ぺた語義

【第 5 回】

contents

［コラム］
ジグソーパズルのピースを組み合わせると
…掛下哲郎

［解説］
高度情報処理技術者の基礎力育成を目指した学部教育
…伊藤克亘・荒川傑・善甫康成・藤田悟

［解説］
京大における Lisp を使ったプログラミング教育
…湯淺太一・奥乃博・尾形哲也

Column

ジグソーパズルのピースを組み合わせると

教育コーナー「ぺた語義」では情報分野における教育や人材育成に関するさまざまな取り組みを広く取り上げることにしている．「教育」は主に学校教育を指し，「人材育成」は主に社会に出てからの教育・訓練等を意図しているつもりである．所管する省によって用語が違うのも妙な話ではあるが，教育と人材育成は本来，密接に連携すべきだと思う．

「ぺた語義」で取り上げるさまざまな取り組みには，互いに関連しているものがある．たとえば，JABEE 認定については 52 巻 12 号と 53 巻 1 号で取り上げる予定だ．情報分野における JABEE 認定プログラムの修了は技術士（情報工学）第一次試験の合格と同等である．彼らは 52 巻 12 号で取り上げる技術士 IPD の対象者であり，技術士資格を取得して技術士 CPD を行うことで，社会に貢献できる高度な IT 人材として公的にも認められる．情報処理学会は 52 巻 10 号で紹介する高度 IT 資格制度の構築に取り組んでいるが，これは，技術士資格とも連携した国際資格とすることを目指している．編集上の都合で，関連記事を一括して掲載することはできなかったが，個別の取り組みを関連付ける，より大きなストーリーを理解していただくと，「ぺた語義」をもっと楽しめると思う．

上記は編集者が意図したストーリーの一例だが，執筆者や編集者が意図しなかったストーリーを発見する読者がおられるかもしれない．筆者も，そのような発見を楽しみにしている．

東日本大震災の直前に JABEE 主催の国際審査員研修会に参加する機会があった．その中で，学部全体を審査する主査および個別の学科等を専門別に審査する審査員によって審査チームを構成するのが世界では一般的だとの話があった．日本には，機関別認証評価（7 年ごと）と専門別認証評価（専門職大学院対象，5 年ごと）の制度がある．2 種類の認証評価には重複する部分が多いので，それを整理するための議論が中央教育審議会でも行われているが，2 種類の認証評価の評価サイクルを揃えた上で，機関別評価を行う主査と，学科等を単位とする専門別評価を行う評価者によって認証評価チームを構成するのは，評価の無駄を減らすための良い方法かもしれない．

こういった新たなアイディアを生み出すための素材として「ぺた語義」をご愛読いただけると幸いである．

世の中は広い．教育や人材育成に関する良い取り組みであっても，編集者が知らないために記事として取り上げられないのはもったいない．そうした取り組みをご存知の方は，取り組みの中心人物をぜひご紹介いただきたいと考えている．

掛下哲郎（佐賀大学）

ロゴデザイン ● 中田 恵　　ページデザイン・イラスト ● 久野 未結

解説

高度情報処理技術者の基礎力育成を目指した学部教育

伊藤克亘[1]　荒川　傑[2]　善甫康成[1]　藤田　悟[1]
[1]法政大学　[2]（株）グルージェント

まえがき

情報系の学生は，ニーズに比べ人数が少ない．この現状では，情報系学部としては，当該分野に就職できることに甘んじるだけでなく，IT 人材市場において指導的役割を果たせる人材を一人でも多く排出しなければならないだろう．そこで，指導的役割を果たせる人材像として，IT スキル標準☆1 のレベル 4☆2 以上の人材を「高度情報処理技術者」とする．この「高度情報処理技術者」の人材育成を目標に，卒業生の 1/3 がそのレベルに到達できる素地をつくる教育体制を設計し，その一部が文部科学省の「大学教育推進プログラム」☆3（GP: Good Practice）に採択された．

GP のタイトルは，「高度情報処理技術者を目指す学士力の育成」である☆4．これを実現する主な取り組みとして，プログラミング教育における仮想少人数クラスの実現，ガラス箱オフィスアワーセンターの設置，情報リテラシー科目の新設，リクエスト集中講義の開講，1 万行演習の新設，ポートフォリオ評価，専門スキルに関する情報提供がある．これらの取り組みが有機的に相互作用して，目標の実現を目指している．本稿では，まず，プログラミング教育への取り組みとして，仮想少人数クラスと 1 万行演習を紹介する．次に，それを学生組織で支えるガラス箱オフィスアワーセンター，最後に，情報科学のためのリテラシー科目の中に位置づけるコミュニケーション講義について概要を紹介する．

☆1 情報処理推進機構制定.
☆2 IT スキル標準より引用：プロフェッショナルとしてスキルの専門分野が確立し，自らのスキルを活用することによって，独力で業務上の課題の発見と解決をリードするレベル．社内において，プロフェッショナルとして求められる経験の知識化とその応用（後進育成）に貢献しており，ハイレベルのプレーヤとして認められます．スキル開発においても自らのスキルの研鑽を継続することが求められます.
☆3 http://www.mext.go.jp/a_menu/koutou/kaikaku/gp.htm
☆4 http://cis.k.hosei.ac.jp/gp/

プログラミング教育

法政大学情報科学部では，2000 年の開設以来，情報処理学会制定のカリキュラム標準 J97 に準拠したカリキュラムを実施しており，2010 年のカリキュラム改革以降は，新しい標準である J07 に準拠している．それらの標準に準拠して，多くの項目を効率よく教えるためには，積上げ式のカリキュラムとならざるを得ない．積上げ式のカリキュラムに沿って学生を成長させるには，土台となる導入科目できちんと理解させ満足させ，後続科目への動機づけをすることが肝心である．本学部のカリキュラムで，そのような位置にあるのが，1 年春学期の「プログラミング入門 1」である．

❑ プログラミングの初年次教育「プログラミング入門 1」

● コース設計

プログラミング入門 1 は，プログラミングそのものに慣れ，実現したいことを自由にプログラムで表現できるような知識とスキルを学生に身につけさせることを目標とする．内容は J07 のラーニングユニット（LU）を参考に構成した．

問題番号	全体正答率	中下位正答率	メソッド	概要
Q1	86.0%	91.7%	なし	繰り返し
Q2	95.5%	95.3%	なし	条件分岐
Q3	76.0%	80.6%	なし	単純な 2 重繰り返し
Q4	57.5%	38.3%	含む	メソッド宣言のライティング
Q5	50.0%	43.9%	なし	複雑な繰り返し
Q6	54.3%	48.6%	含む	プログラムのリーディング
Q7	54.7%	37.2%	含む	メソッド宣言
Q8	64.7%	60.3%	なし	配列要素の抽出
Q9	45.3%	33.6%	なし	2 次元配列（発展問題）

