情報処理学会論文誌 Vol.62 No.7 1393–1414 (July 2021)

PBLによるプログラミング入門科目の提案：一般情報教育における入門カリキュラムの構築

内田 奈津子[1,2,a)] 久野 靖[2,b)] 中山 泰一[2,c)]

受付日 2020年10月4日, 採録日 2021年4月6日

概要：現代社会において，プログラミングは義務教育の 1 つとなり，誰もが学ぶべきものとなったが，高等教育においては，プログラミングの内容を含む必須カリキュラムの実施には至っていない．新しいカリキュラムでは，言語を学びコードが書けるようになることを目的とするのではなく，プログラミングの概念を学び，その原理を理解することに加え，プログラムを活用するために，ソフトウェア開発プロジェクトの知識を含める必要があると考えた．著者らは，1 科目のみで構成する，誰もが学ぶべき入門レベルのカリキュラムと位置づけ，プログラミング入門に PBL（Project-based Learning）を組み合わせた方法を提案し，この提案に基づき 2 単位 90 分 15 回のカリキュラムを構築した．本論文では，プログラミング入門に PBL を組み合わせたカリキュラムを設計・構築し，実践授業から得られたデータ（コンピテンシー評価，履修者の課題や最終レポートなどの提出物）をもとにカリキュラムの妥当性を検証した．

キーワード：プログラミング入門教育，PBL（Project-based Learning），一般情報教育，コースデザイン

A Proposal for PBL-based Introductory Programming Course: Designing a Curriculum for General Informatics Education

Natsuko Uchida[1,2,a)] Yasushi Kuno[2,b)] Yasuichi Nakayama[2,c)]

Received: October 4, 2020, Accepted: April 6, 2021

Abstract: In modern society, programming has become a part of basic education and should be learned by everyone. However, there is still no mandatory, universal, higher-education curriculum that includes programming content. The authors consider that a practical, new curriculum requires not so much having students being able to code, as having them learn concepts, understand principles, and gain knowledge of software development projects in order to build a foundation in programming. We propose a curriculum that includes content required of all people in contemporary society. It is a single-subject, introductory-level curriculum implemented with PBL for a two-credit, ninety-minute, fifteen-class, university-level course. In this paper, we introduce the curriculum that we designed and analyze its validity based on data (competency evaluations, student assignments, final reports, etc.) obtained from the recent practical implementation of the course at a women's university in Japan.

Keywords: introductory programming education, PBL (Project-based Learning), general informatics education, course design

1. はじめに

情報技術（IT）は，今ではあらゆる領域に活用され，人々の暮らしの中にまで入り込み，なくてはならない存在となった．IT の利活用により豊かな暮らしを実現するために，プログラミングの知識は，専門に限らず誰もが社会で

[1] フェリス女学院大学
Ferris University, Yokohama, Kanagawa 245–8650, Japan
[2] 電気通信大学
The University of Electro-Communications, Chofu, Tokyo 182–8585, Japan
a) uchida@ferris.ac.jp
b) y-kuno@uec.ac.jp
c) nakayama@uec.ac.jp

© 2021 Information Processing Society of Japan

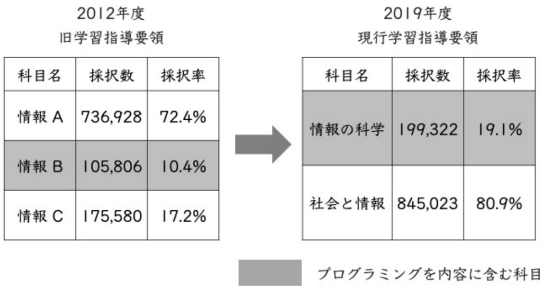


図 1 高等学校における情報科目と教科書の採択率
Fig. 1 Information subjects and textbook adoption rates in high schools.

求められるようになってきている．

2013 年，世界最先端 IT 国家創造宣言 [1] が閣議決定され，プログラミング教育の必要性が示され，小学校へのプログラミング教育の導入が決まった．また，日本学術会議において，2016 年に「情報学の参照基準」[2] が，2020 年に「情報教育課程の設計指針—初等教育から高等教育まで」[3] がとりまとめられ，公表された [4]．

2022 年から実施される高等学校情報科の新教育課程では，情報の科学的な理解に重点を置き「情報 I」（2 単位）を必履修科目としたうえで，発展的内容として情報システムなどを扱う「情報 II」（2 単位）を選択科目とすることになっている [5]．さらに，大学入学時点での情報の素養を問うために，大学入学共通テストで「情報」が出題されようとしている [6], [7]．

このように，情報の能力は，文系の専攻，理系の専攻にかかわらず求められるようになり [8]，大学において学士力 [9] を身につけるための土台となるものとして必須のものとなってきた．

しかし，大学入学までにプログラミングを学ぶ機会がある学生は，教科書の採択率の数字から推測すると，2012 年度には約 10% [10]，2019 年度には約 20% [11] であり（**図 1**），少しは増えつつあるが，8 割以上の生徒は高等学校でプログラミングを学ばずに大学に進学していることになる．

高等教育に対しては，2016 年 11 月から 12 月にかけて，文部科学省が，「超スマート社会における情報教育の在り方に関する調査研究」[12] を実施した．この一部として，国内の全大学における情報学分野の教育に関する調査が行われ，「アルゴリズムとプログラミング」は「コンピュータリテラシー」や「情報倫理とセキュリティ」など他の項目と比較して “教えていない” 学部学科の率が高いという結果が得られている [13]．

このように，高等教育においては，プログラミング入門の内容を含む必須カリキュラムの実施には至っていないことを受け，著者らは，高等教育における学習の機会を補うだけでなく，情報化社会における教養としてすべての人に求められる内容を包括したプログラミング入門のカリキュラムが必要であると考える．

プログラミングの学習を通じて，最低限学習するべきであると期待される内容は，言語を学びコードが書けるようになることではなく，プログラミングの概念を学び，プログラムがどのように作られ，どのように動いているかの原理を理解することである．また，プログラムをどのように活用するかといった点において，ソフトウェア開発プロジェクトに関する知識も含めて学ぶ必要がある．

なぜなら，コンピュータに依存する現代社会における活動を考慮すると，専門家だけでなく誰もがソフトウェア開発の基礎的知識を持ち，専門家と一緒にチームを組み，どのようなソフトウェアを作るかを適切に判断したり，スケジュールに合わせて開発を進めソフトウェアを完成させたりする能力が求められているからである．

そのため，高度な IT 人材の育成だけでなく，誰もが基礎知識を持ち活動できるよう，人材育成の底上げを図る必要があると考える．

英国内閣府の Potter は，次のように述べている [14]．

> 技術そのものの導入が重要なのではなく，公共サービスにおける職員の業務がどのようなものなのか，ユーザーがそれをどう使っているのか，そこに発生している課題は何なのかを知ることが重要です．そこにどうやって技術を使い改善していくかを見極める力が必要なスキルだと考えています．

Potter は，技術者の面から求められるスキルについて述べているが，著者らは，発注者や利用者の立場にあっても，技術者と一緒に仕事をしていくうえで，情報技術の基礎的知識は必要であると考える．情報化社会においては，多くの課題解決には技術だけで解決できる問題は少ない．そのため，誰もが情報技術に関する基礎的な知識を持つことにより，様々な立場の人が関わり技術者と交わり円滑なコミュニケーションを図ることができるようになることが期待でき，より良い社会を築くことができると考える．

社会に出る以前に，ソフトウェア開発プロジェクトに関する知識を含めたカリキュラムで学ぶことにより，チームで協力して何かを作る経験ができ，プログラミングの基礎知識に加え，「問題認識から課題解決までを主体的に学ぶ」ことの理解が進むと考える．

ワインバーグも，『プログラミングの心理学』[15] で，“社会活動としてのプログラミング” について，プログラマは孤立して働いたりしないことやチームでの効率について述べている．

しかし，既存の多くのプログラミング入門の授業は，プログラミングの技能を学ぶことを目的としていて，それを使ってチームで何かをするという内容にはなっていない．情報系の専攻であれば，入門科目の後，情報系の実践科目等様々な科目を通じて，ソフトウェア開発的な考え方を学び実践する機会が提供されるであろう．しかし，それ以外

© 2021 Information Processing Society of Japan

の専攻，特に文系の専攻では，プログラミングの内容を含む科目はあっても，ソフトウェア開発の体験を含みチームでプロジェクトを行うような学習の機会はないままに終わってしまう．

そのため，専攻によらず誰もがプログラミングの概念を学び，その原理を理解することに加え，“プログラムを活用するためのソフトウェア開発プロジェクトを理解することを含む” という要求を満たすことはできていない．

著者らは，この課題に対する 1 つの解として，PBL（Project-based Learning）を組み合わせた 1 科目のみで構成する，誰もが学ぶべき入門レベルのカリキュラムを提案する．また，提案に基づき 2 単位 90 分 15 回のカリキュラムを構築し，実践授業を行い，その妥当性を検証する．

本論文では，プログラミング入門に PBL を組み合わせたカリキュラムを設計・構築し，実践授業から得られたデータ（コンピテンシー評価，履修者の課題や最終レポートなどの提出物）をもとにカリキュラムの妥当性を検証した．

2. 先行研究

2.1 高等教育におけるプログラミング入門教育

高等教育においては，一般教養科目の一般情報教育において，様々なプログラミング入門教育が行われてきている [16], [17], [18], [19]．

吉田は，PEN を用いることで 4 コマの授業でプログラミングの導入を行うことを述べている [16]．河村は，文系大学の一般情報教育におけるプログラミング教育について，JavaScript を用いた教育の留意点や事例を提示している [20]．西田らは，コースウェアが学習に与える影響について述べ，その中で，図形描画型の教材を用いることの効果を取り上げている [21]．

これらはいずれも，学習方法・取り上げる題材の工夫に焦点をあてているが，PBL を活用したり，ソフトウェア開発プロジェクトに必要な事柄を学んだりという視点はない．しかし，図形描画型の教材の活用や導入教育を行ううえでの時間数の設定などに関しては，著者らの授業設計にも参考となる．

2.2 高等教育における PBL

学習とは，受動的に知識を取り入れることではなく，題材に能動的に取り込み，それによって学習者の内部にある既存知識と新たに学ぶことがらの関連が多く作られることを通じて行われる [22], [23]．

文部科学省の高大接続システム改革会議では，従来型の（知識伝達型の）学習方法の限界について言及し，これからの学校教育では，アクティブラーニングを中心に据えるべきであると提言している [24]．

アクティブラーニングの 1 つの方法として PBL がある．近年では，医学や工学に限らず多くの分野で PBL による教育が行われ，目的もコンピテンシー育成，問題解決力の育成，地域振興など様々である [25], [26], [27]．

PBL では，様々なことを主体的に学ぶことにより，コンピテンシーの育成や問題解決力の育成に効果的であり，本提案でも，同様に期待する項目である．

しかし，総合的なアウトプットを求める性質の PBL は，基礎知識を身につけた後の教育課程終盤での実施が学生にとって受講しやすいと述べられており [28], [29]，はこだて未来大学の事例 [30] では，開学当初から 3 年生の必修科目として位置付けられている．

一方で，内田は，PBL 実施後の学習に対するモチベーションの向上や就職活動に必要なグループワーク等のスキルの獲得を狙い，教育課程中盤となる 2 年生後期のカリキュラムに取り入れた [27]．

内田の知見からは，教育課程終盤によらず，教育課程の前半でも PBL での授業実践において効果が得られている点は，著者らが考える入門レベルの授業においても，PBL の導入に期待が持てる．

2.3 PBL とプログラミング入門

プログラミング入門における PBL の事例として，Nuutila らが行った，医学部で広く使用されている 7 ステップメソッド（**表 1**）に着目したものがある [31]．様々な種類の PBL の事例と他の学習の組合せをテストし，プログラミング入門教育における PBL のカリキュラムを実装する最良の組合せを検討した．そして，情報系コースの初年時の学生を対象とした Java によるオブジェクト指向プログラミングを学ぶクラスでアプリケーション開発を課題とした実

表 1 PBL の 7 ステップメソッドの内容
Table 1 PBL seven step method.

Opening session – half an hour, in the group
Step 1: Examination of the case. The group gets familiar with the case material. Step 2: Identification of the problem. An initial title for the case is specified. Step 3: Brainstorming. The students present their associations and ideas about the problem to find out what is already known and how does the case relate to the previous knowledge. The ideas are said aloud and written on self-stick notes, which are organized on a white board. Step 4: Sketching of an explanatory model. An initial version of the explanation for the problem is constructed and most important concepts and their relations are identified. Step 5: Establishing the learning goals. Those parts of the explanatory model that are mysterious, fuzzy, or simply unknown are identified and the central ones are chosen as learning goals for the group.
Study period -- one week, each student working independently
Step 6: Independent studying. Each student independently studies to accomplish *all* learning goals. This phase includes information gathering and usually a substantial amount of reading (e.g., 50--150 pages).
Closing session -- one to two hours, in the group
Step 7: Discussion about learned material. Equipped with the newly acquired knowledge, the group reconvenes to discuss the case. The discussion includes *explanation* of central concepts and mechanisms, *analysis* of the material, and *evaluation* of its validity and importance.

