情報教育におけるプログラミングの役割
～教育用プログラミング言語に関するワークショップ2007基調講演～

久野　靖*

2007.3.10

1　はじめに

□ 「基調」講演って何でしょうか?

- 「教育用プログラミング言語に関するワークショップ2007」実施に当たってこれは共通の土台としようね、という内容 (久野の解釈)
- そんなことが簡単に合意できるなら苦労しないわけで (久野の懸念)
- 結局、我々が「これが共通理解だといいなあ」と思うことをまとめる (当り前な結論)

□ 以下の内容のあらまし

- 情報処理学会の「提言」「コメント」について
- 情報「新・試作教科書」について
- なぜ「プログラミング」を教えることが望まれるのか
- 昨年の本ワークショップの紹介
- 教育用プログラミング言語の設計:ドリトルの場合 (時間があれば)

2　情報教育と情報処理学会の提言

□ 情報処理学会…日本最大の「情報」関連学会

- 歴史的に「学者・研究者」だけでなく「企業人」が多数所属
- 活動分野も多岐に渡る (学術研究から社会活動まで)
- 情報処理教育委員会…学会が対象とする分野 (情報・情報処理) の教育に関わる検討や活動
- カリキュラム翻訳や制定、教科書執筆、社会提言←
- http://www.ipsj.or.jp/12kyoiku/

□ 「2005年後半から2006年初等にかけての事件と情報教育の関連に関するコメント」

- この時期に多くの「事件」があった→「情報処理と情報システムの原理に対する理解の欠如」が共通の要因ではないか、という指摘
- わが国の産業界ひいては社会全体の態度に問題がある、という指摘

□ 例:構造計算書偽造事件

- 概要:ある建築士が確認申請に出した構造計算書に偽造があり、強度不足で住めない建物が大量に存在することが判明した
- 要因:構造計算書は「大臣認定プログラム」が計算出力した結果を改ざん無しに提出されるべく制度が設計されていた。しかし出力を直すことは現にできるようになっていた
- ディジタル情報は基本的に変更が容易。改ざんが無いことを保証する必要があるならそのような技術を使う。しかしそもそも「結果」だけ添付して「入力データ」を添付しないのがおかしすぎる

□ 例:「1円61万株」誤発注事件

- 概要:ある証券会社の担当が「61万円で1株」の売り注文を出すところを間違えて「1円で61万株」の売り注文を出した。多くの顧客がこの格安株を購入し、証券会社は膨大な損失を被った
- 要因:東証のシステムはヘンな注文でもプロ (証券会社) が出したのだからそのまま受け入れるという方針で作られた。出した注文を取り消す機能に間違いがあり取り消しができなかった。証券会社のシステムは「ヘンな注文」に対する警告機能を持っていたがその警告はしょっちゅう出るので常に「OK」と返答する習慣ができていた
- 人間は必ず間違いを侵すこと、ソフトには必ず間違いが含まれることを分かっていればこうは作らないはず。しょっちゅう出る警告に効果がないことも常識であるはず。

*筑波大学大学院経営システム科学専攻

□ 例:フィギュアスケート採点ミス事件

- 概要:2005.12.24 の全日本選手権で採点ミスがあり表彰式後に 1 位と 2 位が入れ替わることになった。2 位の選手が本来許される回数以上に特定の技を行ったのに普通に加点したため。
- 要因:採点プログラムは上記の制限に対応しておらず、採点委員が技を入力しないことで対応するはずだったのに採点に追われて失念しそのまま入力してしまった。
- 人間は必ずミスをする。このため、本来ならこのようなチェックはプログラムに組み込まれているべきだが、それが無理ならこのようなチェックは別の人が二重に行うべきなのにそうしなかった (国際スケート連盟ではそうしていた)

□ すべてに共通する要因…

- 「情報システムでこういうことをすれば当然こうなるでしょ」という「常識」「原理」理解の欠如
- 「あなたまかせ」…コンピュータのことは分からないから技術者にまかせるといって責任を放棄
- 日本の社会すべてがそう→小学校〜高校の「情報」教育を通じて日本人全体の「情報水準」の底上げが必要

