OUKA
The University of Osaka
Institutional Knowledge Archive

Title	JavaScript 高等学校の教室で準備なしに利用可能なプログラミング言語
Author(s)	辰己, 丈夫
Citation	サイバーメディア・フォーラム. 2006, 7, p. 5-10
Version Type	VoR
URL	https://doi.org/10.18910/70221
rights	
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

https://ir.library.osaka-u.ac.jp/

The University of Osaka

JavaScript 高等学校の教室で準備なしに利用可能なプログラミング言語

辰己　丈夫
東京農工大学総合情報メディアセンター[1]

はじめに

1980 年代、コンピュータの入門といえば BASIC 言語であった。この頃の「個人用マイコン」のほとんどは BASIC インタプリタを内蔵していた。あれから 25 年が経過した現在、「ほぼすべてのパソコン」に備わっているインタプリタは、JavaScript である。発表者が実際に高校生に 100 分で授業をしたときの様子、その教材などを紹介しながら「すべての高校で実施可能な言語を、すべての高校の教室で実施する教材」についての考え方を述べたい。

1　プログラミングか、アルゴリズムか

情報教育の中でも、情報処理教育といわれている部分がある。1997 年に文部省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が発表した「体系的な情報教育の実施に向けて」では、「情報の科学的な理解」として取り上げられている部分である。

多くの人の理解は、「アルゴリズム (を学ぶこと) は易しいが、プログラミング (を学ぶこと) は難しい」となっている。その結果が、現在用いられている高等学校の教科「情報」の教科書構成である。筆者の手元にある文部科学省検定教科書の「コンピュータにおける情報の処理」「情報の表し方と処理手順の工夫」の内容を調べてみると以下のようになった。

表 1. フローチャートとプログラムの個数 (1)

出版社名	発行	FC	プログラム	
A社	H17	7	4	VB
A社	H15	7	4	VB
B社	H15	3	5	DNCL 風
B社	H17	5	5	DNCL 風
C社	H15	2	1	BASIC
C社	H17	2	1	BASIC
D社	H15	5	3	Pascal 風
E社	H15	3	7	DNCL 風
F社	H15	0	12	DNCL 風

なお、上の表の FC の欄は掲載されているフローチャートの個数 (単なる部品説明は除く) であり、アルゴリズムの説明の度合と捉えることができる。また、プログラムの欄は掲載されているプログラムの個数と記述言語である。

DNCL とは、大学入試センターが情報関係基礎の出題に用いている日本語を用いた処理言語のことである。本ワークショップで取り上げられた PEN を利用すれば、実行可能である。

さて、この表を見ると、

FC > プログラム

VB, BASIC, Pascal 風を採用した各社

FC ≦ プログラム

DNCL 風日本語疑似プログラミング言語を採用した各社

となっていることがわかる。これは、日本語を利用しないプログラムを掲載している会社は、「フローチャートの方がわかりやすい」としているからである。また、DNCL 風言語を利用している会社は、それをプログラムというよりも、「アルゴリズムを日本語で記述したもの」と扱っていることがわかる。そこで、DNCL 風言語の個数を FC として数えれば、先ほどの表は次のように変わる。

表 2. フローチャートとプログラムの個数 (2)

出版社名	発行	FC	プログラム	
A社	H17	7	4	VB
A社	H15	7	4	VB
B社	H15	8	0	
B社	H17	10	0	
C社	H15	2	1	BASIC
C社	H17	2	1	BASIC
D社	H15	5	3	Pascal 風
E社	H15	10	0	
F社	H15	12	0	

この表を見れば、教科書執筆者はプログラミングが難しいという前提であることがわかる。

しかし、この感覚は正しいものであろうか、ということを改めて考え直してみたい。

[1] tttt @cc.tuat.ac.jp

[2] もちろん、存在するのだが、多くの小学生は発見しない。

2　具体的な対象を学ぶこと

小学校 2 年生では、算数で九九を学ぶ。九九で覚えるべき内容には法則性はあまり存在せず[2]、具体的な例をなるべく多く覚えることが求められる。日本語 (国語) でのひらがな、カタカナ、漢字の学習にしても、それらをつなげて出来上がる単語の学習にしても、直接の学習対象になっているのは具体的な実現例 (instance) であって、その仕組みではない。