文献 [31] p.127 より引用

© 2021 Information Processing Society of Japan

践を行った．

Nuutila らは，PBL セッションは，抽象概念を学ぶモチベーションを作り出す効果的な方法であり，問題解決とプログラミングの設計を学習するうえで有益であると述べている．一方で，プログラミングスキルを習得するためには演習が必要であり，反復練習によりスキルを身につけるような分野では，PBL は必ずしも有効でないとしている．そのため，反復練習によりコーディングスキルを身につけさせるには，プログラミング演習を追加する必要があると述べている．

研究の成果として，PBL セッションは，従来型授業と比較してドロップアウト率の大幅な低下（従来のプログラミングコースの 45%に対し 17%）と，プログラミングスキル獲得に加えて，グループワーク・共同作業・独立した学習・知識の外部化に関するジェネリックスキル獲得に有益であったと述べている．

著者らは，PBL とプログラミング入門について，非情報系の学生を対象にした実践事例は探すことができなかった．しかし，Nuutila らの先行研究は，抽象概念を学ぶモチベーションを作り出す効果的な方法であること，問題解決とプログラミングの設計を学習するうえで有益であるという結果から，うまく設計すれば，著者らの提案する入門教育におけるカリキュラムへの反映も期待できる．

Nuutila らが行った 7 ステップメソッドは，我が国における標準の 90 分 15 回で構成される授業の枠にあてはめることは困難であるが，この考え方を応用し，Step 1〜6 を各グループで取り組み，Setp 7 を教室全体でのディスカッションととらえることにより，本提案の PBL のカリキュラム構成の参考となる．

3. カリキュラムの設計

3.1 設計における基本方針

本提案のカリキュラム設計の方針は，情報化社会における教養として，すべての人に求められる内容を包括した高等教育におけるプログラミング入門のカリキュラムを構築することである．

カリキュラムを設計するにあたり，以下のことに留意した．

- プログラミング言語を学ぶのではなく，プログラミングの概念を学ぶこと
- ソフトウェア開発の知識をともなって学ぶこと
- 多様な学生集団が対象であり，誰もが容易に学べること
- 限られた時間の中で実施できること
- 学士力 [9] の向上に配慮すること

既存のプログラミング入門のカリキュラムでは，プログラミングの技能を学ぶことを目的としており，様々な立場の人が関わり，協力・分担してより良いものを築きあげていくという経験をすることはできない．基礎的な項目をおさえながら，入門レベルであっても，ソフトウェア開発プロジェクトの知識を含めた設計をすることにより，これからの社会で役に立つような知識・技能の習得の入り口となることを期待した．

そのため，知識を学ぶだけでなく，それを活用する視点を持ち，技術者の立場や，利用者の立場なども考慮して学ぶ環境をつくることを考え，プロジェクトを取り入れることを考えた．

対象は，大学における一般教養に相当するものであり，多様な学生を対象とすることになる．向き不向き，できるできないにかかわらず，多様な学生がそれぞれに満足できるような配慮が必要である．プログラミングからチーム活動，成果物のとりまとめまで多様な活動を含むことにより，それぞれの学生が自分に合った場所で貢献し，自分に合った学習成果を持ち帰れるのではないかと考えた．

さらに，誰もが学ぶためには，多くの時間を割くようなカリキュラムでは浸透しない．そのため，複数科目で構成されるようなカリキュラムも実現性が薄いと考え，90 分 15 回の標準的な 2 単位の時間で実現できるよう，設計することが望ましいと考えた．

そして，高等教育のカリキュラムとして，学習成果に関する指針として 4 分類からなる「学士力」[9] の向上が図れるよう配慮することも忘れてはならない項目であると考えた．

3.2 カリキュラムの構成

初学者の学生たちに，何も知らない状態からいきなりチームでプログラムを作らせることは難しい[*1]．

先行研究 [17] では，プログラミング入門は 4 コマあれば実施できていたこと，また，文献 [31] で示されている反復練習の必要性から，本カリキュラムでは，授業構成の前半（以下，プログラミングパートと記す）を，個々にプログラミングの知識を学ぶ時間とし，6 回を設定した．

後半（以下，プロジェクトパートと記す）は，ソフトウェア開発の工程に沿ったプロジェクトをグループワークで実施する構成を考え [32], [33]，ソフトウェア開発の工程を基準とした 9 回を設定した．

授業の構成とその流れを，**図 2** に示す．

プログラミングパートでは，後半のプロジェクトにつながる知識を精選したうえで，実習中心で個々に取り組ませる．プロジェクトパートでは，3〜4 名を 1 チームとして，チームで取り組ませる．

この構成をすることにより，プログラミングパートで十

[*1] プログラミングの素養を持つ学生が一定の割合でいたり，多数の TA がついたりなど，ファシリテータ役を勤められるような人材がいる環境があればこの限りではないが，そのような恵まれた状況はあまりない．

© 2021 Information Processing Society of Japan

表 2 15 回のカリキュラム内容
Table 2 15-Lesson class schedule.

回数	項目	内容	課題
1 回目	プログラミングの導入	プログラムとは何であるか理解し，簡単なプログラムを動かせるようになる．例題：三角形の面積を求める．	課題 A/課題 B
2 回目	分岐と繰り返し	基本的な制御構造を理解し，これらを使ったプログラムが書けるようになる．例題：入力 x の絶対値を計算する．平方根の計算．	課題 A/課題 B，コンピテンシーチェック
3 回目	制御構造と配列	制御構造の組み合わせについてとデータ構造と配列を理解する．例題：fizzbuzz 問題（0〜99 の数を順に打ち出す．ただし 3 の倍数の時だけは fizz と打ち出す）	課題 A/課題 B
4 回目	手続き・関数と抽象化	手続き（メソッド）による抽象化を理解する．例題：RPN（逆ポーランド記法）電卓	課題 A/課題 B
5 回目	2 次元配列と画像	2 次元配列・レコード型と画像の表現を理解し，様々な形を描画する．	課題 A/課題 B
6 回目	プログラミングまとめ	自分が生成したいと思う画像のために何が必要かを考え，画像の内容に合わせてプログラム構造を実現する．	中間課題
7 回目	グループ作成・課題とゴール設定	プロジェクトの課題説明とグループ作りおよび役割分担を決定する．	ワークシート
8 回目	アイデア出し・ブレスト・仕様決め	ソフトウェア開発について説明し，デザイン決定に向けてのアイデアを話し合う．	ワークシート
9 回目	デザイン・設計・分担・制作	作品のデザイン決定と各パーツの分担を決める．	ワークシート
10 回目	試作・中間ドキュメント作成	パーツごとの試作および中間発表に向けてのドキュメントを作成する．	ワークシート
11 回目	中間発表・仕様見直し	中間発表のコメントを受け，仕様の見直しを行う．	ワークシート
12 回目	制作・単体テスト	単体テスト，統合テスト，コードレビューなどの概念について理解する．仕様の見直しを受けての修正と単体テストを行う．	ワークシート
13 回目	コードレビュー	コードレビューについて理解する．試作したパーツについてのレビューを行う．	ワークシート
14 回目	制作・統合テスト	各自で作成したパーツを統合し，作品を完成させる．	ワークシート
15 回目	最終成果物（まとめ）	グループの成果について発表し，完成した作品の評価を行う．	最終レポート，コンピテンシーチェック

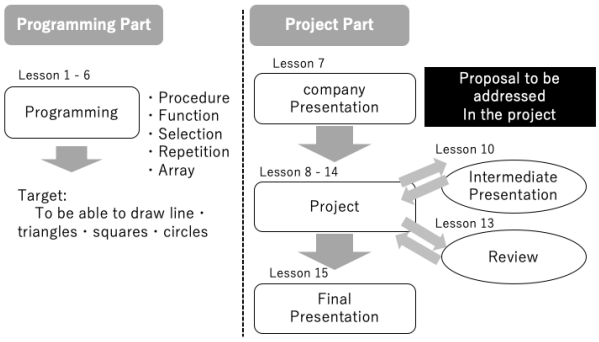


図 2 授業の構成
Fig. 2 Course structure.

分に身につけられなかったプログラミングの基礎知識は，後半のプロジェクトパートで，繰り返し取り組み補うことが可能であり，プロジェクトを通じて体験的に身につけることが可能となると考えた．

各授業の終わりにアンケートや感想を課し，授業の進捗をチェックし，15 回目の授業後には，最終レポートを課す．これに加えて，2 回目の授業後と 15 回目の授業後に，教育の成果や影響を分析するために，コンピテンシー評価のチェック（3.6 節で，詳しく述べる）を行う．

表 2 に，15 回の構成を示す．授業の具体的な内容については，内容欄に記述した．各パートの詳細は，以下の節で説明する．

3.3 プログラミングパート

本授業は，初めてプログラミングを学ぶ学生が対象であり，後半のプロジェクトパートまでに「自分で思ったようにコードを書く」力を養う必要がある．プログラミングを学ぶ際の指針として，第 2 著者の久野のまとめた指針 [34] を基本とした．

以下に，本カリキュラムに用いた指針をあげる．

- プログラミング言語の文法や機能を逐一，順番に説明していくのではなく，初回から数行のプログラムを

© 2021 Information Processing Society of Japan

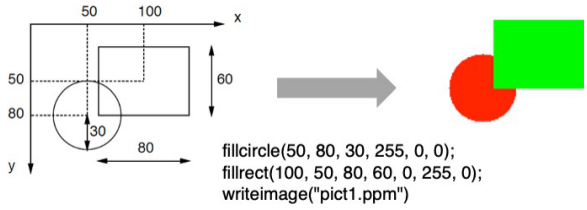


図 3 図形描画の例
Fig. 3 Sample: Drawing shapes.

書く．

- 例題を選んで丁寧に説明し，試験をするような暗記中心の授業構成にはしない．
- 例題をそのとおりに打ち込んで実行させることに注力するのではなく，自分のレベルにあった問題で取り組ませる．

使用するプログラミング言語は，以下の理由から，構文がシンプルで初学者にも比較的学習が容易なプログラミング言語 Ruby を採用した．

- 社会で一般的に使われている環境でコードを書く体験をさせたいこと
- 記述量が少なく構文がシンプルであること
- 他のプログラミング言語に応用しやすいこと

コードを書く際には，実行時のエラー処理をはじめ，初学者が 1 人で学習する際には困難がともなうが，それを軽減させるために，ペアプログラミング [35], [36] が有効であると考えた．特に，TA 不在の授業運営では，演習時の対応を教員 1 人で賄わなければならず，その対応で授業時間の浪費が発生しかねない．ペアで取り組むことにより，学生同士での解決が期待できるだけでなく，1 人→ 2 人→ 4 人へとプロジェクトパートを意識し，緩やかに牽引できると考えた．

先行研究 2.1 節で示した文献 [16], [17], [21] の図形描画型の教材の利用を参考に，本カリキュラムでも図形生成の教材を用いることを考えた．図形生成の教材については，事前に準備した (1) 各種図形（丸，三角，四角，楕円）を描くメソッドと (2) 生成した絵をファイルに出力するメソッドを活用させた（**図 3**）．

たとえば，円を描くメソッドでは，中心座標，半径，色の RGB 値（0〜255）を引数として指定することで円を描く．入門レベルのため，複雑な構造は作れないことから，必要な引数の部分を考えさせることにより，データ構造と抽象化を理解させることができると考えた．

各回は，「説明→演習→時間内の課題（課題 A）→次回までの課題（課題 B）」で構成した．課題は，非常に平易なもの（例題を少し手直しすれば提出できる）から高度なものまで用意し，学生は自分のレベルに応じて選んで解答する．これにより，レベルに応じた学びを得ることができ，「例題をコピーして動かす」のではない，自分なりのコードを書くことが可能となることを期待した．最低限，自分で考えコードを書き動かすことを念頭に置き，プログラムを記述・実行するための手段を理解し，プログラムを実行することができることを目標とし，プロジェクトパートにつなげることを考えた．

各課題には，3〜4 問で構成されたアンケート（授業や課題の感想，今後の展望など）をつけ，理解の進捗や授業の感想などを求めた．これにより，学生の進捗を把握し，理解が芳しくないところについては，翌週の授業でフォローすることができると考えた．

また，プログラミングパートで十分に理解できなかった内容は，プロジェクトパートで繰り返し学習することで，学び直しができると考えた．

上述した考えに基づき，4 回でプログラミングの基礎知識となる，制御構造と抽象化を学び，残りの 2 回で図形生成のプログラムに取り組む構成とした．

テキストは，上述の課題やアンケートを盛り込み，各回分冊としてオリジナルに作成した．

テキストと演習問題は，電気通信大学共通教育部情報部会「基礎プログラミングおよび演習 2017」[37] の#1〜#5 を使用し，多少ボリュームを落とし，微分積分などの範囲は削除するなどして，修正を加えた．付録 A.1 に，使用した演習問題のリストを示す．

3.4 プロジェクトパート

後半では，ソフトウェア開発プロジェクトを基とした PBL を取り入れ，3〜4 名で 1 チームを構成し，プロジェクトに取り組む．

本格的なソフトウェアの開発は難しいことから，1 つの絵をソフトウェアと模し，チームで 1 つの絵を描くプロジェクトに取り組ませることを課題に考えた．ここでの目的は，グループワークにより，実践的にプログラミングの知識を定着させるとともに，ソフトウェア開発プロジェクトを理解し，プロジェクトの運用能力を身につけることである．

「ソフトウェア開発プロジェクト」にかかわる能力についての検討は，誰もが学ぶ入門レベルに対応する資料がないことから，IT 人材白書 2013 年版 [38] の「産業界（IT 企業）が教育機関に重視して欲しい教育と教育機関が近年重視している教育の比較」を参考にした．

1〜4 位は，ジェネリックスキル（上位から，コミュニケーション，問題解決，文章・文書作成，チームワーク・協調性）であり，産業界も教育機関も同様に上位にあげている．要求分析，プロジェクトマネジメントは，教育機関では重視する割合が低いが，産業界では 6，7 位にある．

卒業後の社会への接続を意識し，プログラミングの活用の経験を持たせたいことから，この要求分析，プロジェクトマネジメントに注目した．

© 2021 Information Processing Society of Japan

表 3 PBL の欠点
Table 3 Disadvantages of PBL.

