Webブラウザ上で動作する「ある」プログラミング学習環境の実装に関する考察

中西渉

watayan@meigaku.ac.jp
名古屋高等学校

2023年1月7日

「ある」プログラミング学習環境=PyPEN

自己紹介

氏名	中西渉
ハンドル	わたやん
勤務先	名古屋高等学校（情報科教諭）
出身	名古屋大学理学部数学科
Web site	https://watayan.net
Twitter	@watayan
AtCoder	緑の下の方
作ったもの	何もしないプログラムとそれに毛のはえたやつ PenFlowchart，WaPEN，PyPEN など

1. はじめに

教科「情報」（赤字はプログラミングが明示的に入っている）

- 2003～ 「情報 A」「情報 B」「情報 C」（選択必履修）
- 2013～ 「社会と情報」「情報の科学」（選択必履修）
- 2022～ 「情報 I」（必履修）「情報 II」（選択）

全高校生がプログラミングを学習する

- → どういう言語？
- → どういう環境？

プログラミング教育の研究は昔から

「教育用プログラミング言語に関するワークショップ 2006」

（2006.3.29）

Java, JavaScript, C/C++, Basic, LOGO, Lisp, ワンダーボーグ, Squeak, 言霊, ひまわり/なでしこ, ドリトル, Tonyu, PEN, HSP, Rosetta, Viscuit

→ 勤務校で PEN を使用

PEN とは…

- 初学者向けプログラミング学習環境
  Programming Environment for Novices
- 大阪学院大学西田研究室，大阪私立大学大学院松浦研究室で開発
- DNCL を拡張した xDNCL
  （DNCL はセンター試験「情報関係基礎」プログラミング問題で使われた言語）
- 入力支援ボタン
  - 入力が楽
  - コードの日本語的な「ブレ」を防ぐ
  - 構造を掴む上で有効？

PEN の実行画面


*NewFile - PEN ver1.20
ファイル 編集 ヘルプ
新規
開く
保存
始めから実行
一行実行
始めに戻る
(速) ← 実行速度 → (遅)
実行終了
編集画面
1:整数 a
2:a ← 2022 - 1964
3:もし a > 17 ならば
4: | 「おいおい」 を表示する
5:を実行する
6:
コンソール画面
実行画面
履歴
おいおい

変数表示画面
型
変数名
値
整数
a
58
プログラム入力支援ボタン
もし
もし～そうでなければ
～の間、繰り返す
～になるまで実行する
増やしながら繰り返す
減らしながら繰り返す
入力
出力
改行無出力
代入
整数
実数
文字列
画面関数
ファイルI/O
文字列関数
数学関数
「」

DNCL について

- 情報関係基礎のプログラミング問題（以前は BASIC，COBOL，Pascal）
- 2002 年に DNCL と命名
- 2011 年に仕様公開
- 共通テスト に合わせて仕様変更
- 2025 年度共通テスト では新 DNCL?

