情報処理学会論文誌 Vol. 49 No. 10 3409–3427 (Oct. 2008)

アルゴリズム構築能力育成の導入教育：実作業による概念理解に基づくアルゴリズム構築体験とその効果

杉浦 学[†1] 松澤芳昭[†2]
岡田 健[†1] 大岩元[†2]

一般情報教育の一環として実施するプログラミング教育においては「与えられた課題を理解・分析して，それを手順に分解・詳細化し，プログラムとして記述可能なアルゴリズムを組み立てる」という「アルゴリズム構築能力」をどのような方法で育成するかが課題となる．本論文では，アルゴリズムの概念を体感的に理解させるための導入教育を行うことにより，効果的にアルゴリズム構築能力を育成するという教育方法を提案し，その効果を検証した．提案方法における導入教育では，学習者に手作業でアルゴリズムを実行・理解させ，その体験を忠実にプログラミングさせる．その際，使用した教具を抽象データ型として用意し，手作業によるアルゴリズムをそのまま実行可能なプログラムとして記述できるようにした．これにより，学習者が理解したアルゴリズムを自力でプログラミングすることが可能になった．高校と大学で実験授業を行ったところ，演習に高い意欲を持って取り組ませることができ，受講者全員が挿入法による整列プログラムを作成できた．演習の実施後に受講者が取り組んだ自由作品制作の結果から，受講者にアルゴリズム構築能力が身についていることも確認できた．さらに，1）演習によって，人間とコンピュータそれぞれの情報処理の特性と関連付けたアルゴリズム概念の理解が得られること，2）アルゴリズム概念の理解が，アルゴリズムの構築における分析や詳細化の段階の重要性を認識させること，3）プログラミング言語として，Squeak eToys を拡張した「ことだま on Squeak」を用いたことが，演習を円滑に実施できた要因の 1 つであること，が分かった．

Introductory Education for Algorithm Construction: Understanding Concepts of Algorithm through Unplugged Work and Its Effects

Manabu Sugiura,[†1] Yoshiaki Matsuzawa,[†2]
Ken Okada[†1] and Hajime Ohiwa[†2]

It is an important issue of informatics education for non-major students to develop “algorithm construction ability”. We regard this ability as that of understanding and analyzing a given problem and constructing an algorithm which can be descried as a procedure in sufficient and complete detail. In this paper, we proposed an effective method for this issue by understanding the concept of the algorithm through real “unplugged” procedure. In this method, experiences of both “unplugged” work and those with computers of the same algorithm are given to learners. In order to facilitate for learners to develop programs, we use an abstract data type for describing objects used in “unplugged” work so as to reduce the difference between the algorithm description and the program. Those results of our experiments both in a high school and in a university showed that all students had completed developing and programming insertion sort algorithm by themselves. Further improvement in the students’ ability to construct algorithms was observed in the result of the students’ works. In addition, we also have found that 1) learners understand the algorithm concept with regards to the characteristics of the processing of information by human and by computer, 2) learners understand the importance of the stages of analyzing and refining of the algorithm construction, 3) “Kotodama on Squeak (extended edition of Squeak eToys)” contribute to the success of the practices.

1. はじめに

高度 ICT 社会の到来により，情報分野を専門とする者に限らず，高校や大学における一般情報教育の一環として，プログラミング教育を実施する必要性が指摘されている[1)]．

我々は，一般情報教育で実施するプログラミング教育における目標を「与えられた課題を理解・分析して，それを手順に分解・詳細化し，プログラムとして記述可能なアルゴリズムを組み立てる」という「アルゴリズム構築能力」の育成とした．これは，UNESCO による中等教育における情報教育のカリキュラム提案[2)] に従ったものである．UNESCO の提案によると，プログラムの入門教育（ICT SPECIALIZATION — Specialization Preparation Module）における目標は「現実の問題をアルゴリズミックに解決できるようになること」である．一般情報教育で扱うべきと考えられる入門教育の導入部分（Unit SP1 — Introduction to Programming）の内容は「技術的なものでなく，“自分ができる何らかの仕事” を他人や機械が実行可能なように十分詳細に完全にその手続きを記述することである」とされてい

†1 慶應義塾大学政策・メディア研究科
Graduate School of Media and Governance, Keio University
†2 慶應義塾大学環境情報学部
Faculty of Environmental Information, Keio University

る．我々が実践すべきと考えている教育内容は，完成されたプログラムを学習者に与え，そのプログラムを実行し，その動作結果を確認するだけの「プログラミングの体験」ではない．授業で扱うことのできる課題は簡単であっても，学習者自らがアルゴリズムを組み立てることを実践させ，またそれを実行可能なプログラミング言語で記述し，その正しさを検証することまでを含めた教育を行うべきであると考えている．

このような教育を実施するうえで問題となるのは，教育の方法である．楠らは[3]，プログラミングの初学者が，アルゴリズムの学習と同時にプログラミング言語を学習することによって，アルゴリズムを十分に学習するまでの負担が大きくなり，プログラミング言語の学習（特に文法の習得）にすべての注意を向けがちとなることを指摘し，学習者が作成したアルゴリズムが正しいかどうかを十分注意させるのが困難であることを問題点としている．専門教育では，プログラミング言語を習得した後に時間をかけてアルゴリズムに関する学習をすることも可能である．しかしながら，一般情報教育の一環としてプログラミング教育を実施する場合，専門教育と比較して，教育全体にかけられる時間は短くせざるをえない．したがって，楠らの指摘した「アルゴリズムを十分に学習できない」という問題はより顕著に現れることになる．

本論文では，この問題を解決するための教育方法を提案し，その効果を実験授業によって検証した結果について述べる．我々は，アルゴリズム構築能力の育成を円滑に実施するためには，講義による説明ではなく，アルゴリズムの概念を体感的に理解させることが重要であると考えた．そこで，制御構造と変数が理解できるようになった学習者に，アルゴリズムの概念を理解させるための導入教育を実施することにした．具体的には，まず題材となるアルゴリズムを手作業で実行，理解させ，次にその体験を忠実にプログラミングさせる方法を採用した．こうした導入教育を実施した後に，アルゴリズム構築能力を育成するための演習を実施するという方法を試みた．

まず，2 章で提案方法に関連する先行研究について述べ，3 章で我々が提案する教育方法の内容を説明する．4 章で提案方法を採用した具体的な演習について述べる．5 章では，4 章で説明した演習を高校と大学で実施した実験授業の結果について述べる．6 章では，その結果についての考察を行い，7 章で総括を行う．

2. 先 行 研 究

我々が提案する教育方法は，先行研究として個別に提案されている手法を効果的に組み合わせ，手法どうしの連携を図ることによって実現したものである．本章ではそれらの先行研究について整理する．プログラミングの初学者，もしくはプログラミングの未経験者を対象としたプログラミング教育に関する先行研究のうち「手作業を取り入れたアルゴリズム教育」と「プログラミング環境による学習支援」について取り上げ，それぞれが学習者にとってどのような効果があるかを述べる．

2.1　手作業を取り入れたアルゴリズム教育

本節では，プログラミング言語を使わず，学習者が手で扱うことができる教具を利用してアルゴリズムの教育を行う研究について述べる．

コンピュータを使わず，情報科学を体験的に学習することができる教育方法として Computer Science Unplugged[4]（以下 Unplugged と略す）がある．Unplugged には探索や整列といったアルゴリズムを題材とした教材も用意されており，探索や整列アルゴリズムの考え方や効率に関する学習を行うことができる．同様の方法が Holmes ら[5] によって提案されており，プログラミング教育の準備段階として有効であることが示されている．

前述した体験的な学習とプログラミングを組み合わせた試みとして，阿部による提案[6] がある．阿部は，アルゴリズムを理解する教具として，学習者が自らの手で扱えるカードを用いる方法を提案している．学習者がカードを使ってアルゴリズムを手作業で実行するという段階を設け，その後に探索や整列のプログラミングの学習を行う方法が述べられている．

これらの研究における体験的な学習は，プログラミング言語を使ったアルゴリズムの学習の前段階で実施することが想定されている．プログラミングの初学者は複雑なアルゴリズムをプログラムから読み解く能力がないため，プログラムを読解し，アルゴリズムを理解することは困難である．教具を使って手作業でアルゴリズムを実行することで，そのアルゴリズムを体感的に理解することができる．また，複雑なプログラムを提示され，それを読み解くことに意欲を失うことがないと考えられる．

2.2　プログラミング環境による学習支援

本節では，プログラミング環境による学習支援に関する先行研究について述べる．

楠らは[3]，プログラミングの学習の前半から中盤にかけて，アルゴリズムに関する学習のみを構造化チャート（PAD）を使ったツールを用いて実施し，プログラミング言語に関する学習は授業後半の 3 割弱の時間をかける方法を提案，実践した．斐品ら[7] も，一般情報教育の一環として実施するプログラミング教育に，PAD を使った学習ツールを導入することを提案している．斐品らは高校や大学での一般情報教育における実践を想定しているため，楠らと異なり，具体的なプログラミング言語を学習させることは実施していない．

加藤らは[8] 中学生以上を対象に，ゲームを題材としたプログラミング教育を実践してい

る．加藤らは Logo をベースにしたプログラミング環境を用意し，中学生であってもアルゴリズムの構築が可能な学習環境を用意している．

初学者用のプログラミング言語としては，長ら[9] の Nigari，西田ら[10] の PEN，兼宗ら[11] のドリトルがある．これらの研究もプログラミングの初学者がプログラミング言語の習得にかかる負担を減らすために実施されている．

本節で述べた研究は，プログラミング言語を記述するための学習コストを減らし，プログラミング教育を円滑に実施できるような環境を用意する方法を採用している．プログラミングの初学者がアルゴリズム構築能力を身につけるためには，本節で述べたような，学習者のプログラミングを支援するための環境を用意したうえで教育を実施することが有効である．

3. 教育方法の提案

本章では，2 章で述べた先行研究をふまえ，我々が提案する教育方法について説明する．

3.1 提案方法の概要

2.1 節と 2.2 節で述べた先行研究の要点をまとめると，1）アルゴリズムを理解するために，プログラミング言語の読解ではなく，手作業からアルゴリズムを理解することが有効である，2）アルゴリズムを組み立てる能力を育成するためには，プログラミング言語を記述するための学習コストを減らし，アルゴリズムの構築に注力できるような環境を用意することが有効である，といえる．

我々は，1）と 2）の手法を組み合わせ，それらの連携を図る工夫を行うことで，プログラミングの初学者（制御構造と変数の学習が終わった段階の学習者）に，体感的に「アルゴリズム概念」を理解させることができると考えた．まず，学習者にアルゴリズムを手作業で実行させ，理解させる．次に，そのアルゴリズムをそのままプログラミングできるように工夫したプログラミング環境を用意し，手作業のアルゴリズムを忠実にプログラミングさせる．このような方法で導入教育を実施した後に，アルゴリズム構築能力の育成を行うことにより，効果的に教育が実施できるという仮説を立てた（**図 1**）．

