情報処理学会論文誌　教育とコンピュータ　Vol.1　No.2　1–6 (Mar. 2015)

招待論文

識字教育としてのプログラミング

大岩 元[1,a)]

受付日 2014年12月4日, 採録日 2014年12月20日

概要：21 世紀の市民社会を形成するための基礎としての識字教育として，プログラミング教育を行う必要があることを主張する．コンピュータの安易な教育利用が日本の教育に弊害を与えた例として，大学入試センター試験を示す．識字教育として行うプログラミング教育の内容は，市民が共有すべき知識/スキルであることから，目的/手段展開という一般性のある視点から行い，母語としての日本語を使うことで，プログラミング「言語」教育から決別すべきことを提案する．

キーワード：識字教育，日本語プログラミング，目的/手段展開，入れ子構造

Programming as a Literacy Education

Hajime Ohiwa[1,a)]

Received: December 4, 2014, Accepted: December 20, 2014

Abstract: We propose the programming education should be regarded as a literacy education for 21st century citizens. Development method from purpose to means is introduced as a design methodology for programming together with mother language programming.

Keywords: literacy education, programming in Japanese, intention to means expansion, nesting box structure

1. はじめに

日本の教育問題の根本にあるのは，情報化への対応ができていないことであるように思われる．実際，国民 1 人あたりの投資額は世界のトップレベルであるが，情報化の浸透度は，スイスで毎年行われる World Economic Forum の Network Readiness という評価によれば，日本は 20 位以下と評価されている．この投資効率の悪さは，情報教育が単なる利用法教育として日本ではとらえられていること，またその内容は，そこから生じる不都合への場あたり的な対処法に終始しているからであろう．21 世紀の教育の高度化の観点から，情報教育の中核に位置づけるべきプログラミング教育を識字教育の一環として具体的に提案する．

2. 市民社会形成の基盤としての学校教育

学校という教育制度を組織的に始めたのはチェコのコメニウスである．17 世紀の戦乱のヨーロッパで，「すべての人にすべてのことを教える」ことを理念に教育の方法論としての「大教授学」や，子どもが学びやすいようにと絵入りの教科書「世界図絵」を刊行した．同じころの日本でも寺子屋が始まっていたが，寺子屋の特長は，社会生活を送るうえで必要となる文書の書き方や，計算の方法として算盤を教えたことにある．これらの教育はその後の産業革命を経て生まれた工業化社会においても，社会生活を送る市民の知的能力の基盤を形成するという重要な役割を果たしてきた．

しかし，経営学者のドラッカーは，コンピュータ・ネットワークが社会基盤となった 21 世紀の社会では，学校の役割が高度化しなければならないと主張している [1]．特に，「学校教育は，内容にかかわる知識とともに，方法に関わる知識も教えなければならない」という指摘をしている．これを具体的に実現するには知識だけでなく，これを作り出す方法を教育する必要がある．

学校教育は，工業化社会で生活するための基礎能力とし

[1] 慶應義塾大学
Keio University, Fujisawa, Kanagawa 252–8520, Japan
[a)] ohiwa@sfc.keio.ac.jp

© 2015 Information Processing Society of Japan

て，まず識字教育を行う．その内容は，文字の読み書きと計算である．識字教育は国語と算数で教えられ，国語では単に文字の読み書きだけでなく，文字を使ってことばによる意志疎通が行えるように教育する．算数では単に機械的な計算をするだけでなく，その意味を理解させる．こうしたリテラシー教育は社会生活を送る基礎能力であり，本稿ではこうした教育を識字教育と呼ぶことにする．さらに，我々が生活する世界の動作原理を学ぶ理科と人間社会の成り立ちを学ぶ社会科が小学校では教えられる．

21 世紀の社会は，コンピュータ・ネットワークに依存する社会になった．我々が生活する世界を理解するために，コンピュータとネットワークの動作原理を理解することが必要となったのである．また，コンピュータ理解に欠かせないのがプログラムであるが，これは人間が理解する文書であると同時に，コンピュータも理解してそれを解釈・実行するものである．文書の役割が拡大されたことから，プログラムも識字教育として行う必要が生じてきた．