旧 DNCL

```
a ← 2022 − 1964
もし a > 17 ならば
｜「おいおい」を表示する
を実行する
```

新 DNCL

```
a = 2022 − 1964
もし a > 17 ならば：
└ 表示する (”おいおい”)
```

筆者は旧新 DNCL の学習環境をいくつか開発

- PenFlowchart
- WaPEN
- PyPEN

開発体制に問題があるのでは？

- 一人開発
- この手の教育を受けていない

→ 開発経緯・過程を明らかにする

- 問題点を指摘いただきたい
- 後の人への反面教師？

2.1 PenFlowchart の開発

2006 年度から PEN を使用（それ以前は Yabasic など）

- 日本語ベースの擬似言語
- 入力支援ボタン

→ 学習者の負担は小さくなった

しかし，こんなエラーが…（日本語としては確かにその方が自然）

正しいプログラム

```
a ← 2022 - 1964
もし a > 17 ならば
｜「おいおい」を表示する
を実行する
```

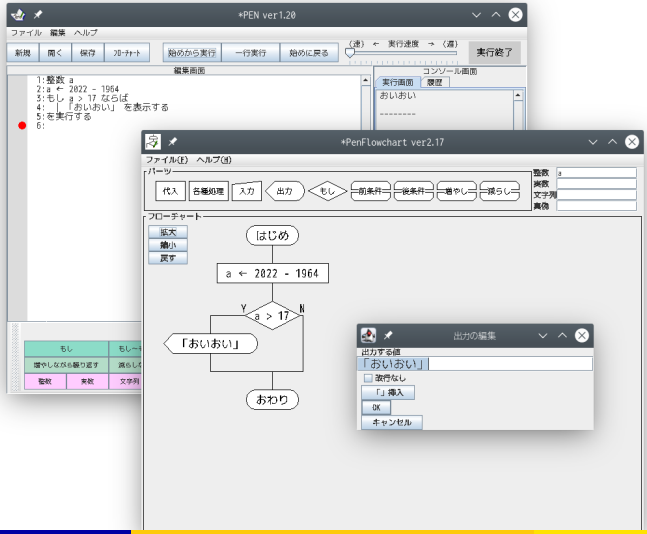
生徒が引き起こすエラー

```
a ← 2022 - 1964
もし a > 17 ならば
「おいおい」を表示する
```

このエラーをなくすには…
→フローチャートからコードを生成すればいいのでは？
→PenFlowchart の開発

- フローチャート →コード
- コード →フローチャート

PenFlowchart の実行画面



- フローチャート $\rightarrow$ コード
  - 簡単
  - 部品の連結情報→コード
- コード $\rightarrow$ フローチャート
  - 難しい
  - コードを構文解析→部品の連結情報

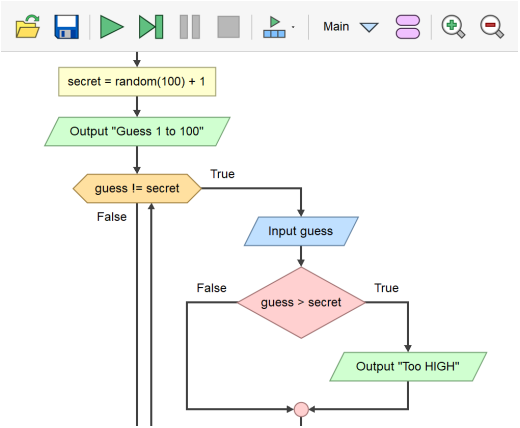
  $\rightarrow$ パーサが必要

PenFlowchart の場合

- javacc，jjtree によるパーサ
- PEN が実行用に生成する構文木→部品の連結情報に転用

たぶん Flowgorithm はそんな感じで多くの言語対応

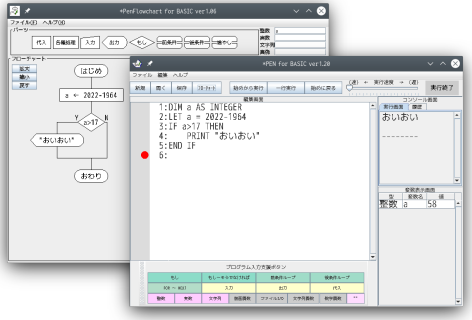
Ada 95, AppleScript, Bash, C#, C++, Fortran 2003, Java, JavaScript, Lua, MATLAB, Nim, Pascal, Perl, PHP, Powershell, Python, QBasic, Ruby, Scala, Smalltalk, Swift, Transact-SQL, TypeScript, VBA, Visual Basic.NET, (その他いくつかの擬似言語)



(画像は Flowgorithm.org より)

