初学者用プログラミング学習環境PENの実装と評価

西田知博[†1] 原田章[†2] 中村亮太[†3,☆]
宮本友介[†4] 松浦敏雄[†3]

制御構造などのプログラミングの基礎を短時間で習得することを目指したプログラミング学習環境PENを開発した．本論文では，PENの実装とその評価について報告する．PENでは，大学入試センターなどの入試で用いられている言語を用いているので，付加的な説明を行わなくても容易にプログラムが理解できる．また，プログラムの入力補助機能を備えることで，プログラム作成時の誤りの混入を減らすことに寄与している．また，ステップ実行機能，スロー実行機能，変数表示機能などにより，プログラムの動作を観察しやすくしている．授業実践のアンケート結果から，PENは初学者におおむね好評であることを確認した．また，JavaScriptを用いた授業との比較では，自己評価と試験による分析の結果，双方ともPENを用いたクラスの方が理解度が高くなり，プログラミングの入門教育環境としてのPENの有用性が示唆される結果が得られた．

Implementation and Evaluation of PEN: The Programming Environment for Novices

Tomohiro Nishida,[†1] Akira Harada,[†2] Ryota Nakamura,[†3,☆]
Yusuke Miyamoto[†4] and Toshio Matsuura[†3]

We have developed a programming environment 'PEN.' This is for a novice to learn the basic features of programming such as control structures. In this paper we describe the implementation of PEN and report the comparative experiments for novice users' comprehensions between PEN and JavaScript. PEN's language specification was derived from a programming language DNCL which has been carried out National Center for University Entrance Examinations. In this language, the keywords such as 'if', 'then' are written in Japanese. Therefore novices easily understand the points without translations. Other PEN's features are as follows; (1) input methods that assist programming, avoiding syntax errors caused by typing error, (2) step-by-step execution, (3) execution speed control, (4) displaying variables. According to the questionnaire submitted from the subjects, PEN had a good reputation among novice users. Furthermore, in comparison of DNCL using PEN to JavaScript with a conventional environment, PEN showed a significant advantage in subjects' comprehension of programming by the results of the subjects' self-evaluations and the examination scores. From the above results, we conclude that PEN is effective in programming education for novices.

1. はじめに

学習指導要領の改訂により，高等学校の教科「情報」をはじめとして，初等中等教育において本格的な情報教育が開始された．しかし，現行の学習指導要領では「情報自体の理解」を重視し，コンピュータに代表される情報機器が「情報を扱う便利な道具」と位置付けられ,「情報処理の理解」を深めるものとはなっていない．しかし,「情報処理の理解」のないままに，利用法のみを学んでもその理解は不完全となり，本質的な誤解や認識不足によって大きなトラブルを引き起こすこともありうる[1]．こういった背景から，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」[2]を出し，プログラミングをはじめとした『「手順的な自動処理」の理解』の重要性を訴えている．

しかし，現実を見ると，プログラミング教育は大学

†1 大阪学院大学情報学部
Faculty of Informatics, Osaka Gakuin University
†2 甲子園短期大学
Koshien Junior College
†3 大阪市立大学大学院創造都市研究科
Graduate School for Creative Cities, Osaka City University
†4 大阪大学人間科学研究科
Graduate School of Human Sciences, Osaka University
☆ 現在，大阪府立泉北高等学校
Presently with Semboku High School

の情報教育においてさえも軽視されている傾向にある．かつては,「コンピュータの教育＝プログラミング教育」といった時代もあったが，アプリケーションプログラムの充実にともない，現在では，理系の一部の学部/学科を除いて，プログラミング言語を用いた実習をともなうような授業はほとんど行われていない．プログラミングが取り上げられない大きな理由としては，プログラミング言語は難しいものであり，その習得には多くの時間が必要と思われていることがあげられる．しかし，プログラミングの入門教育の目標を，職業プログラマの養成ではなく，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と定めるならば，適切なサポートツールを用意することで，比較的短い学習時間でこの目標を達成できると思われる．

我々は，上記のプログラミング教育を達成するためのサポートツールとして，初学者教育用プログラミング環境 **PEN**（Programming Environment for Novices）を構築した．PEN は，手続き的なプログラミングを短期間で容易に習得できるような，プログラミング学習環境の提供を目指し，分かりやすいプログラミング言語を用い，構文エラーなども生じにくくするための入力支援機能を備えている．また，プログラムの実行の様子を把握しやすくするための機能も備えている．本論文では，2 章で初学者用プログラミング学習環境への要求の考察，3 章で PEN の実装について述べ，4 章で教育実践においての評価を行う．

2. 初学者用プログラミング学習環境への要求

この章では，従来のプログラミング学習で初学者が直面する問題をあげ，本研究で考えるプログラミング教育の目標を明白にしたあと，その教育目標を達成するためのプログラミング環境としてどのような機能が必要であるかについて考察する．

2.1 プログラミング学習の難しさ

初学者は，プログラミング学習の初期段階でつまずくことが多い．学習初期段階ではタイプミスに起因する文法エラーが多く発生し，意味の理解できないコンパイルエラーメッセージを（多くの場合英語で）突き付けられ，それだけで自信をなくす者が多い．文法エラーの修正ばかりに気をとられていると，プログラムの構造をどのように組み立てるかといった，全体への配慮ができず，プログラミングの力がなかなか身につかない．また，コンパイルエラーがなくなっても，論理的なエラーのために予想と異なる結果が返ってくると対処することが難しい．論理的なエラーはエラーメッセージが表示されないので，間違い箇所を見つけるためには，変数に何が代入されているかを逐次チェックしていかなければならない．デバッガを用いれば，このようなチェックは比較的容易に行うことができるが，多くのデバッガは初学者が使うことを考慮しておらず，使いにくい．