表 -1　2009 年度の学期末試験の平均正答率（N=146）

テーマ	導入年度	概要
ベーシック	2010 年度	Java 標準の仕組みを使い，基礎的なプログラミングを学ぶ．市販の教科書のような題材が主．
スプレッドシート	2010 年度	講義用に用意した仕組みを使い，統計などの表データの処理や，コンピューターの内部の仕組みを学ぶ．
キャンバス	2010 年度	講義用に用意した仕組みを使い，画像処理やユーザインタフェースなどの主に視覚効果を学ぶ．
Web アプリケーション	2011 年度	講義用に用意した仕組みを使い，ネットワーク処理や Web アプリケーションを学ぶ．
音声	2011 年度	講義用に用意した仕組みを使い，ディジタル信号処理や音声の仕組みを学ぶ．

表 -2　仮想少人数クラス制課題のテーマ

プログラミング入門 1 では，以下のうちプログラミングに関するものを中心に学ぶ．

- LU#1001 システムと IT の概念
- LU#1303 コンピュータシステムのリテラシー
- LU#0431 統合開発環境の基礎

そのほかに，後続の授業と組み合わせて習得させることを目標とした項目として，プログラミング入門 1 で簡単な導入にとどめたものもある．

- LU#0130 手続き／イベントドリブン・プログラミング
- LU#0131 簡単なアルゴリズムの展開
- LU#0432 プログラミングスタイル
- LU#0703 システム開発プロセス

2010 年度にプログラミング入門 1 の授業計画を再設計するに当たり，前年度の学期末試験と課題提出率を分析した．2009 年度の学期末試験の結果は**表 -1** のとおりである．

ただし，全体正答率は学生全体の平均正答率を表し，中下位正答率は期末試験の総合得点順に学生を 4 等分し，上から 3 番目のグループに属する学生の平均正答率を表している．

この分析は，主に落ちこぼれる可能性のある学生がどこで躓くかを推定するためのものである．分析結果を検討し，学生がメソッドを十分に習得していないという仮説を立て，2010 年度にはメソッドを一連の授業の前半で教えるように授業計画を更新した．初年次のプログラミング教育としては，かなり早い段階でのメソッドの導入であろう．しかし，メソッドが重要であることを明示的に意識させ，何度も反復訓練させる効果を期待している．

● 仮想少人数クラス

プログラミング入門 1 は，90 分の授業時間 2 回分を連続して確保し，授業時間の前半では新しい内容の解説を行い，後半では解説した内容を習得するための演習を行う．後半の演習で取り組む課題を 5 種類（**表 -2**）用意し，学生の興味に合わせて取り組む課題を 1 つ以上選択させる．興味が異なる多様な学生に対応できるよう「仮想少人数クラス」を構成して演習に取り組ませている．

2010 年度の実績では，キャンバス，ベーシック，スプレッドシートの順に選択者が多かった．2 学科のうちのディジタルメディア学科への入学者は，CG に興味を持つものが多いことを反映している．また，理解の早い学生は，複数の課題を選択する傾向が見られた．カリキュラム改革以前には，そのような学生のために，別クラスの特訓クラスを設けていたが，結果としては，特訓コースよりもよりたくさんの課題をこなすことができ効果的であった．

● 授業外学習の支援

プログラミングを習得するには授業時間内だけでなく，授業時間外にも時間をとってさまざまなプログラムを書くことが効果的である．我々は，そのような授業外学習をサポートするため，次のような施策をとった．

- オンライン資料

授業で利用した資料はいつでも参照できるよう

にオンライン☆5で公開している．また，同資料には学習目標や具体的な指導手順を含めており，TA（Teaching Assistant：大学院生によるアシスタント）やSA（Student Assistant：学部生によるアシスタント）が学生を補助する際にも利用できる内容である．

- 学習計画の共通化
  プログラミング入門1は初年次に4クラスに対して開講している．それぞれのクラスで共通化された授業目標と教材を利用し，さらにそれを公開することで，異なるクラスの学生どうしでも情報交換を行いやすくしている．
- GBC（後ほど詳述）との連携
  SAから組織されるGBCと連携して，学生どうしでの学び合いも取り入れている．詳細は次の章で述べる．

● 成績評価基準の明確化

LUの明確化によって授業内容だけではなく，当該科目のそれぞれの単元の目標と，その目標達成を確認する方法を明確にした．具体的には，(a) 講義全体の目標設定に対する評定試験を最初に作成し，関連講義の担当者とレビューする，(b) 単元ごとに「〜ができるようになる」という形式で目標を記述し，対応する練習問題を教材に織り込む，という手法をとった．また，それぞれの目標は授業の各回の前後で学生に提示した．

従来の授業計画では科目間の連携が十分とは言えず，後続科目の最初の数回は復習という名目で授業に必要なプログラミングの項目を教えるという慣行が多少見受けられた．成績評価基準の明確化により科目間の連携が容易になり，また関連科目の担当者はプログラミング入門1の内容に対して要望を出しやすくなった．また，明確になった達成目標と評価基準を学生にも提示することで，何をどの程度学べたかが明確化したためか「この授業を履修してよかったですか」という項目について「はい」が75パーセントと高い評価を得た．

☆5 http://java2010.cis.k.hosei.ac.jp/

実績に基づく授業計画の見直し

2010年度には，毎回の授業の終わりに，学生に対して達成度の小項目ごとに主観的な理解度を自己申告させた．その結果，全体的に主観的な理解度は高くなかった．2010年度の期末試験と授業改善アンケートを分析したところ，前年度までと比較して次のような傾向が見られた．まず，授業内容の理解について，メソッドに関連する学期末試験の正答率は向上したが，代わりに8週目以降の教務内容である繰り返しや複合データに関する正答率が低下したことが分かった．つまり反復練習が理解に影響するのである．

