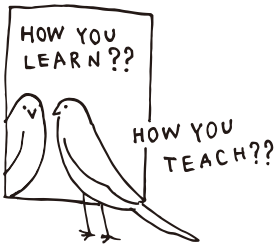
べた語義

Vol. 102

CONTENTS

COLUMN

教え方と学び方を学ぶ



基般

筆者は，教育工学を専門とし，インストラクショナルデザイン（ID）という領域を中心に活動している．ID の研究分野では，教育の効果・効率・魅力を高めるための，教科領域によらない方法論を主に蓄積してきた．ID の内容を教職課程の科目で取り上げるうちに，教職を志望する学生だけではなく，より多くの学生にも広げるべきではないかと考えるようになった．誰にでも他者に教える機会は存在し，学業，仕事，アルバイト，家庭，サークルなど多様である．そのような日常にありふれた，人に教えるという営みが，本人の経験や勘に基づいている場合が多く，その質が高いとは言いがたい．これは，学び方についても同様である．高等教育でよく取り上げられるアクティブラーニングは，学生にとって学び方を学ぶ機会でもあるが，逆に自ら学べる学生でないとその効果を十分に享受できない．

教え方と学び方は表裏一体の関係にあり，ID の理論やモデルを自らの学びに適用すれば，学び方の改善にもつながる[1]．たとえば，ARCS という動機づけのデザインモデルは，教員が学習者の意欲を高めるためだけでなく，学習者自身が学習意欲を維持するために利用できる．筆者は ID を扱う教職の授業において，学生自身の学び方についても考えさせるようにしている．また，学部の 1 年生全員を対象としたスタディスキル科目においても，普段の教えあいや学びあいの質向上をねらい，ID の考え方に触れる機会を提供してきた[2]．

複雑で不確実性の高い社会を生き抜くための能力として，国際団体 ATC21s による 21 世紀型スキルには学び方の学習が挙げられており，OECD Learning Compass 2030 には見通し・行動・振り返りをしながら絶えず学習していく存在が描かれている．新学習指導要領では学びに向かう力（自己調整学習を含む）が柱の 1 つになっている．今後，適応型学習などテクノロジーを活用した手厚い支援も可能となってくる中で，学び方を学ぶ機会や環境をどのように整えていくのかは重要な課題となる．学生には経験だけでなく科学的な根拠も踏まえながら，自分の学びや人の学びを支援できるようになってほしいと思う．

参考文献

1) 鈴木克明，美馬のゆり（編著）：学習設計マニュアル：「おとな」になるためのインストラクショナルデザイン，北大路書房，京都（2018）．
2) 市川　尚，鈴木克明：インストラクショナルデザイン理論を学ぶスタディスキル科目の実践，日本教育工学会研究報告集，JSET14-5, pp.127-130 (2014).

市川 尚（岩手県立大学・ソフトウェア情報学部）

LOGOTYPE DESIGN...Megumi Nakata, ILLUSTRATION&PAGE LAYOUT DESIGN...Miyu Kuno

ARTICLE

Processing でプログラミングに挑戦！ —第 2 回 変数を使ってみよう—

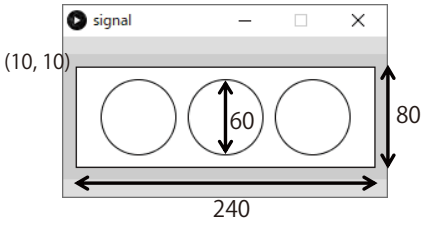
杉浦　学
鎌倉女子大学

宿題の解説

前号では，図形を組み合わせて信号機（図 -1）を描く宿題を出題しました．

● 前号の宿題

お手本を参考に，信号機を描いてみましょう．今回はモノクロですが，次号で色を付けていく予定です．



画面のサイズは横260，縦100
円の中心は左から(60, 50)(130, 50)(200, 50)

図 -1　信号機のお手本

この宿題の解答例をスケッチ 1 に示します．信号機の部品は 4 つ（枠 1 つと灯火 3 つ）なので，ウィンドウの大きさを指定する size 関数と合わせ，5 つの命令があれば十分です．解答例では，5 つの命令を連続で書くだけでなく，コメントと空行を使って全体を 3 つに分割し，それぞれの部分の役割を分かりやすくしてみました．スケッチの最初の行に /* 信号機を描くスケッチ */ のようなタイトルをコメントとして書いておくのもよいでしょう．

