Scratch を用いたプログラミング導入授業の現状と課題

齋藤　朗宏・池田　欽一・平山　克己・隈本　覚

１．はじめに

文科系学生に対するプログラミング導入授業は、理科系大学生に対するそれと比較して困難が多い。その理由について佐野（1998）では東京外語大を例にとり、「一般数学や数理論理学に関する講義がないために、演算子、論理演算子、条件式などのプログラミング用語の理解が難しいものと思われる」と指摘している。北九州市立大学経済学部経営情報学科（以下本学科）では、１年生の必修科目としてプログラミング言語 Java の授業を開講しているが、そこでは同様の問題が常に存在した。隈本他（2014）において既報の通り、本学科では、2013 年度入学者より、プログラミング入門のさらに前段階の授業として、情報科学入門という科目を開講し、プログラミング言語 Scratch を使用し、プログラミングの基礎となる考え方の習得を目指している。

本論文では、プログラミング導入授業における Scratch 導入の意図並びに効果を述べ、今後に向けての課題を検討する。

２．プログラミング導入教育と Scratch

Scratch は MIT メディアラボ[1]によって開発されたプログラミング導入教育のためのプログラミング言語であり、ソフトウェアである。Resnick et al.(2009）によると、同ソフトウェアの概要は以下の通りである。

Scratch は 2007 年５月に公開され、それまでプログラマーになることを想定していなかったような人々が、プログラミングを始める方法を開発することを目的としている。公式サイト[2]では、個人が作成したプログラム（プロジェクト）を公開可能であり、サイト使用者の中心的な年代は８歳から 16 歳である。

図１　100 × 3 の結果を出力するプログラム

1　http://www.media.mit.edu/

2　http://scratch.mit.edu/

具体的には、図１のように、プログラムの各要素をブロックに分け、そのブロックを組み合わせることでプログラムを作成する。こうすることで、より直感的なプログラム作成、また、プログラミングに対する理解が可能となる。

このような背景から、小中高校生向けのプログラミング教育に Scratch が使用されているケースは多い。森他（2011）では、小学４年生向けプログラミング授業をデザイン、実践している。その結果、26 時間の授業でほぼすべての生徒が繰り返しや条件分岐を理解し、プログラム内に導入することができた。

一方で、Scratch は、高等教育にも十分耐えられるソフトウェアでもある。森（2014）では、文科系大学生を対象とした授業に Scratch を導入している。全体の 95.5 %がプログラミングに初めて触れるという状態の大学生を対象に Scratch による入門教育を試みたところ、スプライト制御や繰り返しなどで高い理解度が示された一方で、変数に関しては高い理解は示されなかったという結果を得た。文科系のみならず、理科系学部においても導入例は見られ、嶋他（2013）では、電気電子工学系の１年生に対して導入教育として Scratch を使用したところ、発展的な学びへの高い期待が得られた。その他にも、鈴木（2014）では高等専門学校における制御工学の教材として使用できる可能性が示されている。

ブロックの組み合わせによる平易なプログラム作成という特性は、導入教育以外にも活かされている。宮田他（2013）では、Scratch のもう一つの重要な特徴である多言語対応も活かし、多国籍間でのオンライン・コラボレーションによる協同制作を試みている。

以上のように、Scratch は導入教育場面を中心に、様々な場面で採用されている優れたソフトウェアである。

３．情報科学入門と Scratch

本学科では、前述の通り情報科学入門という必修科目において、Scratch による導入教育を行っている。Scratch のメリットとして、日本語によるプログラミングが可能であることなど、敷居の低さが挙げられるが、本学科では、構造化プログラミングにおける３つの主要な構造である順接、分岐、反復が直感的に理解できることと、フローチャート（流れ図）との親和性の高さに特に注目している。

プログラミングの基礎を学ぶ上では、構造化プログラミングをベースとして、フローチャートを用いてその考え方、処理の流れを理解するのがわかりやすいが、特に分岐、反復はプログラミング以外の場面ではあまり見ることのない概念であり、単純に Java 等の言語を学習する、あるいはフローチャートの読み書きを学ぶだけで理解するのは困難である。

一方 Scratch では、図２、３に見られるように、ブロックから動作が直感的に想像しやすい。また、図４のプログラム例と図５の同じプログラムのフローチャートからわかる通り、プログラムとフローチャートとが極めて類似し、多くの場合 Scratch における１つのブロックとフローチャートにおける１つの記号とが対応している。そのため、このプログラムを動かすことで、分岐や反復などの動作を体験し、フローチャートの意味を考えることができる。

