「情報関係基礎」プログラミング問題で省略されている変数初期化部分に関係するWaPEN，PyPENの実装

中西渉

watayan@meigaku.ac.jp
名古屋高等学校

2021-12-05

1. はじめに

「なでしこ」が DNCL 対応

2021 年 10 月 13 日

プレスリリース

日本語プログラミング言語なでしこ

なでしこ 3 大学入学共通テスト手順記述標準言語

DNCL 対応のお知らせ

日本語プログラミング言語なでしこは、なでしこ 3 が大学入学共通テスト手順記述標準言語(DNCL) に対応したことについてプレスリリースをいたしました。

DNCL とは大学入学共通テスト（大学入試センター試験）で利用される日本語のプログラミング疑似言語です。

（2021/10/13 プレスリリースより）

対応表で書き換えればいいだけ

DNCL　制御文2

例
もし x < 3 ならば x ← x + 1 を実行する

例
もし x < 3 ならば
|x ← x + 1
を実行し，そうでなければ
| x ← x − 1
を実行する

なでしこ3　記載例

例
!インデント構文
もし、x<3ならば、x=x+1
→　プログラム実行

例
!インデント構文
もし、x<3ならば、
　x=x+1
違えば
　x=x-1
→　プログラム実行

（なでしこ 3 DNCL 対応表より）

そのままでは動かない？
→でもセンター試験等の問題自体，そのままでは動かない
∵初期化が省略されている

本稿の主旨

- 省略された初期化部分を調査
- そのために WaPEN，PyPEN に施した追加実装

2. DNCL の概要

DNCL

- センター試験等「情報関係基礎」プログラミング問題で使用
- 日本語を基本とした擬似言語
- 言語仕様は Web で公開（共通テストに合わせて更新）
  - 変数は宣言不要
  - 配列（添字は 1 から，たまに 0 から）
  - 臨時で関数が用意されることもある

if 相当

もし～ならば
　　｜《処理》
そうでなければ
　　｜《処理》
を実行する

while–wend 相当

～の間，
　　｜《処理》
を繰り返す

do–until 相当

繰り返し，
　　｜《処理》
を，～になるまで実行する

for–next 相当

《変数》を《値》から《値》まで《値》ずつ増やしながら，
　　｜《処理》
を繰り返す

2025 年度からの共通テスト
←試作問題，サンプル問題が公表
「新」DNCL（Python 風?）...詳細は不明

if 相当

もし～ならば：
　　《処理》
そうでなければ：
　　《処理》

while–wend 相当

～の間：
　　《処理》

do–until 相当

for–next 相当

《変数》を《値》から《値》まで《値》ずつ増やしながら：
　　《処理》

3.1 DNCL 学習環境

Java アプリケーション

PEN 大阪学院大学，大阪市立大学で開発

PenFlowchart PEN にフローチャートを付加

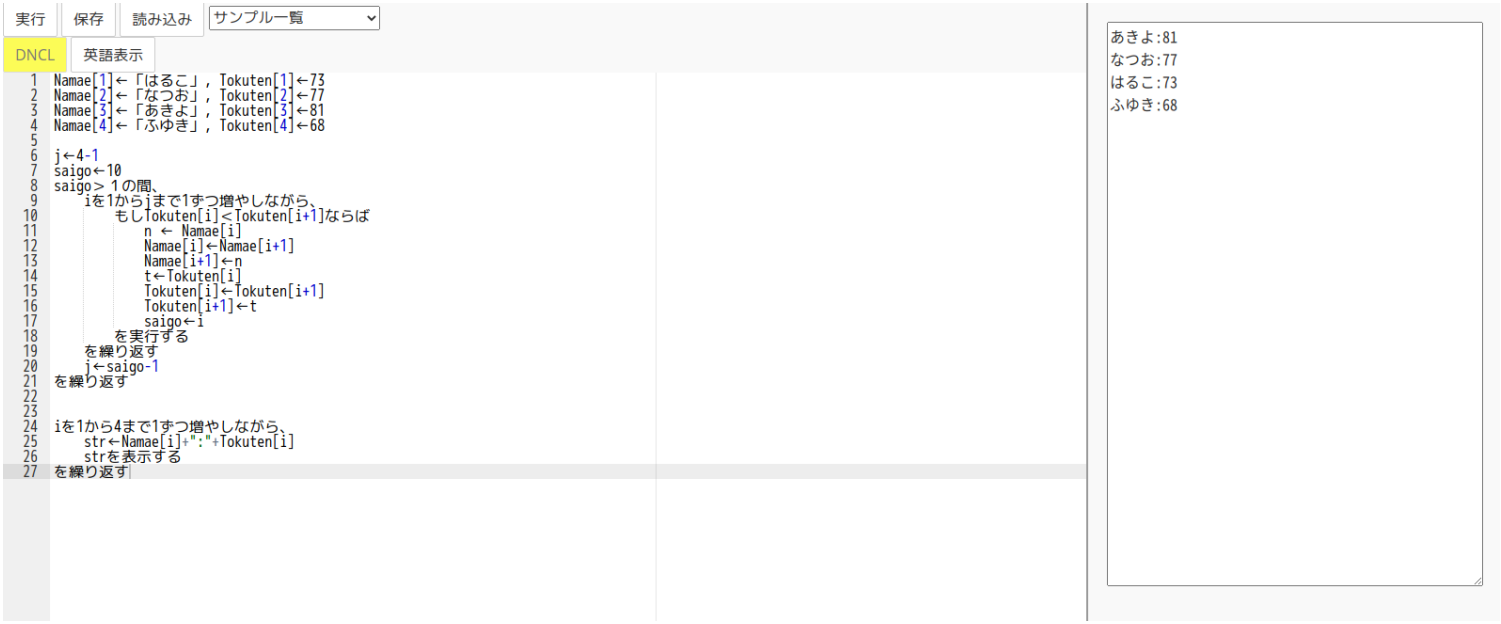
Java アプリケーションゆえに...