No.	内容
1	学生も教員も多くの時間を要する
2	ファシリテータのレベルに応じてアウトプットの質に差が生じる
3	分担による学生の活動量に差が生じる
4	メンバーの能力やスキル差によりアウトプットの質に差が生じる

要求分析については，第 1 著者が企業とのプロジェクトを行う中で，専門外の学生にも社会に出る前に学んで欲しい項目であると要望されたことがあり，データによらず必要性を感じる項目であった．

前半で獲得したプログラミングのスキルを活用し，よりリアルな課題を設定し「どのような作品を作るか」を決めるプロセスで，要求分析を体験することができると考えた．

1 回のプロジェクトですべてを学ぶことは難しいが，プロジェクトマネジメントを意識し，期限内にプロジェクトを完了できるよう経験的に学ぶ環境を考え，表 2 の 7 回目以降の構成を設定した．

PBL の導入に際しては，**表 3** にあげる PBL の欠点 [39], [40], [41] が問題とならないように，カリキュラムを設計する必要がある．

欠点 1 については，図形を生成する教材を用いた課題のみで完結するようにカリキュラムを設計するという本論文の提案全体が，学生・教員に要求する時間を自ずと制限することとなり，解決策となると考えた．

欠点 2 については，多様なプロジェクトを扱うのでなく，画像を生成するという枠組みに限定することで，教材や学習内容をあらかじめ準備し，教員がファシリテータの役割をすることにより，ファシリテータのスキルへの依存を相対的に軽減することができると考えた．

欠点 3 および 4 については，チーム内での役割の明確化とチーム内・チーム間での情報共有が有効であると考えた．

著者らは，これまでに，単にグループで取り組むだけで，役割が明確化されていないグループワークを見聞きしてきた．このような状況を回避するために，本提案では，最初に話合いで担当を決めた後，どの部分が誰の担当か，それがどれだけ進捗しているのかを誰の目にも明らかなようにつねに維持するため，ワークシートの活用（役割分担の明記）と発表の場の活用（11・13 回目）を考えた．これにより，自分の分担についていい逃れできなくするだけでなく，遅れていればチーム全体の問題としてメンバが手助けすることで，チーム内での学びが起こり，スキルの劣ったメンバが学んで追い付くことを促すことを期待した．

以上，述べたことから，到達目標の項目には，プロジェクトマネジメント，要求分析，チームでの活動を考えてメンバの多様性，活動を通じての情報共有や資料の作成などから IT スキルの 4 項目を考えた．3.5 節で，詳しく述べる．

表 4 各回の確認項目
Table 4 Confirmed items for project sections.

回数	項目	確認内容
7	(2)	プロジェクトの運営をどのようにしていくかという予定
	(4)	チームでそれぞれの個性を活かし役割分担するのにどういう風にすればいいと思うかの目論見
	(5)	運営においてコンピュータはどのように活用していきたいか
8	(3)	仕様策定に当たって「発注者の要求」はどのように活かされましたか
9	(2)	やることがたくさんあるうちで重要度をどのように考え作業を進めましたか
	(4)	実際に役割を割り振るところで予想外のことはありましたか
	(5)	設計の内容をどのように記録し管理することにしましたか
10	(2)	ここまでのプロジェクトの運営の流れはどうだったか
	(3)	開発予定のものが「発注者の要求」と合致することをどのように確認していますか
11	(4)	メンバーのこれまでの分担状況についてよかったこと，悪かったことは何ですか
	(5)	管理してきた情報をどのようにうまく修正しましたか
12	(2)	プロジェクトはどのようにうまく進められましたか
13	(4)	コードに不具合が見つかるとして，それは個人が「悪い」のでなく誰でも間違いはおかすという姿勢を保てましたか．やったことが「平等でない」としてもそれは織り込みずみであるという姿勢を保てましたか
14	(2)	プロジェクトをどのように進められたと認識していますか
	(3)	要求にかなった成果物になったと考えますか
	(4)	役割分担はうまく行きましたか
	(5)	情報活用はうまく行きましたか

項目：(2) PM (3) 要求分析 (4) メンバの多様性 (5) IT スキル

個人用のワークシートには，授業のまとめと翌週の課題，授業の感想を毎回共通で書く．また，学生の活動状況を確認するため，到達目標に基づき該当する項目についての質問も加えた．各回異なる内容については，**表 4** に示す．各項目の () 内の数字は，到達目標（後述，**表 5**）の番号と同様である．(5) では，特に情報共有についての内容を取り上げた．

本授業設計では，全員が同じゴールに向かってグループワークを進めているため，チーム内だけでなく，チーム間の教室全体で，同じ視点でディスカッションできるメリットがある．

“グループワーク→発表” のサイクルを 3 回（7 → 11，12 → 13，14 → 15 回）繰り返すことで，各チームでは，“制作（創造）→共有→振り返り” のスパイラルが生まれ，教室内では，“共有→共感→競争” という流れが生まれることを期待した．

© 2021 Information Processing Society of Japan

表 5 授業の目的と到達目標
Table 5 Course aims & course goals.

目的
現代は，情報社会である．身の回りには，情報技術によって実現されたサービスが氾濫し，個人的利用からビジネス等での活用まで不可欠であり，情報技術を用いないサービスを見つけることは困難である．特に，これらを構成するプログラミングの基礎知識は，誰もが必要とされる知識の 1 つである． 「プログラミング」の知識は，「単に言語の文法を学ぶ」「例題をそのまま打ち込んで動かす」だけでは明らかに，社会に出て役立つという要件は満たせない．プログラミングをともなう活動には，「ソフトウェア開発プロジェクト」がある．これからの社会においては，ソフトウェア開発に直接かかわるかかわらないに関係なく，誰もがある程度の基礎的な知識を持っていることが期待される．そのため，単にプログラミングを学ぶのではなく，プロジェクトもあわせて学ぶことが重要である． 本授業では，初めてプログラミングを学ぶ人を対象に，「社会に出てから役に立つような形でプログラミングを学ぶ」ことを目的とする．
到達目標
(1) プログラミング
プログラムを記述・実行するための手順を理解し，プログラムを実行することができる． プログラム中の誤りを発見し，修正することができる． 制御構造・抽象化について，説明することができる． 自ら考えたことをプログラムとして記述し，表現することができる．
(2) プロジェクトマネジメント
チームで決めた計画（スケジュール）に沿って自分の作業を行うことができる． 自分の役割を理解し，チームの仲間と協力して取り組むことができる． 問題解決とそのための思考過程を体験し，問題認識から課題解決までを主体的に取り組むことができる．
(3) 要求分析
目的にかなった開発のために「何が重要か」「どのような機能を実装するか」を理解し，創意・工夫することができる． 相手の要求を隅々まで聞き出すこと，顧客にもチーム内にも正確に伝えること（コミュニケーション力，記述力）について理解し，適用することができる．
(4) チームメンバの多様性
チームメンバの多様性を理解し，個々の特性に合わせた役割分担を行うことができる． 個性やそれぞれが持つ知識が異なる集団でプロジェクトを進める際に，個人を理解し認め，尊重し，各自の得意不得意を理解してプロジェクトを行うことができる．
(5) IT スキル
プログラムの記述・実行に必要なツールを操作することができる． 課題やレポート作成，発表に必要なツールを適切に活用することができる． メールや LMS などを用い，チームのメンバとコミュニケーションすることができる．

このスパイラルにより，“考えをまとめる→試作する→再考する→作品を完成させる”というプロセスの中で，ソフトウェア開発の工程について理解を深め，前半で不十分であったプログラミングのスキルを補うことができると考えた．

3.5 授業の目的と到達目標

授業設計の基本方針に基づき，カリキュラムの構成をした．これに合わせて，シラバスに掲げる目的と到達目標（表 5）を設定した．到達目標には，3.4 節で述べたとおり 5 項目を設定した．

到達目標 (1) は，プログラミングの入門レベルであること，向き不向きに寄らず，誰もが学ぶ点を配慮し，「自分で思ったようにコードを書く」力を養うことを基準とした項目を設定した．

到達目標 (2)〜(4) の 3 項目は，本提案の特徴的な部分である．入門のカリキュラムでは取り扱うことが難しいが，ソフトウェア開発プロジェクトの要素を入れることにより，社会におけるプログラミングの活用を意識させた体験から，コミュニケーション，問題解決，文章・文書作成，チームワーク・協調性などのジェネリックスキルの向上も含めて，学習効果を期待するところである．

到達目標 (5) は，15 回の授業を通じてコードの編集やデバックだけでなく，課題やレポートの作成，グループワークのための情報共有で，メールやその他のツールを使う場面がたくさんあることから，操作に陥りがちな情報教育を脱し，実践の中で活用することによりスキルアップを期待して設定した．

3.6 コンピテンシー評価の設計

教育の成果や影響を分析するうえで広く用いられている評価指標として，コンピテンシー評価がある．

独立行政法人情報処理推進機構は，大学の出口と企業の入口における人材のマッチングを円滑するため大学と企業のシームレスな評価モデルとして活用することを目的とし，『コンピテンシー評価項目表（参照モデル）[42]』を提

© 2021 Information Processing Society of Japan

表 6 コンピテンシー評価の基本項目
Table 6 Basic competency check.

・コミュニケーション力		
1.	傾聴力	相手の意見を丁寧に聴き，正しく理解し，受け止める
	読解力	記述された内容を正しく理解する
2.	記述力	正しい文章で他人が理解できるように記述する
3.	議論力	議論の目標を設定し，それに合わせて効果的な議論を進める
・問題発見・解決力		
4.	課題発見力	現状と目標（あるべき姿）を把握し，その間にあるギャップの中から，解決すべき課題を見つけ出す．現状の問題点自体が明確でない場合も，顕在化している問題や現場観察から，真の課題（テーマ）を見出す．
5.	課題分析力	課題の因果関係を理解し，真の原因を見出し，その本質を整理する
6.	解決策立案力	問題点の原因に対し，それを解決する方策を複数立案し，その中で解決につながるものを選択し，アクションプランを導く（解決のプロセス全体を論理的に思考し実践する）
・知識獲得力		
7.	学習力	専門知識のみならず人文社会に関するものも含めて，幅広い分野で知識やノウハウを習得する
	応用力	入手した知識やノウハウを関連付けて活用する
	知識創造力	学んだ知識や自身で経験したこと・気づきなど全体の関係性を整理し，単なる知識の断片にとどめないで，新たな知恵を生み出すための知識化の作法を習得する
・組織的行動能力		
8.	役割認識（チームワーク）	チーム，組織の目標を達成するために個人の役割を理解し，当事者意識を持ってチームとして行動する
	主体性	物事に対して自分の意志・判断で考え，自分から進んで行動する
	協働	共通の目標を達成するためにお互いの考えを尊重し，信頼関係を築くような行動をとる

供している．

大学全体，学部全体などで実際に運用されている先行的な実施例や，文献 [42] の WG メンバの企業各社における実例・意見をもとに，6 分類 17 項目が設定された．また，これらの項目は，一般的なコンピテンシー項目としてあがる項目である，経済産業省から提唱された概念である『社会人基礎力』[43] で定義されている「3 つの能力 12 の要素」も参考にされている．

著者らは，カリキュラムの目的や到達目標に，コミュニケーション，問題解決，文章・文書作成，チームワーク・協調性などの項目を含むことから，この参照モデルの策定の経緯に賛同し，活用することとした．カリキュラムに該当しない，2 分類 4 項目（自己実現力，多様性の理解）を除外し，独自の 4 項目を追加して 12 項目を決めた．以下で，詳しく述べる．

表 7 コンピテンシー評価の追加項目
Table 7 Additional competency check.

