新課程のプログラミング教育で押さえておきたいこと

千葉県立千葉中学校・千葉高等学校 主幹教諭
大橋 真也

1. この原稿について

本稿は，新学習指導要領の新科目として登場した「情報Ⅰ」および「情報Ⅱ」において，どのようにプログラミングを教えたらよいか，プログラミングに自信がないがどうしたらよいか，という先生向けにプログラミングへの入門の方法の一つとして書いている。ことプログラミングに関しては，一家言ある先生も多くいらっしゃることと思われるが，こんな考え方を持った人もいるのだなぁ，と笑って読んでいただければ幸いである。

私自身は，千葉県の千葉中学校の3年生の技術科でプログラミングを教え，千葉高等学校では，本年度は数学を教えているが，以前に「情報の科学」を教えていた。本稿では，自分のプログラミング指導の中で，気付いたことや学んだことについて書いていく。読んでも先生方のためになるような話はあまりないかもしれないが，気軽に読んでいただきたい。

2. プログラミングに入門するとき

プログラミング言語の文法や構文をテキスト等で学ぶのは退屈であると感じたことのある方は多いのではないだろうか。市販のプログラミング言語の入門書を1冊読んでも，自分でプログラミングできる気がしない。そんなことも多いのではないだろうか。私自身もこれまでに何度となくそんな経験を繰り返している。また，プログラミング言語を学ぶ際に同様のことを感じる生徒も多いことが考えられる。

本校の中学生にプログラミング言語 Processing を教えるときに留意していることは，毎回「完成物を作る」ということである。文法のパーツだけを説明しても明確には理解できない。また簡単すぎる例示だけでは応用ができないからである。本校では Processing を教えるとき，次のような内容で教えているが，毎時間に必ず完成物またはそれのバージョンアップ版を作れるように留意している。

◆ Processing の指導内容

1時間目　Processing について，形を作る
2時間目　色を作る，順次構造
3時間目　条件分岐
4時間目　反復構造
5時間目　パックマンを動かす
6時間目　複数のパックマンを動かす
7時間目　配列構造（1,000 匹のパックマン）
8時間目　ゲームを作ろう

これらは，Processing の作者でもある Casey Reas と Ben Fry の著書でもある「Processing をはじめよう 第2版」[1]を題材に指導内容を作っている。順次構造については，チョコレート色の長方形の上に桜色の円を重ねたものを作成させることにより，プログラムの順序，色・形，座標について学ばせている。この際に色や大きさに関しては，規定していない。つまり正解のプログラムを見せずに，結果のみ見せて自分なりにアレンジしたものを作らせるのである。桜色やチョコレート色については，標準のものがあるが，この授業の中では，隣の人と相互に評価し合って，他者が認めるような色や形を作らせる。そのような自由度のあるアレンジができることで，自分で考えるプログラミングになっていくと考えている。

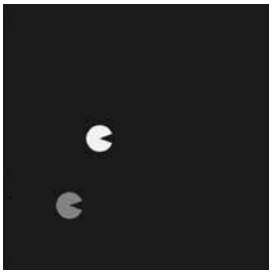

図1　④の2匹のパックマン

上記の5～7時間目に作成させるパックマンについては，次のような手順で，作るごとに新たな内容を学びながらバージョンアップさせていく。

①扇型でパックマンを作り，たどたどしく動かすために random 関数を使う。
②画面の右側に消えたパックマンを左から登場させる。（条件分岐）
③パックマンの口をパクパクさせる。（条件分岐 2）
④パックマンを 2 匹にして競走させる。（2 匹は独立させる，図 1）
⑤パックマンを 1,000 匹にする。（配列構造）

そもそも中学生に Processing を学ばせようと考えたのは，Java や Python のプログラミングスタイルを学ばせることができることと，実際にプログラミングした結果が，画像として表示され，自分の考えた処理が正しく行われているのかを確認しやすいことからである。