プログラムを書けばコンピュータ上にある情報に対して可能なすべての情報操作ができる．プログラムを書くことの基礎部分は幼児でも教えればできることが知られているが [2]，このような簡単なプログラムでも大学生に教えたのではうまく書けない．この意味でプログラミングは従来の識字教育と同じ性格のものであり，手遅れになる前に教えなければならない．

コンピュータは何ができて何ができないかを市民全体が理解しておくことは情報化社会の重要な知識基盤であろう．これがないと，不可能ないしきわめて困難なことを，いずれはコンピュータが可能にするという幻想を生んでしまう．

コンピュータは何ができて，何ができないかということを理解するには，自分でプログラムを書く経験を持つのが最善の方法であり，多分唯一の方法であろう．

自分でプログラムを書いてみると，コンピュータは書かれたとおりに動き，書いた人が思ったとおりには動かないことを経験する．この乖離を解消する作業がプログラミングの重要な部分である．アルゴリズム構築におけるこの体験は，プログラミングに限らず，問題解決の教材としても有効性が期待できる．そして，「知識」ではなく，それを作り出す「方法」の教育としても有効性が期待できる．

しかし，プログラムを書くためには，プログラミング言語を用いて，自分がコンピュータにやらせたいことを記述する能力を身につける必要がある．従来のプログラミング教育では，この部分に多大の時間を費して，アルゴリズム構築の段階まで進めないで終わってしまう場合が多く，専門学科の教育においてすら，この段階を教育することがむずかしくなってしまっている．この問題は後述するように，母語としての日本語をプログラミング言語として使用することで解消される．

3. コンピュータを教育に導入した弊害

日本では，1979 年から共通 1 次試験が大学入試に導入され，センター試験へと発展して現在に至るまで続いている．センター試験は，人間の手を介在させずに客観的に評価ができるところが，日本人の公平観に訴えることができて長く続いたのではないかと思われる．最近になってようやくその弊害が指摘されて，改善の検討が始まった．しかし，コンピュータを試験に導入することに関する根本的な議論は行われていないように見うけられる．

試験は，受験者の能力を評価する実験であるととらえると，そこでは何を測定しているのか，誤差はどれだけあるのかが問題となる．測定では，再現性が問題となるが，この点についての議論はほとんど行われていない．一方で，米国で一般化したこの種の試験では，結果の再現性についての努力がなされており，そうした方法論を用いている TOEFL や TOEIC といった試験の方が，実社会では評価されるようになってしまった．

センター試験の問題の根源は，科目間の不公平が生じないように，平均点を揃えることに起因している．これを単純に各科目の試験の平均点を 60 点にするよう出題を努力する，ということで行うために，受験勉強の成果として平均点の上昇が起こり，それを 60 点にするために問題の難化が起こり，受験技術を身につけないと，必要な成績を確保できなくなってしまっている．この結果として，数学においても，考えていたら時間内に解に到達できず，必要な解法をすべて覚えて，出題された問題に対して適用すべきものを選び出して，ひたすら正確な計算を行う，という解法をとらざるをえなくなっている．受験生に，データベースと計算という，コンピュータの代替をさせるようなことを強制しているのである．これは，コンヒュータ導入の弊害といってよかろう．この結果，大学への進学を目指す学生に対しては，正しいことを覚えるだけの教育が日本中で行われるようになってしまった．

こうした事態が生じるのは，コンピュータで採点することに関して，教育の観点から根本的な問題を検討することなく，実利性だけで導入したことから生じたように思われる．コンピュータの利用が社会的に大きな弊害を生んでいる顕著な例ではなかろうか．

4. ユネスコの勧める情報教育

2002 年に発表されたユネスコの勧める情報教育 [3] では，ICT Specialization として推奨している教育内容が，情報技術の専門家とともに，高等教育に進学する学生のための内容であるとしている（**表 1**）．実際，欧米の大学卒のビジネスマンはみな，プログラミングができるのが当然で，できなければ大学に進学しなかった人であると見なされるそうである．

© 2015 Information Processing Society of Japan

表 1　UNESCO の勧める情報教育
Table 1　Informatics education recommended by UNESCO.