練習のための短いスケッチであっても，コメントや空行を使って分かりやすく書いておくことは重要です．しばらくたってから自分が書いたスケッチを見返して，自分で改造する場合にも役立つからです．

```
//描画の準備
size(260,100);

//枠を描く
rect(10,10,240,80);

//3つの灯火を描く
ellipse(60,50,60,60);
ellipse(130,50,60,60);
ellipse(200,50,60,60);
```

スケッチ 1　信号機を描く（宿題の解答例）

塗りつぶし

宿題の信号機に色を付けてみましょう．fill 関数を使うと，図形の塗りつぶしの色を指定することができます．色の指定は赤（Red），緑（Green），青（Blue）の 3 色（RGB と呼ぶことがあります）のそれぞれが，どの程度含まれているかを 0 から 255 の数値で指定します．色の指定には，灰色の明るさを 1 つの数値で表現したグレースケールを使うこともできます．この場合は，0 が黒で 255 が白になり，128 は白と黒の中間の灰色となります．fill 関数で塗りつぶしの色を一度指定すると，その後で描いた図形はすべて指定された色で塗りつぶされることに注意する必要があります．

この fill 関数を使って，宿題の信号機に色を付けたものを図 -2 とスケッチ 2 に示します．色の指定のための命令を追加したので，スケッチ 1 の状態か

らコメントや空行の入れ方なども変更してあります．

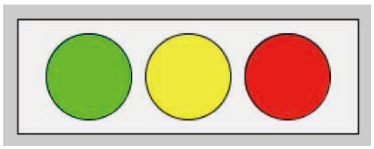


図 -2　色をつけた信号機

```
//描画の準備
size(260,100);

//薄い灰色の枠を描く
fill(240);
rect(10,10,240,80);

//青信号(実際は緑色)の灯火を描く
fill(0,255,0);
ellipse(60,50,60,60);

//黄信号の灯火を描く
fill(255,255,0);
ellipse(130,50,60,60);

//赤信号の灯火を描く
fill(255,0,0);
ellipse(200,50,60,60);
```

スケッチ 2　色をつけた信号機を描く

スケッチの中で使う RGB の値を確認するには，エディタのツールメニューから「色選択」をクリックして表示されるカラーパレットが便利です（図 -3）．パレットを調整して好きな色を選択すると，RGB の値を確認することができます．

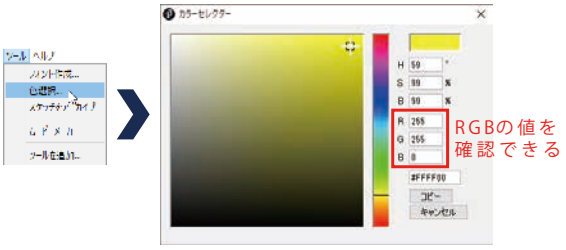


図 -3　色選択からカラーパレットが利用できる

なお，塗りつぶしをなくしたい場合は，noFill 関数を使います．この noFill 関数のような命令は，動作を指定する数値や文字が不要なため，パラメータを指定するカッコの中には何も書きません．

事前に手順を整理する

単純なスケッチであれば，一息に最初から最後まで書いてしまうのも大変ではありません．しかし，スケッチが複雑になってくると，「何をどのような順番ですべきか」をあらかじめ整理しておく方がうまくいきます．1 つの方法として，大まかな手順を最初にコメントとして記述し，その後で具体的な命令をそのコメントの下に書いていくという方法が有効です．スケッチ 2 を作りはじめる前に，スケッチ 3 のように手順をコメントで整理したスケッチを用意します（この時点でスケッチを実行しても何も図形は描かれません）．手順が整理できたところで，それぞれのコメントの下の行に具体的な Processing の命令を順番に追加していきます．命令を少し追加したらこまめに実行をし，結果が意図どおりかを確かめながら進めるとよいでしょう．こうした手順を踏むことで，できあがったスケッチは自然と構造が整理されて，適切なコメントが挿入されたものになります．

```
//描画の準備

//薄い灰色の枠を描く

//青信号(実際は緑色)の灯火を描く

//黄信号の灯火を描く

//赤信号の灯火を描く
```