ここでいう「アルゴリズム概念」とは，a）ある目的を達成するための仕事は単純な手順の積み重ねによって構成されていること，b）それぞれの手順は解釈に曖昧さがないように詳細化されていること，を指す．従来の教育では，プログラミングの初学者にも分かりやすい身近な例（料理の手順，学校へ行くまでの手順など）を用いて講義を行うことが多かった．しかし，これでは学習者の理解は知識的なものにどどまり，アルゴリズムの概念とプログラミングとの関連が理解できない．アルゴリズム概念をプログラミングと関連付け，体感的に理解させることにより，効果的に「アルゴリズム構築能力」を育成できると考えた．

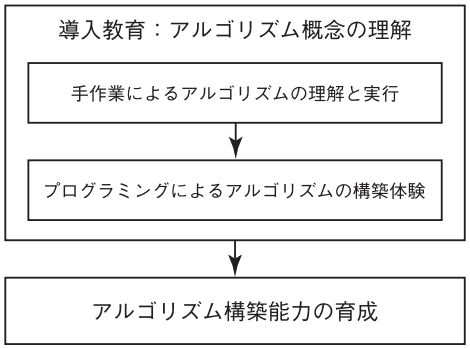

図 1　提案方法の教授方略
Fig. 1　Education method.

a）に関しては，複雑なアルゴリズムを作った経験のない学習者は，単純な手順が積み重なって，目的の仕事が達成されるという実感を持つことが難しい．たくさんの単純な処理を組み合わせ，一定以上の複雑な処理を構築する体験が必要になる．b）に関しては，曖昧な仕事の仕様を分析し，それを詳細化してプログラミングした経験がないと理解は難しい．現実世界で人間が行っている手順をコンピュータ上で再現するという体験を持たせることが効果的である．

学習者に複雑なアルゴリズムを理解させるために，2.1 節で述べた，手作業でアルゴリズムを実行，理解させるという方法を採用した．また，手作業で実行，理解したアルゴリズムをプログラミングさせるためには，2.2 節で述べたような，学習者がアルゴリズムの構築に注力できるような環境を用意する必要がある．我々は，こうした環境として，処理内容を日本語で記述できる教育用のプログラミング言語を用いるとともに，学習者が利用するプログラミング環境に手作業のアルゴリズムをプログラミングしやすいような工夫を行った．この工夫については 3.3 節で詳しく述べる．

次節から，図 1 に示した教育内容の詳細を順番に説明する．

3.2 手作業によるアルゴリズムの理解と実行

「手作業によるアルゴリズムの理解と実行」では，指導者が提示する問題を手作業で解決することを通じて，問題そのもの，またそれを解決するためのアルゴリズムを理解させる．

指導者は題材となる現実世界の問題を手作業によって解決するアルゴリズム記述を学習者

に提示する．提示するアルゴリズムはプログラミングが可能な程度に詳細化しておく必要がある．手作業で実行可能なアルゴリズムであれば，提示の形式は任意[⋆1]である．学習者は手作業で扱うことができる教具（たとえば，数字が記載されたカードなど）を使ってアルゴリズムを実行し，その結果を得る．

題材となる問題は，学習者が体を動かして解決でき，興味を持って取り組めるものを選択する．また，アルゴリズムは一定以上の複雑さを持つように，かつ学習者が手作業を通じて理解できる範囲内で設計する．我々は「数字の書かれたカードを昇順に整列させる（整列問題）」を選んで教育を行った．他の題材の候補として「カード束の中から指定された数字が記述されたものを探索する（探索問題）」などがある．

3.3　プログラミングによるアルゴリズムの構築体験

「プログラミングによるアルゴリズムの構築体験」では，学習者が手作業で実行したアルゴリズムをプログラミング言語で記述させ，手作業で解決したものと同一の問題をコンピュータを使って解決する体験を持たせる．

この教育で想定しているのはプログラミングの初学者であるから，手作業で実行したアルゴリズムをプログラミングするための方法に工夫が必要である．2.1 節で述べた阿部の提案では，手作業で理解したアルゴリズムとそのプログラミングを連携させる手法は提案されておらず「手作業によるアルゴリズムに関する学習」と「プログラミングによるアルゴリズムの学習」は分割されている．実際，阿部の論文には「カードの並べ替えで行った作業手順をそのままコンピュータアルゴリズムとしてプログラミングするのは，かなり能力の高い学習者でないと手に負えないはずである」と述べられており，手作業とプログラミングの連携については言及がない．

整列アルゴリズムを題材とした場合，手作業でカードを整列させるときには，カードをいくつかの束に分割して行う方が自然である．しかし，コンピュータで行う整列アルゴリズムは 1 つの配列に整数値を格納したデータ構造を用いることが効率的であり，これで教育を行うことが一般的である．このように，手作業で整列を行う場合と，プログラムで整列を行う場合のデータ構造が異なると，両者は異なったアルゴリズムを持つことになる．また，プログラムに配列を利用した場合，配列のインデックスを操作しなければならず，手作業のアルゴリズムにはない処理をプログラミングする必要が生じる．

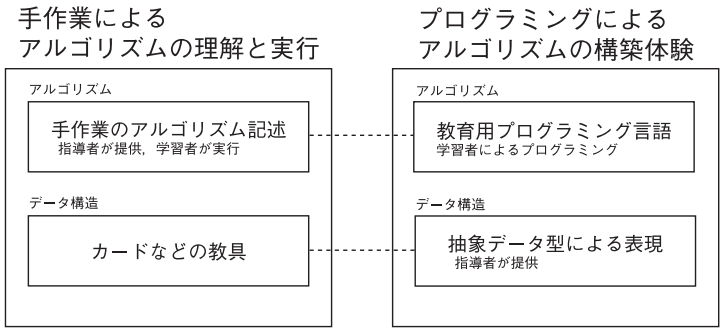

図 2　導入教育における手作業とプログラミングの関係
Fig. 2　Relation between programming environment and “unplugged” work.

手作業のアルゴリズムをプログラミングするために必要な技術や知識を最小限におさえることができれば，学習者は手作業のアルゴリズムを正確に記述する作業だけに注力できる．そこで，我々は学習者が利用するプログラミング環境に，以下の 2 点の工夫を実施（**図 2**）することにより，学習者が手作業のアルゴリズムを自力でプログラミングできるようにした．

(1)　現実世界の教具を抽象データ型を使って表現する．
(2)　手作業によるアルゴリズム記述とプログラミング言語による処理の表現の乖離をできる限り少なくする．

(1) に関しては，カードを重ねた束であれば，それをスタック[⋆2]を用いて表現する．また，手作業で使用したカード束が複数あれば，その数と同数のスタックを用意する．現実世界の教具を抽象データ型を用いて表現することによって，手作業のアルゴリズム記述の制御構造や操作の内容を変更することなくプログラミングが可能となる．データ構造の定義部分のプログラムは指導者が提供する．

(2) に関しては，手作業のアルゴリズム記述をプログラミング言語の表現に翻訳することは学習者にとって負担になる．これを解決するために，処理内容を日本語で記述できる教育用プログラミング言語を利用する．

3.4　アルゴリズム構築能力の育成

3.2 節と 3.3 節で述べた導入教育（「手作業によるアルゴリズムの理解と実行」と「プロ

⋆1 擬似言語，フローチャート，構造化チャートによる方法が考えられる．提案方法を実施する前の学習者の前提知識によって最適なものを選択すればよい．我々の実験授業（5 章）では，1）記法が単純でチャートの読み書きに必要な前提知識が少なく済む，2）擬似言語に比較し，繰返しや分岐の処理の流れが初学者に理解しやすい，という理由からフローチャートを採用した．

⋆2 スタックを使う場合は，その操作を記述するための最小限の構文だけは教育する必要がある．

グラミングによるアルゴリズムの構築体験」）の後に，適切な難易度のアルゴリズムを構築する訓練を行う．これには課題を提示し，それを解決させるためのアルゴリズムを設計し，プログラミングさせる方式をとる．

学習者が新たに習得すべき知識を少なくし，段階的に教育を実施するため，導入教育で扱った課題を題材として，異なるアルゴリズムを学習者に構築させる演習を実施する．こうした方法で演習を実施することにより，学習者はアルゴリズムの構築だけに注力できる．

4. 整列・探索アルゴリズム演習

本章では，3 章で述べた教育方法を採用し，整列と探索アルゴリズムを題材とした演習について述べる．4.1 節で演習に利用するプログラミング言語「ことだま on Squeak」について説明し，4.2 節で 4 つの演習の内容と手順について説明する．

4.1 ことだま on Squeak

演習で利用するプログラミング言語として Squeak eToys [12] の日本語版を拡張した「ことだま on Squeak」[13]（**図 3**）を使用した．Squeak eToys は小学生が扱えるように設計された，ビジュアルプログラミング環境である．「ことだま on Squeak」の設計指針や詳しい拡張内容については文献 14)，15) を参照されたい．

4.1.1 特　　徴

Squeak eToys から引き継いだものを含めて，「ことだま on Squeak」の特徴は以下のとおりである．

- 「ことだま on Squeak」は Squeak eToys のタイルスクリプティングシステム[⋆1]を引き継いでいる．プログラムの文法を覚える必要がなく，構文エラーが発生しない．
- 「ことだま on Squeak」では，Squeak eToys のタイルを改良し，タイルに記述された処理内容が理解可能な日本語に変更[⋆2]されている．
- Squeak eToys では，プログラムで操作可能なオブジェクトが画面上につねに表示されている．オブジェクトの移動等のプログラムの実行結果は即座に画面に表示される．学習者はアルゴリズムを意図したとおりにプログラミングできたかを視覚的に確認することができる．

こうした特徴を持つ「ことだま on Squeak」は 3.3 節で述べた，手作業のアルゴリズム記述をプログラミングするための環境として適している．他のプログラミング言語を演習に利用することの可能性に関しては，実験授業の結果をふまえて，6.2 節で考察する．

⋆1 マウスを使って，タイルを組み合わせてプログラムを作るシステム．
⋆2 タイルの語順変更，「を」「に」といった助詞の追加，「ドット」といった助数詞の追加，条件分岐の表現の改善が行われている．詳しくは文献 14)，15) を参照されたい．

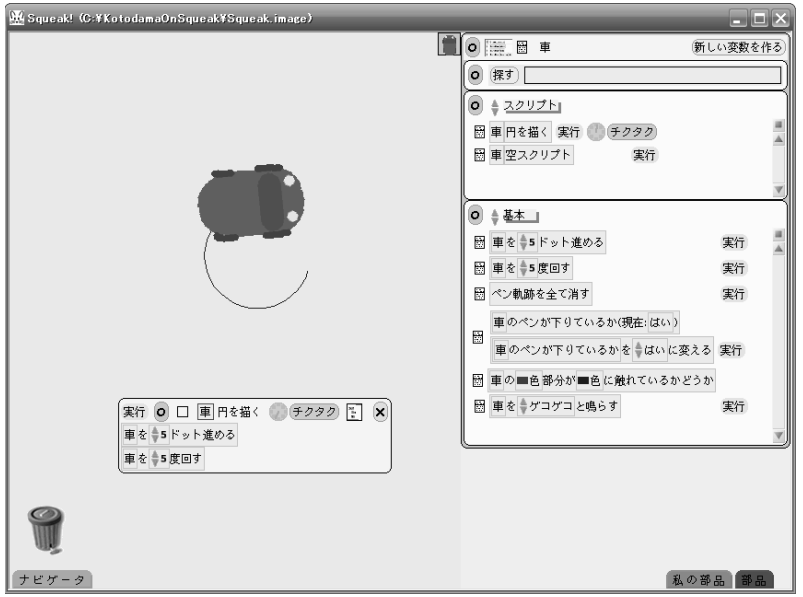

図 3　「ことだま on Squeak」の画面イメージ
Fig. 3　Screenshot of “Kotodama on Squeak”.