これらの問題を解決するために，2011年度からは，「予習教材」を学生に頒布している．

予習教材は次回の授業に関するオンライン資料の位置の提示と，次回の授業内容に関する簡単な問題を出題している資料である．ただ漠然と「予習をしなさい」というだけでは効果は薄かったが，予習にもある程度のインストラクションを提示することで，予習をやりやすくした．予習に取り組んだ結果は本講義の評定に含めないことを明示しているが，提出率は8割以上を維持しており，「予習では分からなかったが，授業を聞いてよく理解した」という学生の声も上がっている．同じ教授方法でも，予習をさせると満足度が向上する傾向があるようだ．また，2011年度では新たに「認定試験」というものを導入した．従来の評定制度では評定試験や課題提出などを利用して単位認定の是非を判定していたが，従来の手法ではすべての単元の内容を漏れなく一定の達成度に到達していることを確認するのは困難であった．2010年度では，期末試験だけでその判定を行おうとした結果，我々が想定したよりも平均点が5点以上高くなり，最上位層の識別や単位認定の是非についての識別にやや難があった．2011年度では，単位認定の手法と評定の手法を明確に分離する．認定試験は100点満点で86点と合格点を高く設定し，必要なすべての単元をマスターしていることを確認できるようにする．評定試験は，理解度を測るもので，想定平均点を低く設定する．同制度の導入

授業名	年次	内容
プログラミング入門 1	1 年前期	値，変数，制御構造，関数（メソッド）など，命令型のプログラミングに共通する言語機能の理解と習得
プログラミング入門 2	2 年前期	オブジェクト指向や標準ライブラリを利用したアプリケーション設計，開発手法の理解と習得
プログラミング演習 1	1 年後期	プログラミング入門 1 で習得した内容を基にした小規模なアプリケーション開発演習
プログラミング演習 2	2 年後期	オブジェクト指向やアルゴリズムなどを取り入れた中規模なアプリケーション開発演習
プログラミング演習 3	3 年後期	専門領域特有のデータ構造やアルゴリズムを取り入れた中規模なアプリケーション開発演習

表 -3　プログラミングの授業

によって，単位取得時の達成度を保証しやすくなり，後続科目でプログラミング入門 1 の範囲の習得を前提とした授業計画をさらに立てやすくなることを期待する．

❑ 1 万行演習

我々が開講しているプログラミング授業の一覧は**表 -3** のとおりである．

演習を中心に，必修科目だけで卒業研究配属前までに，トータルで 1 万行程度のプログラムを作成させるように設計した．本学部では，この部分を「1 万行演習」と銘打って，学生自身にも，たくさんプログラムを書かなければならない，ということを自覚させている．プログラミング入門とプログラミング演習は，相補的である．プログラムの基礎を入門で学び，応用力を演習で身につけることになる．演習では，1 年後期から，1 つのプログラムで 1000 行を超えるものを作りあげることを体験させることで，断片化されたプログラミング技術の価値を体感させる．メソッドによる構造化の価値，複合データの有用性，コメントの重要性などを実感し，後続のプログラミング教育の設置意図を明確にする効果があると考えている．

専門科目の学びを通したコミュニケーション力と自主性の養成― GBC の取り組み―

カリキュラムや講義をどのように設計しても，一定数の学生は「うまく」＝「要領よく」勉強することができない．この原因の 1 つに，同級生どうし，同級生と下級生，学生と教員，学部生と大学院生など，さまざまな交流が不足していることがあげられる．

これらの問題を解決するため，オフィスアワー制度が導入された．しかし，利用者が少なかった．利用しない理由としては，教員がなかなかつかまらない，という意見も多かった．その一方で，学生が抱える問題は，学生が相談しようとする教員でなくても，他の教員や，場合によっては同級生や上級生でも解決できるものが多かった．そこで，オフィスアワーを実施する常設の場を用意することにした．物理的にも内容面でも，中での活動が外からも分かるようにと，ガラス箱オフィスアワーセンター（GBC）と名付けた．学生が講義や課題の内容，学生生活について気軽に相談できる場所を提供することが目的である．

❑ GBC の構成

GBC の構成は，SA 約 35 名，臨床心理士の資格を持つ相談員 2 名と情報科学部の専任教員であり，設置場所は学生ラウンジの一角で，文字通りガラス張りであり，外から中の様子がよく見えるようになっている．

GBC の運営主体は情報科学部 2 ～ 3 年生の SA である．これに補助的に 4 年生，大学院生が加わる．SA 全体をまとめる執行部といくつかのチームで構成されており，それぞれのチーム内でリーダを選出する．執行部とリーダは毎週ミーティングを開き，活動の方針などについて話し合う．活動は，月曜から金曜の各 2 ～ 5 限で，それぞれの時間帯に割り当てられたチーム 5 ～ 6 人で訪問者に対応する．

また GBC には臨床心理士の資格を持つ相談員が常駐している．教員とは違った視点で，進路や学生生活全般について相談にも対応する．訪問者だけでなく，GBC の SA にとっても良き相談相手である．教員にとっても学生の状況を早めに把握し，それに適切に対応する上で心強い存在になっている．

❑ GBC の役割

まず，オフィスアワー（教員が必ず研究室にいて質問を受け付ける時間帯）とは異なり，教員が自分の研究室ではなく GBC にいるという形態をとる．また，他の学年の学生にいろいろなことを質問や相談できる場の提供が，大きな役割となっている．部活動やサークルに属していない学生にとっては，先輩たちの経験談を聞いたり，アドバイスをもらったりできる貴重な場となっている．生活面でも，他の学年の学生，教員，相談員など年代の異なった層との対話ができることも大きな特色である．大学入学を機に一人暮らしを始めた学生にとっては，貴重な時間である．

また，訪問者，SA のどちらの立場でも，適切な言葉使いやマナーなどの TPO を意識したコミュニケーションが必要となってくる．これらのことも GBC に参加するうちに自然と求められるし，教え合える．学生の間に，ぜひ身につけてもらいたい．

❑ GBC の効果

GBC の SA の役割は当初，学生からの質問に答えるだけであったが，活動が進むにつれてさまざまな企画が出されるようになった．たとえば，学生と教員の交流会や，GBC SA による新入生への GBC 紹介，学生視点での研究室紹介などがあげられる．また，訪問者への接し方についてもお互い意見を出し合い改善する様子も見られる．SA はセメスターごとに公募で募集をするが，GBC を訪れた学生が後に SA になるケースも多い．また，以前 SA をしていた学生が 4 年生や大学院生になって GBC を訪れて，ごく自然にサポートしてくれることもある．まさに学生相互に支え合っている．

GBC がスタートしてほぼ 2 年だが，その間の顕著な効果として GBC SA の主体性と，学生どうしの輪が広がっていることがある．また，訪問者への対応の形態が 1 対 1 から多数対多数へと多岐に変化している．たとえば課題の相談に来た時間帯の担当の SA だけでは対応しきれない場合は，SA どうしで連絡をとって他の SA や教員に協力を求めて解決するというようなケースも増えている．他の学生も巻き込みながら活動を拡大していく方向に向かってきている．

