初学者向けプログラミング学習環境
PEN と教材に関する研究

提出先　　大阪大学大学院情報科学研究科

提出年月　2017 年 1 月

西 田　知 博

関連論文リスト

学術論文

1. 西田知博, 原田章, 中村亮太, 宮本友介, 松浦敏雄, “初学者用プログラミング学習環境 PEN の実装と評価”, 情報処理学会論文誌 第 48 巻 第 8 号, pp.2736–2747, 2007 年 8 月.
2. 西田知博, 原田章, 中西通雄, 松浦敏雄, “プログラミング入門教育における図形描画先行型のコースウェアが学習に与える影響”, 情報処理学会論文誌教育とコンピュータ (TCE) （採録決定）.

国際会議

3. Tomohiro Nishida, Akira Harada, Tomoko Yoshida, Ryota Nakamura, Michio Nakanishi, Hirotoshi Toyoda, Kota Abe, Hayato Ishibashi, Toshio Matsuura, “PEN: A Programming Environment for Novices — Overview and Practical Lessons —”, ED-Media 2008, pp.4755–4760, Jul.2008.
4. Tomohiro Nishida, Ryota Nakamura, Liu Lu, Chan Myae Thu and Toshio Matsuura, “Development of Learning Support Software and Educational Materials for Studying Measurement and Control by Programs”, ED-Media 2013, pp.108–114, June.2013. (`http://www.editlib.org/p/111940/`)
5. Tomohiro Nishida, Ryota Nakamura, Yuki Shuhara, Akira Harada, Michio Nakanishi, Toshio Matsuura, “A PROGRAMMING ENVIRONMENT FOR NOVICES WITH VISUAL BLOCK AND TEXTUAL INTERFACES”, Educational Alternatives, Vol. 14, pp.470–478, Sep. 2016. (`http://www.scientific-publications.net/en/article/1001315/`)

著書

6. 川合 慧, 高岡　詠子, 西田 知博, “計算事始め”, 放送大学教育振興会, 2013 年 3 月.

国内会議（査読付）

7. 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”, SSS2006 情報処理学会情報教育シンポジウム論文集, IPSJ Symposium Series Vol.2006, No.8, pp.69–74, 2006 年 8 月.

その他の発表論文リスト

学術論文

1. 都倉信樹, 西田知博, “コンピュータネットワークを用いた維持・管理の容易な教育支援システム”, 日本工学教育協会会誌, 第 44 巻第 3 号, pp.22–25, 1996 年 5 月.
2. 大坪正明, 西田知博, 辻野嘉宏, 都倉信樹, “日常時と実験時のマウス操作の比較”, 電子情報通信学会論文誌 (D-II), Vol.J81–d–II No.10, pp.2487–2489, 1998 年 10 月.
3. 齊藤明紀, 西田知博, 辻野嘉宏, 都倉信樹, “マウスドライバの改良によるポインティング精度改善について”, 情報処理学会論文誌, Vol.40 No.2, pp.405–413, 1999 年 2 月.
4. 齊藤明紀, 西田知博, 中西通雄, 安留誠吾, 馬場健一, 重弘裕二, 原田章, 山井成良, 松浦敏雄, “大規模教育用計算機システムにおける授業・運用支援システムの設計と実装”, 電子情報通信学会論文誌 (D-I), Vol.J84–D–I No.6, pp.956–965, 2001 年 6 月.
5. 長 慎也, 保福 やよい, 西田 知博, 兼宗 進, “De-gapper —プログラミング初学者の段階的な理解を支援するツール”, 情報処理学会論文誌 第 55 巻 1 号, pp.45–56, 2014 年 1 月.
6. 中西 渉, 辰己 丈夫, 西田 知博, “PenFlowchart を用いたフローチャートによるプログラミング学習の効果に対する評価”, 情報処理学会論文誌教育とコンピュータ (TCE) 第 1 巻 4 号, pp.75–82, 2015 年 12 月.

国際会議

7. Tomohiro Nishida, Akinori Saitoh, Yoshihiro Tsujino, Nobuki Tokura, “Lecture Supporting System by Using e-mail and WWW”, ACM SIGCSE ’96, pp.280–284, Feb.1996.
8. Tomohiro Nishida, Junichi Yahara, Kazutoshi Fujikawa, Hayato Ishibashi, Kota Abe, Toshio Matsuura, “Designing Simulator for Construction of a Virtual Computer System Using Arbitrary Levels of Abstraction”, ED-Media 2001, pp.1410–1411, Jun.2001.
9. Tomohiro Nishida, Junichi Yahara, Toshimitsu Masuzawa, Toshio Matsuura, “ECAS: A Computer Simulator that has the adjustable degree of abstraction of the observation according to the educational objectives, and its evaluation”, ED-Media 2003, pp.906–907, Jun.2003.

10. Tomohiro Nishida, Junichi Yahara, Toshimitsu Masuzawa, Toshio Matsuura, “Development and Evaluation of ECAS: A Computer Simulator with Adjustable Degree of Abstraction Based on Educational Objectives”, International Conference on Computers in Education 2003, pp.954–958, Dec.2003.
11. Tomohiro Nishida, Yukio Idosaka, Yayoi Hofuku, Susumu Kanemune, Yasushi Kuno, “New Methodology of Information Education with “Computer Science Unplugged” ”, Informatics Education - Supporting Computational Thinking, Lecture Notes in Computer Science,Vol.5090, pp.241–252, Jul.2008.
12. Tomohiro Nishida, Susumu Kanemune, Yukio Idosaka, Mitaro Namiki, Tim Bell and Yasushi Kuno, “A CS Unplugged Design Pattern”, ACM SIGCSE 2009, pp.231–235, Mar.2009.
13. Yayoi Hofuku, Shinya Cho, Tomohiro Nishida and Susumu Kanemune, “Why is programming difficult? Proposal for learning programming in “small steps” and a prototype tool for detecting “gaps” ”, Informatics in Schools: Local Proceedings of the 6th International Conference ISSEP 2013 – Selected Papers, pp.13–24, Feb. 2013.

著書

14. 西田知博, 川本芳久, “CGI 入門講座”, オーム社, 1997 年 6 月.
15. 久野靖，辰己丈夫，大岩元，小原格，兼宗進，佐藤義弘，橘孝博，中野由章，西田知博, 半田亨, “情報科教育法 改訂 3 版”, オーム社, 2016 年 8 月.

国内会議（査読付）

16. 西田知博, 澤村雅之, 中駄康博, 都倉信樹, “ペンを用いた一般教科向け授業支援システムとそのインタフェース評価”, SSS2000 情報処理学会情報教育シンポジウム論文集, IPSJ Symposium Series Vol.2000, No.9, pp.83–86, 2000 年 8 月.
17. 西田知博, 矢原潤一, 増澤利光, 松浦敏雄, “目的に応じて観察の抽象度が変更可能な計算機シミュレータ ECAS を用いた教育とその評価”, SSS2002 情報処理学会情報教育シンポジウム論文集, IPSJ Symposium Series Vol.2002, No.12, pp.245–252, 2002 年 8 月.

18. 西田知博, 渡辺博芳, 中西通雄, 神沼靖子, 武井惠雄, “大学における一般情報教育のためのコンポーネント構成型教材情報システムと現代社会」の開発とその展開”, SSS2004 情報処理学会情報教育シンポジウム論文集, IPSJ Symposium Series Vol.2004, No.9, pp.95–102, 2004 年 8 月.
19. 西田知博, 井戸坂幸男, 兼宗進, 久野靖, “コンピュータサイエンスアンプラグドの分析と CS アンプラグドデザインパターンの提案”, SSS2008 情報処理学会情報教育シンポジウム論文集, IPSJ Symposium Series Vol.2008, No.6, pp.179–186, 2008 年 8 月.
20. 保福 やよい, 西田 知博, 長 慎也, 兼宗 進, “なぜプログラミングは難しいのか? 繰り返しの理解構造と C の教科書分析からのアプローチ”, SSS2012 情報処理学会情報教育シンポジウム論文集, IPSJ Symposium Series Vol.2012, No.4, pp.177–180, 2012 年 8 月.
21. 中野 由章，谷 聖一，筧 捷彦，村井 純，植原 啓介，中山 泰一，伊藤 一成，角田 博保，久野 靖，佐久間 拓也，鈴木 貢，辰己 丈夫，永松 礼夫，西田 知博，松永 賢次，山崎 浩二, “「大学情報入試全国模擬試験」の実施と評価”, SSS2014 情報処理学会情報教育シンポジウム論文集, IPSJ Symposium Series, Vol.2014, No.2, pp.11–17, 2014 年 8 月.
22. 中野 由章，久野 靖，佐久間 拓也，谷 聖一，筧 捷彦，村井 純，植原 啓介，中山 泰一，伊藤 一成，角田 博保，鈴木 貢，辰己 丈夫，永松 礼夫，西田 知博，松永 賢次，山崎 浩二,
“大学情報入試の必要性と情報入試研究会の活動”, 情報処理学会第 57 回プログラミングシンポジウム予稿集, pp.155–169, 2016 年 1 月.
23. 谷 聖一 , 佐久間 拓也 , 筧 捷彦 , 村井 純 , 植原 啓介; 中野, 由章 , 中山 泰一 , 伊藤 一成 , 角田 博保 , 久野 靖 , 鈴木 貢 , 辰己 丈夫 , 永松 礼夫 , 西田 知博 , 松永 賢次 , 山崎 浩二, “「第 3 回・第 4 回大学情報入試全国模擬試験」の実施と評価”, SSS2016 情報処理学会情報教育シンポジウム論文集, IPSJ Symposium Series, Vol.2016, pp.7–14, 2016 年 8 月.

内容梗概

2020 年度からの小学校でのプログラミング必修化が決まり，小学校から高等学校まで広くプログラミング学習が行われることとなった．この背景には，IT 人材の不足に危機感があり，内閣の日本再興戦略に小学校からプログラミング教育等の IT 教育を推進することが明記されたことにある．すべての児童・生徒がプログラムを作成することを職業とすることはないが，コンピュータに代表される情報機器を単に「情報を扱う便利な道具」として扱うのではなく，プログラミングに触れ，「情報処理の理解」を深めることは，どのような職業に就くとしても重要になると考えられる．

そこで本研究では，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た．さらに，高校生や大学生の初学者を主たる対象とし，自分の知識状態や考え方を明確にする力を養うことも目標とした．これら目標を達成するために，本研究ではコードを書くことにより，制御構造等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 の実装について述べ，その評価を行う．また，対象を中学生に広げるため，PEN を利用した「プログラムによる計測・制御」を学ぶための学習支援ソフトウェアおよびその教材の開発について述べる．また，プログラミングの教材として，図形描画を伴った演習が学習者にどのような効果を与えるかを評価した．

PEN では，大学入試センター等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 は初学者に概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．

「プログラムによる計測・制御」を学ぶための学習支援ソフトウェアは，PEN に任意の関数を付加できるプラグイン機能を追加し，これを利用して，ハードウェア (Arduino) を制御するための新たな関数群を追加することにより実現した．また，ハードウェアを接続していなくてもプログラムの動作が確認できるシミュレータを用意した．また，この学習支援ソフトウェアを利用して，「プログラムによる計測・制御」を学ぶための教材を提案し，高校生を対象として授業実践を行った．その結果，ハードウェア制御に関し，短時間でも一定の理解が可能であるという結果が得られた．

プログラミング学習教材における図形描画課題の効果に関しては，情報科学を専門としない大学 1 年生の情報リテラシー科目の中で評価を実施した．キーボードから数字を読み込んで計算結果をディスプレイ画面に出力するような例題から始めて逐次・条件分岐・繰り返しを学び，最後に図形を描画するコースウェアと，図形描画の例題から始めるタイプのコースウェアを用意し，その比較を行った．2011 年度から 4 年間の授業におけるアンケートおよび試験成績を統計的に分析した結果，図形描画を伴う例題を扱う方が，繰り返しのようなつまずきやすい学習内容でも，理解度や楽しさを下げることなく学習できていることがわかった．

目次

図目次

表目次

第 1 章

序論

1.1 はじめに

かつては大学が中心であった情報教育は，1999 年の学習指導要領改訂 [1] で 2003 年に高等学校教科「情報」が設置されたことを始めとして，初等中等教育においても本格的に行われるようになった．初等中等教育における情報教育の目標は，文部科学省の協力者会議により「情報活用の実践力」,「情報の科学的理解」,「情報社会に参画する態度」の 3 つの観点にまとめられている [2]．当初は高等学校ではこれらそれぞれを中心とした 3 つの科目が設置されたが，選択必履修であったため,「情報活用の実践力」を中心とした科目が多くの学校で選択されることとなり，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」を深める教育とはなっていない傾向にあった．2009 年の指導要領の改訂 [3] により,「情報の科学的理解」,「情報社会に参画する態度」をそれぞれ中心とした「情報の科学」と「社会と情報」の 2 科目に再編され，選択必履修となったが，多くの学校で「社会と情報」が選択されている．しかし，2017 年に改訂される予定の次期指導要領では共通必履修科目「情報 I」が設置されることとなり [4],「情報の科学的理解」を目標とする内容がすべての学校で等しく教育される環境が整いつつある．

また，2020 年度からの小学校でのプログラミング必修化が決まり，初等中等教育において広くプログラミング学習が行われることとなった [5]．この背景には，IT 人材の不足に危機感があり，内閣の日本再興戦略 [6] に小学校からプログラミング教育等の IT 教育を推進することが明記されたことにある．ただし，すべての子ども達がプログラムを作成する技術者になるわけではないため，文部科学省の有識者会

議の議論の取りまとめ [5] では，将来どのような職業に就くとしても，時代を超えて普遍的に求められる「プログラミング的思考」を育むようなプログラミング教育を求めている．また，便利さの裏側でどのような仕組みが機能しているのかについて思いを巡らせ，コンピュータが「魔法の箱」ではなく，プログラミングを通じて人間の意図した処理を行わせることができるものであることを理解してもらえるような教育も求めている．このことは，2005 年に情報処理学会 情報処理教育委員会が出した「日本の情報教育・情報処理教育に関する提言 2005」[7] において，プログラミングを始めとした『「手順的な自動処理」の理解』の重要性を訴えていたこととも符合している．

本論文の端緒は，この提言に先駆け，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，比較的短い学習時間でこの目標を達成するためのプログラミング教育環境の検討を 2003 年から始めた [8] ことにある．その後，高校や大学などでの教育に用いることを想定した，初学者教育用プログラミング環境 **PEN**(Programming Environment for Novices) を 2005 年に構築した [9, 10]．また，2008 年に改訂された中学校学習指導要領 [11] によって技術・家庭科の技術分野で必履修となった「プログラムによる計測・制御」の学習支援のために，PEN を用いてハードウェア (Arduino[12]) を制御する機能とそのシミュレータを提供した [13]．また，プログラミング環境だけではなく教材の重要性に着目し，図形描画を伴った演習において，教材の違いがプログラミング初学者の学習にどのような効果を与えるかを評価した [14, 15].

1.2　初学者用プログラミング環境

前節で述べた小学校からのプログラミング教育の必修化の流れもあり，初学者用プログラミング環境の注目度は高くなっている．日本においては特に，Scratch[16, 17]，ドリトル [18, 19]，ビスケット [20] などが広く用いられている．

Scratch はブロックの組み立てでプログラムを作成するビジュアルプログラミング環境で，文法エラーが生じず，文字入力に慣れていない子どもでもプログラム作成をすることができる．また，メインルーチンのないスクリプトの並列実行で画面上に描いたオブジェクトなどを動かすことができ，子ども達の創造性を高めることができるプログラミング環境である．

ドリトルはオブジェクト指向言語の初学者用プログラミング環境で，日本語表記によりタートルなどの

オブジェクトに命令を送る形でプログラムを動作させる．プログラムは文字入力して作成するが，少ない命令で画面に表示されるオブジェクトを動作させることができ，学校で利用できる 1 時間程度の入門コースウェアも整備されている．また，画面上のオブジェクトの代わりにハードウェアを制御することができ，幅広い対象のプログラミングが可能である．

ビスケットは「コンピュータは粘土である」という発想の元，絵を描いて，その変化のルールを定義することにより，アニメーションやゲームの作成ができる．ルールの定義は「メガネ」の中に記述され，左目の絵にマッチすれば，右目の絵に変化するという形でプログラムを作成することができる．手続き型のプログラム作成とは異なるが，ルールを順に追加することにより子どもでもゲームのような複雑性を持つプログラムを作成することができる．

本論文の研究は，前述の通り 2003 年に始まった．当時でも，初学者用のプログラミング環境に関する研究は多く行われていた．たとえば，INTELLITUTOR[21] 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[22] は，プログラミング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った．

また，日本の教育現場で実践報告がなされていた初学者用のプログラミング環境は，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 [23] 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かった．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，タイル等の 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ものであった．

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[24] があった．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，本研究で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，最初に開発するプログラミング環境では，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，すでにドリトルによる実践，評価が行われており，その他にも Nigari[25] などがあった．オブジェクト指向の考え方は極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．

本論文の研究では，高校生や大学生でのプログラミング入門教育を念頭に置いているため，初学者に対して制御構造を意識する手続き的なプログラミング教育が必要であると考えた．そこで PEN では，手続き型で，理解が容易な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．

1.3　本論文の構成

本論文では，第 2 章で作成したプログラミング環境 PEN の紹介とその評価を行う．ここでは，PEN 開発時に行った初学者用プログラミング学習環境への要求の分析と，それに基づいて開発したプログラミング環境 PEN の概要を説明する．また，PEN が活用されている授業実践の紹介と，大阪大学の情報リテラシー科目において，JavaScript を用いたクラスと理解度を比較した結果について紹介する．

第 3 章では，PEN を利用した「プログラムによる計測・制御」を学ぶための学習支援ソフトウェアおよびその教材の開発について述べる．ここでは，それを実現するために実装した PEN のプラグイン機能と，追加したハードウェア (Arduino) シミュレータについて説明する．また，それを利用し，プログラミングの初学者でも計測制御を使った学習ができるような入門用の教材を紹介する．さらに，これを用いて高校生を対象とした授業を行った結果について紹介する．

第 4 章では初学者プログラミングの教材として，図形描画を伴った演習が学習者にどのような効果を与えるかを評価する．ここでは，大阪大学の情報リテラシー科目におけるプログラミング入門授業において，図形描画を主としたコースウェアと従来型のコースウェアとの学生のモチベーション，理解度，成績の差について 4 年間にわたり調査した結果について述べる．

第 5 章ではまとめと，PEN に関連するプログラミング環境の紹介を行う．

第 2 章

プログラミング環境 PEN の構築と評価

2.1 はじめに

第 1 章で述べたように，本研究では，プログラミング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し，それ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．本章では，PEN を構築する過程で検討した初学者用プログラミング学習環境への要求の考察，PEN の実装と評価を紹介する．

2.2 初学者用プログラミング学習環境への要求

この節では，従来のプログラミング学習で初学者が直面する問題を挙げ，本研究で考えるプログラミング教育の目標を明確にしたあと，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．

2.2.1 プログラミング学習の難しさ

これまで大学や高校においてさまざまなプログラミングの入門授業を行ってきたが，その中で初期段階でつまずく学習者を多く見てきた．その原因の 1 つは，プログラミング環境における言語の問題にあると考える．多くのプログラミング言語は英語をベースとしたキーワードを用いるため，日本人の初学者にとっては，その意味を理解していなかったり，スペルがあやふやになったりすることが，文法エラーを起こす要因となる．また，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．その結果，文法エラーの修正ばかりに気を取られ，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くなる．このようなプログラミング環境における言語の問題への対応の必要性は，文献 [26] など，古くから指摘されている．

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，間違い箇所をみつけるためには，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．

以上に挙げた問題を，ここでは次の 3 つに整理した．

P1:**プログラム言語のとっつきにくさ**　英語を基本とした表現であるためキーワードが憶えずらく問題 P2 の要因にもなる．また，プログラムの理解が進まないため，問題 P3 の要因ともなる．さらに，エラーメッセージが英語であるため，問題 P2 の解決を妨げる．

P2:**文法エラーの多発**　問題 P1 とタイプミスに起因して，文法エラーが多く発生する．文末の区切り記号を忘れた場合など，1 つの文法エラーでそれ以降の多くの箇所でエラーが発生してメッセージが表示され，それを理解していないため自信をなくす学習者も多い．

P3:**論理エラーへの対処の困難さ**　エラーメッセージが出ない論理エラーはプログラムをトレースし，変数の変化などを観察する必要があるが，一般的なプログラミング環境に付属するデバッガは初学者には複雑で使いづらい．

2.2.2　「プログラミング」入門教育の目標

ここで想定している「プログラミング入門」は，文献 [27] 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 [28] に示されている通りで，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，自らの考え方を明確にする力を養うことである．また，情報教育という観点では，「手を動かす」プログラミングにより静的な形で記述された「情報」を「実際に対象として動かせるモノ」として体得する過程を通じて，「情報」に関する知識をこれまでどれだけ得ており，どのような知識を得ようとしているのか（知識状態）を明確に意識させることができる．加えて，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 [29]．このように考えることは，文献 [7] 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中で広くプログラミング教育を行ってもらうためには，その一部として組み込み可能なように短時間でも修得が可能なものとする必要がある．

以上のことよりここでの「プログラミング」入門教育の目標を，以下のように定める．

T1:**プログラミング体験にもとづいてコンピュータの本質を理解する**

T2:**短時間でプログラミングの基本を習得する**

2.2.3　学習環境に必要な要件

2.2.1 節の問題を解決し，2.2.2 節で示した「プログラミング」入門の目標を，達成するためには，プログラミング学習環境には以下の要件が必要である．

R1:**読みやすく記述可能なプログラム言語の提供**　問題 P1，すなわちプログラム言語のとっつきにくさを

解消し，目標 T1,T2 を達成するためには，まずプログラムを読めるということが必要である．このため，前提知識がほとんどなくても読めるような，母国語（日本語）を使うなどした，分かりやすい言語を用意する必要がある．ただし，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，擬似コードではなく，実行可能な処理系を備えた環境でなければいけない．

R2:**プログラム作成を支援する機能の提供**　問題 P2 を解消する，すなわち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ず，目標 T2 にあるように短時間でプログラミングの基本を修得するためには，プログラムの作成を支援する機能を用意する必要がある．

R3:**プログラムの実行状況を観察できる機能の提供**　問題 P3，すなわち論理エラーへの対処の困難さを解消し，目標 T1 にあるようにプログラミングを通じてコンピュータの本質を理解するためには，プログラムのどこが実行されているかや変数の値を容易に観察できるようにすることにより，動作をトレースできる機能が必要である．

2.3　PEN の実装

2.2.3 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 2.1 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等をもつ．

2.3.1　プログラミング言語

要件 R1 を満たす，すなわち読みやすく記述可能なプログラム言語として，本研究では，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**[30] 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るからである．拡張した点は，以下の通りである．

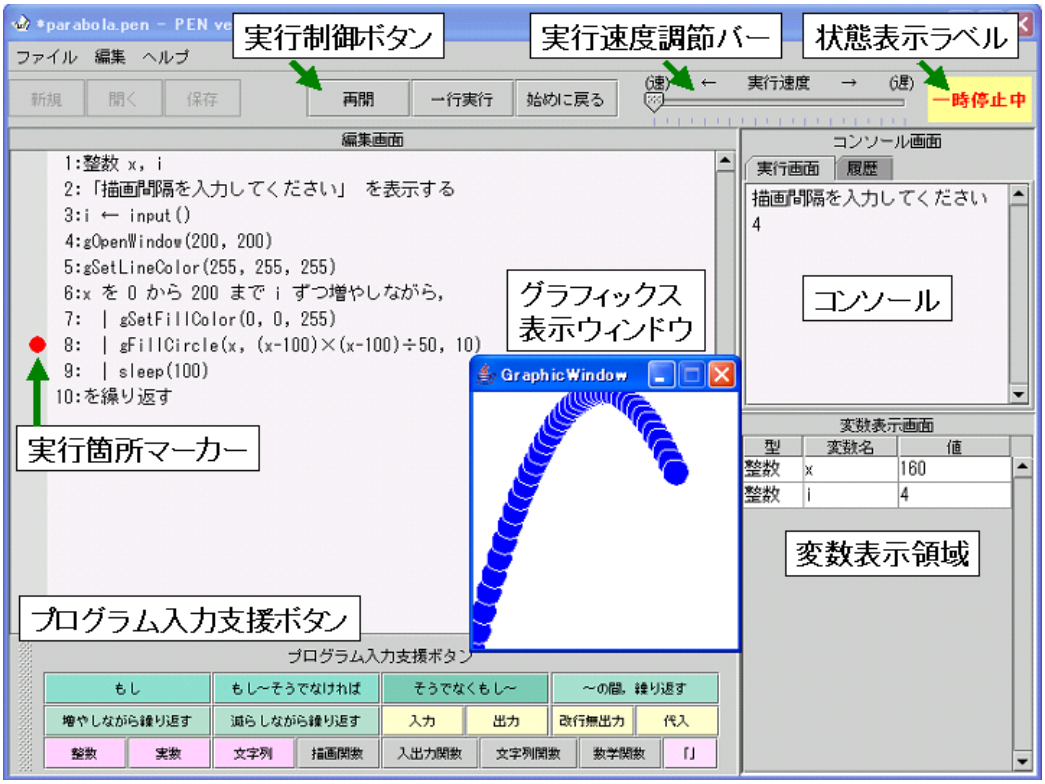

図 2.1　プログラミング環境 PEN

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，繰り返しからの脱出，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 [31, 32]

