

もし  $a > b$  ならば:

└  $m = a$

そうでなければ:

└  $m = b$

もし  $c > m$  ならば:

└  $m = c$

表示("最大値", m)

Essence 13

試験の得点及び欠席数を入力し, 得点が30点未満ならば「欠点」, 欠席日数が23日を超えれば「未履修」, それ以外は「修得」と表示するプログラムを作成せよ。(「欠点」にも「未履修」にも該当する場合は, 「未履修」とする)

答) shiken = 【キーボードから入力】

kesseki = 【キーボードから入力】

text = "修得"

もし shiken < 30 ならば:

└ text = "欠点"

もし kesseki > 23 ならば:

└ text = "未履修"

表示する(text)

研究 この Essence11 と等価なプログラムを示す。

shiken = 【キーボードから入力】

kesseki = 【キーボードから入力】

もし shiken < 30 and not kesseki > 23 ならば:

└ text = "欠点"

そうでなくもし kesseki > 23 ならば:

└ text = "未履修"

そうでなければ:

└ text = "修得"

表示する(text)

Essence 14

試験の結果の一覧が配列で与えられている。この一覧から, 最高点と最低点を探し出せ。

答) 【最高点】

Ten = [47, 85, 62, 69, 83, 60, 97, 53, 38, 65, 73, 57]

ninzu = 要素数(Ten)

ten\_max = 0

iを0から ninzu-1 まで1ずつ増やしながら繰り返す

└ もし ten\_max < Ten[i] ならば:

└ └ ten\_max = Ten[i]

ten\_min = 100

iを0から ninzu-1 まで1ずつ増やしながら繰り返す

└ もし ten\_min > Ten[i] ならば:

└ └ ten\_min = Ten[i]

表示する("最高点:", ten\_max, "最低点:", ten\_min)

## § 4. 値の入れ替え

Essence 15

変数 a の値と変数 b の値を入れ替えよ。

答) temp = a  
a = b  
b = temp

**類題** 配列 Data の任意の要素を入れ替えよう。

```
Data = ["晴", "雨", "風", "雪", "霜", "霰"]
index1 = int(input())
index2 = int(input())
temp = Data[index1]
Data[index1] = Data[index2]
Data[index2] = temp
print(Data)
```

Essence 16

配列 Data = [7, 3, 8, 0, 2, 3, 6, 9, 3, 7, 4, 1, 5]  
の要素を小さい順に並べ替えよ。

※ここではバブルソートを用いた単純なものを扱う。

☞ [2] 情報 I 授業内容から No. 6 探索と整理

答) Data = [7, 3, 8, 0, 2, 3, 6, 9, 3, 7, 4, 1, 5]  
表示する ("ソート前の配列:", Data)  
n = 要素数(Data)  
i を 0 から n-2 まで 1 ずつ増やしながら繰り返す:  
| j を 0 から n-2 まで 1 ずつ増やしながら繰り返す:  
| | もし Data[j] > Data[j + 1] ならば:  
| | | temp = Data[j]  
| | | Data[j] = Data[j+1]  
| | | Data[j+1] = temp  
表示する ("ソート後の配列:", Data)

Essence 17

2つの整数を入力し、最大公約数と最小公倍数を求めるプログラムを作成せよ。

※ユークリッドの互除法を用いる。(最下段の図参照)

### 変数テーブル

a, b ..... 元の 2 数 (入力)  
nl ..... 比較 2 数のうち大きい方 large  
ns ..... 比較 2 数のうち小さい方 small  
r ..... 除算を行った余り  
G ..... 最大公約数 G. C. D.  
L ..... 最小公倍数 L. C. M

### トレース

答) a = 【キーボード入力】  
b = 【キーボード入力】  
もし a >= b ならば:

| nl = a  
| ns = b

そうでなければ:

| nl = b  
| ns = a

ns > 0 の間繰り返す:

| r = nl % ns  
| nl = ns  
| ns = r

G = nl

L = a \* b / G

表示する (a, b, "⇒", G, L)

a=221, b=493 として

| a   | b   | nl  | ns | r   |
|-----|-----|-----|----|-----|
| 221 | 493 |     |    |     |
|     | 493 |     |    |     |
|     |     | 221 |    |     |
|     |     |     | 51 |     |
|     |     | 221 |    |     |
|     |     |     | 51 |     |
|     |     |     |    | 17  |
|     |     |     | 17 |     |
|     |     |     |    | 0   |
|     |     |     | 17 |     |
|     |     |     |    | 0   |
|     |     |     |    | → G |

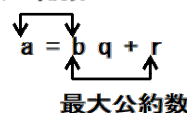
☞ 最大公約数が求まると、分数の約分ができますね!  
フローチャート ⇒ [joho.ms/20222/13.html](http://joho.ms/20222/13.html)

## ユークリッド (Euclid) の互除法

※敷き詰められる最大の正方形を探索して最大公約数を得る手法 (ユークリッドの原論では線分図を用いている)

