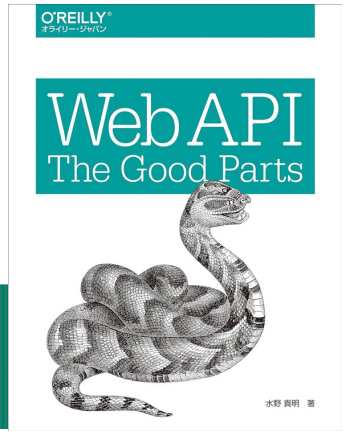
水野貴明

難解プログラミング言語入門

自己紹介



- 水野貴明 (Takaaki Mizuno)
- 英AIスタートアップ Nexus FrontierTech Co-founder / CTO
- 出版系SIer → はてな → Baidu → DeNA → 現職など
- 普段の仕事
  - ソフトウエア設計/開発
  - スタートアップの開発チーム立ち上げ支援
  - 海外開発チームのセットアップ/マネジメント支援
  - 炎上案件の鎮火 / メンテできないコードの調査
  - 本の執筆/翻訳

ストレンジコード
日本語訳2/16発売

- プログラミング言語の歴史から始まり、現代の主流言語とは違う言語や、難解プログラミング言語を紹介
- 最終的に難解プログラミング言語を自作
- 難解プログラミング言語を使って機械学習や遺伝的アルゴリズムにも挑戦
- 604ページの読み応えのある書籍

ストレンジコード
日本語訳2/16発売

- 1章 歴代のプログラミング言語たち
- 2章 プログラミング言語の本質
- 3章 チューリングマシンとチューリング完全
- 4章 Forth
- 5章 SNOBOL
- 6章 CLIPS
- 7章 ABC
- 8章 FRACTRAN
- 9章 Piet
- 10章 Brainfuck
- 11章 Befunge
- 12,13章 Filska
- 14,15章 Firefly

本セッションの目的

- 難解プログラミング言語とは何かが理解できる
- 難解プログラミング言語の面白さに興味を持つ
- 「ストレンジコード」を読みたくなる

前フリ

難解プログラミング言語とは

難解プログラミング言語とは、意図的に読解が困難なように設計されたプログラミング言語である。英語では、Esoteric programming language（略してesolangとも）と言われる。

基本的には、実用性を目指したものではなく、ジョークのプログラミング言語の一種で、いわゆるハッカーの間では、この種のジョークはたしなみとみなされており、難解プログラミング言語に区分されるプログラミング言語はいくつも作られてきた

Wikipedia 「難解プログラミング言語」より

https://ja.wikipedia.org/wiki/%E9%9B%A3%E8%A7%A3%E3%83%97%E3%83%AD%E3%82%B0%E3%83%A9%E3%83%9F%E3%83%B3%E3%82%B0%E8%A8%80%E8%AA%9E

BF(BrainF*ck)によるHello World

```
>+++++++++[<+++++++++>-]<.>+++++++
[<++++>-]<+.+++++++..+++.
[-]>++++++++
[<++++>-]<.>++++++++++
[<+++++>-]<.>++++++++[<+++>-]
<.+++.------.--------.[-]>++++++++[<++++>-]<+.
[-]++++++++++.
```