- JRE 必要
- インストール必要

→ Web アプリケーションが開発される

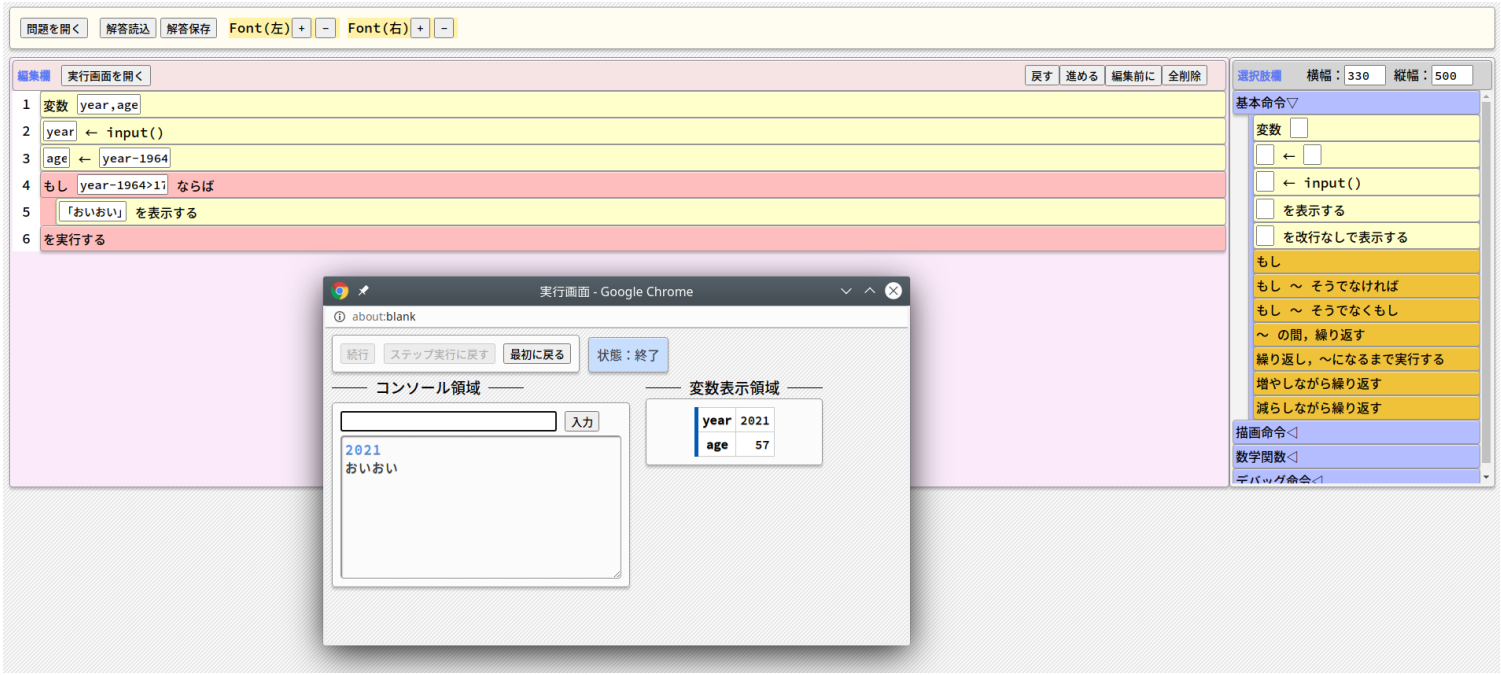
どんくり

- C に似た構文の言語との相互変換
- 変数一覧機能
- プロファイル機能