スケッチ 3　スケッチ 2 の下書き

手順を事前に考えて整理しておく訓練は，プログラミングだけでなく「試験勉強やイベントの計画を立てる」「文章の章立てを考える」「慣れていない仕事の作業効率を改善する」といったさまざまなことに応用ができるでしょう．

輪郭線

図形の輪郭などの線の太さの初期値は 1 ピクセルに設定されていますが，これを変更するには strokeWeight 関数を使います．カッコの中に指定するパラメータは 1 つで，太さを数値で指定します．

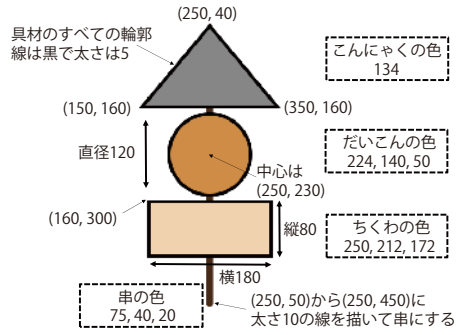
輪郭線の色を指定するには，stroke 関数を使います．fill 関数と同じように，カッコの中には RGB の 3 つの数値か，グレースケールの場合は 0 から 255 の数値を 1 つ指定します．

輪郭線をなしにしたい場合は，noStroke 関数を使います．noFill 関数と同じようにパラメータを指定するカッコの中には何も書きません．

❑ 練習問題

以下のお手本を参考に「おでん」のイラストを描いてみましょう．スケッチ 3 のように，最初にコメントで串や具材などの部品を描く流れを日本語で整理してから始めましょう．

ウィンドウの大きさは縦横 500 にして，背景を白で塗りつぶします．背景の色を設定するには，background 関数を使います．串はこれまでに解説した，strokeWeight 関数，stroke 関数を使います．基本的な図形の描画については，前号で解説しました．



＜解説＞

おでんを描くスケッチの例

```
//描画の準備
size(500,500);
background(255);//背景は白

//串を描く
strokeWeight(10);//太さは10
stroke(75,40,20);//色は茶色
line(250,50,250,450);

//具材に共通の輪郭線を設定
strokeWeight(5);//太さは5
stroke(0);//色は黒

//こんにゃくを描く
fill(134);//色は灰色
triangle(250,40,350,160,150,160);

//だいこんを描く
fill(224,140,50);//色は濃い茶色
ellipse(250,230,120,120);

//ちくわを描く
fill(250,212,172);//色は薄い茶色
rect(160,300,180,80);
```

変数の宣言・代入・参照

信号機のデザインを図 -4 のように変更したい場合について，考えてみることにしましょう．

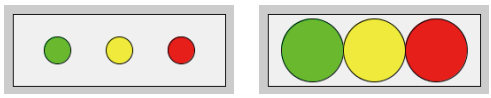


図 -4　信号機のデザイン変更

スケッチ 2 の内容をよく検討してみると，灯火の直径を表す 60 という数字が 6 回書かれています．灯火の大きさを調整したい場合は，青黄赤のそれぞれの灯火を描いている ellipse 関数のパラメータを変更する必要があります．具体的な変更個所は 6 カ所になります．このような変更の手間を減らす 1 つの方法が，「変数」を使うことです．

スケッチの中で使う値をコンピュータのメモリに保存しておき，スケッチの中で利用できるようにするのが変数の役割です．変数には分かりやすい名前をつけておきます．灯火の直径を d という変数（直径 diameter の頭文字）に保存して利用するスケッチの例を見てみましょう．今回のスケッチから左に行番号を付けてあります．

```
1  //描画の準備
2  size(260,100);
3
4  //灯火の直径 d(diameter)
5  int d;
6  d = 60;
7
8  //薄い灰色の枠を描く
9  fill(240);
10 rect(10,10,240,80);
11
12 //青信号(実際は緑)の灯火を描く
13 fill(0,255,0);
14 ellipse(60,50,d,d);
15
16 //黄信号の灯火を描く
17 fill(255,255,0);
18 ellipse(130,50,d,d);
19
20 //赤信号の灯火を描く
21 fill(255,0,0);
22 ellipse(200,50,d,d);
```