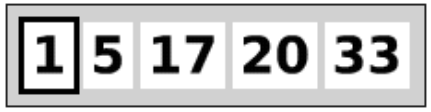

図 4　「入れ物」オブジェクト
Fig. 4　Holder object.

4.1.2 入れ物オブジェクト

Squeak eToys では「入れ物」（**図 4**）という，可変長のコレクションオブジェクトが用意されている．これを手作業の演習で利用した教具を表現する抽象データ型として利用する（3.3 節参照）．「入れ物」に関する操作を記述したタイルの一覧を**図 5** に示す．

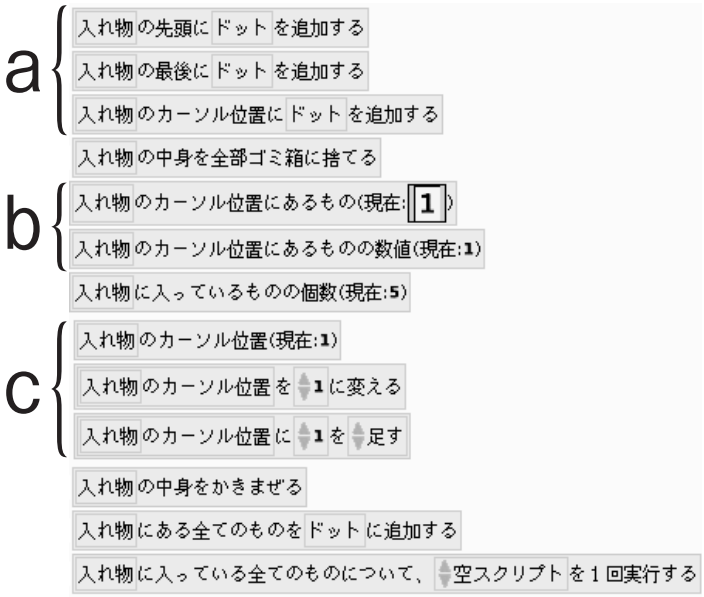

図 5　「ことだま on Squeak」における「入れ物」の操作タイル
Fig. 5　Script tiles of holder object.

「入れ物」は格納されているオブジェクトを参照するための「カーソル」を持っている．カーソルの位置は自由に移動させることができ（図 5 中の c），カーソル位置にあるオブジェクトの情報を参照することができる（図 5 中の b）．カーソル位置にあるオブジェクトには太枠が表示される（図 4 では先頭の位置にカーソルがある）．

「入れ物」の先頭，最後，カーソルが参照している位置にオブジェクトの挿入が可能である（図 5 中の a）．図 5 中の「ドット」の部分に挿入するオブジェクトを指定すればよく，挿入によってカーソル位置の変更はない[⋆1]．

また，図 5 中の「ドット」の部分には他の入れ物のカーソル位置にあるオブジェクトも設定することができる．たとえば，**図 6** に示したプログラムを実行すると「入れ物 B のカーソル位置にあるオブジェクト」は入れ物 A へ移動する．

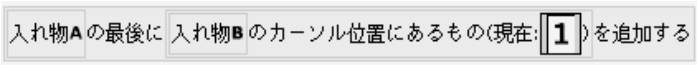

図 6　挿入にともなってオブジェクトが「入れ物」間を移動するプログラム
Fig. 6　Example program of insert object with movement of object.

表 1　整列・探索アルゴリズム演習の一覧
Table 1　List of exercises.

教育内容（3 章参照）	演習の名称	演習の概要
手作業によるアルゴリズムの理解と実行	手作業による最小値選択法	指導者から提示されたフローチャートを参考に選択法のアルゴリズムを手作業で実行し，理解する
プログラミングによるアルゴリズムの構築体験	最小値選択法の実装	「手作業による最小値選択法」で利用したフローチャートを参考に「ことだま on Squeak」を使って選択法のアルゴリズムをプログラミングする
アルゴリズム構築能力の育成	挿入法の構築	挿入法のアルゴリズムを詳細化し，「ことだま on Squeak」を使ってプログラミングする
	探索アルゴリズムの応用	指導者から提示された「絵辞書」の探索アルゴリズムを理解し，それを応用して簡単なゲームを作成する

4.2　演習の内容と手順

本節では，演習の内容と手順をその実施の順番に従って述べる．4 つの演習の一覧を**表 1** に示した．

3.3 節の図 2 に対応した「手作業とプログラミングの関係」を**図 7** に示す．手作業で扱ったカードを再現したカードオブジェクト[⋆2]，カード束と同数の「入れ物」をスタックとして用意した．

4.2.1　「手作業による最小値選択法」演習

「手作業による最小値選択法」は選択法[⋆3]によるカードの並べ替えを手作業で実施する．

演習の手順は以下のとおりである．

(1)　学習者に 2 人でペアを組ませ，計測係と並べ替え係を担当させる．

⋆1 これは「ことだま on Squeak」で利用可能な「入れ物」の独自の仕様である．Squeak eToys の「入れ物」はオブジェクトの挿入を行うと，挿入されたオブジェクトにカーソルの位置が自動的に移動する仕様であった．多くの場合，オブジェクトを追加した後にカーソルの位置を修正する必要があり，プログラムが複雑化する原因であった．この「入れ物」は「ことだま on Squeak」とともに Web [13] からダウンロード可能である．

⋆2「ことだま on Squeak」におけるテキスト形式のオブジェクトに数値を入力したものを利用した．これによって，本論文で述べた演習では，手作業で実施したカードの整列問題と同じ状況を再現できた．一般に，全順序集合の要素を属性として参照可能なオブジェクトを入れ物に格納すれば，整列をさせることが可能である．

⋆3 学習者にアルゴリズムが分かりやすいように「最小値」という言葉を補った．

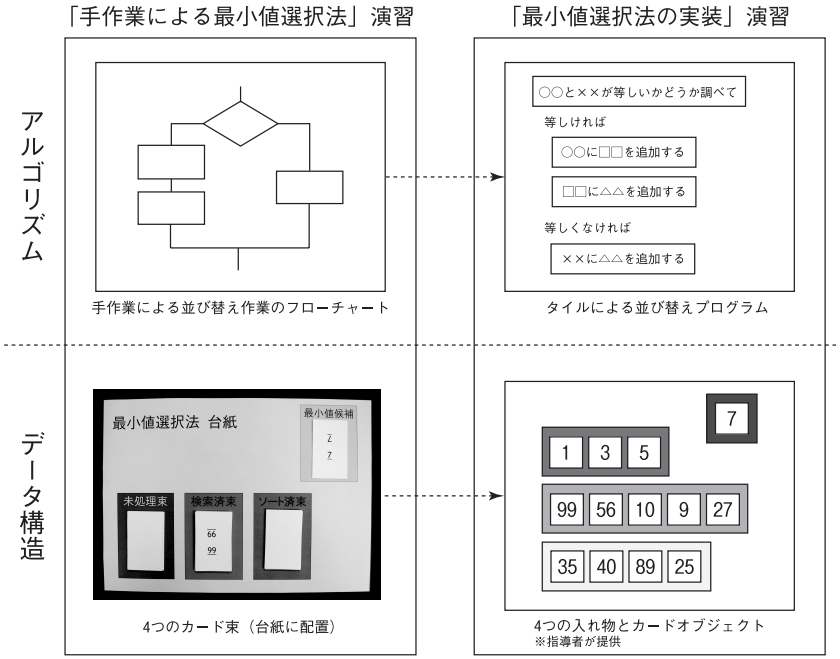

図 7　「手作業による最小値選択法」と「最小値選択法の実装」の関係
Fig. 7　Relation between “Unplugged selection sort” and “Selection sort implementation”.

(2)　各ペアに 30 枚程度のカード（名刺大のカードの片面にランダムな番号を書いたもの，**図 8**），並べ替えの際にカードを置くための台紙（**図 9**），選択法によるカードの並べ替えを手作業で実施するためのアルゴリズムを記述したフローチャート（**図 10**）を配布する．

(3)　並べ替え係は，指示に従って台紙の上にカードを並べる．台紙には未処理束，検索済束，ソート済束，最小値候補があり．シャッフルしたカードを未処理束に裏向きに置く．最小値候補には任意の 1 枚のカードを置く．

(4)　並べ替え係はフローチャートの指示どおりに並べ替えを行う．カードはソート済束に整列されて移動する．計測係はフローチャートの記述に沿って並べ替えが行われているかどうか確認しながら，1 枚のカードを処理するためにかかった時間を記録する．

(5)　並べ替えが終わったら，並べ替えるカードの枚数と並べ替えに必要な時間の関係について，(4) の記録からグラフを作成させ，考察を実施させる．

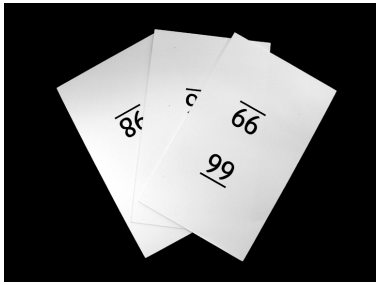

図 8　演習で使用するカード
Fig. 8　Card used by “unplugged” work.

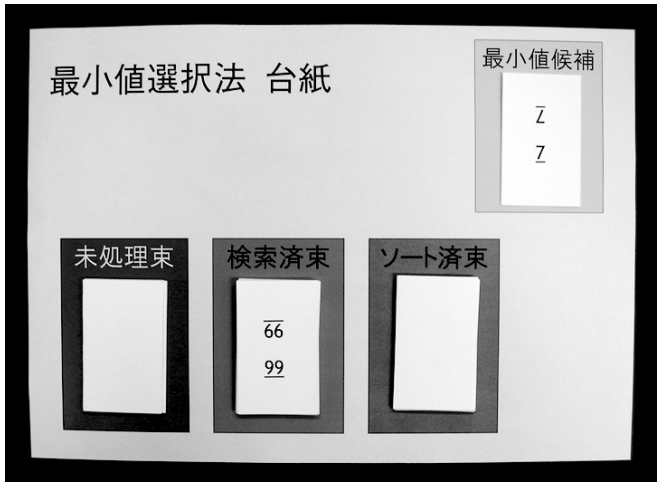

図 9　カード束が配置された台紙
Fig. 9　Workspace for sorting exercise.