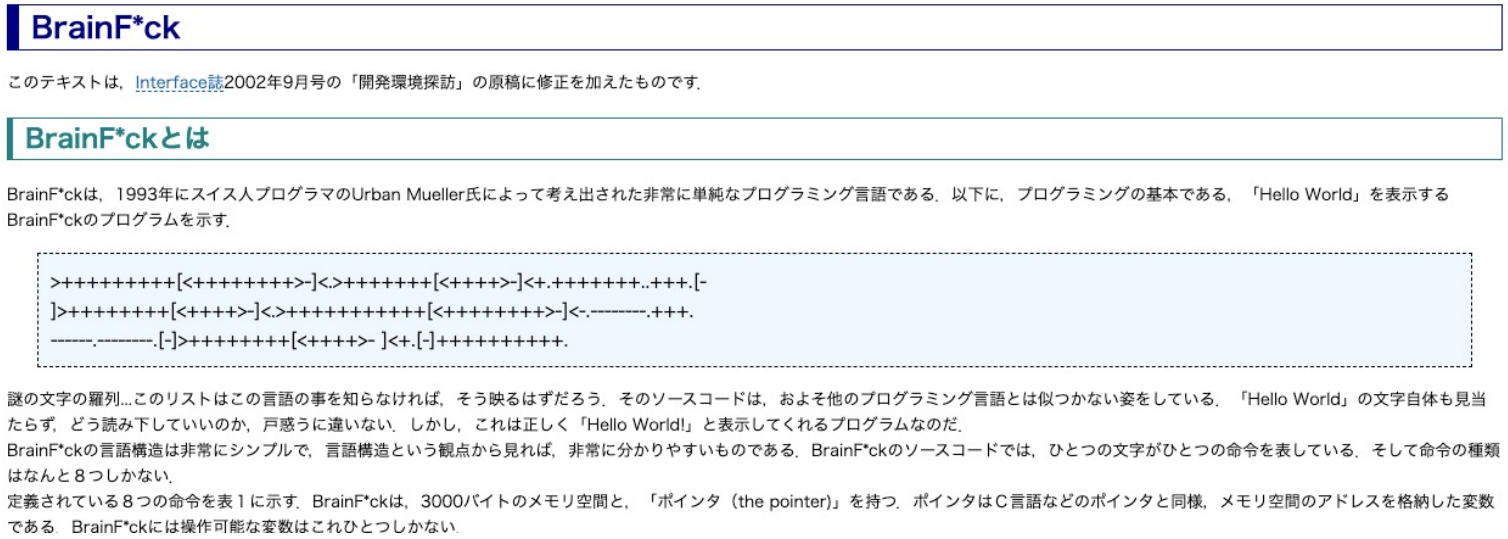
PietによるHello World

BefungeによるHello World

```
>                     v
v"Hello World!"<
>:v
^,_@
```

難解プログラミング言語と私

2002年、雑誌「Interface」にてBF紹介



BrainF*ck

このテキストは，Interface誌2002年9月号の「開発環境探訪」の原稿に修正を加えたものです．

BrainF*ckとは

BrainF*ckは，1993年にスイス人プログラマのUrban Mueller氏によって考え出された非常に単純なプログラミング言語である．以下に，プログラミングの基本である，「Hello World」を表示するBrainF*ckのプログラムを示す．

```
>+++++++++[<++++++++>-]<.>+++++++[<++++>-]<+.+++++++..+++.[-
]>++++++++[<++++>-]<.>++++++++++[<+++++++++>-]<-.--------.+++.
------.--------.[-]>++++++++[<++++> ]<+.[-]++++++++++.
```

謎の文字の羅列…このリストはこの言語の事を知らなければ，そう映るはずだろう．そのソースコードは，およそ他のプログラミング言語とは似つかない姿をしている．「Hello World」の文字自体も見当たらず，どう読み下していいのか，戸惑うに違いない．しかし，これは正しく「Hello World!」と表示してくれるプログラムなのだ．
BrainF*ckの言語構造は非常にシンプルで，言語構造という観点から見れば，非常に分かりやすいものである．BrainF*ckのソースコードでは，ひとつの文字がひとつの命令を表している．そして命令の種類はなんと８つしかない．
定義されている８つの命令を表１に示す．BrainF*ckは，3000バイトのメモリ空間と，「ポインタ（the pointer)」を持つ．ポインタはC言語などのポインタと同様，メモリ空間のアドレスを格納した変数である．BrainF*ckには操作可能な変数はこれひとつしかない．

Wikipediaの「難解プログラミング言語」の項目から、当時の記事（をWebにあげたもののInternet Archive）にリンクあり

https://ja.wikipedia.org/wiki/%E9%9B%A3%E8%A7%A3%E3%83%97%E3%83%AD%E3%82%B0%E3%83%A9%E3%83%9F%E3%83%B3%E3%82%B0%E8%A8%80%E8%AA%9E

2004年「Ultimate Perl」発売

- 様々なJAPH (Just Another Perl Hacker)スクリプトを紹介

```
        $LOVE=                  AMOUR.
     true.cards.           ecstacy.crush
   .hon.promise.de      .votion.partners.
  tender.truelovers.  treasure.affection.
devotion.care.woo.baby.ardor.romancing.
enthusiasm.fealty.fondness.turtledoves.
lovers.sentiment.worship.sweetling.pure
.attachment.flowers.roses.promise.poem;
 $LOVE=~ s/AMOUR/adore/g; @a=split(//,
  $LOVE); $o.= chr (ord($a[1])+6). chr
   (ord($a[3])+3). $a[16]. $a[5]. chr
    (32). $a[0]. $a[(26+2)]. $a[27].
     $a[5].$a[25]. $a[8].$a[3].chr
       (32).$a[29]. $a[8].$a[3].
        $a[62].chr(32).$a[62].
         $a[2].$a[38].$a[4].
             $a[3].'.';
               print
                $o;
```

https://www.cpan.org/misc/japh

ストレンジコード 日本語訳2/16発売

20年の時を経て再び難解プログラミング言語に帰ってきました

BF入門

BF (BrainF*ck)

- 1993年にスイス人プログラマのUrban Müller氏がコンパイラがなるべく小さくなる言語として考案
  - コンパイラのサイズはわずか123バイト、インタプリタは98バイト
- チューリング完全
- わずか8種類の命令セットで構成されている
- 30000個以上の要素を持つバイト配列、データポインタ、入出力ストリームが用意されている