自然数 a を自然数 b の最大公約数は、  
a を b で割った商を q、余りを r とするとき、  
・ r > 0 のとき、b と r の最大公約数に等しい。  
・ r = 0 のとき、b に等しい。

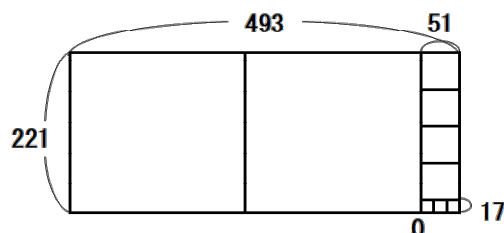
最大公約数



### 【証明の指針】

- ・ a と b の公約数 → b と r の公約数
- ・ b と r の公約数 → a と b の公約数
- ・ a と b の公約数の集合と、  
b と r の公約数の集合は完全に一致
- ・ 当然、最大公約数も一致

Example) 221 と 493 の最大公約数



$$493 = 221 \cdot 2 + 51$$

$$221 = 51 \cdot 4 + 17$$

$$51 = 17 \cdot 3 [+ 0]$$

$$\begin{array}{r} 3 \quad 4 \quad 2 \\ 17 \overline{) 51} \overline{) 221} \overline{) 493} \\ \underline{51} \quad \underline{204} \quad \underline{442} \\ 0 \quad 17 \quad 51 \end{array}$$

$$\begin{array}{r} 221 \quad 493 \\ 4 \quad \underline{221} \quad \underline{51} \\ 204 \quad 51 \\ 17 \quad \underline{51} \\ 51 \\ 0 \end{array}$$

以上より 最大公約数は G. C. D = 17

ところで、最小公倍数 L. C. M は、 $221 \cdot 493 / 17 = 6409$  である。

## § 5. 条件設定

## Essence 18

条件により繰り返す手続きにおいて、条件や計算順を変えると結果が全く異なることを確認せよ。

確認問題)

|  |   |
|--|---|
| <p>① sum = 0<br/>         i = 0<br/>         i &lt; 3の間繰り返す:<br/>             sum = sum + i<br/>           └ i = i + 1<br/>         表示する(i, sum)</p> | <p>② sum = 0<br/>         i = 0<br/>         i &lt;= 3の間繰り返す:<br/>             sum = sum + i<br/>           └ i = i + 1<br/>         表示する(i, sum)</p> |
|--|---|

```

③ sum = 0
   i = 0
   i < 3の間繰り返す:
   | i = i + 1
   | sum = sum + i
   表示する(i, sum)

④ sum = 0
   i = 0
   i <= 3の間繰り返す:
   | i = i + 1
   | sum = sum + i
   表示する(i, sum)

```

出力確認)

$$\begin{array}{r} \textcircled{1} \quad 33 \\ \textcircled{3} \quad 36 \\ \hline \end{array} \qquad \begin{array}{r} \textcircled{2} \quad 46 \\ \textcircled{4} \quad 410 \\ \hline \end{array}$$

## Essence 19

国語, 数学, 英語の試験を行い, 2教科以上で50点以上をとれば, 今日の掃除当番は免除される。この関係のプログラムを完成させよ。

## 変数テーブル

k: 国語点数, s: 数学点数, e: 英語点数  
r: 結果 (掃除 または 免除)

※3教科の試験結果 k, s, e が与えられた後…

```

① もし  $k \geq 50$  ならば:
    |   もし  $s \geq 50$  or  $e \geq 50$  ならば:
    |       |    $r = \text{"免除"}$ 
    |   そうでなくもし  $s \geq 50$  and  $e \geq 50$  ならば:
    |       |    $r = \text{"免除"}$ 
    |   そうでなければ:
    |       |    $r = \text{"当番"}$ 

```

```

② r="当番"
  もし  $k > 50$  ならば:
    |   もし  $s > 50$  ならば:
    |   |    $r = \text{"免除"}$ 
    |   |   もし  $e > 50$  ならば:
    |   |   |    $r = \text{"免除"}$ 
  そうでなければ:
    |   もし  $s > 50$  ならば:
    |   |   もし  $e > 50$  ならば:
    |   |   |    $r = \text{"免除"}$ 

```

※ 1 行で論理式を書くこともできます。

(ko)>=50 and su>=50) or (ko)>=50 and ei>=50) or (su)>=50 and ei>=50)

- Essence 20

素数を発見(生成)しよう！  
対象とする整数の約数の個数を調べ、それが  
2個であれば素数と判定していく手法である。  
見つけた素数を配列に順に格納し、最後にそ  
の配列を表示している。

