OUKA
The University of Osaka
Institutional Knowledge Archive

Title	初学者のためのプログラミング学習環境PEN
Author(s)	西田, 知博; 松浦, 敏雄
Citation	サイバーメディア・フォーラム. 2006, 7, p. 11-16
Version Type	VoR
URL	https://doi.org/10.18910/70222
rights	
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

https://ir.library.osaka-u.ac.jp/

The University of Osaka

初学者のためのプログラミング学習環境 PEN

西田知博 (大阪学院大学 情報学部)
松浦敏雄 (大阪市立大学大学院創造都市研究科
兼　学術情報総合センター)

1　はじめに

昨年度のサイバーメディアフォーラムの中で，筆者らは、『高等学校における教科「情報」関連の現状と今後の展望』[1]、および、『情報処理教育の 2006 年問題への対応』[2] と題して、記事を書かせて頂いた。これらの記事でも指摘したが、高等学校の教科「情報」の指導要領では、情報の科学的理解を求めているものの、教育の現場ではコンピュータの使い方を教える「操作教育」に偏っている傾向が強いと言われている。また、大学の教育現場においても同様の傾向が見られることも良く知られている。しかし、「情報処理と情報システムの原理に対する理解の欠如」が原因と思われる事件が多発しているとの報告もある [3] など、情報教育のあり方に対する疑問も生じてきている。

こうした背景から、情報処理学会 情報処理教育委員会は、「日本の情報教育・情報処理教育に関する提言 2005」[4] を出し、「情報処理の理解」の重要性を指摘している。 この提言では、「情報処理の理解」を『コンピュータの本質は、「手順的な自動処理」であることを、体感的かつ具体的に理解していること』とし、「手順的な自動処理」の構築を何らかの形で体験することが必要であることを強調している。「手順的な自動処理」の学習は、必ずしも「プログラミング」に限定したものではないが、高校・大学レベルでは、「プログラミング」による実習が適していると思われる。

現在、プログラミング実習は、理系の一部の学部/学科を除いて、ほとんど行われていない。 一般情報処理教育として、プログラミングがあまり取り上げられない理由としては、「プログラミングは難しいものであり、その習得には多くの時間が必要である」と思われていることが挙げられる。 しかし、プログラミング言語の習得を目的とするのではなく、プログラミングの体験を通して、コンピュータの本質を理解することを目的とするならば、適切なサポートツールを用意することで、比較的短い時間でこの目標を達成できると考えた。本稿で紹介する PEN(Programming Environment for Novices) は、このようなプログラミング入門教育をサポートするためのツールである [5][6][7]。 PEN は、分かりやすいプログラミング言語を備え、構文エラーなどを生じにくくするための入力支援機能を備えており、比較的容易にプログラミングを学ぶことが出来るようにしている。

2　PEN の概要

図 1 は PEN の実行時のスナップショットである。PEN は Java アプリケーションで、プログラムの入力/編集を行うためのエディタ機能、プログラムの実行・一時停止・一行実行などの実行制御機能、実行結果とその履歴を表示するコンソール機能、実行中の各変数の値を表示する機能等をもつ。

2.1　プログラミング言語

PEN で用いるプログラミング言語は、読んで分かりやすいものであることが第一の条件であると考え、日本語表記を用いることとした。日本語プログラミング言語については、さまざまなものが開発されているが、我々は、大学入試センターの「情報関係基礎」で用いられている手順記述言語 **DNCL**[8] に着目した。これは日本語をベースとした記述言語であり、受験生に対する事前の特別な説明なしで入試で用いられているということからも、分かりやすさの点では特に優れていると思われるからである。

一方で、DNCL は試験用言語であるために、実装に必要な命令が欠けていたり、機能が不十分な部分がある。そこで、我々はこれに拡張を加えて整理した言語を作成し、**xDNCL** と呼んでいる。拡張した点は、試験用言語であるために欠けている実装に必要な命令の補強、変数や型を意識させるための変数宣言の追加、数学関数やファイル I/O、グラフィックスなどの組み込み関数の追加、続き・関数の定義機能である。なお、xDNCL では、変数型としては、