具体的なことは「すぐに利用できること」に直結する。例えば、アサガオの育て方を学習すれば、アサガオを育てることができるようになる。一方、その方法でチューリップを育てることはできない。もし、小学生に「植物を育てる一般論」を説明しても、それを応用してアサガオを育てられるようになる小学生はほとんどいないであろう。それゆえに、「植物を育てる一般論」は役に立たないことであるといわれても仕方ない。

また、大学の数学科 (専門教育) では、群論、環論、体論、整数論を学んでから初等整数論を学ぶが、たとえ専門性を身に付けたい生徒を対象としていても、高校生 (初等中等教育) には、むしろ初等整数論から入るべきであり、実際の高校・大学の教科書はそのようになっている。

これらと同じことが、コンピュータのプログラミングとアルゴリズムの関係にもいえる。小学生に整列プログラムを教えても、そこからバイナリサーチのプログラムを作れる可能性は少ない。だからといって、小学生にアルゴリズムの一般論を説明しても、整列アルゴリズムを深く理解することはできないであろう。むしろ、なるべく多くのプログラムを具体例として学ぶことが、逆に効果的になる。

ここでは小学生を例にしているが、一般的には初等中等教育においてこれは真実であると筆者は考える。すなわち、初等中等教育においてはアルゴリズムの学習よりもプログラミングの学習を先に行なうべきであり、それも、一般的な内容を追求するのではなく、具体的な手順を学ぶことを優先すべきである。

3　モデルとセマンティクス

プログラミング学習の際に問題になるのが、モデル構築との関係である。

日本の初等中等教育の現場でモデル構築を学ぶ機会としては、算数における「文章題」があげられる。「文章題」とは、単純な計算問題とついになって扱われる問題であり、例えば「A さんは、学校に行くときは時速 6km で、帰るときは時速 4km で歩きました。平均の速さを求めなさい」に代表される問題である。この問題を解くために、家と学校の距離を x と設定すると問題を解ける[3]ようになる。

このように、モデルを構築して問題を解く能力は、単に計算問題を解く能力よりも能動性と抽象性が求められる。

プログラミングの学習においても、同じ問題がある。単純に「文字をたくさん表示する」という問題は、算数における計算問題と同じである。一方、問題解決のためにプログラミングや表計算を利用するということは、モデル構築を必要とすることになり、多くの場合は困難さが増大する。

これは、**シンタックスとセマンティクスの問題**であるといってもよい。

改めて、シンタックス (文法) に属する話題を取り上げてみると、「初等幾何の証明問題」「単純な計算問題」「俳句・短歌」「音楽 (楽譜)」「特定の地域の地理・歴史」「一見無意味なプログラミング問題」である。一方、セマンティクスに属する話題を取り上げてみると、「算数の文章題」「物理における運動方程式」「化学における化学式」「地球全体での気候分布・系統地理」などがある。

初等中等教育における情報処理教育は、既存の各教科・科目の内容を見る限り、

1. 一見無意味な内容のプログラミング
2. 具体的な内容のプログラミング
3. プログラミングの意味を考えたり、アルゴリズムやデータ構造を見てプログラムを考え直す

というプロセスをたどるべきであろう。

[3] 往復では $\frac{x}{6}+\frac{x}{4}$ 時間かかる。故に、平均の速さは時速 $\frac{2x}{\frac{x}{6}+\frac{x}{4}}=\frac{24}{5}=4.8$ km となる。

4　教科書の執筆

筆者は、平成19年度から使用される高等学校「情報B」の教科書(B社)を執筆する機会を得た。そこで重視したことは、前節で述べたことである。具体的には、他の著者の先生方と念密な議論を重ね、最終的に以下の方針をとることになった。

1. 「同じ文字列を10回表示する」というような無意味なプログラミングを練習するところから始め、シンタックスの準備を行なう。
2. 次に、入力や出力の機能について学び、よく知られている簡単な計算手順のみを、プログラミングの題材とする。
3. フローチャートなどを学び、簡単な計算手順を図式化して理解する。
4. フローチャートとプログラムの両方を利用してアルゴリズムを学ぶ。
5. アルゴリズムの比較を行ない、データ構造について考えながらセマンティクスを学ぶ。