高校生の情報科に関しても同様である。プログラミング言語の候補として Python が注目されているが，「高等学校『情報』実践事例集」[2]に例として掲載した Life Game のプログラミングなど，ある程度の完成物を通じて文法を学ぶ指導内容をもとに実践している。あくまでも個人的な感想であるが，授業を実践するにあたり，Python に関するさまざまな入門書を読んでみたが，「Head First はじめてのプログラミング－頭とからだで覚える Python プログラミング入門」[3]が最もよくできていると感じた。これから Python を学ぼうとしている先生方にもお勧めの書であると考えている。

3. データサイエンスもプログラミング？

新課程の「情報Ⅰ」，「情報Ⅱ」で新たに導入されるのは，プログラミングだけではない。情報デザインやデータサイエンスも新たな指導内容である。情報デザインは，PowerPoint などで単にスライドを作らせるだけでなく，より効果的に可視化することなども求められている。また，データの可視化や文書構造の可視化なども求められている。データサイエンスと合わせて，データの可視化に関しては，Excel などの表計算ソフトウェアだけではなく，プログラミング言語を使用してみるのも一考である。Data Visualization という分野も情報デザインやデータサイエンスに関連している。それらの図がどのようなソフトウェアで作成されているのか，調べてみることも大切である。もとより，Excel などの表計算ソフトウェアでは，データサイエンスの内容のほんの一部にしか対応することができない。そのため，可視化に特化したソフトウェアやデータサイエンスに特化したプログラミング言語を選択してみることも必要である。複数のプログラミング言語を教える準備があるならば，「プログラミング言語は目的別に選択する」ということも学ばせるとよいだろう。

Python でこれらをすべて網羅することも可能ではあるが，データサイエンスに関してのお勧めのプログラミング言語は，R であろう。Python よりも長いプログラムを作らなくとも効果的なデータサイエンスのプログラミングができるのが特徴である。データサイエンスというと，数学や統計学の知識がないと指導できないとお考えかもしれないが，それほど心配する必要もない。ある程度は数学科の履修内容と連携させた上で，直接 R を学ばせることも大切である。統計学の知識なしに純粋に言語として R を学ぼうとする先生方にお勧めの本は，「R Studio ではじめる R プログラミング入門」[4]である。この本は，「サイコロを作る」，「トランプを作る」，「スロットマシーンを作る」という 3 部からなる完成物を作りながら学べる本である。これである程度 R に親しむことができたならば，データサイエンスを扱った R の本を自由に読むことができるようになるだろう。同じ処理でも，R の方が Python より短いプログラムで記述できるのは，ライブラリの充実と，そのリスト構造や関数型プログラミングが利用可能なためである。長いプログラムを指導するのが，生徒のタイピング速度等の関係で難しい場合は，関数型プログラミング言語を検討することも重要かもしれない。文部科学省が公開している「情報Ⅰ」および「情報Ⅱ」の教員研修用教材[5][6]では，データサイエンスの単元の多くについて R でプログラミングしているので，こちらも是非活用していただきたい。

4. 自己流を教えるのではなく…

先生方の中には，プログラミング言語は自己流で学び，それを教えている方も多いかもしれない。誤解を避けるために言っておくが，自己流が悪いと言いたいのではない。標準のコーディング規約が存在することを生徒にも知らせてほしいのである。Python には，PEP8 と呼ばれるコーディング規約が存在するのはご存知だろうか[7]。PEP8 は，Python

の開発者である Guido van Rossum の書いたものに Barry Warsaw が追記したものからなり，日本語訳などもインターネットで検索すればすぐに見つかるので一読してほしい。Guido 自身も「一貫性にこだわりすぎるのは，狭い心の現れである」と言っているように，これだけに執着する必要はないと考えるが，プログラムの可読性を増すためには，コーディングのスタイルを教えることも重要かもしれない。プログラムの可読性をよくすることは，生徒の指導をする上でも大切なことであるが，それをさらに生徒自身が改善する場合やそれらのプログラムを評価する際には重要であろう。他にも Google Python Style Guide というコーディング規約もあるが，興味のある方はご覧いただきたい。