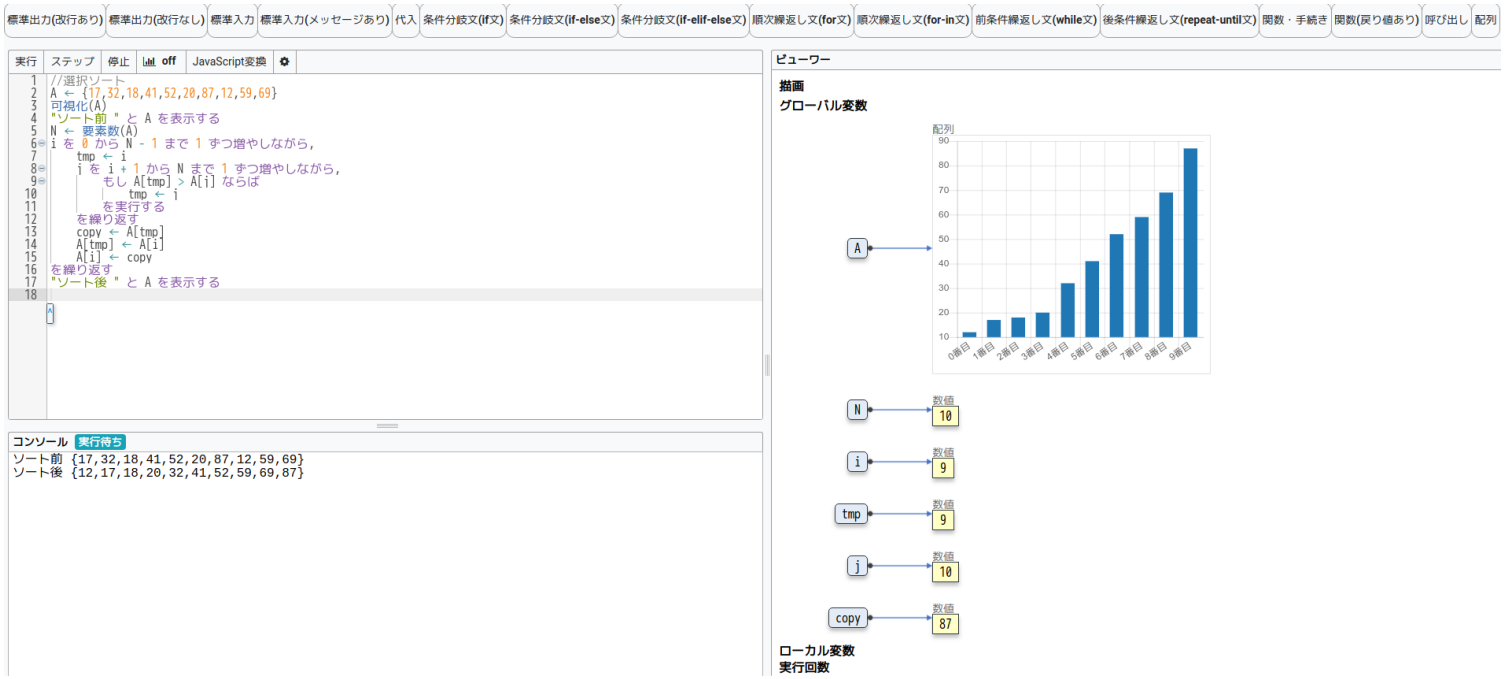
wPEN

- 短冊プログラミング
- 作問・解答機能



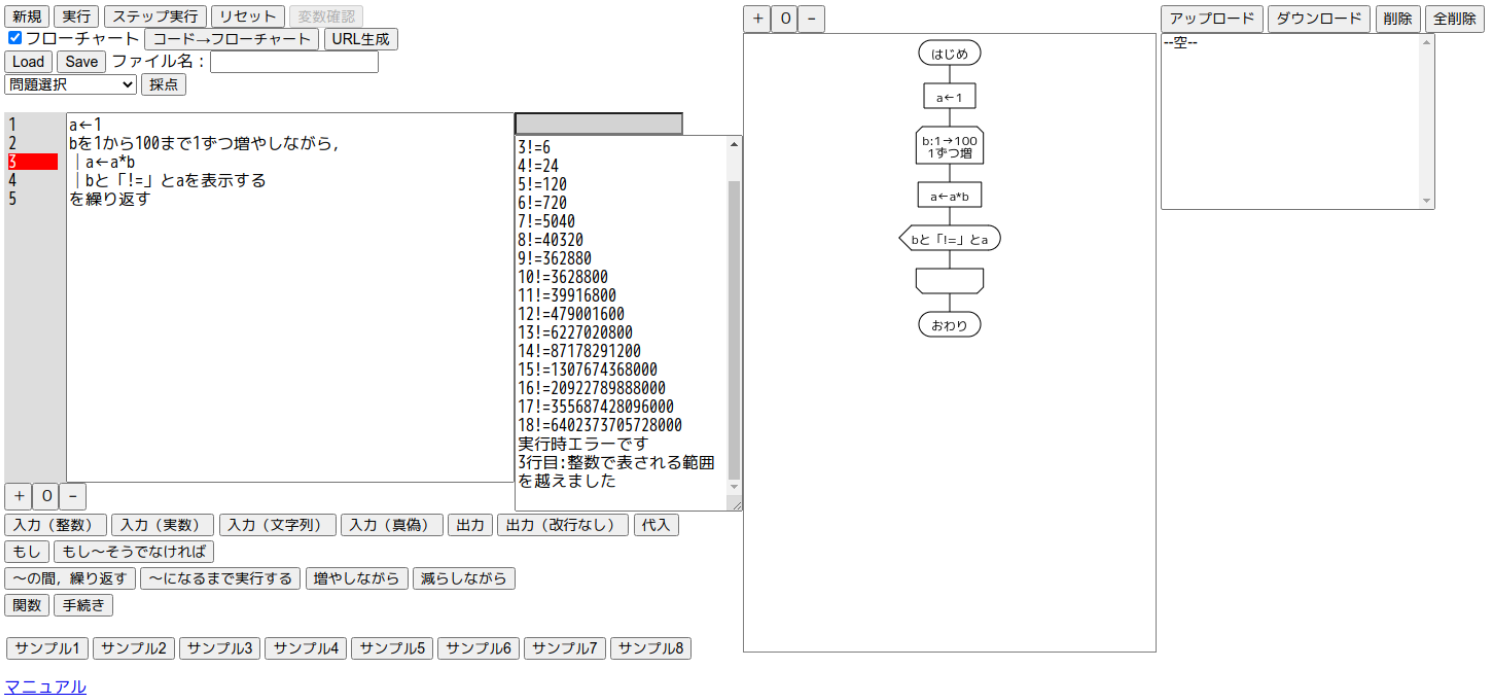
XTetra

- ビジュアル・タイピング両対応