スケッチ 4　変数を使った信号機

変数を使う場合は最初に 1 回だけ「宣言」をする必要があります（「定義」という場合もあります）．スケッチ 4 の 5 行目で，整数を保存するための変数 d を宣言しています．宣言の際には，変数に保存できるデータの種類である「データ型」を指定します．Processing で利用できるデータ型はたくさんありますが，入門の段階では，int 型（整数）と float 型（小数点のついた数）の 2 つが使えるようになれば十分です．宣言の際にはデータ型（この場合は int）を先頭に書き，半角のスペースを挿入したあとに変数の名前（この場合は d）を書きます．他の命令と同じように最後のセミコロンも忘れないようにしましょう．

変数に値を保存することを，「代入」と呼びます．6 行目では変数 d に 60 という値を代入しています．スケッチ 4 では宣言と代入を別の行で行っていますが，`int d = 60;` のように，宣言と代入を同時に行うこともできます．

変数の値を読み出すことを，変数を「参照」すると言います．14・18・22 行目では具体的な数値をパラメータに指定せず，d に代入してある変数の値（スケッチ 4 の例では 60）を読み出し，その値を直径にした円（灯火）を描いています．具体的な値の代わりに変数の名前（この場合は d）を書けばよいということです．

スケッチ 4 のようにしておけば，3 つの灯火の直径を変更する場合は，6 行目で d に代入している数値の 1 カ所を変更するだけで済むことになります．実際に 60 の数値を変更して，図 -4 のようなデザインの調整が手間なくできることを確認してみましょう．

変数と計算

Processing で使える変数は，スケッチ 4 の変数 d のように自分で宣言をしたものに加えて，宣言をしなくても使えるものがあります．その代表例がウィンドウの幅と高さが保存されている width と height です．スケッチの最初に size 関数で指定したウィンドウの幅と高さはこれらの変数に保存されて，スケッチの中で参照することができます．

スケッチ5ではウィンドウのサイズをsize関数でどのように指定しても，四隅を結んで×印を描くことができます(図-5)．2行目のline関数では，ウィンドウの左上の角(0, 0)から右下の角(480, 120)に直線を引きます．3行目では，ウィンドウの右上の角(480, 0)から左下の角(0, 120)に直線を引きます．ここで示した座標はウィンドウの横幅が480，高さが120の場合のものですが，直線の始点と終点の座標はウィンドウの大きさによって変わってきます．具体的な座標を書かずに，変数widthと変数heightを使うことで，さまざまな場合に対応できるスケッチになります．

```
1 size(480,120);
2 line(0,0,width,height);//左上から右下へ
3 line(width,0,0,height);//右上から左下へ
```

スケッチ5　ウィンドウのサイズに応じて×印を描く

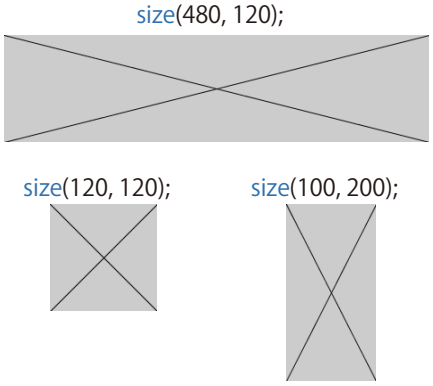


図-5　ウィンドウのサイズと連動する×印

スケッチ5の最後に新しい命令を加えたものがスケッチ6です．実際にスケッチを書いて結果を試してください．4行目の命令にある / は割り算(普段は÷と書くことが多いと思います)を指示する「演算子」と呼ばれる記号です．width/2でウィンドウの横方向の中心を，width/2でウィンドウの縦方向の中心を計算し，それを円の中心の座標としています．

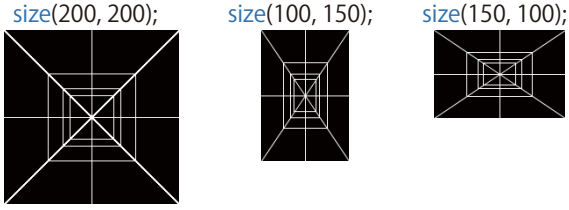
```
1 size(480,120);
2 line(0,0,width,height);//左上から右下へ
3 line(width,0,0,height);//右上から左下へ
4 ellipse(width/2,height/2,60,60);
```