9. **計画力**：目標達成のために，計画的に作業を進める
1. チームで決めた計画に沿って自分の作業を進められる
2. チームで決めた計画に不備を発見して皆と相談できる
3. チームで決めた計画をさらに改良する提案が出せる
10. **共有データ**：効果的にデータを作成し，共有する
1. チームで決めた共有場所のデータを参照して作業できる
2. チームで決めた箇所に共有データを追加して他人に使ってもらえる
3. チームで情報を共有する場所の新しい使い方の提案が出せる
11. **要求仕様の利用**：要求仕様に沿って作業を行う
1. チームで決めた仕様に合わせて自分の作業を行える
2. 自分の担当箇所がチームで決めた仕様と合わない時に相談できる
3. チームで決めた仕様が発注者の意図と合わないことを指摘できる
12. **メンバーの多様性**（多様性の活用）：個々の性格や特性を理解し，それを活かしたチーム作業を行う
1. チームで決めた自分担当の作業をこなすことができる
2. チームで決めた自分担当と他者担当の曖昧な部分を相談して調整できる
3. チームで担当を決める際にチームに適した分担方法を提案できる

まず，基本項目として，『コンピテンシー評価項目表（参照モデル）』[42] から，本提案で活用した 8 項目を**表 6** に示す．

参照モデルの表の「目標の設定/到達レベルの確認」列を基準として，傾聴力・読解力，学習力・応用力・知識想像力，および役割認識（チームワーク）・主体性・協働は，項目をまとめて扱った．

回答内容は，レベル 1（基本行動）からレベル 3（卓越行動）の 3 段階での自己評価とする．表 6 では，3 段階のレベルについては省略する（文献 [42] 参照）．

次に，カリキュラムの到達目標に合わせ，独自に追加する 4 項目を**表 7** に示す．基本項目と同様，回答内容は，3 段階のレベルに分けた．

プロジェクトマネジメントについては，グローバル標準化された資格があるが，ここでは計画力とメンバの多様性を項目に取り上げた．このカリキュラムは入門レベルであるため，資格取得が目的ではないことから，基本的な概念を学び，プロジェクトを実践することでプロジェクトマネジメントの概念を体験的に学び，今後の活動に活かすきっかけとなることを目的としたためである．

© 2021 Information Processing Society of Japan

4. 授業実践

4.1 実践環境

第1著者は，3章で設計したカリキュラムの妥当性を検証するために，2017年度後期（2017年9月～2018年1月）に，フェリス女学院大学国際交流学部国際交流学科専門科目「情報発信と世界」で授業実践を行った．この科目は，1年生から4年生までが履修可能な選択科目である．他学部・他学科開放科目でもあるため，他学部の学生の履修もあった．履修状況を，**表 8** に示す．

必須の情報系科目がないため，学生の知識レベルは様々であり，パソコンが苦手であることを理由に履修した学生も多数いた．

授業は，週1回90分で，15週で実施した．教員は1名で，TAなどのサポート要員はなし．履修登録は26名で，うち23名が最後まで履修した．

途中で中断した3名のうちの2名は，4年生であり，卒業論文の提出時期（12月2週目）と重なり，両立ができなかったことが理由であった．もう1名は1年生で，授業開始時から欠席が多かった．この3名は，プロジェクトパートが始まる以前に履修を中断したため，チーム分けには含めなかった．

教室の環境は，Windowsで，プログラミング言語RubyとImageMagick（画像表示ソフト）をインストールしただけの簡易的な設定で，学生の所有するパソコンでも対応できるよう，特別な設定は行わなかった．エディタには，メモ帳を使用した．宿題など教室外での学習のため，Macユーザには，エディタとしてAtomを勧めた．

授業のテキストは，学修支援システム（LMS）を通じて配布し，課題等の提出もこのシステムを利用した．

コンピテンシー評価は，履修登録の都合から，授業の2回目と15回目に行った．この結果については，6章で，詳しく述べる．

4.2 授業実践：プログラミングパート

3.2節で設計した構成に従い，プログラミングパートは，6回で実践した．授業の前半で，その時間の項目の説明（前週の課題Bの解説を含む）をし，残りの時間で演習を行った．5回目以降で実施した図形の生成に関しては，図形ライブラリとしてあらかじめ教師が用意したメソッドを教材として配布した．

表 8 履修状況
Table 8 Course enrollment.

学年	国際交流学部	その他の学部学科	合計
1	9	5(1)	14(1)
2	1	2	3
3	4	2	6
4	2(1)	1(1)	3(2)
合計	16(1)	10(2)	26(3)

かっこ内は履修中断人数

サポートスタッフとしてのTAの確保ができないことから，授業の設計段階ではペアプログラミングを検討したが，結果としては，単にペアを組んで取り組ませることとなった．なぜなら，1人1台のパソコンがあること，課題のコードの行数がそれほど多くないことからである．コードを書き実行すること，授業内の課題Aと宿題の課題Bは，学生ごと個別に行い，コードを考えることやエラーの処理は，ペアで取り組ませた．

ペアでのワークによる考える行動や相互学習の効果として，授業後のフィードバックから「分からない時にペアの子に相談できるのは先生に頼りっぱなしでなく，自分たちで考えながらできるので良かったです」「ペアに教えてあげたり，エラーが出たときに何故だろうと考えたりする上で，解決策，間違った箇所を見つけられたので自分にも力がついた」などの回答があった．

各回の授業は，3章の設計に基づき，表2のスケジュール通りに進めた．1回目は，履修登録期間であることから，履修者が確定してないため，授業のガイダンスを含め，コードを書き，実行する手順を体験をした．4回目までは，数学的な演習問題を中心として行った．5回目以降は，図形生成の教材を用いた．最初は，円を描くメソッドのみを用い，6回目に，線，三角，四角，楕円を描くメソッドを追加した．

2回目の課題では，考え方はできているが，細かい文法でエラーとなるケースがあった．また，学生が所有する実行環境がないパソコンで課題に取り組むケースや，キーボードの入力が遅いことからくる遅れなど，プログラミングではない部分で躓いているケースなどもあった．授業時間外での取り組み環境を構築するためには，設定の手順書を配布して対応した．一部，対応できない学生については，オフィスアワーを利用してサポートをした．提出された課題の一部を**図 4** に示す．

図4の例1では，繰り返しを用いて，階乗の計算のプログラムを書いているが，繰り返しの条件の記述の仕方に戸惑っていることが分かる．例2では，プログラムは正しいが，環境がないことを理解しておらず，コマンドをむやみに打ち込んでいる様子が分かる．

図形生成の視覚化教材の導入後のフィードバックからは，「いつもよりもわかりやすくて，パズル感覚でできた」「久しぶりに自分自身が満足いく結果を出せた」「自分の打ち出したプログラムが実際に画像になるのがすごく面白かった」などの回答があった．

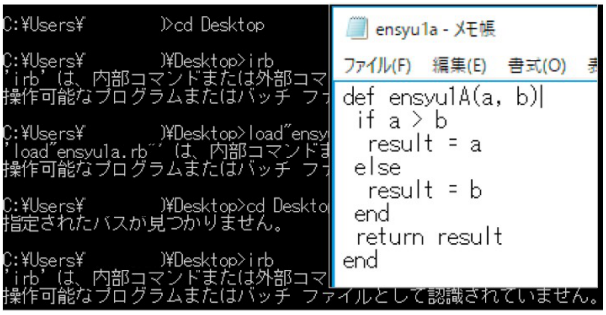
6回目の学生のフィードバックからは，「絵が描けた時は，感動した．色々な絵を描いてみたい」「最初は理解できなかったけど，実際に絵がかけたら，なんとなく理解できました」「だんだん今までやってきたことが形になっていっ

© 2021 Information Processing Society of Japan

```
def fact_(n)
      for i in n*(n-1) ..2
 i=1
end
```

```
irb(main):002:0> fact_ 4
=> 24
irb(main):003:0>
```

例 1：階乗の計算をするプログラム



例 2：環境がないためエラーとなる

図 4 課題の回答例
Fig. 4 Sample assignment answers.

ているのがわかって面白くなってきた」などのポジティブな回答が得られた．絵を描くことにより理解が進んでいるが，一方で，提出されたコードは，図形を単にプロットしたものであった．

他の学生と比較して出席率や課題の提出率が悪い学生からは，基礎知識の理解や考える行動につながっておらず，「まだ，言われたことを入力することしかできない」「難しくてついていけなくて焦ります」などの回答があった．

プログラミングパートでは，4 回目の抽象化の単元で理解されていない部分があった．また，5，6 回目では，図形の生成の仕組みは，理解されたが，その中で，制御構造や抽象化を使うところまでは及ばなかった．この点は，後半のプロジェクトパートで繰り返し活用しながら，補う授業設計により，後半のパートにつなげた．

取り組んだ課題については，5.1 節で，詳しく述べる．

4.3 授業実践：プロジェクトパート

プロジェクトパートでは，1 チーム 4 名を基本として，6 チームを構成した．チーム分けは，プログラミングパートの各学生の提出物や授業時の様子などから見とった理解度と中間アンケート（5 回目の授業後）の個々の回答に基づき，担当教員である第 1 著者が行った（プログラムが得意なものが含まれることと，協力関係が築けることを考慮した）．

中間アンケートの項目には，プログラミングの理解，グループワークに関連すること，その他のスキルに関することの 3 項目をあげ，4 択式の自己回答とした．

各チームの特徴を**表 9** に示す．

リアルな課題を設定することにより，学生のモチベーションを高め，社会における知識の活用イメージを持たせることができると考えた．取り組み課題を設定するにあたり，印刷機を扱う機械メーカに協力を依頼し，完成した作品を実際に印刷して製品化する環境を作った．

表 9 各チームの構成と特徴
Table 9 Team configuration and characteristics.

GID	構成
A	留学生 3 名＋ 1 名の日本人学生
B	国際交流学部 2 名（学年混在）＋音楽学部 2 名
C	1 年生のみ
D	1 年生 3 名＋ 3 年生 1 名
E	1 年生 1 名＋ 3 年生 3 名
F	3 年生 3 名（学部混在）

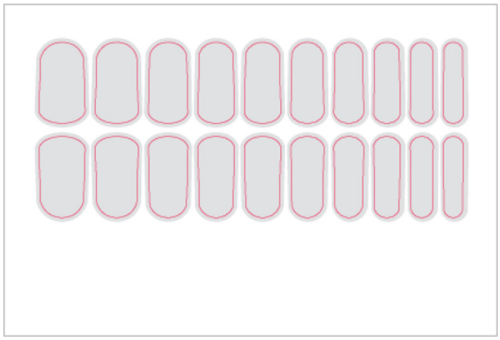


図 5 ネイルシールのフォーマット
Fig. 5 Nail-Sticker format.

ターゲットを決め実際に販売することを目標としたネイルシールの作成を取り上げ，ネイルシールとその PR のための gif アニメーションを作成する課題を設定した．

前半で学習したことを活用して，1 人で 1 作品を作成するのではなく，デザインを複数のパーツに分け，チーム内で分担し，ソフトウェア開発の工程に従って 1 つの作品とすることを条件とした．これにより，各パーツを抽象化してメソッドを作り統合作業を行うこと，アニメーションを作る工程で，制御構造を活用することを期待した．

ネイルシールは，枠内に制作した図案が収まるよう，ハガキサイズ大のフォーマット（**図 5** 参照）を示した．アニメーションのサイズは，特に指定はせず各チームで自由とし，10 コマ程度で構成するよう指示した．

初回（7 回目）の授業には，協力企業の方に参加いただき，課題の発表を行った．これにより，課題がより現実的なものとなり学生たちのモチベーションがさらに高まった．

しかし，プロジェクトパートの最初の 2 回は，コードの検討ではなく，図柄のデザインに集中してしまう傾向が見られた．その結果，ワークシートの記録や中間発表では，すべてのチームで**図 6** に示すようなイラストによる設計図となった．

プロジェクトの進め方については，表 4 の 10 回目の (2) のワークシートの記録に，以下のような記述があり，各チーム様々な工夫を凝らし，ゴールに向かってうまく進める工夫をしていたことが分かる．

- デザインが難しそうなものを 2 人 1 組で分担して協力

© 2021 Information Processing Society of Japan

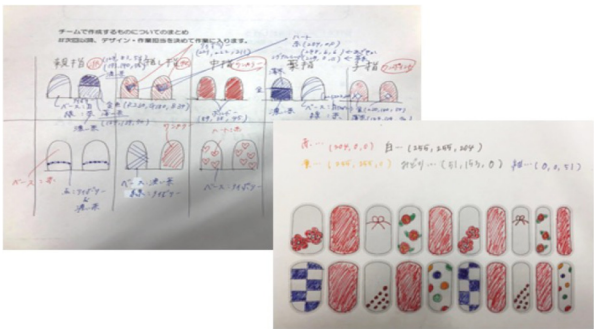


図 6 ネイルシールの設計図
Fig. 6 Nail-Sticker design chart.