- ICT Literacy
- Application of ICT in Subject Areas
- Infusing ICT across the Curriculum
- ICT Specialization
  - For those who plan to go into professionals that uses ICT
  - ***For those who plan to go into higher education***

表 2　ICT specialization の内容
Table 2　Contents of ICT specialization.

Students of ICT Specialization should be able to solve routine everyday problems in an algorithmic form.

- Specialization Preparation Modules
  - SP1 Introduction to Programming
  - SP2 Top-Down Program Design
- General Specialization Modules
  - GS1 Foundation of Programming and Software development
  - GS2 Advanced Elements of Programming
- Vocational Specialization Modules
  - VS1 Business Information Systems
  - VS2 Process Control Systems
  - VS3 Project Management

ユネスコの提案では，ICT Specialization は日常生活の問題をアルゴリズムとして解く能力を身につけることを目的としている（**表 2**）．そして，その最初の内容である Introduction to Programming では，課題を解くアルゴリズムを設計（Design）すること，その設計をプログラムとして実行可能な表現に変換すること，最後にプログラムを実生活で利用できるようにする所までを体験させるよう，推奨している．

このことは，プログラミングの入門段階から，プログラムが使用される環境の中で，何を実現するかという仕様を考え，それを実現するアルゴリズムを構築したうえで，それをプログラムとして表現して，実際に使用して評価するところまで体験させること意味する．これこそが，ドラッカーが主張する「方法」に関する教育に関する具体的な内容をなすものである．

5.　目的を手段に展開する設計

自然言語だけを用いて行う問題解決，特に最近普及してきたプロジェクトを行う学習においては，結果の評価が困難で，専門家が当然として行う判断が，学習者には理解できないことが多い．理解が困難なために，指導者も，評価を明示的に行わず，学習者には目的が達成できたような幻想を与えてしまうことが多い．

プログラミングが自然言語を用いた学習とちがうところは，少数の語彙と，単純な文法だけで表現できることである．プログラムによって実現される機能は，小さなものであっても，何らかの目的を達成するものであり，その目的が達成されたかどうかを判断することは，学習者でも容易に行うことができる．

このことから，プロジェクトを行うような「方法」に関する学習の教材として，プログラミングは最適であるといえよう．小さな問題から大きな問題まで，その制作をプロジェクトという視点で行うことができるからである．そのための方法論として，HCP チャートによる設計 [4], [5] が有効である．

HCP チャートは，構造化プログラミングが導入されたときに，流れ図の代替として日本で提案された多くの設計図法の 1 つである．その基本的な考え方は，作りたいプログラムの目的を手段に展開した結果を図示することである．手段はそれを実行する順に縦に並べて記述する．さらに，展開された目的はそれぞれ，次の展開の目的となる．このような展開を行うと，最後は実行されるコードとなる．HCP チャートを用いて具体的なプログラムの設計を行うことで，抽象化という Computing Science の最も重要な概念を教育することができる．

目的/手段展開は，組織運営のための経営学にも中核概念として役に立つ．トヨタは 1990 年代に「ワークデザイン」という方法論を現場に導入したが，その発展形であるブレークスルー思考においては，目的/手段展開が中核概念として用いられている [6]．

このような展開を行うと，右側にいくほど具体的な記述となる．逆に左側は抽象的な記述ということになる．そして，同じレベルの手段の抽象度がそろっていなければならない．もう 1 つこの展開で問題となるのは，ある目的を実現する展開手段の粒度がそろうことである．