言語には JavaScript を採用した。これは、以下の理由を重視したからである。

処理系のインストールが不要

Windows, Mac OS, Linux など、ほとんどのOSで標準で処理系が導入されている。

宿題を自宅で取り組める

特定のアプリケーションを使用する言語では、生徒が自宅で言語処理系を購入 (あるいはフリーダウンロード) し、インストールをする必要がある。本質的ではない。

インタープリタ言語

間違いを探しながら、すこしずつプログラムを改良することができる。

特にJavaScriptの言語仕様に理由があったわけではない。むしろ、上に述べた内容を学校で障害なく取り扱える言語であれば、どんな言語でも構わないのだが、2007年の学校現場を想像すれば、JavaScript以外の選択肢はないともいえる。

実際に執筆をしてみると、その過程で、他の著者から重要なコメント・意見を頂いた。特に、情報Bを採用する高等学校の先生方の御意見を伺い、問題量や難易度の調整などに工夫をすることができた。

教科書の目次

B社「新・情報B」(平成19年度以降使用可能)の第2章 コンピュータによる情報の表現と処理 (合計36ページ) (筆者が主に執筆を担当した) の目次は以下の通りである。

表3. B社「新・情報B」目次の一部

頁	節. 項	内容
24-25	2.1.1	コンピュータの五大機能としくみ
26-29	2.1.2	コンピュータでのソフトウェアの実行
30-33	2.2.1	コンピュータにおける文字の表現
34-37	2.2.2	コンピュータにおける色と画像の表現
38-39	2.2.3	コンピュータにおける音の表現
40-43	2.3.1	コンピュータにおける数値の取り扱い
44-47	2.3.2	情報とデータ量
48-49	2.4.1	簡単なアルゴリズムとプログラム
50-51	2.4.2	アルゴリズムとフローチャート、プログラム
52-55	2.4.3	いろいろなアルゴリズム
56-59	2.4.4	処理手順の工夫

1コマ2ページ程度、合計18コマ (9週) を目安に授業をすることを想定している。本論の冒頭で述べた表に追加するならば、以下のようになった。

表4. フローチャートとプログラムの個数 (3)

出版社名	発行	FC	プログラム
B社	H15	8	0
B社	H17	10	0
B社	H19	7	9　JavaScript

なお、C社のH19年度版「情報B」も、JavaScriptを全面採用している。また、D社のH19年度版「情報B」では、BASICの記述が大幅に増加している。

5　実際の授業

執筆した教科書が実際に授業可能であることを示すために、筆者は実際に、高等学校で授業を行なった。

演習を伴ったプログラミングの授業を行なう際には、エディタや実行方法、環境についての説明は不可避である。もし、その授業がOSの基本的な利用方法も学習目標の一部にしているのであれば、このような説明も授業の一貫となるのだが、今回計画した授業はとにかくプログラミングの体験を得ることを最重要な目的としている。そのため、ファイルタイプや拡張子、あるいはメモ帳の使い方やブラウザの使い方などの主目的から外れることを取り扱うことは避けたい。

そこで原稿原案をもとに web を利用したオンライン教材を作り、

1. リンク上で右クリックをさせ、ソースコードをダウンロードさせる。
2. ダウンロードしたソースコードのアイコンを、

あらかじめ起動しておいたメモ帳 (Windows の場合) に Drug & Drop で開かせて編集させる。
3. 編集が終わったらメモ帳を閉じさせずに、こんどはブラウザ (MSIE) に Drug & Drop で開かせて実行させる。
4. 実行を確認したら、プログラムに修正・改変を加え、動作の変化を比較する。