図 1: PEN の実行時の表示例

整数、実数、文字列の 3 つを用意している。また、日本語の曖昧さを考慮し、たとえば「もし〜ならば」という記述が「もし〜なら」と、っても許すなど、一部の予約語については記述のゆらぎを認めるようにしている。図 2 に xDNCL によるプログラムの記述例を示す。いくつかの拡張は行っているものの、記述法は基本的にほとんど同じであり、プログラムを読むのは容易である。

2.1.1 グラフィックス関数

グラフィックスの描画は、専用のウィンドウを作成し、その上で行う。描画関数は、引数の数が多くなりすぎないように、図形の描画とそれぞれの描画属性の指定を分離した。描画可能な図形は、点、線分、矩形、円、楕円、円弧で、点と線分を除き、内部を塗りつぶすか否かで異なる関数を用意している。また、文字を出力する関数も用意している。描画属性としては、線の太さ・種類、線への矢印の付加、線や塗りつぶしの色、フォントの大きさ・種類などがある。それぞれの属性はデフォルトの値を持っており、属性値を設定した場合は、同じ属性に対して新たに値が設定されるまでそれが有効となる。このような、属性設定と描画の分離は、ドローツールではよく見られる形式であるので、プログラミングの初学者であっても、それらのツールでの作図経験があれば理解しやすいと思われる。図 3 は、ランダムに定めた座標を元に、少しずつ始点・終点ずらした線分を重ねることで模様を描くプログラムである。

```
1:整数 i, ans, inkey, seikai
2:ans ← random(99) + 1
3:i ← 1, seikai ← 0
4:seikai = 0 かつ i ≦ 5 の間、
5:  | inkey ← input()
6:  | もし ans > inkey ならば
7:  |   | 「入力値より大きいです」を印刷する
8:  | を実行し、そうでなくもし ans < inkey ならば
9:  |   | 「入力値より小さいです」を印刷する
10: | を実行し、そうでなければ
11: |   | 「正解です」を印刷する
12: |   | seikai ← 1
13: | を実行する
14: | i ← i+1
15:を繰り返す
16:もし seikai = 0 ならば
17:  | 「正解は」と ans と「です」を印刷する
18:を実行する
```

図 2: xDNCL の記述例（数当てゲーム）

2.1.2 ファイル I/O 関数

ファイル I/O には、UNIX の open、read/write などの関数を用いる低レベル I/O と、fopen、getchar、putchar などの関数を用いる高レベル I/O がある。ファイル I/O の仕組みなどの学習には前者が適しているが、関数の使い方はやや難しい。後者は、抽象度が高く、バッファを意識することなく、文字単位、行単位で入出力が可能である。 PEN では、初学者向けということから、後者の関数群を用意している。ファイル I/O 関数を用いたプログラム例として、ファイルコピープログラムの例を図 4 に示す。

2.2 プログラム入力支援機能

xDNCL で記述されたプログラムは読みやすいが、日本語で記述されているので、プログラム作成時は

```
 1: 整数 i, j, x1, y1, x2, y2
 2: gOpenWindow(300, 300)
 3: i を 1 から 10 まで 1 ずつ増やしながら、
 4:   | gSetLineColor(random(255),
      |                 random(255), random(255))
 5:   | x1 ← random(300)
 6:   | y1 ← random(300)
 7:   | x2 ← random(300)
 8:   | y2 ← random(300)
 9:   | j を 1 から 30 まで 1 ずつ増やしながら、
10:   |   | gDrawLine(x1+3*j, y1+2*j, x2+2*j, y2-3*j)
11:   | を繰り返す
12: を繰り返す
```