2.2 「プログラミング」入門教育の目標

1 章でも述べたとおり，ここで想定している「プログラミング入門」は，文献 3) で述べられている「プログラミング」を目指すものであり，職業技能としてのプログラミング言語の習得を目的とするものではない．ここでの「プログラミング」習得の目的は，文献 4) に示されているとおりで，自分が考えたことを決められたルールに従って，きちんと記述するという訓練を通じて，自分の知識状態や考え方を明確にする力を養うことである．また，デバッグという間違い発見のプロセスを通して，自ら発見するという学習をさせ，自立的な思考力を養うことができるということも重要である．

さらに，プログラミングを体験することで，コンピュータというものの本質，すなわち，コンピュータは，プログラムに書かれたことを忠実に実行するだけの機械であって，それ以上でもそれ以下でもないことを実感できる[5]．

以上のことよりここでは,「プログラミング」の習得の目標を，上記のような体験に基づいてコンピュータの本質を理解することとする．このことは，文献 2) の,『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』が「情報処理の理解」につながるとの主張とも一致している．また，高校の教科「情報」や，大学の一般情報教育の一部として組み込み可能となるように，短時間で習得が可能であることも目標とする．

2.3 学習環境に必要な要件

2.2 節で示した「プログラミング」入門を，なるべく短時間に習得するために必要なプログラミング環境について考える．

プログラミングを理解するためには，まずプログラムが読めることが必要である．このため，前提知識がほとんどなくても読めるような，分かりやすい言語を用意する必要がある．もちろん，プログラムを自分で書いて実際に動かしてみることが重要であるので，そのために必要であれば処理系を実装しなければならない．また，プログラムを書く際にタイプミスに起因する文法エラーなどでつまずかないように，プログラム作成を支援する機能が必要である．さらに，論理的な

間違いを発見しやすいように，初学者がプログラムの実行状況を観察できる機能などが必要である．

2.4 先行研究との比較

初学者用のプログラミング環境に関する研究は昔から多く行われている．たとえば，INTELLITUTOR[9]は，学習者が書いたプログラムに対して，知的処理を用い適切なアドバイスを送ることを目指したシステムである．論理的な間違いに対してもシステムが補助することは，学習者の大きな助けになる．X-Compiler[10]は，プログラムがどのように翻訳されて実行されるのかという，コンパイラの中身や実行の過程を見せることに力点が置かれ，プログラミングを通してコンピュータの仕組みを理解させる環境として優れている．しかし，これらのシステムは，教育現場においての評価がなされておらず，学習者に対する教育効果が不明確である．

現在，日本の教育現場で実践報告がなされている初学者用のプログラミング環境は，たとえば，タイルスクリプティングを用いる Squeak eToys を利用した小学生などへの教育[11]など，GUI を利用し，絵などのオブジェクトに対する機能を組み立てる形でプログラムを作成するものが多い．しかし，このような環境では，絵を描き，それを単に動かすというだけの体験的内容に終始しがちである．高校生や大学生が「自分の知識状態や考え方を明確にする力を養う」ためには，タイルなどの GUI 部品を組み立てるだけのプログラミングでは不十分である．Squeak はより進んだプログラミングを Smalltalk で行うことができるが，Smalltalk を用いたプログラミングは難易度が高く，優れた指導者が必要となるなどの理由で，初学者教育には向かない．

高校生や大学生向けの環境としては，構造化チャートである PAD を利用した JPADet[12]がある．これは，アルゴリズム教育に用いるためには優れた環境ではあるが，構造化チャートの理解には一定の時間が必要で，短期間での学習には向かない．また，我々はプログラムコードを書くことが「プログラミング」の学習には必要だと考え，図式を利用した表現は採用しなかった．

オブジェクト指向言語を用いた初学者用プログラミング環境としては，ドリトル[13]や Nigari[14]などがある．オブジェクト指向の考え方はきわめて重要ではあるが，メソッドの記述は手続き的に書かざるをえず，オブジェクト指向の考えに至る前段階で手続き的なプログラミングの習得が必要になる．我々は，初学者に対してまず手続き的なプログラミング教育が必要と考えている．

そこで PEN では，手続き型で日本語表記の言語を採用し，制御構造を意識しながらプログラム記述ができるような入力サポートの機能などを充実することにより，初学者が使いやすく，短期間で「プログラミング」が理解できるようになる環境を提供することを目指した．

3. PEN の実装

2.3 節で述べた要件を満たす初学者向けプログラミング学習環境 PEN を作成した．

図 1 は PEN の実行時のスナップショットである．PEN は Java アプリケーションとして実装されており，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などを持つ．

3.1 プログラミング言語

2.3 節にあげた要件を満たす言語として，我々は，大学入試センターの入試科目「情報関係基礎」で用いられている手順記述言語 **DNCL**[6]に注目した．これは，DNCL は日本語をベースとした記述言語であり，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優れていると思われるからである．

PEN では，DNCL に拡張を加えた言語を用いることとした．これを **xDNCL** とよぶ．これは，DNCL が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ったりするからである．拡張した点は，以下のとおりである．

(1) 変数や型を意識させるための変数宣言の追加
変数型としては，整数，実数，文字列の 3 つを用意した．また，それぞれの配列（多次元も許す）を使用することもできる．

(2) 命令の補強
多分岐処理や，繰返しからの脱出，型変換，注釈文などの命令を追加した．

(3) 組み込み関数の追加
文字列操作関数，数学関数，グラフックス関数，ファイル I/O 関数を用意した[7),8)]．

図 2 に xDNCL によるプログラムの記述例を示す．言語仕様は DNCL に，変数宣言の追加と演算子の定義で若干の変更を行っているが，基本的にはほとんど変化がない．図 2 から分かるように xDNCL のプログラムを読むのは容易である．