たとえば，タートルグラフィックスで三角形と四角形で家を描くプログラムを考えてみる．家を描くという目的が，屋根と本体を描くという手段に展開される．さらに，屋根は三油形で，本体は四角形で描くと設計を展開できる．

しかし，単に家を描くことを実現するだけであれば，三角形を描く，四角形を描くという具体レベルの手段を 2 つ縦に並べるだけで十分である．しかし，これでは三角形と四角形の関係が，意味の視点からも，図形構成の視点からも不明であって，コードを実行するまで，それが何を目的としたものであるかが理解されない（**図 1**）．

こうした設計を行う利点として，HCP チャートに書かれた文をそのままソースプログラムにコメントして利用できることがある．こうしたコメントをソースから抽出して HCP チャートを作ることができるし，逆にチャートか

© 2015 Information Processing Society of Japan

らソースプログラムのスケルトンを作成して，そこにコードを埋め込むこともできる．この方式を用いれば，ソースコードに設計文書を埋め込むことができる．設計と実現が1つの文書に統一できることになる（図 2）．

このような目的/手段展開を行うと，具体化が進んでいくに従って，上位の構造の設計に無理があることが見つかる場合がある．こうした場合には，上位の設計をやり直す必要がある．また，上位の目的が入れ代わると，下位の構造が影響を受けて，目的と手段が逆転する場合も出てくる．

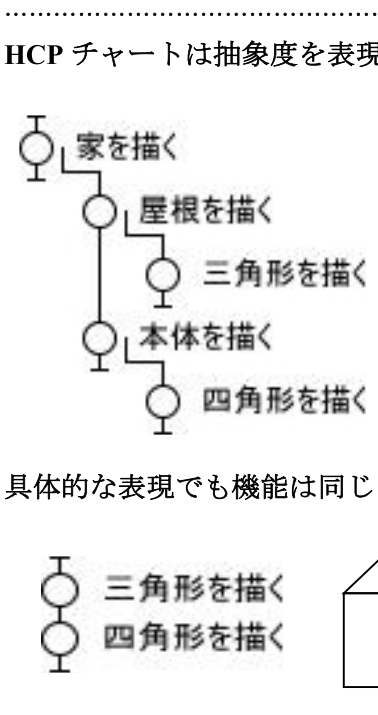

図 1　抽象構造を表示したチャートと具体的な処理だけを示すチャート
Fig. 1　HCP chart showing abstract structutre of house drawing and that of showing concrete drawing process.

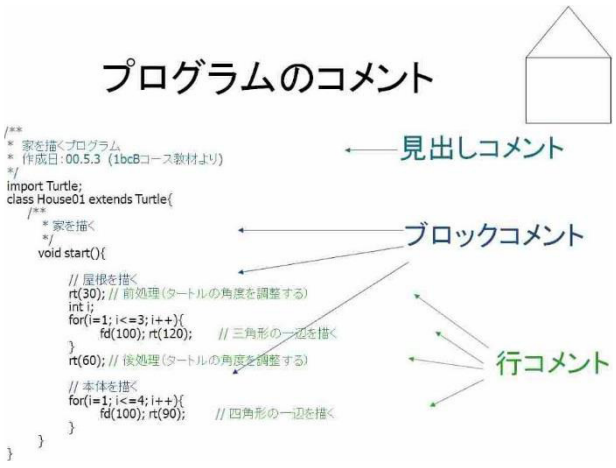

図 2　HCP チャーとから作ったプログラム
Fig. 2　A program coded from HCP chart.