これらの点は、構造化プログラミングの考え方を理解する上で、大きな助けになると考えられる。尚、これら以外の情報科学入門の講義内容については、隈本他（2014）に詳述している。

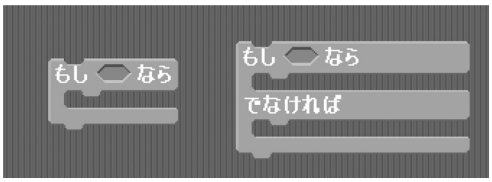

図２　分岐に関する主要なブロック

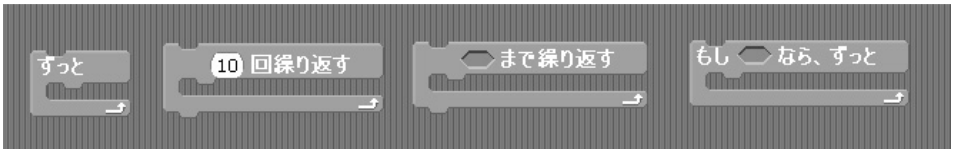

図３　反復に関する主要なブロック

図４　Scratch によるプログラム例

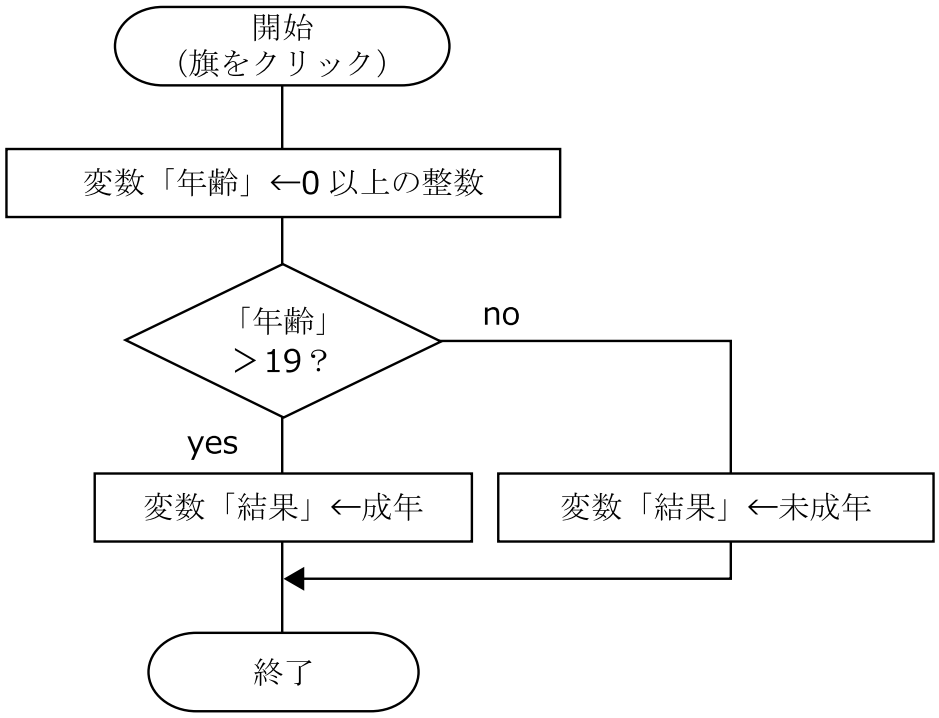

図５　図４と同内容のプログラムのフローチャート

４．情報科学入門理解度調査

４．１　調査実施概要

情報科学入門で目標としているプログラミングの基本的な考え方の理解が達成されているか、また、達成に不安のある点、講義の中でわかりにくかった点を確認するために、講義の履修者を対象として、情報科学入門理解度調査を実施した。実施の概要は以下の通りである。

- 調査日時：2014 年 7 月 18 日、25 日
- 調査対象：本学科の情報科学入門２クラス
- 回答者数：94 名
- 調査方法：Google ドライブのフォーム作成機能を用いたオンライン調査
- 回答方法：「全く理解できていない」～「よく理解できた」（問１～問７）、「作成できない」～「作成できる」（問８、９）の４件法
- 質問項目
  １．コンピュータの５大装置は理解できましたか？
  ２．２進数とはどういうものか理解できましたか？
  ３．２進数から 10 進数への変換、10 進数から２進数への変換は理解できましたか？
  ４．Scratch の変数、リストはどのようなものか理解できましたか？
  ５．下のような Scratch の繰り返しの使い方は理解できましたか？（アンケートフォーム