GBC を訪問してくる学生の中で特に質問が多いのがプログラミングである．全体を通して，単純な全角半角のタイポなども多い．しかし，自分で考える力の有無で，抱えている問題を解決する割合が変わるようである．プログラミングに関しては，その場しのぎで GBC に行き丁寧に説明してもらっても，力が付くようなものではない．何度か訪問して自分で考える力がつくと解決するという様子が多々見られる．

1 年生向けのプログラミングは，初心者向けなので問題が具体的であり，課題には例題などで直接的な枠組みが与えられている．したがって，多くの場合改造する過程で分からないところが出てくるようである．たとえば，予想していた動作をしない場合，エラーが出ないようにするだけでは期待した動作が得られないことが多い．そういう時にそれ以上どうやってデバッグすればよいか分からない学生が多い．これについては，SA のちょっとしたアドバイスで解決する例が多く見られる．高学年向けの科目の課題では，自分でプログラム全体を設計しなければならない．自分で考える力をつけておかなければ，細部に目が行ってしまって全体像を見ることができないようである．

現在在籍している 1 年 2 年生は入学時から GBC があるので，ちょっとした疑問でも気楽に聞きにくる．上級生になるに従って，SA の中心が 2 ～ 3 年生であるため，下級生には相談しにくくなる傾向がある．3 年 4 年の場合は友人に SA がいれば GBC 訪問の障壁は比較的少ない．留年している学生は，課題には取り組んでいるが自力で理解し提出まで完遂することが得意でない人が多いようであり，そういう学生連中が相談する場として GBC が使われている．その場しのぎでやってくる学生，今さら友人・知人には恥ずかしくて，聞けないと思っている学生も多い．また，試験前で一気に疑問を解決するため，確認しにくる者も多い．SA 自身にとっても同級生

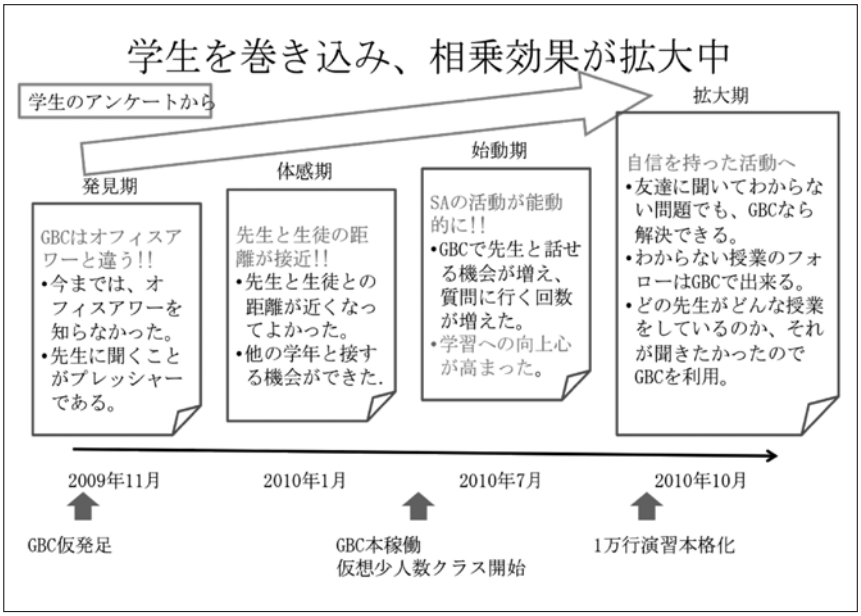

図 -1　GBC の活動による学生の意識の変化

の相談に対応することで，試験前の準備に役立つ場合もあるようである．このように GBC に行けば分からないとき何とかなると考えている学生が確実に増えている．

一方で，理解に不安があるのに自力では GBC にも訪れることができない学生（分からないけど，何を質問してよいのかも分からなくて申し訳ない，などという）も多いため，プログラミング入門などの重点科目では，GBC 主催の補習も行い参加者を集めている．

❑ GBC の今後

GBC 活動を通して，学生の意識も変わってきている．**図 -1** は，GBC 発足から約 1 年間，学生の GBC についてのアンケート調査の結果である．発見期でオフィスアワーとの違いを知り，実際に教員や他の学年と接する機会を実感する体感期，学生が能動的に活動をはじめ学習への向上心が高まり，その活動を通して自信をもって GBC での活動・利用を行う段階へと変化している．

GBC 担当者は GBC 活動の評価として，各期の試みやその結果をもとにフィードバックを行い，改善に役立てている．最近では上述の学生の意識の変化に伴い，積極的に SA からの提案が出てくるようになっていることも大きな効果である．教員からの提案に基づいて活動を行っていた受動的な初期からは，大きな変化があった．

今後の GBC 活動では，学生どうしの輪を広げることに注力したい．他の学年の学生や，相談員，教員との交流で身につけた経験を将来に活かしてほしい．また，授業の理解を促進するために，授業中の疑問や質問なども GBC で解決できるように，授業と一体となったシステム等を検討していきたい．GBC は基本的に向上心を持った学生によって構成されている．同じ向上心を持った学生どうしが集える場となるよう努めたい．また，このような場を通して，専門的な問題解決につながる自主性とコミュニケーション力が養成されることを期待している．

コミュニケーション講義

❑ グループ活動の機会を与える

GBC で学生の交流を増加させても，それだけでは，社会が要求するコミュニケーション力にはならないだろう．企業の多くの活動はグループで行われており，グループの中で他人の意見を聞きながら，自分の意見を述べることができ，全体のリーダーシップをとる人材が必要とされている．そこで，

我々は情報系学生向けのコミュニケーション能力の向上を目指した講義を企画することにした．孤立する学生を減らすこと，より積極的な意見を述べる手法を身に付けること，そして，企業の要望に合い，かつ，就職活動にも強い学生を育てることなど多岐にわたる目的を持った企画である．さらに情報系学生の教育という面から，一般的なコミュニケーション能力だけではなく，プログラミングの共同開発につながる能力を育成したいという気持ちがあり，次節に述べるようなコースを設計し，実践している．

❑ グループ討議の方法を学び，グループで活動する

基本コースは，3 日間のグループワーク集中講義のコースとして設計した．このコースでは，最初の 2 日間に，自己探求とコミュニケーション能力を高めるための実習と講義を行う．そして，2 日目の最後の講義から 3 日目にかけて，ロボットを使ったソフトウェアの共同開発を行う．本節では，この基本コースの設計思想と構成について，詳細に述べる．