- プロファイル機能
- JavaScript への変換



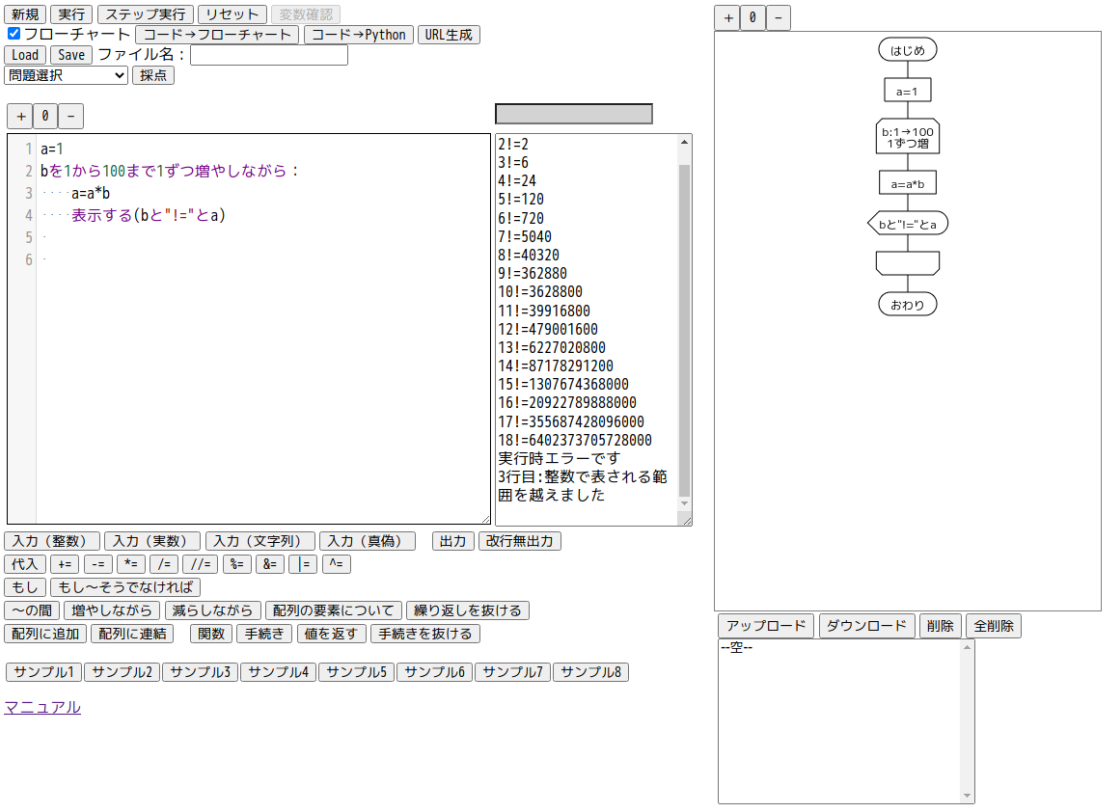
WaPEN

- コード ↔ フローチャート
- サーバ不要（ローカルで実行可能）



新 DNCL に近づけた版も...