受講者にペアを組ませる理由の 1 つ目は，並べ替えと時間の計測を 1 人で同時に行うことが難しいためである．計測係と並べ替え係を 2 人で分担することにより，並べ替えの作業を中断することなく，時間の計測を正確に行うことができる．2 つ目の理由は，指導者から提示されたフローチャート（図 10）の手順に忠実に並べ替えを実施させるためである．計

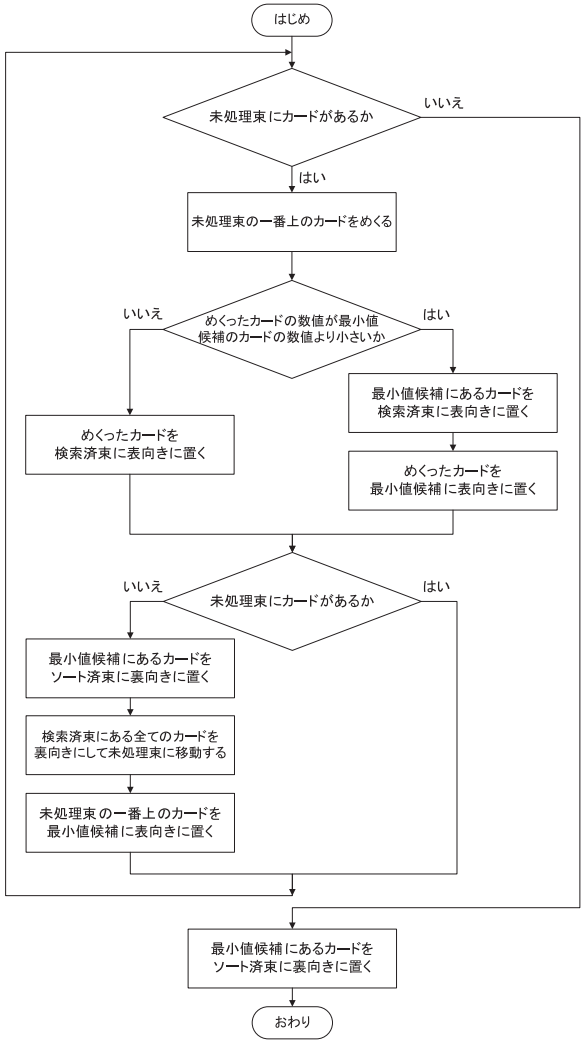

図 10　学習者に提示するアルゴリズム記述
Fig. 10　Algorithm description (selection sort).

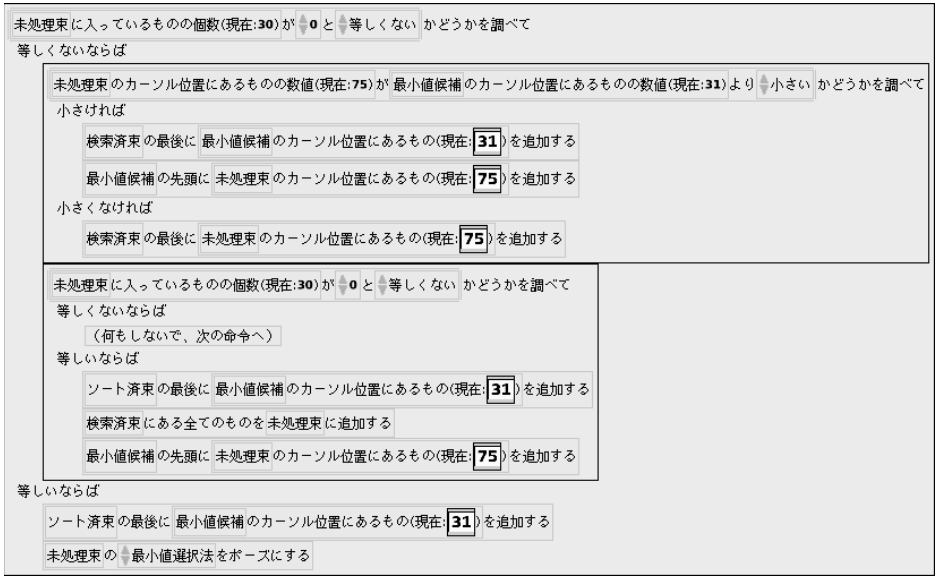

図 11　選択法の整列プログラム
Fig. 11　Selection sort program.

測係は並べ替え係の作業手順が適切かどうかを確認し，間違いがあれば指摘するように指導する．これにより，指導者が受講者全員の様子を見て回る手間を省くことができる．

4.2.2 「最小値選択法の実装」演習

「最小値選択法の実装」は前項で述べた「手作業による最小値選択法」で手作業によって実行したアルゴリズムを「ことだま on Squeak」でプログラミングする演習である．

学習者がプログラミングする選択法のプログラムを**図 11** に示す．このプログラムが繰り返し実行されることによって，未処理束に入っているカードが昇順に整列されてソート済束に移動する．手作業で使ったカード束と同数，同名の入れ物オブジェクトを用意することで（図 7），図 10 に示した手作業によるアルゴリズム記述とプログラムの制御構造を同じにし，処理内容の表現の乖離を少なくすることができる．

演習の手順は以下のとおりである．

(1)　指導者が「入れ物」の操作方法に関する簡単な説明を行う．

(2)　学習者に実装のための「演習テンプレート」（**図 12**）を配布する．演習テンプレート

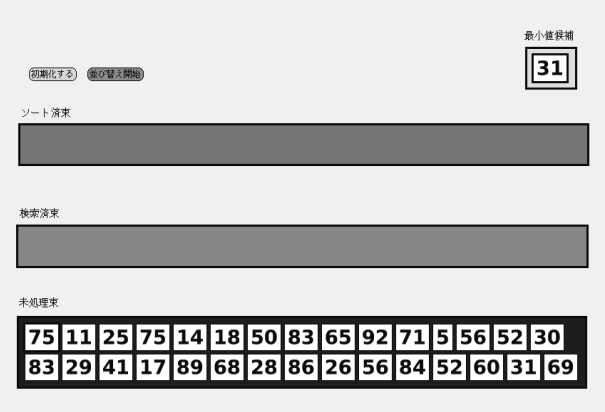

図 12　「最小値選択法の実装」演習テンプレート
Fig. 12　Programming environment of “Selection sort implementation”.

には初期化を行うためのボタン，カード束に対応した 4 つの「入れ物」とカードオブジェクトが配置されている．

(3)　学習者は手作業による選択法のアルゴリズム記述（図 10）を参考にしながらプログラミングを行う．

4.2.3 「挿入法の構築」演習

「挿入法の構築」は「挿入法によるカードの整列」という課題を理解して，コンピュータが実行可能なアルゴリズムを組み立て，それを「ことだま on Squeak」でプログラミングする演習である．

「最小値選択法の実装」では「入れ物」のカーソルを操作する処理は必要なかったが，この演習ではカーソルの操作をプログラミングさせる．

学習者がプログラミングする挿入法のプログラム例を**図 13** に示す．初期状態では，未処理束にシャッフルされたカードが格納されている．また，ソート済束には番兵値（999 のカード）となるカードが 1 枚格納されている．このプログラムが繰り返し実行されることによって未処理束からソート済束にカードが移動して，昇順に並ぶ．

演習の手順は以下のとおりである．

(1)　指導者が「入れ物」のカーソルに関する説明を行う．

(2)　完成したプログラムの動作の様子を録画した動画を観察させて解説し，学習者にアルゴリズムの概要を理解させる．

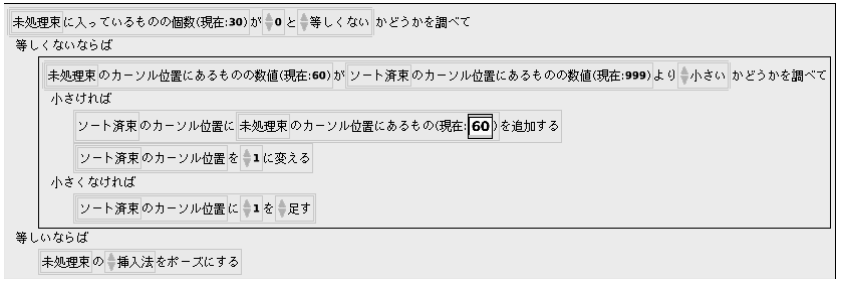

図 13　挿入法による整列プログラムの例
Fig. 13　Example of insertion sort program.

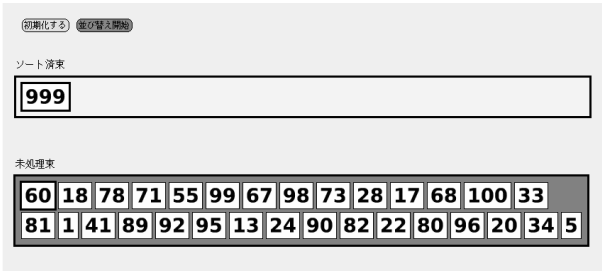

図 14　「挿入法の構築」演習テンプレート
Fig. 14　Programming environment of “Constraction of insertion sort”.

(3)　学習者は理解したアルゴリズムを詳細化して組み立て，結果をフローチャートとして記述する．

(4)　学習者に実装のための「演習テンプレート」（**図 14**）を配布する．演習テンプレートにはプログラムの初期化処理を行うためのボタンと 2 つの「入れ物」やカードオブジェクトが配置されている．

(5)　学習者は (3) で作成したフローチャートを参考にしながらプログラミングを行う．

4.2.4 「探索アルゴリズムの応用」演習

「探索アルゴリズムの応用」は，キーと値を格納した 2 つの「入れ物」を使った辞書プログラムを題材として，探索アルゴリズムの理解と応用を行う演習である．

「絵辞書」プログラムを学習者にサンプルとして提示する．絵辞書は果物の名前を入力す

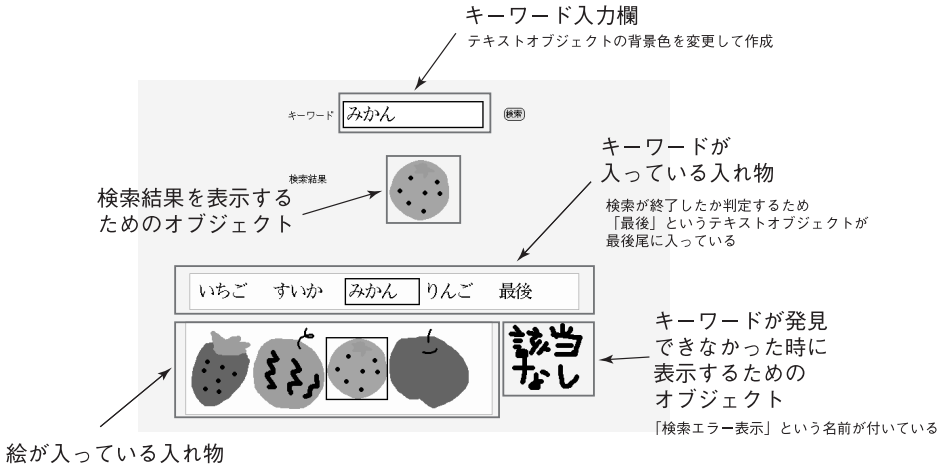

図 15　絵辞書プログラムの画面
Fig. 15　Screenshot of “Picture Dictionary”.