図 3: 線分を重ねて模様を描くプログラム

文法通りに間違わず入力することは必ずしも容易ではない。また、キーボードに慣れていない初学者にとっては、かな漢字変換の操作も煩雑である。キー入力の操作を減らし、補助するための機能として、図 1 の下部のような**プログラム入力支援ボタン**を用意した。これらを選択することにより、図 5 左に示すような制御構造のスケルトンをエディタに挿入する。<< 条件式 >> などの << と >> で囲まれた部分はマウスクリックやその部分にカーソルを移動することにより、まとめて全体を選択でき、実際の条件式等に簡単に書き換えることができる（図 5 右）。

DNCL 等では、インデントを縦棒記号 (|) で表している。PEN では、インデントの縦棒記号を自動で挿入するようにした。インデントを自動的に付加することにより、プログラムの記述を助けるだけではなく、プログラムの構造を明確に意識する手助けにもなる。

2.3 プログラムの実行状態表示機能

プログラムの実行は図 1 上部中央にある実行制御ボタン群の**実行ボタン**により行う。また、プログラム実行の流れを理解できるよう、１行ずつ実行できるステップ実行や、実行速度を調節し実行できるスロー実行の機能を持つ。実行時には、どの行が実行されているかを図 1 左にある「●」印の**実行箇所マーカー**で常に明示している。**変数表示画面**では、実行

```
 1: 文字列 InputFile, OutputFile, str
 2: 整数 fdin, fdout, stop
 3: 「コピー元ファイルは?:」を改行なしで印刷する
 4: InputFile ← input()
 5: 「コピー先ファイルは?:」を改行なしで印刷する
 6: OutputFile ← input()
 7: fdin ← openr(InputFile)
 8: fdout ← openw(OutputFile)
 9: (stop=0) の間、
10:   | str ← getstr(fdin, 1)
11:   | もし str=EOF ならば
12:   |   | stop ← 1
13:   | を実行し、そうでなければ
14:   |   | putstr(fdout, str)
15:   | を実行する
16: を繰り返す
17: close(fdin)
18: close(fdout)
```

図 4: ファイルコピープログラム

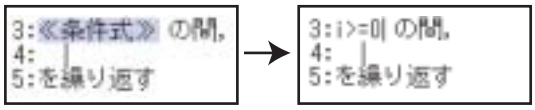

図 5: 入力支援機能

中のプログラムで用いているすべての変数の値を常に表示している。これらの情報を提示することで、プログラムがどのように実行されているかなどの状況を把握しやすくしている。また、**実行状態表示ラベル**でプログラムの実行時の状態を明示している。プログラムの状態としては、「実行待ち」「実行中」「一時停止中」「入力待ち」「実行終了」の 5 つの状態がある。これにより、プログラムが途中で停止しているのか動いているのか、入力を求められて停止しているかなどが、一目でわかる。

2.4 入力支援ボタンのカスタマイズ

プログラム入力支援ボタンは外部ファイルによって定義されており、ボタンの表示名やサイズ、色、押した時に入力される文字列をカスタマイズできるようにしている [5]。カスタマイズすることにより、初期段階ではボタン数を絞り込むなど、教員が授業の進行状況に合わせて、必要なボタンのみを表示させておくことも可能である（図 6）。また、個々の入力支援ボタンには任意の文字列を対応付けておくことができるので、「≦」「≧」のような全角演算子など入力しづらい文字をボタンに割り当てることもできる。

支援ボタン定義ファイル：

```
初学者向け入力支援ボタン@48@204,204,204
もし～そうでなければ@12@140,220,200@もし ≪条件式≫ ならば<br>
                        | <br>を実行し，そうでなければ<br>  | <br>を実行する
～の間，繰り返す@12@180,220,200@≪条件式≫ の間，<br>  | <br>を繰り返す
入力@6@255,255,204@≪変数≫ ← input()
出力@6@255,255,204@≪出力文≫ を印刷する
代入@6@255,255,204@≪変数≫ ← ≪式≫
整数@6@255,204,255@整数 ≪変数≫
```