BFの命令セット

命令	説明
>	データポインタをインクリメントする
<	データポインタをデクリメントする
+	データポインタが示す場所の値をインクリメントする
-	データポインタが示す場所の値をデクリメントする
.	データポインタが示す場所の値をASCII文字として出力
,	1文字入力を受け付け、データポインタが示す場所にASCIIコードを格納
[	ポインタが指す値が0なら、対応する] の直後にジャンプ
]	ポインタが指す値が0でないなら、対応する [（の直後）にジャンプ

Hの出力部分を読んでみよう

```
>+++++++++[<++++++++>-]<.
```

“H”のアスキーコード = 72

コマンド	意味
>	データポインタをループカウンタの位置に移動
+++++++++	ループカウンタを9回インクリメント
[	ループ開始
<	データポインタを出力データの位置に移動
++++++++	出力データの値を8回インクリメント
>	データポインタをループカウンタの位置に移動
-	ループカウンタをデクリメント
]	ループカウンタが0になるまでループ
<	データポインタを出力データの位置に移動
.	データポインタ（8x9で72になっている）を出力

データ領域

0 … 出力するデータ
1 … ループカウンタ

BF interpreter in BF

```
#!/usr/bin/bf

[
Brainfuck interpreter in brainfuck

example:
$ echo '++++++++++[>+++++++>++++++++++>+++>+<<<<-]>++.>+.+++++++..+++.>++.<<'¥
   '+++++++++++++++.>.+++.------.--------.>+.>.!' | bf bf.bf

A '!' separates program input from the input into the program.

Currently limited to 255 char programs as the program counter and etc are one
byte. (Can't run itself yet :()

See <https://github.com/canoon/bfbf/blob/master/bf.rb> for how it works.
]

[-]+>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>[-]<<<<<<<<<<<<<<<<<<<<<<
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<[>[-],>[-]++++++++++++++++++++++++++++++++++>>>[-]>>[-
]<<<<<<[->>>>+>>+<<<<<<]>>>>>>[-<<<<<<+>>>>>>]<[-]>[-]<<<<<[->>>>+>+<<<<<]>>>>>[
-<<<<<+>>>>>]<<<[-]+[[-]>>>[-]>[-]<<<[->>+>+<<<]>>>[-<<<+>>>]<[>[-]>[-]<<<[->>+>
+<<<]>>>[-<<<+>>>]<[<<<<[-]+>->->>[-]]<[-]]<<<]<[-]+>>>>[-]>[-]<<<[->>+>+<<<]>>>
[-<<<+>>>]<[<<<<[-]>>>>[-]][-]>[-]<<[->+>+<<]>>[-<<+>>]<[<<<<[-]>>>>[-]]<<<[-]>[
-]<<[->+>+<<]>>[-<<+>>]<[<<<<[-]>>>>[-]]<<[-]>[-]<<<[->>+>+<<<]>>>[-<<<+>>>]<[>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>[-]<<<<<<<<<<<<<<<<<<<<<<
```

チューリング完全なので
BFでBFのインタプリタも書ける

https://github.com/canoon/bfbf

BFのバリアントはたくさんある

- ８個の命令が作れれば良いので、三文字種あれば2つの組み合わせでバリアントが作れる

Ook!

Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook! Ook? Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook? Ook! Ook! Ook? Ook! Ook? Ook. Ook! Ook.
Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook!
Ook? Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook? Ook! Ook! Ook?
Ook! Ook? Ook. Ook. Ook. Ook! Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook! Ook. Ook! Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook. Ook.
Ook? Ook. Ook? Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook? Ook! Ook! Ook? Ook! Ook? Ook. Ook! Ook. Ook. Ook? Ook. Ook? Ook. Ook?
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook! Ook? Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook? Ook! Ook! Ook? Ook! Ook? Ook.
Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook. Ook? Ook. Ook? Ook. Ook? Ook. Ook? Ook.
Ook! Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook. Ook! Ook! Ook! Ook! Ook! Ook! Ook!
Ook! Ook! Ook! Ook! Ook! Ook! Ook. Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook!
Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook. Ook. Ook? Ook. Ook? Ook. Ook. Ook!
Ook.

https://esolangs.org/wiki/Ook!

