Bulletin of Aichi Univ. of Education, 66（Educational Sciences）, pp. 157 − 161, March, 2017

小学校プログラミングの指導法に関する一考察

松永 豊

情報教育講座

A Study on the Teaching Method of Programming for Elementary School Students

Yutaka MATSUNAGA

Department of Information Sciences, Aichi University of Education, Kariya 448-8542, Japan

1. はじめに

近年，プログラミング教育の重要性が世界中で叫ばれている。米Google社の人工知能囲碁ソフト（アルファ碁）が世界トップ棋士を破ったという話題は記憶に新しいが，人工知能の驚異的な進化などにより第4次産業革命が到来するとまで言われている［1］。明らかにコンピュータは人間を凌駕する存在になりつつあり，様々な業種が機械（コンピュータ）に置き換わってしまう可能性が指摘されている。

この結果，（ちょうど逆向きとも言える）二種類の不安が発生している。一方は，本来，人間にしかできなかった職種がコンピュータに奪われることに対する不安であり，もう一方は，AI，IoT，自動運転，ドローンなど大幅な成長が期待できる分野での人材不足に対する不安である。どちらの不安に対しても解消のためには将来の職種選択における大胆なシフトを視野に入れなくてはいけない。すなわち，大規模な人材育成が急務となっている。

言うまでもなく，コンピュータを動作させるためにはハードウェアとソフトウェアの両方が必要であるが，最も一般的なプログラミング教育においては，

ハードウェア：パソコン（ノートパソコン含む）

ソフトウェア：コンパイラ，インタプリタ等

といった構成が主流である。プラットホームはOSに依存するため，選択肢は大雑把に以下のようなものになるだろう。

・Windows
・iOS
・macOS
・Linux
・Android

プログラミング言語は膨大な種類が存在するため様々な要因で選択することになるが，大学等でプログラミングを教える場合は実践的なプログラミング言語が用いられることが多い。種類別に考えれば，

・コンパイラ（JAVA，C++など）
・インタプリタ（BASICなど）
・スクリプト（PHP，Pythonなど）

の3つになるだろう。（実際には明確に区別できないケースもある）

しかしながら，これはあくまでも大学等でプログラミングを行う場合の環境である。理由はあとで述べるが，小学校プログラミングの場合，プラットフォームの選択肢はこれとは全く異なると言っても過言ではない。

そこで，本論文では大学でのプログラミング教育のノウハウを踏まえたうえで小学校プログラミング教育の指導法について議論を深めることを目的とする。

2. 教育すべき内容

筆者は大学でプログラミング教育を行っているが，そこでの主な環境は以下の通りである。

本学，愛知教育大学では学生全員が個人のノートPCを所有しているため（入学時に必携としてアナウンス），プラットフォームはそれを利用できる。当然，自宅学習等が可能となるので，授業内容もそれに合わせたものになっている。

卒業研究においては，研究人数にもよるが必要に応じてごく小規模なハードウェアを準備するだけで済むため（場合によっては1台），様々なプラットフォームとプログラミング言語の組み合わせが可能である。原則，個別指導となる卒業研究に関しては本来分けて考えるべきであるが，卒業研究時に初めて学ぶプログラミン言語も存在するため，議論が深まる可能性を考慮して敢えて挙げてある。

上述の通りプログラミング教育においては一定範囲

表1　大学におけるプログラミング指導環境

学部授業	★主なプラットフォームおよびOS ・学生個人が所有するWindowsノートPC ★プログラミング言語 ・PEN ・C++ ・C#
卒業研究	（卒業研究に関しては，過去の指導時における例を指している） ★プラットフォームおよびOS ・Windows ・Linux ・Android ・LEGO MindStorms ★プログラミング言語 ・C++ ・C# ・JAVA ・JavaScript（HTML，CSS） ・PHP ・C（組み込み用クロスコンパイル） ・LISP（Genetic Programmingの遺伝子として）

ハードウェアやOSの知識・操作スキル等が必要となる。大学で行うプログラミング教育においては，OS操作等は自学習をベースとして純粋にプログラミングのことだけを追求する授業構築もあり得るが，小学生に事前学習等を望むのは難しい。