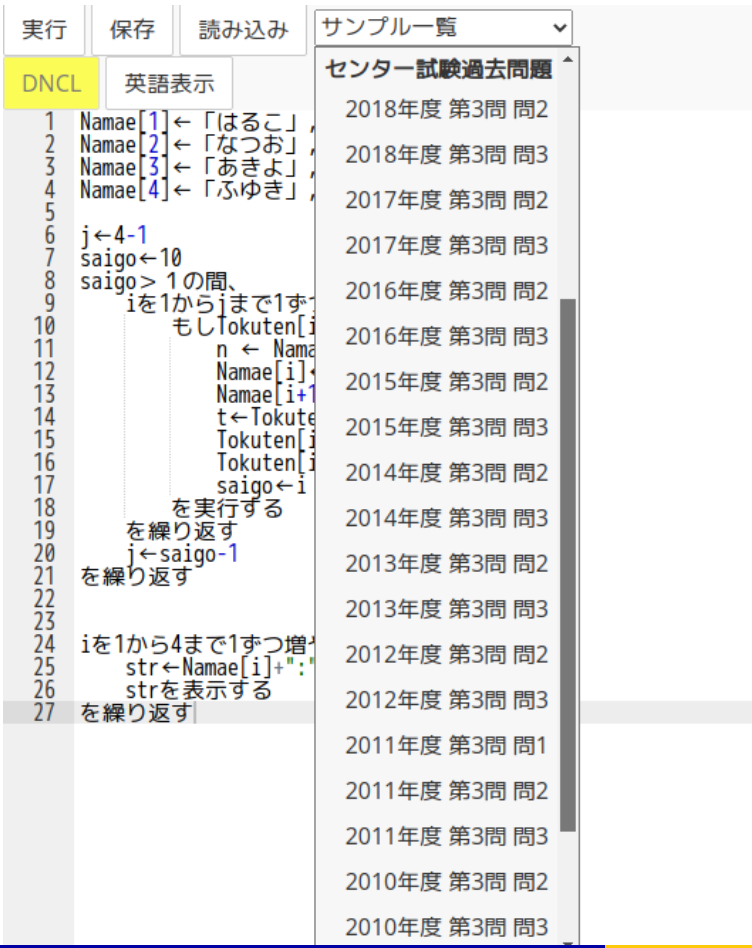
- どんくり → 新 DNCL 対応版
- wPEN→ 新 DNCL 対応版
- XTetra→ つちのこ
- WaPEN→PyPEN



3.2　センター試験等の問題で省略された部分

配列の初期化部分は問題に掲載されないことが多い（表１参照）

「どんくり」は 2008〜2018 年度問題をサンプルに採用



4.1 PyPEN から WaPEN への移植

最近 PyPEN にかかりきり
→ WaPEN の開発が停滞

PyPEN を WaPEN に移植（構文解析などを除く）
→その結果

- 両方での新機能実装が容易に
- 互換性の放棄
  - 変数宣言の廃止→本来の DNCL には近づいた

4.2　長さの決まっている配列

PEN...配列への代入が１つずつ
PenFlowchart...まとめての代入をサポート

PEN

te[0] ← 「グー」
te[1] ← 「チョキ」
te[2] ← 「パー」

PenFlowchart

te ←{「グー」,「チョキ」,「パー」}

WaPEN での書き方 1

te ← [「グー」,「チョキ」,「パー」]

WaPEN での書き方 2

te ← []
te に「グー」を追加する
te に「チョキ」を追加する
te に「パー」を追加する

さらに...

a ← [0,0,0,0,0,0,0,0,0,0]
b ← 10 個の 0
c ← [0]*10

細かい違い：

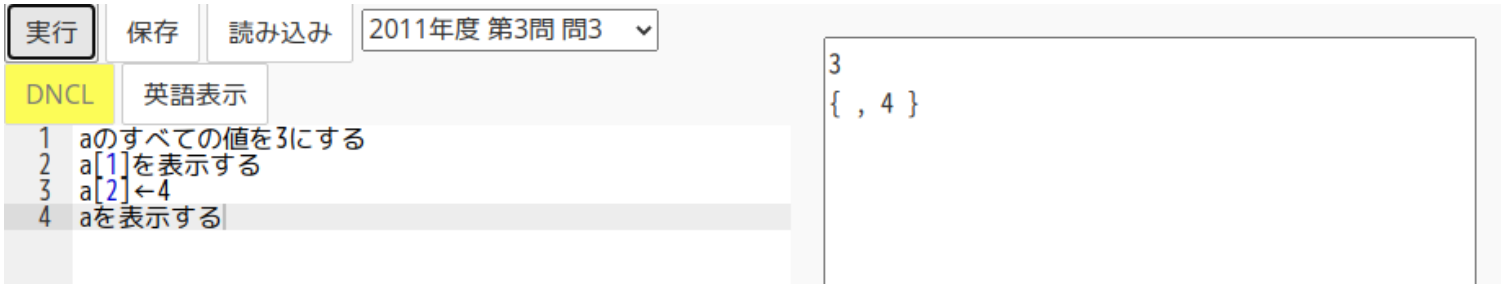
```
1  a←10個のrandom(5)
2  b←[random(5)]*10
3  変数を確認する
```

```
*** 変数確認 ***
a:[5,1,5,1,3,5,3,4,2,0]
b:[5,5,5,5,5,5,5,5,5,5]
---
```