   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．グラフィクス関数は英語名の他に一部を除いて日本語名の関数を用意した．

xDNCL の言語仕様を付録 A に示す．xDNCL の言語マニュアルおよび，図形描画マニュアルは PEN の配布キット [31] に含まれ，授業資料としても配布されている [33]．

```
 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 2.2　xDNCL の記述例（BMI 算出）

図 2.2 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同じである．図 2.2 からわかるように xDNCL のプログラムを読むのは容易である．

DNCL（および xDNCL）では，言語の意味をほとんど説明していない．このため，「繰り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．

2.3.2　プログラム入力支援機能

ここでは，要件 R2 を満たす，すなわちプログラム作成を支援するために提供する機能について述べる．

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．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．

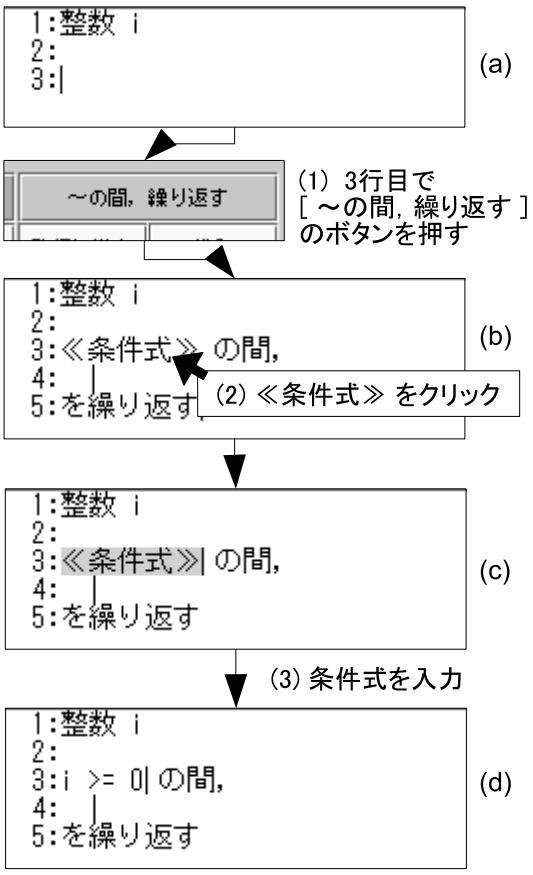

図 2.3　入力支援ボタンによるプログラミング例

キー入力の操作を減らし，補助するための機能として，図 2.1 の下部に配置している**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 2.3).

1. 図 2.3 (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 2.3 (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 (この部分全体で一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 2.3 (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等に書き換えて，さらに 4 行目以降に命令を挿入していくことにより，プログラムを作成することが可能である．

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 2.4).

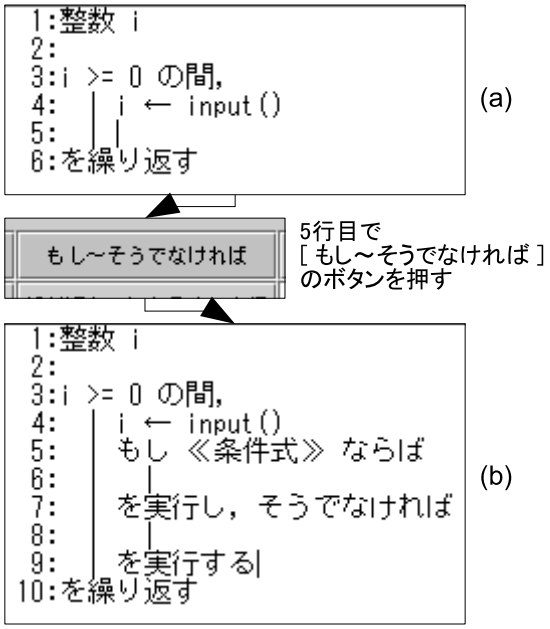

図 2.4　インデントの自動挿入

図 2.4 (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 2.4 (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．

2.3.3　プログラムの実行状態表示機能

ここでは，要件 R3 を満たす，すなわちプログラムの実行状況を観察するために提供する機能について述べる．プログラムの実行は図 2.1 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**実行箇所マーカー**で常に明示している．**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．

実行状態表示ラベル

図 2.1 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 2.1 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，一目でわかる．

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 2.1 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．

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 2.1 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

表 2.1　実行状態の一覧

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．

箇所マーカー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ーが移り，実行状態が「一時停止」となる．

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 2.1 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．

2.4　評価

2.4.1　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．著者は，大阪学院大学において 2005 年より 1 年生前期開講のプログラミング導入科目において，PEN を用いた教育を行っている．

2005, 2006 年度の PEN を用いた大学での授業実践の一覧を表 2.2 に示す．全て半期の授業であり，回数欄は PEN を用いた授業回数[*1]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 [34]．

[*1] 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，概ね講義 1 コマと演習 1 コマを行っている．

表 2.2　大学での PEN を用いた授業実践の一覧 (2005,2006)

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

目標 T1 として「プログラミング体験にもとづいてコンピュータの本質を理解する」を挙げていたが，

> 自分でプログラムをつくることで，コンピューターを少し身近に感じることができるようになりました．（大阪大学 2006 年度）

> 今までは自分がコンピュータに指示されっぱなしだったけど，プログラミングでは自分がコンピュータに指示することから始まったので，コンピュータを使っているという実感がわいた．（大阪大学 2006 年度）

など，コンピュータの本質の理解につながっていることが伺える結果が得られた．また，このアンケート結果から，問題点 P1 とした「プログラム言語のとっつきにくさ」に関し，

> まず，「日本語」で表記できる部分があるということです．パッと見た感じでも意味をとらえやすいし，落ち着いて解釈することができると思います．逆に他の英数字・記号ばかりの言語だと，パッと見ても意味がすぐわからなかったりするし，混乱するととまりません．（ノートルダム女子大学 2005 年度）

などの回答が得られ，xDNCL を用いたプログラミングが，プログラム言語のとっつきにくさの解消につながっていることが伺える．

また，目標 T2 として「短時間でプログラミングの基本を習得する」を挙げていたが，2005 年度の大阪大学の実践では 3 コマでプログラミングの基本を学ぶ授業を行った．その成果について，JavaScript とその実行環境を用いた授業との比較により評価した．この結果については，次節以降で詳述する．

高等学校においては大阪学院大学高等学校において 2007 年度から 2013 年度まで，3 年生対象とした選択のプログラミング授業を PEN により行った．また，2014 年度からは 2 年生の希望者を対象として「情報の科学」の授業を担当し，この中でも PEN を用いたプログラミング演習を行なっている [35].

この他にも，2013 年度から放送大学においての講義「計算事始め」を分担し，この中でも PEN を用いた授業を行なっている [36].

2.4.2 JavaScript との比較

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 [37, 38]．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（付録 B）.

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おこないながら，「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 [37, 39].

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．

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[*2] 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 回を割り当てた[*3]．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[*4]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おこなう形式で構成し，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・繰り返し）
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挙げる．なお，文学部 (JavaScript) でもほぼ同様の内容を扱っている [40].

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 をおこな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 2.5(a))

第２回 制御構造（条件分岐・繰り返し）

入力・演算・出力 の処理の流れに条件分岐と繰り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 2.5(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，図形描画をおこなうプログラムを作成した．また，繰り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 2.5(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

---

*2 OS: Vine Linux Educational Edition 2.0

*3 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．

*4 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．

(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 2.5　練習問題の例

用いてフォームへの入力を処理する問題を扱った．

2.4.3　結果

調査対象

プログラミング教育の成果についての評価は 1 年生（人間科学部:142 名，文学部:179 名）を対象とした．ただし，必要な指標がすべて揃っている必要があるため，プログラミングに関する授業すべてに出席した学生のみを分析対象とした．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 ($\chi^2(1) = 0.535$).

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 (付録 B.1 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 (付録 B.2 の質問 2 参照) から，表計算 (7 項目)，プ

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 2.6　コンピュータ利用経験を問う項目の代表例

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 2.7　習熟度自己評価を問う項目の代表例

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共に学習済みの項目のみを分析対象としたからである．

図 2.6，2.7 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あらわしたものである．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 までの値を取り，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 2.7 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 2.3 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 [37] 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

表 2.3　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

	$\bar{x}_h$	$\bar{x}_l$	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 $df_1 = 1, df_2 = 190$

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．

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．授業後に行ったアンケート項目を付録 C に示す．

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．

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」，「繰り返し (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 2.4 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 2.5 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

表 2.4　理解度 (自己評価) に関する分散分析表 (人，文とも 3 回目)

	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	1081.001	194		

表 2.5　理解度 (自己評価) に関する分散分析表 (人：3 回目，文：4 回目)

	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	1579.427	194		

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．

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みるため，期末試験（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上でおこなった．

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

表 2.6　理解度 (試験成績) の分散分析表

	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	2504.000	194		

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もらえることになった．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 2.6 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．

2.4.4　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よ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 [38, 41]．その中で，コンピュータリテラシーの観点から見て，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付けるためには，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ー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づらいことなどから，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 [38, 41] 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よ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 [42] 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ひとつとなるだろう．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．

2.5 結論

本章では，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（目標 T1 を満たす）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（目標 T2）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 [31] からダウンロードし，利用可能であり，実際に高校や大学の授業で用いられている．アンケートから初学者には概ね好評で，目標 T1 に挙げたコンピュータの本質の理解につながっていることが伺える回答が得られた．また，目標 T2 に挙げた短時間でプログラミングの基本を習得が行えているかという評価を行うために，3 コマで PEN を用いたプログラミングの基本を学ぶ授業と，JavaScript とその実行環境を用いた授業との理解度の比較した．その結果，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，既存の言語に比べて，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．

第 3 章

プログラムによる計測と制御の仕組みを学ぶための学習支援ソフトウェアと教材開発

3.1 はじめに

2012 年より施行された，中学校の学習指導要領 [11] において，技術・家庭科の技術分野において「プログラムによる計測と制御」が必履修に指定された．それ以前の学習指導要領において「計測と制御」は学校選択項目であったため，中学校の教育現場で扱われることは少なかった．このため新しい指導要領に準拠した教材の充実が急務であった．

本研究では「プログラムによる計測・制御」を学ぶための学習支援ソフトウェアおよびその教材の開発を行ない，その有効性を確認することを目的としている．この目的を達成するため，まずプログラミング学習環境 PEN[9, 10, 31, 32] に任意の関数を付加できるプラグイン機能を追加した．これを利用して，ハードウェア (Arduino[12]) を制御するための新たな関数群とシミュレータを用意し，「プログラムによる計測・制御」を学ぶための学習支援ソフトウェアを作成した．また，本学習支援ソフトウェアを利用して，「プログラムによる計測・制御」を学ぶためのいくつかの教材を用意し，高等学校において授業実践を行った．

3.2　関連研究

ロボットの制御やタートルグラフィックスを応用したゲーム作りなどを通じてコンピュータの仕組みを体験的に学ぶ授業がいくつかある．

西ヶ谷らは 2006 年 6 月から 11 月まで，中学校 2 年生 (120 名) を対象に 2 軸を使ったロボットを製作し，コンピュータで制御する授業を行った．さらに，2007 年 4 月から 12 月まで中学校 3 年生 (120 名) に 3 軸を使ったロボットの製作と制御の学習を行った [43]．最初から 3 軸制御ロボットの学習に入らず，2 年生の授業で簡単な 2 軸ロボットの製作と制御の学習を行い，3 年生で 3 軸制御の学習を無理なく行えるようにした．さらに，西ヶ谷らは 3 年生 117 名を対象として 2008 年 4 月から 11 月までの期間，自律型 3 モータ制御ロボットの授業を行った [44]．授業でロボットの先端につけたリミットスイッチを利用してシーケンス制御を体験させ，ロボットを目的に合わせて動かす制御の基本を学ばせ，その後，光センサと距離センサを用い，計測結果をロボットの制御に使うという流れで行った．

井戸坂らは松阪市立飯南中学校において，2009 年度に自律型制御ロボットを使った授業を実施した [45]．授業のねらいは，総合学習のねらいに沿って，生徒の自主的な学習活動を重視し，自律型制御ロボットの仕組みや使い方を自分たちで調べ，実際に動作させる方法を学ぶことである．

以上の授業実践例では，ハードウェアに関する高度な知識を有する教員が求められる．また，授業で利用するロボットなどの機器を簡便に扱えるような工夫がなされており，そのため，かえって学習者には，プログラムによる計測と制御の基本的な仕組みが見えにくくなっている．さらに，ハードウェアの組み立てで失敗することがあることも問題点の一つであると考える．

3.3　提案するソフトウェア

本研究では初学者向けプログラミング学習環境 PEN にハードウェアを制御する機能を追加することで，計測と制御の仕組みを容易に学べる学習環境を提供する．ここでは，本研究で提案するソフトウェアの概要を述べ，プログラミング環境 PEN へのプラグイン機能の利用方法を示す．

3.3.1　提案するソフトウェアの概要

「プログラムによる計測・制御」を学ぶ際に，ハードウェアを組み立てることから始める方法もある．しかし，ハードウェアを組み立てるには，多くの時間を必要とし，また必ずしも正常動作するものを作成できるとは限らない．そこで，本研究では生徒らに自分でハードウェアを組み立てることをさせず，組立て済の機器を利用することにし，プログラムによって機器をいかに制御するかを学ぶことに力点を置いた．

提案するソフトウェアの概要を図 3.1 に示す．図 3.1 の左は，ハードウェア制御機能をプラグインで付加した PEN であり，右側のハードウェア (Arduino) を PEN 上のプログラムで直接制御する．PEN を使って Arduino を制御する際，予め Arduino 側に，通信プログラムを書きこんでおき，PEN からの命令で動作させるモデルを採用した．PEN のプログラムはこの通信プログラムと対話しながら，ハードウェア制御を行う．また，実際のハードウェアを接続していなくても制御用のプログラムが動作するか確認できるように Arduino のシミュレータを用意し，制御ポートを変更するだけで使えるようにした．

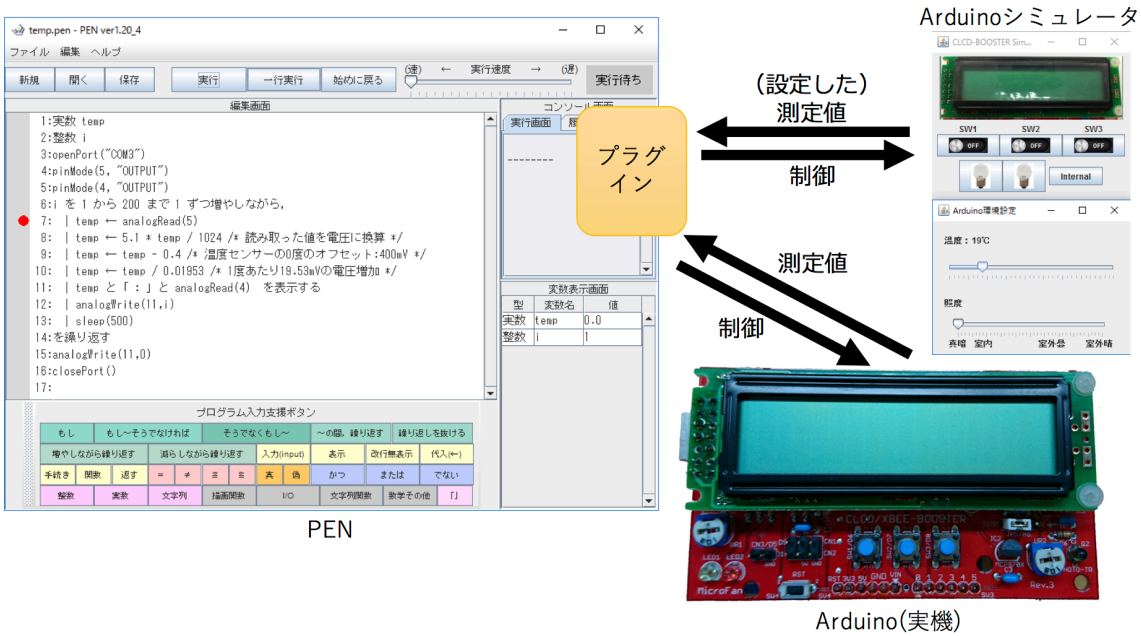

図 3.1　ハードウェア制御学習ソフトウェアの概要

3.3.2　PEN の拡張

これまで，PEN に関数を追加するには，構文解析を行うためのプログラムを修正し，インタプリタにその関数で処理する命令を記述し，PEN 全体をリコンパイルしなければならなかった．この作業は PEN の開発に携わっている者以外には困難であった．今回，PEN を再コンパイルせず容易に関数を組み込むための仕組みとして「プラグイン機能」を追加した．

このプラグイン機能を用いて PEN に新しい関数を追加する場合，新たに作成した Java プログラムの class ファイルを所定のディレクトリに置き，PEN で使用する関数名と Java のクラス名・メソッド名およびその引数の型の対応を，関数対応表 (functionTable.ini) に記述するだけで良い．

3.4　制御対象のハードウェア

既に述べたように，本学習支援ソフトウェアが対象とするハードウェアは Arduino であるが，Arduino に加えて，標準的な入出力装置群 (LED, スイッチ，温度センサ，光センサ) を一括して利用できるシールドとして CLCD-BOOSTER を利用することにした（図 3.2）．

LED1(緑色) は Arduino のデジタル出力 13 番に接続されている．プログラムによって点灯・消灯を制御することができる．LED2(赤色) は Arduino のデジタル出力，アナログ出力 10 番と接続されている．スイッチ 1〜3 は Arduino のデジタル入力 06 番〜08 番に接続されている．光センサは Arduino のアナログ入力 04 番と接続されている．

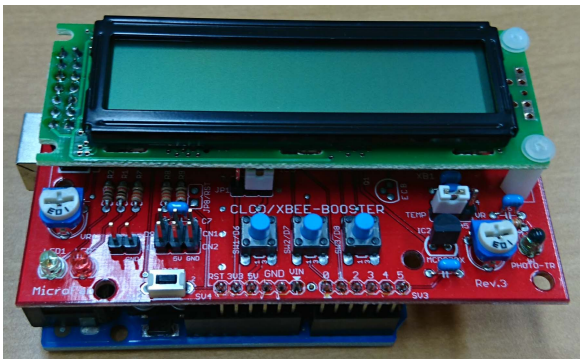

図 3.2　CLCD-BOOSTER

3.5 Arduino を制御するための関数群

PEN から Arduino を制御するために，以下の関数を追加した．

1. openPort(port)

   Arduino と通信するためのポートを開く．

   ［パラメータ］ port：通信のためのシリアルポート（Windows では`"COMx"`）

2. closePort()

   開いたポートを閉じる．

3. pinMode(pin, value)

   指定したピンを入力に用いるのか出力に用いるかを設定する．

   ［パラメータ］ pin：ピンの番号, value：`"INPUT"` または `"OUTPUT"`

   ［戻り値］ なし

4. digitalRead (pin)

   指定したピンの値を読み取る．その結果は 0 または 1 となる．

   ［パラメータ］ pin：読み取るピンの番号

   ［戻り値］ 0 または 1

5. digitalWrite (pin, value)

   指定したピンにデジタル出力を設定する関数である．value として 0 を指定した場合は出力 pin に低電圧 (Low : 0V) が， 1 を指定した場合は高電圧 (High : 5V) が出力される．

   ［パラメータ］ pin：ピンの番号, value：0 または 1

   ［戻り値］ なし

6. analogRead (pin)

   指定したアナログピンから値を読み取る．Arduino ボードは 6 チャネルの 10 ビット AD (analog to digital) コンバータを搭載している．これにより，0 から 5V の電圧を 0 から 1023 の数値に変換できる．

   ［パラメータ］ pin：読み取るピンの番号

［戻り値］ 0〜1023 の整数

7. analogWrite (pin, value)

   指定したピンからアナログ値を出力する．この機能は LED の明るさを変えたいときや，モータの回転スピードを調整したいときに使える．

   ［パラメータ］ pin：出力ピンの番号, value：0〜255 の整数

   ［戻り値］ なし

これらの関数を使い，PEN 上で Arduino を制御するプログラム例を図 3.3 に示す．この例は，赤色 LED を 7 秒間点灯させるプログラムである．

```
1: openPort("COM3")
2: pinMode(10, "OUTPUT")
3: digitalWrite(10, 1)
4: sleep(7000)
5: digitalWrite(10, 0)
6: closePort( )
```

図 3.3　制御プログラムの記述例

3.5.1　Arduino シミュレータ

Arduino シミュレータは，3.5 節で紹介した `openPort()` の引数を`"SIM"` と変更すれば使用でき，図 3.4 のようなウィンドウが表示される．

シミュレータを使うことにより，Arduino を接続しなくてもプログラム動作の確認ができ，学習者が多い場合でもハードウェアをシェアして演習を進めることが可能になる．また，図 3.4 上の “Internal” ボタンを押すことで，温度センサー，光センサーなどの内部状態を値で確認することも可能で，デバッグにも役立つようになっている．

また，センサーに与える温度と光の強さ（照度）は図 3.4 下に示すコントローラから．スライダーで手動で調整し，入力できるようになっている．

図 3.4　Arduino シミュレータ

3.6　教材

本節では，中学校の授業で用いることを想定した，Arduino を制御する教材例を示す．テーマを設けた例題に対していくつかの演習問題を用意するという構成になっている．なお，演習時間が限られることを想定し，演習問題は*を付けた問題は解答必須とし，残りは余裕がある場合に取り組む問題とした．

3.6.1　例題 1:LED を点灯する（デジタル出力）

赤色 LED を 5 秒間点灯させる．

```
openPort("COM3")
pinMode(10, "OUTPUT")
digitalWrite(10, 1)
sleep(5000)
digitalWrite(10, 0)
closePort()
```

演習 1-1*　ポート 13 番に接続されている，緑色 LED を 10 秒間点灯させるプログラムを作成せよ．

演習 1-2*　赤色 LED を 5 秒間点灯させた後に，緑色 LED を 5 秒間点灯させるプログラムを作成せよ．

演習 1-3　赤色 LED を 2 秒点灯させた後に，緑色 LED を 5 秒間点灯させよ．ただし，緑色 LED が点灯して 3 秒後に赤色 LED を消灯させること．

3.6.2　例題 2:LED を点灯する（アナログ出力）

赤色 LED の明るさを変化させるプログラム．

```
openPort("COM3")
pinMode(10, "OUTPUT")
analogWrite(10, 100)
sleep(1000)
analogWrite(10, 255)
sleep(1000)
analogWrite(10, 0)
closePort()
```

演習 2-1*　赤色 LED を 1 秒間隔で 50, 100, 150, 200, 255 と明るくするプログラムを作成せよ．

演習 2-2　赤色 LED を 1 秒間隔で 50, 100, 150, 200, 255 と明るくした後，200, 150, 100, 50, 0 と暗くするプログラムを作成せよ．

3.6.3 例題 3:条件分岐

点数で異なる色の LED を 5 秒間点灯させる.