という手順にした。この方法であれば、生徒達は既にワープロソフトや表計算ソフトの演習で慣れているので、編集・実行などのために学ぶ分量を減らすことが可能である。

また、授業を開始するに当たっては、冒頭で簡単な数学的な知識のクイズを行なった。

- 三平方の定理を知っているか？
- $1+2+\cdots+100$ の計算結果と計算方法は？
- 円周率を知っているか。知っているなら、おおよその値は？

これらのクイズを行なっておくことで、生徒達に冒頭で安心感を与え、その後の授業スムーズに進行することができた。

なお、上記は、授業方法に関する一般的な技術である

- 試行錯誤する時間を設けること
- 遊ぶ時間を設けること
- 既にわかっていることに結びつけること

をすべて満たすようにした結果である。

実際に授業を行なってみると、生徒達 (主に高校 1 年生) が熱中したのは、「じゃんけんゲーム」(付録 2.1 参照) であった。何回も reload ボタンを押して「じゃんけんゲーム」で遊ぶ過程で、乱数と 1 次不等式を体験的に理解することができた。

6　指導書・演習本

教えたい内容が素晴らしくとも、下手な人が教えてしまっては授業の魅力は全くなくなり、結果として「プログラミングは不要である」との意見につながりかねない。プログラミングが情報教育・情報処理教育に有効であることを主張するためには、実際の授業例を見せることは重要である。特に、その授業が成功するためには授業方法の基本を押える必要がある。また、その際には、「(準備さえすれば) 誰でもできる授業例」を見せることも重要である。有名なスポーツ選手や芸術家の演技や作品は確かに素晴らしいが、それを見た多くの人は「自分には真似ができない」と思う。「誰でも実現可能な授業」であることを主張するためには、「誰が見ても、準備さえすれば実施可能」と思えることが必要である。

そこで、指導書と呼ばれる、授業進行を補佐する本の存在が重要である。指導書に授業進行の TIPS を取り上げる予定である。

また、教科書に執筆したのは、プログラミングのシンタックスの入門と、基本的な計算手順のみである。JavaScript が持つオブジェクト指向的なプログラミング手法、特にブラウザが表示する領域のプロパティを変更する手法などについては、今回の教科書で取り扱うことはできなかった。

そこで、別途に演習本を計画している。演習本では、教科書に入っている内容の類題を扱いながら教科書を補充し、さらに、プロパティ変更などを学びながら、ブラウザ制御言語としての JavaScript を学ぶようにデザインされている。高等学校でも、学習意欲の高い生徒や、意欲のある教員を対象として想定している。

7　まとめ

高校の教科「情報」においてプログラミングを教えたい場合の一つの解として JavaScript を提案し、実際に検定教科書に執筆した。

また、執筆内容の妥当性を検証する為に、高等学校で授業を行なった。

参考資料

[1] **文部科学省「情報」検定済教科書**, オーム社, 啓林舘, 実教出版, 清水書院, 第一学習社, 日本文教出版 (五十音順)

[2] **高等学校学習指導要領　情報**, 文部省, 1999, ISBN4-17-153520-4 C0037

[3] **高等学校学習指導要領解説　情報編**, 文部省, 2000, MESSC 1-9942

[4] **高等学校におけるプログラミング教育で何を教えるべきか**, 辰己 丈夫, 筧 捷彦, 情報処理学会, 1998 年度夏のプログラミングシンポジウム論文集, pp.55-66

[5] **情報教育の音楽化**, 辰己 丈夫, 情報処理学会コンピュータと教育研究会, 2001-101, pp.39-46

付録 (教科書執筆原稿の原案)

以下に、教科書に執筆した原稿の原案となったものの一部のみを紹介する。

1　簡単なプログラム

1.1　簡単なプログラム

JavaScript で書かれた非常に簡単なプログラムは以下のようなものである。

01kurikaeshi.html

行番号	プログラムの内容	その意味
1	<html>	HTML に埋め込むために必要な部分
2	<body>	
3	<script>	
4	for (i=1; i<=10; i++){	i の値を 1，2，…，10 まで変化させながら、
5	document.write("情報 B");	「情報 B」と記す。
6	}	第 4 行のカッコ開きと対応するカッコ閉じ
7	</script>	HTML に埋め込むために必要な部分
8	</body>	
9	</html>	