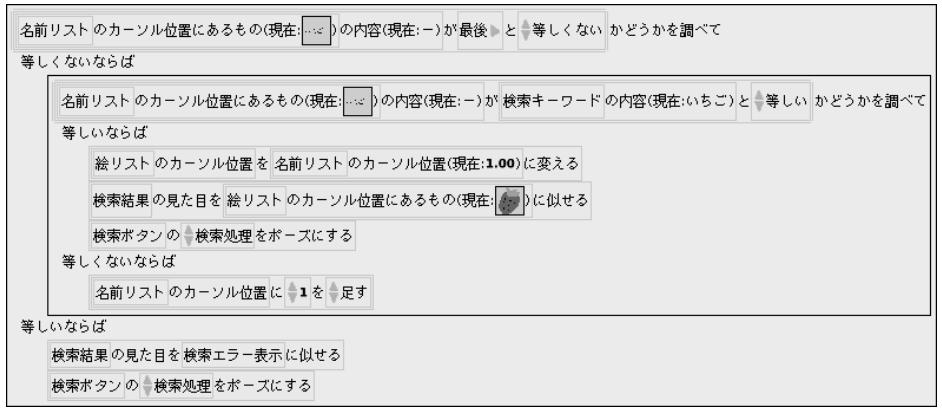

図 16　絵辞書における探索プログラム
Fig. 16　Searching program of “Picture Dictionary”.

ると，それに対応する果物の絵が表示されるというプログラムである．**図 15** に絵辞書プログラムの画面を，探索のためのプログラムを**図 16** に示す．

学習者は**表 2** に示す「タイピングゲーム」，「クイズゲーム」，「記憶君」のうちから任意の 1 つを選択し，絵辞書のプログラムを改造して作成する．

表 2　学習者が作成するゲームの概略
Table 2　Specifications of games.

タイピングゲーム	スタートボタンを押すと，絵が表示される． 絵を表現した単語を入力し，決定ボタンを押す． 単語が正しく入力されていれば，次の問題が表示される． 単語の入力が間違いであれば，決定ボタンを押しても次の問題は表示されない． 問題の出題がすべて終わると，結果（入力文字数と経過時間）を表示する．
クイズゲーム	スタートボタンを押すと，3 択の問題が表示される． 答えだと思う番号のボタンを押すと次の問題が表示される． 問題の出題がすべて終わると，結果（正解数）を表示する．
記憶君	質問を入力し，質問ボタンを押す． 質問に対する回答がすでに登録されていれば，対応する回答を表示する． 新しい質問ならば，回答を入力するためのフォームを表示する． フォームに回答を入力し，教えるボタンを押すと，質問と回答のセットが登録される．

演習の手順は以下のとおりである．

(1)　指導者が「絵辞書」の説明を行い，学習者が作成する 3 種類のゲームを実行するデモンストレーションを行う．

(2)　学習者は配布された「絵辞書」のプログラムを読解し，それを改造して自分が選択したゲームを作成する．

5. 実験授業と結果

本章では，4 章で述べた「整列・探索アルゴリズム」演習を実験授業として実施した結果を述べる．まず，5.1 節で授業の実施環境，5.2 節で演習実施前の基礎教育について説明する．次に「アルゴリズム概念」が理解されたか，「アルゴリズム構築能力」が育成できたかどうかを評価するため，5.3 節で各演習の達成状況，所要時間，演習中の様子について述べ，5.4 節で演習実施後の自由作品制作の結果について説明する．5.5 節では，各演習で受講者に理解された事柄を明らかにするため，受講者の授業後の感想やレポート記述を抜粋する．

5.1 実施環境

高校（実験授業 A）と大学（実験授業 B）において，「整列・探索アルゴリズム」演習とそれに必要な基礎教育を実施した．実施環境の詳細を**表 3** に示す．どちらの授業も指導者は筆頭著者が担当した．

受講者に共通の前提知識と技能はコンピュータの基本操作（メールの送受信や文書作成）

表 3　実験授業の実施環境
Table 3　Environment of trial lectures.

	高校における実験授業（実験授業 A）	大学における実験授業（実験授業 B）
場所	学習院高等科	慶應義塾大学 湘南藤沢キャンパス
期間	2007 年 4 月〜7 月，9 月〜11 月	2007 年 9 月〜11 月
形態	選択科目「情報 1」1 学期〜2 学期	論理思考とプログラミング 前半
受講者数と学年	12 名（高校 2 年生）	36 名（1 年：33 名，3 年：2 名，4 年：1 名）
受講者のプログラミング経験	なし	4 名（大学の授業や高校にて）
授業時間数	50 分 × 2 コマ/週	90 分 × 2 コマ/週
アシスタント	なし	Squeak の指導経験がある学部生 2 名
基礎教育	15 時間（うち自由作品制作に 8 時間）	12 時間
「整列・探索アルゴリズム」演習	5 時間	6 時間

表 4　基礎教育の内容
Table 4　Contents of preparation course.

内容（括弧内はテキスト[16] の章番号）	実験授業 A（分）	実験授業 B（分）
ことだま on Squeak のインストール（Project0）	30	20
ペイントツールの使用方法とオブジェクトの作成（Project1）	70	70
繰り返し・順次実行（Project2）	100	90
条件分岐（Project3）	100	180
変数の概念と値の書き込み，読み出し（Project4）	100	90
複数のオブジェクトの変数の連携（Project5）	30	45
計算と乱数（Project6）	—	45
変数の定義（Project7）	—	180
自由作品制作（Project8・Project9）	470	—
計	900	720

である．実験授業 B を実施した「論理思考とプログラミング」という授業は，文系の学生を対象とした授業であり，コンピュータに苦手意識のある受講者が大半である．

5.2 基礎教育

「整列・探索アルゴリズム」演習の実施前には，我々が開発したテキスト[16] に沿って「基礎教育」を実施した．授業内容の一覧と，各実験授業における授業時間数を**表 4** に示す．表 4 中，「—」は授業内で学習をしなかったことを示す．実験授業 A では，基礎教育のまとめとして，ゲームをテーマとした作品制作を実施したため，基礎教育後半の内容（計算と乱数，変数の定義）は扱っていない．実験授業 B では，作品制作の時間を確保できなかったため，作品制作は実施していない．

授業の実施にあたっては，毎回の授業のはじめに指導者による解説を行い，その後受講者が各自のペースで演習を実施する形態とした．指導者による解説は，テキストに掲載されている例題の動作例を受講者に閲覧させて説明することを中心に行った．適宜，前回の授業の復習や，学習した概念を整理するための解説も実施している．演習はテキストに掲載されている問題を使用し，例題を基にプログラムを改造する問題，例題で学習した概念を使って新しいプログラムを作る問題を複数題出題する形式とした．

表 5　実験授業における演習の達成度と所要時間
Table 5　Performance records and necessary time of exercises.

	実験授業 A（高校）		実験授業 B（大学）	
	所要時間	プログラムの完成率	所要時間	プログラムの完成率
「手作業による最小値選択法」演習	100 分	11 名/11 名中（100%）	90 分	31 名/31 名中（100%）
「最小値選択法の実装」演習	70 分	11 名/11 名中（100%）	90 分	31 名/31 名中（100%）
「挿入法の構築」演習	50 分	11 名/11 名中（100%）	90 分	28 名/28 名中（100%）
「探索アルゴリズムの応用」演習	100 分	9 名/12 名中[⋆1]（75%）	90 分	18 名/28 名中（64%）

5.3 演習の達成状況と所要時間

5.2 節で述べた基礎教育が終了した後に，「整列・探索アルゴリズム」演習を実施した．各演習の達成状況と所要時間を**表 5** に示す．表 5 に示した記録は，受講者の自己申告ではなく，演習中に指導者と学生アシスタントが巡回し，プログラムの動作確認を実施して集計したものである．次から，4 つの演習ごとに結果を述べる．

5.3.1 「手作業による最小値選択法」演習

「手作業による最小値選択法」では，受講者全員が選択法のアルゴリズムを理解し，手順に従ってカードを並べ替えることに成功した（表 5 中，「手作業による最小値選択法」演習を参照）．

並べ替え係と計測係でペアを組ませたことにより（4.2.1 項），並べ替え作業と時間計測の作業が同時に円滑に実施できた．

演習の開始前には，フローチャートの指示どおりに並べ替えを行うように指示していた．しかし一部のペアは，アルゴリズムが記述されたフローチャートの詳細な部分を確認せず，アルゴリズムの一部を勝手に変更して並べ替えを行おうとしていた．そうしたペアには再度フローチャートの詳細な部分までよく確認し，指示されたアルゴリズムどおりに作業をするように指導する必要があった．

5.3.2 「最小値選択法の実装」演習

「最小値選択法の実装」では，受講者全員が「手作業による最小値選択法」の演習で理解したアルゴリズムをプログラムとして記述し，その動作を検証することに成功した（表 5

⋆1 実験授業 A において「探索アルゴリズムの応用」だけに出席した 1 名の受講者に関しては，他の受講者とペアを組ませて演習に取り組ませた．欠席した演習の内容に関しては，ペアの相手に教えてもらうように指導した．

中，「最小値選択法の実装」演習を参照）．技術的な事柄でプログラミングにつまずく受講者はいなかった．

表 5 中の所要時間には「入れ物」の説明等の時間が含まれているので，受講者がプログラミングに取り組んだ時間は所要時間の 8 割程度である．

3 割程度の受講者は，手作業で実行したアルゴリズムを完全に理解していると考え，手作業で使ったフローチャートを確認せずにプログラミングを行っていた．これにより条件分岐の条件設定やカードの移動の順番を誤って実装してしまい，プログラムが正しく動作せず，試行錯誤を行っていた．指導者は手作業で利用したフローチャートをよく確認し，フローチャートとプログラムとの対応関係を確認するように指導した．このような指導を受けた受講者はプログラミングの間違いに自分で気付き，それを修正して，正しく動作するプログラムを構築していた．

5.3.3 「挿入法の構築」演習

「挿入法の構築」では，すべての受講者が挿入法のアルゴリズムを理解し，それをフローチャートとして記述し，プログラミングすることに成功した（表 5 中，「挿入法の構築」演習を参照）．講師やアシスタントは「ことだま on Squeak」の操作方法に関する数件の質問のみに回答した．受講者が組み立てたアルゴリズムが正しく動作しない場合も，具体的な修正点を教えず，フローチャートを見直すこと，動作結果を再度検証すること，をアドバイスしたのみである．

表 5 に掲載した所要時間は演習内容の解説，アルゴリズムの分析とフローチャートによる設計，プログラミングと動作検証を合計した時間数である．実験授業 B における演習中の受講者の活動時間の内訳とその分布を**図 17** に示す．図 17 中の「アルゴリズムの分析とフローチャートによる設計」は，完成目標のプログラムが動作している動画を観察することで，挿入法のアルゴリズムを分析・理解し，それをフローチャートに記述する活動を指す．「プログラミングと動作検証」は記述したフローチャートを「ことだま on Squeak」上にプログラミングし，実行結果を確認し，必要があれば修正する活動を指す．図 17 の内容に関する考察は 6.1.2 項（2）で述べる．

5.3.4 「探索アルゴリズムの応用」演習

「探索アルゴリズムの応用」では，受講者の約 7 割が，指導者が提示した完成品のサンプルと同じ機能を持つプログラムを完成させることができた（表 5 中，「探索アルゴリズムの応用」演習を参照）．

残りの 3 割の受講者に関しては，点数計算などの機能の追加が間に合わず，プログラムは未完成となった．しかし，探索アルゴリズムを理解し，それを異なったデータに対応できるように修正するという本質部分のプログラムが未完成の受講者はいなかった．