4.3 長さの決まっていない配列

未実装
PyPEN…Python のコードに「直訳」したいので実装しない
WaPEN…実装したい

「どんくり」が実装の参考になりそう



4.4　2 次元配列

2018 年度問題の初期化部分

Tetra 版

複数行での初期化が可能

```
/*2018年 情報関係基礎 第3問 問2*/
tate←9,yoko←11
Masu←{{1,1,1,1,1,1,1,1,1},
{1,0,0,0,1,0,0,0,1},
{1,1,1,0,0,0,1,0,1},
{9,0,0,0,1,0,1,0,1},
{1,1,1,0,1,0,1,0,1},
{1,0,1,1,1,0,1,1,1},
{1,0,0,0,1,0,1,0,1},
{1,1,1,0,0,0,1,0,9},
{1,0,0,0,1,1,1,0,1},
{1,0,1,0,0,0,0,0,1},
{1,1,1,1,1,1,1,1,1}}
```

（当時の）WaPEN 版

複数行での初期化が不可能

```
1  整数 tate,yoko,Masu[11,9],x,y,nutta,s
2  tate ← 9
3  yoko ← 11
4  Masu[1] ← [0,1,1,1,1,1,1,1,1,1]
5  Masu[2] ← [0,1,0,0,0,1,0,0,0,1]
6  Masu[3] ← [0,1,1,1,0,0,0,1,0,1]
7  Masu[4] ← [0,9,0,0,0,1,0,1,0,1]
8  Masu[5] ← [0,1,1,1,0,1,0,1,0,1]
9  Masu[6] ← [0,1,0,1,1,1,0,1,1,1]
10 Masu[7] ← [0,1,0,0,0,1,0,1,0,1]
11 Masu[8] ← [0,1,1,1,0,0,0,1,0,9]
12 Masu[9] ← [0,1,0,0,0,1,1,1,0,1]
13 Masu[10]← [0,1,0,1,0,0,0,0,0,1]
14 Masu[11]← [0,1,1,1,1,1,1,1,1,1]
```

コンマ後などの改行を許すよう拡張

当時の WaPEN 版

```
整数 tate,yoko,Masu[11,9],x,y,nutta,s
tate ← 9
yoko ← 11
Masu[1] ← [0,1,1,1,1,1,1,1,1,1]
Masu[2] ← [0,1,0,0,0,1,0,0,0,1]
Masu[3] ← [0,1,1,1,0,0,0,1,0,1]
Masu[4] ← [0,9,0,0,0,1,0,1,0,1]
Masu[5] ← [0,1,1,1,0,1,0,1,0,1]
Masu[6] ← [0,1,0,1,1,1,0,1,1,1]
Masu[7] ← [0,1,0,0,0,1,0,1,0,1]
Masu[8] ← [0,1,1,1,0,0,0,1,0,9]
Masu[9] ← [0,1,0,0,0,1,1,1,0,1]
Masu[10]← [0,1,0,1,0,0,0,0,0,1]
Masu[11]← [0,1,1,1,1,1,1,1,1,1]
```

今の WaPEN 版

```
tate←9
yoko←11
Masu←[
[0,0,0,0,0,0,0,0,0,0],
[0,1,1,1,1,1,1,1,1,1],
[0,1,0,0,0,1,0,0,0,1],
[0,1,1,1,0,0,0,1,0,1],
[0,9,0,0,0,1,0,1,0,1],
[0,1,1,1,0,1,0,1,0,1],
[0,1,0,1,1,1,0,1,1,1],
[0,1,0,0,0,1,0,1,0,1],
[0,1,1,1,0,0,0,1,0,9],
[0,1,0,0,0,1,1,1,0,1],
[0,1,0,1,0,0,0,0,0,1],
[0,1,1,1,1,1,1,1,1,1]]
```

4.5 Webストレージを用いたFile I/Oもどき

PEN，PenFlowchart...File I/O アリ
→ Web アプリケーションへの移植は？

- ローカルファイルは扱えない
- WaPEN や PyPEN はサーバを絡ませたくない

試作問題第 5 問

```
(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 )
(05)  もし bangou != −1 ならば:
(06)    コ = コ + 1
(07)表示する(Hindo)
```

ケ ⓪Angoubun[i]

コ ④Hindo[bangou]