## 変数テーブル

```
n_max : 素数探索の上限
Prime : 素数一覧用の配列  indx : 添字
koho : 素数であるか否かの判定対象
yakusu : koho の約数の個数
```

```

答) n_max = 100
配列変数 Prime を初期化する
indx=0
kohoを1からn_maxまで1ずつ増やしながら繰り返す:
|   yakusu = 0
|   nを1からkohoで1ずつ増やしながら繰り返す:
|   |   もし koho % n == 0 ならば:
|   |   |   L   L   yakusu = yakusu + 1
|   |   もし yakusu == 2 ならば:
|   |   |   |   Prime[indx] = koho
|   |   L   L   indx = indx + 1
表示する(Prime)

```

## § 6. 配列における添字(インデックス)の活用

Essence 21

都道府県名がコード番号で与えられている。  
0. 東京 1. 千葉 2. 神奈川 3. 埼玉 4. 茨城  
コード番号を入力して県名を表示させたい。

答) Pref = [東京, 千葉, 神奈川, 埼玉, 茨城]  
n = 【コード番号の入力】  
表示する(Pref[n])

Essence 22

〔統計データ処理〕40人の試験結果のデータ  
8, 6, 4, 8, 8, 7, 4, 8, 7, 9, 7, 5, 9, 6, 5, 6, 8, 7, 5, 8,  
7, 8, 6, 7, 8, 7, 4, 4, 7, 4, 6, 6, 7, 6, 3, 8, 7, 5, 7, 9  
の度数分布を作成せよ。

答) 配列変数 Data に値を入力する  
配列変数 Stat を初期化する(要素数11)  
i を1から10まで1ずつ増やしながら繰り返す:  
    | Stat[i] = 0  
i を0から39まで1ずつ増やしながら繰り返す:  
    | Stat[Data[i]] = Stat[Data[i]] + 1  
i を1から10まで1ずつ増加させ繰り返す:  
    | 表示する(i, "点:", Stat[i], "人")

❖出力結果 ⇒ 1 点: 0 人  
                  2 点: 0 人  
                  3 点: 1 人  
                  4 点: 5 人  
                  5 点: 4 人  
                  6 点: 7 人  
                  7 点: 11 人  
                  8 点: 9 人  
                  9 点: 3 人  
                  10 点: 0 人

〔発展〕 続けて Essence 09 で学んだグラフ化の手法  
を用いてグラフ化してみると  
i を1から10まで1ずつ増やしながら繰り返す  
    | graph = ""  
    | j を0からStat[i]まで1ずつ増やしながら繰り返す  
        | graph = "\*" + graph  
    | 表示する(i, graph)

❖出力結果 ⇒ 1  
                  2  
                  3 \*  
                  4 \*\*\*\*\*  
                  5 \*\*\*\*  
                  6 \*\*\*\*\*  
                  7 \*\*\*\*\*  
                  8 \*\*\*\*\*  
                  9 \*\*\*  
                  10

Essence 23

「いちかわがくえん」で待っています。  
と彼女は、鍵「6, 0, 4, 2, 7」を渡して去って行っ  
てしまった。さて、この謎を解こう！

答) Data=["い","ち","か","わ","が","く","え","ん"]  
Key=[6, 0, 4, 2, 7]  
表示する(Data[Key[0]], Data[Key[1]], ...,  
            ..., Data[Key[4]])

❖出力結果 ⇒ え い が か ん

〔類題〕 学校へ行って勉強しなくて良いのか、と  
彼女に問うたら、黙って「1, 6, 4, 3, 5」を  
渡してくれた。いったい……何を言  
いたかったのだろうか。

※ここでは全てを復号してから表示する手法をとる。

答) Data=["い","ち","か","わ","が","く","え","ん"]  
Key=[1, 6, 4, 3, 5]  
word = ""  
i を0から4まで1ずつ増やしながら繰り返す  
    | word = word + Data[Key[i]]  
print(word)

❖出力結果 ⇒ ち え が わ く

〔発展〕 下の怪文書(secret)に秘められた真実を、  
鍵(key1 と key2) を使って暴こう！

secret="add eill mmooorr tv"  
key1=[17, 11, 9, 3, 10, 0, 14, 18, 12, 6, 13, 8, 15, 5, 1, 2, 7, 4, 16]  
key2=[13, 3, 5, 2, 11, 9, 1, 6, 14, 18, 7, 8, 16, 15, 17, 4, 10, 12, 0]

答) secret="add eill mmooorr tv"  
Key1=[17, 11, 9, 3, 10, 0, 14, 18, 12, 6, 13, 8, 15, 5, 1, 2, 7, 4, 16]  
Key2=[13, 3, 5, 2, 11, 9, 1, 6, 14, 18, 7, 8, 16, 15, 17, 4, 10, 12, 0]  
word1 = ""  
i を0から要素数(key1)-1まで1ずつ増やしながら繰り返す:  
    | word1 = word1 + secret[Key1[i]]  
word2 = ""  
i を0から要素数(key2)-1まで1ずつ増やしながら繰り返す:  
    | word2 = word2 + word1[Key2[i]]  
print(word1)  
print(word2)

❖出力結果 ⇒ tom marvolo riddle  
                  i am lord voldemort