```
整数 s, pin
openPort("COM3")
pinMode(10, "OUTPUT")
pinMode(13, "OUTPUT")

「点数を入力してください」 を表示する
s を入力する
もし s>=40 ならば
  | pin ← 13
を実行し，そうでなければ
  | pin ← 10
を実行する

digitalWrite(pin,1)
sleep(5000)
digitalWrite(pin,0)
closePort()
```

演習 3-1* 入力された値が正ならば緑, 0 以下ならば赤の LED を 5 秒間点灯するプログラムを作成せよ.

演習 3-2 入力された値が偶数ならば緑，奇数ならば赤の LED を 5 秒間点灯するプログラムを作成せよ.

3.6.4　例題 4:繰り返し

色 LED を 5 回点滅する．

```
整数 i
openPort("COM3")
pinMode(10, "OUTPUT")

i を 1 から 5 まで 1 ずつ増やしながら,
  | digitalWrite(10, 1)
  | sleep(1000)
  | digitalWrite(10, 0)
  | sleep(1000)
を繰り返す

closePort()
```

演習 4-1*　緑色 LED を 10 回点滅させるプログラムを作成せよ．

演習 4-2*　赤色 LED を 5 回点滅させた後，緑色 LED を 5 回点滅させるプログラムを作成せよ．

演習 4-3　赤色 LED と緑色 LED を交互にそれぞれが 10 回点滅するプログラムを作成せよ．

3.6.5 例題 5:スイッチからの入力（デジタル入力）

スイッチによって点灯する LED を切り替える．

```
整数 i, a
openPort("COM3")
pinMode(6, "INPUT")
pinMode(10, "OUTPUT")
pinMode(13, "OUTPUT")
sleep(1000)
i を 1 から 1000 まで 1 ずつ増やしながら,
  | a ← digitalRead(6)
  | もし a=0 ならば
  |   | digitalWrite(10, 1)
  |   | digitalWrite(13, 0)
  | を実行し，そうでなければ
  |   | digitalWrite(13, 1)
  |   | digitalWrite(10, 0)
  | を実行する
  | sleep(500)
を繰り返す
closePort()
```

演習 5-1* 左のスイッチが押されたら赤の LED を点灯，中央のスイッチが押されていれば赤の LED を消灯するプログラムを作成せよ．

演習 5-2 赤の LED が消灯時に左のスイッチが押されれば赤の LED を点灯し，赤の LED が点灯時に左のスイッチが押されれば赤の LED を消灯するプログラムを作成せよ．

演習 5-3* 左のスイッチが押されたら赤の LED を点灯，右のスイッチが押されたら緑の LED を点灯，そうでなければどちらの LED も消灯するプログラムを作成せよ．

3.6.6　例題 6:センサーからの入力（アナログ入力）

温度センサーから値を取得する．

```
実数 t
openPort("COM3")
pinMode(5, "INPUT")
t ← analogRead(5)
「温度センサーの値は」と t を表示する
closePort()
```

演習 6-1*　光センサーの値（アナログ入力 4 番ピン）を取得するプログラムを作成せよ．

演習 6-2*　光センサーで得られた値が，80 以上ならば緑，それより小さければ赤の LED を 5 秒間点灯するプログラムを作成せよ．

演習 6-3　温度センサーの出力は電圧 0V〜5V の信号で，それが値 0〜1023 として得られる．温度センサーの出力を電圧に変換するプログラムを作成せよ．

演習 6-4*　温度センサーの出力は 400mV(=0.4V) が 0 ℃となり，1 ℃につき 19.53mV 電圧が上がる．演習 6-3 で得られた電圧を元に，温度を計算して出力するプログラムを作成せよ．

3.6.7　例題 7:繰り返してセンサーから入力を得る

100 ミリ秒ごとに光センサーの値を 100 回表示する．

```
整数 a, i
openPort("COM3")
pinMode(4, "INPUT")

i を 1 から 100 まで 1 ずつ増やしながら,
  | a ← analogRead(4)
  | 「光センサーの値は」と a を表示する
  | sleep(100)
を繰り返す

closePort()
```

演習 7-1*　温度センサーから読み取られた値を 100 ミリ秒毎に 100 回表示するプログラムを作成せよ．

演習 7-2*　例題 7 と同様に 100 ミリ秒ごとに光センサーの値を 100 回得て，80 以上ならば緑，それより小さければ赤の LED を点灯するプログラムを作成せよ．

3.6.8　例題 8:総合問題

光センサーの値をスピーカーに出力．

```
整数 a,i
openPort("COM3")
pinMode(4,"INPUT")
pinMode(11,"OUTPUT")
a ← analogRead(4)
analogWrite(11,a)
sleep(2000)
analogWrite(11,0)
closePort()
```

演習 8-1*　光センサーから読み取られた値を 100 ミリ秒毎に 100 回，スピーカーに出力するプログラムを作成せよ．なお，プログラム終了前には音を止めるようにすること．

演習 8-2　光センサーから読み取られた値を 100 ミリ秒毎に 100 回，赤色 LED にアナログ出力するプログラムを作成せよ．なお，プログラム終了前には LED を消灯すること．

演習 8-3*　左のスイッチを押せば赤の LED を点灯，中央のスイッチを押せば緑の LED を点灯，右のスイッチを押せばスピーカーに 100 を出力して鳴らすプログラムを作成せよ．

演習 8-4　3 つのスイッチのうち 1 つを乱数で決め，そのスイッチが押されたら，1 秒間，緑の LED を点灯し，スピーカーを鳴らし，違うスイッチが押された場合は赤の LED を点灯するプログラムを作成せよ．なお，0〜2 の乱数は，random(2) で作ることができる．

3.7　授業実践

作成した学習支援ソフトウェアを使い，高校生を対象に実践を行った．実践授業は，2012 年に PEN のプログラミング経験がある生徒を対象に，続いて 2013 年に初めてプログラミングを学ぶ生徒を対象に行った．

3.7.1　2012 年の授業

大阪学院大学高等学校の 3 年生の選択授業「プログラミング演習」において，2012 年 10〜11 月の 3 回 (1 回 100 分) を使い，15 名を対象に授業を行った．3.6 節で示した教材のうち，例題 1（LED）,6 (温度センサー), 3（条件分岐,LED）,7 (繰り返し, 温度センサー),8（総合問題，スピーカーと押しボタンスイッチ）に相当する内容を扱った．3 コマの授業のうち，初回では Arduino との通信に不都合が生じたため，シミュレータを用いた演習とし，残りはシミュレータと実機を併用した演習を行った．授業後にアンケートを行い，ハードウェアの構成要素の制御に関する理解度を集計した結果が表 3.1 である．時間不足もあり，最後に扱ったスピーカーや押しボタンスイッチに関する理解度は高くないが，温度センサーなどはおおむね理解できていることが伺える．

表 3.1　ハードウェア制御に関する理解度 (2012 年)

LED	温度センサー	光センサー	スピーカー	押しボタンスイッチ
77%	80%	75%	62%	50%

3.7.2　2013 年の授業

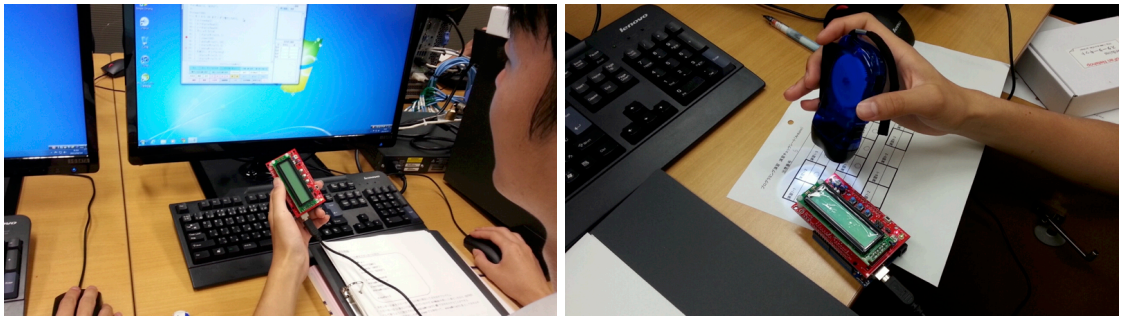

図 3.5　授業の様子（2013 年）

2012 年の授業の結果を受け，3.6 節で示した教材を作成し，これを使って，大阪学院大学高等学校の 3 年生を対象に，同じ「プログラミング演習」において，授業を実施した（図 3.5）．受講生は 12 名で，プ

ログラミング経験は，経験ありが 2 名，ほとんど経験なしが 7 名，まったく経験なしが 3 名であった．また，PEN を用いたプログラミングの経験がある生徒は居なかった．授業は，2013 年 4~6 月の 6 回 (1 回 100 分) を使って行った．

毎回の授業後にアンケートをとり，ハードウェアの構成要素の制御に関する理解度の変化を表したものが図 3.6 である．最初の 2 回は LED に関してのみの演習内容であったので，その理解度のみを聞いており，2 回目で理解度が 100% となった．3 回目以降は，温度センサー，光センサー，押しボタンスイッチについて理解度を尋ねた．その結果，複合的な課題に取り組み始めた 5 回目では大きく理解度が下がったが，6 回目では 90% まで上昇し，ほとんどの生徒に理解してもらえた結果となった．

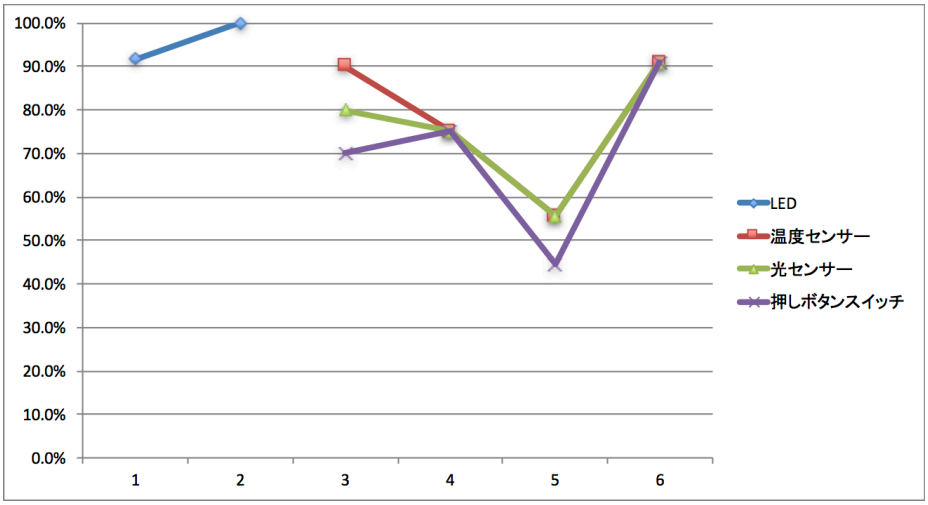

図 3.6　ハードウェア制御に関する理解度の変化 (2013 年)

3.8　結論

本研究では「プログラムによる計測・制御」の仕組みを学ぶための学習環境として，初学者用プログラミング学習環境 PEN を改良し，簡単に任意の関数追加できるプラグイン機能を実現し，これを用いて新たなハードウェア制御関数と Arduino シミュレータを追加した．また，プログラミングの初学者でも計測制御を使った学習ができるような入門用の教材を用意した．

提案した教材を使用することで，9 割の生徒が演習したハードウェア部品の制御方法を理解したと自

己評価する結果を得られたが，演習に要する時間は 100 分の授業が 6 回と多くの時間を要した．ハードウェアのセットアップ等，時間を要する部分はあったが，一般的な授業での展開を考えた場合は，より短時間で演習が終えれるように，例題の数を半分（LED への出力，センサーからの入力，条件分岐，繰り返し）にするなど，教材内容の見直しが必要であると考える．

また，パソコンと Arduino を接続するという演習形態は，パソコン側のシリアルの設定など，一般的な授業ではトラブルが生じる可能性も大きいので，Raspberry Pi[46] などの小型コンピュータを利用した演習環境を構築することで，中学校などの授業に広く展開できると考える．

第 4 章

プログラミング入門教育における図形描画先行型のコースウェアが学習に与える影響

4.1 はじめに

PEN は，2006 年度から大阪大学の文学部と人間科学部の各 3 クラスにおいて，1 年生対象の情報リテラシー科目の中のプログラミング演習のための環境として，10 年にわたって使用されてきた．その例題として図形描画を学習させることで学生の学習意欲が上がることは経験的には感じられていた．これを客観的に評価するために，2008 年に図形描画を中心とした例題から始めるコースウェアと，キーボードから数字を入力して，計算結果をコンソールに出力する例題から始める伝統的なコースウェアとを比較し，学生の学習意欲や理解度に差が生じているかどうかを調査した [47]．しかし，結果として，2 つのコースウェアにおける学生の意識および理解度に関する明確な有意差を見い出すことはできなかった．この原因として，以下のことが考えられる．

1. 2 つのコースウェアに難易度の差があった
2. 各クラスの授業の進め方に違いがあった
3. 各例題に費やす時間がクラスによって異なっていた

そこで，これらの点について考慮し，教材の構成と授業の進め方を見直し，2011 年度からコースウェア間の難易度と学習内容を統一し，実施するコースウェアが特定の学部や教員に偏らないように，教員が担当するコースウェアを 1〜2 年ごとに交換するなど工夫して再調査を行ってきた．本章では図形描画先行型のコースウェアが学習に与える影響について 2011 年度から 2014 年度まで 4 年にわたって調査し，アンケートによる学習効果の調査と期末試験による学習達成度の調査を分析した結果について述べる．

4.2　関連研究

初学者にプログラミングをどのように教えるべきかについては多くの研究がなされている．[48] では，「カリキュラム」「教授法」「言語選択」「ツール」の 4 つのカテゴリーに分け，これまでの初学者へのプログラミング教育の研究を広範囲に分析している．[49] では，プログラミングの敷居を下げるという視点で，プログラミング言語や環境を分類している．これらはいずれも広範囲なサーベイではあるが，プログラミング学習のコースウェアの違いを比較するような研究は見当たらない．

初学者教育におけるプログラミング環境としては，視覚的な対象物を動かすことによりプログラム学習を進める，Scratch[16, 17, 50]，ドリトル [18, 19]，ビスケット [20] などが広く用いられている．最近では，プログラミング教育活動である Hour of Code のチュートリアル [51] で，親しみやすいキャラクタを使い，学習を進めていくようなコースウェアも提供されている．これらのプログラミング環境で用いられる教材は，プログラムの実行結果を視覚的に確認できるため，初学者がプログラミング課題に取り組みやすく，学習しやすくなることが示唆されている．また，視覚的な教材を開発することに着目した研究としては，岡本らが画面上だけではなく実物を使った「視覚的顕在化」に着目して，マイコンボードを使用したコースウェアの開発とアンケートを元にした授業の評価を行っている [52]．このように実行結果を視覚的に出力するプログラミング教材の開発や有効性について多くの研究がなされているが，それらと従来のテキストベースの出力をするプログラミング教材との比較研究は見当たらない．

表 4.1　各年度の受講者数と授業内容

年度	学部	第 1 教室	第 2 教室	第 3 教室	合計
2011	人間科学部	45（図）	45（図）	49（従）	139
	文学部	59（図）	64（従）	59（従）	182
2012	人間科学部	48（従）	48（従）	46（図）	142
	文学部	56（従）	69（図）	56（図）	181
2013	人間科学部	48（従）	48（図）	45（図）	141
	文学部	60（従）	63（従）	59（図）	182
2014	人間科学部	48（図）	48（従）	46（従）	142
	文学部	60（図）	62（図）	60（従）	182

（従）：従来型，（図）：図形型

4.3　授業の概要

4.3.1　対象とするクラス

大阪大学では，全学部において 1 年次における情報教育科目が開講されており，文学部では 1994 年以降，人間科学部では 1995 年以降，情報教育科目「情報活用基礎」が必修科目となっている．本研究では，この 2 学部の授業における実践を対象とした．

両学部の「情報活用基礎」は 1 年次前期に開講され，それぞれ受講者を 3 つのクラスに分け（表 4.1），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）で担当している．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ている．プログラミング演習は終盤のテーマとし，両学部とも 90 分 ×4 回を割り当てている．

4.3.2　コースウェアの概要

コースウェアは，特徴の異なる 2 種類を用意した．一つは，キーボードからの入力に対して，計算結果をコンソールに文字列として出力する形式の演習を繰り返すもの (従来型と呼ぶ) であり，もう一つは，

描画用のウィンドウに図形を描画する形式の演習を繰り返すもの (図形型と呼ぶ) である．ただし，4 回目の授業では，従来型のコースウェアで図形描画の演習を取り上げ，図形型のコースウェアで従来型の演習を取り入れており，一方のコースウェアでしか学べない学習項目はないように配慮している．表 4.1 に各クラスでどちらの形式の授業を行ったかを示す．表 4.2，4.3 にそれぞれのコースウェアで取り上げたすべての例題と練習問題を示す[*1]．

各コースウェアで取り上げた例題・練習問題は，それまでの数年の経験に基づいて、授業担当教員の間で相談して改良を加えてきたものである．特に，両コースウェア間の難易度の差をできるだけ小さくするように配慮してきた．ただし，これはあくまで授業担当者の主観に基づいたものである．

[*1] 例題および練習問題の詳細は，[33] の授業資料 (従来型) および授業資料 (図形型) を参照されたい

表 4.2　2014 年度の授業時間配分 (従来型コースウェア)

第 1 回	予定	文 3	人 2	人 3
準備: プログラミングとは	25	20	22	16
例題 1.1 (入力の 3 倍)	20	20	20	26
練習 1.1 (長方形の面積)	10	10	11	15
例題 1.2 (3 の倍数の判定)	10	10	7	7
例題 1.3 (3 の倍数か否かの判定)	05	10	0	8
第 2 回	予定	文 3	人 2	人 3
宿題の解説	10	10	9	10
練習 1.6 (合否判定・乱数)	15	15	10	13
例題 1.4 (成績判定)	10	10	10	14
練習 1.10 (うるう年)	15	17	11	16
例題 1.5 (数当てゲーム)	15	8	10	12
例題 1.6 (10 回繰り返し)	10	9	6	10
第 3 回	予定	文 3	人 2	人 3
宿題の解説	10	10	12	10
練習 1.15 (繰返・乱数の平均)	10	11	15	12
例題 1.7 (二重ループ)	06	5	10	7
練習 1.23 (□を n 行目に n 個)	14	13	12	11
例題 1.8 (繰返 (2)–乱数)	05	3	5	3
例題 1.9 (実数–温度変換)	15	14	10	10
練習 1.29 (階乗)	15	19	13	20
第 4 回	予定	文 3	人 2	人 3
宿題の解説	10	5	10	9
例題 1.10 (描画関数の使い方)	12	22	12	18
例題 1.11 (円の水平配置)	13	8	12	13
練習 1.33,34(円を並べる (1,2))	17	13	16	16
例題 1.12 (円の描画・乱数)	08	4	8	8
練習 1.38,39(円の描画 2・乱数)	10	12	12	12

単位：分

表 4.3　2014 年度の授業時間配分 (図形型コースウェア)

第 1 回	予定	文 1	文 2	人 1
準備: プログラミングとは	25	20	20	25
例題 1.1 (緑の円)	30	17	20	30
練習 1.2 (長方形)	10	18	17	10
第 2 回	予定	文 1	文 2	人 1
宿題の解説	10	11	12	10
例題 1.2 (3 の倍数の判定)	15	15	7	17
例題 1.3 (3 の倍数か否かの判定)	05	9	9	10
練習 1.5 (合否判定・乱数)	15	9	16	13
例題 1.4 (成績判定)	10	12	10	7
練習 1.9 (うるう年)	15	17	13	13
第 3 回	予定	文 1	文 2	人 1
宿題の解説	10	10	7	10
例題 1.5 (半径当て・乱数)	15	18	12	18
例題 1.6 (繰り返し・乱数)	10	14	8	12
練習 1.14,15(繰返・乱数)	15	12	16	15
例題 1.7 (円の水平配置)	10	10	8	10
第 4 回	予定	文 1	文 2	人 1
宿題の解説	10	11	6	10
例題 1.8 (二重ループ)	08	13	9	10
練習 1.23 (n 行目に n 個並べる)	12	13	14	10
例題 1.9 (繰返 (2)–乱数)	05	1	6	5
例題 1.10 (実数–座標の平均)	15	23	17	20
練習 1.27 (階乗)	15	18	28	15

単位：分

4.3.3　授業の概要

以下，両コースウェアの各回の授業の進め方および実習時間の配分について説明する．

授業の進め方

授業は基本的に例題を解説し，その後その例題に関連した一つもしくは二つの練習問題に取り組ませることを繰り返した．以下に，従来型の例題 1.1 を例に説明する．

例題 1.1 キーボードから 1 つの整数を入力し，それを 3 倍した値を出力するプログラムを書いてみよう．

プログラム例:

```
1: 整数 a
2: 整数 b
3: 「整数を入力してください：」 を表示する
4: a ← input()
5: b ← a * 3
6: b を表示する
```

毎回の授業に先立ち，各担当教員には「授業時間記録シート」[33] を配布している．図 4.1 にこのシートの一部を示す．

20 分:例題 1.1 (入力の 3 倍)

・PEN の使い方を解説しつつ、「入力支援ボタン」を活用しながら、例題 1.1 のプログラムを入力する。

(学生と一緒に…)

・実行させてみる

・一行実行させながら、プログラムの各行を解説する

・プログラムを保存する (ファイル名は ex1.01.pen)

図 4.1　授業時間記録シート（一部）

このシートは，各回の授業で実施予定の例題と練習問題に費やす時間の目安と，授業の進め方を簡潔に

記している．

図 4.1 に記されているように，教員は PEN の使い方を解説しつつ，受講者と一緒にプログラム例を入力する，次に「一行実行」によりプログラムを動作させつつプログラムの各行について説明する．説明する内容は，文献 [33] の「授業資料」の各例題の下に書かれた解説と同じである．

このように，「授業時間記録シート」を用いることで，各クラスの授業の進め方をなるべく統一するように努めた．その後，通常はノーヒントで予め決めておいた練習問題に取り組ませる．しばらく時間をおいて受講者の様子をみながら，必要に応じてヒントを出している．授業で扱った例題および練習問題は表 4.2,4.3 のものがすべてである．授業資料 [33] には，表 4.2,4.3 に掲載されていない例題も多数あるが，これらは練習問題を早くやり終えた受講者のためのオプション問題として用意している．

授業時間配分の調整

過去の授業経験から，担当教員の合意によって，それぞれの例題，練習問題等に割りあてる予定時間を定めている (表 4.2,4.3 参照)．各担当教員は，この予定時間にできるだけ沿って授業を進めたが，受講者の反応などから，必要に応じて個々の教員の判断で授業の進行速度を調整した．実際に各例題，練習問題に要した時間 (2014 年度) を表 4.2,4.3 に示している．表からわかるように，どのクラスもほぼ予定通りの時間で授業を進めることができたと思われる．

例題および練習問題以外に要した時間は以下の通りである．両コースウェアともに、初回の授業では，最初の 25 分 (予定時間、以下同様) でプログラミング学ぶ意義の説明と実習の準備作業を行っている．2 回目から 4 回目の授業では，最初の 10 分で、前回の宿題について解説している．また、すべての回で授業の最後の 10 分程度でアンケートを実施した．上記に挙げた時間以外はすべて実習 (例題および練習問題) の時間であり，両コースウェア間での実習時間の差はない．

また，第 4 回を除いて毎回プログラム作成の宿題を課した ([33] 参照)．宿題に要した時間は各回のアンケートで答えてもらっており，どの回も概ね 90% 以上の受講者が 30 分 ∼ 1 時間以内と回答している．

4.4　学習効果の検証

本研究は，コースウェアの違いが学生のプログラミング学習にどのような影響を与えるかについて検討することが目的である．そこで，学習効果について検証するために，各授業回の終了時に Web 上でアン

ケート調査を行った．また，学習の達成度は期末試験の成績を用いて検証した．本節では，分析対象者とアンケート調査の概要について説明し，学習効果についての分析および考察を行う．

4.4.1　分析対象者

分析対象となる授業の受講者数は表 4.1 に示す通りである．4 回の授業でのアンケート結果の変化と，期末試験の成績を分析するために，本研究では 4 回の授業に出席してその際に行ったすべてのアンケートに答え，かつ，期末試験を受けた学生のみを分析対象とした．表 4.4 にその人数内訳を示す．

表 4.4　分析対象学生の内訳

年度	男性	女性	従来型	図形型	合計
2011	55	114	88	81	169
2012	50	79	69	60	129
2013	42	76	61	57	118
2014	66	109	86	89	175

4.4.2　調査の方法

授業アンケート調査の内容は以下の通りである．

1. 前回の宿題に対する達成度の自己評価[*2]
2. 授業で扱った個々の例題・練習問題に対する理解度や難易度の自己評価
3. プログラミング内容に関する理解度の自己評価
4. 授業に対する理解度や面白さの自己評価
5. プログラミングに対する楽しさや容易さの自己評価

本研究では，授業を行った 4 回の回答結果について対応を取る必要があったため，ログイン名の入力を求めた．ただし，入力の強制は行わなかった．質問項目は質問文と 5 段階から 7 段階の回答選択肢に

[*2] 第 1 回アンケートにはこの項目はない．

よって構成されていた．

この調査は，毎回の授業の終了時に行った．回答者は Web ページを用いてアンケートに回答する．回答に要する時間は 10 分程度であった．

4.4.3　授業に対する理解度

コースウェアの違いが学生に与える影響を評価するために，質問項目の中から，まず，授業に対する理解度について尋ねた項目について分析することとした．この項目の質問文は「今日の授業は理解できましたか？」で，6 段階の回答選択肢を用意した．回答選択肢の内容とその数量化は，理解度が高いと判断された場合に値が高くなるよう，「理解できた = 6，だいたい理解できた = 5，どちらかというと理解できた = 4，どちらかというと理解できなかった = 3，あまり理解できなかった = 2，理解できなかった = 1」とした．

この回答を従属変数とし，「年度 (2011 年・2012 年・2013 年・2014 年の 4 水準)」，「授業方式 (従来型授業・図形型授業の 2 水準)」，「授業回 (第 1 回から第 4 回までの 4 水準)」を要因とする 3 要因分散分析を行った．要因「年度」と「授業方式」は受講生が年度で異なるので「対応なしの要因」，要因「授業回」は同一の受講生の分析であるので「対応ありの要因」であった．表 4.5 は分散分析の結果を表したものである．要因「授業方式」以外の主効果・1 次交互作用・2 次交互作用が有意水準 5% もしくは 0.1% で有意となった．また，図 4.2 は，授業に対する理解度の授業回別平均を授業方式および年度別に表したものである．値が高いほど，理解度が高いことを表している．

表 4.5 より，2 次交互作用が有意であることから，年度・授業方式・授業回によってさまざまな有意差があることが分かる．そこで，表 4.2 から，授業方式によって各授業回の平均値の変動 (第 1 回から第 2 回，第 2 回から第 3 回，第 3 回から第 4 回の 3 つ) が有意であるかどうかについて年度ごとに検討した．統計的検定としては対応のある母平均の差の検定を用いたが，有意かどうかを判断する基準となる有意水準は，全体の危険率が 5% となるように Bonferroni 法を用いて調整した値を用いた[*3]．以下では，このような方法を「有意水準 5% の Bonferroni 法」と呼ぶこととする．

[*3] この検定での有意水準は，授業方式 2 水準，年度 4 水準，各年度で比較する母平均の対が 3 個であったので，全部で 24 回の有意性判断を繰り返すことになる．したがって，全体で危険率を 5% とするために，ひとつひとつの検定については，$5\%/24 = 0.208...\%$ を有意水準として有意かどうかの判断を行った．

表 4.5　授業に対する理解度の分散分析表

要因	df	平方和	平均平方	F
年度	3	33.0	11.01	3.799 *
授業方式	1	.5	.48	.167
年度 ∗ 授業方式	3	31.7	10.56	3.645 *
誤差 1	583	1688.9	2.90	
授業回	3	68.0	22.66	40.408 ***
年度 ∗ 授業回	9	29.0	3.22	5.741 ***
授業方式 ∗ 授業回	3	186.9	62.28	111.049 ***
年度 ∗ 授業方式 ∗ 授業回	9	26.0	2.88	5.144 ***
誤差 2	1749	981.0	.56	

(*:$p < .05$, **:$p < .01$, ***:$p < .001$)

その結果，従来型授業では，すべての年度で第 2 回から第 3 回の変動が有意であり，2012 年以外の年度で第 3 回から第 4 回の変動が有意であった．図形型授業では，2012 年以外の年度で第 3 回から第 4 回の変動が有意であった．

次に，各年度の従来型授業と図形型授業の平均値を授業回ごとに調べた．検定方法は対応のない母平均の差の検定で，有意性判断については有意水準 5% の Bonferroni 法を用いて行った．その結果，すべての年度において第 3 回の平均値間に有意差が見られた．また，2011 年以外の年度において第 4 回の平均値に有意差が見られた．

検定の結果および図 4.2 の平均値変動を見ると，従来型授業では第 3 回の内容が他の回に比べて難しいと感じられたと思われる．一方，図形型授業では第 4 回の授業内容が他の回に比べて難しいと感じられたようである．従来型授業も図形型授業も，第 3 回の授業は「繰り返し」を扱っていたが，すべての年度において，第 3 回の授業方式別平均値の間に有意差が見られたことから，2012 年を除く他の年度は，図形型授業の方が繰り返しについて理解できたと自己評価していることがわかった．これは同じ内容であっても，図形型授業の方が理解してもらいやすい場合があることを示している．

第 4 回では，従来型授業は図形描画の演習を行い，一方，図形型授業は最後の練習問題としてテキスト

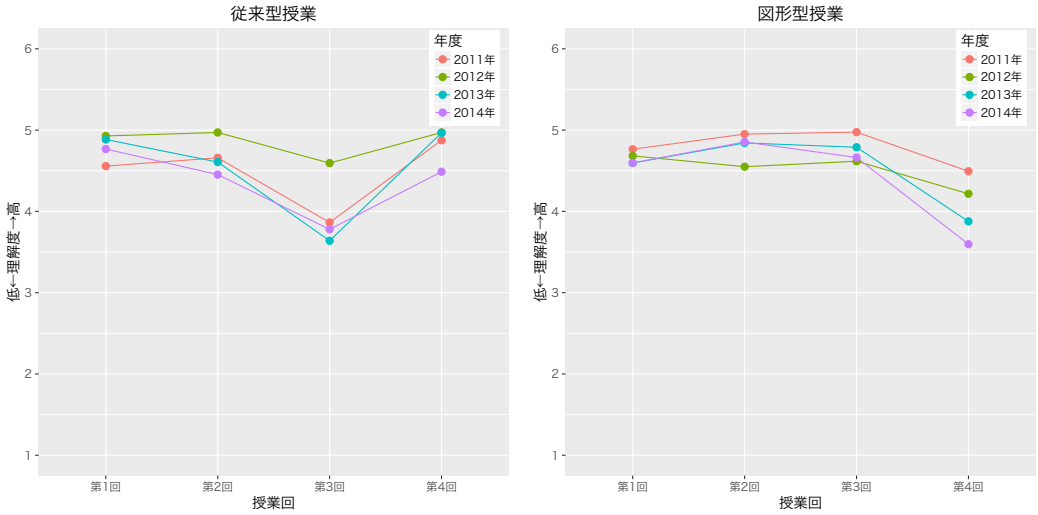

図 4.2　授業に対する理解度の変化

問題の演習を行った．図形型授業を受けている学生にとってはそれまでの内容と一線を画すもののように捉えられたために，難しいと感じたのではないかと思われる．

4.4.4　プログラミングに対する楽しさ

コースウェアの違いが学生に与える影響を評価するために，プログラミングに対して楽しさを感じているかについて尋ねた項目についても分析した．この項目の質問文は「プログラミングはあなたにとって楽しいですか？」で，7 段階の回答選択肢を用意した．回答選択肢の内容と数量化は，楽しいと判断された場合に値が高くなるよう，「とても楽しい = 7，かなり楽しい = 6，どちらかというと楽しい = 5，どちらでもない = 4，どちらかというと楽しくない = 3，あまり楽しくはない = 2，全く楽しくはない = 1」とした．

この回答値を従属変数とし，授業に対する理解度と同じ要因で 3 要因分散分析を行った．表 4.6 は分散分析の結果である．要因「授業方式」の主効果が有意水準 5% で，要因「授業回」の主効果が有意水準 0.1% で有意であった．また，要因「年度」と「授業回」の 1 次交互作用が有意水準 5% で，要因「授業方式」と「授業回」の 1 次交互作用が有意水準 0.1% で有意であった．また，図 4.3 は，プログラミングに対する楽しさの授業回別平均を授業方式および年度別に表したものである．値が高いほど，楽しさが高

いことを表している．

表 4.6　プログラミングに対する楽しさの分散分析表

要因	df	平方和	平均平方	F
年度	3	50.0	16.65	2.218
授業方式	1	43.9	43.88	5.847 *
年度 ∗ 授業方式	3	38.0	12.82	1.708
誤差 1	583	4376.0	7.51	
授業回	3	22.3	7.43	12.068 ***
年度 ∗ 授業回	9	11.9	1.32	2.144 *
授業方式 ∗ 授業回	3	51.0	16.99	27.585 ***
年度 ∗ 授業方式 ∗ 授業回	9	6.8	.75	1.224
誤差 2	1749	1077.1	.62	

(*:$p < .05$, **:$p < .01$, ***:$p < .001$)

本研究の目的は，授業方式の差異による影響を調べることにあるので，有意であった 1 次交互作用の内，授業方式と授業回の 1 次交互作用について検討することとした．図 4.4 は，各授業回の 4 年間の平均値を授業方式別に表したものである．そこで，各授業回において従来型授業と図形型授業の平均値間に有意差があるかどうかについて調べた．検定方法は対応のない母平均の差の検定で，有意性判断は有意水準 5% の Bonferroni 法を用いて行った．その結果，第 2 回と第 3 回において平均値間に有意差が見られた．

図 4.3，図 4.4 より，図形型授業では楽しさの平均値が第 2 回で大きく増加し，それが第 3 回でも維持されている．一方，従来型授業では第 1 回と比べて第 2 回はほぼ同じで，第 3 回は減少している．これらの回では，条件分岐と繰り返しの制御構造を学ぶため，プログラミングに対する難しさを感じ始め，楽しさが低くなる可能性が考えられる．しかし，図形型授業では図 4.2 に示す通り理解度はほぼ変化せず，楽しさが増加しているので，図形型問題が学生に難しさを感じさせず，楽しく課題に取り組ませる結果につながっていると考えられる．一方，従来型授業では 3 回目で楽しさが減少しているが，4 回目に図形描画を学ぶことにより楽しさが向上し，2 回目までと同水準もしくは高い値になっている．また，図形型授業と比べてもほぼ同じ値となっている．このことより，文字・数値の出力のみで演習を進め，繰り返しな

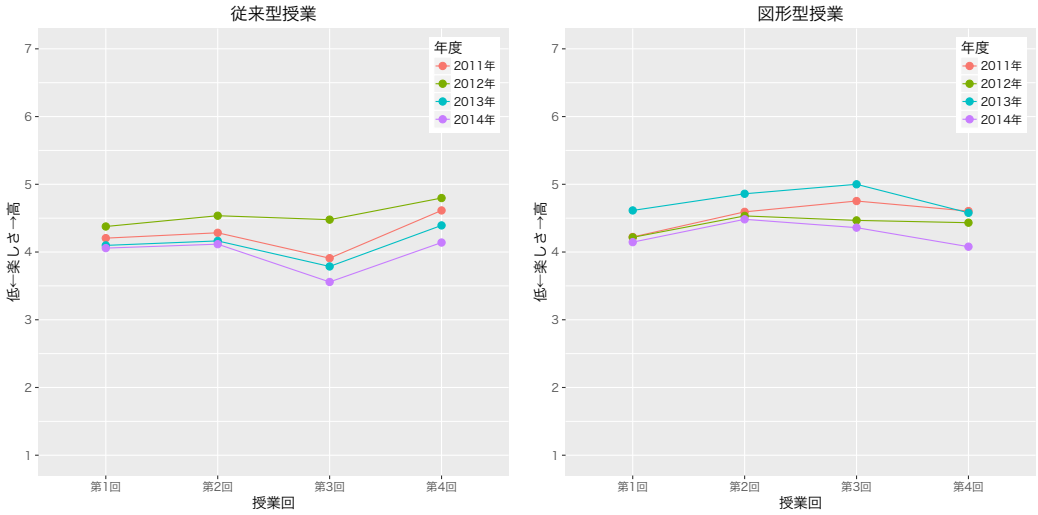

図 4.3　プログラミングに対する楽しさの変化

どの難しい内容で楽しさが減っても，図形描画の演習を行うことにより，その減少を補えていることが見て取れる．

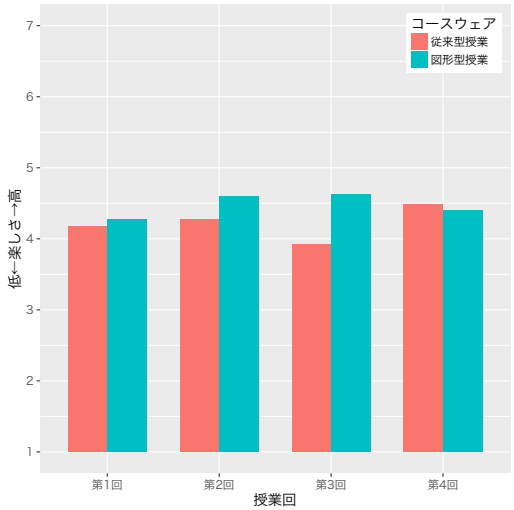

図 4.4　プログラミングに対する楽しさの平均値（授業回，授業方式別）

問題 1　以下のプログラムは，図 A のような図形を描画するものである．　ウィンドウの大きさは 450 × 450．点線は説明上の補助線であり　無視しても良い．点線の間隔は，50 ピクセルである．　空欄に入れるのに適当な語句・数を解答欄に書け．

```
 1: 整数　n
 2: gOpenWindow(　《あ》　)
 3: gSetLineWidth(3)
 4: n ← 1
 5: n < 8 の間,
 6:   | gSetLineColor(255, 0, 0)            /* 赤 */
 7:   | gDrawLine(50, 《い》, 《う》, 《え》)
 8:   | gSetLineColor(0, 0, 255)            /* 青 */
 9:   | gDrawLine(　《お》,　50, 《か》, 《き》)
10:   | n ← (《く》)
11: を繰り返す
```

問題 2　3+6+9+…と 3 から順に 3 の倍数を加えていき，その和が 20000 より大きく　なったとき，最後に加えた数を求めるプログラムを書け．また，最後に加えた数も示せ．

問題 3　400 × 400 のウィンドウに，乱数を用いて一様に分布する半径 3 の円を 2000 個描くプログラムを書け．ただし，円の中心座標に応じて，図 B のように色分けせよ。

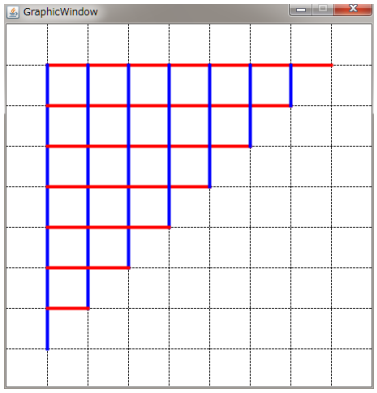

図 A 問題 1 の実行結果

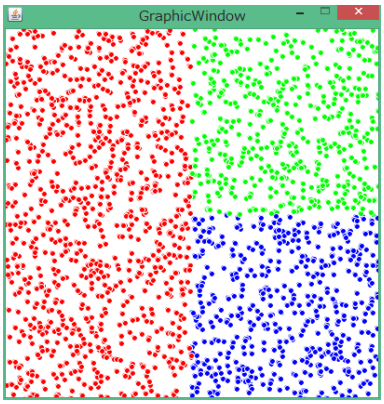

図 B 問題 3 の実行結果

図 4.5　2014 年度の期末試験問題

4.5　学習達成度の検証

本節では，学習達成度の分析を行うために行った期末試験の概要について説明し，その結果の分析と考察を行う．

4.5.1　期末試験の方法

学習の達成度の測定は授業終了後の試験により行った．この試験は学期の期末試験を兼ねており，プログラミングに関する 3 問以外にも 2〜3 の問題を出題している．これらの出題の影響は完全に排除できていないが，プログラミングに関する問題に解答する時間が十分確保されるように配慮した．

プログラミングに関する 3 問について，年度毎に問題内容は異なるが、難易度はほぼ同じと考えており，以下の点は共通している．問題 1 は一部に空欄を含む図形描画を行うソースプログラムを提示し，その空欄を埋める問題であり，問題 2 はコンソールにテキストの結果を出力する従来型の問題であり，問題 3 は図形型の問題である．問題 2，および，問題 3 は，いずれも繰り返しの中に条件分岐を含む構造のプログラムを作成する問題である．満点は，問題 1（以降「穴埋め問題」と呼ぶ）が 5 点，問題 2（以降「テキスト問題」と呼ぶ）が 7 点，問題 3（以降「図形問題」と呼ぶ）が 8 点である．2014 年度に文学部の学生対象の試験で出題した問題を図 4.5 に示す．人間科学部に対しては，数値等を一部変更したほぼ同じ問題を出題した．

試験中，学生は PC を利用してプログラムの動作を確認することができ，任意の Web ページを閲覧することもできる．また，配布した資料を含めて，任意の資料や書籍の閲覧も許されている．ただし，メールや SNS を用いて他の学生とコミュニケーションを取ることは禁止した．プログラミングに関する問題について，学生のほとんどは PC でプログラムを作成・実行してから解答を記入していた．

4.5.2　期末試験の結果

表 4.7 は，各問題の得点を従属変数として，年度 (2011 年・2012 年・2013 年・2014 年) と授業方式 (従来型授業・図形型授業) を要因とする 2 要因分散分析を行った結果である．穴埋め問題 (表 4.7(1)) では，年度の主効果が有意水準 0.1% で，年度と授業方式の交互作用が有意水準 1% で有意となった．テキ

スト問題 (表 4.7(2)) では，年度と授業方式の主効果がいずれも有意水準 0.1% で有意となった．図形問題 (表 4.7(3)) では，年度の主効果，年度と授業方式の交互作用が有意水準 5% で有意となった．図 4.6 は，3 つの問題それぞれについて，授業方式別の平均点を年度ごとに表したものである．

表 4.7　試験結果の分散分析表

(1) 穴埋め問題

要因	*df*	平方和	平均平方	*F*	
年度	3	83.7	27.89	12.351	***
授業方式	1	4.3	4.33	1.961	
年度 ∗ 授業方式	3	29.0	9.67	4.281	**
誤差	583	1316.6	2.26		

(2) テキスト問題

要因	*df*	平方和	平均平方	*F*	
年度	3	246.6	82.20	11.138	***
授業方式	1	343.6	343.57	46.556	***
年度 ∗ 授業方式	3	57.1	19.02	2.577	
誤差	583	4302.4	7.38		

(3) 図形問題

要因	*df*	平方和	平均平方	*F*	
年度	3	69.9	23.30	3.109	*
授業方式	1	8.4	8.44	1.127	
年度 ∗ 授業方式	3	67.5	22.51	3.004	*
誤差	583	4368.6	7.49		

(*:$p < .05$, **:$p < .01$, ***:$p < .001$)

穴埋め問題では，交互作用が有意になったことから，年度によって授業方式の違いによる平均の差の傾向が異なると予想される．そこで，授業方式別平均値の違いを年度別に調べた．検定方法は対応のない母平均の差の検定で，有意性判断は有意水準 5% の Bonferroni 法を用いて行った．その結果，2014 年度の

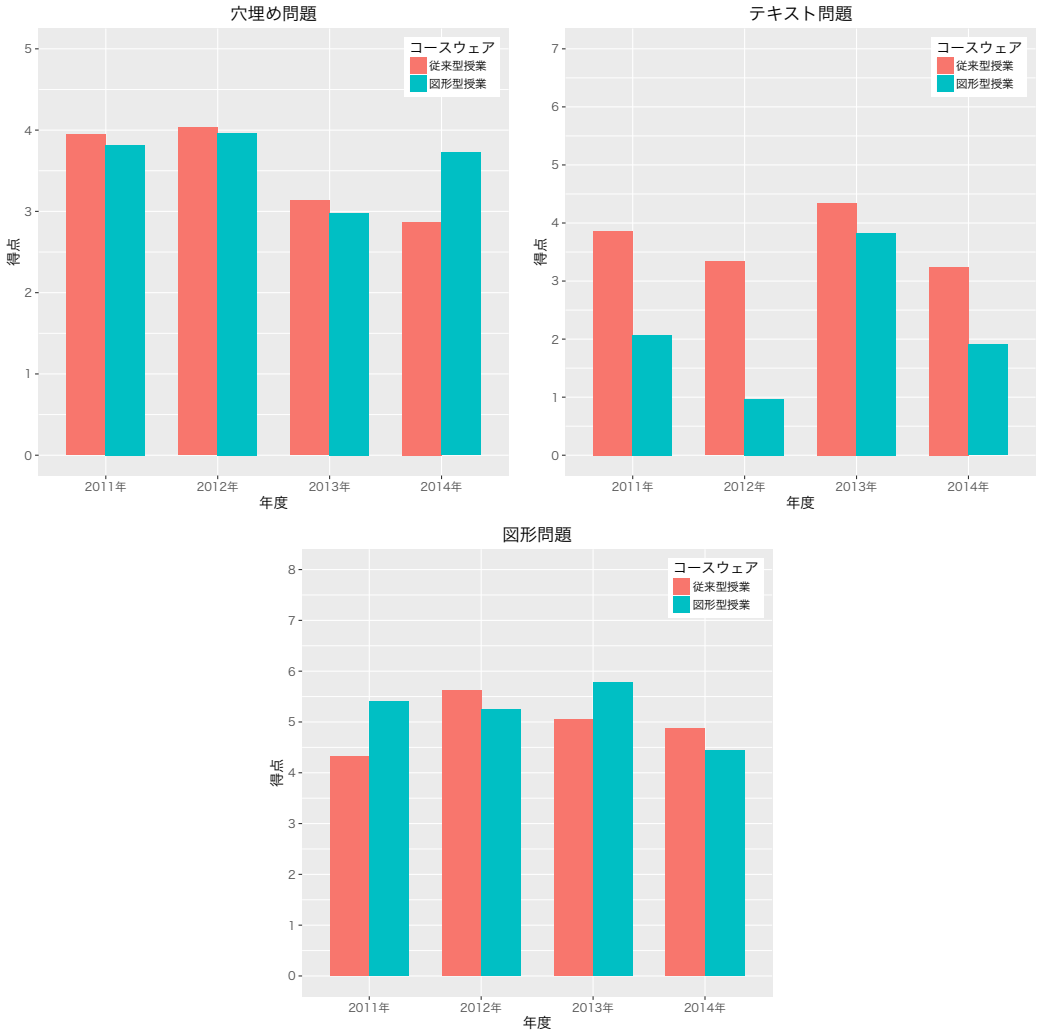

図 4.6　期末試験の年度・授業方式別の成績比較

み授業方式による差が見られた．図 4.6 の傾向から 2014 年度に関しては図形型授業の方が成績がよかったと考えられる．

テキスト問題では，交互作用が有意とはならず，主効果のみが有意となった．このことから，年度に関係なく授業方式の違いによる平均の差が見られた．図 4.6 の傾向と合わせて考えると，従来型授業の方が図形型授業よりも平均点が高いことが分かる．また，年度の主効果についても有意であったことから，問題の難易度については，一定ではなかったと考えられる．こうしたことから，テキスト問題については，

問題の難易度に関わらず，従来型授業を受けた学生の方がより高い点数を取る傾向であったと言える．

図形問題では，交互作用が有意になったことから，年度によって授業方式の違いによる平均の差の傾向が異なることが予想される．そこで，授業方式別平均値の違いを年度別に調べた．検定方法は対応のない母平均の差の検定で，有意性判断は有意水準 5% の Bonferroni 法を用いて行った．その結果，いずれの平均値間にも有意な差が見られなかった．以上のことから，図形問題において，図形型授業を受けた学生の方が高い点数を取るといった傾向は見られなかったと考えられる．

4.5.3　試験結果の考察

本研究での 2 つのコースウェアでは，プログラミングに関する学習内容が同じになるように配慮している．異なっていたのは，学習を進める際，従来的なテキストベース問題を多く扱ったか，図形的な問題を多く扱ったかという点である．このことから，テキスト問題は，従来型授業を受けた学生，穴埋め問題と図形問題は図形型授業を受けた学生に取って解答しやすい問題であったと考えられる．したがって，授業で扱った時間が長いほど学習到達度が高くなるとすれば，解答しやすい問題ほど点数が高くなるはずである．

分析の結果，テキスト問題では授業方式による差が見られたが，穴埋め問題と図形問題では授業方式による差が全体的には見られなかった．図 4.6 のテキスト問題結果を見ると，どの年度においても従来型授業の成績が高いことが分かる．このことから，テキスト問題の成績は従来型授業の方が高いと考えられる．先の前提に従うならば，この結果は妥当である．

一方，穴埋め問題と図形問題で授業方式間での差が見られなかったことは，先の前提と矛盾する．これは，従来型授業の第 4 回目（試験の直近）が図形問題を扱った内容であったためと考えられる．図 4.3 からも分かるとおり，従来型授業を受けた学生は第 4 回目の授業後，プログラミングの楽しさが増している．このことから，図形問題に対して主観的には親しみやすいと感じていたと思われる．図 4.2 の理解度のアンケート結果も合わせて考えると，これは問題に対する積極性や記憶・知識の定着においてよい効果をもたらし，その結果，図形問題において授業方式間に差が見られなかったと思われる．このことが正しければ，テキスト問題においても差が小さくなっていなければならない．しかし，そのような結果は得られなかった．これは，図形型授業では，テキスト問題を扱ったのは第 4 回目の最後の練習問題だけで時間が短かったことが一因である．また，アンケート（図 4.2，図 4.3）の結果から分かるとおり，学生は従来

型授業の場合とは逆の心理的影響を受けていた．したがって，テキスト問題では図形問題のようなことが起こらなかったと考えられる．

以上のことから，本研究の結果は，従来型授業が図形型授業よりも学習達成度の観点から効果的であるということを表しているのではなく，図形描画を用いることでプログラミング学習が促進される可能性を示唆している．教材が与える心理的影響と学習到達度の関係性をより明確にすることは今後の課題である．

4.6　結論

本研究では，2011〜2014 年の 4 年間にわたり，図形描画を主としたコースウェアと，従来型のコースウェアとで学生のモチベーション，理解度，成績に差があるかどうかを調査した．授業内容の変化との対応ではその特徴に差があることがわかり，繰り返し処理の内容を学習する際，従来型授業では，その内容が難しいと評価されているが，図形型授業では同じ内容であっても難しいと評価されなかった．この結果，図形型では繰り返しのようなつまづき易い学習内容でも理解度や楽しさをほとんど下げることなく学習できていることが伺えた．一方で，従来型では繰り返しの学習において理解度や楽しさを下げることがわかった．ただしこの授業では，同等の教育内容を提供するために，最終回である 4 回目に従来型授業で図形描画を，図形型授業で文字列の出力や実数計算を扱っている．その結果，図形型授業では 4 回目に理解度や楽しさが下がっている．一方で，従来型授業では 4 回目に理解度や楽しさが上がっている．これらのことより，図形描画を用いた課題は学習者のモチベーションや理解度を上げ，これを主としたコースウェアを提供することは初学者に対する短期間のプログラミング教育に有用であると考えられる．

本研究では，分析の性質上，4 回の授業すべてに出席し Web での調査に回答した受講学生が対象であったが，欠席したりアンケートに未回答の学生がいたため，実際の受講学生数に比べて分析対象者数が少なかった．今後，調査の方法を改善し，分析対象者数を増やす工夫が必要である．また，どのような条件・状況のときにどちらのコースウェアを用いるのが適切かといった検討を行っていく必要があるだろう．

第 5 章

まとめ

本論文では，プログラミング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し，それを達成するためのサポートツールとして開発した初学者教育用プログラミング環境 PEN とその拡張ツールの開発について紹介し，評価を行った．また，プログラミング入門教育のためのコースウェアについての評価も行った．

第 2 章では，プログラミング環境 PEN の紹介と，その評価を行った．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に，日本語キーワードを用いた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，変数値の観察など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．大阪大学の情報リテラシー科目における評価では，JavaScript を用いたクラスと理解度を比較した結果，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 の有用性を示せた．

第 3 章では，「プログラムによる計測・制御」の仕組みを学ぶための学習環境として，PEN にプラグイン機能を実装することにより，新たなハードウェア制御関数と Arduino シミュレータを追加した．また，それを利用し，プログラミングの初学者でも計測制御を使った学習ができるような入門用の教材を用意した．高校生を対象とした評価の結果，提案した教材を使用することで，9 割の生徒が演習したハードウェア部品の制御方法を理解したと自己評価する結果を得られた．

第 4 章では，大阪大学の情報リテラシー科目におけるプログラミング入門授業において，4 年間にわたり，図形描画を主としたコースウェアと，従来型のコースウェアとで学生のモチベーション，理解度，成績に差についての調査結果について述べた．繰り返し処理の内容を学習する際，テキストでの出力を行う

課題を用いる従来型では，その内容が難しいと評価されているが，出力に図形描画を伴う課題を用いる図形型では同じ内容であっても難しいと評価されなかった．この結果，図形型では繰り返し処理のようなつまづき易い学習内容でも理解度や楽しさをほとんど下げることなく学習できていることが伺えた．

PEN は，入力補助のボタンは用意しているが，プログラミングを記述するという形態をとっているので，プログラムの構造を理解していない学習者が入力補助ボタンで挿入された構造を壊し，結果として文法エラーを引き起こしてしまうことが少なくない．そこで，名古屋高校の中西渉教諭により PEN を拡張し，フローチャートによる PEN 上のプログラムの作成，および，PEN のプログラムからフローチャートへの変換が可能な PenFlowchart[53]（図 5.1）が作成された．PenFlowchart は，大阪学院大学高等学校における授業の序盤で用いた分析により，課題を解くペースが早くなるなど，初学者の学習がスムーズに行われることが分析できている．

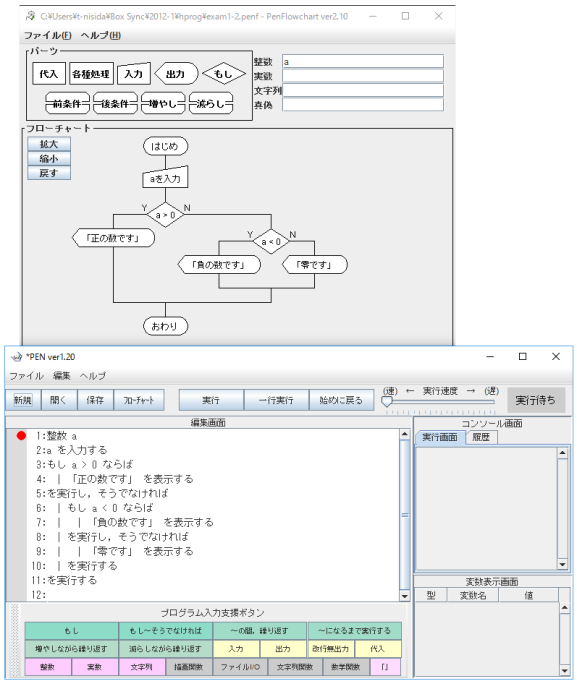

図 5.1　PenFlowchart

また，最初はブロック型のプログラミングから始め，その後にテキストを用いたプログラミングに移行する学習スタイルを提供する oPEN[54]（図 5.2）も開発されている．これは，Openblocks[55] を用いた

ブロックプログラミング環境でブロックで組み立てたプログラムを PEN などで用いることができるテキストプログラムに変換する機能を持つ．oPEN は現在，大阪学院大学高等学校における「情報の科学」のプログラミング演習に利用しており，プログラミング導入教育の実践と検証を行なっている．

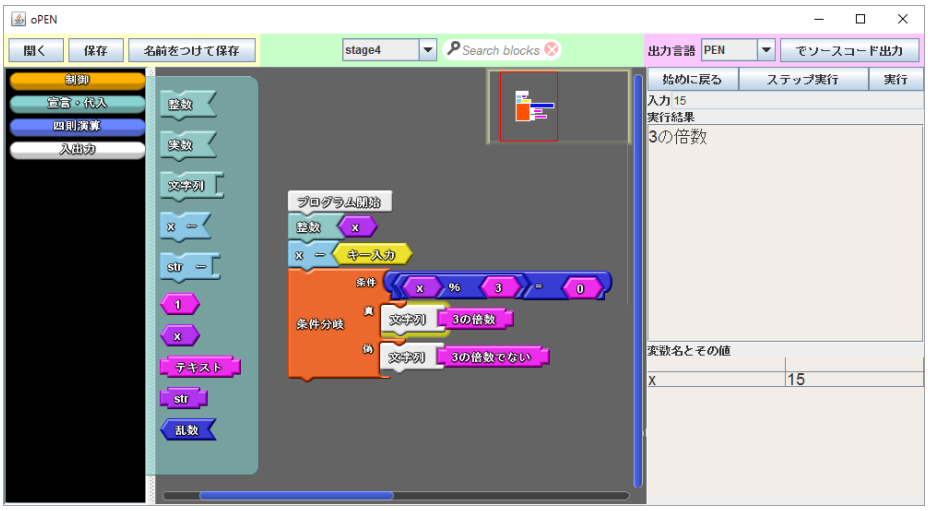

図 5.2　oPEN

この他，「プログラムによる計測・制御」の仕組みを学ぶための PEN の利用に関しては，京都ノートルダム女子大学における LilyPad Arduino を用いた授業実践がある [56]．ここでは，本論文で紹介した PEN のプラグイン機能を利用した LilyPad Arduino のシミュレータを活用することにより，文系の共通教育科目でも制御を含むようなプログラミングの授業が行えることを示している．

現在はタブレットが普及し，学校においてもパソコンを導入する代わりに，授業でタブレットを導入する動きが高まっている．今後は，テキスト型のプログラミングを意識しながら，Google が提供しているビジュアルプログラミングエディタのライブラリ Blockly[57] などを用い，タブレットの操作でプログラミングが可能な環境を提供したいと考えている．

付録 A

xDNCL の言語仕様

xDNCL は大学入試センターの「情報関係基礎」で用いられている試験用手順記述標準言語 DNCL に準拠しており，一部拡張したものである．

A.1 定数

定数を用いる場合，以下の規則に従う必要がある．

- 整数型の定数は，小数点を含めてはいけない (例: 12, − 4).
- 実数型の定数は，小数点を含めなければならない (例: 12.0, − 4.0).
- 文字列型の定数は，「」または" "で囲まなければならない (例: 「プログラム」, "abc").

A.2 変数

変数の型として「整数」「実数」「文字列」がある．変数は使用に先だってその型を宣言しておかねばならない．変数名は半角英字から始まり，2 文字目以降は半角英数字でなければならない．また，整数型の変数の初期値は 0, 実数型の場合は 0.0, 文字列型の場合は NULL(空の文字列) である．

整数 ≪変数≫ ，≪変数≫ ， . . . ，≪変数≫
実数 ≪変数≫ ，≪変数≫ ， . . . ，≪変数≫
文字列 ≪変数≫ ，≪変数≫ ， . . . ，≪変数≫

使用例