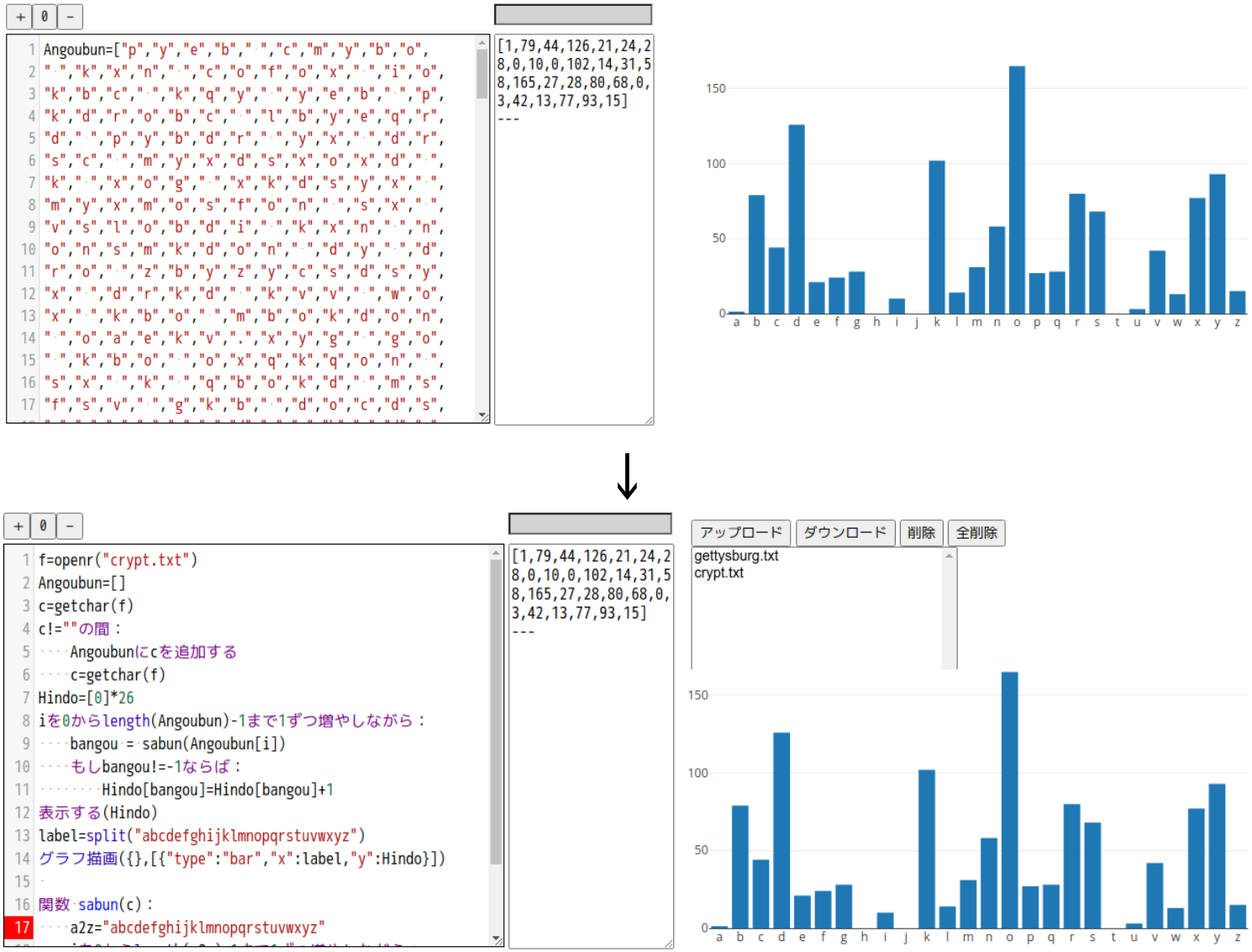
（省略）は千数百文字

pyeb cmybo kxn cofox iokbc kqy yeb pkdrobc lbyeqrd pybdr yx drsc myxdsxoxd, k xog xkdsyx, myxmosfon sx vslobdi, kxn nonsmkdon dy dro zbyzycsdsyx drkd kvv wox kbo mbokdon oaekv.
xyg go kbo oxqkqon sx k qbokd msfsv gkb, docdsxq grodrob drkd xkdsyx, yb kxi xkdsyx cy myxmosfon kxn cy nonsmkdon, mkx vyxq oxnebo. go kbo wod yx k qbokd lkddvo-psovn yp drkd gkb. go rkfo mywo dy nonsmkdo k zybdsyx yp drkd psovn, k c k psxkv bocdsxq zvkmo pyb dryco gry robo qkfo drosb vsfoc drkd drkd xkdsyx wsqrd vsfo. sd sc kvdyqodrob psddsxq kxn zbyzob drkd go cryevn ny drsc.
led, sx k vkbqob coxco, go mkx xyd nonsmkdo -- go mkx xyd myxcombkdo -- go mk x xyd rkvvyg -- drsc qbyexn. dro lbkfo wox, vsfsxq kxn nokn, gry cdbeqqvon ro bo, rkfo myxcombkdon sd, pkb klyfo yeb zyyb zygob dy knn yb nodbkmd. dro gybv n gsvv vsddvo xydo, xyb vyxq bowowlob grkd go cki robo, led sd mkx xofob pybq od grkd droi nsn robo. sd sc pyb ec dro vsfsxq, bkdrob, dy lo nonsmkdon robo dy dro expsxscron gybu grsmr droi gry pyeqrd robo rkfo drec pkb cy xylvi knfk xmon. sd sc bkdrob pyb ec dy lo robo nonsmkdon dy dro qbokd dkcu bowksxsxq lo pybo ec -- drkd pbyw droco ryxybon nokn go dkuo sxmbokcon nofydsyx dy drkd mk eco pyb grsmr droi qkfo dro vkcd pevv wokcebo yp nofydsyx -- drkd go robo rsq rvi bocyvfo drkd droco nokn crkvv xyd rkfo nson sx fksx -- drkd drsc xkdsyx, exnob qyn, crkvv rkfo k xog lsbdr yp pboonyw -- kxn drkd qyfobxwoxd yp dro zo vzvo, li dro zovzvo, pyb dro zovzvo, crkvv xyd zobscr pbyw dro okbdr.