Whitespace

https://en.wikipedia.org/wiki/Whitespace_(programming_language)

そのほかの難解プログラミング言語

Fractran

分数だけでできたプログラミング言語（チューリング完全）

$$\left(\frac{3}{2}\right)$$

☜ 足し算を行うプログラム

$2^a 3^b$ という数値を与えると、3^{a+b} という答えが出力される

https://ja.wikipedia.org/wiki/FRACTRAN

Fractran

1. 整数に分数を順番に掛ける
2. 積が整数なら、それを新しい整数として利用する。そして、ステップ1から繰り返す
3. 積が整数ではないなら、次の分数に移る
4. 全ての分数を使い果たしたら終了する

$$\left(\frac{3}{2}\right)$$

$108\ (\ 2^2 * 3^3\ )$

$\rightarrow 162\ (\ 2^1 * 3^4\ )$

$\rightarrow 243\ (\ 2^0 * 3^5\ )$

Fractranの引き算プログラム

$$\left(\frac{1}{6}\right)$$

$$2^a 3^b \rightarrow 2^{a-b}$$

Fractran（コンウェイのPRIMEGAME）

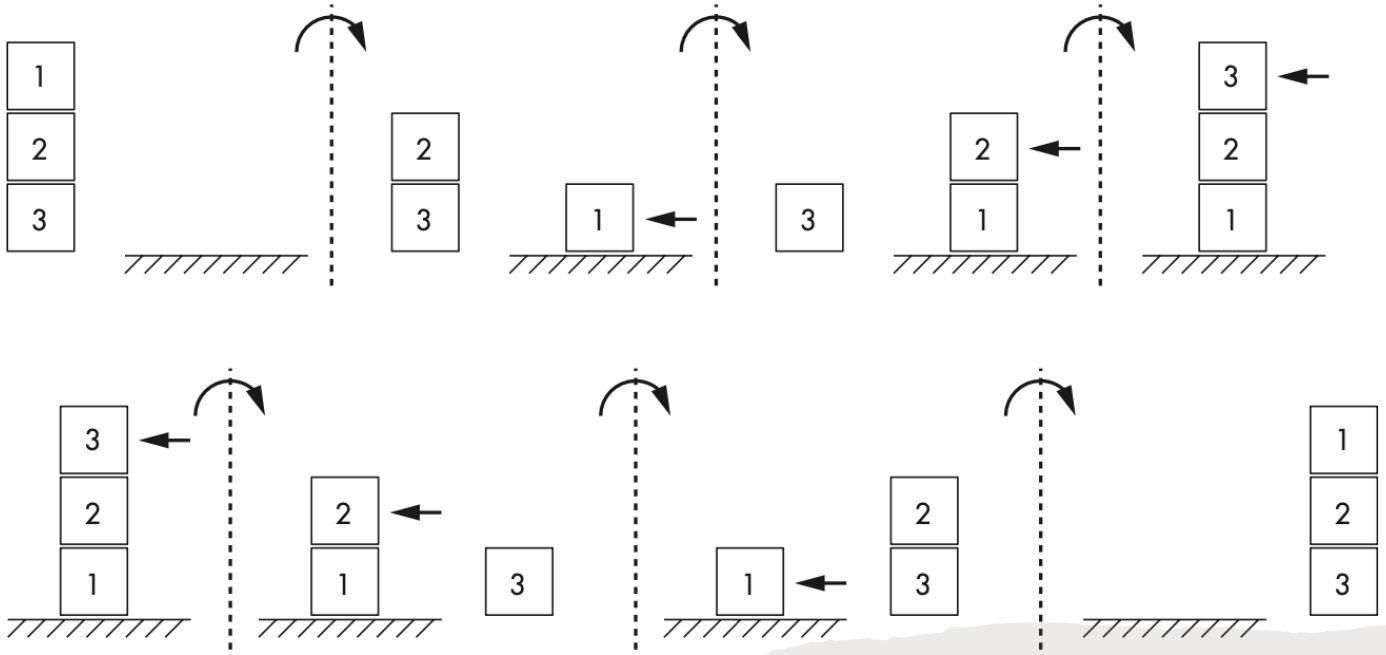
$$\left(\frac{17}{91}, \frac{78}{85}, \frac{19}{51}, \frac{23}{38}, \frac{29}{33}, \frac{77}{29}, \frac{95}{23}, \frac{77}{19}, \frac{1}{17}, \frac{11}{13}, \frac{13}{11}, \frac{15}{2}, \frac{1}{7}, \frac{55}{1}\right)$$