する
- 時間がかかる作業を考えて先にやる
- 得意な分野で分担して進める
- 1 人でできること，チームで集まらないとできないことに分ける

途中，天候による授業時間の短縮などの予期せぬ事故が発生し，後半の授業を予定どおり進めることができなかった．特に，13 回目のコードレビューが実施できなかったことは，コードの質を高めることに大きな影響があった．また，成果物の印刷のための提出期限がタイトであったことから，絵を生成することに集中してしまった．

最終回の 15 回目は，学内の通常教室ではなく，協力企業のスペースに場を移して発表会を行った．

グループワークを通じ，チーム内の役割分担やコミュニケーションに注力した．役割分担は，グループワーク 3 回目（9 回目）のワークシートで，チーム内の分担を記録し，役割の明確化をはかった．

最終発表会では，分担についてスライドを用い，各チームが発表した．学生の活動の様子を示すため，そのスライドの一部を第 1 著者が修正し，個人名を消去したものを**図 7** に示す．上段は，D チーム，下段は，F チームが作成したものである．

D チームは，デザインごとに分担をし，ペアで作業をしていた．F チームは，チームメンバが 3 名であったことから，パーツごとに役割分担をし，アニメーションと発表のパワーポイントの作成で分担を分け，個々の特性に応じて，たくさんある仕事を効率を考えて分担していた．

役割分担については，12 回目の個人用のワークシートで次の 2 つの質問をし，作業状況について記録した．ただし，計画どおりに授業の進行ができなかったことから，前掲の表 4 の質問項目とは，異なっている．
- チームで分担して作業を進めていますが，これまでの問題点を教えてください．
- チームで作業していくうえで，個々の能力の違いや特性を活かしてチーム作りをしていますか？ 特定の人に負担がかかったりしていませんか？

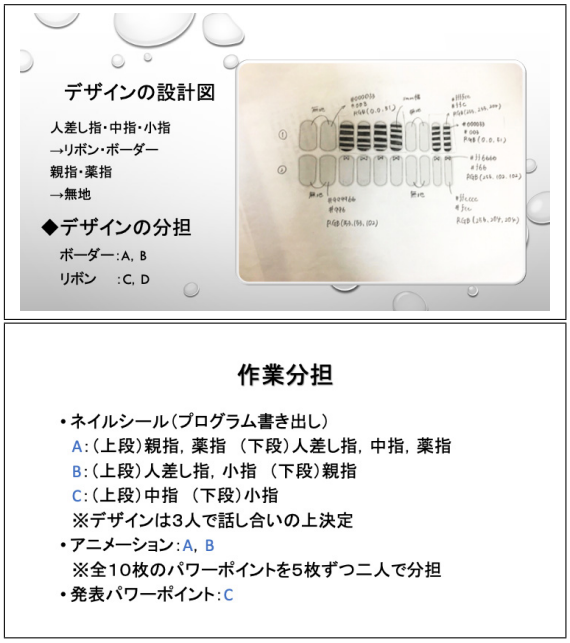


図 7 チーム内での作業分担の事例
Fig. 7 Role allocation in a team.

6 チーム中 5 チームに属していた学生が，「特に大きな問題はない」と回答した．また，「得意分野でそれぞれ役割分担をしており，特に問題はなく，むしろ順調である」と回答した学生は，6 チーム中 3 チームに属していた（到達目標 (4)）．

当初の計画どおりにいかない部分があったが，すべてのチームが自ら考えたデザインでネイルシールとアニメーションの作品を完成させ，期限通り成果物を提出することができた（到達目標 (1)，(2)）．

成果物の一部を，付録 A.2 に示す．

5. プログラミングの課題の分析

5.1 プログラミングパートの課題

本節では，プログラミングパートの 2 回目から 5 回目の提出物について取り上げる．

履修登録のルールにより，1 回目の授業は履修者が確定しておらず，履修人数が流動的であったこと，提出用の学修支援システム（LMS）の準備ができなかったことから，この期間を選択した．

課題 A は，授業時に取り組んだもの（締め切りは，授業の 2 日後）であり，課題 B は，宿題（締め切りは，翌週の授業日の昼休み中）である．各課題は，複数の演習問題で構成されており，学生は，その中から指定されたものや各自で選択したものに取り組む．

ここでは，有効なデータとして取得できた履修者 18 名を対象とする．提出物のコードの行数とファイル数（回答した演習問題の数）についてまとめた結果を**表 10** に示す．

ID は個々の学生に番号を割り当てたものである．GID は，プロジェクトパートのチーム名を表す．課題名のカッ

© 2021 Information Processing Society of Japan

表 10 提出されたプログラムのコードの行数とファイル数
Table 10 Line and file counts of submitted programs.

ID	GID	2A(3)	2B(10)	3A(3)	3B(17)	4A(3)	4B(9)	5A(7)	5B(17)	合計
1	A	8/1	21/2	17/1	17/2					63/6
2	A	14/2	15/3	33/3	24/3	6/1	25/1	108/4	0	225/17
3	B		15/2							15/2
4	B	8/1		68/4						76/5
5	C		17/2	7/1	11/2	16/1	0	17/1	17/1	85/8
6	C							0		0
7	C	8/1	17/2	18/2	18/2	13/1	0	29/2	13/1	116/11
8	C			8/1				-		8/1
9	D	8/1	31/3	16/2	17/2	6/1		17/1	0	95/10
10	D	7/1	25/2	9/1	17/2	26/2	0	17/1	0	101/9
11	D	8/1		11/1	10/1	7/1	22/1	17/1	24/1	99/7
12	D	8/1		13/1						21/2
13	E	7/1	42/6	13/1	61/4			26/1	54/2	203/15
14	E	7/1	20/2	13/1	17/1			16/1		73/6
15	F	8/1	24/2	29/3		16/1		-		77/7
16	F			-				0		0
17	F	105/15	21/4	27/3	16/3	34/5	4/1	51/2	82/3	350/36
18	-		15/2	8/1						23/3
提出人数		12	12	16	10	8	6	13	7	-

コ内の数字は，演習問題の数を表す．表中の数字は，「(コードの総行数)/(ファイル数（回答した演習問題の数))」を表す．() 内の数字のばらつきは，問題中の小問（1-(a) のようなもの）を含めてカウントしたためである．また，課題 B には，課題 A で対象とした問題も含んでおり，重複してカウントしているものもある．

提出されたファイル数が 2 つ以上のものは，複数の演習問題を提出している（したがって意欲の高い）学生であることを表す．また，演習問題の数よりも大きな数字は，複数の考え方で回答していることを表す．

行数は，すべてのファイルの合計であることから，ファイル数のわりに行数が多い場合は，より複雑な（行数の多い）演習問題にチャレンジしたことを表している．

各課題は，コードと実行結果・コードに対する考察・アンケートから構成するように指示している．“0” は，課題は提出されたが，コード・コードに対する考察が含まれていなかったこと（アンケートのみ）を示す．“-” は，実行結果のみで，コードが含まれていないため，行数がカウントできなかったものを表す．

次に，各課題で取り組んだ演習問題を**表 11** に示す．演習問題の番号は，付録 A.1 のものと同様である．

課題は，完璧にできあがらなくても，分からなかったことやできなかったことを説明したものでも可としたが，提出状況は良くなかった．

この原因は，学生が過去の経験から「完璧にできあがったものしか提出してはならない」と考えていたことと，大学における宿題文化のないことの 2 点が考えられる．また，学修支援システム（LMS）の使い方に慣れていないことから，提出ができないものもいた．

表 11 提出された課題の演習問題
Table 11 Practice exercises chosen by students.

課題番号	演習問題の番号 (人数)
2A	1a(12)，1b(2)，1c(1)
2B	1a(1)，1b(6)，1c(4)，2(6)，3a(4)，3b(3)，3c(2)
3A	1a(11)，1b(9)，1c(5)
3B	1a(1)，1b(3)，2(4)，3(5)，6a(3)，6d(1)，7(1)
4A	1a(6)，2a(1)，2b(2)，2c(1)
4B	1a(1)，2a(2)，2b(1)，2c(1)
5A	2a(6)，2b(2)，2c(1)，2d(1)，3a(2)
5B	2a(4)，3a(1)，3b(3)，3c(1)，3d(1)

留学生は，日本語能力の問題から読み書きに時間がかかり，時間どおりに提出することが困難であったと，後日，該当の学生との会話から事情を得た．

表 10 からは，3 回目に提出のピークがあり，その後，課題が難しくなり提出状況が悪くなった様子が読み取れる．5 回目は，絵を描く課題であり，再び学生の興味が現れるが，宿題までには手が及ばなかったものと推察される．

この時期は，大学祭の準備とも重なった等の要因もあるが，半分以上提出している学生は 11 名しかおらず，全回提出できた学生は 4 名のみであった．

課題 A については，授業時間内に実施し，まとめる時間を作って提出させていることから，課題 B と比較し実行可能なプログラムと実行結果が提出されていた，しかし，課題 B については，宿題であり，課題 A と比較すると提出率

© 2021 Information Processing Society of Japan

が低く，「わからない」「エラーがでてしまった」などの回答もあり，提出された物すべてが正解というものではなかった．ただし，実行時にエラーが出ているケースでも，考え方はできており，細かい文法のミスによるものであった．

表 11 からは，A 課題は，授業中に教員の説明に続いて課題に取り組んでいることから，取り組んだ課題に偏りが見える．一方，B 課題は，宿題であり，各自が自分にあった課題を選択して取り組んでいることから，選択した課題も様々であったことが分かる．これは，3.3 節での設計方針で述べたように，「学生は自分のレベルに応じて選んで解答する」に該当し，この方針が機能していたと推察する．

制御構造に関する課題は，2A～3B が該当する．この提出状況は，2B（52.17%）と 3A（69.57%）となっており，10 回の中で提出率は一番高かった．

課題 2B の中で行ったアンケートでは，提出者のうちの 63.6%の学生が分岐について理解し，45.5%の学生が繰り返しについて理解したと回答されていた．

提出されたコードやコメントからは，2A では，テキストの模範回答に習い，入力して実行するケースが多くあったことが読み取れた．2B からは，自分で考え，エラーを修正する行動が見られるようになった．一方で，分岐や繰り返しの考え方はできているが，end を入れる場所の理解が不十分なケースもあった．

抽象化については，4A，4B が該当するが，課題のアンケートからは，「RPN 電卓の仕組みについてよく理解出来たが，コードを書くところが難しかった」との回答があった．数学が苦手なことから，RPN 電卓の仕組みを理解することに力を入れ，プログラミングの知識としての抽象化の理解にまで至らなかったと考える．演習問題としては，加減乗除の計算をする比較的易しいものであったが，本実践授業の学生が興味関心を持てる内容ではなく，抽象化を学習する目的ではない部分でマイナス要素があったと推察できる．これが要因となり，提出率の低下や理解不足につながったと推察する．

5A からは，図形の生成の可視化教材となり，前述したとおり，分かりやすく理解が進んだことにより，提出率が上がっている．

しかし，Ruby のプログラムの実行環境と画像ファイルの変換については，同じコマンドプロンプトを使って行うため，この相違について，理解が難しい学生がいた．補足資料を提示して，説明を行ったが，手順が増えることで理解が追いつかない学生がいた．

この点は，プログラミングの問題だけでなく，コンピュータの仕組みの理解や操作について，大学入学以前の経験が乏しいことが要因として考えられる．

授業時の見取りからは，まったく理解できずに取り組むことができない学生はおらず，出席率も高かったが，課題の提出には反映されなかった．提出された課題の質については，入門レベルということもあり，課題 3B までは数行程度のものであり，大きな問題はなかった．

表 12 最終課題におけるネイルシールのプログラムの状況
Table 12 Status of programs for final nail-sticker assignment.