省略しないで書くと…

～（略）～

外部ファイルから読み込ませたい
→ Web ストレージを用いた File I/O（もどき）を実装


+ 0 -
Angoubun=["p","y","e","b","","c","m","y","b","o",
"","k","x","n","","c","o","f","o","x","","i","o",
"k","b","c","","k","q","y","","y","e","b","","p",
"k","d","r","o","b","c","","l","b","y","e","q","r",
"d","","p","y","b","d","r","","y","x","","d","r",
"s","c","","m","y","x","d","s","x","o","x","d","",
"k","","x","o","g","","x","k","d","s","y","x","",
"m","y","x","m","o","s","f","o","n","","s","x","",
"v","s","l","o","b","d","i","","k","x","n","","n",
"o","n","s","m","k","d","o","n","","d","y","","d",
"r","o","","z","b","y","z","y","c","s","d","s","y",
"x","","d","r","k","d","","k","v","v","","w","o",
"x","","k","b","o","","m","b","o","k","d","o","n",
"","o","a","e","k","v",".","x","y","g","","g","o",
"","k","b","o","","o","x","q","k","q","o","n","",
"s","x","","k","","q","b","o","k","d","","m","s",
"f","s","v","","g","k","b","","d","o","c","d","s",
[1,79,44,126,21,24,28,0,10,0,102,14,31,58,165,27,28,80,68,0,3,42,13,77,93,15]

150
100
50
0
a b c d e f g h i j k l m n o p q r s t u v w x y z
+ 0 -
1 f=openr("crypt.txt")
2 Angoubun=[]
3 c=getchar(f)
4 c!="" の間：
5 Angoubunにcを追加する
6 c=getchar(f)
7 Hindo=[0]*26
8 iを0からlength(Angoubun)-1まで1ずつ増やしながら：
9 bangou = sabun(Angoubun[i])
10 もしbangou!=-1ならば：
11 Hindo[bangou]=Hindo[bangou]+1
12 表示する(Hindo)
13 label=split("abcdefghijklmnopqrstuvwxyz")
14 グラフ描画({},[{"type":"bar","x":label,"y":Hindo}])
15
16 関数 sabun(c)：
17 a2z="abcdefghijklmnopqrstuvwxyz"
[1,79,44,126,21,24,28,0,10,0,102,14,31,58,165,27,28,80,68,0,3,42,13,77,93,15]

アップロード
ダウンロード
削除
全削除
gettysburg.txt
crypt.txt
150
100
50
0
a b c d e f g h i j k l m n o p q r s t u v w x y z

5. おわりに

WaPEN，PyPEN に追加実装
→省略されてきた初期化部分への対応
「どんくり」や Tetra に追いつこうとしただけではあるが

高野ら [23]

情報系の標準的な常識からすると，わざわざ曖昧さを残しておくのは不自然である。ここからは著者らの推測の範囲に過ぎないが，ひとつ考えられる理由として，DNCL の設計者は，DNCL を独立したプログラミング言語として学習することを嫌がっていると考えられる。つまり，プログラミング教育は，世の中に実在するプログラミング言語で実習することを望み，（略）

水野 [24]

少し懸念していることは，DNCL だけを使ってプログラミングの授業が行われることがないように，ということです。やはり授業では，教科書で使われている実用的なプログラム言語を使っていただきたい。

DNCL だからダメ？
ダメなのは DNCL にとどまること

- 導入の一方法として
- わからなくなったときに振り返る先として

ということなら意味はあるのではないか

Python で書き方に迷ったときのアンチョコになればいい

```
Angoubun=[]
f=openr('crypt.txt')
c=getchar(f)
c!=''の間：
    Angoubunにcを追加する
    c=getchar(f)
close(f)
Hirabun=''
hukugousuu=26-10
iを0からlength(Angoubun)-1まで1ずつ増やしながら：
    bangou=sabun(Angoubun[i])
    もしbangou!=-1ならば：
        Hirabun+=moji((bangou+hukugousuu)%26)
    そうでなければ：
        Hirabun+=Angoubun[i]
表示する(Hirabun)
```

```
def sabun(c):
    a2z = 'abcdefghijklmnopqrstuvwxyz'
    for i in range(0,25+1,1):
        if c==a2z[i]:
            return i
    return -1
def moji(n):
    a2z = 'abcdefghijklmnopqrstuvwxyz'
    if 0<=n<=25:
        return a2z[n]
    else:
        return 'アルファベットでない'
Angoubun = []
f = open('crypt.txt','r')
c = f.read(1)
while c!='':
    Angoubun.append(c)
    c = f.read(1)
f.close()
Hirabun = ''
hukugousuu = 26 - 10
```