表示例：

図 6: プログラム入力支援ボタンのカスタマイズ例

3　大学の授業においての評価

初学者に対するプログラミング教育においてのPEN の有効性を検証するために、大阪学院大学と大阪市立大学のプログラミング入門授業においてPEN を用いた演習を行った。以下に、その結果を示す。

1. プログラムの基本構造 (3 週)
   - **1.1** 逐次処理（順次処理）
     入出力、計算、値の交換
   - **1.2** 条件分岐（選択処理）
     偶奇判定、大小判定、多分岐判定など
   - **1.3** 繰り返し（反復処理）
     和・階乗の計算、10 進→ 2 進変換など
2. グラフィックス (3 週)
   - **2.1** PEN でのグラフィックスプログラミングの基本
     座標系の説明、描画関数の使い方、RGB の色指定、乱数、繰り返し処理を用いたグラフィックスの描画
   - **2.2** アニメーションを行う
     円の移動、放物線の軌跡、信号機など
3. 配列 (3 週)
   - **3.1** 配列の基本
     数列、配列内容のグラフ表示、最大値・最小値の検索、選択ソートなど
   - **3.2** 2 次元配列
     行列の積、ドットキャラクタの定義と表示、盤面上の指定されたマスにコマを配置
4. ファイル (2 週)
   - **4.1** ファイルからの読み込み
     文字数・行数のカウント、ドットキャラクタ用の配列データのファイルからの読み込み（2 次元配列の例題・演習と連携）
   - **4.2** ファイルへの書き込み
     テキストファイルのコピー、簡易ビットマップエディタなど

図 7: テキストの構成

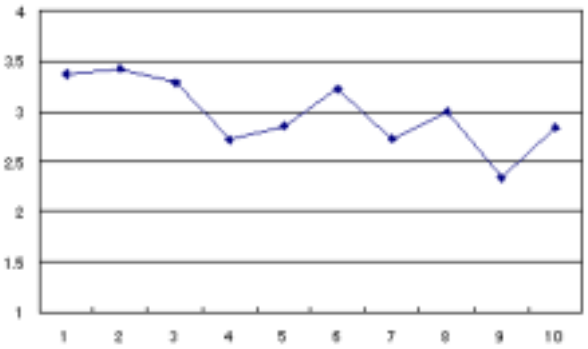

図 8: 理解度自己評価の推移

3.1　大阪学院大学での授業例

大阪学院大学情報学部においては、2005 年度から、前期に開講する 1 年生向けのプログラミング入門の講義 (座学) および、演習 (それぞれ全 14 回、90 分/回) で PEN を利用している。

2005 年度の授業では、変数や制御構造といった、プログラミングの入門教育のみに PEN を用い [6]、その後、C 言語でのプログラミング教育に移行した。しかし、授業日程の関係上、前期の授業では PEN、C ともに 1ヶ月 (4、5 週) の期間しかとれず、PEN(xDNCL) と C の文法の違いやプログラミング環境の違いに混乱し、理解が不十分なまま、その期を終えた学生が少なくなかった。後期は引き続き C 言語を用いて授業を行ったが、夏休みのブランクもあり、前期の学習内容の復習に大きな時間を割かなければいけない結果となった。この問題の解決法としては、PEN から C への移行をスムーズにするための授業法などの改善が考えられる。しかし、現状で受講生間に理解度の大きな差があることを考慮すると、大阪学院大

学での授業においては、プログラミングに関しての一通りの基礎的な内容を PEN を用いて教育する方がよいと判断した。この結果、今年度（2006 年度）は、最初に情報倫理を含めたリテラシー教育を 3 週行い、残りすべての時間を PEN によるプログラミングの講義および、演習にあてることとした。

ここで利用したテキストの内容を図 7 に示す。このテキストを用い、講義では例題の解説と演習問題を考えるためのヒントとなるような説明などを行い、演習ではそれを受けて PEN を実際に用いた例題の入力、検証と演習問題のプログラミングを行ってもらった。なお、講義は著者の一人が担当し、演習は PEN の開発には関わっていない二人の教員が担当した。