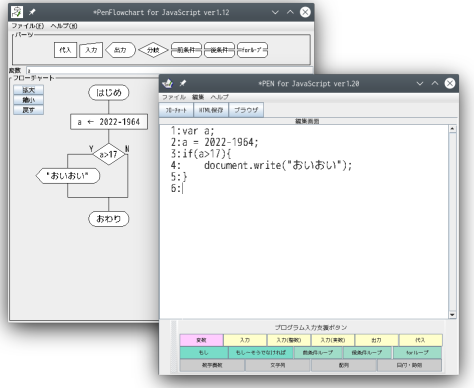
「他」言語版も作ってはみたが…

BASIC 版



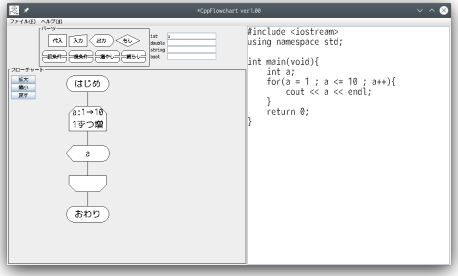
- 実行もできる

JavaScript 版



- 実行は Web ブラウザ上

「他」言語版も作ってはみたが…
C++版



- フローチャート →コードのみ
- 実行はコピペしてコンパイルして…
- ほとんど冗談のレベル

2.2 WaPENの開発

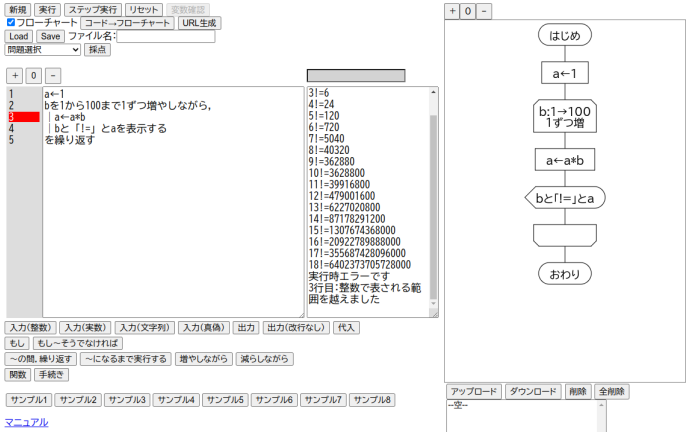
PEN，PenFlowchartはJavaアプリケーション

- インストールが必要（コピーするだけだが）
- JREがない環境も

→Webアプリケーションにすればいい

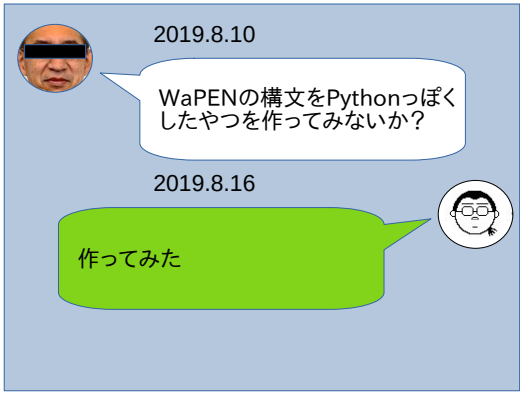
Web-aided PEN=WaPEN

WaPEN の実行画面


新規
実行
ステップ実行
リセット
変数確認
フローチャート
コード→フローチャート
URL生成
Load
Save
ファイル名:
問題選択
採点
+
0
−
1 a←1
2 bを1から100まで1ずつ増やしながら,
3 a←a*b
4 bと「!=」とaを表示する
5 を繰り返す
3!=6
4!=24
5!=120
6!=720
7!=5040
8!=40320
9!=362880
10!=3628800
11!=39916800
12!=479001600
13!=6227020800
14!=87178291200
15!=1307674368000
16!=20922789888000
17!=355687428096000
18!=6402373705728000
実行時エラーです
3行目:整数で表される範囲を越えました
入力(整数)
入力(実数)
入力(文字列)
入力(真偽)
出力
出力(改行なし)
代入
もし
もし～そうでなければ
～の間, 繰り返す
～になるまで実行する
増やしながら
減らしながら
関数
手続き
サンプル1
サンプル2
サンプル3
サンプル4
サンプル5
サンプル6
サンプル7
サンプル8
マニュアル
+
0
−
はじめ
a←1
b:1→100
1ずつ増
a←a*b
bと「!=」とa
おわり
アップロード
ダウンロード
削除
全削除
--空--