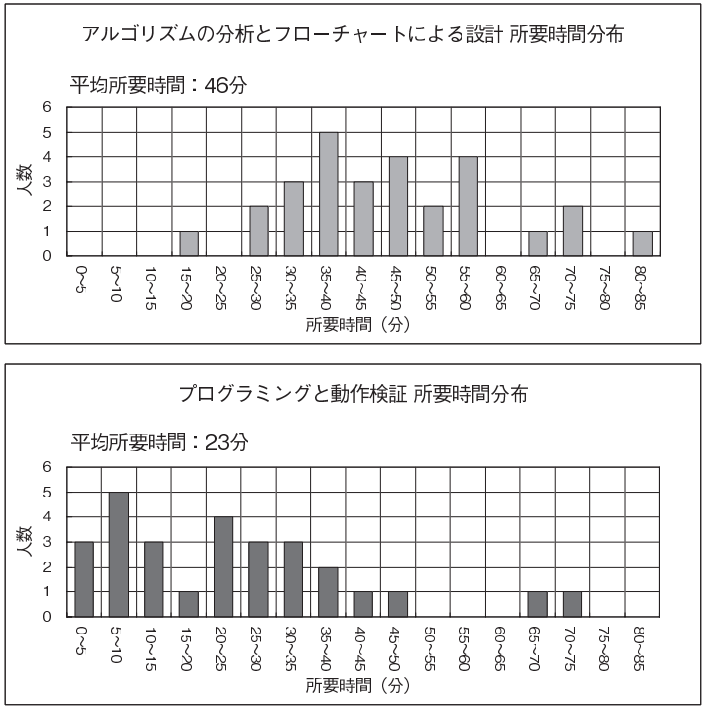

図 17　実験授業 B における「挿入法の構築」演習の所要時間分布
Fig. 17　Distribution of time records about “Constraction of insertion sort” (trial lecture B).

5.4 自由作品制作の結果

実験授業 A では「整列・探索アルゴリズム」演習を実施した後，約 5 時間でテーマを自由とする作品制作を実施した．結果として，約 7 割[⋆1]の受講者が自らの作品の部品として「入れ物」を利用し，複数のオブジェクトの集合を扱うアルゴリズムを組み立て，プログラミングすることができた．このような作品の例として「連想クイズゲーム（**図 18**）」と「バ

⋆1 残りの約 3 割の受講者もアルゴリズムを構築していないわけでない．作品の仕様により，「入れ物」とそれに関連したアルゴリズムを利用しなくても完成できる作品を制作した．

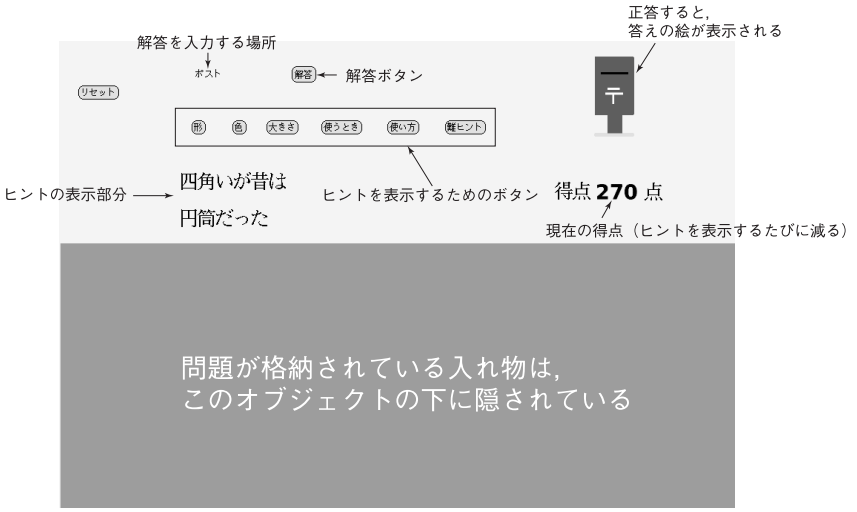

図 18　連想クイズゲーム（実験授業 A の受講者作品例）
Fig. 18　Word Association Game (student work in trial lecture A).

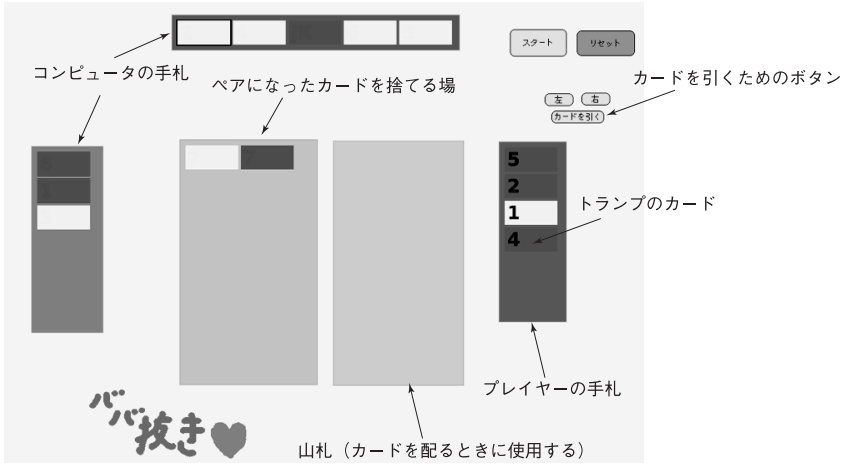

図 19　ババ抜きゲーム（実験授業 A の受講者作品例）
Fig. 19　Old Maid Game (student work in trial lecture A).

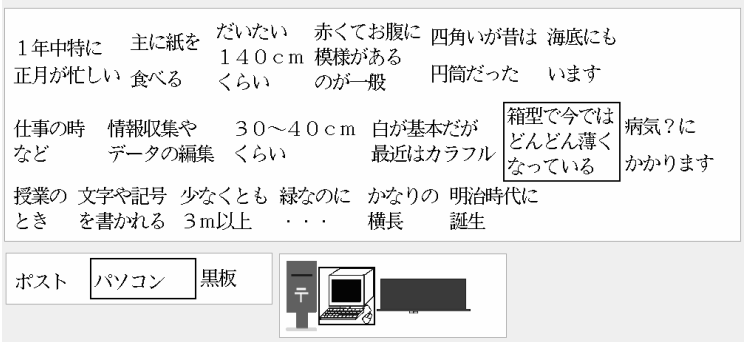

図 20　連想クイズゲームにおける問題データの格納部分
Fig. 20　Question database of Word Association Game.

バ抜きゲーム（**図 19**）」を取り上げ，受講者が作成したアルゴリズムを解説する．2 つの作品を用いた教育効果の分析は，後の 6.1.2 項（1）で述べる．

「連想クイズゲーム」は，問題として出題される物の名前をヒント（形，色，大きさ，使う時，使い方）を参考にしながら，推測するゲームである．図 18 は「形」に関するヒントを表示させ，正解である「ポスト」を入力した後の画面である．3 つの「入れ物」を連携させ，現在出題されている問題に関するヒントを表示するアルゴリズム，正解時に解答の画像を表示し，出題する問題を次に進めるアルゴリズムなどがプログラミングされている．ヒントの表示や答え合わせに関するプログラムは「入れ物」を使って抽象的に記述されている．**図 20** に示す 3 つの「入れ物」に問題のデータ（ヒント，正答の文字列，画像のセット）を追加すれば，プログラムの修正なしで適切な動作をするように作られている．

「ババ抜きゲーム」は，コンピュータが自動操作するプレイヤ 2 名とトランプのババ抜きで対戦するゲームである．カード束を格納しておく部分（各プレイヤの手札，山札，手札を捨てる場）に「入れ物」が利用されている．山札から手札にカードを配るためのアルゴリズム，手札に同じ数字のカードのペアがあるかどうかを検索して場に捨てるためのアルゴリズム，各プレイヤの手札の枚数から，どのプレイヤが勝利したかを表示するアルゴリズムなどがプログラミングされている．

5.5　受講者の感想とレポート記述

実験授業 B では，各演習を実施した授業終了後に数行程度の感想の提出を求めた．さらに，すべての演習が終了した後で，演習を総括したレポートの提出を求めた．

表 6　受講者の感想とレポート記述（抜粋）
Table 6　Students' comments and description of students' report.

アルゴリズム概念の理解に関して	A	普段私たちが考えていることを論理的に表現することの難しさを学んだ．とくに授業内で最小値選択法の演習を行ったときにその大変さを実感した．手作業で数字のカードを昇順に並べ替えるとしたら，見ただけで無意識にカードの比較が行われ，数秒で答えを出すことができる．しかし，それでは他者に理解できるように説明することはできない．
	B	2 時限目の最小値選択法の実装では，1 時限目に自分の手で実行したことをプログラミングに直すのに苦戦した．しかし，論理的な流れが頭の中で組めていたので，組み合わせるプログラム自体はすぐに思いつけた．闇雲にプログラムを組むのではなく，頭の中で流れをはっきりさせておくことが重要なのだということが分かった．
	C	はじめは手作業で仕組みを理解したことによって，次の時間にコンピュータを使って並べ替えのアルゴリズムをプログラムとして書きやすくなりました．それは頭の中を 1 度きれいに整理できたからだと思います．
アルゴリズムの構築に関して	D	挿入法の内容をしっかりと理解して，フローチャートが書ければ，プログラムを作るのが容易になるんだということを改めて感じました．
	E	ゲームを作っているときには，自分の中にフローチャートが浮かんできて，考えていることがプログラムとして反映できるようになってきた．

多くの学生が「手作業で実行した仕事をプログラミングすることは興味深かった」とコメントしていたことから，学習者が高い意欲を持って演習に取り組めたことが読み取れた．

次から「アルゴリズム概念の理解」と「アルゴリズムの構築」に関して，具体的にその理解の様子が記述されている感想やレポート記述を抜粋して（**表 6**），説明を行う．

5.5.1　アルゴリズム概念の理解に関して

表 6 中の A からは，人間は普段手順を意識しないで仕事をしていることを理解し，それをプログラミングするためには手順に分解して詳細化することが必要なことを実感したことが読み取れる．これに類似した記述は多くの受講者のレポートにも見られた．

表 6 中の B，C には，「流れをはっきりさせる」「きれいに整理する」という記述が見られる．これは「アルゴリズムを理解すること」を指していると推測できる．手作業でアルゴリズムを理解したことにより，自力でプログラミングが可能になる体験をしたことが読み取れる．

5.5.2　アルゴリズムの構築に関して

表 6 中の D のコメントによって，アルゴリズムを構築するために必要な一連の活動を体験し，「プログラミングを円滑に行うためには，アルゴリズムそのものを理解し，それを詳細化するという段階が重要である」ことを理解していることが分かる．

一部の受講者からは，表 6 中の E のように，アルゴリズム構築能力の成長を自己評価する感想もあった．

6. 考　　察

本章では，高校と大学における実験授業の実施結果を分析し，考察を述べる．

6.1　教育効果

本節では，演習が「アルゴリズム概念の理解」と「アルゴリズム構築能力の育成」にどのように寄与したかについて考察する．また，演習の題材がプログラミング学習の動機付けに対してどのような意味を持ったかについて述べる．