毎回の演習後には、その日の演習についてのアンケートをとった。その中で共通の質問項目として「今日の授業の理解度は」という設問を用意し、これを「理解できた」(4)、「だいたい理解できたと思う」(3)、あまり理解できていないと思う」(2)、「ほとんど理解できなかった」(1) の 4 段階で毎回、自己評価してもらった。図 8 に、その平均値の推移を示す。4 回目でグラフィックスに関する演習が開始し、複雑さが増すことから一時的に理解度は下がっているが、その後は課題が難しくなっているにもかかわらず、理解度は上昇し、その後も理解度は大きく下がっていない。実際に、アンケートの自由記述では、

> 「x 座標を考えるのが難しかったけどプログラムが完成するとかなり嬉しかったです♪ あと応用演習はキレイに作るために数値を微調整しながら何度も実行するのは面倒な作業だったけど面白かったです☆」

というように、積極的に楽しんで課題に取り組む声が聞かれた。

9 回目の授業は、5 目並べやオセロゲームなどに使えるような 8x8 マスの盤面を設定し、入力により指定されたマスにコマを配置する例題をベースとしたやや複雑な演習が含まれていたため、理解度は低くなったと考えられる。まただし、演習後に発展課題のゲーム作成まで行った者もおり、意欲のある学生には挑戦しがいのある課題となった。また、ファイル I/O に関しては、「面白かった。ファイル入出力を習えばバリエーションが増えるので楽しみです。」といった意見も聞かれた。

表 1: 授業内容の概要

第 1 回:	目的、変数、逐次処理
第 2 回:	条件分岐、繰り返し
第 3 回:	実数の扱い、乱数
第 4 回:	ファイル I/O
第 5 回:	描画関数の扱い

3.2 大阪市立大学での授業例

2006 年度前期、大阪市立大学の 2 部の共通教育科目「プログラミング入門」の中で、PEN を試用した。受講者は 12 名で、この授業、全 15 回（180 分/回）のうち、最初の 5 回で PEN を利用し、6 回目以降はそれを受けて、Java を用いて授業を行った。授業内容の概要を表 1 に示す。授業は 180 分であるが、講義はほぼ 90 分として、残りは、各自のペースで練習問題を解いていくというスタイルにした。

毎回、授業後にアンケートを行い、各例題、練習問題の理解度等、および、全体の理解度／興味などを調査した。アンケートによると、第 4 回のファイル I/O が難しかったようで、理解度／興味がもっとも低く、第 5 回の図形描画が最も興味を引いたようだった。

全回を通じて、学生は意欲的に取り組んでおり、最も難しかった第 4 回に一人が「面白くなかった」と回答しただけで、それ以外の回には「面白くなかった」という回答はなかった。

最後の授業の感想をいくつか紹介する。『プログラムができた瞬間の喜びを知りました。』『この授業は、すべての講義の中で一番難しかったです。ただ一番面白かったです。本当に楽しかったです。自分で考えて (だいぶ教えてもらいましたが) プログラムを作っていく難しさ、楽しさがよくわかりました。』『プログラムを作る難しさが本当にわかりました。こんなに単純なプログラムでさえもこんなに大変なのに、日常生活のプログラムなんかは、とても大変なんだなと思いました。』

また、PEN については、以下のような意見も頂戴した。『...「プログラミング入門」であって、「JAVA 入門」というわけではないので、いっそのこと「PEN」オンリーの授業でもいいのではないかと思いました。(開発途上とはいえ、かなりの機能が実装されてきているようですし、プログラミング初学者にとって