教育したいことと，それ以前に学ばなければいけないこと踏まえると，以下のような段階を考える必要がある。

・スキル（skill）
・リテラシー（literacy）
・フルエンシー（fluency）
・コンピテンシー（competency）

プログラミング教育も当然のことながら，そもそもコンピュータに関連する教育を行う場合はこの段階別教育目標が非常に重要になる。プログラミング指導法の議論を深めるためにも，まず，情報教育で考えてみることにする。（それぞれの単語の前に「情報」を付けると分り易いかもしれない。）

スキルとは教養や訓練を通して獲得した能力のことである。例えば，最低限の専門用語やOS操作等の教育はスキル教育と考える事ができる。

リテラシーとは身に付けた一定範囲のスキルの組み合わせによって様々な問題解決が可能となる能力のことである。例えば，インターネットを用いて情報収集したのち，プレゼンテーションソフトを用いて発表スライドを作成する能力などを指す。この段階の教育はリテラシー教育と考える事ができる。

フルエンシーとは様々な問題解決の経験から本質を見極めて敷衍する能力のことである。例えば，ワープロソフトなど各種アプリケーションは，OSのバージョンやソフトのバージョンによって操作手順が異なることが多いが，本質を見極めた場合，単なる機械的な操作手順ではなく意味によって操作が可能となり，場合によっては一度も触ったことがない違うメーカーのソフトですら操作の類推が可能となる。教育的な観点からはこれは非常に重要であり，「上から何番目をクリックしてください※1」などとは本質レベルで違うと考える事ができる。（※1：百歩譲って演習時の指示としてはアリかもしれないが，上から何番目かを暗記させて試験に出す等は全く無意味である）

コンピテンシーとは直接的には「成果を生む望ましい行動特性」のことであるが，変化への適応力，人間関係の形成能力など，社会的に必要とされる個人が身につける力，単なる知識や技能をこえた能力のことを指すことが多い。

本来，プログラミング教育が目指すものは最低限リテラシー教育以降となるが，残念ながら，かなりの時間をスキル教育に奪われてしまう現実も存在する。

3. プログラミング開発環境

プログラミングを学ぶ前に習得しておかなければならないスキルは多いが，タイピングに関しては軽減する研究が広く行われている。具体的にはビジュアル開発環境が数多く開発されており，実際の初等中等プログラミングの教育現場やワークショップ等でも広く利用されている。よく使われるビジュアルプログラミング開発環境をいくつか紹介すると

・Scratch（スクラッチ）
・Viscuit（ビスケット）
・プログラミン
・blockly（ブロックリー）

などが挙げられる[2][3][4][5]。（詳細については5章参照）

もちろん，テキストベースのプログラミング言語においても入門向けを意識したソフトも数多くある。例えば，英語ベースのプログラミング言語への抵抗感軽減策として，日本語プログラミング言語などの研究も数多く行われている。本学，情報コースで用いられているPEN[6]もその一つであるが，その他，ドリトル[7]，なでしこ[8]などが有名であり，プログラミング教育現場でもすでに様々な形で利用されている。

しかしながら，小学生に対してはこれでもまだ難易度が高いのかもしれない。小学生を対象としたプログラミングの各種ワークショップ等が報告されているが，やり方に違いはあれども，概ね，参加者の感想として「プログラミングは楽しい」などの高評価を得られるようである。しかし不満意見が全くでないわけではなく，テキストベースのプログラミング言語を用いた場合に不評的な意見が出る可能性が高い代表的なも

のが「キーボード操作が苦手で遅れるためつまらない」という類である。本質的なプログラミングが楽しいと感じていることから考えても，スキル習得度のギャップ軽減が重要な対策であることは間違いない。

4. ハードウェアに関して

先述の通り，本学では学生全員がノートPC必携となっている。無論，学生はプログラミング授業の有無とは関係なく，様々な授業や就職活動等でノートPCを活用している。このような環境においてプログラミング教育を行う場合はノートPC上で動作する各種開発環境のことだけを考えればよく，比較的選択肢は広い。

しかしながら，小学生全員に対してのノートPC必携は現時点では現実的ではない。タブレットであればやや現実味もあるが，それでも最大のネックはコストである。小学校のPCルームを充実させる場合は学校にコスト負荷が圧し掛かることになる。