前半の 2 日間は，外部講師に委託して，自己探求とグループ活動の基本について学ぶ．2 日間のグループワークを通して，自分と他者の違いに気付き，お互いを認め，受け入れ合うことを体験する．興味深い点は，そのグループ作りにある．初日の最初に，各自の学習スタイルに対するアンケートを行い，自己分析を行う．この結果をもとに，参加者それぞれが異なる特徴を持った学生を求めて，自らの手でグループ形成を行う．講義に参加する学生の中には，仲の良い友だちどうしが隣あって着席して講義を受け始める者も多いが，このグループ形成の過程を通して，日ごろはまったく知らない学生どうしがグループを組んでいるようである．ちなみに，グループワーク講義は修士の学生から学部 2 年生までが混在して参加しているため，多様な年齢構成のグループが形成されることになる．グループの規模は，おおよそ 6 ～ 7 名であって，このグループ単位で，以降の課題にチャレンジすることになる．

グループ課題は，いずれも全員参加型のプログラムである．個々の課題は，体験（DO），データ収集（LOOK），考察（THINK），変革（GROW）という体験学習型学習に構造化されている．学生にとっては，個人による思考時間とグループ討議が繰り返されることになり，1 日が終わると精神的な疲労感の大きい活動になる．結果として，「コンセンサス」「価値観」「コミュニケーション」「リーダーシップ」などを体得することができるコースに仕上がっている．

2 日目の午後の最後の講義からは，ロボットプログラムの共同開発に引き継がれる．グループワーク講義の設計当初においては，プログラムの共同開発の候補として，企業のアプリケーション開発を模擬したグループによる機能分担型の開発を行うことも検討した．しかし，他者とのコミュニケーションの重要性に気付いた学生に対して，プログラム開発だけに没頭させるのではなく，グループで議論しながら良い物づくりを目指してもらいたいという気持ちから，ロボットプログラムの開発を題材とした．講義で用いるロボットは，バンダイのネットタンサーであり，画像認識，音声認識，赤外線センサなどを備えた小型ロボットである．認識されたデータをイベントとして，簡易なスクリプト言語が提供されており，1 時間程度の講義で，基本動作を習得できることも利点であった．

各グループに与える課題は，グループ間で成果を競わせるかたちのものにしている．具体的には，迷路をすり抜ける速さを競ったり，迷路を走る相手チームのロボットの動きを邪魔するロボットを設置させたり，ビーチフラグのように後ろ向きにスタートしてゴールを目指してタイムを競わせたりするものである．**図 -2** に迷路を抜けるロボットと邪魔するロボットの課題についての例を示す．

ロボットプログラムの利点は下記の点にあると考えた．

- ロボットの動きという成果を目で見て確認できることから，グループの知識共有が容易になる．課題を具体的なロボットの動きとして議論できる．これにより，失敗から修正に至るまでの周期を短くすることができる．
- 競争の要素を組み込むことで，グループの連帯感

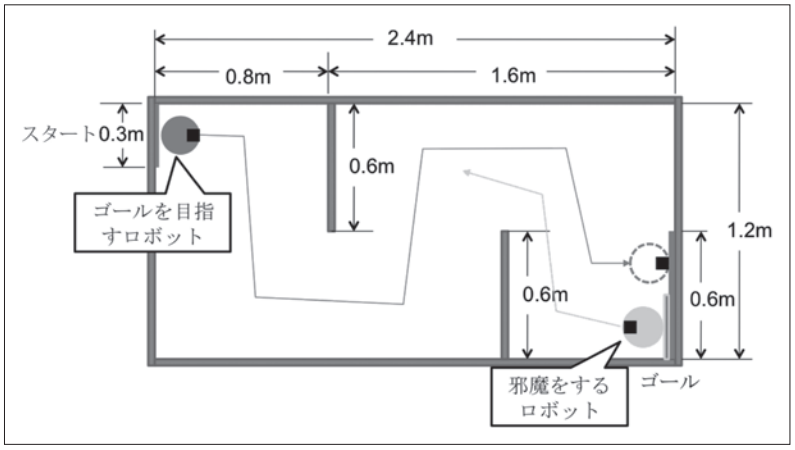

図 -2　ロボットの迷路走行に関する課題例

が高まり，興味を継続しやすくなる．

ただし，単なるプログラミングの競い合いにならないよう，講師を顧客に見立てて，事前の機能仕様書，詳細仕様書の作成と承認，テストの実施などをグループ構成員で分担を決めて行うことを課し，プログラムの共同開発も模擬体験できるように工夫した．

講義スケジュールとしては，2 日目の最後の講義で基本プログラミングを教え，課題を明示して，グループでロボットのテスト走行を実施させる．グループによっては，居残って翌日の戦略を考えるグループもあり，興味を持続させるという意味で，題材の選択は正しかったようである．

3 日目の大部分は，グループに分かれて，仕様作成とプログラム作成に費やされる．昼前に，一度だけ全員に集合をかけ，グループワークの観点についての振り返りを行う．自分が，どのようにグループに貢献しているのか，他人の意見を取り入れるよう努力しているのか，自分の意見を積極的に述べているかを再確認することで，午後の活動に向けた修正を行うことを意図している．そして，最後に，実際にグループごとの成果を披露するコンテストを実施する．実際のロボットだけに意外な結末に終わることも多く，学生に大いに楽しんでもらうことになる．

❑ 学生の反応と課題

受講した学生に対して，直後に行ったアンケート結果では，非常に高い満足度が得られていることが分かる．アンケート結果を**図 -3** に示す．

記述式の回答にも，学生の積極的な意見が表れている．興味深い感想について，**表 -4** に挙げてみた．

自分を見出すこと，相手を理解することに新たな価値観を感じて，グループワークの大切さを体感することに成功したことが，この感想の中から読み取れる．また，受講生との話の中で，受講前の期待は低く，受講後の満足度が高いという話が多く聞かれ，他の学生にももっと講義内容を分かりやすく伝えて，より多くの学生に受講してもらうべきだとの意見もあった．これらは，貴重な意見として参考にしていきたい．

ただし，以上の感想が受講直後の学生の感想であることも，明記しておきたい．明らかに，一時的に意識が高まった状態での回答であり，実際にこの後の活動にどう活かされたかについての追跡調査も