DNCL（および xDNCL）では，言語の意味をほと

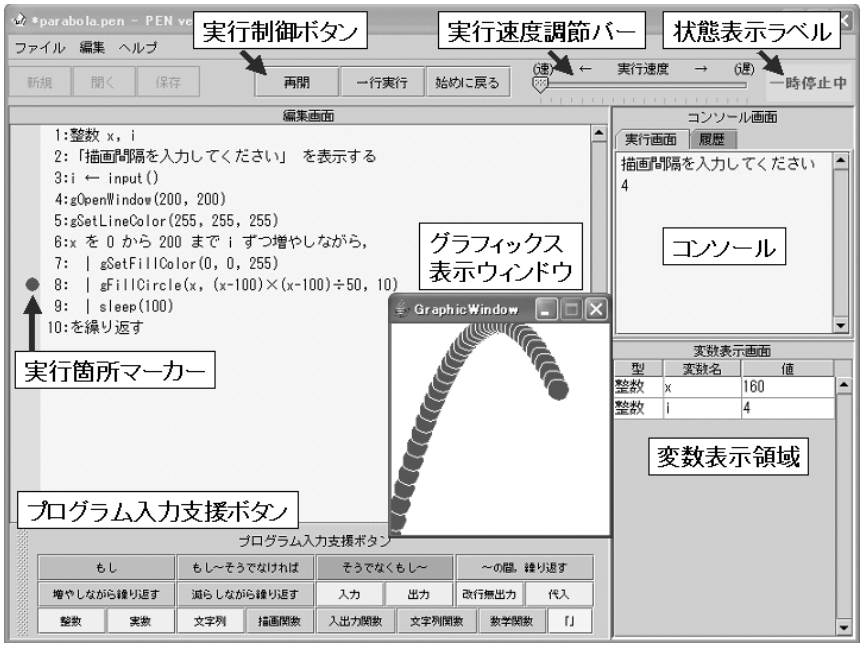

図 1 PEN の実行時の表示例
Fig. 1 A snapshot of PEN.

んど説明していない．このため,「繰返しの終了条件式をいつ評価するか」など，プログラミング言語としての曖昧性が残されている．プログラミング言語を利用する際に，厳密な言語の意味定義は重要ではあるが，この種の曖昧性については，プログラミングの学習がある程度進んだ段階でないと学習者が理解することは困難である．指導者側から学習者にこの種の曖昧性が問題となるプログラムを提示しないのはもちろんであるが，学習者がプログラムを作成する際にも指導者がそれをチェックし，適切な助言を与えることでこの種の問題を回避できる．

3.2 プログラム入力支援機能

3.2.1 プログラム入力支援ボタン

前節で述べたように，xDNCL で記述されたプログラムは読みやすいが，日本語をベースとしていることから，キーワードなども長めであり，プログラム作成時に，文法どお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．そこでプログラムを書く際の入力支援機能の整備が重要となる．

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．以下に，プログラム入力支援ボタンの使用例を示す（**図 3**）．

```
 1: 実数 shin,tai,bmi
 2: 「身長を入力してください (cm)」 を表示する
 3: shin ← input()
 4: 「体重を入力してください (kg)」 を表示する
 5: tai ← input()
 6: bmi ← tai ÷ ((shin ÷ 100) × (shin ÷ 100))
 7: 「BMI = 」と bmi を表示する
 8: もし bmi < 20 ならば
 9:   | 「痩せています」を表示する
10: を実行し，そうでなくもし bmi < 24 ならば
11:   | 「正常です」を表示する
12: を実行し，そうでなくもし bmi < 26.5 ならば
13:   | 「やや肥満です」を表示する
14: を実行し，そうでなければ
15:   | 「肥満です」を表示する
16: を実行する
```

図 2 xDNCL の記述例（BMI 算出）
Fig. 2 xDNCL sample program (BMI calculator).

(1) 図 3 (a) の状態で入力支援ボタン **[〜の間，繰り返す]** をクリックすると，図 3 (b) のように制御構造のテンプレートが編集画面に挿入される．

(2) 次に**≪条件式≫**など，≪と≫で囲まれた部分（この部分全体で 1 つの文字として扱われる）をマウスでクリックするか，その位置にカーソルを移動することにより，図 3 (c) のように≪と≫で囲まれた部分が選択される．

(3) 選択された部分を実際の条件式などに書き換えて，さらに 4 行目以降に命令を挿入していくこ

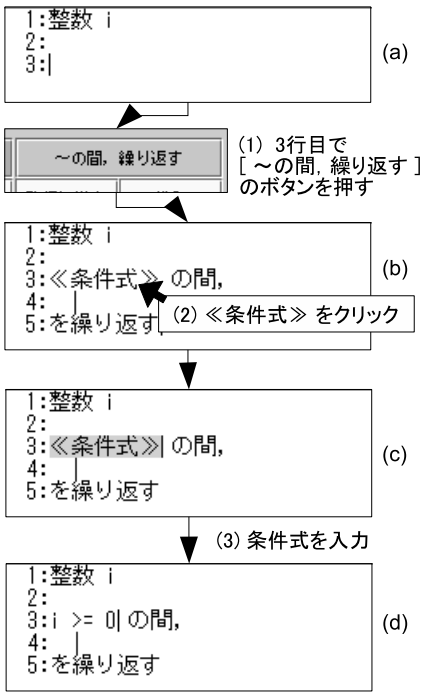

図 3 入力支援ボタンによるプログラミング例
Fig. 3 An example of using supporing buttons for input.