GID	A	B	C	D	E	F
ファイル数	19	4	1	1	1	3
合計行数	2,396	515	279	94	102	258
制御構造	0	0	0	0	0	0
作品：ネイルシール	○	○	○	○	○	○
作品：アニメーション	○	○	○	○	○	○

しかし，以上の結果から，4 回目以降の例題や数学的な教材については，見直す必要があることが分かった．図形描画の教材の活用は，プログラミングの基礎知識を学んだ上での導入を考え，授業を設計した．提出状況や学生の回答から効果があるが，4 回目までの教材との接続を含めて，導入のタイミングを再考する必要があることが分かった．

5.2 プロジェクトパートの課題

プロジェクトパートでは，チームでネイルシールとアニメーションの 2 つの課題に取り組んだ．本節では，このうちネイルシールのプログラムについて取り上げる．

チームごとのファイル数とコードの行数などを調べた結果を，**表 12** に示す．チームの構成は，表 9 と同様である．

プログラミングパートでは，プログラムに自信がなく課題の提出ができなかった学生も，プロジェクトパートの終わりにはチーム全体で少なくとも 100 行程度のプログラムが作れるようになり，作品も完成している．特に，A，B グループは，前半の課題の提出状況と最終課題の状況を比較すると，作成したプログラムの数や行数に大きな成長が見られる．

当初の予定では，提出するファイルは，各自が作ったパーツを 1 つのファイルに統合して提出されることが理想であった．しかし，次の理由から統合の作業ができず，チームごとに大きな差が生じたと推察する．

- 授業時間の不足（天候による授業時間の短縮）による作業時間の不足
- 課題の提出期限
- 複雑なデザイン

第 1 著者の授業時の見取りでは，チーム内での分担に多少の負担の差はあったものの，4.3 節で述べたように適切に分担され，それぞれが手を動かしており，また他のメンバのコードについて意見を互いに述べたり，助け合いをしたりするなどのことも多くの学生ができていた．

最終レポートでは，23 人中 19 人の有効回答があり，そのうち，17 名（89.47%）が学習した範囲でプログラミングについて理解したという内容の記述をしており，また，8 名（42.11%）が自分たちで考えた絵をプログラムで実現し

© 2021 Information Processing Society of Japan

たことについて述べている．ただし，プログラムの水準については，図形を順にプロットしていくコードであるために行数が多くなっており，制御構造や手続きを組み合わせたプログラムで 100 行のものが作れているわけではない．

この原因としては，前述の図 6 でも示したように，作成する絵のデザインにこだわり，図形をどのように組み合わせたら，そのデザインに近づけるのかにこだわり，具体的な絵を考えることに注力しまったことが考えられる．

しかし，自分たちで考えた課題に対してコードを書き，表現でき，作品が完成していることから，おおむね到達目標 (1) は達成できたといえよう．

本提案のカリキュラムでは，前半のプログラミングパートで理解が不十分な点は，プロジェクトパートで補うことを考え設計した．しかし，実践授業では，プログラミングパートの演習問題やプロジェクトパートへの接続や時間数の問題など，いくつかの課題があり，制御構造や手続きの活用までは至れず，良い結果は得られなかった．

この課題点は，図形描画のメソッドの導入のタイミングと演習問題の例題を見直すことで，改善ができると考え，2018 年度以降，改善を行い，実践を続けている．15 回の全体の構成は，図 2 に示した構成で，2017 年度と同様の流れで実施したところ，制御構造および抽象化を実現することができた．

プログラミングパートでは，テキストの例題を 1 回目から図形を描画する内容に修正し，制御構造などの基礎知識を学びならが図形の生成について繰り返し学べるように改善をはかった．プロジェクトパートでは，課題を横浜をテーマとした A4 サイズのクリアファイルのデザインに修正した．

その結果を**表 13** に示す．また，付録 A.3 に，2019 年度の事例を示す．

この回も提出するファイル数は，1 つのファイル（両面のデザインの場合は 2 つ）にまとめることが理想であったが，2 チームは，まとめることができなかった．しかし，制御構造の活用やメソッド化などは，実現できており，前半の例題を見直すことにより，制御構造・抽象化の部分を補うことができたと推察できる．

以上から，本実践授業においては，問題もあったが，2018 年度以降，問題点が解消されていることから，15 回の構成としては，妥当な構成であったと考える．

表 13 2018 年度の結果
Table 13 Result of 2018 final assignments.

GID	A	B	C	D
ファイルの数	3	1	1	4（両面）
メソッドの数	10	9	4	5
制御構造	6	3	4	5

6. コンピテンシー評価の分析

PBL による取り組みで期待される効果として，コンピテンシーの育成がある．本実践授業では，授業の 2 回目と最後の回に学修支援システム（LMS）を通じて，コンピテンシー評価を行った．その結果を，**表 14** に示す．

各質問項目に対する回答は，レベル 1（基本行動）からレベル 3（卓越行動）の 3 段階の自己評価による選択であり，これを数字の 1〜3 に置き換え計算した．対象は，履修者 23 名のうち 2 回とも回答した 16 名である．

12 項目のうち半数の 6 項目（表 14 の*，**）で，t 検定により有意水準 5%で有意差ありと判定された．また，4 項目（表 14 の ⋆）で有意傾向あり（$0.05 \leq p < 0.10$）と判定された．特に，課題分析力，解決策立案力，メンバの多様性（多様性の活用）の 3 項目では，t 検定により有意水準 1%で有意差ありと判定された．

課題分析力，解決策立案力に対しては，本実践授業で対象とした学生は，単に図形をプロットするだけのプログラムでも，出力される絵には微妙な色の違いも含めてこだわりがあり，この色やデザインの出力結果の修正やプログラム実行時のエラーの対応などの作業を通じての効果である

表 14 コンピテンシー評価
Table 14 Competency evaluation.

項目	1 回目平均	2 回目平均	t 値	p-value
	2-1 回目平均		95%信頼区間	
傾聴力・	1.412	2.000	2.582	0.020 *
読解力	0.588		0.105〜1.071	
記述力	1.471	1.706	1.167	0.260
	0.235		−0.192〜0.663	
議論力	1.235	1.588	1.852	0.083 ⋆
	0.353		−0.0511〜0.757	
課題発見力	1.412	1.824	1.951	0.069 ⋆
	0.412		−0.036〜0.859	
課題分析力	1.176	1.882	2.634	0.018 *
	0.706		0.138〜1.274	
解決策	1.118	1.941	4.667	0.0003 **
立案力	0.824		0.138〜1.274	
学習力･応用力	1.235	1.471	2.219	0.041 *
・知識創造力	0.235		0.010〜0.460	
役割認識・	1.353	1.765	1.595	0.130
主体性・協働	0.412		−0.135〜0.959	
計画力	1.471	1.941	2.057	0.056 ⋆
	0.471		−0.0144〜0.956	
共有データ	1.529	2.000	2.057	0.056 ⋆
	0.471		−0.014〜0.956	
要求仕様	1.412	1.882	2.219	0.041 *
の利用	0.471		0.021〜0.920	
メンバーの	1.294	1.941	4.400	0.0004 **
多様性	0.647		0.335〜0.959	

* ：$p < 0.05$，** ：$p < 0.01$，⋆ ：$0.05 \leq p < 0.10$
$n = 16$

© 2021 Information Processing Society of Japan

と推察する．

本研究のみで追加した到達目標に関連する 4 項目については，要求仕様とメンバの多様性の 2 項目において統計的に有意判定された．

役割認識（チームワーク）・主体性・協働については，「一人一人やるべき事をしっかりと考え行動」「滞ったプログラミングの分は得意な人がサポートするなどして補った」など，最終レポートにも該当する回答があった．第 1 著者が意味を理解し，該当するものをカウントしたところ，19 名中 16 名（84.2%）が回答をしていた．統計的な有意判定されなかったが，統計的に有意傾向ありと判定できる範囲であり，最終レポートの回答と総合して解釈すると役割認識（チームワーク）・主体性・協働についても有効であったと推察できる．

上述のことから，本提案のカリキュラムは，15 回という限られた時間の中でも，コンピテンシーの向上に有効に働いたと推測できる．この結果は，カリキュラムの妥当性を示唆するものにもなりうると考える．

7. 追跡調査

2017 年度に履修した学生に対して，履修後の活動にどのような効果が出ているのかを調べるため，2020 年 12 月 17 日～2021 年 1 月 7 日の期間で，オンラインでの記述式アンケートを行った．

回答者の内訳は，在学生 5 名（就職活動中 1 名，SE 内定 2 名，その他の職に内定 2 名），既卒生 3 名（すべて IT 関連企業ではない）である．個人情報に触れるため，既卒生の現在の職業は伏せる．

質問項目は，授業の履修後の活動への効果を調べるために，次の 4 項目を設定した．

- 就職を考える際や就職活動をする際に，本授業がどのように影響したか？
- 履修後の活動や仕事でどのように役立っているか？（到達目標を示し，その観点から）
- 授業の際に，もっとやっておけばよかったことについて
- 授業の履修の有無により，他の社員と違いがあるか？（既卒生のみ）

SE に内定している 2 名は，授業を受けたことが就職を決める際に直結したと回答した．現在就職活動中の学生も，大学入学時は視野に入れていなかった IT 企業や SE を視野に入れ，将来を考えるきっかけになったと回答した．

業種は様々であるが，「グループで協力し，一つの目的に向かってそれぞれの作業を進めることを意識出来るようなり，選考の際のグループワークの場面で，授業を通して学んだ，一方的な意見を押し付けるのではなく，他人の意見を聞き入れ，互いの意見を尊重して最終の結論に導くことが実践できた」「授業ではグループで一つの課題を役割分担して行い，タイムマネジメント，進捗状況なども共有してきた．その経験が就職活動面接時におけるグループディスカッションに役立った」など，全員が就職活動の選考面接において，プロジェクトパートでのグループワークの経験が役に立ったと回答した．

これらの回答は，到達目標 (2)，(3)，(4) の達成に該当する．

近年，様々な授業でグループワークを取り入れているが，何週にもわたり同じメンバで活動することがないと聞いている．単に数名が集まってのワークであり，分担が決まりゴールに向かって何かに取り組むようなグループワークの経験が乏しい．そのため，多くのグループワークは，仲良しグループでの活動の域を抜けることがないようである．

しかし，本授業実践では，プログラミングをともなうことにより，論理的に考えプログラムを書くことと役割を分担することを通しての学びが大きく，学年も専攻も関係なく，初めて授業で顔を合わせた仲間と 1 つのものを作り上げていくプロセスは，その後の活動に役立っていることが分かった．

また，既卒生からは，プログラマーやスペシャリストにならずとも，職種にかかわらずシステムをともなう業務があり，「学習経験からトラブル時の初期行動が取れるようになった」「学生時代に情報技術系のことに慣れ親しんでいる者は業務も効率的に進められている」「情報共有システムを使い，仕事効率を大いに高めていくことの重要さを学び，実践につなげていくことができた」などの回答があった．この回答は，既卒生 3 名全員からの回答であり，授業を履修したことによる，効果であると推察できる．

これから SE として就職する学生は，授業後の活動においてどのように役立っているかに対し，「プログラミング自体に関心を持つようになりパソコンを使って何かを作り出すことに興味を持った」「ニュースなどでも IT に関わる内容に関心を持つようになった」と回答した．

アンケート後に，この 2 名に対して，履修後のプログラミングの学習の継続や入社までの事前研修でプログラミングを学んでいるかを聞いたが，履修後，継続してプログラミングの学習をすることはなく，事前研修もまだ始まっていないとのことで，コードを書くことに対してどのように影響しているかは調査できなかった．

今回の調査では，アンケート期間が短かったこともあり，プログラムを書いたり，ソフトウェア開発に直接関わったりするような内容のコメントは，収集することができなかった．

コードを書くことに対しては，授業後，環境がなくなってしまい覚えていないと回答しているものが多数いたが，チームで働くこと，IT を活用する場面においては，授業での経験が有効に働いていることが分かった．

© 2021 Information Processing Society of Japan

8. 考察

8.1 プログラミングの観点から

多様な学生の集団が履修することに配慮し，簡単なコードを書くことにより，プログラムがどのように作られ，どのように動いているかの原理を理解することを念頭にカリキュラムを構築した．

1〜4 回目までは，数学的な例題を用い，制御構造・抽象化の学習が終わった 5 回目から図形ライブラリを活用した視覚化教材を使った．5 回目以降の視覚化教材は，4.2 節で述べたように，学生の理解が進み，視覚化教材の有効性が伺えた．しかし，5 章で明らかになったように，コードの質までは高めることができなかった．

履修した学生の特性を考えると，数学的な問題を多用した演習問題の修正が必要であり，先行研究 2.3 節 [31] にもあったように，図形生成の導入部分で，反復練習の時間の確保が必要であったと考える．

今回の実践授業で対象とした学生は，具体的に実現することに関しては興味を持ち，デザインや色を修正しながら楽しく取り組んでいたが，設計をし，抽象化することについては時間をかけて練習する必要があったと考える．

この問題を回避するためには，前半の例題の修正（特に 4 回目および図形生成の教材導入のタイミング）が必要である．また，本実践において，実施できなかった部分を含め，中間発表（11 回目）やコードレビュー（13 回目）の進め方を見直す必要があることが分かった．プログラミングパートの課題については，学生のレポート「プログラミングの細かい内容よりも絵などから入っていったほうがやはりわかりやすいのではないかと思います．文章を先に学んでも活用の仕方が難しいです」の回答にも現れている．