```
整数 i, integer  // 変数名 i, integer は整数型の変数であると宣言
実数 x, real     // 変数名 x, real は実数型の変数であると宣言
文字列 str, name // 変数名 str, name は文字列型の変数であると宣言
```

A.3　配列

変数の配列を用いるには以下の宣言が必要である．

```
整数 a[n] // a[0], a[1], a[2], ... , a[n] の n+1 個の整数の変数領域を確保
実数 r[n] // r[0], r[1], r[2],  ... , r[n] の n+1 個の実数の変数領域を確保
```

宣言例

```
整数 count[5]  // count[0]〜count[5] までの 6 つの変数領域を確保
実数 pos[10]   // pos[0]〜pos[10] まで 11 個の変数領域を確保
```

2 次元以上の配列を宣言することもできる．

宣言例

```
整数 board[8,8]    //board[0,0]〜board[8,8] の 9 × 9 の変数領域を確保
実数 space[2,3,5] //space[0,0,0]〜space[2,3,5] の 3 × 4 × 6 の変数領域を確保
```

A.4　演算子

演算子については以下にまとめたものが利用できる．全角文字の利用も可能としている．

A.4.1　算術演算子

演算子	意味	例	式の値
＋, +	加算	8 + 3	8 に定数 3 を加えた値 (=11)
−, −	減算	x − 2	変数 x の値から 2 を引いた値
×, *	積算	y * 1.5	変数 y の値を 1.5 倍した値
÷, /	除算	z / 2	変数 z の値を 2 で割った値
%, ％	剰余	z ％ 5	変数 z の値を 5 で割った余りの値

A.4.2　比較演算子

演算子	意味	使用例	式の値
=, =	等しい	x = 0	x が 0 ならば真, それ以外ならば偽
>, >	より大きい	y > 5	y が 5 より大きければ真, 以下なら偽
≧, >=	以上	y ≧ 5	y が 5 以上ならば真, ちいさければ偽
<, <	より小さい	z < 1.2	z が 1.2 より小さければ真, 以上なら偽
≦, <=	以下	z ≦ 1.2	z が 1.2 以下ならば真, 大きければ偽
≠, !=	等しくない	z ≠ 6	z が 6 以外ならば真, 同じであれば偽

A.4.3　論理演算子

演算子	意味	使用例	式の値
かつ	積集合	a ≧ 0 かつ a ≦ 10	a が 0 以上 かつ 10 以下ならば真, それ以外は偽
または	和集合	b < 0 または b > 100	b が 0 未満 または 100 より大きければ真, それ以外は偽
でない	否定	c = 5 でない	c が 5 でないなら真, 5 ならば偽

A.4.4　演算結果のデータ型について

演算結果のデータ型は演算対象のデータ型によって決まる．なお，演算前に演算結果のデータ型に変換してから，演算が行われる．

演算例	演算結果のデータ型
整数 ＋ 整数	整数
実数 ＋ 整数	実数
実数 ＋ 実数	実数
文字列 ＋ 整数	文字列として，文字列結合される
文字列 ＋ 実数	文字列として，文字列結合される
文字列 ＋ 文字列	文字列として，文字列結合される

A.5　式

定数もしくは変数を演算子で結合したものは式である．また，式と式を演算子で結合したものも式である．単独の定数もしくは変数も式である．

例

```
x + 5
x * y - 5.0
"abc" + str
x = y
```

A.6　コメント文 (注釈)

```
/*　《コメント》　*/
//　《コメント》
```

コメント文として書かれた文字列は，プログラム実行時には無視される．

A.7　出力文

A.7.1　改行あり出力

```
≪出力文≫ を表示する
```

≪出力文≫で指定された式や文字列定数をコンソール画面に表示し，その後改行する．

使用例 (変数 x が 1.6 の場合)

```
「x の値:」を表示する
x を表示する
```

出力例

```
x の値:
1.6
```

A.7.2　改行なし出力

```
≪出力文≫ を改行なしで表示する
```

≪出力文≫で指定された式や文字列定数をコンソール画面に表示し，その後改行をしない．

使用例 (変数 x が 1.6 の場合)

```
「x の値:」を改行なしで表示する
x を表示する
```

出力例

```
x の値:1.6
```

A.7.3　複数の変数・文字列の出力

変数や文字列を一緒に出力する場合，それらを「**と**」で結び列挙する．

使用例

```
「答えは」 と ans と 「です」 を表示する
```

変数 ans が 50 の場合の出力例

```
答えは 50 です
```

A.7.4　予約語

改行などを行うための以下の文字列は定数，変数などに利用することはできない．

LF, ¥n	改行
CR, ¥r	その行の先頭へ移動
HT, ¥t	水平タブ
NL	改行文字 (システムに依存)
EOF	ファイルの終端

A.8　代入文

A.8.1　代入

≪変数≫ ← **≪式≫**

≪式≫に書かれた定数や式の演算結果を≪変数≫に指定された変数へ代入する．なお，右辺の≪式≫の値の型のいかんに関わらず，代入によって，左辺の≪変数≫のデータ型に自動変換される．例えば，≪式≫の演算結果が実数型で，左辺の≪変数≫が整数型の場合の演算では，小数点以下は切り捨てられる．

使用例

```
x ← 2.34
y ← x + 4.32
str ← "abc"
```

A.8.2　入力

≪変数≫ ← input()
≪変数≫　を入力する

input() で キーボードからの入力が≪変数≫に代入される．または「を入力する」を付けることにより入力が≪変数≫に代入される．型は，≪変数≫のデータ型に合わせて変換される．

使用例（入力が整数値として a に代入）

整数 a
a ← input()

使用例（入力が実数値として x に代入）

実数 x
x **を入力する**

A.9　条件分岐

A.9.1　条件文 (1) if〜then 文

```
もし ≪条件式≫ ならば
  | ≪処理≫
を実行する
```

≪条件式≫が真の場合，≪処理≫を実行し，偽の場合は≪処理≫を実行しない．いずれの場合もその後に次の文へ制御を渡す．

使用例

```
もし x = 10 ならば
  | 「条件が成立しました」を表示する
を実行する
「終了」を表示する
```

変数 x が 10 の場合の出力例

```
条件が成立しました
終了
```

変数 x が 5 の場合の出力例

```
終了
```

A.9.2　条件文 (2) if〜then〜else 文

```
もし ≪条件式≫ ならば
  | ≪処理 1 ≫
を実行し，そうでなければ
  | ≪処理 2 ≫
を実行する
```

≪条件式≫が真の場合，≪処理 1 ≫を実行し，偽の場合は≪処理 2 ≫を実行する．

使用例

```
もし x = 20 ならば
  | 「条件が成立しました」を表示する
を実行し，そうでなければ
  | 「条件が成立しませんでした」を表示する
を実行する
```

変数 x が 20 の場合の出力例

```
条件が成立しました
```

変数 x が 10 の場合の出力例

```
条件が成立しませんでした
```

A.9.3　条件文 (3) else〜if 文

```
もし ≪条件式 1≫ ならば
  | ≪処理 1≫
を実行し，そうでなくもし ≪条件式 2≫ ならば
  | ≪処理 2≫
を実行し，そうでなければ
  | ≪処理 3≫
を実行する
```

≪条件式 1≫が成立した場合，≪処理 1≫を実行し，成立しなくて≪条件式 2≫が成立した場合，≪処理 2≫を実行し，すべて成立しない場合は≪処理 3≫を実行する．

使用例

```
もし x >= 80 ならば
  | 「x は 80 以上です」を表示する
を実行し，そうでなくもし x >= 60 ならば
  | 「x は 79〜60 の間です」を表示する
を実行し，そうでなければ
  | 「x は 59 以下です」を表示する
を実行する
```

変数 x が 95 の場合の出力例

```
x は 80 以上です
```

変数 x が 70 の場合の出力例

```
x は 79〜60 の間です
```

変数 x が 30 の場合の出力例

```
x は 59 以下です
```

A.10　繰り返し

A.10.1　繰り返し文 (1) while-do 文

```
≪条件式≫ の間,
  | ≪処理≫
を繰り返す
```

≪条件式≫が成立していれば，≪処理≫を実行する．≪処理≫の実行終了後，再び≪条件式≫の判定を行い，成立すれば≪処理≫を再び実行し，これを繰り返す．≪条件式≫が成立しない場合は「を繰り返す」の次の行へ進む．

使用例

```
x ← 1
x < 5 の間,
  | x を表示する
  | x ← x + 1
を繰り返す
```

出力例

```
1
2
3
4
```

A.10.2　繰り返し文 (2) repeat-until 文

```
繰り返し,
  | ≪処理≫
を,  ≪条件式≫ になるまで実行する
```

≪処理≫を実行した後，≪条件式≫の判定を行う．条件式が成立していなければ，≪処理≫を再び実行し，成立した場合は次の行へ進む．

使用例

```
x ← 1
繰り返し,
  | x を表示する
  | x ← x + 1
を,  x > 5 になるまで実行する
```

出力例

```
1
2
3
4
5
```

A.10.3　繰り返し文 (3) for 文

```
≪変数≫ を ≪数値 1≫ から ≪数値 2≫ まで ≪増加値≫ ずつ増やしながら,
  | ≪処理≫
を繰り返す
```

≪変数≫の部分に指定されたループ変数に≪数値 1 ≫の値を代入し≪処理≫を実行する．≪処理≫の実行後，ループ変数に≪増加値≫の値を加算し，ループ変数の値が≪数値 2 ≫になるまで繰り返す．

```
≪変数≫ を ≪数値 1≫ から ≪数値 2≫ まで ≪減少値≫ ずつ減らしながら,
  | ＜処理＞
を繰り返す
```

≪変数≫の部分に指定されたループ変数に≪数値 1 ≫の値を代入し≪処理≫を実行する．≪処理≫の実行後，ループ変数から≪減少値≫の値を減算し，ループ変数の値が≪数値 2 ≫になるまで繰り返す．

使用例 1

```
x を 1 から 5 まで 1 ずつ増やしながら,
  | x を表示する
を繰り返す
```

出力例 1

```
1
2
3
4
5
```

使用例 2

```
x を 5 から 1 まで 1 ずつ減らしながら,
  | x を表示する
を繰り返す
```

出力例 2

```
5
4
3
2
1
```

A.10.4　繰り返し文の脱出

```
繰り返しを抜ける
```

繰り返し文を途中で強制的に抜け出す命令.

使用例 2

```
x を 1 から 10 まで 2 ずつ増やしながら,
  | x を表示する
  | もし x = 5 ならば
  |   | くり返しを抜ける
  | を実行する
を繰り返す
```

出力例 2