素数を順番に出力するプログラム

最初の数として2を与えると、$2^2 \rightarrow 2^3 \rightarrow 2^5$ … と素数を出力する

スタック志向プログラミング言語

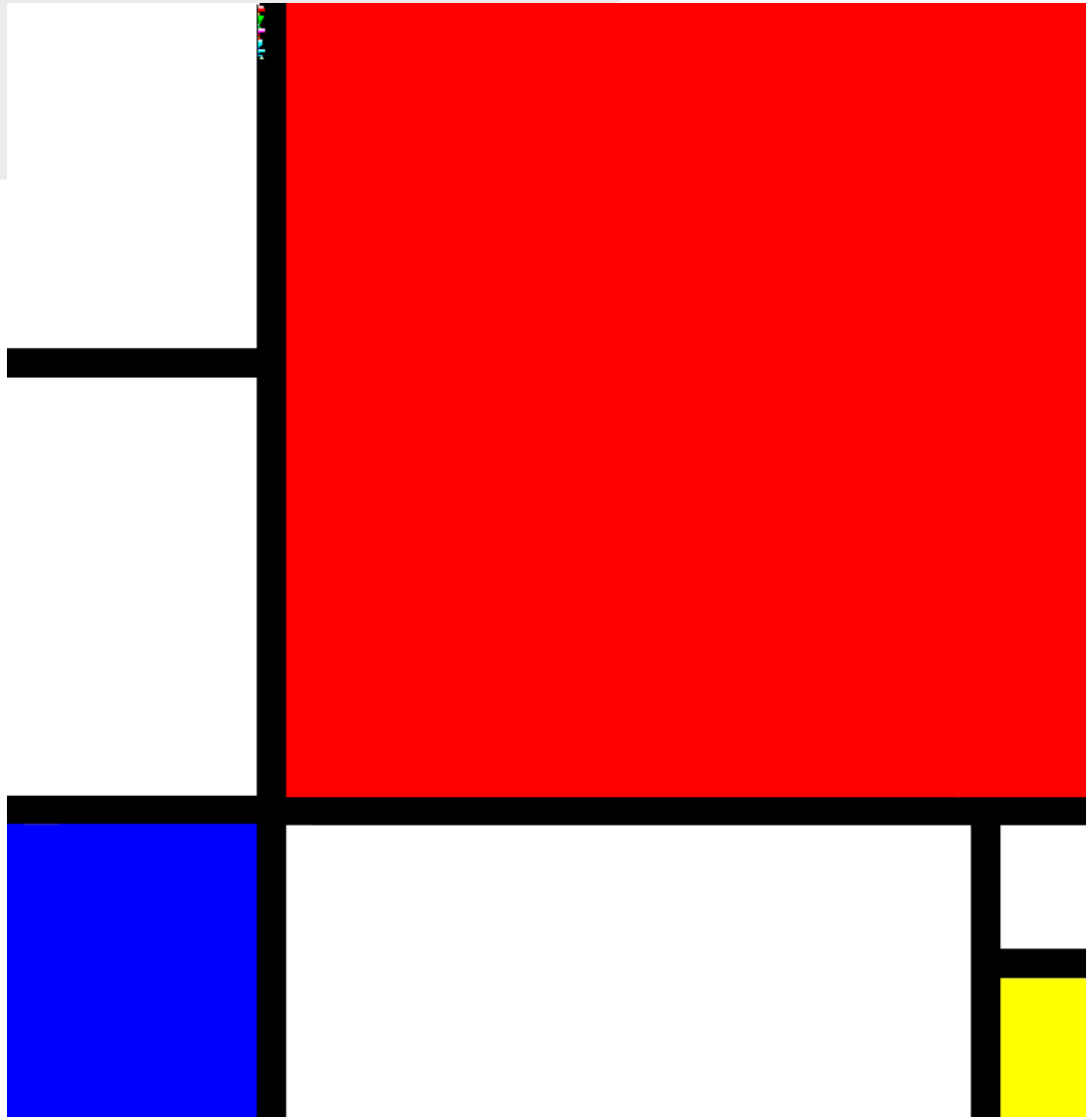
スタックを利用することで、変数なしの言語が作れる（実装が楽になる）



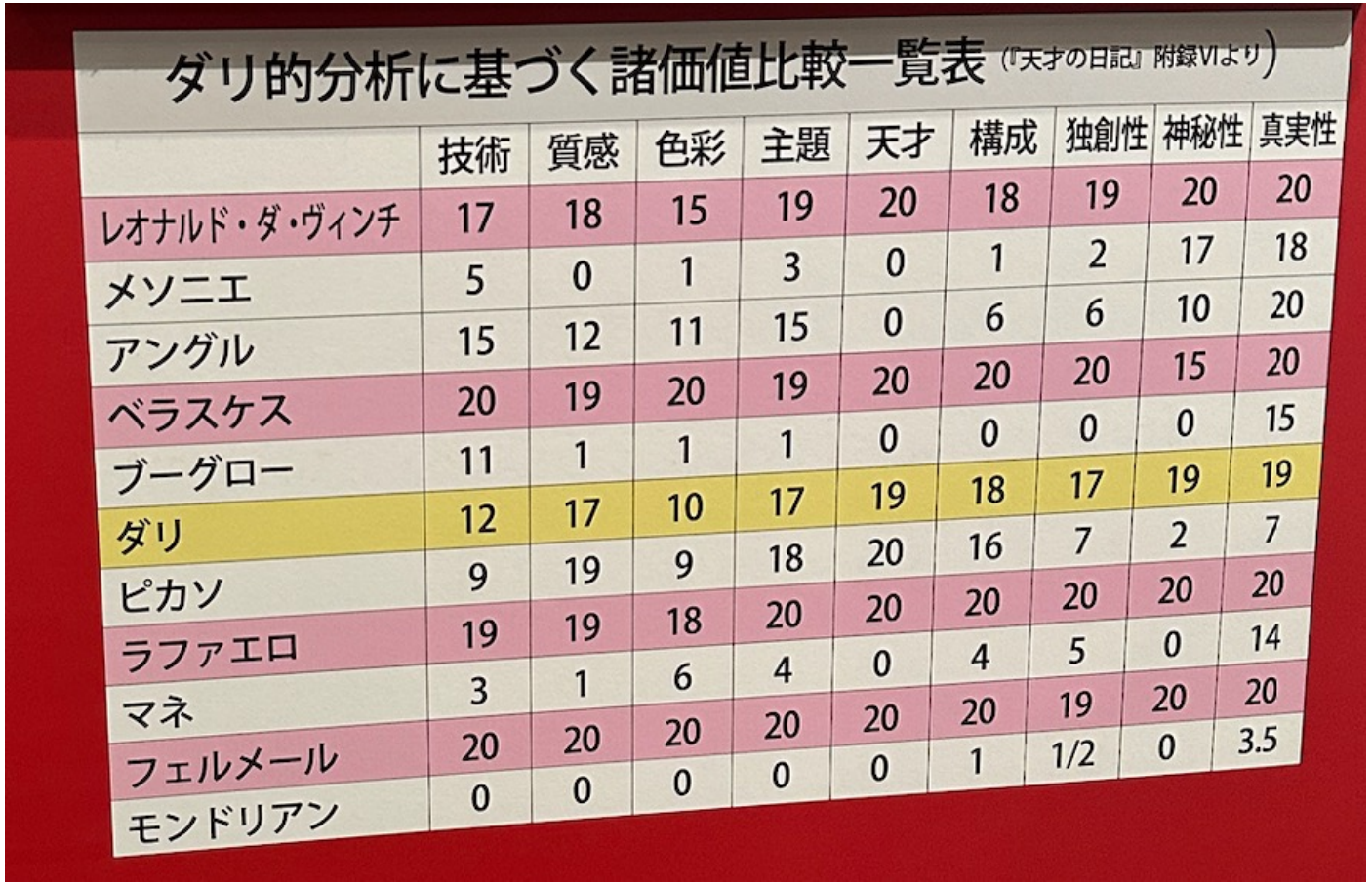
▲図4-1　Forthスタックへの値の追加（上）、値の取り出し（下）

Piet

- ピエト・モンドリアン（オランダの画家）にちなんで名付けられた画像でコーディングする言語
- ピクセルサイズと色相、明度の変化でプログラムを表す
- スタック型プログラミング言語



余談: ダリはモンドリアンは嫌い



ダリ的分析に基づく諸価値比較一覧表（『天才の日記』附録VIより）

	技術	質感	色彩	主題	天才	構成	独創性	神秘性	真実性
レオナルド・ダ・ヴィンチ	17	18	15	19	20	18	19	20	20
メソニエ	5	0	1	3	0	1	2	17	18
アングル	15	12	11	15	0	6	6	10	20
ベラスケス	20	19	20	19	20	20	20	15	20
ブーグロー	11	1	1	1	0	0	0	0	15
ダリ	12	17	10	17	19	18	17	19	19
ピカソ	9	19	9	18	20	16	7	2	7
ラファエロ	19	19	18	20	20	20	20	20	20
マネ	3	1	6	4	0	4	5	0	14
フェルメール	20	20	20	20	20	20	19	20	20
モンドリアン	0	0	0	0	0	1	1/2	0	3.5