大切なのは，上位の目的を設定することにあることが分かる．また，上位の目的を理解して，下位の目的を手段に展開しないと，良い設計にならない．

6.　日本語でプログラムを書くこと

設計ができたら，最終的に実行可能なプログラムを作り出すことになる．このコードをコンピュータで実行可能な日本語で記述することにすれば，プログラミング言語を学ぶ手間が省ける．最近では，何種類かの日本語プログラミング言語が発表されている．

その中の1つの「ことだま on Squeak」[7] の場合，コードはタイルを配置することで行われるために，文法エラーが起きない（図 3）．これによって，アルゴリズム構築に集中することができる．実際，情報の教員免許を持った教師が教えれば，文科系の学生が中心の大学のクラス全員が，挿入ソートをスクラッチから作れることが確認された [7]．

「ことだま」の特長は，タイルに書かれた文が日本語として完全に読めることである（図 4）．これによって，思いどおりに動かないプログラムを書いた学生に対しては，「声を出して自分のプログラムを読め」と指導することが可能

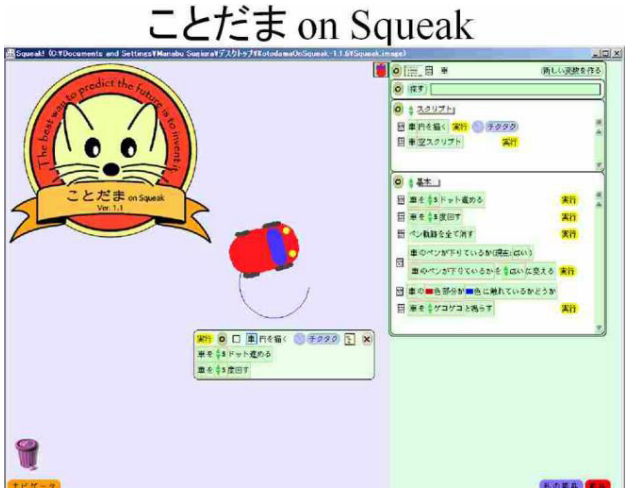

図 3　ことだま on Squeak の画面例
Fig. 3　A Screen shot of Kotodama on Squeak.

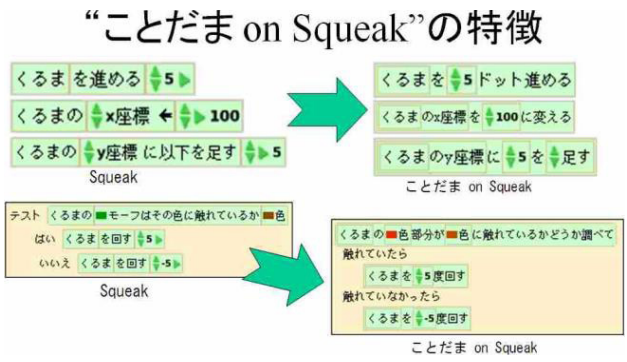

図 4　読める日本語で書かれる「ことだま」プログラム
Fig. 4　Kotodama program can be read as Japanese language.

© 2015 Information Processing Society of Japan

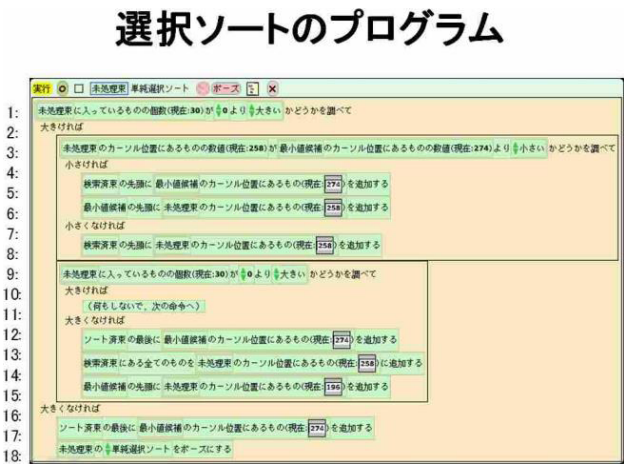

図 5　選択ソートの「ことだま」プログラム
Fig. 5　Kotodama program for selection sort.