```
1
3
5
```

A.11　組み込み関数

A.11.1　数学関数

書式	意味	使用例	戻り値	戻り値のデータ型
random(x)	0〜x の乱数値 (整数) を返す	random(10)	0〜10 の値を返す	整数
floor(x)	x の小数点以下 切り捨て	floor(24.64)	24.0	実数
ceil(x)	x の小数点以下 切り上げ	ceil(24.64)	25.0	実数
round(x)	x の小数点以下 四捨五入	round(24.64)	25	実数
abs(x)	x の絶対値	abs(-234)	234	引数と同じ
int(x)	x の型を「整数」に変換	int(10.2345)	10	整数
sin(x)	角度 x (ラジアン) の正弦を返す	sin(95.0)	0.683261714736121	実数
cos(x)	角度 x (ラジアン) の余弦を返す	cos(50)	0.15425144988758405	実数
tan(x)	角度 x (ラジアン) の正接を返す	tan(70)	1.2219599181369434	実数
sqrt(x)	x の平方根の値を返す	sqrt(5)	2.23606797749979	実数
log(x)	x の自然対数値 (底は e) を返す	log(2)	0.6931471805599453	実数

- 引数 x のデータ型は 実数型 であるが，整数型 で渡した場合，実数型 に型変換されて取り扱われる
- abs(x) の戻り値のデータ型は，引数のデータ型によって決定される

A.11.2　文字列操作関数

書式	意味	使用例	戻り値	戻り値のデータ型
str2int(str)	str の 1 文字目を ASCII コードの数値に変換	`str2int("A")`	65	整数
int2str(int)	引数 int を ASCII コードと見なしたときの対応する文字を返す	`int2str(70)`	F	文字列
length(str)	str の文字列の長さを返す	`length("PEN")`	3	整数
append(str1, str2)	str1 と str2 を結合した文字列を返す	`append("Mr.", "PEN")`	Mr.PEN	文字列
substring(str, i)	str の先頭から i 文字よりも後の文字列を返す	`substring("smiles", 3)`	les	文字列
substring(str, i, len)	str の先頭から i 文字よりも後ろの len 文字を返す	`substring("smiles", 1, 3)`	mil	文字列
insert(str1, i, str2)	str1 の i 文字目の後に str2 を 挿入する	`insert("abc", 2, "123")`	ab123c	文字列
replace(str1, i, len, str2)	str1 の i 文字目の後から len で示される文字数を str2 で置き換える	`replace("abc", 1, 2, "123")`	a123	文字列
extract(str, delim, i)	文字列 str を delim で区切り，i+1 個目にある文字列を返す	`extract("a:b:c", ":", 2)`	c	文字列
compare(str1, str2)	2 つの文字列を辞書式に比較して正数値を返す	`compare("str1","str2")`	-1,0,1(※ 1)	整数

- 引数 str, delim のデータ型は 文字列型 でなければならない
- 引数 int, i, j, len のデータ型は 整数型 でなければならない
- 文字列の i 文字目は先頭を 0 文字目として数える
- ※ 1 str1 が str2 の辞書列の，前の時に -1，同じの時に 0，後ろの時に 1 となる．

A.11.3 図形描画関数

書式	意味	使用例
描画のためのウィンドウのオープン／クローズ関数		
gOpenWindow(整数 w, 整数 h) 描画領域開く (整数 w, 整数 h)	幅 w，高さ h の描画のためのウィンドウを開く．出現するウィンドウの位置は，画面の左上となる．	`gOpenWindow(400, 400)`
gSaveWindow(文字列 filepath, 文字列 format) 描画領域保存 (文字列 filepath, 文字列 format)	描画したウィンドウをファイル名 filepath に保存する．保存形式 (format) は JPEG または PNG のどちらかを指定する．	`gSaveWindow("abc.png", "PNG")`
gCloseWindow() 描画領域閉じる ()	描画のためのウィンドウを閉じる．	`gCloseWindow()`
図形の属性を指定するための関数（日本語名関数の末尾は「設定」の代わりに「変更」でもよい）		
gSetLineColor(整数 r, 整数 g, 整数 b) 線色設定 (整数 r, 整数 g, 整数 b)	r, g, b の値の範囲は 0-255．線の色属性を，赤の要素 r，緑の要素 g，青の要素 b で指定する．初期値は黒 (0, 0, 0)．	`gSetLineColor(255, 255, 0)`
gSetLineShape(整数 type) 線種設定 (整数 type)	描画する線のタイプを変更．線を使用する全ての描画に影響．type=0 のとき実線 (初期値)，type=1 のとき破線，type=2 のとき間隔の狭い破線，type=3 のとき一点鎖線，type ≧ 4 のとき線の種類を変更しない	`gSetLineShape(3)`
gSetLineWidth(整数 width) 線太さ設定 (整数 width)	描画する線の太さを選択した種類に変更．線を使用する全ての描画に影響．初期値は 1．	`gSetLineWidth(3)`
gSetArrowType(整数 type) 線矢設定 (整数 type)	type で矢じりの形状を指定する．(1 種類のみの実装であるため type の数値に関わらず同じ)	`gSetArrowType(1)`
gSetArrowDir(整数 edge) 線方向設定 (整数 edge)	gDrawLine で描画する線分の両端に矢印を付加する関数．edge で矢じりの箇所を指定する．edge=0 のとき矢じりなし (初期値)，edge=1 のとき始点のみ矢じり描画，edge=2 のとき終点のみ矢じり描画，edge=3 のとき両端に矢じり描画，edge ≧ 4 のとき指定箇所を変更しない	`gSetArrowDir(3)`
gSetFillColor(整数 r, 整数 g, 整数 b) 塗色設定 (整数 r, 整数 g, 整数 b)	図形内部の色を，赤の要素 r，緑の要素 g，青の要素 b で指定する．使用方法は gSetLineColor と同じ．初期値は黒 (0, 0, 0)．	`gSetFillColor(255, 255, 0)`
gSetDotShape(整数 type) 点種設定 (整数 type)	gDrawPoint で描く点の種類を指定する．type=0 のとき小さな丸い点 (初期値)，type=1 のとき少し大きな丸い点，type=2 のとき大きな丸い点，type ≧ 3 のとき点の種類を変更しない	`gSetDotShape(2)`
gSetTextColor(整数 r, 整数 g, 整数 b) 文字色設定 (整数 r, 整数 g, 整数 b)	文字列の色属性を，赤の要素 r，緑の要素 g，青の要素 b で指定する．使用方法は gSetLineColor と同じ．初期値は黒 (0, 0, 0)．	`gSetTextColor(255, 255, 0)`
gSetFont(文字列 "String") 文字種設定 (文字列 "String")	gDrawText 描画するフォントの種類を指定変更する．初期値はシステムで定義されているデフォルトフォント．指定できるフォントは以下の通り．（ただし，計算機環境に依存する）明朝，ゴシック，TimesRoman, Courier, Helvetica, Dialog, DialogInput, ZapfDingbats, Serif, SansSerif, Monospaced	`gSetFont("Monospaced")`
gSetFontType(整数 style) 文字タイプ設定 (整数 style)	gDrawText 描画するフォントのスタイルを指定する．style=0 のときプレイン (初期値)，style=1 のときボールド，style=2 のときイタリック，style=3 のときボールド＋イタリック，style ≧ 4 のときスタイルを変更しない	`gSetFontType(2)`
gSetFontSize(整数 size) 文字サイズ設定 (整数 size)	gDrawText で描画するフォントのサイズを指定する．初期値は 10．	`gSetFontSize(20)`

書式	意味	使用例
図形を描画するための関数		
gDrawText(文字列 ”string”, 整数 x, 整数 y) 文字描画 (文字列 ”string”, 整数 x, 整数 y)	座標 (x, y) を一文字目の左下にあわせ，文字列を描く．	gDrawText("文字列", 20, 30)
gDrawPoint(整数 x, 整数 y) 点描画 (整数 x, 整数 y)	座標 (x, y) に点を描く．	gDrawPoint(20, 30)
gDrawLine(整数 x1, 整数 y1, 整数 x2, 整数 y2) 線描画 (整数 x1, 整数 y1, 整数 x2, 整数 y2)	座標 (x1, y1) から座標 (x2, y2) まで線分を描く．	gDrawLine(50, 60, 100, 120)
gDrawBox(整数 x, 整数 y, 整数 width, 整数 height) 矩形描画 (整数 x, 整数 y, 整数 width, 整数 height)	座標 (x, y) から幅 width 高さ height の矩形を描く．	gDrawBox(20, 30, 40, 50)
gFillBox(整数 x, 整数 y, 整数 width, 整数 height) 矩形塗描画 (整数 x, 整数 y, 整数 width, 整数 height)	gDrawBox() の内部を gSetFillColor() で指定した色で塗りつぶした図形を描画する．	gFillBox(20, 30, 40, 50)
gDrawOval(整数 x, 整数 y, 整数 width, 整数 height) 楕円描画 (整数 x, 整数 y, 整数 width, 整数 height)	座標 (x, y) から幅 width 高さ height で矩形を定義し，内接する楕円を描く．	gDrawOval(20, 30, 40, 50)
gFillOval(整数 x, 整数 y, 整数 width, 整数 height) 楕円塗描画 (整数 x, 整数 y, 整数 width, 整数 height)	gDrawOval() の内部を gSetFillColor() で指定した色で塗りつぶしたものを描画する．	gFillOval(20, 30, 40, 50)
gDrawCircle(整数 x, 整数 y, 整数 r) 円描画 (整数 x, 整数 y, 整数 r)	座標 (x, y) を中心に半径 r の円を描く．	gDrawCircle(60, 70, 50)
gFillCircle(整数 x, 整数 y, 整数 r) 円塗描画 (整数 x, 整数 y, 整数 r)	gDrawCircle() の内部を gSetFillColor() で指定した色で塗りつぶしたものを描画する．	gFillCircle(60, 70, 50)
gDrawArc(整数 x, 整数 y, 整数 width, 整数 height, 整数 start, 整数 ext, 整数 type) 弧描画 (整数 x, 整数 y, 整数 width, 整数 height, 整数 start, 整数 ext, 整数 type)	座標 (x, y) から幅 width，高さ height で矩形を定義し，内接する楕円において (中心から真右を 0 度とし左回りに) start 度から (左回りに)ext 度の弧を描画する．type により弧の閉じ方の種類 (OPEN=0,CHORD=1,PIE=2) のいずれかを指定する．	gDrawArc(20, 30, 40, 50, 60, 90, 0)
gFillArc(整数 x, 整数 y, 整数 width, 整数 height, 整数 start, 整数 ext, 整数 type) 弧塗描画 (整数 x, 整数 y, 整数 width, 整数 height, 整数 start, 整数 ext, 整数 type)	gDrawArc() の内部を gSetFillColor() で指定した色で塗りつぶしたものを描画する．	gFillArc(20, 30, 40, 50, 60, 90, 0)
gDrawPolygon(整数列 x[], 整数列 y[], 整数 n)	配列 x, y で指定した点 (座標) を結んだ n 角形を描く．	gDrawPolygon(x, y, 6)
gFillPolygon(整数列 x[], 整数列 y[], 整数 n)	配列 x, y で指定した点 (座標) を結んだ n 角形を塗りつぶしたものを描画する．	gFillPolygon(x, y, 6)
gDrawPolyline(整数列 x[], 整数列 y[], 整数 n)	x, y で指定した点 (座標) を結ぶ直線を描画する．	gDrawPolyline(x, y, 3)
gClearWindow()	描画 Window をクリアする．	gClearWindow()
座標系に関する関数		
gSetOrigin(実数 x, 実数 y)	描画ウィンドウの左上からの右方向に x，下方向に y の点を原点として再定義する．	gSetOrigin(300, 200)
gOpenGraphWindow(実数 width, 実数 height, 実数 x1, 実数 y1, 実数 x2, 実数 y2, 論理値 axis)	幅 width，高さ height の描画ウィンドウを開く．ウィンドウの左下を (x1,y1) 座標とし，右上を (x2, y2) 座標となるような座標平面として定義する．axis の値を true, false に指定することで座標軸の表示，非表示を指定する．	gOpenGraphWindow(500, 500, -50, -100, 350, 300, true)
gSetMap(実数 x1, 実数 y1, 実数 x2, 実数 y2)	描画ウィンドウの左下を点 (x1,y1)，右上を点 (x2,y2) として再指定する．	gSetMap(0, -50, 150, 200)

A.11.4　ファイル入出力関数

書式	意味	使用例	戻り値
ファイルのオープン／クローズ関数			
openr(文字列 filepath)	filepath で指定したファイルを，データの読み込みのために開く．戻り値はファイル識別子となる整数値で，これを代入した変数を用いて，ファイルからの読み込み等を行う．	**整数** `fd` `fd ← openr("data.txt")`	ファイル識別子（整数）
openw(文字列 filepath)	filepath で指定したファイルを，データの書き込みのために開く．指定したファイルが既に存在しているならば，そのファイルはこれから書き出すデータで置き換えられる．	**整数** `fd` `fd ← openw("data.txt")`	ファイル識別子（整数）
opena(文字列 filepath)	filepath で指定したファイルを，データの追記書込みのために開く．指定したファイルが既に存在しているならば，そのファイルの続きとして，これから書き出すデータを追記する．	**整数** `fd` `fd ← opena("data.txt")`	ファイル識別子（整数）
close(整数 fd)	ファイル識別子 fd で指定したファイルを閉じる．	`close(fd)`	なし
ファイルの読み書きのための関数			
getstr(整数 fd, 整数 n)	ファイル識別子 fd で指定したファイルから，n 文字読み込んでその文字列を返す．ただし，改行コードも 1 文字と数える．	**文字列** `str` `str ← getstr(fd, n)`	文字列
getline(整数 fd)	ファイル識別子 fd で指定したファイルから，1 行読み込んでその文字列を返す．	**文字列** `str` `str ← getline(fd)`	文字列
putstr(整数 fd, 文字列 str)	ファイル識別子 fd で指定したファイルに，str の文字列を書き出す．	`putstr(fd, "abc")`	なし
putline(整数 fd, 文字列 str)	ファイル識別子 fd で指定したファイルに，str の文字列を書き出す．putstr() は，文字列だけを書き出すのに比べて，putline() は，文字列の後に改行コードも書き出す．	`putline(fd, "abc")`	なし
ファイル処理のための補助関数			
isfile(文字列 filepath)	filepath で指定したファイルが存在するかどうかを調べる．ファイルが存在する場合 ”true”，存在しない場合”false”の文字列が返される．	`str ← isfile("data.txt")`	`"true", "false"` (文字列)
rename(文字列 filepath1, 文字列 filepath2)	filepath1 で指定したファイルの名前を，filepath2 に置き換える．	`rename("data.txt",` `"work.txt")`	なし
flush(整数 fd)	ファイル識別子 fd で指定されるファイルに書き込み途中のデータを強制的に書込む．(ファイルは通常バッファリングされるので，この関数を呼び出すことで，バッファ内容を吐き出す)	`flush(fd)`	なし
remove(文字列 filepath)	filepath で指定したファイルを削除する．	`remove("data.txt")`	なし

- ファイルのカレントディレクトリは PEN のプログラムが置かれたディレクトリである．

付録 B

2006 年度文学部・人間科学部「情報活用基礎」全体アンケート

B.1　受講前アンケート

情報処理教育に関する調査

この調査は情報処理教育を円滑に行うための基礎研究の一環として行われるものです．結果は一括して統計処理され，ひとりひとりのデータを取り出して検討するといったことはありません．個人的にご迷惑をおかけするようなことはありませんので，率直なご意見をお答えくださるようお願いいたします．記入洩れがあるとデータとして利用できなくなる可能性があります．よろしくお願いします．

質問 0．あなたの所属学部，学年，性別，学籍番号を入力してください．

学部:(　　)　学年:(　　)　性別:(　　)　学籍番号:(　　)　第 (　　) 教室

質問 1．あなたは大学進学以前に, コンピュータをどのように利用していましたか？次にあげる項目について,1〜5 の中より自分が該当する数値に○をつけてください．

1. 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**
2. コンピュータを使って絵を描く ..... **全く行ったことがない** １ ２ ３ ４ ５ **頻繁に行った**
3. インターネット上の Web ページ (ホームページ) を見る .. **全く行ったことがない** １ ２ ３ ４ ５ **頻繁に行った**
4. コンピュータを使って電子メールのやりとりをする ..... **全く行ったことがない** １ ２ ３ ４ ５ **頻繁に行った**
5. コンピュータで作成したものを印刷する ..... **全く行ったことがない** １ ２ ３ ４ ５ **頻繁に行った**
6. インターネット上の掲示板に書き込む ..... **全く行ったことがない** １ ２ ３ ４ ５ **頻繁に行った**

7. インターネット上に自分のホームページを作成する.....**全く行ったことがない** １ ２ ３ ４ ５ **頻繁に行った**

質問 2．あなたは，次にあげる項目についてどの程度できますか？それぞれの項目について該当する内容の数値を○で囲んでください．内容についてよく分からない場合は「わからない」を○で囲んでください．

1. キーボードを見ないで文字入力ができる.....**できない** １ ２ ３ ４ **できる わからない**
2. いらなくなったファイルを削除することができる.....**できない** １ ２ ３ ４ **できる わからない**
3. 必要なファイルをフロッピーディスクや USB メモリなどにコピーできる **できない** １ ２ ３ ４ **できる わからない**
4. ワープロソフト (たとえば Word など) を用いて簡単な文章を作成できる **できない** １ ２ ３ ４ **できる わからない**
5. コンピュータを用いてホームページを見ることができる.....**できない** １ ２ ３ ４ **できる わからない**
6. コンピュータを用いて目的のホームページを探すことができる.....**できない** １ ２ ３ ４ **できる わからない**
7. コンピュータを用いてホームページを作成できる.....**できない** １ ２ ３ ４ **できる わからない**
8. コンピュータを用いて電子メールのやり取りができる.....**できない** １ ２ ３ ４ **できる わからない**
9. コンピュータを用いて簡単なプレゼンテーションを作成できる.....**できない** １ ２ ３ ４ **できる わからない**
10. コンピュータの設定 (壁紙など) を変更することができる.....**できない** １ ２ ３ ４ **できる わからない**
11. 携帯電話を使ってメールのやり取りができる.....**できない** １ ２ ３ ４ **できる わからない**
12. 携帯電話を使って時刻表を調べることができる.....**できない** １ ２ ３ ４ **できる わからない**
13. ワープロの文書やホームページを印刷できる.....**できない** １ ２ ３ ４ **できる わからない**
14. インターネットを使ってファイルをダウンロードできる.....**できない** １ ２ ３ ４ **できる わからない**
15. コンピュータを用いて blog をつけることができる.....**できない** １ ２ ３ ４ **できる わからない**

質問 3．以下の項目に対して大学入学以前のあなたはどの程度当てはまりますか？該当する数値に○を付けてください．

1. 学校ではコンピュータに詳しい方だった.....**全くあてはまらない** １ ２ ３ ４ ５ **非常にあてはまる**
2. 機器の操作には自信があった.....**全くあてはまらない** １ ２ ３ ４ ５ **非常にあてはまる**
3. 友達とコンピュータについて話をよくした.....**全くあてはまらない** １ ２ ３ ４ ５ **非常にあてはまる**
4. コンピュータに関する新聞記事をよく読んだ.....**全くあてはまらない** １ ２ ３ ４ ５ **非常にあてはまる**
5. コンピュータの設定は他人まかせだった.....**全くあてはまらない** １ ２ ３ ４ ５ **非常にあてはまる**
6. コンピュータをよく利用する方だった.....**全くあてはまらない** １ ２ ３ ４ ５ **非常にあてはまる**

質問 4．コンピュータについていろいろな意見がありますが，以下の項目に対してあなたはどう思いますか？ それぞれの項目についてあなたの意見にもっとも近いと思われる数値を ○で囲んでください．

1. 私はコンピュータを操作するのは不安だ.....**全くそうでない** １ ２ ３ ４ ５ **全くそうだ**
2. 私はコンピュータを利用するとき，操作を誤って何かを壊しそうな気がする **全くそうでない** １ ２ ３ ４ ５ **全くそうだ**

3. 私は就職してコンピュータを操作するような職場にまわされるかもしれないと考えると不安になる .....**全くそうでない** 1 2 3 4 5 **全くそうだ**
4. 私はコンピュータをうまく操作できない人を見ると親しみを感じる **全くそうでない** 1 2 3 4 5 **全くそうだ**
5. 私は人がみている前でコンピュータの操作をすると恥をかきそうだ **全くそうでない** 1 2 3 4 5 **全くそうだ**
6. 私はコンピュータと聞いただけでもうお手上げの気持ちだ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
7. 私は自信を持ってコンピュータを操作できる.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
8. 私は自分の操作に対して, コンピュータがどんな反応をするか予測できる . **全くそうでない** 1 2 3 4 5 **全くそうだ**
9. 私はコンピュータをうまく操作できなくても，いつかはできるようになると思う**全くそうでない** 1 2 3 4 5 **全くそうだ**
10. 私はコンピュータは難しそうなので手を出そうとは思わない.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
11. 私はコンピュータの操作をどうしていいのかわからなくなっても、なんとか使えると思う .....**全くそうでない** 1 2 3 4 5 **全くそうだ**
12. 私はコンピュータで予期せぬ問題が発生しても、どうにかできると思う **全くそうでない** 1 2 3 4 5 **全くそうだ**
13. 私はコンピュータに関する専門用語を容易に理解できると思う.....**全くそうでない** 1 2 3 4 5 **全くそうだ**

質問 5. 現代社会におけるコンピュータやインターネットの関わりについて次のようなことが言われていますが，以下の項目に対してあなたはどう思いますか? それぞれの項目についてあなたの意見にもっとも近いと思われる数値を○で囲んでください.

1. インターネット上では自分の実名を出されて中傷されそうだ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
2. インターネットを利用すると知らない間に自分の個人情報が流出しそうだ **全くそうでない** 1 2 3 4 5 **全くそうだ**
3. インターネットを利用すると犯罪に巻き込まれそうだ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
4. インターネット上での買い物は，偽物をつかまされそうだ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
5. インターネットができなくなると生活が成り立たなくなりそうだ...**全くそうでない** 1 2 3 4 5 **全くそうだ**
6. インターネットなしでは，いろいろな活動ができない気がする.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
7. インターネットがあらゆる活動の中心になっていくような気がする **全くそうでない** 1 2 3 4 5 **全くそうだ**
8. インターネットは生活の一部になっている.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
9. 他にすることがあるのについついインターネットを利用してしまう **全くそうでない** 1 2 3 4 5 **全くそうだ**
10. インターネットを利用していて睡眠不足になってしまうことがある **全くそうでない** 1 2 3 4 5 **全くそうだ**
11. インターネットは複雑でよくわからない.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
12. インターネットを使いこなせないと不利益をこうむってしまいそうだ...**全くそうでない** 1 2 3 4 5 **全くそうだ**
13. インターネットを利用すると迷惑メールやウイルスをもらいそうだ **全くそうでない** 1 2 3 4 5 **全くそうだ**
14. インターネットを使用すると知らない間にハッキングされるかもしれないと思う**全くそうでない** 1 2 3 4 5 **全くそうだ**
15. メールでは真意が伝わらないような気がする.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
16. メールだと相手の反応がウソか本当かわからない.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
17. 人と会って話すより，メールのほうが自分の意思がよく伝わると思う...**全くそうでない** 1 2 3 4 5 **全くそうだ**

18. メールが送信した相手にきちんと届いているかどうか不安だ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
19. メールを送信しても，間違った相手に送信してしまうのではないかと不安になる**全くそうでない** 1 2 3 4 5 **全くそうだ**
20. メールは他の誰かに読まれているような気がする.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
21. メールの返信が遅いと不安になる.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
22. 携帯電話が手元にないと不安になる.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
23. 暇があればつい携帯電話を見てしまう.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
24. 携帯電話がないと，取り残された気分になる.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
25. 自分の知らないところで自分のことが扱われているような気がする **全くそうでない** 1 2 3 4 5 **全くそうだ**
26. これからの社会では、誰かが歪んだ情報を流しそうな気がする.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
27. インターネットを利用することによって，本当に必要な情報を見失いそうだ **全くそうでない** 1 2 3 4 5 **全くそうだ**
28. インターネットの普及によって，有害な情報が蔓延しそうだ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
29. そのうちインターネットによって私たちの生活がコントロールされそうだ **全くそうでない** 1 2 3 4 5 **全くそうだ**
30. 情報を持っている人だけが豊かになる社会になりそうな気がする...**全くそうでない** 1 2 3 4 5 **全くそうだ**
31. コンピュータを使っていると，生身の人間とのコミュニケーションが下手になっていく気がする .....**全くそうでない** 1 2 3 4 5 **全くそうだ**
32. コンピュータの普及によって，人間関係が希薄になっていくような気がする **全くそうでない** 1 2 3 4 5 **全くそうだ**
33. インターネットを使っていると，人に会う機会が減ってしまいそうだ...**全くそうでない** 1 2 3 4 5 **全くそうだ**

質問 6. あなたは情報処理教育の授業で何について学びたいと思いますか？以下の空欄 (裏面も利用可) に自由に記述してください.

B.2　中間アンケート

習熟度に関する調査

この調査は，情報活用基礎の授業で学んだ内容を自己評価するためのものです．成績には一切関係しないので，各自思う通りに回答してください．**裏面にも問題があるので**，忘れずに回答してください．よろしくお願いします．

質問 0．あなたの所属学部，学年，性別，学籍番号を入力してください．

学部:(　　)　学年:(　　)　性別:(　　)　学籍番号:(　　)　第 (　　) 教室

質問 1．以下の質問についてあなたはどの程度あてはまっていますか．あなたの意見に最も該当すると思う数値に○をつけてください．

1. 授業の進度は自分にあっていると思う．.....**全くあてはまらない** 1 2 3 4 5 **非常にあてはまる**
2. 授業内容を十分理解できていると思う．.....**全くあてはまらない** 1 2 3 4 5 **非常にあてはまる**
3. 自分はクラスの中で授業内容の理解レベルが劣っているとは思わない．
   .....**全くあてはまらない** 1 2 3 4 5 **非常にあてはまる**
4. 自分が理解できないことはクラスの大半の人も理解できていないと思う．
   .....**全くあてはまらない** 1 2 3 4 5 **非常にあてはまる**

質問 2．あなたは次に挙げた各項目に対してどの程度習熟していますか．該当する数値を○で囲んでください．項目の内容をまだ授業で習っていない場合は，「未学習」を○で囲んでください．

1. デスクトップの操作
   (a) メインメニューからアプリケーションを選択する.....**全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (b) デスクトップのアイコンをダブルクリックして開く.....**全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (c) 開いたウィンドウを移動する.....**全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (d) ウィンドウの大きさを変更する.....**全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (e) 隠れているウィンドウを前面に出す.....**全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
2. キーボード操作 (タッチタイピング)
   (a) ホームポジションに指を置く.....**全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (b) 正しい指を使ってとキーを押す.....**全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (c) キーボードを見ずにタイピングする.....**全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
3. 日本語入力
   (a) 日常的な文章を入力する.....**全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (b) かな漢字変換辞書に単語を登録する.....**全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**

(c) 知らない漢字を検索する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**

4. ファイル・ディレクトリの操作
   (a) 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**
   (b) ファイルやディレクトリを移動する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (c) ファイルやディレクトリを削除する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (d) ファイルやディレクトリの名称を変更する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (e) 新規ディレクトリを作成する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (f) ファイルやディレクトリの容量を調べる ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
5. 文書ドキュメント作成の操作 (StarSuite Writer)
   (a) Writer で文字の大きさを変更する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (b) Writer で段落を中央寄せや右寄せ、左寄せする ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (c) Writer で文書の中に画像を挿入する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (d) Web ブラウザ上の文章を Writer の文書に挿入する .... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (e) Writer で作成した文書を保存する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (f) 保存されている文書を Writer で開く ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
6. 電子メールの操作
   (a) 自分宛のメールを読む ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (b) メールを削除する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (c) メールをメールボックスに分けて整理する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (d) 添付ファイル付きメールの添付ファイルを読む ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (e) テキスト形式で電子メールを送る ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (f) 添付ファイルを付けてメールを送る ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (g) 同時に複数の人宛てにメールを送る ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (h) シグニチャを付加してメールを送る ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (i) 本文の一行文字数が長くならないようにメールを書く
   ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (j) 相手のメールを引用して返事のメールを送る ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (k) 送られて来たメールを他の人に転送する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
7. ドローソフト (StarSuite Draw)
   (a) 簡単な図形を描く ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (b) 図形の線や塗りつぶしの色を変更する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (c) 一度置いた図形を移動する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (d) 一度置いた図形を削除する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (e) 複数の図形をまとめて選択する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
8. ペイントソフト (GIMP)
   (a) 簡単な画像を描く ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (b) 筆 (ブラシ) の線幅や色を変更する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (c) 画像をファイルとして保存する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (d) 作図ソフトと描画ソフトの違いがわかる ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (e) XCF 形式、JPEG 形式、GIF 形式を互いに変換できる
   ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
9. Web ページ
   (a) クリック操作で Web ページを見てまわる ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**

(b) URL を直接入力して Web ページを見る ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
(c) 見たい Web ページを検索する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
(d) ブックマークに Web ページを登録する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
(e) HTML を用いて簡単な Web ページを作成する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
(f) HTML タグを用いて箇条書きする ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
(g) Web ページに画像を挿入する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
(h) Web ページに別のページへのリンクを作成する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**

10. 表計算
(a) スプレッドシートにデータを入力する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
(b) 式や関数を使った計算をする ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
(c) グラフを作成する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
(d) セルの内容をコピー&ペーストできる ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
(e) 絶対参照と相対参照を使い分ける ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
(f) 行や列の追加ができる ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
(g) CSV 形式のファイルを読み込むことができる ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**

11. プレゼンテーション (Applix Presentation)
(a) 文字中心のプレゼンテーションファイルを作る ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
(b) プレゼンテーションに図やグラフを入れる ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
(c) アウトライン画面で編集をする ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**

12. テキストファイルの日本語文字コードを変換する ..... **全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**

質問 3. コンピュータについていろいろな意見がありますが，以下の項目に対してあなたはどう思いますか？ それぞれの項目についてあなたの意見にもっとも近いと思われる数値を○で囲んでください．項目は裏面に続いています。すべての項目にご回答ください。

1. 私はコンピュータを操作するのは不安だ ..... **全くそうでない** 1 2 3 4 5 **全くそうだ**
2. 私はコンピュータを利用するとき，操作を誤って何かを壊しそうな気がする **全くそうでない** 1 2 3 4 5 **全くそうだ**
3. 私は就職してコンピュータを操作するような職場にまわされるかもしれないと考えると不安になる ..... **全くそうでない** 1 2 3 4 5 **全くそうだ**
4. 私はコンピュータをうまく操作できない人を見ると親しみを感じる **全くそうでない** 1 2 3 4 5 **全くそうだ**
5. 私は人がみている前でコンピュータの操作をすると恥をかきそうだ **全くそうでない** 1 2 3 4 5 **全くそうだ**
6. 私はコンピュータと聞いただけでもうお手上げの気持ちだ ..... **全くそうでない** 1 2 3 4 5 **全くそうだ**
7. 私は自信を持ってコンピュータを操作できる ..... **全くそうでない** 1 2 3 4 5 **全くそうだ**
8. 私は自分の操作に対して, コンピュータがどんな反応をするか予測できる . **全くそうでない** 1 2 3 4 5 **全くそうだ**
9. 私はコンピュータをうまく操作できなくても，いつかはできるようになると思う **全くそうでない** 1 2 3 4 5 **全くそうだ**
10. 私はコンピュータは難しそうなので手を出そうとは思わない ..... **全くそうでない** 1 2 3 4 5 **全くそうだ**
11. 私はコンピュータの操作をどうしていいのかわからなくなっても、なんとか使えると思う ..... **全くそうでない** 1 2 3 4 5 **全くそうだ**
12. 私はコンピュータで予期せぬ問題が発生しても、どうにかできると思う **全くそうでない** 1 2 3 4 5 **全くそ**

うだ

13. 私はコンピュータに関する専門用語を容易に理解できると思う.....**全くそうでない** 1 2 3 4 5 **全くそうだ**

質問 4. 現代社会におけるコンピュータやインターネットの関わりについて次のようなことが言われていますが，以下の項目に対してあなたはどう思いますか？ それぞれの項目についてあなたの意見にもっとも近いと思われる数値を○で囲んでください.

1. インターネット上では自分の実名を出されて中傷されそうだ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
2. インターネットを利用すると知らない間に自分の個人情報が流出しそうだ **全くそうでない** 1 2 3 4 5 **全くそうだ**
3. インターネットを利用すると犯罪に巻き込まれそうだ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
4. インターネット上での買い物は，偽物をつかまされそうだ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
5. インターネットができなくなると生活が成り立たなくなりそうだ...**全くそうでない** 1 2 3 4 5 **全くそうだ**
6. インターネットなしでは，いろいろな活動ができない気がする.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
7. インターネットがあらゆる活動の中心になっていくような気がする **全くそうでない** 1 2 3 4 5 **全くそうだ**
8. インターネットは生活の一部になっている.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
9. 他にすることがあるのについついインターネットを利用してしまう **全くそうでない** 1 2 3 4 5 **全くそうだ**
10. インターネットを利用していて睡眠不足になってしまうことがある **全くそうでない** 1 2 3 4 5 **全くそうだ**
11. インターネットは複雑でよくわからない.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
12. インターネットを使いこなせないと不利益をこうむってしまいそうだ...**全くそうでない** 1 2 3 4 5 **全くそうだ**
13. インターネットを利用すると迷惑メールやウイルスをもらいそうだ **全くそうでない** 1 2 3 4 5 **全くそうだ**
14. インターネットを使用すると知らない間にハッキングされるかもしれないと思う**全くそうでない** 1 2 3 4 5 **全くそうだ**
15. メールでは真意が伝わらないような気がする.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
16. メールだと相手の反応がウソか本当かわからない.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
17. 人と会って話すより，メールのほうが自分の意思がよく伝わると思う...**全くそうでない** 1 2 3 4 5 **全くそうだ**
18. メールが送信した相手にきちんと届いているかどうか不安だ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
19. メールを送信しても，間違った相手に送信してしまうのではないかと不安になる**全くそうでない** 1 2 3 4 5 **全くそうだ**
20. メールは他の誰かに読まれているような気がする.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
21. メールの返信が遅いと不安になる.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
22. 携帯電話が手元にないと不安になる.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
23. 暇があればつい携帯電話を見てしまう.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
24. 携帯電話がないと，取り残された気分になる.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
25. 自分の知らないところで自分のことが扱われているような気がする **全くそうでない** 1 2 3 4 5 **全くそうだ**
26. これからの社会では、誰かが歪んだ情報を流しそうな気がする.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
27. インターネットを利用することによって，本当に必要な情報を見失いそうだ **全くそうでない** 1 2 3 4 5 **全くそうだ**
28. インターネットの普及によって，有害な情報が蔓延しそうだ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
29. そのうちインターネットによって私たちの生活がコントロールされそうだ **全くそうでない** 1 2 3 4 5 **全く**

そうだ

30. 情報を持っている人だけが豊かになる社会になりそうな気がする...**全くそうでない** 1 2 3 4 5 **全くそうだ**
31. コンピュータを使っていると，生身の人間とのコミュニケーションが下手になっていく気がする ...**全くそうでない** 1 2 3 4 5 **全くそうだ**
32. コンピュータの普及によって，人間関係が希薄になっていくような気がする **全くそうでない** 1 2 3 4 5 **全くそうだ**
33. インターネットを使っていると，人に会う機会が減ってしまいそうだ...**全くそうでない** 1 2 3 4 5 **全くそうだ**
34. 携帯電話が圏外になると落ち着かない ...**全くそうでない** 1 2 3 4 5 **全くそうだ**
35. インターネットが利用できない場所にいると，不安になる.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
36. インターネットを活用できないと，社会から取り残されそうだ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**

質問 5. この後行われる中間試験で、あなたは何点取れると思いますか？

(　　) 点

質問 6. 情報活用基礎で学んだ授業内容で，もっと詳しく取り上げて欲しいと思ったものは何ですか？

B.3　期末アンケート

習熟度に関する調査

この調査は，情報活用基礎の授業で学んだ内容を自己評価するためのものです．成績には一切関係しないので，各自思う通りに回答してください．**裏面にも問題があるので**，忘れずに回答してください．よろしくお願いします．

質問 0．あなたの所属学部，学年，性別，学籍番号を入力してください．

学部:(　　)　学年:(　　)　性別:(　　)　学籍番号:(　　)　第 (　　) 教室

質問 1．以下の質問についてあなたはどの程度あてはまっていますか．あなたの意見に最も該当すると思う数値に○をつけてください．

1. 授業の進度は自分にあっていると思う．.....**全くあてはまらない** 1 2 3 4 5 **非常にあてはまる**
2. 授業内容を十分理解できていると思う．.....**全くあてはまらない** 1 2 3 4 5 **非常にあてはまる**
3. 自分はクラスの中で授業内容の理解レベルが劣っているとは思わない．
   .....**全くあてはまらない** 1 2 3 4 5 **非常にあてはまる**
4. 自分が理解できないことはクラスの大半の人も理解できていないと思う．
   .....**全くあてはまらない** 1 2 3 4 5 **非常にあてはまる**
5. 授業内容について友達と一緒に勉強した .....**全くあてはまらない** 1 2 3 4 5 **非常にあてはまる**
6. 授業内容について友達によく質問した .....**全くあてはまらない** 1 2 3 4 5 **非常にあてはまる**
7. 授業内容を教えてくれる友達がいた .....**全くあてはまらない** 1 2 3 4 5 **非常にあてはまる**
8. わからないことがあると，友達に質問をした .....**全くあてはまらない** 1 2 3 4 5 **非常にあてはまる**
9. 授業内容について，友達から質問を受けることがしばしばあった
   .....**全くあてはまらない** 1 2 3 4 5 **非常にあてはまる**
10. 友達同士で教え合うことがしばしばあった .....**全くあてはまらない** 1 2 3 4 5 **非常にあてはまる**

質問 2．あなたは次に挙げた各項目に対してどの程度習熟していますか．該当する数値を○で囲んでください．項目の内容をまだ授業で習っていない場合は，「未学習」を○で囲んでください．項目は裏面に続いています．すべての項目にご回答ください．

1. デスクトップの操作
   (a) メインメニューからアプリケーションを選択する.....**全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (b) デスクトップのアイコンをダブルクリックして開く.....**全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (c) 開いたウィンドウを移動する.....**全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**
   (d) ウィンドウの大きさを変更する.....**全く未習熟** 1 2 3 4 5 **完全に習得**　**未学習**

(e) 隠れているウィンドウを前面に出す ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**

2. キーボード操作 (タッチタイピング)
   (a) ホームポジションに指を置く ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (b) 正しい指を使ってとキーを押す ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (c) キーボードを見ずにタイピングする ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
3. 日本語入力
   (a) 日常的な文章を入力する ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (b) かな漢字変換辞書に単語を登録する ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (c) 知らない漢字を検索する ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
4. ファイル・ディレクトリ (フォルダ) の操作
   (a) 指定されたディレクトリの中を見る ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (b) ファイルやディレクトリを移動する ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (c) ファイルやディレクトリを削除する ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (d) ファイルやディレクトリの名称を変更する ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (e) 新規ディレクトリを作成する ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (f) ファイルやディレクトリの容量を調べる ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
5. 文書作成の操作 (StarSuite Writer)
   (a) Writer で文字の大きさを変更する ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (b) Writer で段落を中央寄せや右寄せ、左寄せする ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (c) Writer で文書の中に画像を挿入する ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (d) Web ブラウザ上の文章を Writer の文書に挿入する .... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (e) Writer で作成した文書を保存する ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (f) 保存されている文書を Writer で開く ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
6. 電子メールの操作
   (a) 自分宛のメールを読む ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (b) メールを削除する ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (c) メールをメールボックス分けて整理する ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (d) 添付ファイル付きメールの添付ファイルを読む ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (e) テキスト形式で電子メールを送る ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (f) 添付ファイルを付けてメールを送る ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (g) 同時に複数の人宛てにメールを送る ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (h) シグニチャを付加してメールを送る ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (i) 本文の一行文字数が長くならないようにメールを書く
   ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (j) 相手のメールを引用して返事のメールを送る ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (k) 送られて来たメールを他の人に転送する ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
7. 作図ソフト (StarSuite Draw)
   (a) 簡単な図形を描く ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (b) 図形の線や塗りつぶしの色を変更する ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (c) 一度置いた図形を移動する ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (d) 一度置いた図形を削除する ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
   (e) 複数の図形をまとめて選択する ..... **全く未習熟** 1 2 3 4 5 **完全に習得** **未学習**
8. 描画ソフト (GIMP) など

(a) 簡単な画像を描く ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(b) 筆 (ブラシ) の線幅や色を変更する ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(c) 画像をファイルとして保存する ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(d) 作図ソフトと描画ソフトの違いがわかる ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(e) XCF 形式、JPEG 形式、GIF 形式を互いに変換できる
..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**

9. Web ページ
(a) クリック操作で Web ページを見てまわる ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(b) URL を直接入力して Web ページを見る ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(c) 見たい Web ページを検索する ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(d) ブックマークに Web ページを登録する ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(e) HTML を用いて簡単な Web ページを作成する ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(f) HTML タグを用いて箇条書きする ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(g) Web ページに画像を挿入する ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(h) Web ページに別のページへのリンクを作成する ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**

10. 表計算 (StarSuite Calc)
(a) スプレッドシートにデータを入力する ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(b) 式や関数を使った計算をする ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(c) グラフを作成する ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(d) セルの内容をコピー&ペーストできる ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(e) 絶対参照と相対参照を使い分ける ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(f) 行や列の追加ができる ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(g) CSV 形式のファイルを読み込むことができる ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**

11. 自分の印刷出力をキャンセル ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**

12. プレゼンテーション (StarSuite Impress)
(a) 文字中心のプレゼンテーションファイルを作る ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(b) プレゼンテーションに図やグラフを入れる ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(c) アウトライン画面で編集をする ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**

13. テキストファイルの日本語文字コードを変換する ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**

14. プログラミング
(a) 簡単な図形を表示するプログラムを作成する ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(b) アニメーションを表示するプログラムを作成する ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(c) 繰り返しのプログラムを作成する ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**
(d) 条件分岐のプログラムを作成する ..... **全く未習熟** 1 2 3 4 5 **完全に習得　未学習**

質問 3. コンピュータについていろいろな意見がありますが，以下の項目に対してあなたはどう思いますか？ それぞれの項目についてあなたの意見にもっとも近いと思われる数値を○で囲んでください．項目は裏面に続いています。すべての項目にご回答ください。

1. 私はコンピュータを操作するのは不安だ ..... **全くそうでない** 1 2 3 4 5 **全くそうだ**
2. 私はコンピュータを利用するとき，操作を誤って何かを壊しそうな気がする
..... **全くそうでない** 1 2 3 4 5 **全くそうだ**

3. 私は就職してコンピュータを操作するような職場にまわされるかもしれないと考えると不安になる
.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
4. 私はコンピュータをうまく操作できない人を見ると親しみを感じる **全くそうでない** 1 2 3 4 5 **全くそうだ**
5. 私は人がみている前でコンピュータの操作をすると恥をかきそうだ **全くそうでない** 1 2 3 4 5 **全くそうだ**
6. 私はコンピュータと聞いただけでもうお手上げの気持ちだ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
7. 私は自信を持ってコンピュータを操作できる.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
8. 私は自分の操作に対して, コンピュータがどんな反応をするか予測できる
.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
9. 私はコンピュータをうまく操作できなくても，いつかはできるようになると思う
.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
10. 私はコンピュータは難しそうなので手を出そうとは思わない.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
11. 私はコンピュータの操作をどうしていいのかわからなくなっても、なんとか使えると思う
.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
12. 私はコンピュータで予期せぬ問題が発生しても、どうにかできると思う **全くそうでない** 1 2 3 4 5 **全くそうだ**
13. 私はコンピュータに関する専門用語を容易に理解できると思う.....**全くそうでない** 1 2 3 4 5 **全くそうだ**

質問 4. 現代社会におけるコンピュータやインターネットの関わりについて次のようなことが言われていますが，以下の項目に対してあなたはどう思いますか? それぞれの項目についてあなたの意見にもっとも近いと思われる数値を○で囲んでください.

1. インターネット上では自分の実名を出されて中傷されそうだ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
2. インターネットを利用すると知らない間に自分の個人情報が流出しそうだ
.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
3. インターネットを利用すると犯罪に巻き込まれそうだ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
4. インターネット上での買い物は，偽物をつかまされそうだ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
5. インターネットができなくなると生活が成り立たなくなりそうだ...**全くそうでない** 1 2 3 4 5 **全くそうだ**
6. インターネットなしでは，いろいろな活動ができない気がする.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
7. インターネットがあらゆる活動の中心になっていくような気がする **全くそうでない** 1 2 3 4 5 **全くそうだ**
8. インターネットは生活の一部になっている.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
9. 他にすることがあるのについついインターネットを利用してしまう **全くそうでない** 1 2 3 4 5 **全くそうだ**
10. インターネットを利用していて睡眠不足になってしまうことがある **全くそうでない** 1 2 3 4 5 **全くそうだ**
11. インターネットは複雑でよくわからない.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
12. インターネットを使いこなせないと不利益をこうむってしまいそうだ...**全くそうでない** 1 2 3 4 5 **全くそうだ**
13. インターネットを利用すると迷惑メールやウイルスをもらいそうだ **全くそうでない** 1 2 3 4 5 **全くそうだ**
14. インターネットを使用すると知らない間にハッキングされるかもしれないと思う
.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
15. メールでは真意が伝わらないような気がする.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
16. メールだと相手の反応がウソか本当かわからない.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
17. 人と会って話すより，メールのほうが自分の意思がよく伝わると思う...**全くそうでない** 1 2 3 4 5 **全くそうだ**

18. メールが送信した相手にきちんと届いているかどうか不安だ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
19. メールを送信しても，間違った相手に送信してしまうのではないかと不安になる
.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
20. メールは他の誰かに読まれているような気がする.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
21. メールの返信が遅いと不安になる.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
22. 携帯電話が手元にないと不安になる.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
23. 暇があればつい携帯電話を見てしまう.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
24. 携帯電話がないと，取り残された気分になる.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
25. 自分の知らないところで自分のことが扱われているような気がする **全くそうでない** 1 2 3 4 5 **全くそうだ**
26. これからの社会では、誰かが歪んだ情報を流しそうな気がする.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
27. インターネットを利用することによって，本当に必要な情報を見失いそうだ
.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
28. インターネットの普及によって，有害な情報が蔓延しそうだ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
29. そのうちインターネットによって私たちの生活がコントロールされそうだ
.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
30. 情報を持っている人だけが豊かになる社会になりそうな気がする...**全くそうでない** 1 2 3 4 5 **全くそうだ**
31. コンピュータを使っていると，生身の人間とのコミュニケーションが下手になっていく気がする
.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
32. コンピュータの普及によって，人間関係が希薄になっていくような気がする
.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
33. インターネットを使っていると，人に会う機会が減ってしまいそうだ...**全くそうでない** 1 2 3 4 5 **全くそうだ**