そこで最近では（ややむき出しの）組み込みコンピュータを使った研究も数多く報告されている。この背景には，プログラミング可能なICの急激な発展が寄与している。プログラミング可能なICとしてはPICやAVRなどが以前から有名であり，H8なども含め，ロボット研究などでは比較的多く使われてきた。ものによっては極めて安価な（数百円など）ICであるがプログラムを書き込むために専用の装置を用いる（あるいは自作する）など工学的な知識がかなり必要であったため一般的ではなかった。しかしながら，最近はICチップと入出力ポートを備えた基盤が安価で手に入るようになったため，大幅にしきいが下がった言える。

現在，比較的安価に手に入るものとしては

・Raspberry Pi（ラズベリーパイ）
・Arduino（アルデュイーノ）
・IchigoJam（イチゴジャム）
・mbed（エンベッド）
・Intel Edison（インテル エジソン）

などが挙げられる［9］［10］［11］［12］［13］。比較的安価（数千円以下など）なのでワークショップで用いる場合等は参加者に購入してもらい自宅学習も可能な授業構築をすることが多い。

また，このようなハードウェアを用いる場合，各種センサーやLEDとの連携が学習可能となるメリットが生ずることも大きい。小学校プログラミング教育の目的の一つとして身近な機器にもコンピュータが使われていることへの理解が挙げられる。例えば，自動販売機やロボット掃除機やエアコン等がコンピュータで制御されているなどの理解である。PCを用いたプログラミングの場合，キーボードやマウスからの入力，実行結果はディスプレイに表示させることが一般的なので，全体的にバーチャルなものになりがちである。そのため，リアルな機器とプログラムとの関係が理解しにくいかもしれない。しかしながら，センサーやLEDを直接接続して「光を遮ったらLEDが点滅する」などであれば恐らく直観的にも理解しやすいだろう。ブレッドボードを利用すれば半田付けの手間もかからないので本来の教育に集中できる。

なお，バーチャルからリアルへの変換だけを考慮するならLEGO MindStormsや低学年向けに開発されたLEGO WeDoなどを使う方法もある［14］［15］。カスタマイズ等は充実しているが少々高価なので大量に導入するにはコストがかかるかもしれない。

5. 様々なプログラミングツール

3章でも述べた通り，小学校プログラミングではビジュアル開発環境が用いられることが多いが，ここではそのいくつかの特徴を説明する。

★Scratch（スクラッチ）

MITメディアラボが開発したビジュアルプログラミング環境である。

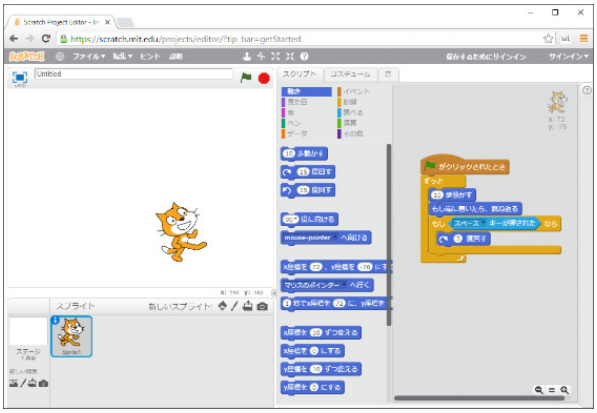

図1　Scratch

マウスだけでもプログラミングができると謳われている通り，命令パネルをマウスで移動してピースをはめ込むだけでプログラムを作ることができる。表示させるメッセージやスプライト名の変更などでキーボード入力することはあるが，命令文やコマンド名の直接打ち込み等は全く不要である。

また，Raspberry PiのGPIOコントロールや，Scratioを用いたArduinoとの連携なども可能なため，ハードウェア制御に使われることもある。

★Blockly（ブロックリー）
Googleが開発したビジュアルプログラミング環境である。

図2　Blockly

Scratch同様，命令パネルをマウス操作するだけでプログラムを作ることができる。公式サイトのBlockly Gamesではパズルや問題文などいくつかの学習用教材が揃っており，パズルがチュートリアルの役割を兼ねている。
また，Blocklyで作成したプログラムを既存のプログラミング言語（Pythonなど）に変換することも可能となっている。