Piet

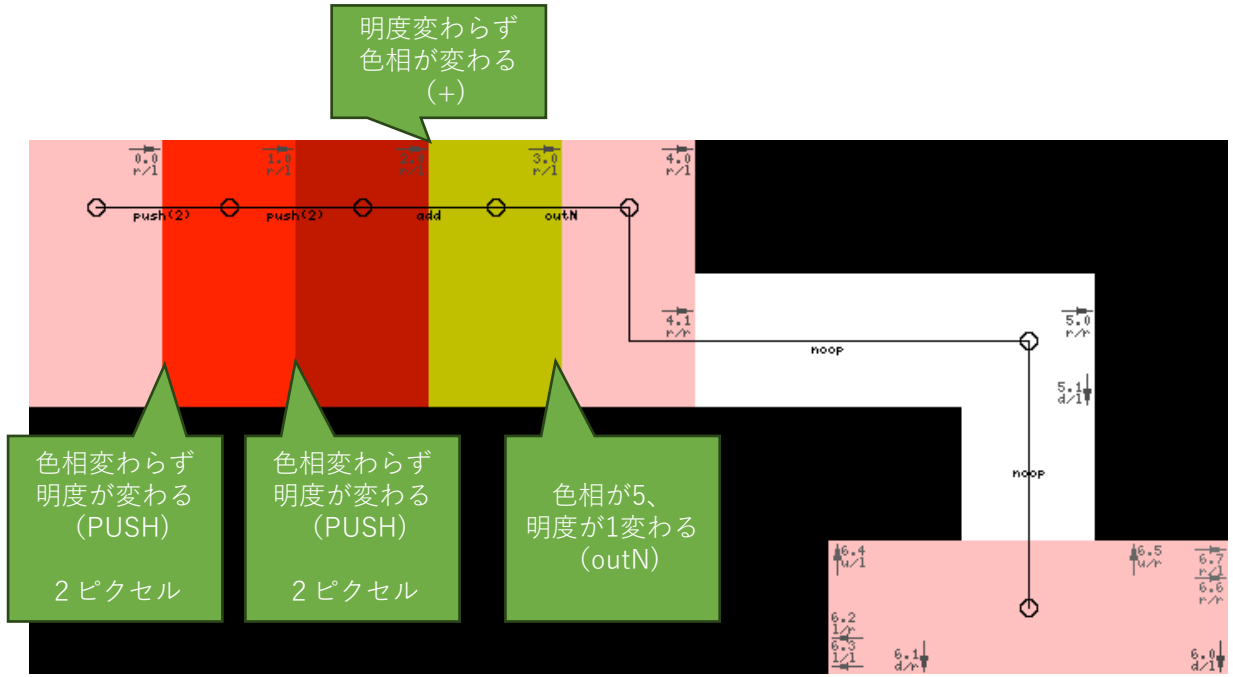
- 単色の領域が数値を表す
- プログラムは画像の中を二次元に四方向いずれかに進む（方向ポインタが存在）
- 黒は壁、白はなにもしない

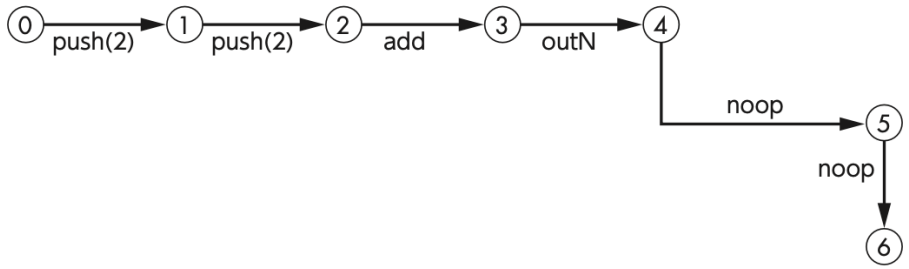
Light red (#FFC0C0)	Light yellow (#FFFFC0)	Light green (#C0FFC0)	Light cyan (#C0FFFF)	Light blue (#C0C0FF)	Light magenta (#FFC0FF)
Red (#FF0000)	Yellow (#FFFF00)	Green (#00FF00)	Cyan (#00FFFF)	Blue (#0000FF)	Magenta (#FF00FF)
Dark red (#C00000)	Dark yellow (#C0C000)	Dark green (#00C000)	Dark cyan (#00C0C0)	Dark blue (#0000C0)	Dark magenta (#C000C0)

	0	1	2
0	none	push	pop
1	+	-	×
2	÷	mod	not
3	>	pointer	switch
4	dup	roll	inN
5	inC	outN	outC

▲表9-1　色相（行）と明度（列）のブロック間の遷移によって表されるPietのコマンド

Piet





▲図9-4　add.pngのトレース

Light red (#FFC0C0)	Light yellow (#FFFFC0)	Light green (#C0FFC0)	Light cyan (#C0FFFF)	Light blue (#C0C0FF)	Light magenta (#FFC0FF)
Red (#FF0000)	Yellow (#FFFF00)	Green (#00FF00)	Cyan (#00FFFF)	Blue (#0000FF)	Magenta (#FF00FF)
Dark red (#C00000)	Dark yellow (#C0C000)	Dark green (#00C000)	Dark cyan (#00C0C0)	Dark blue (#0000C0)	Dark magenta (#C000C0)

	0	1	2
0	none	push	pop
1	+	-	×
2	÷	mod	not
3	>	pointer	switch
4	dup	roll	inN
5	inC	outN	outC