34. 携帯電話が圏外になると落ち着かない.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
35. インターネットが利用できない場所にいると，不安になる.....**全くそうでない** 1 2 3 4 5 **全くそうだ**
36. インターネットを活用できないと，社会から取り残されそうだ.....**全くそうでない** 1 2 3 4 5 **全くそうだ**

質問 5. この後行われる期末試験で、あなたは何点取れると思いますか？

(　　) 点

質問 6. 情報活用基礎の感想を余白に記入してください.

付録 C

2006 年度文学部・人間科学部「情報活用基礎」授業後アンケート

C.1 『情報活用基礎』授業アンケート (PEN 第 1 回)

1. あなたのログイン名を書いてください
2. プログラミングの経験について
   プログラミングの経験があれば、どのような言語を知っていてどのくらいの経験があるのかを簡単に書いてください。経験のあるすべての言語について書いてください。(例: Visual Basic を高校の数学の授業で (50 分)8 コマ程習って、数十行程度のプログラムを書いたことがある。)
3. プログラミング以外のコンピュータの利用経験について
   コンピュータでどんなことが出来ますか？ 簡単に書いて下さい。(例: ワープロで簡単な文書作成ができる。Web を見る。表計算ソフトで簡単な表を作成できる。)
4. 今日の授業の理解度は
   理解できた / だいたい理解できたと思う / あまり理解できていないと思う / ほとんど理解できなかった
5. 各々の例題、練習問題はどの程度できましたか
   5.1 例 01 (入力の 3 倍を求める)
   理解できた / 大体理解できたと思う / あまり理解できていないと思う / 全く分からなかった（5.4 の選択肢も同じ）
   5.2 練習 01-01 (積と商)
   ほぼ自力でプログラムを作ることができた / 少し教えてもらって、できた / ほとんど教えてもらって、なんとかできた / できなかった / 時間がなくて、できなかった（5.3,5.5,5.6 の選択肢も同じ）
   5.3 練習 01-02 (三角形の面積)
   5.4 例 02 (3 の倍数の判定)
   5.5 練習 02-01 (成績判定)
   5.6 練習 02-02 (大小判定)
6. PEN でのプログラミングに関し各項目について理解度を答えてください
   6.1 変数とはどんなものか。変数への代入／変数を使った計算など
   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（6.2 の選択肢も同じ）
   6.2 条件分岐（もし〜ならば）の使い方
7. 全体的に授業は面白かったですか
   とても面白かった / どちらかというと面白かった / どちらかというと面白くなかった / 面白くなかった
8. その他、面白かったところ、難しかった点など、自由にご意見をご記入ください。

C.2 『情報活用基礎』授業アンケート (PEN 第 2 回)

1. あなたのログイン名を書いてください
2. 受講している教室はどこですか？
3. 各々の例題、練習問題はどの程度できましたか
   - 3.1 例 02 (3 の倍数の判定) (先週の授業資料の例題)
     先週、アンケートに回答済 / 理解できた / 大体理解できたと思う / あまり理解できていないと思う / 全く分からなかった
   - 3.2 練習 02-01 (成績判定) (先週の授業資料の練習問題)
     先週、アンケートに回答済 / ほぼ自力でプログラムを作ることができた / 少し教えてもらって、できた / ほとんど教えてもらって、なんとかできた / できなかった / 時間がなくて、できなかった（3.3 の選択肢も同じ）
   - 3.3 練習 02-02 (大小判定) (先週の授業資料の練習問題)
   - 3.4 例 03 (階乗を求める)
     理解できた / 大体理解できたと思う / あまり理解できていないと思う / 全く分からなかった（3.8 の選択肢も同じ）
   - 3.5 練習 03-02 (10 から 20 までの 2 乗)
     ほぼ自力でプログラムを作ることができた / 少し教えてもらって、できた / ほとんど教えてもらって、なんとかできた / できなかった / 時間がなくて、できなかった（3.6,3.7,3.9,3.10 の選択肢も同じ）
   - 3.6 練習 03-03 (1 から 100 までの数のうち 3 の倍数の和)
   - 3.7 練習 03-05 (九九の表)
   - 3.8 例 04 (温度変換 (F → C))
   - 3.9 練習 04-01 (BMI 指数)
   - 3.10 練習 04-02 (利息計算)
4. PEN でのプログラミングに関し各項目について理解度を答えてください
   - 4.1 変数とはどんなものか。変数への代入／変数を使った計算などについて
     理解できていると思う / 先週よりは理解が進んだ / 先週と同じくらいで、余り理解できていない / 理解できていない（以下の選択肢も同じ）
   - 4.2 条件分岐（もし〜ならば）の使い方
   - 4.3 繰り返し (... の間... を繰り返す) の使い方
   - 4.4 実数の扱い方
   - 4.5 整数と実数の扱いについて
5. 全体的に授業は面白かったですか
   とても面白かった / どちらかというと面白かった / どちらかというと面白くなかった / 面白くなかった
6. 今日の授業の理解度は
   理解できた / だいたい理解できたと思う / あまり理解できていないと思う / ほとんど理解できなかった
7. 先週に比べて理解は進みましたか？
   先週も理解できていた / 先週よりも理解が進んで、少し分かるようになってきた先週と同じくらいで、理解できていないところがある / 先週はある程度理解していたのに、今週はあまり理解できていないと思う / 先週も今週もほとんど理解できていない
8. 自力でプログラムが書けそうですか？
   今くらいの問題なら自力で書けると思う / ちょっとヒントをもらえたら書けると思う / ちょっと自信がない / 書けそうな気がしない
9. プログラミングって何をするのかわかりましたか？
   わかった / だいたいわかった / あまりよくわからない / まったくわからない
10. その他、面白かったところ、難しかった点など、自由にご意見をご記入ください。

C.3 『情報活用基礎』授業アンケート (PEN 第 3 回)

1. あなたのログイン名を書いてください

2. 受講している教室はどこですか？
3. 各々の例題、練習問題はどの程度できましたか
   - 3.1 例 3 (階乗の計算) (先週の授業資料の例題)
     先週、アンケートに回答済 / 理解できた / 大体理解できたと思う / あまり理解できていないと思う / 全く分からなかった
   - 3.2 練習 03-02 (10〜20 の 2 乗) (先週の授業資料の練習問題)
     先週、アンケートに回答済 / ほぼ自力でプログラムを作ることができた / 少し教えてもらって、できた / ほとんど教えてもらって、なんとかできた / できなかった / 時間がなくて、できなかった（3.3 の選択肢も同じ）
   - 3.3 練習 03-03 (1〜100 の 3 の倍数の和) (先週の授業資料の練習問題)
   - 3.4 例題 (1) (基本図形の描画)
     理解できた / 大体理解できたと思う / あまり理解できていないと思う / 全く分からなかった / 授業で説明がなかった（3.5 の選択肢も同じ）
   - 3.5 例題 (2) (アニメーション)
   - 3.6 練習 05-01 (日の丸の描画)
     ほぼ自力でプログラムを作ることができた / 少し教えてもらって、できた / ほとんど教えてもらって、なんとかできた / できなかった / 時間がなくて、できなかった（3.7,3.8 の選択肢も同じ）
   - 3.7 練習 05-02 (3 本線の描画)
   - 3.8 練習 05-03 (方眼紙の描画)
4. PEN でのプログラミングに関し各項目について理解度を答えてください
   - 4.1 変数とはどんなものか。変数への代入／変数を使った計算などについて
     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（以下の選択肢も同じ）
   - 4.2 条件分岐（もし〜ならば）の使い方
   - 4.3 繰り返し (... の間... を繰り返す) の使い方
   - 4.4 実数と整数の違い
   - 4.5 図形の描画
5. プログラミングの授業について
   - 5.1 今日の授業の理解度は？
     理解できた / だいたい理解できたと思う / あまり理解できていないと思う / ほとんど理解できなかった
   - 5.2 先週に比べて理解は進みましたか？
     先週も理解できていた / 先週よりも理解が進んで、少し分かるようになってきた / 先週と同じくらいで、理解できていないところがある / 先週はある程度理解していたのに、今週はあまり理解できていないと思う / 先週も今週もほとんど理解できていない
   - 5.3 自力でプログラムが書けそうですか？
     今くらいの問題なら自力で書けると思う / ちょっとヒントをもらえたら書けると思う / ちょっと自信がない / 書けそうな気がしない
   - 5.4 プログラミングとは何をするものかがわかりましたか？
     わかった / だいたいわかった / あまりよくわからない / まったくわからない
   - 5.5 プログラミングは面白かったですか？
     とても面白かった / 意外と面白かった / あまり面白くなかった / 二度とやりたくない
   - 5.6 以下の項目の中で、同意するすべての項目にチェックを入れて下さい。
     銀行のオンラインシステムの障害や、東京証券取引所でのトラブルなど、世の中でコンピュータに関連するトラブルが起きているが、何故、このようなトラブルが起きるかということが、少しは理解できるようになった。 / コンピュータはプログラムに書かれたことのみを忠実に実行するだけであるということがわかった。 / コンピュータの能力は凄いと再認識した。 / コンピュータの能力は大したことはない... という面もあるということがわかった。 / プログラミングなどやる必要ないと思っていたが、意外と面白かった。 / プログラミングなどやる必要ないと思っていたし、授業後もその気持は変わらない。 / プログラミングよりも、ワープロ等の使い方をもっと詳しく教えて欲しかった。 / あまり理解できなかった。 / さっぱりわからなかった。2 度とやりたくない。
   - 5.7 プログラミングを経験することで、コンピュータに対する認識がどのように変わったか、自由に書いて下さい。
6. その他：面白かったところ、難しかった点、感想など、自由にご意見をご記入ください。

C.4 『情報活用基礎』授業アンケート (JavaScript 第 1 回)

1. あなたのログイン名を書いてください
2. 受講している教室は何処ですか？
3. プログラミングの経験について
   プログラミングの経験があれば、どのような言語を知っていてどのくらいの経験があるのかを簡単に書いてください。経験のあるすべての言語について書いてください。(例: Visual Basic を高校の数学の授業で (50 分)8 コマ程習って、数十行程度のプログラムを書いたことがある。)
4. プログラミング以外のコンピュータの利用経験について
   コンピュータでどんなことが出来ますか？ 簡単に書いて下さい。(例: ワープロで簡単な文書作成ができる。Web を見る。表計算ソフトで簡単な表を作成できる。)
5. JavaScript でのプログラミングに関し各項目について理解度を答えてください
   5.1 変数とはどんなものか。変数への代入／変数を使った計算など
   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（5.2 の選択肢も同じ）
   5.2 条件分岐 if 文 の使い方
6. 練習問題はどの程度できましたか
   6.1 練習 1-1 条件分岐 (1)
   ほぼ自力でプログラムを作ることができた / 少し教えてもらって、できた / ほとんど教えてもらって、なんとかできた / できなかった / 時間がなくて、できなかった（6.2 の選択肢も同じ）
   6.2 練習 1-2 条件分岐 (2)
7. 今日の授業の理解度は
   理解できた / だいたい理解できたと思う / あまり理解できていないと思う / ほとんど理解できなかった
8. 全体的に授業は面白かったですか
   とても面白かった / どちらかというと面白かった / どちらかというと面白くなかった / 面白くなかった
9. 自力でプログラムが書けそうですか？
   今くらいの問題なら自力で書けると思う / ちょっとヒントをもらえたら書けると思う / ちょっと自信がない / 書けそうな気がしない
10. プログラミングって何をするのかわかりましたか？わかった / だいたいわかった / あまりよくわからない / まったくわからない
11. その他、面白かったところ、難しかった点など、自由にご意見をご記入ください。

C.5 『情報活用基礎』授業アンケート (JavaScript 第 2 回)

1. あなたのログイン名を書いてください
2. 受講している教室は？
3. 練習問題はどの程度できましたか
   3.1 練習 2-1 繰り返し (1) — 1〜100 までの和
   ほぼ自力でプログラムを作ることができた / 少し教えてもらって、できた / ほとんど教えてもらって、なんとかできた / できなかった / 時間がなくて、できなかった（以下の選択肢も同じ）
   3.2 練習 2-2 繰り返し (2) — 1〜100 までの 3 の倍数の和
   3.3 練習 2-3 繰り返し (3) — 約数を求める
4. JavaScript でのプログラミングに関し各項目について理解度を答えてください
   4.1 繰り返し (while 文) の使い方
   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 / 授業で説明がなかった（4.2 の選択肢も同じ）
   4.2 関数の使い方
5. プログラミングの授業について
   5.1 今日の授業の理解度は?
   理解できた / だいたい理解できたと思う / あまり理解できていないと思う / ほとんど理解できなかった
   5.2 先週に比べて理解は進みましたか？

先週も理解できていた / 先週よりも理解が進んで、少し分かるようになってきた / 先週と同じくらいで、理解できていないところがある / 先週はある程度理解していたのに、今週はあまり理解できていないと思う / 先週も今週もほとんど理解できていない

5.3 全体的に授業は面白かったですか
とても面白かった / どちらかというと面白かった / どちらかというと面白くなかった / 面白くなかった

5.4 自力でプログラムが書けそうですか？
今くらいの問題なら自力で書けると思う / ちょっとヒントをもらえたら書けると思う / ちょっと自信がない / 書けそうな気がしない

5.5 プログラミングとは何をするものかがわかりましたか？
わかった / だいたいわかった / あまりよくわからない / まったくわからない

5.6 プログラミングは面白かったですか？
とても面白かった / 意外と面白かった / あまり面白くなかった / 二度とやりたくない

6. その他: 面白かったところ、難しかった点、感想など、自由にご意見をご記入ください。

C.6 『情報活用基礎』授業アンケート (JavaScript 第 3 回)

1. あなたのログイン名を書いてください
2. 受講している教室は？
3. 例題、練習問題はどの程度できましたか

3.1 イベントハンドラの使用例 (2) — ボタンラベルの変更 —
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 / 授業で説明がなかった（3.2 の選択肢も同じ）

3.2 イベントハンドラの使用例 (4) — cm から inch への変換 —

3.3 練習 3-1 イベントハンドラ (1) — BMI —
ほぼ自力でプログラムを作ることができた / 少し教えてもらって、できた / ほとんど教えてもらって、なんとかできた / できなかった / 時間がなくて、できなかった（3.4 の選択肢も同じ）

3.4 練習 3-2 イベントハンドラ (1) — BMI の拡張 —

4. JavaScript でのプログラミングに関し各項目について理解度を答えてください

4.1 変数とはどんなものか。変数への代入／変数を使った計算などについて
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（他の選択肢も同じ）

4.2 条件分岐（if 文）の使い方

4.3 繰り返し (while 文) の使い方

4.4 フォームの使い方

4.5 イベントハンドラの使い方

5. プログラミングの授業について

5.1 今日の授業の理解度は?
理解できた / だいたい理解できたと思う / あまり理解できていないと思う / ほとんど理解できなかった

5.2 先週に比べて理解は進みましたか？
先週も理解できていた / 先週よりも理解が進んで、少し分かるようになってきた / 先週と同じくらいで、理解できていないところがある / 先週はある程度理解していたのに、今週はあまり理解できていないと思う / 先週も今週もほとんど理解できていない

5.3 自力でプログラムが書けそうですか？
今くらいの問題なら自力で書けると思う / ちょっとヒントをもらえたら書けると思う / ちょっと自信がない / 書けそうな気がしない

5.4 プログラミングとは何をするものかがわかりましたか？
わかった / だいたいわかった / あまりよくわからない / まったくわからない

5.5 プログラミングは面白かったですか？
とても面白かった / 意外と面白かった / あまり面白くなかった / 二度とやりたくない

5.6 以下の項目の中で、同意するすべての項目にチェックを入れて下さい。
銀行のオンラインシステムの障害や、東京証券取引所でのトラブルなど、世の中でコンピュータに関連するトラブルが起きているが、何故、このようなトラブルが起きるかということが、少しは理解できるようになった。 / コンピュータはプログラムに書かれたことのみを忠実に実行するだけであるというこ

とがわかった。 / コンピュータの能力は凄いと再認識した。 / コンピュータの能力は大したことはない... という面もあるということがわかった。 / プログラミングなどやる必要ないと思っていたが、意外と面白かった。 / プログラミングなどやる必要ないと思っていたし、授業後もその気持は変わらない。 / プログラミングよりも、ワープロ等の使い方をもっと詳しく教えて欲しかった。 / あまり理解できなかった。 / さっぱりわからなかった。2 度とやりたくない。

5.7 プログラミングを経験することで、コンピュータに対する認識がどのように変わったか、自由に書いて下さい。

5.8 プログラミングの講義について、もっと時間を掛けて欲しかったところを 1 つだけ挙げるとすればどこですか。

6. その他: 面白かったところ、難しかった点、感想など、自由にご意見をご記入ください。

C.7 『情報活用基礎』授業アンケート (JavaScript 第 4 回)

1. あなたのログイン名を書いてください
2. 受講している教室は？
3. 演習問題はどの程度できましたか
   - 3.1 演習問題 4-2 (乱数)
     ほぼ自力でプログラムを作ることができた / 少し教えてもらって、できた / ほとんど教えてもらって、なんとかできた / できなかった / 時間がなくて、できなかった（他の選択肢も同じ）
   - 3.2 演習問題 4-3 (成績判定)
   - 3.3 演習問題 4-4 (数当てゲーム (1))
4. JavaScript でのプログラミングに関し各項目について理解度を答えてください
   - 4.1 変数とはどんなものか。変数への代入／変数を使った計算などについて
     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（4.2 の選択肢も同じ）
   - 4.2 条件分岐（if 文）の使い方
   - 4.3 繰り返し (while 文) の使い方理解できた / 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 / 授業で説明がなかった
5. プログラミングの授業について
   - 5.1 先週に比べて理解は進みましたか？先週も理解できていた / 先週よりも理解が進んで、少し分かるようになってきた / 先週と同じくらいで、理解できていないところがある / 先週はある程度理解していたのに、今週はあまり理解できていないと思う / 先週も今週もほとんど理解できていない
   - 5.2 自力でプログラムが書けそうですか？今くらいの問題なら自力で書けると思う / ちょっとヒントをもらえたら書けると思う / ちょっと自信がない / 書けそうな気がしない
   - 5.3 プログラミングとは何をするものかがわかりましたか？わかった / だいたいわかった / あまりよくわからない / まったくわからない
   - 5.4 プログラミングは面白かったですか？とても面白かった / 意外と面白かった / あまり面白くなかった / 二度とやりたくない
6. その他: 面白かったところ、難しかった点、感想など、自由にご意見をご記入ください。

謝辞

本研究をまとめるにあたり，その全過程において，懇切なるご指導とご支援をいただきました，大阪大学大学院情報科学研究科 増澤 利光 教授に心より深く感謝いたします．また，本論文作成にあたり，有益なご教示とご指摘をいただきました，大阪大学大学院情報科学研究科 萩原 兼一 教授，井上 克郎 教授に感謝を申し上げます．

本研究を共に進め，温かいご指導とご鞭撻を賜った，大阪市立大学大学院創造都市研究科 松浦 敏雄 教授に心より深く感謝いたします．

また，本研究を共に進め，さまざまな場面でご協力とご支援をいただいた，大阪市立大学 中村 亮太 氏，追手門大学 原田 章 教授，大阪工業大学 中西 通雄 教授，大阪大学 宮本 友介 助教に深く感謝いたします．

さらに，PEN の授業実践と評価に御協力頂いた大阪工業大学 安留 誠吾 准教授，大阪大学 清川 清 准教授，間下 以大 講師，外川 直子 氏，京都ノートルダム女子大学 吉田 智子 教授，名古屋高等学校 中西 渉 教諭，大和大学 松本 宗久 講師，大阪学院大学高等学校 横山 成彦 講師に感謝申し上げます．

加えて，本研究にさまざまなアドバイスとご援助をいただいた大阪電気通信大学 兼宗 進 教授，筆者の研究者としての基礎を築くあたって多岐にわたるご指導をいただいた京都工芸繊維大学 辻野 嘉宏 教授に感謝いたします．

最後に，筆者が研究・教育の道に進む礎を築いていただき，その活動において随所で，温かいご指導とご鞭撻を賜った，大阪大学 都倉 信樹 名誉教授に心より深く感謝いたします．また，本論文の執筆にあたり，温かく励まし支えてくれた妻 智子に感謝します．

参考文献

[1] 文部科学省: 高等学校学習指導要領(平成 11 年 3 月), <http://www.mext.go.jp/a_menu/shotou/cs/1320144.htm> (1999, 参照 2016-11-03).

[2] 文部科学省 情報化の進展に対応した初等中等教育における情報教育の推進等に関する調査研究協力者会議: 体系的な情報教育の実施に向けて（第 1 次報告）, <http://www.mext.go.jp/b_menu/shingi/chousa/shotou/002/toushin/971001.htm> (1997, 参照 2016-11-03).

[3] 文部科学省: 高等学校学習指導要領（平成 21 年 3 月）, <http://www.mext.go.jp/component/a_menu/education/micro_detail/__icsFiles/afieldfile/2011/03/30/1304427_002.pdf> (2009, 参照 2016-11-03).

[4] 文部科学省 中央教育審議会: 幼稚園、小学校、中学校、高等学校及び特別支援学校の学習指導要領等の改善及び必要な方策等について（答申）, <http://www.mext.go.jp/b_menu/shingi/chukyo/chukyo0/toushin/1380731.htm> (2016, 参照 2017-01-02).

[5] 文部科学省 小学校段階における論理的思考力や創造性、問題解決能力等の育成とプログラミング教育に関する有識者会議: 小学校段階におけるプログラミング教育の在り方について（議論の取りまとめ）, <http://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/attach/1372525.htm> (2016, 参照 2016-11-03).

[6] 内閣官房: 日本再興戦略 2016,
<http://www.kantei.go.jp/jp/singi/keizaisaisei/pdf/2016_hombun1.pdf>
<http://www.kantei.go.jp/jp/singi/keizaisaisei/pdf/2016_hombun2.pdf>
(2016, 参照 2016-11-05).

[7] 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.html> （参照 2016-08-03）.

[8] 中村亮太, 松浦敏雄, 西田知博: 初心者向きアルゴリズム学習環境の構築,
2004PC カンファレンス論文集,pp.102–103 (2004).

[9] 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コンピュータと教育 (CE), Vol.2005 − CE − 81, No.10, pp. 65–71,(2005).

[10] 西田知博, 原田章, 中村亮太, 宮本友介, 松浦敏雄: 初学者用プログラミング学習環境 PEN の実装と評価, 情報処理学会論文誌, Vol.48, No.8, pp.2736–2747 (2007).

[11] 文部科学省: 中学校学習指導要領（平成 20 年 3 月），<http://www.mext.go.jp/a_menu/shotou/new-cs/youryou/chu/__icsFiles/afieldfile/2010/12/16/121504.pdf> (2008, 参照 2016-11-06）.

[12] Arduino Home Page,<https://www.arduino.cc/>(参照 2016-11-07）.

[13] Tomohiro Nishida, Ryota Nakamura, Liu Lu, Chan Myae Thu and Toshio Matsuura: *Development of Learning Support Software and Educational Materials for Studying Measurement and Control by Programs*,ED-Media 2013, pp.108–114, (2013).

[14] 西田知博, 原田章, 中西通雄, 松浦敏雄: プログラミング入門教育における図形描画先行型のコースウェアが学習に与える影響, 情報処理学会論文誌教育とコンピュータ (TCE) （採録決定）.

[15] 西田知博, 原田章, 中西通雄, 松浦敏雄: プログラミング導入教育におけるコースウェアの違いによる学習効果の比較, 情報処理学会研究報告コンピュータと教育 (CE), Vol.2013 − CE − 122, No.2 (2013).

[16] MIT Media Lab: Scratch, <https://scratch.mit.edu/>(参照 2016-11-08）.

[17] 阿部和広: 子どもの創造的活動と ICT 活用, 情報処理, Vol.56, No.4, pp.350–354 (2015).

[18] 兼宗進, 御手洗理英, 中谷多哉子, 福井眞吾, 久野靖: 学校教育用オブジェクト指向言語「ドリトル」の設計と実装, 情報処理学会論文誌プログラミング, Vol.42, No.SIG11(PRO12), pp.78–90 (2001).

[19] 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).

[20] 原田康徳: プログラミング言語ビスケットを用いた基礎としてのプログラミング教育の提案と実践, 情報処理学会デジタルプラクティス, Vol.6, No.2, pp.105–111 (2015).

[21] H. Ueno : “An integrated knowledge-based intelligent programming environment for novice programmers”, Proc. IEEE Computer Software and Application Conf., pp.124–129(1991).

[22] G. Evangelidis, V. Dagdilelis, M. Satratzemi, V. Efopoulos: “X-Compiler: yet another integrated novice programming environment”, Proc. IEEE Int’l Conf. Advanced Learning Technologies, pp.166–169(2001).

[23] 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).

[24] 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).

[25] 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).

[26] 高橋延匡: 新しいプログラミング環境：プログラミング・パラダイムと環境：日本語プログラミング環境, 情報処理, Vol.30, No.4, pp.363–372 (1989).

[27] 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 (平成 4 年度報告書)（オンライン），<https://www.ipsj.or.jp/12kyoiku/monbu4-1.html> （参照 2016-08-03）.

[28] 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).

[29] 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).

[30] 独立行政法人大学入試センター: センター試験用手順記述標準言語 (DNCL) の説明, <http://www.dnc.ac.jp/albums/abm.php?f=abm00003020.pdf&n=H23_dncl.pdf>（参照 2016-08-03）.

[31] 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/>.

[32] 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).

[33] 情報活用基礎 プログラミング教育の記録データ – 2011〜2014 –, <http://www.s.ogu.ac.jp/pen/penData/>（参照 2016-08-03）.

[34] PEN を用いた授業 アンケート結果集, <http://www.s.osaka-gu.ac.jp/pen/enq/>(2006, 参照 2016-11-07）.

[35] 西田 知博, 横山 成彦: 高大連携による「情報の科学」の授業設計高大連携による「情報の科学」の授業設計, 日本情報科教育学会 第 7 回全国大会論文集 (2014).

[36] 川合 慧, 高岡　詠子, 西田 知博: 計算事始め, 放送大学教育振興会 (2013).

[37] 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).

[38] 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).

[39] 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, pp.1128(2002).

[40] 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).

[41] M. Nakanishi, A. Harada: “Reorganizing computer literacy classes in the middle of a term”, *Advanced Research in Computers and Communications in Education(2)*, pp.507–514 (1999).

[42] L. Beckwith, M. Burnett: “Gender: An Important Factor in End-User Programming Environment?”, Proc. 2004 IEEE Symp. Visual Languages and Human Centric Computing, pp.107–114 (2004).

[43] 西ヶ谷浩史, 兼宗進, 青木浩幸, 紅林秀治: アーム付き自律型移動ロボットを使った授業実践, 情報処理学会研究報告コンピュータと教育 (CE), Vol.2008-CE-93, No.13, pp. 17–23,(2008).

[44] 西ヶ谷浩史, 青木浩幸, 井上修次, 江口啓, 紅林秀治: 自律型 3 モータ制御ロボット教材を用いた計測の授業, 情報処理学会研究報告コンピュータと教育 (CE), Vol.2009-CE-98, No.15, pp.

113–120,(2009).

[45] 井戸坂幸男, 兼宗進, 久野靖: 中学校における自律制御ロボット教材の評価と授業, 情報処理学会研究報告コンピュータと教育 (CE), Vol.2010-CE-103, No.22, pp. 1–7,(2010).

[46] RASPBERRY PI FOUNDATION: Raspberry Pi ホームページ,<https://www.raspberrypi.org/> (参照 2016-11-08）.

[47] 安留誠吾, 中西通雄, 景村幸弘: 図形描画によるプログラミング入門 – 大阪大学人間科学部・文学部での PEN を用いた実践, 情報教育研究集会講演論文集, 2008 年度, pp.531–534 (2008).

[48] Pears, A., Seidman, S., Malmi, L., Mannila, L., Adams, E., Bennedsen, J., Devlin, M. and Pa- terson, J.: A survey of literature on the teaching of introductory programming, Working group reports on ITiCSE on Innovation and technology in computer science education, ACM ITiCSE-WGR '07, pp. 204–223 (2007).

[49] Kelleher, C. and Pausch, R.: Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers, ACM Comput. Surv., Vol. 37, No. 2, pp.83–137 (2005).

[50] 森秀樹: Scratch を用いた文系大学生向けプログラミング教育, 日本教育工学会論文誌, Vol.34, pp.141–144 (2010).

[51] Code.org: Hour of Code チュートリアル，<https://code.org/learn> （参照 2016-08-03）.

[52] 岡本雅子, 村上正行, 吉川直人, 喜多一: 「視覚的顕在化」に着目したプログラミング学習教材の開発と評価, 日本教育工学会論文誌, Vol.37, pp.35–45 (2013).

[53] 中西 渉, 辰己 丈夫, 西田 知博: PenFlowchart を用いたフローチャートによるプログラミング学習の効果に対する評価, 情報処理学会論文誌教育とコンピュータ (TCE) 第 1 巻 4 号, pp. 75–82 (2015).

[54] Tomohiro Nishida, Ryota Nakamura, Yuki Shuhara, Akira Harada, Michio Nakanishi, Toshio Matsuura: A PROGRAMMING ENVIRONMENT FOR NOVICES WITH VISUAL BLOCK AND TEXTUAL INTERFACES, Educational Alternatives, Vol. 14, pp.470–478, <http://www.scientific-publications.net/en/article/1001315/> (2016, 参照 2016-11-08).

[55] MIT STEP: Open Blocks, <http://web.mit.edu/mitstep/openblocks.html> (2009, 参照 2016-11-08）.

[56] 中村亮太，吉田智子，松浦敏雄：LilyPad Arduino シミュレータ機能付 PEN を利用した教育実践の報告, 情報処理学会研究報告コンピュータと教育 (CE), Vol.2016-CE-134, No.14, pp. 1–6,(2016).

[57] Google: Blockly, <https://developers.google.com/blockly/> （参照 2016-11-08）.