とにより，プログラムを作成することが可能である．

3.2.2 インデントの自動挿入

DNCL 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（**図 4**）．

図 4 (a) のように 5 行目に挿入ポイントがある状態で，**［もし～そうでなければ］**の入力支援ボタンを押すと，インデントを付加した制御構造のテンプレートが図 4 (b) のように挿入される．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．

3.3 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**実行箇所マーカ**でつねに明示している．**変数表示画面**では，実行中のプログラムで用いているすべての変数の値をつねに表示している．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．

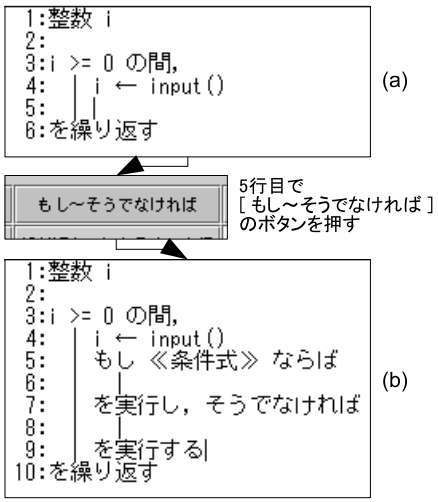

図 4 インデントの自動挿入
Fig. 4 An example of auto-indentation function.

表 1 実行状態の一覧
Table 1 A list of executing status.

状態表示	意味
実行待ち	プログラムを実行していない状態．プログラムの編集が可能．
実行中	プログラムを実行している状態．
一時停止中	「一時停止」や「一行実行」によって実行が停止されている状態．
入力待ち	プログラム内の入力文による入力待ち状態．入力はコンソール画面で行う．
実行終了	プログラムの実行が終了した状態．再度，プログラムを実行する場合は「始めから実行」ボタンを押す．また状態を「実行待ち」に初期化するには「始めに戻る」ボタンを押す．

3.3.1 実行状態表示ラベル

図 1 の右上に**実行状態表示ラベル**があり，プログラムの実行状態を把握できるようにしている．プログラムの状態としては，**表 1** に示すように，5 つの状態がある．プログラムの実行状態を表示することにより，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，一目で分かる．

3.3.2 スロー実行

プログラムを実行しながら観察できるよう，実行速度を調節できる機能を実装した．実行速度の調節は図 1 上部右の**実行速度調節バー**で行う．実行速度はプログラムの実行中でも変更できるので，注目箇所で実行速度を遅くし，じっくりと観察するといった使い方も可能である．

表 2 授業実践の一覧
Table 2 A list of practical classes.

時期	大学	科目名（必修/選択）	回数	受講者数
2005 年前期	大阪学院大	プログラミング入門/演習 I（必修）	4	111 名
2006 年前期	大阪学院大	プログラミング入門/演習 I（必修）	11	94 名
2005 年前期	大阪市立大（2 部）	情報処理 I（選択）	3	40 名
2005 年後期	大阪市立大	情報処理 II（選択）	3	29 名
2006 年前期	大阪市立大（2 部）	プログラミング入門（選択）	4	15 名
2006 年後期	大阪市立大	プログラミング入門（選択）	5	30 名
2005 年後期	京都ノートルダム女子大	コンピュータ基礎演習（選択）	5	36 名
2006 年前期	京都ノートルダム女子大	コンピュータ基礎演習（選択）	7	45 名
2006 年前期	大阪大人間科学部	情報活用基礎（必修）	3	149 名

3.3.3 ステップ実行

プログラムの実行過程を詳しく観察できるよう，**ステップ実行**機能を用意している．ステップ実行は，図 1 上部中央にある実行制御ボタン群の**一行実行ボタン**で実行できる．一行実行ボタンを押すと，実行箇所マーカの指し示している行を実行し，次の命令に実行箇所マーカが移り，実行状態が「一時停止」となる．

3.3.4 変数表示画面

プログラムの実行過程を観察する際，変数にどのような値が代入されているかなどの情報が必要になる．変数に関する情報は，図 1 の右下にある**変数表示画面**に表示される．表示される項目は「データ型」,「変数名」および「変数の値」である．

4. 評　　価

4.1 PEN を活用した授業実践

2005 年前期より PEN を活用した授業実践を開始した．我々が直接担当するなどで関係した授業を**表 2** に示す．すべて半期の授業であり，回数欄は PEN を用いた授業回数☆である．これらの授業で，毎回の授業後にアンケート調査を行った[15]．これらのアンケートから，学生のモティベーションは高く，

> 「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，楽しさがよく分かりました」

といった声など，PEN がおおむね好評であることがうかがえる．このうち，JavaScript とその実行環境を用いた授業との比較が可能であった大阪大学での実践について，次節以降で詳述する．

☆ 1 コマ 90 分．大阪学院大学での授業は，座学の講義 1 コマとそれを 4 クラスに分けた演習 1 コマがセットになっている．大阪市立大学での授業は 2 コマ連続であるが，おおむね講義 1 コマと演習 1 コマを行っている．

4.2 JavaScript との比較

4.2.1 経　　緯

大阪大学では現在，全学において 1 年次における情報教育科目が開講されているが，文学部では 1994 年以降，人間科学部では 1995 年以降において，情報教育科目「情報活用基礎」が必修科目となった．以後，約 10 年にわたる調査で，両学部間では受講前でのコンピュータ習熟度やコンピュータ不安がほぼ同質と考えられることが明らかとなっている[16),17)]．2006 年度も，受講前，中間（8 回目の講義時），期末に 3 回のアンケート調査を行った[18)~20)]．

また，両学部の授業担当者は共同して授業内容や教材についての情報交換を行いながら,「情報活用基礎」におけるクラス別教育の効果などさまざまな工夫を実践してきた[16),21)]．

2006 年度からは，高等学校において教科「情報」を履修してきた学生を受け入れることから，授業内容を見直し，両学部において授業の一部にプログラミングを取り入れることで合意した．しかし，どのプログラミング言語を用いるかの判断は各学部に委ねられ，文学部では実用性を重視し JavaScript を採用し，人間科学部では学習の容易さを重視し，PEN を用いることにした．