6.1.1　アルゴリズム概念の理解に対する効果

実験授業では，受講者全員が手作業で実施した選択法による整列アルゴリズムを理解し，プログラミングを行うという体験をすることができた（表 5）．題材となったアルゴリズムはプログラミングの初学者にとっては複雑なものである（図 11）．このような複雑なアルゴリズムを手作業を通じて理解し，それを自力でプログラミングさせることができたことにより，我々が教育の目標とした，a）ある目的を達成するための仕事は単純な手順の積み重ねによって構成されていること，b）それぞれの手順は解釈に曖昧さがないように詳細化されていること，という「アルゴリズム概念」を体験的に理解させることができた．

前述したアルゴリズム概念の理解の内容に関しては，手作業とプログラミングの両方を取り入れた演習によって，コンピュータと人間のそれぞれの特性に注目した理解が得られることが分かった．表 6 中の A に代表例を示したように，実験授業の受講者のレポートの多くに，人間とコンピュータの特性を比較し，普段の活動では人間が手順を意識していないことを実感し，それをプログラミングするためには詳細化が必要であることが理解できたという記述が見られた．これは，選択法による整列問題を，手作業とコンピュータの両者で解決するという演習の成果である．

「最小値選択法の実装」の演習後の受講者の感想には「フローチャートどおりにやるってことがやっと分かった」という記述が見られた．実験授業では，基礎教育における条件分岐の学習の際に，フローチャートを用いて，順次実行や条件分岐の概念と「ことだま on Squeak」上での表現方法に関する 90 分程度の演習を実施していた．しかし，この基礎教育だけでは，フローチャートが詳細化されたアルゴリズムを記述したものであり，それを忠実にプログラミングすれば，正しく動作するプログラムが構築できるということ，また順次実行や条件分岐の概念がどのように「ことだま on Squeak」上で表現できるかについてが受講者に理解

されていなかった．これは，基礎教育の段階で構築可能な単純なアルゴリズムの場合，順次実行や条件分岐の概念理解が曖昧であっても，それに受講者本人が気付かず，学習を進めてしまうことを示している．したがって，表 4 で示した順次実行や条件分岐に関する学習内容は十分ではなく，一定以上の複雑さを持つアルゴリズムを構築する体験を通じて，アルゴリズムの概念を理解することにより，順次実行や条件分岐の概念理解を補強する必要があることを示している．

6.1.2　アルゴリズム構築能力の育成効果

（1）演習の達成度と自由作品制作の結果

表 5 に示したように，実験授業で実施した演習の結果は，「挿入法の構築」に関しては受講者全員，「探索アルゴリズムの応用」に関しては約 7 割の受講者が自力でプログラムを完成させることができた．「挿入法の構築」では，「挿入法を用いてカードを整列する」という課題を受講者が理解・分析し，そのアルゴリズムを詳細化してフローチャートとして記述し，「ことだま on Squeak」でプログラミングを行うという一連の活動が実践できた．「探索アルゴリズムの応用」では，指導者から提示された探索アルゴリズムのプログラムを理解し，仕様が指定されたゲームに応用するという活動を実践することができた．これらの演習の結果から，1 章で述べた「与えられた課題を理解・分析して，それを手順に分解・詳細化し，プログラムとして記述可能なアルゴリズムを組み立てる」という「アルゴリズム構築能力」の育成が実践できたと考える．

さらに，5.4 節で述べた作品制作の結果は，受講者が演習の題材となった特定のアルゴリズムを理解しただけでなく，完成目標である作品（課題）を分析し，それに必要なアルゴリズムを組み立てられたことを示していると考える．5.4 節で述べた「連想クイズゲーム」を制作した受講者のレポートには「最初の授業で最小値選択法の独自のアルゴリズムを作成し，ミニゲームを作成したことにより，最終作品の作成時に作品のフローチャートを思い浮かべながら作業を行ったので 1 学期に作成したもの（演習実施前の基礎教育における作品制作を指す）よりも，多くの場合に対応できた作品が作成できたと思う」という記述が見られた．この記述から，演習で扱った具体的なアルゴリズムを理解しただけではなく，演習におけるアルゴリズム構築体験のプロセスを，作品の制作にも適用できていることが読み取れる．5.4 節で述べた「ババ抜きゲーム」を作成した受講者のレポートには「入れ物 1 つに対して 1 つのカーソルがあるわけだが，ばば抜きにおいては 2 枚のカードを比較するということをしなくてはならない．つまり 1 つの入れ物だけでは，物と物を比較することができないということが，作り始めて最初にぶつかった壁だった．（中略）1 枚のカードをそのほかのカードとすべて比較しなければならなかったため，テキストにカードの数字を記憶させる．という方法を使って制作にあたった」と述べられている．これは，作品における課題（同一の数字が書かれたカードのペアを探索し，手札から場に捨てる処理）と「入れ物」の仕様の関係を適切に分析し，その解決策とアルゴリズムを組み立てたことを示すものである．ここで述べた 2 名以外の受講者についても，「整列・探索アルゴリズム」演習を実施した後の作品においては，演習実施前の作品より，構築の難易度が高いアルゴリズムをプログラミングしていることが確認できた．これは「整列・探索アルゴリズム」演習を実施したことが「アルゴリズム構築能力」の育成に寄与したことを示すものであると考えられる．

（2）「アルゴリズム概念の理解」の影響

演習中の受講者の活動を分析した結果から，「アルゴリズム概念の理解」が「アルゴリズムの構築」における分析や詳細化の段階の重要性を認識させる効果があることが分かった．プログラミングの初学者は，アルゴリズムの分析や詳細化という段階を軽視し，たとえ指導者からフローチャートを記述せよという指示があっても，十分にアルゴリズムを詳細化しないままにプログラミングを始める傾向が多く見られる．実験授業における「挿入法の構築」演習においてはそうした受講者はいっさい観察できなかった．図 17 に示したように，受講者はアルゴリズムの分析と詳細化に演習時間の約 7 割（28 名の平均 46 分）をかけている．演習中の受講者の様子を観察した限りでも，アルゴリズムの分析とフローチャートによる設計に熱心に取り組み，プログラミングの段階でアルゴリズムの不備が分かった受講者はすぐにフローチャートを見直していた．こうした行動は「プログラミングを行うためには，アルゴリズムを理解することが必要である」（5.5.1 項で述べた受講者のレポート，表 6 中の B，C に示した）という，アルゴリズム概念の理解が影響していると推測される．このことより，「アルゴリズム概念」を理解させる演習が，アルゴリズム構築能力の教育に重要な意味を持つものと思われる．

6.1.3　演習の題材と学習の動機付け

5.5 節の冒頭でも述べたように，本論文で提案した演習は，大多数の受講者が高い意欲を持って取り組んだ．

プログラミングの初学者の場合，理解可能なアルゴリズムは単純なものに限られてしまう．また，プログラミング言語の習熟度は低いため，自力でプログラミングできる問題は「決められた回数だけ画面に文字を出力する」「指定された図形を描画する」といった，コンピュータ上でしか意味を持たず，解決に対する意欲が湧かない題材となりがちである．

我々は，プログラミングの初学者に，現実世界で意味を持つ問題（たとえばカードの整

列）を，自力でプログラミングさせる体験を与えることができた．これは 1 章で述べた UNESCO の提案における「現実の問題をアルゴリズミックに解決できるようになること」とプログラミングの学習を結びつける役割を果たす．

これに関連して，実験授業 B の受講者のレポートには，「プロジェクトを遂行する」「論理的な文章を書く」といったことにプログラミングで学習した概念が応用できるといった記述が見られた．これは，本論文で述べた演習の題材が，「プログラミング」と「現実世界の問題解決」のつながりが理解しやすいものであったことが影響していると考えられる．

6.2 「ことだま on Squeak」の効果

4.2.3 項で述べた「挿入法の構築」演習が円滑に実施できた要因の 1 つは「ことだま on Squeak」を使用したことであると推測される．その理由として，以下の 3 点があげられる．

(1)　構文エラーがいっさい発生しない．
(2)　プログラムの実行結果が具体的なオブジェクトとしてつねに画面上に表示されている．
(3)　学習者が考えるアルゴリズム記述とプログラムによる表現の乖離が少ない．

これらの理由のうち，(1) と (2) に関しては，Squeak eToys の日本語版から引き継いだ特徴である．(3) は「ことだま on Squeak」の持つ特徴である．

以下に，これらの理由を，高校や大学での使用実績が報告されている教育用プログラミング言語「PEN」[10] と比較して考察する．

(1) に関しては，「PEN」には定型構文を入力するためのボタンが用意されているが，自由にプログラムを書くことができるため，構文エラーが発生する可能性は残されている．「ことだま on Squeak」では構文エラーが発生することはない．学習者が組み立てたアルゴリズムが正しければ，構文エラーの修正作業をしないで，プログラムを完成させることができる．構文エラーを修正する作業に必要な時間をなくすことができ，アルゴリズムを考えるための時間を長くとれたことが演習の成功要因の 1 つである．

(2) に関しては，「ことだま on Squeak」では，カードを表現したオブジェクトや「入れ物」の中身がつねに表示されており，プログラムの実行結果が視覚的に確認できる．「PEN」にもプログラムで使われている変数と値が一覧表示できる機能があるが，その表示形式は変数や配列とその値という変数表の形式を採用している．したがって，変数表で表現されていることが，カードやカード束を抽象化しているものであることを説明し，理解させる必要がある．4.2.3 項で述べた「挿入法の構築」演習では，受講者に完成目標のプログラムの動作を観察させ，アルゴリズムを理解させた．カーソル（配列のインデックス）に関する説明は必要だが，それは 10 分程度の時間しか必要としない．「ことだま on Squeak」を利用することにより，学習者はアルゴリズムを「カード」と「カード束」という具体物に近い「カードオブジェクト」や「入れ物」を使って視覚的に理解することができる．カードと変数の関係，カード束と配列の関係に関する説明は必要なく，アルゴリズムを理解させることだけに時間を使うことができた．さらに，プログラミングの初学者は変数や配列といったプログラミング言語に特有の概念を使って，アルゴリズムを考えることは難しい．「ことだま on Squeak」では，カードやカード束といった具体物を想像しやすい環境でアルゴリズムを考えることができる．

(3) に関しては，学習者がフローチャートを考える際には，その処理内容を自らが理解できる表現で記述する．たとえば配列のインデックスを増加させる処理は，実験授業の受講者が作成したフローチャートのほとんどに「ソート済束のカーソルを右に 1 つずらす」と記述されていた．これは，学習者が完成目標となるプログラムの動作結果を見て記述したものである．つまり，学習者がアルゴリズムを自ら試行錯誤する段階の処理表現は，このような日本語表記であることが分かる．同じ処理を「ことだま on Squeak」で記述する場合の表現は「ソート済束のカーソル位置に 1 を足す」である．学習者が記述する処理表現と若干の乖離は存在するものの，実験授業の演習中には自分が考えた処理をどのようにプログラミングするかに関する質問はいっさい発問されなかった．