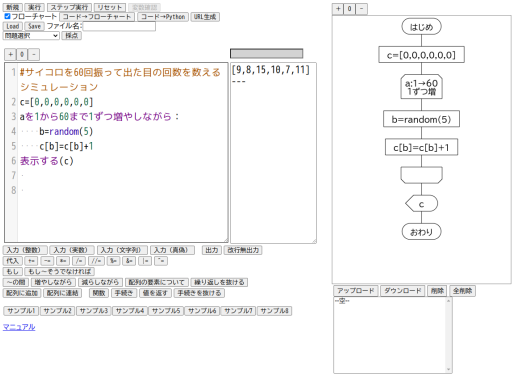
2.3 PyPENの開発

友人との会話（実際はオフライン）



- Python は最近流行
- 文科省の教材も Python 推し？

PyPEN の実行画面



PyPEN の方が先なのです…

2025 年度共通テスト 試作問題

（2020.11.10）

```
(01)Angoubun = ["p","y","e","b",…(省略)…"k","b","d","r","."]
(02)配列 Hindo のすべての要素に 0 を代入する
(03)i を 0 から 要素数(Angoubun)-1 まで 1 ずつ増やしながら:
(04)  bangou = 差分( ケ )                    ケ ⓪Angoubun[i]
(05)  もし bangou != -1 ならば:
(06)    コ = コ + 1                          コ ④Hindo[bangou]
(07)表示する(Hindo)
```

3. 関連研究

DNCL の Web ブラウザでの学習環境は多数ある

- どんくり（Bit Arrow）
- XTetra，つちのこ
- なでしこも対応

「情報 I を新 DNCL でやればいいのでは?」→ それは違う!

- ✓ 新 DNCL が Python を意識しているのは明らか
- ✗ ブラウザの中でしか動けないのはダメ
- ✗ Python に似てるのなら Python を使えばいい

4.1 WaPENの構文解析

PenFlowchart のパーサ…PEN のものをそのまま利用
WaPEN のパーサ…自分で書くか？

- ✓ 自分で作れば自由
- ✗ そんな余裕（と能力）はない

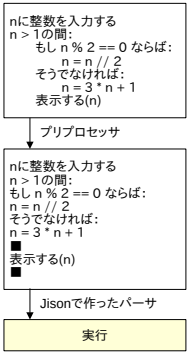
→Jison を使おう

- Bison の JavaScript 版？
- 使い方はほぼ同じ
  （Bison 使ったことないけど）

4.2 PyPENの構文解析

インデントでのブロック…どうやって Jison で書くの?
→ プリプロセッサ的なものを書いてみた

こんな感じ



ほんとにこれでいいの?

- 二度手間っぽい
- エラー表示の行数をだますためのパッチが必要

どうしたらいい?

- パーサを自分で書く?
- Jison をもっとうまく使う?