上では繰り返しの簡単なプログラムを示している）

6．下のような Scratch の条件分岐の使い方は理解できましたか？（問５に同じ）

7．フローチャートの書き方は理解できましたか？

8．Scratch で、以下のようなプログラムを作成できますか？

◇ 変数「点数」に０ら 100 の数値を保存しておいて、条件判断を使って、70 点未満の場合「不合格」、70 点以上 99 点以下で「合格」、100 点で「満点合格」とスプライト（キャラクター）に言わせるプログラム

9．Scratch で、以下のようなプログラムを作成できますか？

◇ リストに保存された金額（整数）を繰り返しを使ってすべて合計するプログラム。金額はリストにあらかじめ保存されていて、データ数は 10 個とする。

- 分析：記述統計量の確認を行った上で項目を整理し、因子分析を行った。因子分析には R x64 3.1.2（R Core Team, 2014）を使用した。尚、推定には最尤法、プロマックス回転を用いている。

４．２　調査結果

各項目に対する回答の平均値、標準偏差は表１の通りとなった。平均的な評価が 2.5 程度であることを考えると、まだ難しい部分があった可能性が高いが、反復と分岐に関しては３を超えており、この部分の理解度は十分に高いことが確認された。一方で、フローチャートの得点はかなり低くなっている点、また、実際にプログラムを作成可能かという質問に対しては、個々のブロックの理解度と比較すると自信のない回答がやや多かった点も確認された。変数・リストの問題については、評価としては高いものの、反復・分岐の理解度と比較すると低い結果となった。

表１　各項目への回答

	平　均	標準偏差
１：５大装置	2.766	0.966
２：２進数	2.755	0.969
３：２進数10進数変換	2.617	0.996
４：変数・リスト	2.968	0.796
５：反復	3.489	0.699
６：分岐	3.298	0.774
７：フローチャート	2.372	0.961
８：変数・分岐作成	2.777	1.028
９：リスト・反復作成	2.606	0.930

次に、項目間の相関係数を確認した。その結果は表２の通りである。尚、相関係数が 0.5 を上回った部分は太字にしてある。ここから、２進数の問題、２進数と 10 進数の変換問題、変数・リストと反復、分岐、実際のプログラム作成２つの相関が高い一方で、フローチャートに関しては他の項目との相関が低いことが確認された。

表２　項目間の相関係数

	問１	問２	問３	問４	問５	問６	問７	問８	問９
１：５大装置	**1.000**								
２：２進数	0.455	**1.000**							
３：２進数10進数変換	0.398	**0.860**	**1.000**						
４：変数・リスト	0.102	0.227	0.161	**1.000**					
５：反復	−0.004	0.179	0.179	0.473	**1.000**				
６：分岐	−0.050	0.156	0.150	**0.522**	**0.721**	**1.000**			
７：フローチャート	0.014	0.307	0.274	0.367	0.158	0.384	**1.000**		
８：変数・分岐作成	0.012	0.139	0.063	0.083	0.019	−0.024	0.031	**1.000**	
９：リスト・反復作成	0.040	0.202	0.114	0.114	0.068	0.015	0.142	**0.604**	**1.000**

これらの点を考慮し、また、固有値のスクリー基準、ガットマン基準も考慮すると、３因子構造が想定される。そこで、問７フローチャートを取り除き、３因子で探索的因子分析を行った結果は表３、４の通りとなった。尚、表３は因子負荷の絶対値が 0.1 を超えるもののみ掲載している。

表３　因子分析における因子負荷行列

	因子１	因子２	因子３
１：５大装置	0.489	−0.120	
２：２進数	0.988		
３：２進数10進数変換	0.867		
４：変数・リスト		0.556	
５：反復		0.792	
６：分岐		0.918	
８：変数・分岐作成			0.761
９：リスト・反復作成			0.795