以下では，これら 2 学部の授業実践を比較する．

4.2.2 授 業 内 容

人間科学部では 149 名，文学部では 201 名の受講生を 3 つのクラスに分け，各クラスを 1 名の教員と 2 名もしくは 3 名のティーチングアシスタント（TA）で担当した．両学部の担当教員は互いに授業の進行状況や講義資料の情報を交換しながら，同じ計算機環境☆☆で授業をすすめ，プログラミング演習は終盤のテーマとし，人間科学部では 3 回，文学部では 4 回を割り

☆☆ OS: Vine Linux Educational Edition 2.0

(a)

キーボードから三角形の底辺と高さを入力し，面積を出力するプログラムを作成せよ．

(b)

キーボードから点数 (x) を入力し，x が 60 以上なら seiseki に 1 を 代入し， x が 60 未満なら seiseki に 0 を代入し，seiseki の値を印刷するプログラムを書け．

(c)

1 から 10 までの整数の和 $\sum_{n=1}^{10} n$ を求めるプログラムを作成せよ．

(d)

100×100 ピクセル のウィンドウに，縦横 20 ピクセルずつの間隔で直線を引き，方眼紙を作成せよ．

図 5 練習問題の例
Fig. 5 Examples of exercises.

当てた☆．プログラミング環境は，人間科学部は PEN を用い，文学部は JavaScript とその実行環境☆☆を用いた．

プログラミング演習の授業は，まず例題で概念やアルゴリズムを説明し，その後は各自で練習問題を解く演習を行う形式で構成し，人間科学部（PEN）では次の項目を扱った．

(1) 変数の宣言・代入・出力文・逐次処理
(2) 制御構造（条件分岐・繰返し）
(3) 関数の呼び出し

以下に人間科学部（PEN）の授業で用いた練習問題の一部をあげる．なお，文学部（JavaScript）でもほぼ同様の内容を扱っている[22]．

第 1 回 変数の宣言・代入・出力文・逐次処理

変数と型の概念を説明し，入力・演算・出力を行う基本的なプログラムを作成した．また，PEN のトレース機能を用いて，プログラムが逐次的に処理される様子を観察した（**図 5** (a)）．

第 2 回 制御構造（条件分岐・繰返し）

入力・演算・出力の処理の流れに条件分岐と繰返しを組み合わせることにより，より複雑な処理が可能となることを観察した（図 5 (b), (c)）．

第 3 回 関数の呼び出し

PEN に用意されている関数を用いて，図形描画を行うプログラムを作成した．また，繰返し構文などと組み合わせることで，幾何学的な図形の描画やアニメーションが実現できることを観察した（図 5 (d)）．

なお，文学部（JavaScript）では，第 3 回目の授業として，図形描画の代わりに，イベントハンドラを用いてフォームへの入力を処理する問題を扱った．

4.3 結　　果

4.3.1 調 査 対 象

プログラミング教育の成果について評価を行う際，プログラミングに関する授業すべてに出席し，必要な指標がすべて揃っている 1 年生（人間科学部：142 名，文学部：179 名）を分析対象とした．その人数は，人間科学部が 93 名（男性：36 名，女性：57 名），文学部が 101 名（男性：34 名，女性：67 名）の計 194 名であった．学部による男女の偏りを適合度検定によって調べたところ，有意水準 5%で偏りは見られなかった（$\chi^2(1) = 0.535$）．

4.3.2 学部間の差異

人間科学部では PEN，文学部では JavaScript を用いてプログラミング教育を行ったが，その成果を比較する前に，まず，プログラミング教育を行う前の段階で学部間にコンピュータの利用経験や習熟度自己評価に差があるかどうかについて検討した．

それぞれの差を調べるための指標として，コンピュータ利用経験得点と習熟度自己評価得点の 2 つを算出した．利用経験得点は，受講前アンケートの 7 項目（文献 18) の質問 1 参照）の回答を平均して求めた．習熟度自己評価得点は，中間アンケート 63 項目（文献 19) の質問 2 参照）から，表計算（7 項目），プレゼンテーション（3 項目）を除いた 53 項目の回答を平均して求めた．こうした内容を分析に含めなかったのは，アンケート実施時点で，人間科学部と文学部がともに学習済みの項目のみを分析対象としたからである．

図 6，**図 7** は，利用経験および習熟度自己評価の代表的な項目と回答様式を表したものである．回答者はこうした項目に対して，自分の状態に近い数値を選択することによって回答した．分析にあたっては，回答者が選択した数値をそのまま用いた．したがって，利用経験得点と習熟度自己評価得点は，1 から 5 までの値をとり，値が高くなればなるほど経験および自己評価の程度が高くなることを示している．なお，図 7 の「未学習」は該当する項目に対して，回答者がまだ授業で習っていないと判断した場合の選択肢である．

このようにして算出した 2 つの指標を従属変数として，学部（人間科学部，文学部），性別を要因とする二元配置分散分析をそれぞれ行った．なお，要因として

☆ 文学部でも，3 回で人間科学部とほぼ同じ内容を履修できるように授業を構成した．4 回目は，それまでの学習の復習を行うための演習という位置づけである．
☆☆ テキストエディタは gedit 2.4.1 を，Web ブラウザは Mozilla 1.7.12 を用いた．

コンピュータを使って文章を作成する
... **全く行ったことがない** 1 2 3 4 5 **頻繁に行った**

図 6 コンピュータ利用経験を問う項目の代表例
Fig. 6 An example question to ask experience for a computer.

指定されたディレクトリ (フォルダ) の中を見る
..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**

図 7 習熟度自己評価を問う項目の代表例
Fig. 7 An example question for self-evaluation of proficiency.

表 3 プログラミング教育前の学部間差に関する分散分析結果
Table 3 ANOVA table of the pre-test.