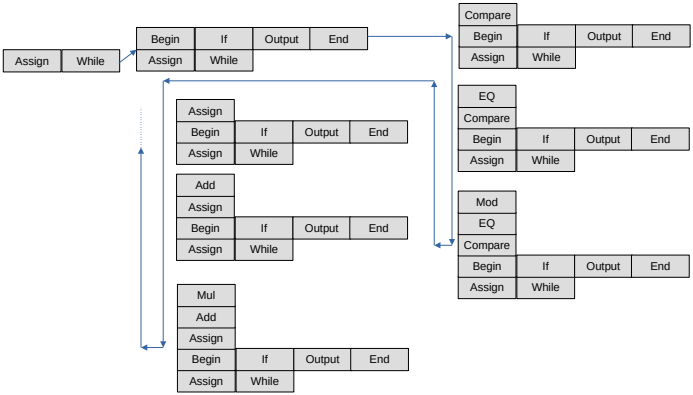
4.3 実行の仕方

構文解析はできた → どうやって実行するの？

- 順に解釈？
- JavaScript のコードに変換して実行？
- 無限ループをどうやって止める？

PyPEN での実行（概略）

1. コードの命令を配列に並べる→スタックの底
2. 行う処理をスタックに積む
3. スタックの一番上を実行（終わったら捨てる）
4. 2.〜3. を（再帰しつつ）ひたすら繰り返す




```
n = 27
n > 1の間：
    もしn % 2 == 0ならば：
        n = n // 2
    そうでなければ：
        n = 3 * n + 1
    表示する(n)
```

```
106
53
160
80
40
20
10
5
16
8
4
2
1
---
```

無限ループ対応

- setTimeout →遅い（気がする）
  - キューに 5 つ以上積むと 4ms のウェイト？
  - 0 を指定しても 1ms 待つ?

→ postMessage で回避

- どこまで小分けすべきか
  - 1 行の中まで分ける必要があるか？
  - 関数から何か呼び出していたりとかあるかも

実際のコード

```
var messageName = "zero-timeout-message";
var timeouts = [];

function setZeroTimeout(fn) {
    timeouts.push(fn);
    window.postMessage(messageName, "*");
}

function handleMessage(event) {
    if (event.source == window && event.data == messageName) {
        event.stopPropagation();
        if (timeouts.length > 0) {
            var fn = timeouts.shift();
            fn();
        }
    }
}

if(window.addEventListener) window.addEventListener("message", handleMessage, true);
else if(window.attachEvent) window.attachEvent("onmessage", handleMessage);
```

プロシージャは「関数」「手続き」で区別（返り値の有無）
→ 本当は区別をなくしたい

文法定義での不備？
「式」とか「文」とか…

4.4 型

xDNCL が DNCL でない理由の一つ：変数宣言

DNCL には変数宣言がない
→PyPEN でもいらないのでは？
→ 変数宣言の廃止（変数は型をもたない）
※「代入されると型が決まる」ではない

変数が型を持たないのでこんなこともできる

```
a = 123
a = ”123”
a = [1,2,3]
a = {”Number”:[1,2,3],”English”:[”one”,”two”,”three”]}
```

DNCL の配列←長さを決めずに使える（初期値設定可能）

- 実装しないと入試問題のコードが動かない
- 実装すると Python のコード出力が危うい

「どんくり」では実装されている

こうすると ２０２３ が表示される，みたいな

```
a のすべての要素に 2 を代入する
a[1] = 0
a[3] = 3
i を 0 から 3 まで 1 ずつ増やしながら：
    表示する (a[i])
```

これを実装したものか…

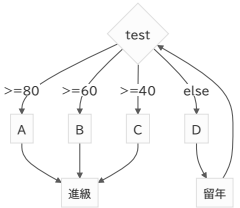
4.5 そうでなくもし

PenFlowchart，WaPEN，PyPEN にないもの：「そうでなくもし」
∵フローチャート が面倒だから

解決策

1. フローチャート を対応させる
   JIS の記法を頑張る？
   mermaid みたいなのにしてもいいかも
2. フローチャート を諦める
   既にユーザ定義関数では諦めてる

こんな感じ？



4.6 Internet Explorer への対応

IE は滅びた（はず）！
→ これからは class も Number も Promise も async/await も使い放題

懺悔

私は IE で使えないことを言い訳に
Promise や async/await の勉強を
サボっていました

5. おわりに

自信を持って開発を進められるような

- 基礎知識
- 背景にできる何か
- 技量

が欲しい