R については，PEP8 にあたるものはないのか，とよく質問を受けるが，現時点では R の神様 Hadley Wickham の tidyverse に従うのがよいかもしれない。tidyverse は，正確に言えばコーディング規約ではないが，PEP8 などを参考にインデント等については従い，tidyverse にある dplyr などで提供される pipe(%>%)や tibble などのデータ構造を活用して整理することで，プログラムの可読性だけでなく，実行速度なども期待できる。

コーディング規約のようなプログラムの書式だけでなく，プログラム自体の内容やアルゴリズムも重要である。「プログラムは動けばよい」という考えではなく，基本的な形や考え方を教えることのできるプログラムを作成することは大切である。「教えるためのプログラム」は，市販の入門書には載っていないが，これからの情報科のプログラミング教育を進めていく中で必要な考え方となるだろう。もともとこの新課程でプログラミングが導入され，小学校からのプログラミング教育が始まった経緯は，ご存知のことと思うが，各国のプログラミング教育の影響でもあり，また発達段階に応じた「プログラミング的思考」を育成するための手段でもあった。プログラミング的思考とは，「自分が意図する一連の活動を実現するために，どのような動きの組合せが必要であり，一つ一つの動きに対応した記号をどのように組み合わせたらいいのか，記号の組合せをどのように改善していけば，より意図した活動に近づくのか，といったことを論理的に考えていく力」と文部科学省では定義している。またみなさんもご存知の通り，この「プログラミング的思考」という言葉は，Jeannette Wing の Computational Thinking（日本語訳あり）から来ているといわれている[8]。この中で「プログラミング的思考」は，すべての人に必要であり，問題を解決するための手法であるとされている。単に「プログラムが動いたね」で終わるのではなく，それを改良し，修正することによって，生徒がより最適な問題解決ができるようにするとともに，生徒のプログラミング的思考を育成できるようにするための基礎となるプログラムを教えていく必要があると感じている。先にあげた実践事例集の中で紹介している Life Game のプログラムについては，指導した後に数人の生徒は，「総合的な探究の時間」の自身の研究でそれを応用し，「セル・オートマトンによる車の渋滞のシミュレーション」をはじめとする自分の興味・関心のあるテーマへのさまざまな活用を行っていた。すなわち，情報で学んだプログラミングが活用され，自分の研究に結びついたのである。

その一方で PEP8 には違反する部分もあるが，少し大きなデータを用いてその処理速度を競ってみたり，ワンラインプログラムでどこまで技巧的なグラフィックス作品ができるかを追求させたりすることもプログラミングを学ぶ上では，面白いテーマであろう。教科書にはさまざまなソートの例などが例として書かれているが，実際にバブルソートと挿入ソートではどちらが速いのだろうか。少し大きなデータをもとにソートしてその処理時間を比較してもよいだろう。またモンテカルロ法による円周率の計算などのよくある例に関しても，どのようにしたらより円周率計算の精度を上げることができるのかを追求してみてもよいだろう。計算速度や計算効率も「情報Ⅱ」においては，必要な考え方である。

データサイエンスでは，ビッグデータを扱うことも必要である。処理が授業時間内に終わらないようなデータを使うことは難しいが，ある程度の大きさのデータを使い，その処理速度がアルゴリズムや構文の工夫でどの程度速くなるのかを追求していくことは面白い。例えば，R で行うならば，リスト構造をうまく使うとか，ベクトル化された関数を使うなどの工夫で簡単に数百倍の高速化ができる。遅く，汚い（可読性の悪い）プログラムは，だれしも好まないだろう。だからといって速度だけを追求してみると可読性の悪いプログラムができてしまうこともあ

る。実行(処理)速度と可読性のバランスのとれたプログラミングを教えることができるということが，学校教育の中では大切であると考える。

ここまで書くと，そんな難しいこと教えられる訳はないと，非難されそうであるが，大切なことは，生徒にとっても可読性の高い，そして品質の良いプログラムを生徒に提供するということである。数万行のコーディングを行うのとは大きく異なり，数十行で十分に機能するようなプログラムを作るということである。これは，学校の先生だからこそできることではないだろうか。