	$\bar{x}_h$	$\bar{x}_l$	F 値	p 値
利用経験	2.76	2.86	0.193	0.661
習熟度自己評価	4.02	3.86	3.348	0.069

F 値の自由度は $df_1 = 1, df_2 = 190$

性別を含めたのは，学部間の差を比較する際に，性別の影響を除去するためである．本分析では，学部間の差に関して考察することを目的としているため，性差に関する詳細な結果については触れないこととする．これは以下の分析結果においても同様である．

両方の変数で学部の主効果および学部，また，念のため性別の交互作用を分散分析した結果，有意差は見られなかった．**表 3** は，各変数の学部別平均（人間科学部：$\bar{x}_h$，文学部：$\bar{x}_l$，以下同様）と分散分析の結果をまとめたものである．

こうした結果は先行研究 16) とも一致しており，プログラミングの授業を行う前の段階では，コンピュータの利用経験や習熟度自己評価に関して 2 つの学部間で大きな差異はなかったということが示唆された．このことから，プログラミングの授業を行った後の各指標に関して学部間の差が認められたとすれば，授業の効果であると考えることができる．

4.3.3 理解度（自己評価）の差異

プログラミングに関する授業内容に対して，受講学生が自身の理解度をどのように評価したかについて，授業後に行ったアンケートの回答結果から両学部を比較した．なお，授業後に行ったアンケート項目の詳細は文献 15) を参照していただきたい．

まず，人間科学部，文学部とも 3 回目の授業後に行ったアンケート結果を比較した．これはプログラミングに関する基礎的な内容の説明が終了する回に対応している．

表 4 理解度（自己評価）に関する分散分析表（人，文とも 3 回目）
Table 4 ANOVA table of self-evaluation (h, l: 3rd).

	平方和	自由度	F 値	p 値
学部	82.831	1	181.042	0.000
性別	1.067	1	2.331	0.128
学部 × 性別	0.348	1	0.761	0.384
誤差	86.929	190		
総和	1,081.001	194		

表 5 理解度（自己評価）に関する分散分析表（人：3 回目，文：4 回目）
Table 5 ANOVA table of self-evaluation (h: 3rd, l: 4th).

	平方和	自由度	F 値	p 値
学部	1.835	1	4.191	0.042
性別	0.436	1	0.997	0.319
学部 × 性別	0.318	1	0.726	0.395
誤差	83.203	190		
総和	1,579.427	194		

分析に用いた項目は，「変数とはどんなものか」，「条件分岐（if 文）の使い方」，「繰返し（while 文）の使い方」であった．いずれの項目も「理解できた」，「まあまあ理解できた」，「あまり理解できなかった」，「ほとんど理解できなかった」の 4 段階の判断基準で回答するようになっていた．分析では，数値が大きくなるほど理解度が高くなるよう，4 つの判断基準に対して，1 から 4 までの整数を割り当てた．そして，3 項目の平均をプログラミングに対する基礎的理解得点と考えた．

この値を従属変数として，学部（人間科学部，文学部），性別を要因とする二元配置分散分析を行った．**表 4** はその分散分析表である．この表から分かるとおり，有意水準 1%で学部の主効果が有意であった．そこで，学部別に平均と標準偏差を見てみると，$\bar{x}_h = 2.86$（$s_h = 0.702$），$\bar{x}_l = 1.51$（$s_l = 0.655$）となった（s_h，s_l はそれぞれ人間科学部，文学部の標準偏差を表す．以下同様）．このことから，プログラミングに関する理解度の自己評価は，人間科学部の方が高いと思われる．

次に，人間科学部の 3 回目と文学部の 4 回目の結果を上と同じ手順で分析した．**表 5** はその分散分析表である．この表から分かるとおり，有意水準 5%で学部の主効果が有意であった．そこで，文学部の平均と標準偏差を見てみると，$\bar{x}_l = 2.69$（$s_l = 0.705$）となった．このことから，3 回目のときに比べて人間科学部と文学部の差は縮まったが，なお，理解度の自己評価は，人間科学部の方が高いと思われる結果が得られた．

4.3.4 理解度（試験成績）の差異

演習によるプログラミングの達成度を見るため，期

表 6 理解度（試験成績）の分散分析表
Table 6 ANOVA table of the examination result.

	平方和	自由度	F 値	p 値
学部	20.289	1	5.516	0.020
性別	0.223	1	0.061	0.806
学部 × 性別	0.022	1	0.006	0.938
誤差	698.907	190		
総和	2,504.000	194		

末試験（ペーパーテスト）においてプログラミングに関する問題を出題した．これは JavaScript でプログラミング演習を行った文学部で出題された問題とも対応しており，プログラムのトレース問題，穴埋め問題，応用問題となっている．なお，このテストは資料の持ち込みおよびプログラムの実行を許可したうえで行った．

出題された問題のうち，トレース問題と穴埋め問題については，トレース機能を有する PEN の方が有利であるので，この問題を除き，ここでは以下の応用問題についてのみの比較を行う．

> ある村人が殿様から褒美を貰えることになった．褒美は米粒で，1 日目 1 粒，2 日目 2 粒，3 日目 4 粒，... と毎日，前日の倍の米粒を貰えるとのことである．30 日目まで，毎日，何粒貰えるのかを表示するプログラムを作成せよ．

この問題の答案を 5 点満点で採点した．なお，採点時に両学部で協議し，採点基準が同様になるようにした．

この得点を従属変数とし，学部と性別を要因とする二元配置分散分析を行った．**表 6** はその結果である．分析の結果，有意水準 5%で学部の主効果が有意であった．そこで，両学部の平均を調べると，$\bar{x}_h = 3.39$（$s_h = 1.85$），$\bar{x}_l = 2.70$（$s_l = 1.96$）となった．このことから，人間科学部の方が試験の点数が高かったと結論した．