▲表9-1　色相（行）と明度（列）のブロック間の遷移によって表されるPietのコマン

Chef

Ingredients.には定数定義が入る

Methods.には実際の処理が入る

Mixing Bowlはスタック

```
Hello World Souffle.

This recipe prints the immortal words "Hello world!", in a basically brute force way. It also makes a lot of food for one person.

Ingredients.
72 g haricot beans
101 eggs
108 g lard
111 cups oil
32 zucchinis
119 ml water
114 g red salmon
100 g dijon mustard
33 potatoes

Method.
Put potatoes into the mixing bowl. Put dijon mustard into the mixing bowl. Put lard into the mixing bowl. Put red salmon into the mixing bowl. Put oil into the mixing bowl. Put water into the mixing bowl. Put zucchinis into the mixing bowl. Put oil into the mixing bowl. Put lard into the mixing bowl. Put lard into the mixing bowl. Put eggs into the mixing bowl. Put haricot beans into the mixing bowl. Liquefy contents of the mixing bowl. Pour contents of the mixing bowl into the baking dish.

Serves 1.
```

https://www.dangermouse.net/esoteric/chef.html

難解プログラミング言語を作る

あなたが難解プログラミング言語を作るべき理由

- 一般的なプログラミング言語に比べてシンプルな実装で済む
  - （複雑な）トークナイズは必要ない
- プログラム言語とは何なのかについて深く考えることができる
  - いわばパズルを作成するようなイメージ
  - 既存の言語仕様に縛られる必要は全くない
- 言語創造欲を満たすことができる

まずはBFのバリアントを作ってみよう

- 文字列を変えればバリアントの出来上がり
- インタプリタも簡単に作れる
- Esolang WikiにはBFバリアントの一覧がある
  - https://esolangs.org/wiki/Category:Brainfuck_equivalents

例: Pikalang

```
pi pi pi pi pi pi pi pi pi pi pika pipi pi pi pi pi pi
pi pi pipi pi pi pi pi pi pi pi pi pi pi pipi pi pi pi
pipi pi pichu pichu pichu pichu ka chu pipi pi pi
pikachu pipi pi pikachu pi pi pi pi pi pi pi pikachu
pikachu pi pi pi pikachu pipi pi pi pikachu pichu pichu
pi pi pi pi pi pi pi pi pi pi pi pi pi pi pikachu
pipi pikachu pi pi pi pikachu ka ka ka ka ka ka pikachu
ka ka ka ka ka ka ka pikachu pipi pi pikachu pipi
pikachu
```

Esolang Wiki を参考にしよう

Create account Log in

Main Page | Discussion | Read | View source | View history | Search Esolang

Main page
Community portal
Language list
Browse by category
Recent changes
Random page
Help

Tools

What links here
Related changes
Special pages
Printable version
Permanent link
Page information

Welcome to **Esolang**, the esoteric programming languages wiki!

Why not join us on IRC?

This wiki is dedicated to the fostering and documentation of programming languages designed to be unique, difficult to program in, or just plain weird.

For readers

You'll probably want to find out what on earth an esoteric programming language is in the first place.

Then, you might want to explore the complete list of languages, or find something more specific with the categories.

You could also visit the joke language list, which lists languages that can't even be programmed in.

Failing that, you could take a look at a random language.

You could also take a look at the list of special pages.

After getting bored, you could visit the Sandbox and have fun.

Featured language

Thue is an esoteric programming language based around the idea of a "semi-Thue system": a system which specifies strings that can be rewritten to certain other strings; a program is simply a list of search strings, and possible replacements for them. As a nondeterministic language, a program has the potential to halt if there is some way to reach an end state via applying replacements, even if rules such as "always apply the first replacement" would lead to an infinite loop. No data storage is necessary, apart from a single string that holds the entire state of the running program, although this often causes programs to run slowly due to delays in communicating information from one part of the string to another. *(**more**···)*

Previously featured: Funciton · Brainfuck · Deadfish · Emmental · *more*···

For creators

If you've just created a language, you can create an article for it by typing its name into the search box, assuming the name is not already taken, but be sure to take a look at the help guide first. Then you should add it to the language list (or the joke language list, as appropriate). **DON'T** create pages with the `Esolang:` prefix for your esolang because that prefix is for general information about the wiki.

If you haven't got that far yet, take a look at the list of ideas for inspiration.

Meta

- Learn about this wiki
- Check out the recent changes
- View the site policies
- Download an XML dump of the wiki's content
- Discuss the wiki on the community portal's talk page
- Talk with other esolang enthusiasts in the places listed in the community portal
- Go to the main page

https://esolangs.org/

難解プログラミング言語設計のポイント

- スタック型にする（変数を使わない）
- トークンは1文字、あるいは先頭から読んでいくだけで区切れるものが良い

最後に

ストレンジコード
日本語訳2/16発売

ぜひご購入ください

https://www.amazon.co.jp/dp/4798069744

ありがとうございました