プログラムを書く時は、漢字が必要なところ以外はすべて半角英数字で書くようにする。このプログラムを「 js01.html 」という名前で保存して、それを web ブラウザで開いてみよう。

情報 B 情報 B 情報 B 情報 B 情報 B 情報 B 情報 B 情報 B 情報 B…

と、「情報 B」という文字列を 10 個表示する。

プログラムがうまく動かなかったり、動作がおかしかったり、新しい改良を加えてみたかったりするときは、プログラムを編集した後で再読み込みをしてデバグや変更を行なう。

なお、この後に紹介するプログラムの冒頭の 3 行と最後の 3 行は、このプログラムと同じなので、今後は省略する。（したがって、この後に紹介するプログラムの行番号は 4 から始まる。）

1.2　数値を表示させてみよう

次のプログラムは、1 から 100 までの数の和を求めて、それを表示する。

02ichikarahyaku.html

4	s=0;	和を格納する変数を 0 にしておく
5	for (i=1; i<=100; i++){	i の値を 1，2，…，100 まで変化させながら、
6	s=s+i;	s の値を i だけ増やす
7	}	第 5 行のカッコ開きと対応するカッコ閉じ
8	document.write("s=",s);	「s=」と記し、その直後に s の値を記す。

変数の値を変化させたり設定したりするときは、変数名=数式というように、変数名を左辺に、新しい値を右辺に書く。s=s+i というのは、「新しい s の値は、古い s の値に i を加えたもの」という意味で、すなわち、s の値を i 増やす効果がある。また、for の { と } の間では、i の値は i=1,2,3,… と変化する。この性質を利用して、1 から 100 までの和を求めることができる。

1.3　入力を読みとる

次のプログラムは、キーボードから入力を求めて、その文字列を 10 回繰り返して表示する。

03sukinakotoba.html

4	m=prompt("好きな言葉はなんですか？");	キーボードから文字列を読み込んで m に代入する。
5	for (i=1; i<=10; i++){	i の値を 1，2，…，10 まで変化させながら、
6	document.write(m);	m を表示する。
7	}	第 4 行のカッコ開きと対応するカッコ閉じ

1.4 数値を入力させる

次のプログラムは、キーボードから入力を求めて、それを数値 n に変換して、1 から n までの和を求めるものである。

04nwonyuuryoku.html

4	s=0;	s に 0 を代入する。
5	n=parseInt(prompt("n を入力して下さい"));	キーボードから文字列を読み込んで数値に変換してから n に代入する。
6	for (i=1; i<=n; i++){	i の値を 1，2，…，n まで変化させながら、
7	s=s+i;	s の値を i だけ増やす
8	}	第 5 行のカッコ開きと対応するカッコ閉じ
9	document.write("s=", s);	「s=」と記し、その直後に s の値を記す。

既に見たように、”813” と数字を並べても、それは文字コードに従って”8”, ”1”, ”3”の文字番号が並んだだけで数値ではないので、数値として解釈する parseInt という関数を用いる。

2 いろいろなアルゴリズム（乱数の利用）

2.1 じゃんけんゲーム

JavaScript には、乱数といって、サイコロを振るのと同じように一定の規則に従ってない数を沢山生成する機能がある。例えば、x=Math.random(); とすると、x には、0 以上 1 未満の数が代入されるが、その値は、実行するたびに異なる。このとき、3*x の値は 0 以上 3 未満の数なので、それを元にしてじゃんけんゲームを作ることができる。

06janken.html

4	y=3*Math.random();	y に 0 以上 3 未満の乱数を代入する
5	document.write("コンピュータの手は");	「コンピュータの手は」と記す。
6	if (y<1){	もし y の値が 1 未満ならば
7	document.write("グー");	「グー」と記す。
8	} else if (y<2){	そうでなくてもし y の値が 2 未満ならば
9	document.write("チョキ");	「チョキ」と記す。
10	} else {	それでもないときは
11	document.write("パー");	「パー」と記す。
12	}	第 10 行のカッコ開きと対応するカッコ閉じ
13	document.write("です。");	「です。」と記す。

このプログラムの 6 行目の if と、8 行目の else if と、10 行目の else が、条件判断を実現している。