4.4 考　　察

本研究では，JavaScript およびその実行環境と，PEN という 2 つの異なるプログラミング実行環境を用いて，異なる群に対してプログラミング教育を行ったことから，両者のプログラミングに対する理解度を比較することを試みた．その結果，自己評価および試験成績の観点から PEN を用いた人間科学部の方がより良い値であることが示唆された．このような結果から PEN がプログラミングの入門的な実行環境として優れていると考えてよいかどうかについて考察する．

まず，比較を試みた両学部の学生が，質的に同等であったかどうかという問題を検討する．両学部については，これまでも習熟度別クラス編成の研究において，受講前のコンピュータ利用経験やコンピュータ不安について比較を行ってきた[17),23)]．その中で，コンピュータリテラシの観点から見て，両学部の習熟度が著しく異なるという結果は得られていない．また，本研究で行った調査結果からも，コンピュータの利用経験やプログラミングの授業前に行った習熟度自己評価において，両学部に差があるという結果が得られなかった．こうしたことから，プログラミング学習に関して両学部を比較することは問題がないと結論した．

次に，授業の内容および方法が両学部で同じであったかという点が問題となる．本研究の比較結果をプログラミング言語とその実行環境の違いに起因すると結論づけるためには，プログラミング教育で用いたプログラミング言語とその実行環境が異なるだけで，他のことは両学部で同じであることが必要となる．特に，両学部ではそれぞれ別の教員が授業を担当しており，教員の差がどの程度結果に影響しているかが重要な問題である．この問題は，比較を試みた授業がコンピュータリテラシの向上を前提とする必修科目であり，厳密な授業進行の調整が現実的に難しかったことや，教員の影響の現れ方が予測しにくいことなどから，結果に一定量の影響を与えていることは否定できない．しかし，対象とした授業では，内容や教員の差を埋めるために，頻繁に情報交換を行い授業を進行させた．また，本研究では，各学部 3 名ずつの教員が授業を行った結果を分析しており，教員の教え方の差は学部間で比較する際には相殺されている．このようなことから，授業内容や教育方法に事実上の差はないと結論した．

さらに，比較に用いた指標の妥当性について検討する．プログラミングの授業後に行ったアンケート以外のアンケート項目については，これまでの研究[17),23)]で何度も利用しており，おおよそ同様の結果が得られている．これより，内容的に妥当であったと結論した．

プログラミング教育時の毎回の授業後に行ったアンケートの項目は，理解度の自己評価として，簡単な文法に関する理解度を尋ねたものであり，これをプログラミングの基礎的内容に関する理解度とするには，内容の妥当性にやや問題がある．今後はより精度の高い評価項目を考案する必要があるだろう．しかし，本研究で行ったのが初学者へのプログラミング教育であることを考慮すれば，的外れとはいえない．

以上のことから，PEN を用いた人間科学部の方が理解度が高かったということは，PEN のプログラミング学習での優位性を表している．JavaScript とその

実行環境を用いた文学部は人間科学部よりも授業回数が 1 回多かったが，心理的側面および実技的な側面から比較した 2 つの学部において理解度が同じであるという仮説が棄却されたことの意味は大きい．

また，本研究では，実際にプログラミングができるかどうかだけでなく，プログラミングがどれくらいできると思っているかという理解度の自己評価を指標として用いた．こうした心理的な指標は，プログラミングに対するポジティブなイメージにつながり，今後のコンピュータ学習に積極的に取り組む態度と関係していると考えることができる．本研究の結果は，PEN を用いた方がそうした側面でも良い効果をもたらすことを示している．

なお，本研究では性差に関して詳細に検討することはしなかった．これは性差の検討が本研究の主目的ではなかったからであるが，プログラミング学習と性差に関しては，先行研究[24] などにおいて検討されており，性差の検討は重要なテーマの 1 つとなるだろう．ただし，性差はさまざまな要因の連関によって現れると考えるべきであり，性差に取り組むにはそうした要因を適切な方法で測定し，背景に潜む要因間の関係性を十分に検討する必要がある．そのためには，厳密に計画された実験や調査が不可欠である．本研究の調査結果では，性差に関する有意差は認められなかったが，このことは必ずしも性差とプログラミング学習との関連性を否定するものではない．今後，さらなる吟味が必要であろう．

5. おわりに

本論文では，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 の実装についての紹介と評価を行った．PEN は基本的なプログラミングを短期間で習得することを目指し，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．PEN は，Web サイト[7] からダウンロードし，利用可能である．

いくつかの大学での授業実践のアンケート結果から，PEN は初学者におおむね好評で，高いモチベーションを持った感想も多く聞かれた．また，JavaScript とその実行環境を用いた授業との理解度の比較では，自己評価，試験成績の双方で PEN を用いたクラスが高くなり，プログラミングの入門教育環境としての PEN の有用性が示唆される結果が得られた．

現在，PEN は本論文でとりあげたほか，千里金蘭大学などの大学でも利用していただいている．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．

謝辞 PEN の授業実践および評価にご協力いただいた大阪工業大学の中西通雄教授，安留誠吾准教授，大阪大学の清川清准教授，外川直子氏，東京大学の小川剛史講師，京都ノートルダム女子大学の吉田智子准教授，鹿児島大学の山之上卓教授，大阪市立大学の豊田博俊氏，甲子園大学大学院の菅澤拓生氏，景村幸弘氏に感謝する．なお，本研究の一部は日本学術振興会科学研究費補助金（18500719）および，大阪学院大学研究助成金の補助による．