「PEN」は日本語でプログラムを記述することができ，プログラムの文法の説明がなくても処理が理解できるような配慮がなされている．しかしながら，プログラムの表現にプログラミング言語に特有の記号[⋆1]が入る．したがって，「ことだま on Squeak」の場合より，学習者が理解可能な処理表現とプログラムの表現との乖離は大きくなる．学習者がアルゴリズムを考える段階の表現とプログラミング言語による表現の乖離が大きい場合，学習者は設計したアルゴリズムをプログラミング言語に変換する作業を行う必要がある．この作業はアルゴリズムを考える時間を奪うことになる．

「ことだま on Squeak」には，前述したような利点がある一方で，1）関数内のプログラムを 1 ステップだけ実行することができない，2）入れ子になった繰返しがプログラミングできない[⋆2]，という欠点もある．このような欠点は「PEN」には存在しない．とくに 2）の欠点は，a）実装可能なアルゴリズムが一重の繰返しループを使ったもの（たとえば，図 10）に制限され，不必要な処理が増えること，b）入れ子の繰返しを使ったアルゴリズムを組み

⋆1 配列の宣言の角括弧「例：array[10]」や，代入記号の「←」など．
⋆2「ことだま on Squeak」の実装基盤である Squeak eToys の仕様による．

立てた受講者に対して，アルゴリズムの制御構造を単一の繰返し構造に変形するように指導する必要があること，の原因となる．しかし，4 章で述べた「整列・探索アルゴリズム」演習で扱った複雑さのアルゴリズムであれば，これらの欠点は大きな問題にはならなかった．

我々は，学習者の能力の発達段階に合わせてプログラミング言語を使い分けることで，こうした欠点を解消すればよいと考えている．学習者が「アルゴリズム概念」の理解を得た後の教育では，「PEN」をはじめとした教育用プログラミング言語や，一般的な実用言語を利用した方が効果的に教育を実施できる．こうした考え方に基づき，実験授業 B では学期の後半は Java を利用して教育を実施した．

6.3　教育に必要な時間

基礎教育を実施した後，4 章で述べた「整列・探索アルゴリズム」演習を実施するのに必要な時間は約 6 時間である．各演習におけるプログラムの完成率，演習の様子から判断して，これ以上の短期間で実施するのは難しいことが分かった．

実験授業では，基礎教育に約 12 時間，「整列・探索アルゴリズム」演習に約 6 時間をかけている．完成されたサンプルプログラムを与え，それをそのまま動作させたり，一部を改変するだけの演習であれば，より短時間で実施が可能と推測できる．しかし，本論文で述べた演習では，課題を与え，そのアルゴリズムを受講者自らが組み立て，プログラムとして記述するという形式で演習を実施している．プログラミングの初学者を対象として，こうした形式で演習を実施する場合は，本論文で述べたように，18 時間から 20 時間程度の授業時間が必要であると考える．実際，専門学科の入門教育[9] でも 3 時間 × 12 回の最後の 2 回を使って，整列プログラムに関する教育を実施している．

6.4　今後の課題

「整列・探索アルゴリズム」演習に要求される「ことだま on Squeak」の操作はそれほど多くないため，基礎教育の内容を洗練させ，より早い段階で「整列・探索アルゴリズム」演習を実施することも可能と考えられる．6.1.1 項でも述べたように，順次実行や条件分岐の概念理解は基礎教育（表 4）の実施だけでは不十分であるため，変数の概念の学習が終わった直後に「整列・探索アルゴリズム」演習を実施することを検討すべきである．学習の初期の段階で「アルゴリズム概念」を理解させることにより，その後に実施する教育の効果を高めることができる可能性がある．

本論文で述べた演習のうち，その手順が複雑なものに関しては，受講者に配布可能なワークシートとして整理してある．また，基礎教育として実施すべき内容は，我々が執筆したテキスト[16] にまとめられている．実験授業 A，B はこれらのワークシートやテキストを使用して実施した[⋆1]．本論文で述べた演習の内容は中学生でも理解可能なものであり，冒頭で示した UNESCO のカリキュラム提案も中等教育を対象としたものである．しかし，現状の学習指導要領の内容，全国の教科「情報」の実施実態を考慮すれば，本論文で提案した教育を国内の中等教育課程で実施することは難しいというのが実情であろう．中等教育で本格的なプログラミング教育が実施しにくい現状では，これを当分の間は大学の入門教育で実施する必要がある．したがって，演習内容も大学生に適するように充実させる努力も必要である．

7. おわりに

本論文では，一般情報教育の一環として実施するプログラミング教育において「アルゴリズム構築能力」を効果的に育成するための方法を提案し，その効果を検証した．

アルゴリズムの概念を教育する方法として，手作業で実行可能なアルゴリズムを実行・理解させ，それをプログラミングさせるという方法について述べた．この際，1）現実世界の教具を抽象データ型を使って表現する，2）手作業によるアルゴリズム記述とプログラミング言語による処理の表現の乖離をできる限り少なくする，という 2 点の工夫をしたアルゴリズム実行環境を用意した．これにより，手作業で実行したアルゴリズムを，学習者が自力でプログラミングすることが可能になることを示した．

高校と大学で実験授業を行った結果，受講者全員が「挿入法によるカードの整列」という課題を理解して，コンピュータが実行可能なアルゴリズムを自力で組み立てられるようになったことが確認できた．また，探索アルゴリズムを応用したゲーム作成の演習では，約 7 割の受講者がプログラミングに成功した．さらに，演習実施後に実施した自由作品制作の結果から，演習で扱った特定のアルゴリズムを理解するだけでなく，作品の完成に必要な課題を分析し，それを解決するためのアルゴリズムを組み立てられるようになったことが確認できた．これらの結果より，提案方法が「アルゴリズム構築能力」の育成に効果があることが示された．

演習の結果を分析した結果，人間とコンピュータのそれぞれの情報処理の特性と関連付けたアルゴリズム概念の理解が得られること，アルゴリズム概念の理解が，アルゴリズムの構築における分析や詳細化の段階の重要性を認識させること，が分かった．さらに，演習の成功要因として，プログラミング言語として採用した「ことだま on Squeak」が果たした役割が大きいと推測される．

⋆1 テキストやワークシートは Web[13] からダウンロードできる．

プログラミング学習の初期の段階で，本論文で提案した教育を行うことによって，その後のプログラミング教育が効率良く行えることが期待できる．

謝辞　本研究の一部は，経済産業省による平成 17，18 年度 IT クラフトマンシップ・プロジェクト事業の支援を受けて実施されました．実験授業に参加していただいた皆様，論文のレビューを引き受けて下さった神沼靖子博士に謝意を表します．

参 考 文 献

1) 情報処理学会情報処理教育委員会：日本の情報教育・情報処理教育に関する提言 2005 (2005). http://www.ipsj.or.jp/12kyoiku/proposal-20051029.html
2) UNESCO: ICT Curriculum for School / Program of Teacher Development (2002). http://unesdoc.unesco.org/images/0012/001295/129538e.pdf
3) 楠　房子，宮内　新，小沢慎治：アルゴリズムスタイルを重視した情報処理教育，電子情報通信学会論文誌 A，Vol.J75-A, No.2, pp.441–448 (1992).
4) Computer Science Unplugged (2007). http://csunplugged.com/
5) Holmes, G. and Smith, T.C.: Adding some spice to CS1 curricula, *ACM SIGCSE Bulletin*, Vol.29, No.1, pp.204–208 (1997).
6) 阿部哲也：アルゴリズム教育のための教材の開発，日本教育工学会論文誌，Vol.27, No.Suppl., pp.45–48 (2004).
7) 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.2454–2467 (2004).
8) 加藤　浩，井出有紀子，鈴木栄幸：状況論的アプローチによる情報教育のための協同学習環境のデザインと評価：プログラム対戦ゲーム「アルゴアリーナ」の開発と実践，情報処理学会論文誌，Vol.40, No.5, pp.2497–2507 (1999).
9) 長　慎也，甲斐宗徳，川合　晶，日野孝昭，前島真一，筧　捷彦：プログラミング環境 Nigari—初学者が Java を習うまでの案内役，情報処理学会論文誌：プログラミング，Vol.45, No.SIG09, pp.25–46 (2004).
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) 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).
12) Viewpoints Research Institute, Inc.: Welcome to Squeakland (2007). http://www.squeakland.org/
13) 慶應義塾大学大岩研究室：ことだま on Squeak コミュニティサイト (2007). http://www.crew.sfc.keio.ac.jp/squeak/
14) 岡田　健，杉浦　学，松澤芳昭，大岩　元：プログラミングの初歩概念を学ぶための日本語プログラミング環境，情報教育シンポジウム論文集，Vol.2006, No.8, pp.109–112 (2006).
15) 岡田　健，杉浦　学，松澤芳昭，大岩　元：初心者用プログラミング環境「ことだま on Squeak」の試み，CIEC 2007PC カンファレンス論文集，pp.229–232 (2007).
16) 大岩　元，松澤芳昭，杉浦　学：ことだま on Squeak で学ぶ論理思考とプログラミング，イーテキスト研究所 (2008). http://www.crew.sfc.keio.ac.jp/squeak/

(平成 19 年 12 月 10 日受付)
(平成 20 年 7 月 1 日採録)

杉浦　学（学生会員）

2003 年慶應義塾大学環境情報学部卒業．2005 年慶應義塾大学大学院政策・メディア研究科修士課程修了．現在，慶應義塾大学大学院政策・メディア研究科博士課程在学中．2005 年より学習院高等科非常勤講師（情報），2006 年より慶應義塾大学環境情報学部非常勤講師兼任．情報教育，教育学習支援情報システムの研究に従事．所属学会：CIEC（Council for Improvement of Education through Computers），IEEE Computer Society，日本教育工学会．

松澤　芳昭（正会員）

2000 年慶應義塾大学環境情報学部卒業．2002 年慶應義塾大学大学院政策・メディア研究科修士課程修了．2007 年慶應義塾大学大学院政策・メディア研究科博士課程単位取得退学，博士（政策・メディア）．2004 年より慶應義塾大学環境情報学部非常勤講師，2008 年より静岡大学情報学部特任助教．オブジェクト指向技術を応用したソフトウエアの設計と開発，情報教育，情報システム開発教育の研究に従事．所属学会：CIEC（Council for Improvement of Education through Computers），日本教育工学会．

岡田　　健（学生会員）

2001 年慶應義塾大学環境情報学部卒業．2003 年慶應義塾大学大学院政策・メディア研究科修士課程修了．現在，慶應義塾大学大学院政策・メディア研究科博士課程在学中．2006 年より慶應義塾大学環境情報学部非常勤講師，獨協大学経済学部経営学科非常勤講師兼任．日本語プログラミング言語「言霊」の研究に従事．所属学会：日本教育工学会．

大岩　　元（正会員）

1965 年東京大学理学部物理学科卒業．1971 年東京大学大学院理学系研究科博士課程修了．理学博士．東京大学理学部助手，豊橋技術科学大学講師，同助教授，同教授を経て 1992 年慶應義塾大学環境情報学部教授，2008 年より帝京平成大学現代ライフ学部教授．キー入力訓練法と日本語入力方式の開発，KJ 法支援，都市景観設計支援，ソフトウェア技術者育成法の開発，情報教育の理念と方法，等の研究に従事している．所属学会：CIEC（Council for Improvement of Education through Computers），日本ソフトウェア科学会，電子情報通信学会，教育システム情報学会，日本教育工学会，日本オペレーションズリサーチ学会，人工知能学会．