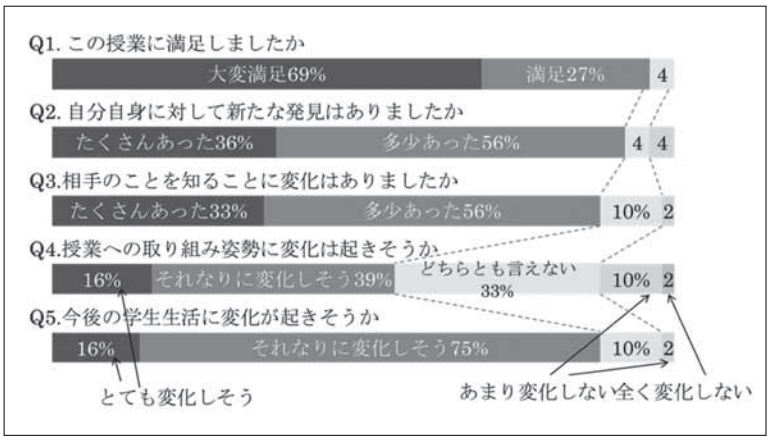

図 -3　受講後の選択式アンケートの結果

・グループで積極的にとても真剣に話し合って，何かを決めることができると知ることができた．正直に楽しかった．
・今まで話したことのないような人と話せ，コミュニケーションについて学ぶことができた．また，いろいろなことについてグループで話し合いをすることができた．
・久しぶりにこんな風に意見を言いました．小学校以来です．こんな風に話をすることもなくなっていたものですから，本当に楽しかったです．
・話すことが下手だと思っていたが，自分の予想以上に意志を人に伝えることができた．
・自分自身はどういった考えなのかが非常にはっきりと見えたことに加えて，同じグループから自分のことについて客観的に見てくれたことが，新たな発見でした．
・この人はどういう考えで主張しているか，どうして納得しよう，させようか？　相手の意見から，正しく知ることは大事だと思った．
・価値観が違っていても，説明によって理解できることが多い．自分の考えをはっきりと説明するのは誰でも難しい．
・普段の講義では絶対に体験できないグループでの話し合いができました．初対面でも，ここまで真剣に意見を言い合える場は初めてでした．

表 -4　受講後の記述式アンケートの回答例

必要になるであろう．

また，グループワーク講義は短期集中型のしかけであるが，継続的にグループワークを支援する体制も必要になる．前章で紹介した GBC も，その場の 1 つとなると考えている．実際，グループワーク講義に参加した SA が，GBC の運営にグループワーク講義で学んだコンセンサス形成などを自主的に応用している．さまざまな取り組みが折り重なって，学生のモチベーションを高め，卒業後の進路においても，リーダーシップを発揮できる学生に育っていくことを期待したい．

おわりに

最後に，本稿で述べた教育の評価方法について簡単に述べる．教育効果を測り，学士力の保証という観点では，学習の達成度／知識の定着度をはかる仕組みの導入を試みる．年度末に GRE[☆6] を受験させ，個人の成長や学部の教育効果を客観的に示す指標として利用可能かを検討する．情報処理推進機構の基本情報技術者試験も併用する．

また，これらが効果的に実施できているかどうかは，外部評価委員会を設置し，年に 2 回評価を受けている．メンバは産業界から 1 名，他大学（理系）から 1 名，学内他学部（文系）から 1 名となっている．この評価委員会では，発足時に，良いところを褒めることに重点を置くようにお願いした．この点が功を奏して，改革の推進の大きな支えとなっている．

GP 終了後も，プログラミング教育においては，本稿でも述べた実績に基づく授業計画の見直しを行ってゆく予定である．また，コース設計，教材作成，コースの連携などの方法は，数学や物理などの基礎科目にも波及させる予定である．

（2011 年 5 月 30 日受付）

☆6　米国の大学院入学希望者が受験しなければならない試験．

伊藤克亘（正会員）　it@fw.ipsj.or.jp
1993 年東京工業大学大学院理工学研究科博士課程修了．博士（工学）．2006 年より法政大学情報科学部教授．音とことばの情報処理の研究に従事．

荒川　傑　arakawa@gluegent.com
2007 年法政大学大学院情報科学研究科修士課程修了．2005 年より法政大学情報科学部の教育改革に陰ながら参加．現在は（株）グルージェントで分散システムやコンパイラなどの研究開発に従事．

善甫康成　zempo@hosei.ac.jp
1985 年広島大学大学院理学研究科物性学専攻博士課程修了．理学博士．住友化学（株）を経て，2009 年から法政大学情報科学部教授．材料設計，物性予測のための計算手法の開発および大規模並列計算技術の開発と応用などの研究に従事．

藤田　悟（正会員）　fujita_s@hosei.ac.jp
1989 年東京大学大学院工学系研究科電気工学専攻博士課程修了．工学博士．日本電気（株）を経て，2008 年から法政大学情報科学部教授．マルチエージェント，ヒューマンプローブなどの研究に従事．

解説

京大におけるLispを使ったプログラミング教育

湯淺太一　奥乃　博　尾形哲也
京都大学大学院情報学研究科

はじめに

Lispが世に現れたのは1950年代後半であり，現在も使われているプログラミング言語の中では，Fortranに次いで古い言語である．ハードウェアや実装技術の進歩，プログラミングパラダイムや応用分野からの要望など，さまざまな要因によって，数多くの方言が設計・実装されてきた．1980年代の人工知能ブームを契機に，いくつかの標準規格が作られている[8),9)]．当時と比べると，利用者の数こそ大幅に減少しているが，他に例を見ない強力な言語機能を備えているために，特定の応用分野では現在も実用に供されている．

京都大学工学部情報学科では，2004年度からLispを使ったプログラミング教育を行っている．講義科目としては，1年生後期の「アルゴリズムとデータ構造入門」（以下「データ構造」と略す）と2年生前期の「プログラミング言語」（以下「言語」と略す）があり，実験演習科目としては，3年生後期の「計算機科学実験及演習4」（以下「演習4」と略す）がある[4)]．本稿では，Lispの教育への利用を，これらの科目を実例として紹介する．

SICP

「データ構造」と「言語」は，教科書としてStructure and Interpretation of Computer Programs（略称SICP）[1)]を使ってプログラミングの講義を行っている．SICPは，基礎的なものから高水準なものまで，プログラミングのさまざまな概念をカバーしており，国内外の多くの大学でプログラミングの教科書として採用された実績を持っている．原著は英文であるが，邦訳もあり，また英文のフルテキストを出版社のWebページから無料でダウンロードすることもできる．例題や演習に使うプログラミング言語はLisp（正確には，Scheme[8)]）である．

SICPは次の5つの章から構成されている．

1. 手続きによる抽象化
2. データによる抽象化
3. モジュール化，オブジェクト，状態
4. 超言語による抽象化
5. レジスタ・マシンによる計算

このうち，「データ構造」が第1章と第2章を，「言語」が第3章と第4章をそれぞれ担当している．時間の都合もあるが，第5章の内容は2年生後期の「コンパイラ」の講義がある程度カバーしているので，「言語」の講義では扱っていない．

「データ構造」も「言語」も，毎週あるいは隔週で演習課題を出しており，受講生は実際にLisp処理系を使って課題を解く．講義には演習時間を設けていないので，受講生は各自の都合に合わせて計算機環境や処理系を選択する．演習結果はレポートとして提出し，TAが採点して返却する．一部の演習課題について講義で解説を行ったり，採点過程で気がついた事項を講評することもある．