Processing の指導例の 8 時間目にあげたゲームを作らせるプログラムの基本的な部分は，ボールとラケットがあるスカッシュゲームのようなものである(図 2)。これは，20 行程度で書いている。そして可

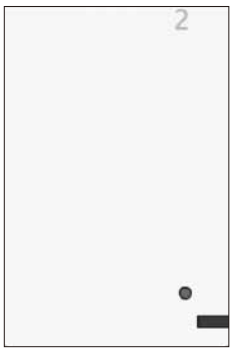

図 2　スカッシュゲーム

読性が高いと，生徒がそれをアレンジすることも可能になる。自分でアレンジすることによって，さまざまなプログラミングの構文を覚えていく。そんな授業を期待しているのである。

中学生のプログラミングの授業は楽しい。自分だけで解決がつかないことは，生徒同士が教え合いながら解決することができているからである。高校生では問題を自己解決させる力を育成することは重要であると考えているが，ときにはペアプログラミングなどの手法をとると，理解が深まることがある。

5. 大学入学共通テストの対策

令和 3 年度より始まった大学入学共通テストに令和 7 年度から新たに「情報Ⅰ」が加わるという発表を聞いて驚かれた方も多いだろう。実際に大学入試センターのサイトには，そのサンプル問題が掲載されている[9]。現時点では，時間や配点等を設定した問題とはなっていないが，プログラミングの問題に関して，見慣れないプログラミング言語が使用されていることが注目されている。令和 3 年度の共通テストの「情報関係基礎」の第 3 問に使われている言語とも異なる。「情報関係基礎」で使用されているプログラミング言語は，仮想言語である DNCL であるが，「情報Ⅰ」のサンプル問題第 2 問に使用されているプログラミング言語は，その後継にあたる DNCL 2.0(仮称)である。日本語によるプログラミングであるが，インデントや書き方を見ていると Python に似ているように感じるが，気のせいだろうか。サンプル問題では，比例代表制選挙で用いられているドント方式をプログラミングしている。配列の使い方などが理解できていれば，Python 以外の言語を学んでいる生徒でも十分に解答可能な内容であると考える。またサンプル問題の第 3 問は，データサイエンスの問題であるが，ここで登場する散布図相関行列を表計算ソフトウェアで作成することは大変である。先にあげた Python や R では数行のプログラムで作成可能なものであるが，このようなグラフの読み取りだけでなく，作成できるような指導も授業の中では必要なのではないだろうか。

参考文献

1)Casey Reas,Ben Fry 著，『Processing をはじめよう 第 2 版』，オライリージャパン，2016 年
2)文部科学省，「高等学校『情報』実践事例集」，2021 年，https://www.mext.go.jp/a_menu/shotou/zyouhou/detail/mext_01342.html
3)Eric Freeman 著，『Head First はじめてのプログラミング－頭とからだで覚える Python プログラミング入門』，オライリージャパン，2019 年
4)Garrett Grolemund 著，『R Studio ではじめる R プログラミング入門』，オライリージャパン，2015 年
5)文部科学省，「高等学校情報科『情報Ⅰ』教員研修用教材」，https://www.mext.go.jp/a_menu/shotou/zyouhou/detail/1416756.htm
6)文部科学省，「高等学校情報科『情報Ⅱ』教員研修用教材」，https://www.mext.go.jp/a_menu/shotou/zyouhou/detail/mext_00742.html
7)「PEP8」，https://pep8-ja.readthedocs.io/ja/latest/
8)Jeannette M. Wing 著，中島秀之訳，「Computational Thinking(計算論的思考)」，https://www.cs.cmu.edu/afs/cs/usr/wing/www/ct-japanese.pdf
9)大学入試センター，「令和 7 年度大学入学共通テストからの出題教科・科目について - サンプル問題『情報』」，https://www.dnc.ac.jp/kyotsu/shiken_jouhou/r7ikou.html
（ウェブページはすべて 2021 年 7 月 13 日アクセス）