スケッチ6　×印と円を描く（文献1）P.43より）

● 宿題

スケッチ6を応用し，お手本のようなクモの巣の模様を描くスケッチを書いてみましょう．size関数で指定したウィンドウのサイズに合わせて形が変化するように，widthとheightの変数と割り算を使って作ります．

ヒント：rect関数は初期状態では四角の左上の座標をパラメータで指定します．これを四角の中心に変更するためには rectMode(CENTER); という命令をrect関数の前に書いておきます．



四角形は小さい順に，画面の縦・横の長さのそれぞれの四分の一，三分の一，二分の一になるように割り算で計算します

参考文献

1)　Reas, C., Fry, B. 著，船田　巧 訳：Processingをはじめよう第2版，オライリージャパン(2016)．

(2019年11月16日受付)

杉浦　学（正会員）　manabu@kamakura-u.ac.jp

鎌倉女子大学家政学部家政保健学科准教授．慶應義塾大学大学院政策・メディア研究科後期博士課程修了．博士（政策・メディア）．プログラミング教育をはじめとした情報教育に関する研究に取り組む．中高生向けの著書に『Scratchではじめよう！プログラミング入門Scratch3.0版』（日経BP社）など．

ARTICLE

基般

高校における新教科「情報」ができたころのこと

大岩　元
慶應義塾大学

1999 年 3 月 29 日に「学校教育法施行規則の一部を改正する省令」が出されて，「高等学校学習指導要領の全部の改定」の中で，高等学校において新教科「情報」が「情報 A」，「情報 B」，「情報 C」それぞれ 2 単位のどれか 1 科目以上の必修教科として設置された．戦後，新たに生まれた教科として小学校低学年教科「生活科」があるが，これは理科と社会が統合されたものであり，また，高校教科「社会科」が「地理歴史科」と「公民科」に分離されて新教科が生まれた例があるが，戦後の日本の教育で初めて，それまでにない教科が設置されたのである．その前後の状況については，田中規久雄「教科『情報』新設に見る中等情報教育政策の一断面」[1] に詳細に記述されている．本稿では，田中論文を参照しながら，筆者の個人的な見解を述べることにする．

1989 年に施行された学習指導要領

1989 年高等学校学習指導要領は，1987 年に出された「臨時教育審議会」の答申における 3 つの原則「個性重視」，「生涯学習体系への移行」，「国際化・情報化など変化への対応」を受けて，作成されている．そこにおける情報教育は，以下に示すように各教科・科目の学術規範（discipline）に従属する教育手段，教育内容として挙げられている．

この他，美術 I，世界史 A，地理 B，現代社会にも，情報教育に関連する言及がある．

表 -1 に示された内容とその他の科目における情報の扱いを見ると，各教科の中で行おうとしているが，現在小学校におけるプログラミング教育もそれと同じようなことを行おうとしている．そして当時はプログラミングは数学に，ハードウェアとしてのコンピュータは物理で扱うことになっていた．

活用について「家庭科」の中に「情報についての基本的事項を理解させ，コンピュータの基本的な操作を中心とした指導を行うよう配慮すること」という言及があることは，2020 年からの指導要領において中学校の「技術・家庭」の中でプログラミング教育が始まることにつながっている．

しかし，教えているのは数学，物理，家庭科の教師であってコンピュータ科学（Computer Science 略称 CS）を学んでいるわけではなく，当然のことながら多くの先進国で目指している情報教育を期待することは難しい．

表 -1　1989 年施行の学習指導要領における情報教育

教科名	内容	具体項目
数学 A	(4) 計算とコンピュータ	ア コンピュータの操作
		イ 流れ図とプログラム
		ウ コンピュータによる計算
数学 B	(4) 算法とコンピュータ	ア コンピュータの機能
		イ いろいろな算法のプログラム
数学 C	応用数理の観点からコンピュータを活用	
総合理科	データの整理にコンピュータを活用	
物理 IA	(4) 情報とその処理	ア 情報の伝達
		イ 情報の処理
		ウ 情報の記憶
物理 IB	探求活動の報告書作成でコンピュータを活用	
物理 II	(4) 課題研究	問題解決にあたり，コンピュータを活用
家庭科	(5) 家庭生活と情報	ア 情報の収集と選択
		イ コンピュータの活用
		ウ 家庭生活とコンピュータ