レポートの提出は，原則としてメールで行う．メールに添付あるいは貼り付けられたプログラムは直ちに実行できるので，提出されたプログラムが

正しく動作するかどうかを確認できるからである．しかし，レポート課題の中には，データ構造や関数閉包内の変数環境などを図示するものもある．描画ツールできれいに描いたり，手書きの図をスキャナーで撮ってメイルに添付する学生が多いが，図示の課題がある場合は手渡しでもよいことにしている．スキャナーで撮るとサイズが大き過ぎてメイルに添付できないことがあるからである．

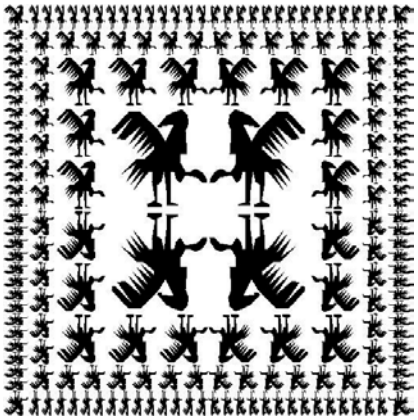
図 -1　図形言語による再帰的な描画例

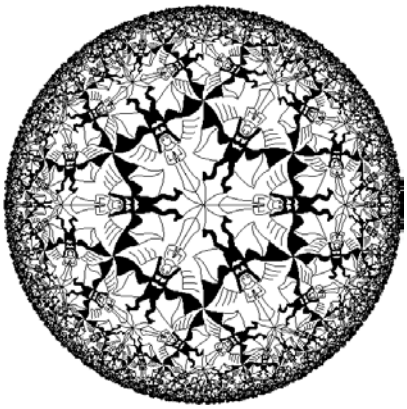
図 -2　キャンバスを円形にする

SICP は Lisp 言語を教えるための教科書ではない．新しい概念が出てきたときに，必要に応じて Lisp の機能を紹介する形式をとっている（たとえば，代入を行う `set!` は，第 3 章で代入の概念といっしょに紹介される）．だから，先頭から順に読み進めば自然と Lisp の言語も理解できるように工夫されている．しかし，講義を始めるにあたって，Lisp の全体像を受講生がある程度把握しておけるように，Lisp の簡単な入門講義を行っており，後述する演習用処理系の使用法をついでに説明している．

「データ構造」では，第 2 章までの知識を基礎にして，講義名にあるデータ構造とアルゴリズムに関する講義を行っている．また，学生のレベルに応じて，advanced な課題を出すこともある．SICP の第 2 章では，再帰的計算を画像生成に応用するための図形言語（**図 -1** 参照）が登場するが，描画キャンバスは正方形に限定されている．これを円形にする（**図 -2** 参照）とどうなるか，といった難題もある [3]．

ロボットプログラミング

「演習 4」では 6 件のテーマが用意されており，受講生はその中から 2 つを選択し，学期の前半と後半に 1 つずつ取り組む．テーマの 1 つである「ロボットプログラミング」[2] では，実ロボットのプログラミングを通じて，実世界と繋がったプログラミングを体験する．実ロボットとして，Lego Mindstorms[6] を使っている．ブロック玩具で知られる Lego 社が販売しているロボットキットで，8bit CPU を搭載した RCX というコントローラ・ブロックに，モータや各種センサのブロックを組み合わせることによって，ロボットを自作できる（**図 -3** 参照）．

受講生には，次のような課題がいくつか与えられる．

【演習○】

光センサを 1 つだけ使用して黒い線に沿って動くロボットを作成せよ．さらに，黒い線のコース上に障害物を置いたときに，それをタッチセンサで検出し，回避してコースに復帰する機能を付加せよ．

課題の中には，やや難易度の高い次のような「自由課題」も含まれている．

【自由課題○】

迷路課題（省略）において，ロボットがゴールに到着した後，スタート地点まで“効率的に”帰還するプログラムを，リスト操作を使用して作成せよ．

演習に使用する言語は，NQC[7] と XS Lisp[11] である．NQC（Not Quite C）は，C 言語の機能を大胆に削り落とし，ロボット部品との API を加え，マルチスレッド機能を追加したものである．これによって，RCX の 32Kbyte という極小のメモリ空間でも，C 言語風のロボット制御プログラムを動作させることができる．NQC で書かれたプログラムは，フロントエンドの PC でクロス・コンパイルされ，バイナ

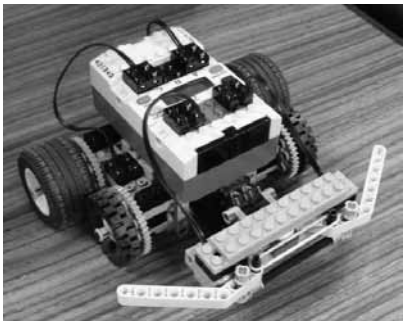

図 -3　移動ロボットの例

リを RCX に送り込んで実行される．

もう一方の XS Lisp は，筆者の一人が Mindstorms 用に開発した超小型の Lisp 処理系である（“XS” は，eXtra Small の意）．RCX の中で Lisp インタプリタが動作し，フロントエンド PC から Lisp の式を 1 つ受け取っては解釈実行する．結果はフロントエンドのディスプレイに表示され，あたかも通常の Lisp 処理系を使っているような感覚でプログラムを開発することができる．言語仕様は，文献 8)の仕様をベースに，大幅に機能削減し，ロボット部品を制御する関数を追加している．NQC よりもはるかにメモリの制約が厳しいために，マルチスレッドではなく，非同期イベントのための割り込み処理構文 `with-handler` を備えている．

(with-handler ((*event*$_1$. *handler*$_1$))
　　...
　　((*event*$_n$. *handler*$_n$))
　. *body*)

この式を実行すると，基本的に本体 *body* が実行されるが，その間に各条件式 *event*$_i$ が定期的に評価され，その値が真になると本体の実行に割り込んで，対応する処理 *handler*$_i$ を実行する．処理が終わると，再度本体の実行に戻る．

演習では，それぞれの課題について，NQC と XS の両方でプログラムを書く．静的な手続き型言語と動的な関数型言語を同時に使用することによって，これらの相違を体得することが重要な狙いである．また，これらの言語で書いたプログラムを実際のロボットで動作させることによって，アルゴリズム以外に，時間やインタラクションの概念を扱うことができるマルチスレッドや割込みといった動的制御テクニックを学ぶことが期待されている．