★Viscuit（ビスケット）
コンピュータを粘土のように，をコンセプトに原田らが開発したビジュアルプログラミング環境である。

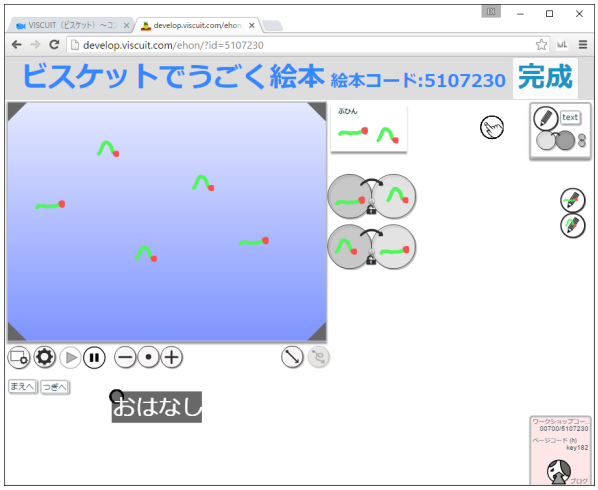

図3　Viscuit

メガネと呼ばれるルール設定が基本であり，アニメーションをさせるだけに仕組みに特化しているため，非常に低学年から使用できるという特徴を持っている。基本的にアニメーションだけなので「それはプログラミングか？」という問いかけもあるが，開発者はむしろ純粋な情報教育に使える可能性を述べている[16]。

★プログラミン
文部科学省が開発したビジュアルプログラミング環境である。

図4　プログラミン

公式ページに「ユーザーインターフェイス設計にはScratchを参考にした」と書かれているように，Scratch同様，命令パネルをはめ込むことでプログラミングができるようになっている。

6. 扱うテーマについて

小学校プログラミングにおいては，プログラミング言語を教えることが目的ではない。最終的に演習課題の教育目標到達度に関しては何らかの仕組みが必要になると考えられるが[17]，差し当たり，論理的思考力の向上や達成感などが目的になるであろう。重要なのは，子どもに楽しく学んでもらうことである。そのため，右や左に歩かせて目的地に到達させるプログラムなども教育の対象になり得る。その意味では，お菓子を命令とみなしたGLICODEなどもプログラミング教育素材と考えることができるだろう［18］。

★GLICODE
グリコが開発したスマフォ，タブレット用アプリであり，総務省が推進する平成28年度「若年層に対するプログラミング教育の普及推進」事業に選定されている。ポッキー，ビスコ，アーモンドチョコなどを配置し，スマフォ等のカメラで撮影すると命令を認識して行動する仕組みになっている。

図5　GLICODE

7. おわりに

本研究では小学校プログラミング教育の指導法について意見を述べた。現在，プログラミング教育の重要性は非常に高まっているが，小学校でプログラミングを行うことの賛否は分かれている。変な詰込み指導にならないよう気を付けながら，やはり，子どもにとって楽しいと感じることができる指導が重要と考えられる。

参考文献

［1］経済産業省，「今が日本の，第4次産業革命の分かれ道。」
［2］https://scratch.mit.edu/
［3］http://www.viscuit.com/
［4］http://www.mext.go.jp/programin/
［5］https://blockly-games.appspot.com/
［6］http://www.media.osaka-cu.ac.jp/PEN/
［7］http://dolittle.eplang.jp/
［8］http://nadesi.com/top/
［9］https://www.raspberrypi.org/
［10］https://www.arduino.cc/
［11］http://ichigojam.net/
［12］https://www.mbed.com/en/
［13］http://edison-lab.jp/
［14］http://www.lego.com/ja-jp/mindstorms
［15］https://education.lego.com/ja-jp/learn/elementary/wedo-2
［16］原田康徳，小学生に分かるコンピュータサイエンスとしてのプログラミング教育―ビスケットを用いて―，情報処理2016年04月号特集，2016
［17］松永 豊，プログラミング演習授業支援システムの開発，愛教大学研究報告59輯（教育科学編）2009
［18］http://cp.glico.jp/glicode/

（2016年9月23日受理）