どこの国でも情報教育で一番問題になるのは教師教育である．オーストラリアでは，コンピュータ科学と教育学を大学で修得した情報教師の育成をしている．そのようなことを考えないで情報化社会を築こうとして，日本の社会は大きく社会の情報化が遅れたのであるが，そのことに気づいていない．ほとんどすべての情報処理はコンピュータがない時代と同じように紙の上のデータしか使おうとせず，電子データとしての利用を考えていない．用紙をWebサイトからダウンロードさせて，そこに記入した紙の書類を提出させることが一般化している．この状況を，今は政府が率先して変えようと動き出している．

マルチメディアを活用した21世紀の高等教育の在り方に関する懇談会

1996年10月18日，第1回「情報化の進展に対応した初等中等教育における情報教育の推進等に関する調査研究協力者会議」（以下，協力者会議と略す）が開催され，「情報」を独立教科にするのかなどが議論された．その一環として開かれた「マルチメディアを活用した21世紀の高等教育の在り方に関する懇談会」には筆者も委員の一人として参加した．この会議の主査は東大医学部教授の開原成允氏，副主査は放送教育開発センター所長の坂元　昂氏で，情報分野の専門家としてNTT取締役・通信網総合研究所長の青木利晴氏，学術情報センター教授の浅野正一郎氏，慶應義塾大学環境情報学部教授の大岩　元が参加している[2]．この会議で印象に残っているのは，ジャーナリストの野中ともよ氏が「ここで行われている議論は，（今の時代に）飛脚の脚をどうやって強くするかというものである」と批判したことである．コンピュータの導入で社会がどのように変わるのか，それに伴って教育はどのように変わるべきなのかといった本質的な議論が欠如しており，紙の上の情報処理から電子化された情報が大きな意味を持つことになる社会について議論がなされないことへの批判をされたのだと思う．

その後，この協力者会議は教科「情報」の設立に向けて上記懇談会のメンバ東京工業大学教育工学開発センター長の清水康敬氏を主査に活動を行い，「体系的な情報教育の実施に向けて（1997年10月3日）」（情報化の進展に対応した初等中等教育における情報教育の推進等に関する調査研究協力者会議「第1次報告」[3]）によって教科「情報」への道筋をつけたのである．教科「情報」に対しては，数学，物理なども影響力を持とうとしたと思われるが，清水氏はこれらを排除して教育工学主体の，利用者教育としての「情報」設立の方向が決まったのである．この報告書の作成協力者の中には，CSの専門家は入っていない．この結果，情報教育の基礎学問としてのコンピュータ科学は専門高校で学ばれるべき内容とされ，普通高校の内容としては縮小を余儀なくされたのである．

こうした検討の結果として1999年に指導要領が策定され，高校の独立必修教科「情報」が成立し，情報A，B，Cの3科目から2単位選択必修という最初の制度が実施されるようになったのである．コンピュータ科学は情報Bの中にとりあげられている．その目標は

「コンピュータにおける情報の表わし方や処理の仕組み，情報社会を支える情報技術の役割や影響を理解させ，問題解決においてコンピュータを効果的に活用するための科学的な考え方や方法を習得させる」

となっているが，その内容構成は指導要領解説[4]によると，

1. 問題解決とコンピュータの活用
2. コンピュータの仕組みと働き
3. 問題のモデル化とコンピュータを活用した解決
4. 情報社会を支える情報技術

であり，コンピュータ科学を扱う2.では，「コンピュータの効果的活用ができる判断材料となることをねらいとして，技術的，数理的な深入りはせずに，3.や4.に直接役立つ基本的な見方，考え方を認

識させる程度とする」となっており，「教養・知識」の扱いとなっている．3. の「問題のモデル化」は数式によるモデル化だけであって，コンピュータ科学におけるモデル化はまったく入っていない．

ユネスコの提案する情報教育

2002 年にユネスコは IFIP（International Federation for Information Processing）の協力のもとに情報教育の報告書 "Information and Communication Technology in Education : A Curriculum for Schools and Programme of Teacher Development" を発表している [5]．その内容は，表 -2 の通りである．