□ 「日本の情報教育・情報処理教育に関する提言 2005」

- 「情報水準の低さ (前述)」「高度 ICT 人材の不足」の 2 問題を指摘。後者については…
- 中国・インドなどにソフトの「下請け発注」するとしても、それを管理する人材すら不足
- 顧客側 (情報システム発注側) に分かる人材がいない→それで開発企業に丸投げしても本当に求めるものができるかは運まかせ
- これらを克服するには→「情報水準の底上げ」に加えて「適性・関心を持つ生徒への深く学ぶ機会提供」も必要

□ 「提言」のうち初等中等教育に関わる部分

- 小・中・高それぞれの発達段階に応じて適切な「手順的な自動処理」の体験を持たせる
- 高等学校の教科「情報」に選択科目を追加することで「手順的な自動処理」に関心を持った生徒が系統的に学べる場を設ける
- 「手順的な自動処理」の構築とは…(1) 問題の同定/記述/定式化/解法考案、(2) 手順化/具体化しコンピュータで実行、(3) 検証し必要なら最初から反復、のプロセスを指す
- 「手順的な自動処理」はプログラミングに限定するものでない…表計算のワークシート設計、音楽ソフトの演奏機能、等多様なものを含む概念

3 新・試作教科書

□ 普通教科「情報」で (今後) 教えることが望ましいと考える内容を「教科書」のような形態で提案

- 全体方針:現行指導要領で提示している「実践力」「科学的理解」「情報社会」の 3 本柱による「情報社会を生きる力」の枠組は維持
- ただし「生きる力」とは「全員が一定以上の水準に到達」と「情報社会をリードする人材の輩出」が合わさったものであるべき
- 現行の「情報 A/B/C」から 1 科目選択では共通の土台という点で不十分 (とくに高等教育に進む場合)

□ 新・試作教科書の全体構成

- 現行「情報 A」単独の内容は中学校「情報とコンピュータ」でカバーされると考える→「情報 B+情報 C」の内容を統合した「情報 I(仮称)」の 1 科目 2 単位 (必履修) に
- 適性・関心を持つ生徒の学習機会のため「情報 II(仮称)」の 1 科目 2 単位 (選択)。希望する生徒は必ず選択可能とし、高等教育進学者には必須としたい
- さらに進んだ内容の「情報 IIIx」を複数設け、この部分は高校ごとの独自性を持たせる。新・試作教科書としては内容例の提案のみ。内容例:「総合製作」「情報デザイン」「情報科学のエッセンス」「情報社会学」など

□ 「情報 I(仮称)」2 単位 (必履修)

- 1 章:ネットワークと情報…ネットワークの利用、安全性、Web サイト作成
- 2 章:情報とその活用…メディアとコミュニケーション、情報の表現、情報機器と人間の接点、Web サイトの構築プロセス
- 3 章:コンピュータと情報…プログラミング入門、対話的プログラム、コンピュータの構造と手順的な自動処理、情報と問題解決
- 4 章:情報社会…現代社会と情報技術、さまざまなストーリー、情報社会における変化と問題点、情報社会と個人

□ 「情報 I(仮称)」において工夫した点

- プロジェクト管理の導入…中学まででは個人作業主体、グループ作業でも「分担」程度→プロジェクト管理では作業内容を記録、各担当部分のレビューし達成度を管理→高校で学んでおいて欲しい内容と考える
- プログラミング…3 章の前半 (全時間の 1/8 程度)、「体験」を主目的とする、ドリトル言語
- 情報倫理…具体的なストーリ+それを読み解くことで理解しておくべき概念を学習、というスタイル

□ 「情報 II(仮称)」2 単位 (選択)

- 1 章:コンピュータとプログラミング…コンピュータの動作とプログラムの関係、JavaScript 言語、入力/処理/出力、枝分かれ、繰り返し、関数定義
- 2 章:アルゴリズムとデータ構造…アルゴリズムの考え方、段階的詳細化、反復、再帰、データ構造、配列、レコード、探索、整列
- 3 章:メディアリテラシーと情報倫理…メディアリテラシー、情報倫理、安全性、情報社会の出来事と考え方
- 4 章:情報システムと情報社会…情報システムとその形態、企業における情報システム、情報システムの構造と特性、データの管理