になる．単に英単語を日本語に置き換えただけでは，このような指導は行えない．

挿入ソートを作らせるための例題として使用した選択ソートのプログラムを**図 5** に示す．

学習者は，自分の書いた日本語プログラムを読むことで，自分が書いたことが何であるかと理解して，自分が意図したこととの乖離を自分で発見することができる．このデバッグ作業こそ，人間が論理的ではないことを思いしらせ，コンピュータを利用することの本質的な困難を実感させてくれる．そのためには，プログラムは日本語として完全であることが必要で，単に日本語の単語を用いただけでは不十分である．

従来のプログラミング教育では，言語の使用方法を教えることに精一杯で，アルゴリズム構築には至らない場合がほとんどであった．言語教育は専門家には必要であっても，一般人には必要ではない [8]．

日本語の語順は目的語の後に動詞がくるため，複雑な処理を記述する場合に，記述者は目的語のスタックだけを脳内に用意しておけばよい，逆の語順の英語の場合は，動詞のスタックも必要となり，脳の負荷が大きくなる．このため，米国人が APL，FORTH，Post Script など，日本語の語順の言語を作っているが，母語の英語と語順が異なるため普及しなかった．日本語はコンピュータとのインタフェースがすぐれた言語なのである．日本人のために，日本語の語順のプログラミング言語を作る必要がある．

最近米国人のロジャー・パルパーズが，日本語は少数の語彙と単純な文法で豊かな表現ができることから，英語より世界語としての可能性が高いという指摘をしている [9]．少数の語彙と単純な文法はプログラミング言語の特徴であり，日本語は，プログラミング言語と同じような特性を持っていることになる．

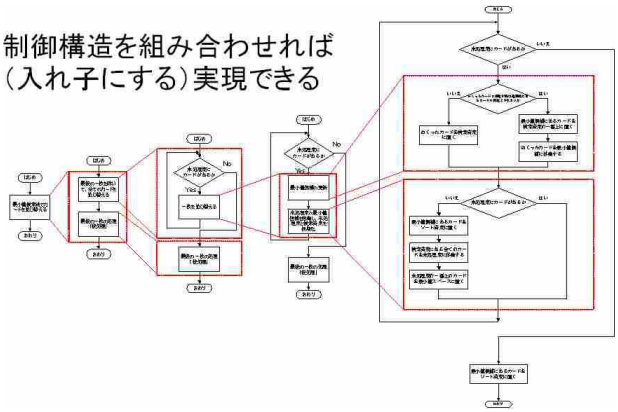

図 6　入れ子で実現した選択ソートの流れ図
Fig. 6　Flow chart of selection sort using nested structure.

7.　プログラミングで何をどこまで教えるか

識字教育はすべての市民を対象に行うものであるから，だれでも学べるものでなければならない．一方，コンピュータの本質を教えるという観点からは，やさしいものだけを教えるのでは意味がない．

まず必要となるのは，逐次処理，条件分岐処理，繰返し処理の 3 つの制御構造を使用できるようにすることである．これは，小学校段階で教えることは十分に可能である．しかし，これだけでは複雑な処理が可能になるという実感がわかない．

次に必要となるのは，複雑な処理を実現する方法論である．これには，上記 3 つの制御構造が入り口が 1 つ，出口も 1 つであることから，制御文の中に制御文を埋め込むことができるということを学ばせる必要がある（**図 6**）．

最近は，専門教育の中でもこのような本質的な方法論が教えられていないように見うけられるが，ここまではすべての市民が学ぶべきことであろう．中学校の技術・家庭では，プログラムによる計測・制御が取り上げられている．この内容を意味あるものにするためには，埋め込みによる複雑さの実現までをあつかう必要がある．

さらに，可能ならば手続きを用いた抽象化まで教えることが望ましい．これによって，情報環境が提供する情報処理リソースを使う場合に必要となる呼び出しの仕組みを理解できることになる．大学進学者に対しては必ず修得されなければならない事項であろう．

8.　数学教育との関係

従来の教育では，プログラミングのような論理思考は数学によって教育されてきた．しかし，最近学校で教えられる数学は，プログラミングで要求されるような構成的な能力を育成する部分がなくなってしまった．また，初等幾何学が教えられなくなったことから，正しいことを主張するための証明の概念がなくなってしまっている．情報化社会