日本の高校普通教科ではコンピュータ科学が「情報 B」の一部に「教養・知識」に限定されて含まれているのにすぎないが，ユネスコの報告書ではコンピュータ科学の主要部である「ICT 自体の特化」が，専門家とともに高等教育進学者のためのものでもあるという文に注目する必要がある．米国ではすでに，プログラミング経験がないことは高等教育卒業生でないことを意味するようだ．

IV ICT 自体への特化の内容は，日常の問題をアルゴリズム形式で解くことができるようになることである．現在世界中でプログラミング教育の実体として議論されている Computational Thinking そのものと言ってよい．そして具体的には 3 種類のモジュールが提案されている（表 -3）．

ここで驚かされるのは，産業界に入る一般人も SP や GS を学んでいることである．日本の専門高校ではこうしたことを教育していない．VS の作業自体を教えているだけである．商業高校の教師の指導をしたことがあるが，COBOL の書き方を知っているだけでアルゴリズムの概念も，それを作り出す方法も知らなかった．

大学進学者がこの内容を学んでいることも重要である．ユネスコレポートは途上国向けに書かれており，欧米では高等教育修了者にこのレベルの情報教育が行われていることを示している．日本では，多くの（特に文系の）大学卒業者が IT を自分で利用することがあまりなく，ソフトウェアの利用もおぼつかないことは嘆かわしい．

特に注目すべきは，SP1 プログラミング入門（Introduction to Programming）である．その内容は表 -4 の通りである．

1970 年に，日本でも情報関連学科が初めて京大，阪大，東工大，山梨大，電気通信大，金沢工大に設置されたが，この時期は日本の高度経済成長が本格化した時期であり，情報産業が勃興した時期でもある．旺盛なプログラマ需要を大学に求めるのは無理で，情報産業は素質のありそうな大学卒業生にプロ

表 -2　ユネスコ報告書の内容 [5]

I　ICT リテラシー（ICT Literacy）
II　各科目における ICT の応用（Application of ICT in Subject Areas）
III カリキュラム全般への ICT の浸透（Infusing ICT across the Curriculum）
IV ICT 自体への特化（ICT Specialization）
ICT を用いるプロフェッショナルになる人を対象とする （For those who plan to go into professionals that uses ICT） 高等教育に進学する人を対象とする （For those who plan to go into higher education）

表 -3　ユネスコ報告書「IV ICT 自体への特化」の内容 [5]

ICT への特化の準備モジュール（Specialization Preparation Modules）
SP1 プログラミング入門（Introduction to Programming）
SP2 トップダウンプログラム設計（Top-Down Program Design）
ICT への特化モジュール（General Spcialization Module）
GS1 プログラミングとソフトウェア開発の基礎 （Foudation of Programming and Software Development）
GS2 プログラミングの上級要素（Advamced Elemwnts of Programming）
専門職業特化モジュール（Vocational Spcialization Module）
VS1 ビジネス情報システム（Bussiness Information Systems）
VS2 プロセス制御システム（Process Control Systems）
VS3 プロジェクト管理（Project Managemint）

表 -4　ユネスコ報告書「SP1 プログラミング入門」の内容 [5]

解くべき課題（タスク）に向けたアルゴリズムを設計する
（そのために）
実現すべき課題を記述し，確定する
確定された課題を解決する，効果的で効率的なアルゴリズムを開発する
単純で標準的な方法を適用する
できあがった設計をプログラムに翻訳する
（そのために）
（手続き型）言語を用いて設計されたアルゴリズムをプログラムに変換する
読みやすく，理解しやすいインタラクティブプログラムを作り出す
プログラムを実生活で利用可能にする
（そのために）
制作したプログラムを入力，編集，コンパイル，デバッグ，アップデートして実行できる開発環境を用意する
制作したプログラムの内部および外部動作を分りやすく，使いやすくなるような文書を作成する