□ 「情報 II(仮称)」において工夫した点

- 全体の半分がプログラミングに基盤を置く情報科学入門的内容…高等教育進学者が大学初年度以降に大学生ならではの水準でプログラミングを道具として使うようになることを想定。高校から社会に出る場合でもこの段階をこなしておくことでコンピュータによるデータ処理の「原理」を「体験的に理解」(この理解と 4 章の「情報システムと情報社会」で相互補完)
- 情報倫理教育 (=何が「正しい」「正しくない」を考えること) と安全性教育 (=どのような「危険性」がありどのような「対処方法があるか」を区別して扱う
- 「出来事と考え方」は「情報 I」と同様にストーリベースだが倫理的側面と安全性側面とを区別して読み解く

□ 新・試作教科書に対するご意見等…

- 「大学の初年度教育に使ってみたい」「難しすぎる」
- 扱う内容の範囲については十分検討したつもりだが、個々の事項 (説明する用語範囲) などは詳しすぎる面もあると思う…
- 「試作教科書」というより「試作指導書?」…旧試作教科書もそうだった
- 教科「情報」の内容としてこれだけのものをカバーして欲しい、教師はこの深さまで理解しておいて欲しい、という我々からの提案ということで…

4 なぜ「プログラミング」を教えるのか

□ 「プログラミング」に対する拒絶反応

- 「プログラミング」の内容に言及すると「非常に強い拒絶反応」
- 現場教員だけでなく報道関係者などが「プログラミング」を敵視?
- 「国民全員がプログラマになる必要などない」それは当り前
- 「プログラマになるのでなければプログラミングを学ぶ必要はないはず」それは違うでしょ?

□ 理由 1:コンピュータの構造や原理を学ぶにはプログラミングを「体験」することが必要

- 例:小学生が試験管を持って「でんぷんのヨード反応」を体験するのは全員を化学研究者/技術者にするため? 違うでしょう? 「物質どうしが反応することでさまざまな現象が起きる」体験を持たせることが重要
- 同様に「計画して作ってもその通りにならない (実は人間が間違っている)」「いつまでも終わらないプログラムを作ってしまうことがある」「与えるデータが正しくないとめちゃくちゃになる」などの「体験」を持たせることは極めて重要
- 現在の多くの「情報」教科書にあるような「CPU は命令を実行していきます」レベルの知識では原理的すぎて有用でない (原子は原子核と電子から成っています、みたいな)

□ 理由 2:初等中等教育レベルで基本的なプログラミングを学び、その土台に立って高等教育に進むのが世界的水準

- 情報教育国際会議で「日本では高校でプログラミングはほとんど教えられていない」と報告すると「日本は大丈夫なんですか」と心配してくれる…

- 中国、韓国、インドなどでは初等中等段階からすぐれた素質を持つ児童生徒を計画的に育成→将来の情報技術においてリードするため
- わが国では英才教育は簡単でないとしても、すぐれた素質を見出す (本人が) きっかけがないと人的資源の浪費

□ 理由 3:「こうしたらどうなるか」の予測が正しく行えるような国民全体の「情報水準」向上のために必要

- 例:細いコップと太いコップで同じ高さまで水を入れて「水の量は同じですね」でだまされる人はいませんよね? →教育水準。これでだまされていたら他国の人と交渉できない
- 「プログラムの印刷出力をそのまま原本としていたら改ざんの恐れがある」とか「警告メッセージをしょっちゅう出していたらそのメッセージは役に立たない」とかの常識は無かったことが明らかになったわけで…
- これらを個々の知識として教えていてもきりがない (いたちごっこ) →「プログラミング体験から各自が見出していく」以外に解はない

□ 繰り返しになりますがまとめると…

- 「国民全員がプログラマになる」ためにやるわけではない
- 「全員体験」のための時間は小・中・高それぞれで数時間ずつ
- コンピュータの構造と原理を理解するために「体験」が必要
- 他国と同程度の水準は維持しないと将来が心配
- 「こうしたらどうなる」の予測ができないと情報社会が立ち行かない

5 ワークショップ 2006 の紹介

□ 昨年度のワークショップの趣旨…

- 教育用プログラミング言語の理念と設計、処理系の実装、利用事例
- プログラミング教育環境とその経験
- プログラミング教育の方法論、教育学的考察
- これらを「各言語単位」で紹介、しかも非常に盛況で多数の言語の発表応募があった→非常に駆け足での紹介「だけ」になってしまったという反省
- ここでは簡単に主要な言語についておさらい