参 考 文 献

1) 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 (2006). http://www.ipsj.or.jp/12kyoiku/statement2006.pdf
2) 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 (2005). http://www.ipsj.or.jp/12kyoiku/proposal-20051029.pdf
3) 情報処理学会一般情報処理教育の実態に関する調査研究委員会：一般情報処理教育の実態に関する調査研究，文部省委嘱調査研究 (1992).
4) 原田悦子：文科系大学・学部における情報教育—その目的と問題，情報処理，Vol.41, No.3, pp.227–233 (2000).
5) 情報処理学会大学等における一般情報処理教育の在り方に関する調査研究委員会：大学等における一般情報処理教育の在り方に関する調査研究，文部科学省委嘱調査研究 (2002).
6) 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 (2003).
7) 初学者向けプログラミング教育環境 PEN Web ページ. http://www.media.osaka-cu.ac.jp/PEN/
8) 西田知博，中村亮太，山本武生，松浦敏雄：プログラミング環境 PEN—描画とファイル I/O 機能の実装，情処学情報教育シンポジウム SSS2006 論文集，pp.69–74 (2006).
9) Ueno, H.: An integrated knowledge-based intelligent programming environment for novice programmers, *Proc. IEEE Computer Software and Application Conf.*, pp.124–129 (1991).
10) Evangelidis, G., Dagdilelis, V., Satratzemi, M.

and Efopoulos, V.: X-Compiler: Yet another integrated novice programming environment, *Proc.IEEE Int'l Conf.Advanced Learning Technologies*, pp.166–169 (2001).
11) 軽野宏樹，木實真一，上林弥彦：ALAN-K プロジェクト：Squeak を活用した創造的な情報教育の試み，情報処理学会研究報告 2003-CE-69, pp.1–8 (2005).
12) 斐品正照，徳岡健一，河村一樹：構造化チャートを用いたアルゴリズム学習支援システム，情報処理学会論文誌，Vol.45, No.10, pp.2452–2467 (2004).
13) 兼宗　進，中谷多哉子，御手洗理英，福井眞吾，久野　靖：初中等教育におけるオブジェクト指向プログラミングの実践と評価，情報処理学会論文誌，Vol.44, No.SIG13, pp.58–71 (2003).
14) 長　慎也，甲斐宗徳，川合　晶，日野孝昭，前島真一，筧　捷彦：Nigari—Java 言語へも移行しやすい初心者向けプログラミング言語，情報処理学会研究報告 2003-CE-71, pp.13–20 (2003).
15) PEN を用いた授業アンケート結果集 (2006). http://www.s.osaka-gu.ac.jp/pen/enq/
16) 鳥居　稔，原田　章，中西通雄：教え方の違いが学生のコンピュータ不安や自己評価に与える影響，2003 PC カンファレンス論文集，CD-ROM (2003).
17) 原田　章，宮本友介，安留誠吾，中西通雄：クラス編成が習熟度自己評価の変化に与える影響，平成 17 年度情報処理研究集会講演論文集，CD-ROM (2005).
18) 大阪大学情報活用基礎受講前アンケート (2006). http://www.s.osaka-gu.ac.jp/pen/data/pretest2006.pdf
19) 大阪大学情報活用基礎中間アンケート (2006). http://www.s.osaka-gu.ac.jp/pen/data/midtest2006.pdf
20) 大阪大学情報活用基礎期末アンケート (2006). http://www.s.osaka-gu.ac.jp/pen/data/posttest2006.pdf
21) 原田　章，鳥居　稔：大学での一般情報処理教育における習熟度別クラス編成とコンピュータ不安，日本心理学会第 66 回大会発表論文集，p.1128 (2002).
22) 松浦敏雄，豊田博俊，原田　章，外川直子，小川剛史，清川　清，西田知博：文系学部を対象としたリテラシー科目におけるプログラミング教育—JavaScript を用いた実践例，平成 18 年度情報教育研究集会講演論文集，pp.197–200 (2006).
23) Nakanishi, M. and Harada, A.: Reorganizing computer literacy classes in the middle of a term, *Advanced Research in Computers and Communications in Education* (*2*), pp.507–514 (1999).
24) Beckwith, L. and Burnett, M.: Gender: An Important Factor in End-User Programming Environment?, *Proc. 2004 IEEE Symp. Visual Languages and Human Centric Computing*, pp.107–114 (2004).

(平成 18 年 11 月 30 日受付)
(平成 19 年 5 月 9 日採録)

西田　知博（正会員）

平成 3 年大阪大学基礎工学部情報工学科卒業．平成 8 年同大学院基礎工学研究科博士後期課程退学．同年同大学情報処理教育センター助手．平成 12 年大阪学院大学情報学部講師．情報教育およびヒューマンコンピュータインタラクションの研究に従事．ACM，電子情報通信学会各会員．

原田　　章（正会員）

平成 6 年大阪大学大学院人間科学研究科博士前期課程修了．同年同大学情報処理教育センター助手．平成 8 年同大学人間科学部助手．平成 15 年甲子園大学人間文化学部講師．平成 18 年甲子園短期大学准教授．情報教育におけるコンピュータ不安や習熟度に関する研究等に従事．日本心理学会，日本行動計量学会，日本認知科学会各会員．

中村　亮太（正会員）

平成 16 年大阪学院大学情報学部情報学科卒業．平成 18 年大阪市立大学大学院創造都市研究科修士課程修了．同年大阪府立泉北高等学校常勤講師．情報教育に関する研究に従事．

宮本　友介

平成 11 年大阪大学人間科学部卒業．平成 15 年同大学院人間科学研究科博士後期課程退学．現在，同研究科助教．情報処理教育・心理統計教育および数理認知心理学モデルの研究に従事．日本認知科学会，日本行動計量学会各会員．

松浦 敏雄（フェロー）

昭和 50 年大阪大学基礎工学部情報工学科卒業．昭和 54 年同大学院基礎工学研究科博士後期課程退学．同年同大学基礎工学部情報工学科助手．平成 4 年同大学情報処理教育センター助教授．平成 7 年大阪市立大学生活科学部教授．平成 8 年同大学術情報総合センター教授．平成 15 年同大学院創造都市研究科教授．博士（工学）．ACM, IEEE，電子情報通信学会各会員．