表４　因子分析における因子間相関

	因子１	因子２	因子３
因子１	1.000		
因子２	−0.199	1.000	
因子３	−0.159	0.095	1.000

表３から、概ね想定通りの３因子構造であり、ほぼ単純構造となっていることが確認された。因子１に該当する３項目は情報科学の基礎知識的な項目群である。因子２にも問１は含まれているが、因子負荷はかなり低く無視しても問題ないと考えられ、残る３項目は Scratch に関する理解を示す項目群である。因子３には実際のプログラムの作成に関する２項目が含まれている。また、表４から、３つの因子の間での相関はほぼないことが確認できる。

４．３　考察

平均値の比較から、反復、分岐、変数・リストの順で Scratch の理解度が高いという結果になった。この結果は、森（2014）と概ね一致しており、Scratch を使ったプログラミング導入教育を採用した場合、反復に関する理解は容易である一方で、変数・リストの概念を理解させるのは若干困難があり、注意を要すると考えられる。

また、使い方の理解に比べ、実際のプログラム作成となると、平均値がやや下がっていることが確認できた。これは、何となくわかってはいるのだけれど、実際に問題を解決するのは困難と感じている状況であり、実用レベルとして見ると理解は十分と言えない学生が多いと考えられる。この点は、因子２と３の因子間相関がほぼない、即ち Scratch のブロックを理解出来ているかと、実際にプログラムを作成できるかとの間には相関がないことからもうかがい知れる。

フローチャートに関する理解の低さと、フローチャートの理解と Scratch や情報科学に関する基礎知識との相関の低さも重要である。これは、フローチャートについての理解ができておらず、さらに、Scratch とフローチャートとが学生にとって十分に繋がっていないことを示していると考えられる。

３つの因子間の相関がほぼない点は、情報科学に関する理解、Scratch に関する理解、Scratch でのプログラム作成能力の３つがお互いに影響を与えていないということを示しており、換言すると学生にとって、この３つは全く別の能力になってしまっているということである。

５．まとめと今後の課題

Scratch を用いたプログラミング導入授業について、プログラミング教育を取り巻く現状、導入の意図を述べた上で、アンケート結果から導入の効果、問題点を確認した。その結果として、Scratch を導入した大きな目的である、構造化プログラミングの理解については、問５、６の平均値の高さから、基本的なところでは達成されたと考えられる。一方で、もう一つの重要な目的であるフローチャートの理解については不十分であった。Scratch とフローチャートとが十分に繋がっていない点が問題である以上、この点を解決する、つまり、Scratch のプログラムとフローチャートとの相互変換により重点を置く講義を行うことで解決できると考えられる。また、この問題に限らず、情報科学入門における分野間の繋がりが十分に理解されていない点も今後の課題であろう。

個々のブロックや実際のプログラムと比較して、実際のプログラム作成の能力は十分ではないという点も、解決すべき問題である。この点に関しては、講義時間との兼ね合いもあるが、自由なプログラム作成の時間をなるべく多く取る、あるいは、自由課題を複数回課すことが必要であると考えられる。

参考文献

R Core Team.(2014).「R: A Language and Environment for Statistical Computing.」R Foundation for Statistical Computing.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver J., Silverman, B. & Kafai, Y.(2009).「Scratch: programming for all」『Communications of the ACM』52(11), 60-67.

隈本覚・池田欽一・齋藤朗宏・平山克己（2014）.「Scratch を用いたプログラミング教育の実践」『北九州市立大学商経論集』 49（3・4）, 23-27.

宮田義郎・杉浦学・亀井美穂子（2013）.「ワールドミュージアム：志を広げる多文化異年齢コラボレーション」『日本教育工学会論文誌』 37（3）, 299-308.

森秀樹・杉澤学・張海・前迫孝憲（2011）.「Scratch を用いた小学校プログラミング授業の実践：小学生を対象としたプログラミング教育の再考」『日本教育工学会論文誌』 34（4）, 387-394.

森秀樹（2014）.「Scratch を用いた文系大学生向けプログラミング教育」『日本教育工学会論文誌』 34（Suppl.）, 141-144.

嶋好博・伊庭健二・大矢博史・野澤昭雄・星野勉・宮村典秀（2013）.「PAD と Scratch を利用した構造化プログラミング基礎教育の取組：設計力と会話力を重視した明星大学電気電子工学系の体験教育」『工学教育研究講演会講演論文集』 61, 724-725.

鈴木聡（2014）.「視覚的プログラミング言語 Scratch を用いた PID 温度制御」『木更津工業高等専門学校紀要』 47,23-28.