□ 現役の実用言語…C++、Java、Lisp、JavaScript

- C++や Java の方がコンパイルする (強い型検査の) 言語なので難しい。大学での初年度教育や専門教育において工夫しつつ活用できる
- Lisp は対話的プログラミング環境としての伝統があり、その側面を生かした大学教育への活用事例。また LEGO MINDSTORMS の中に Lisp 処理系内蔵などの工夫も
- JavaScript はブラウザに処理系が内蔵されているので手軽に授業に導入できる。「情報」教科書でも複数の採用

□ 伝統的な教育用言語…BASIC、LOGO

- これらの言語は教育利用に長い実績を持ち多くの教育実践事例がある
- 一方で「BASIC によるプログラミング教育→プログラミング教育は困難」のようなイメージを作ってしまった面もある
- LOGO はタートルグラフィクスが有名だし効果的。その先の実践は簡単でない。そこを頑張ったという実践報告

□ 新しい教育用言語/環境…Squeak eToys(+言霊)、PEN、ドリトル

- BASIC、LOGO は 30 年前の教育言語→今日的視点で教育言語/ 環境を作ろうとする試み
- Squeak eToys では画面上のお絵描き (モーフ) にスクリプトを付随させて動かす。ワークショップの事例多数。スクリプトはタイルスクリプティング環境で作成→構文エラーが起きない
- 言霊 (ことだま):日本語として自然に読める語順/表現をめざした言語。それを eToys のタイルスクリプトに組み込むことで構文エラーなしに自然な日本語記述が作成可能に
- PEN は大学入試センターの問題記述用日本語表現である DNCL をもとにプログラミング言語/環境として構成。「書きにくさ」(日本語として読めるがゆえに何が正しい/正しくない記述かが区別しにくい) を補う入力補助機能を搭載
- ドリトルは日本語識別子を採用した教育用オブジェクト指向言語。タートルグラフィクスにより描画したものが図形オブジェクトとなり個別に操作可能。ロボット制御、音楽演奏、ネットワークなどの活用事例も

□ 新しいユーザ指向な言語…ひまわり/なでしこ、HSP、Tonyu、Viscuit、Rosetta

- ひまわり/なでしこは日本語の構文を土台にした文法を持ち、多くの人が定型処理をプログラムで片付けられるようにする目的で作られた。実用指向
- HSP(Hot Soup Processor)、Tonyu はいずれもゲームプログラミング環境で、ゲームを作りたい人が簡単に作れることを目指し、それを通じてプログラミング教育にも貢献
- Viscuit は「やわらかなパターンマッチによる絵の書き換え」によるプログラミングが可能。構文エラーがない「例示的プログラミング」
- Rosetta は美術教育むけにペイントツールのペンなどのオブジェクトをプログラムで操作可能にしたというイメージ

□ 全体として…

- 「プログラミング教育をやろう!」という熱気は非常に大きいものがあった
- 言語単位の紹介だったため時間がコマ切れになってしまったというのが反省点→今年は分科会方式に

6 教育用言語の設計:ドリトルの場合 (久野+兼宗)

□ せっかくの「教育用プログラミング言語に関するワークショップ」なので…

- ドリトルって結局、どういう言語なのか
- ドリトルの場合、どんなことを考えて設計したか
- どういうことが良かった/悪かったと思っているか

□ ドリトルはオブジェクト指向言語

- オブジェクト=「データ+機能」→さまざまなオブジェクトを用意することで、そのオブジェクトが持つ多様な機能を利用できる
- 教育用言語として「描画」「制御 (シリアルポート出力)」「ネット」「音楽」などさまざまなこと (=児童に興味を持ってもらえそうなこと) を採り入れたかったから

□ ドリトルに関するいくつかの誤解

- ドリトルはおもちゃ言語で大きなプログラムは書けない (嘘)
- ドリトルの構文は厳密でなく適当に書いても動く (嘘)
- 文字で記述する言語より図的な記述の方が構文誤りがないからよい (嘘)