この点は，2018 年度以降修正を加え，機能していることを図 13 および付録 A.3 に示した．

演習問題では理解されていたと思われる分岐や繰り返しは，プロジェクトパートでの実践においては，絵を描くことに注力してしまい運用するところまでには至れなかった．

知識の習得と活用は，数学の学習経験を例にとると，例題を学び，演習問題をこなして終わってしまい，学習した知識を応用する経験にまで至らないことがある．本実践を行った環境では，このような経験が乏しい学生が対象であり，プログラミングでも同様に応用することが難しく，前半で学んだ知識を応用して後半のプロジェクトにつなげることができなかったと推察する．しかし，最終発表では，このことに気づき，繰り返しや分岐を活用する部分ついて理解しており，今後の課題として発表するチームもあった．このことから，まったく理解できていなかったわけではなく，到達目標 (1) の「制御構造について，説明することができる」が，達成できたと考えることができる．また，制御構造・抽象化を活用するまでは至らなかったが，説明することはできており，最終課題のネイルシールやアニメーションができあがっていることは到達目標 (1) の「制御構造について，説明することができる」以外の 3 項目に該当し，これらが達成できたといえる．

以上のことから，結果としては，到達目標 (1) は，おおむね達成されたと考えることができる．

最終レポートには，「全 15 回が全て座学でプログラミングのことが勉強できればいいのにと思っていたところもあるが，それではプログラミングの知識は増えていかないんだと多くの課題と授業内外の実践を通して実感した」という回答があった．与えられた演習問題を 1 人で解き，単にコードを書くだけではなく，プロジェクトを組み合わせた 15 回の授業を通じ，人と関わることによりプログラミングの知識の幅を広げられたものと推察できる．

8.2 プロジェクトの観点から

文系大学においては，座学中心の授業が多いため，実践的な学びの場が限られてしまう．座学中心に陥りがちなプログラミング入門科目に，成果物の作成をともなう実践的なプロジェクトを組み合わせることは，挑戦的な試みであった．

3.4 節で述べた，PBL の欠点の克服について，授業時のワークシートの記録と最終レポートから考察する．

- 欠点 1 「学生も教員も多くの時間を要する」について，教員は，通常の授業と同程度の時間を本授業に費しており，また学生の授業時のワークシートの記録でも「過大な時間が掛かった」のような記述は見られなかったことから，欠点 1 は，克服できていたと考える．教員の対応については，15 通のメールのやりとりが記録として残っている．この程度の量であれば，特に大きな負担にはならない．
- 欠点 2 については，カリキュラム設計において，教材を制限し，教員がファシリテーターとなり，改善を図ることを考えた．各回の授業の初めに説明を加え，終わりにはワークシートへの記録をし，プロジェクト管理をすることにより，各チームをある程度ファシリテートできていた．数値化して示せるデータはないが，各チーム，仕上がったネイルシールの出来に達成感を感じていることから，欠点にはなっていなかったと推察する．
- 最終レポートでは，計画，役割の認識に関する記述が多く，適切な計画や役割認識ができていたことを示唆し，4.3 節の図 7 からも，欠点 3 「活動量の差が生じる」を回避できていた可能性がうかがえる．最終レポートの記述では，「役割分担できた」と 78.95%の学生が書いており，一方，「分担が偏っていた」のような記述は見られなかった．
- 欠点 4 「メンバーの能力やスキル差によりアウトプッ

© 2021 Information Processing Society of Japan

トの質に差」については，最終レポートでは，「メンバーと助け合って（84.21%）」や，「得意な分野を活かして（42.11%）」などの記述があり，欠点 4 を克服できている可能性がある．

以上のことから，PBL の欠点は，本実践においては問題となっていなかったことが分かる．この結果から，欠点を考慮して PBL を導入できたと考える．

上述の内容は，到達目標 (2)，(4) に関係しており，到達目標 (2)，(4) の到達を示唆するものであると考える．

各チームがそれぞれ相談して成果物を完成できたことは，「問題認識から課題解決までを主体的に学ぶ」を実現したことに相当する．また，各チームが期限内に成果物を完成させたことは，「一定期間内に目標を達成」を実現したことに相当する．このことからも，到達目標 (2) が達成できたといえる．

次のレポートの回答からは，授業を通じて考える行為（到達目標 (2)）に該当し，プロジェクトを通じて多面的・多角的に考える経験をしていたことが分かる．

- プログラミングの授業を受講したことで，他のチームの作品を見て「これはこういう風にプログラミング作成している」「なぜ，これはこういう線が出せるのだろう」と考えられるようになった．
- 物事を進める時に，ある一定の視点で考えるのではなく，視点を変えてみることで，今後何かをする時に広い視野で物事を考えられ自分たちができることも増えるのではないかと思いました．

Ritchhart らは，思考の可視化は，理解を深めることも助けることに加え，思考と学習の関係を示すことである [44] と述べており，図形の生成を用いたプログラミングの学習は，可視化により学習者の思考と理解を明確化できたことから，学習者のモチベーションを向上させ，満足いく経験になったと推察できる．

小林は，大学という場の知を知識ではなく，知の行為という視点から考え直し，大学で何を学ぶかについて述べている [45]．

学生のレポートの記述にも複数，プログラムを書くことを通じて問題に遭遇し，それを解決するためのプロセスから学んだとあり，本提案のカリキュラムから得た学びは，まさに，小林のいう知の行為であり，大学で何を学ぶかという点において有効であったと考える．

課題に取り入れたネイルシールの制作は，本授業実践で対象とした女子大においては，学生のモチベーションも向上させられ，授業後の満足度を高められるエンゲージメント [46] を実現できたと考える．しかし，男女混在の大学においては，学生の興味関心も異なると考えられるため，課題の内容に関しては学習者の集団を意識して検討する必要があるであろう．

到達目標 (3) については，プロジェクトの課題でターゲットを設定したことにより，授業内で要求仕様についての説明し，意識させるような配慮をした．プロジェクトパートの中間発表のワークシートでは，「口頭発表だけでよくわからなかった」「視覚的な発表はわかりやすかった」などの記録があり，正確に伝えることについては理解が進んだ様子が残っている．コンピテンシー評価の分析から，統計的な有意差は出ているが，他の項目と比較して，扱いが難しかった．

到達目標 (4) については，4.3 節で述べたように，6 チーム中 5 チームの学生について該当する記録を残しており，PBL の欠点 3，4 でも述べたように，メンバの得意な分野を活かして役割分担をしプロジェクトを進められていたことからも，この目標は達成できたといえる．

到達目標 (5) については，15 回の授業を通じて，タッチタイピングができるようになったり，課題提出や発表資料の作成などを通じて，様々なツールが使いこなせるようになったり，チームでの情報共有のためのデータ共有やメールおよび学修支援システム（LMS）の日々の利用などから，達成できたと判断した．

上述のことから，各チームでそれぞれ相談して役割分担をし，成果物を構築できたこと，期限内に成果物を完成させたことを総合すると，到達目標の (2)〜(5) はおおむね到達できたと考える．

9. まとめ

本論文では，誰もが学ぶべき 1 科目の入門レベルのカリキュラムと位置づけ，プログラミング入門にソフトウェア開発プロジェクトの知識を盛り込むことを検討した．1 つの解決方法として，PBL を組み合わせた方法を検討し，前半 6 回をプログラミング入門とし，後半の 9 回でその知識を活用したプロジェクトで構成したカリキュラムを構築した．

入門レベルにおけるプロジェクトでは，ソフトウェア開発プロジェクトは困難であることから，絵の生成をソフトウェアに見立て模擬化し，チームで 1 つの絵を完成させる方法により，プロジェクトを実現する課題を用いた．

15 回の授業でプログラミングとプロジェクトのすべての知識を身につけたといい切ることは難しいが，プログラミングを学び，プロジェクトを行う中で，到達目標 (1)〜(5) をおおむね達成している．このことから，90 分 15 回の中で，入門科目として成立するカリキュラムが構築できたと考える．

今後の活用として，2022 年度から実施される，高等学校の「情報 I」で扱われることになっている「情報デザイン」への応用も期待できる．

「情報デザイン」は，平成 30 年高等学校学習指導要領解説（情報編）第 1 部第 2 章第 1 節 2(2) [47] で，次のように示めされている．

© 2021 Information Processing Society of Japan

「情報 I」で扱う情報デザインは，効果的なコミュニケーションや問題解決のために，情報を整理したり，目的や意図を持った情報を受け手に対して分かりやすく伝達したり，操作性を高めたりするためのデザインの基礎知識や表現方法及びその技術のこと

特に，プロジェクトパートのチームで絵をデザインして作品をつくる工程において，意図した絵をコードを書き実現する中でコミュニケーション，情報の明確な伝達．情報の整理などを含む活動ができる．教材は，シンプルで簡単であることから，高等教育に限らず，初等中等教育への応用も期待できる．

本提案のカリキュラムは，2018 年度以降も現在まで継続的に実施しているだけでなく，履修後も Ruby World Conference [48], [49] で発表を行うなど，発展的な学びやモチベーションも保たれている．履修者全員が高度なプログラムが書けるようになったわけではないが，プログラミングの向き不向きや直接関係するしないにかかわらず，履修後の活動に有効に働いているという卒業生からの感想もある [49]．

履修後の効果については，7 章で簡単な調査を行った．今後，さらに範囲を広げて調査を進めていく必要性を感じており，計画をしている．

我が国における大学の環境は，文部科学省『学校基本調査』（令和元年）[50] によると，学部学生数の 61.4%が文系を占めており，義務教育におけるプログラミング教育が浸透するまでのしばらくの期間，これらの学生の多くはプログラミングを学ばないままに社会に出て行くこととなる．これからの情報化社会においては，誰もが基本的なプログラミングの素養を求められる社会 [51] となっており，より良い社会を築いて行くためにも，プログラミングを学ぶ適正な環境の整備が望まれる．

全体として，本研究の手法であれば，教員の負担もそれほど大きくならず，多くの学校で容易に実施可能であると考える．本提案のカリキュラムが広く活用されることを期待したい．

謝辞 本研究および本論文の執筆に関して，津田塾大学の小川貴英名誉教授ならびに，電気通信大学の角田博保元准教授には，たくさんの時間を割いていただき，大変有益な助言をいただきました．この場を借りて深く御礼申し上げます．

参考文献

[1] 首相官邸：世界最先端 IT 国家創造宣言（2013 年 6 月 14 日）（オンライン），入手先 ⟨http://www.kantei.go.jp/jp/singi/it2/kettei/pdf/20130614/siryou5.pdf⟩（参照 2020-08-23）．

[2] 日本学術会議：大学教育の分野別質保証のための教育課程編成上の参照基準情報学分野（オンライン），入手先 ⟨http://www.scj.go.jp/ja/info/kohyo/pdf/kohyo-23-h160323-2.pdf⟩（参照 2020-08-23）．

[3] 日本学術会議：情報教育課程の設計指針—初等教育から高等教育まで（オンライン），入手先 ⟨http://www.scj.go.jp/ja/info/kohyo/kohyo-24-h200925-abstract.html⟩（参照 2021-01-23）．

[4] 萩谷昌己：「情報教育課程の設計指針」解説，情報処理，Vol.62, No.4, pp.e61–e68 (2021).

[5] 和田 勉：小中高等学校の新学習指導要領とそれを取り巻く情報教育の状況，情報処理，Vol.59, No.8, pp.742–746 (2018).

[6] 萩谷昌己：未来投資会議における大学入学共通テストに情報の試験を入れる方針に賛同する提言について—大学情報教育体系化の必要性，情報処理，Vol.59, No.9, pp.778–781 (2018).

[7] 中山泰一：大学入学共通テストへの情報の出題について，ニューサポート高校「情報」，Vol.18, pp.6–7 (2021).

[8] Hagiya, M.: Defining Informatics across Bun-kei and Ri-kei, *Journal of Information Processing*, Vol.23, No.4, pp.525–530 (2015).

[9] 文部科学省：中央教育審議会答申「学士課程教育の構築に向けて」（オンライン），入手先 ⟨https://www.mext.go.jp/b_menu/shingi/chukyo/chukyo0/toushin/1217067.htm⟩（参照 2020-08-23）．

[10] 渡辺敦司：情報 A が再び現象に 2012 年度高校教科書採択状況—文科省まとめ（下），内外教育，2011 年 12 月 13 日号，pp.8–15 (2011).

[11] 渡辺敦司：英語 III，高学年周期でも冊数減 19 年度高校教科書採択状況—文科省まとめ（下），内外教育，2019 年 2 月 22 日号，pp.12–19 (2019).

[12] 文部科学省：「超スマート社会における情報教育の在り方に関する調査研究」報告書（オンライン），入手先 ⟨http://www.mext.go.jp/a_menu/koutou/itaku/1386892.htm⟩（参照 2017-07-05）．

[13] 高橋尚子：国内 750 大学の調査から見えてきた情報学教育の現状 (3) 一般情報教育編，情報処理学会学会誌「情報処理」，Vol.58, No.6, pp.526–530 (2017).