グラミング教育を 1，2 カ月行って現場に投入することが始まった．コンピュータは従来の機器にくらべて超強力な性能を持っていたので，入門教育だけで十分にビジネスが成立したため，以来 50 年間にわたり，プログラミング入門教育が続いてきた．このような日本独自の IT 技術者育成は，進化をとげて現場で使われるプログラムをやさしいものから順番に打ちこんで動かすことで，未経験者もプログラムが書けるようになる「写経プログラミング教育」が生まれた[6]．この結果，ベトナムにアウトソーシングのための開発拠点を設けると，難しい仕事をするのはベトナム人で日本人技術者はやさしい仕事しかできない状況が生まれているようである．CS の存在を知らない日本の IT 技術者なら，そうなることは当然の結果であるが，いつまでこの状況を続けられるのかが心配である．

今後の情報教育

2015 年頃から，先進国で小学生からのプログラミング教育が始まった．これは，IT によって社会構造が急速に変わり，旧来の教育では自分の子弟が職を得ることができなくなると心配した親たちが，政府にプログラミング教育を学校で行うことを求めたためであるらしい．日本も安倍政権が経済活性化政策の一環として小学校からのプログラミング教育を始めることにした．10 年後にはプログラミング能力は読み書き算盤と並ぶ基本的な市民のリテラシーとなるであろう．

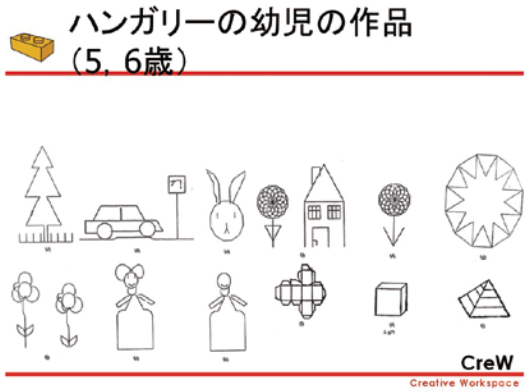


図 -1　Marta T. Szabo 氏の指導した幼児の LOGO 教育結果

プログラミング入門の定番である LOGO は，幼児でもきちんと教えれば修得できることが知られている（図 -1）．手ではとても描く気になれない複雑な花の絵は，プログラム機能を利用すれば幼児でも描けるのである．しかし，修得できない大学生は，この幼児の作品を見て驚く．幼児に行えば可能なことが，大学生になってからでは非常に困難であることは識字教育としてプログラミング教育を行う必要性を示している．

英国のように Computing という新教科を設けることは不可能なので，日本は各教科の中で行うことにした．90 年代までの高校教科「情報」が成立する以前の高校における情報教育と同じ状況であるが，始まるだけで良しとしていかねばならないであろう．

どこの国でも情報教育で一番問題になるのは教師教育である．オーストラリアのように，情報学と教育学を修得した情報教師の育成を考える必要がある．世界標準の情報教育を実現するには，次の学習指導要領の改定に向けて緻密に教育計画を策定する必要があるが，そのためには 30 年先の社会を見越した長期ビジョンを作る必要があるだろう．

参考文献
1) 田中規久雄：教科「情報」新設に見る中等情報教育政策の一断面，大阪教育法研究会，大阪東法研ニュース 第 186 号 (Oct. 1999)．
2) マルチメディアを活用した 21 世紀の高等教育の在り方に関する懇談会協力者，http://www.mext.go.jp/b_menu/shingi/chousa/koutou/001/toushin/960701k.htm
3) 体系的な情報教育の実施に向けて（1997 年 10 月 3 日），http://www.mext.go.jp/b_menu/shingi/chousa/shotou/002/toushin/971001.htm#02
4) 文部省「高等学校指導要領解説 情報編」（2000 年 3 月 30 日）
5) Information and Communication Technology in Education : A Curriculum for Schools and Programme of Teacher Development, https://unesdoc.unesco.org/ark:/48223/pf0000129538
6) 喜多　一，岡本雅子，藤岡健史，吉川直人：写経型学習による C 言語プログラミングワークブック，共立出版 (2012)．

（2019 年 11 月 17 日受付）

大岩　元（正会員）　ohiwahajime@gmail.com

1965 年東大理学部物理学科卒業．1971 年理学博士．東大理学部助手，1978 年豊橋技術科学大学講師，同大助教授，同大教授，1992 年慶應義塾大学環境情報学部教授．2008 年同大名誉教授．帝京平成大学現代ライフ学部教授，相愛大学音楽学部音楽マネジメント学科教授を経て 2017 年（一社）協創型情報空間研究所を設立，代表理事に就任．