□ オブジェクト指向エンジン (=実行機構) の設計

- オブジェクトは名前のついたプロパティ(フィールド)を持つ
- プロパティがコード (実行可能なもの) の時はそれがメソッドとなる
- プロトタイプ方式による継承メカニズム
- ほぼ、JavaScript 言語そのまま← (久野が「入門 JavaScript」という本を書くために JavaScript の勉強をした後だったので…)

□ ドリトルの実行エンジンの機能=JavaScript の実行エンジンと (ほぼ) 同等

- JavaScript 言語→今日のブラウザに搭載され多くの用途に使用
- ドリトルでも同様水準のコードを書いて実行させることは同様に可能 (つまりドリトルはおもちゃではない)
- もちろん、その機能実現に必要なオブジェクトは揃っている必要
- コードの記述性についてもとくに不利な点はないと考えている

□ ドリトルの構文→苦労したところ

- 「どうしたら児童にも使えるオブジェクト指向言語が作れるか?」
- 我々の考え:「Java とか C++の構文では絶対に無理」(記号とか多すぎ、約束が複雑すぎ)

  オブジェクト . メソッド名 (引数, 引数, …)

- 記号は最低限にして、全部並べるだけで済ませる→先頭をオブジェクト、最後をメソッド名にしてはさむ
- 後置記法…日本語の語順に一致するのでいいかも。後置記法といえば Forth みたいなスタック言語だが、スタックは児童に教えるのは難しい→スタックではなく「平らに」ならべるだけ

□ ドリトルのメッセージ送信式の構文

オブジェクト！ 引数 引数 … メソッド名

- 重大な問題:「引数」と「メソッド名」をどうやって区別?

- 「数」(定数) は引数、「名前」はメソッド名 (なんて安易な?!)
- 引数に名前 (変数名) を入れたければかっこで囲めばよい (アイデア!)

  カメ！ 100 歩く (角度) 右回り 100 歩く
- かっこ内は中置記法/メッセージ送信記法のどちも可

  カメ！ 100 歩く (x+y) 右回り 100 歩く
  カメ！ 100 歩く (欄 1！読む) 右回り 100 歩く
- このように構文は厳密に決まっていて構文エラーも出る。ただし記号が少ないので記号の打ち忘れのようなものは出ない

□ 「決まった構文を持つ言語」がいいのかどうか?

- Mindstorms 言語や Squeak eToys のような図的プログラムの方が構文エラーが出ないからいい?
- 構文エラーが出ないからといってそれだけで「正しい」プログラムや「思った通りの」プログラムは書けない (意味的エラー)
- 逆に意味的エラーが取れる程度まで習熟するなら構文エラーは大した問題ではない。だから最初に挫折しない程度でよい

□ 結局、生徒に何を学んで欲しいのか?

- 「ある決まった記述」→「それにきっちり対応した動作」
- その「記述」は「文字による記述」(=言語) であるのがよいと考える
- 理由 1:「言語」であることで高い柔軟性/記述性が可能 (字下げとか識別子の選択とかコメントとかで見た目が工夫できる。教科書の地の文中にすんなり入れられる)
- 理由 2:「決まった書き方の存在」こそ学んで欲しいことの 1 つ

□ よかったこと:

- 相当「特異」な構文だが実際に教えてみると無問題
- 記号が少なく「平ら」なのは間違いが置きにくく成功
- プロトタイプ方式 (← JavaScript)、ブロック (← Smalltalk) の特徴が生きた

□ ちょっと困ったこと:

- 「特異な構文」のため他言語への移行はちょっと心配 (逆にその方がいい?)
- 「普通のアルゴリズム教育」はちょっとやりづらい (自分が慣れてないだけ?)

7 まとめ

□ 「情報教育」ではきちんと情報技術のこと (コンピュータのこと) を教えて欲しい。でないと今後の日本の社会が立ち行かない

□ 教育手段として「プログラミングを教える」のがよいと考える

- レベル 1: 「体験してみる」「ちょっと変えると動きが変わる」程度でも
- レベル 2: 自分で計画し、計画したものを作る→ものつくり

□ これまでに言われている「困難さ」は道具 (=プログラミング言語) が古かったり不向きだったりしたためでは?

- 多くの人が「新しい、より教育に向く言語」を探求 (昨年のテーマ)
- 言語という道具をいかに使うか/教えるかの探求も必要 (今年のテーマ)