は取っ付きやすいのは間違いないと思います。)』

上記のような感想が得られたのは、この授業が選択科目であり、プログラミングを学びたいという意欲のある学生が集まったということが大きな要因だと思われる。

4 おわりに

ここでは、プログラミングとは何かを理解し、コンピュータの本質を理解することの助けとなるようなプログラミング入門教育のための環境である PEN について紹介した。PEN はセンター試験の出題で用いられている日本語表現の入試用プログラミング言語 DNCL に準拠した言語 xDNCL を採用し、入力が煩雑となる日本語表記のプログラムの入力支援機能やプログラムの実行状況の表示機能などを備える。なお PEN は、Web サイト [7] からダウンロード可能である。

xDNCL では制御構造に含まれる予約語は日本語で記述しているが、変数や関数名などの識別子には日本語は使えない。現在、日本語の識別子を導入する方向で検討を行っているが、変数名を日本語とすると制御構造との区別がつきにくくなり、プログラムの可読性が下がる可能性もある。そのような問題が起きないよう表示方法の工夫なども含め作業を進めていきたいと考えている。関数名に関しては、今回の演習を進めていく上でも英語の表記が多くなり、日本語を使ったプログラミング環境とは言えなくなっているという指摘も受けている。日本語の関数名を付けた場合、名前の一貫性をどう保つかなどの問題が出てくるが、現在、入力支援環境の整備も含めて拡張を検討している。

また、初学者環境であるのになぜ変数宣言が必要なのかという意見もある。これは、教える環境によって変わる要素であるので、教員の判断によって変数宣言を省略することを選択できるようにすることを考えている。また、制御構造を表す予約語のゆらぎを許している部分についても異論が予想されるので、同様に選択可能とする方向で考えていきたい。

PEN のエディタは入力支援機能としてボタンで制御構造のスケルトンが挿入できたり、インデントを自動挿入する機能を持っているが、学生によってはその構造を壊してしまう者も居る。そのような学生は、制御構造を認識・理解できていない可能性が高く、サポートが必要である。今後は、構造エディタとしての機能を強化し、インデントなどの構造を壊せないようにすることや、構造を再構築する機能などを追加する必要があると考えている。

現在、PEN は筆者らが所属する大学の他、京都ノートルダム女子大学、千里金蘭大学などでも利用して頂いている。また、本年度前期には、大阪大学人間科学部の「情報活用基礎」の授業の一部でも利用して頂いた。これらの大学の使用経験に基づく評価はこれからの課題である。今後は、大学だけでなく高校での利用など対象を広げ、実際のプログラミング教育での使用経験を積み重ねて、PEN の改良に努めていきたい。

参考文献

[1] 西田 知博 : “高等学校における教科「情報」関連の現状と今後の展望”, サイバーメディアフォーラム No.6, pp.5-10 (2005-09).

[2] 中西通雄, 松浦敏雄 : “情報処理教育の 2006 年問題への対応”, サイバーメディアフォーラム No.6, pp.23-28 (2005-09).

[3] 情報処理学会情報処理教育委員会: “2005 年後半から 2006 年初頭にかけての事件と情報教育の関連に関するコメント”, http://www.ipsj.or.jp/12kyoiku/statement2006.pdf (2006-2).

[4] 情報処理学会情報処理教育委員会: “日本の情報教育・情報処理教育に関する提言 2005”, http://www.ipsj.or.jp/12kyoiku/proposal-20051029.pdf (2005-10).

[5] 中村 亮太, 西田 知博, 松浦 敏雄 : “プログラミング入門教育用学習環境 PEN”, 情報処理学会研究報告 2005-CE-81 (2005-10).

[6] 西田 知博, 中村 亮太, 松浦 敏雄 : “初学者用プログラミング環境 PEN”, 平成 17 年度情報処理教育研究集会講演論文集,pp.467-470(2005-11).

[7] 初学者向けプログラミング学習環境 PEN Web ページ, http://www.media.osaka-cu.ac.jp/PEN/ .

[8] 大学入試センター: “センター試験用手順記述標準言語 —DNCL—”, 平成 15 年度センター試験試験問題評価委員会報告書, pp.258-259, http://www.dnc.ac.jp/old_data/exam_repo/15/pdf/15hyouka41.pdf (2003).