[14] Potter, J.：英国政府内閣府業務自動化への取り組み—RPA 導入を政府横断で推進する Centre of Excellence，行政&情報システム，2019 年 8 月号，一般社団法人行政情報システム研究所 (2019).

[15] Weinberg, G.M.: The Psychology of Computer Programming: Silver Anniversary Edition Annual, Subsequent Edition, Dorset House (1998). 木村 泉，久野 靖，角田博保，白浜律雄（訳）：プログラミングの心理学—または，ハイテクノロジーの人間学 25 周年記念版，毎日コミュニケーションズ (2005).

[16] 吉田智子：文系学部の情報教育へのプログラミングの導入—PEN を用いた実践例，情報処理学会研究報告コンピュータと教育（CE），Vol.2008-CE-095, pp.71–78 (2008).

[17] 松本このみ，吉田智子：文系学部における PEN を用いたプログラミング授業の実践例—繰り返し処理の理解を助ける教材の提案，情報処理学会研究報告コンピュータと教育（CE），Vol.2011-CE-109, pp.1–7 (2011).

[18] 松澤芳昭，酒井三四郎：ビジュアル型言語とテキスト記述型言語の併用によるプログラミング入門教育の試みと成果，情報処理学会研究報告コンピュータと教育（CE），Vol.2013-CE-119, pp.1–11 (2013).

[19] 石川高行：文系大学生のためのオブジェクト指向プログラミング教育実践，情報処理学会情報教育シンポジウム 2003 論文集，pp.77–82 (2003).

[20] 河村一樹：一般情報教育におけるプログラミング教育のあり方について，情報処理学会研究報告コンピュータと教育（CE），Vol.2011-CE-108, pp.1–8 (2011).

[21] 西田知博，原田 章，中西通雄，松浦敏雄：プログラミン

© 2021 Information Processing Society of Japan

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），Vol.3, No.1, pp.26–35 (2017).
[22] Bransford, J., Brown, A. and Cocking, R.: How People Learn: Brain, Mind, Experience, and School, National Research Council (2000). 森　敏昭，秋田喜代美（監訳）：授業を変える—認知心理学のさらなる挑戦，北大路書房 (2002).
[23] Papert, S.: Mindstorms: Children, Computers, and Powerful Ideas, Basic Books, Inc., New York (1980). 奥村貴世子（訳）：マインドストームー子供，コンピューター，そして強力なアイデア（新装版），未来社 (1982).
[24] 文部科学省：高大接続システム改革会議：高大接続システム改革会議「最終報告」（オンライン），入手先 ⟨http://www.mext.go.jp/component/b_menu/shingi/toushin/__icsFiles/afieldfile/2016/06/02/1369232_01_2.pdf⟩（参照 2018-03-08）.
[25] 中山留美子：アクティブ・ラーナーを育てる能動的学修の推進における PBL 教育の意義と導入の工夫，21 世紀教育フォーラム，Vol.8, pp.13–21, 弘前大学 21 世紀教育センター (2013).
[26] 池本有里，鈴木直美，近藤明子，山本耕司：学生のコンピテンシー育成を目指す PBL 型教育プログラムの実施と考察，四国大学紀要，No.42, pp.1–11 (2014).
[27] 内田康之：問題解決力を身につける創造教育の実践，工学教育，Vol.63, No.2, pp.2_40–2_46 (2015).
[28] 経済産業省：社会人基礎力に関する研究会・中間とりまとめ (2006).
[29] 三重大学高等教育創造開発センター：Problem-based Learning 実践の方法論報告書（国際シンポジウム・ワークショップ）(2006).
[30] 美馬のゆり（編著），冨永敦子（著），田柳恵美子（著）：未来を創る「プロジェクト学習」のデザイン，近代科学社 (2018).
[31] Nuutila, E., Törmä, S. and Malmi, L.: PBL and Computer Programming — The Seven Steps Method with Adaptations, *Computer Science Education*, Vol.15, No.2, pp.123–142 (2005).
[32] Uchida, N. and Kuno, Y.: Programming Education in a Women University in Japan: A Case Study of Performance-Based Learning in Liberal Arts, *Proc. International Congress on eLearning* (*ICE2019*), pp.175–189, Philippine eLearning Society (PeLS) (2019).
[33] 内田奈津子：ぺた語義：プログラミング入門をプロジェクトでやってみた，情報処理学会学会誌「情報処理」，Vol.59, No.3, pp.268–271 (2018).
[34] 久野　靖：プログラミング入門をどうするか：1．プログラミング教育／学習の理念・特質・目標，情報処理学会学会誌「情報処理」，Vol.57, No.4, pp.340–343 (2016).
[35] McDowell, C., Werner, L., Bullock, H. and Fernald, J.: The effects of pair-programming on performance in an introductory programming course, *Proc. 33rd SIGCSE Technical Symposium on Computer Science Education* (*SIGCSE '02*), pp.38–42, Association for Computing Machinery (2002).
[36] McDowell, C., Hanks, B. and Werner, L.: Experimenting with Pair Programming in the Classroom, *Proc. 8th Annual Conference on Innovation and Technology in Computer Science Education*, pp.60–64, Association for Computing Machinery (2003).
[37] 電気通信大学共通教育部情報部会：基礎プログラミングおよび演習 2017（オンライン），入手先 ⟨https://joho.g-edu.uec.ac.jp/joho/fp2017/fp2017-180203.pdf⟩（参照 2021-01-19）.
[38] 独立行政法人情報処理推進機構 IT 人材育成本部 編：IT 人材白書 2013，p.150, 独立行政法人情報処理推進機構 (2013).
[39] 駒谷昇一：PBL は教育にどのようなインパクトがあるのか，情報教育シンポジウム 2009 論文集，No.6, pp.131–138 (2009).
[40] 松澤芳昭，大岩　元：産学協同の Project-based Learning によるソフトウェア技術者教育の試みと成果，情報処理学会論文誌，Vol.48, No.8, pp.2767–2780 (2007).
[41] 井上　明，金田重郎：実システム開発を通じた社会連携型 PBL の提案と評価，情報処理学会論文誌，Vol.49, No.2, pp.930–943 (2008).
[42] 独立行政法人情報処理推進機構 IT 人材育成本部イノベーション人材センター：実践的講座構築ガイド 産学連携教育の自律的展開を進めるために 第 3 部評価基準編，pp.3–14，独立行政法人情報処理推進機構 (2013).
[43] 経済産業省：社会人基礎力（オンライン），入手先 ⟨https://www.meti.go.jp/policy/kisoryoku/index.html⟩（参照 2021-01-19）.
[44] Ritchhart, R., Church, M. and Morrison, K.: Making Thinking Visible: How to Promote Engagement, Understanding, and Independence for All Learners, Jossey-Bass (2011). 黒上晴夫，小島亜華里（訳）：子どもの思考が見える 21 のルーチン：アクティブな学びを作る，北大路書房 (2015).
[45] 小林康夫：岩波講座現代の教育第 10 巻 変貌する大学教育 大学教育での意味を再考する—大学で何を学ぶか，pp.315–330，岩波書店 (1998).
[46] 文部科学省：文部科学省教育課程企画特別部会論点整理 補足資料 (5)（オンライン），入手先 ⟨http://www.mext.go.jp/component/b_menu/shingi/toushin/__icsFiles/afieldfile/2015/09/24/1361110_2_5.pdf⟩（参照 2018-03-05）.
[47] 文部科学省：高等学校学習指導要領（平成 30 年告示）解説（情報編），入手先 ⟨https://www.mext.go.jp/content/1407073_11_1_2.pdf⟩（参照 2020-12-10）.
[48] 小川　南，平ひかり，内田奈津子：プログラミング入門をプロジェクトでやってみた—Ruby で取り組むプログラミング実践，Ruby World Conference 2018，入手先 ⟨https://2018.rubyworld-conf.org/⟩（参照 2020-09-28）.
[49] 内田奈津子，伊地知咲希，永井絵梨奈，堀池悠那：Society 5.0 with Ruby—社会で役立つ人材育成の鍵，Ruby World Confernce 2020，入手先 ⟨https://2020.rubyworld-conf.org/ja/program/session-4/⟩（参照 2020-12-18）.
[50] 文部科学省：学校基本調査（オンライン），入手先 ⟨https://www.mext.go.jp/b_menu/toukei/chousa01/kihon/1267995.htm⟩（参照 2020-08-28）.
[51] 首相官邸：未来投資会議 林文部科学大臣提出資料（2018 年 5 月 17 日）（オンライン），入手先 ⟨http://www.kantei.go.jp/jp/singi/keizaisaisei/miraitoshikaigi/dai16/siryou6.pdf⟩（参照 2020-08-23）.

付　　録

A.1 プログラミングパートの演習問題

1〜5 回のプログラミングパートで用いた演習問題について，電気通信大学「基礎プログラミングおよび演習 2017」[37] から利用した演習問題の対応と，追加した問題を示す．

第 1 回目

演習 1〜3：P.7 演習 1〜3

第 2 回目

演習 1 P.21：演習 1

演習 2：独自問題

© 2021 Information Processing Society of Japan

図 A·1 作成したネイルシールの一部
Fig. A·1 Outcome: Nail-Sticker.

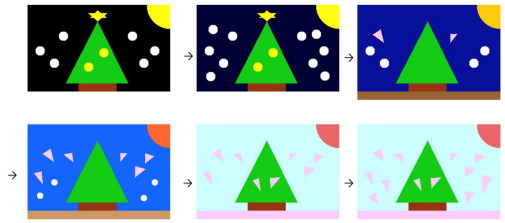


図 A·2 作成した gif アニメーションの一部
Fig. A·2 Outcome: Part of Animation Data.

演習 1 の問題のプログラムを打ち込んで動かせ．動いたら「誤差を指定する版」でも途中経過と回数が表示させるように直して動かしてみよ．

演習 3：P.27 演習 4
演習 4〜6：P.28 演習 5〜7

第 3 回目

演習 1〜4：P.38 演習 1〜4
演習 5：P.40 演習 5
演習 6〜7：P.41 演習 6〜7

第 4 回目

演習 1〜2：P.53 演習 1〜2

第 5 回目

演習 1：P.63 演習 1
演習 2：P.65 演習 2
演習 3〜4：P.67 演習 3〜4
演習 5：P.68 演習 5

A.2 2017 年度の成果物の一部

図 A·1 に，作成したネイルシールの作品の一部（上段は E チーム，下段は F チーム）を示す．

図 A·2 に，gif アニメーションの作品の一部（C チームの最終発表の発表資料より）を示す．

A.3 2019 年度の事例：役割分担とプログラム

プログラミングパートの演習問題を見直し，1 回目から図形生成の教材を用いる方法に変更した．プロジェクトパートの課題は，パーツの細かいネイルシールのデザインから，変更を行った．2019 年度は，プロジェクトパートの課題を 4 分割したステッカーへの変更したところ，学生が 1 人 1 メソッドの作成を行い，1 つの絵に統合するという目的に叶ったプロジェクトができるようになった．図 A·3 に，2019 年度の分担とメソッド化と統合の事例を示す．

図 A·3 2019 年の作品と分担の事例
Fig. A·3 Role Allocation & Team Topics, 2019.

図 A·3 は，その 1 つである．サークル，空，海，船をそれぞれ分担し，分担に対応したパーツのコードを書き，最後に 4 つのパーツを統合している．改善に至ったポイントは，プログラミングパートの演習問題の修正とプロジェクトパートの課題の設定によるところであると考える．

© 2021 Information Processing Society of Japan

内田 奈津子 （正会員）

1970 年生．1993 年津田塾大学学芸学部数学科卒業．1996 年山梨大学大学院博士前期課程修了．修士（工学）．1996 年フェリス女学院大学教育研究用情報センター助手，現在，同大情報センター講師．電気通信大学大学院博士後期課程情報理工学研究科情報・ネットワーク工学専攻在学．教育システムの構築・運用に従事し，情報教育に興味を持つ．

久野 靖 （正会員）

1986 年東京工業大学理工学研究科情報科学専攻博士後期課程単位取得退学．同年筑波大学講師．同大学助教授，教授を経て現在，電気通信大学教授．筑波大学名誉教授．理学博士（1988 年，東京工業大学）．

中山 泰一 （正会員）

1965 年生．1988 年東京大学工学部計数工学科卒業．1993 年同大学院工学系研究科情報工学専攻博士課程修了．博士（工学）．同年電気通信大学情報工学科助手．現在，同大学院情報理工学研究科教授．国立情報学研究所客員教授．オペレーティング・システム，並列処理，情報教育等に興味を持つ．本会では，論文誌ジャーナル編集委員会編集長，初等中等教育委員会副委員長等を務める．現在，教育担当理事．2016 年度山下記念研究賞，2017 年度科学技術分野の文部科学大臣表彰科学技術賞．日本学術会議特任連携会員．電子情報通信学会，IEEE-CS 等の会員．本会シニア会員．

© 2021 Information Processing Society of Japan