JAKLD

講義科目の「データ構造」も「言語」も，JAKLD[10] という Lisp 処理系を使って演習問題を解くことを奨励している．

SICP による講義を始めた当初は，自家製の処理系を改造して，描画機能として Tcl/Tk の Tk 機能を追加した TUTScheme/Tk[5] を用意し，これを計算機センターの PC にインストールした．しかしセンターの利用時間には制限があり，多くの学生は自分の PC に TUTScheme/Tk をインストールしようとした．TUTScheme/Tk は C 言語で記述されており，もともと Unix 系 OS の上で動作していた．これを受講生たちが自分の Windows PC で動かすのは容易ではなく，本来の演習に専念できない状態が続いていた．

このような状況で数年間講義を行ってきて，その間に明らかになったさまざまな不都合を解消し，受講生が演習問題を解くことに専念できるような環境を提供するために，Java で記述した JAKLD に手を加え，演習に必要な諸機能を追加した．演習用の PC にどんな OS が動いていようと，Java VM さえあれば，jar ファイルをダウンロードするだけで直ちに起動することができる．携帯電話で使えるバージョンもある．

JAKLD に追加した機能としては，図形言語はもちろんのこと，SICP の第 3 章で使われる並列処理機能（実際はマルチスレッド機能）や，第 4 章で必要となる遅延評価機構も含まれる．これらはすべて Java の標準的なライブラリを利用して実装されている．また，処理系依存の振る舞いは，できるだけ SICP 本文の例題と一致するよう調整し，学生が違和感なく演習できる環境を提供している．

SICP の第 4 章「超言語による抽象化」には，いくつかのプログラミングパラダイムを実現する metacircular なインタプリタが登場する．これら

のインタプリタはLispで記述され，パラダイムを採用したLisp風言語のプログラムを解釈実行する．実行エンジンはインタプリタごとに異なるが，基本的なLisp関数は，インタプリタが動作する処理系のものを流用する．流用するためのコードは，処理系が提供する関数群に依存する．演習用の処理系をJAKLDに統一することによって，各インタプリタのコードがすべての演習用PCで共通に使えることになった．

ちなみに，昨年度の「データ構造」の受講生の1人が，「自由課題」としてJAKLDをAndroidスマートフォンでも使えるようにした．AndroSchemeと名付けており，Androidマーケットから無償で入手可能である．

おわりに

「Lispによる」教育についての原稿を依頼されたということは，Lispでプログラミング教育を行うのは珍しいのであろう．C言語やJavaといったポピュラーなプログラミング言語を使うのが一般的かもしれない．限られた誌面で，京都大学で行っているLispを使った教育を簡単に紹介してきたが，その範囲で「なぜLispなの？」に答えたいと思う．

言うまでもなく，これらの講義・演習は，Lispプログラマを養成するためのものではない．Lispという具体的な言語を通して，プログラミングを学習するためのものである．ここで，プログラミングというのは，単に与えられた問題をコーディングする技術ではなく，計算機科学の成果であるさまざまな概念や機能を駆使できる能力である．この意味で，講義で採用しているSICPは，きわめて適切な教科書である．プログラミング言語の学習にあまり時間をかけずに，抽象化，metacircular，遅延評価，並列実行，制約プログラミングといった高度な概念や言語機能を学習することができる．SICPは，これらを単に概念として紹介するのではなく，Lispでどのように実現されているかを具体的に解説している．これによって，学生たちは，次々と紹介される概念を，より深く学習することが期待される．

レポートの演習課題を解くために対話的なLisp処理系を利用できる点も，学習効率を高めている．SICPの最初の演習課題（Exercise 1.1）は，(+ 5 3 4)などのLisp式の値を実際の処理系を使って求めよ，というものである．学生たちは処理系を起動して，“(+ 5 3 4)”と打ち込むだけでよい．同じことをC言語やJavaで行えば，初心者の学生にとって結果を得るまでにどれほどの時間がかかるだろうか．

演習に使っているXS Lispについても，同様の学習効率をねらっている．Lisp処理系に共通する対話機能を利用して，関数を定義しては直ちに（コンパイラを呼び出すことなく）実行して動作確認する，といったプログラム開発方法が使える．さらに，マニュアルに書いてあるモータやセンサのためのAPIも，式を入力してテストすれば，簡単に動作が確認できる．

参考文献

1) Abelson, H. and Sussman, G. J. : Structure and Interpretation of Computer Programs, 2nd Edition, http://mitpress.mit.edu/sicp/, MIT Press, Cambridge, MA (1996).
2) 尾形哲也：計算機科学実験及演習4—ロボットプログラミング，http://winnie.kuis.kyoto-u.ac.jp/~ogata/le4-robot/wiki/
3) 奥乃　博：「アルゴリズムとデータ構造入門」講義情報，http://winnie.kuis.kyoto-u.ac.jp/~okuno/Lecture/10/IntroAlgDs/
4) 京都大学工学部シラバス，http://www.t.kyoto-u.ac.jp/syllabus-s/
5) 小宮常康：TUTScheme, TUTScheme/Tkの処理系，http://www.spa.is.uec.ac.jp/~komiya/download/
6) LEGO.com Mindstorms: Home, http://mindstorms.lego.com/
7) Not Quite C, http://bricxcc.sourceforge.net/nqc/
8) Revised(4) Report on the Algorithmic Language Scheme, http://people.csail.mit.edu/jaffer/r4rs_toc.html
9) Steele, G. L. Jr. : Common Lisp the Language, Second Edition, http://www.cs.cmu.edu/Groups/AI/html/cltl/cltl2.html
10) 湯淺太一：Javaアプリケーション組込み用のLispドライバ，http://www.yuasa.kuis.kyoto-u.ac.jp/~yuasa/jakld/index-j.html
11) 湯淺太一：XS: Lego MindStorms用Lisp処理系，http://www.xslisp.com/index-j.html

(2011年5月31日受付)

湯淺太一（正会員） yuasa@i.kyoto-u.ac.jp
1977年京都大学理学部卒業．同大数理解析研究所，豊橋技術科学大学を経て，現職．プログラミング言語処理系の研究・開発に従事．

奥乃　博（正会員） okuno@i.kyoto-u.ac.jp
1972年東京大学教養学部基礎科学科卒業，博士（工学）．NTT，JST，東京理科大学を経て，現職．プログラミング言語，人工知能，音環境理解の研究に従事．

尾形哲也（正会員） ogata@i.kyoto-u.ac.jp
1993年早稲田大学理工学部卒業．学振特別研究員，理化学研究所を経て，2005年より京都大学情報学研究科准教授．さきがけ「情報環境と人」研究員．神経回路モデルと人間ロボットのコミュニケーションの発達研究に従事．