© 2015 Information Processing Society of Japan

に必要な基本的な内容が失われてしまったのである．

一方，代数学や解析学に必要な数式処理の技術は，すべてコンピュータ上で行えるようになった．中学・高校で教えられる数学のほとんどの部分が，人間がする仕事ではなくなったのである．

しかし，このことは微分・積分のような概念を教える必要がなくなったということではない．必要になったのは概念理解だけでなく，必要なときにこれらの概念で現実を表現する能力である．数学教育は，こうした方向に重点を移す必要がある．

このような視点からの数学教育の見直しによって，プログラミング教育を行う時間が学校教育の中に生まれるはずである．また，日本語をプログラミング言語として採用することで，国語教育との関連も生じる．国語，数学という教科内容を見直して，指導要領全体の再構築を行うことが望まれる．

9. おわりに

ネットワーク化されたコンピュータが一般化した 21 世紀の社会では，すでに解決法が知られたことの多くはコンピュータで行われるようになり，人間ができる仕事はどんどん減ってきている．市民に求められるのは問題解決能力と，コンピュータにはできない仕事である．問題が生じたときに，問題となっていることは何であるかを具体的に同定する問題発見も必要となる．さらに，問題自体がしっかり把握されたとしても，正解があるとは限らない．この場合でも，良い解か悪い解かは判別できる．こうした正解のない問題に対処できる教育が現在の学校教育ではほとんどなくなってしまっている．

現在の日本の公教育は，工業化の時代に対応する市民のための教育であって，情報化が進む 21 世紀の市民の教育としては不適切な部分が多い．その結果が日本の経済力低下に反映されていると思われるのであるが，教育の対策として考えられていることは，成功した工業化社会への市民教育の強化であって，情報化への対応はほとんどされていない．

プログラミングは問題解決能力が要求される情報化社会における基礎教育として優れた内容を持ち，社会の基盤を構成する情報技術の本質を理解できるようにすることが可能である．しかし，現行のプログラミング教育はほとんどが言語教育で終わっていて，情報化に対応できる教育になっていない．本報告では 21 世紀の識字教育としてのプログラミング教育の理念のその方法を具体的に示した．

参考文献

[1] P.E. ドラッカー（著），上田惇夫ほか（訳）：ポスト資本主義社会—21 世紀の組織と人間はどう変るか，ダイヤモンド社 (1993).
[2] 子安増生：幼児にもわかるコンピュータ教育 LOGO プログラミングの学習，福村出版 (1987).
[3] UNESCO: Information and Communication Technology in Education - A Curriculum for Schools and Programme of Teacher Development (2002).
[4] 松澤芳昭，青山　希，杉浦　学，川村昌弘，大岩　元：「目的の表現」に注目したオブジェクト指向プログラミング教育とその評価，情報処理学会研究会報告（CE-27-11），pp.77–84 (2003).
[5] 杉浦　学，松澤芳昭，大岩　元：プログラミング教育における HCP チャートの再評価，情報処理学会第 46 回プログラミング・シンポジウム (2005).
[6] 日比野省三：トヨタの思考習慣，講談社＋ α 新書 (2005).
[7] 杉浦　学，松澤芳昭，岡田　健，大岩　元：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，情報処理学会論文誌，Vol.49, No.10, pp.3409–3427 (2008).
[8] ケビン・メイニー：C++ も JAVA も子供に教えなくていい，ニューズウィーク日本版 6 月 24 日号，p.48 (2014).
[9] ロジャー・パルパーズ（著），早川敦子（訳）：驚くべき日本語，集英社インターナショナル (2014).

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），東京大学理学部物理学科助手，豊橋技術科学大学講師，同助教授，教授，慶應義塾大学教授，帝京平成大学教授，相愛大学教授を経て，現在，慶應義塾大学名誉教授．研究分野は情報教育学，ソフトウェア工学，認知工学．

© 2015 Information Processing Society of Japan