

解説



## 情報処理専門教育について

## コンピュータリテラシー教育の一事例†

安村 通見†† 有澤 誠†† 齋藤 信男††

## 1. はじめに

「理工系の学生の製造業離れ」ということが、最近問題視されている。理工系の中でも情報工学あるいは計算機科学（コンピュータサイエンス）系の学生がメーカーよりも金融・証券などに進出する例が多いようであり、理工系の先生方およびメーカーの人々はこの現象を深刻に受け止めている。たしかに、給与や福利厚生などで製造業を中心とするメーカーの改善努力も必要ではある。しかし、この現象は見方を変えれば、理工系、特にコンピュータサイエンスを学んだ学生を製造業以外の分野で必要としていることの証であり、メーカーの人々はともかく、理工系の先生にとってはむしろ歓迎すべきだと思う。

一方、「受験生の理工系学部離れ」の現象も指摘されている。入学した後、暗くきつく厳しい理工系よりも、楽して卒業できる文系を望む受験生が増えているそうである。さらに、メーカーでは人手不足のために、文系の学生を採用して、採用後社内教育を行った後、情報処理関係の仕事につかせる例も出てきている。この問題は、いろいろな課題を含んでいるが、少なくとも一ついえることは、21世紀を目前にした今日の大学で文系の学生にも、一般情報処理教育を始める時期にきている、ということである。すなわち、コンピュータを道具として使うコンピュータリテラシーの教育とコンピュータサイエンスの方法論が、理工系の中だけで閉じこもらずに、広く文系の諸学部の中にも実質的に取り込まれていく必要があることを意味している。

慶應義塾大学では、21世紀を担う学生に対する新しい大学教育の場として、湘南藤沢の地（SFC: Shonan Fujisawa Campus）に総合政策学部と環境情報学部の2学部を、1990年4月に新たに開設した。両学部合わせて約1000名の学生の1年生全員に情報処理言語教育を必修として行っている。キャンパス内に張り巡らされたネットワークとその下のUNIXワークステーションを用いた本格的なコンピュータリテラシー教育である。両学部1年生へのコンピュータ教育は、コンピュータ専門教育への予備段階というよりも、一般情報処理教育としての性格のほうが強い。しかも、入ってくる新入生は、受験で数学か英語の選択のうち、数学を選択したものは3割弱である。このような学生に対しての、コンピュータ教育のあり方はいかにあるべきか、その理念と約1年の経験を紹介して、今後このようなコンピュータ教育のあり方に対する議論の題材を提供する。

## 2. 一般情報処理教育の課題とSFCの考え方

冒頭で述べたように、文系を含む一般学生に対して、一般情報処理教育を始める必要があるが、その際に考慮しておくべき点はいくつかある。非常に大きな問題として、高等教育の大衆化とその結果としての大学のレジャーランド化の問題があるが、これに対しては、学生の知的関心を新しい時代にふさわしく、いかに引き出すかで対処していく必要がある。しかし、ここでは、あまりこの問題には触れないことにする。一般情報処理教育特有の問題としては、(1)機械オンチや数学嫌いを含む学生の質の問題、(2)インフラストラクチャの整備、(3)新しい教育の進め方の方法論の議論、などがあるが、これらをわれわれがどのようにしていったかを事例として示そう。

慶應義塾大学湘南藤沢キャンパス(SFC)の設立

† A Case Study of Computer Literacy Education by Michiaki YASUMURA, Makoto ARISAWA and Nobuo SAITO (Keio University at Shonan Fujisawa, Faculty of Environmental Information).

†† 慶應義塾大学環境情報学部

の理念として5項目の基本方針<sup>1)</sup>があり、その下に具体的には、人工言語(情報処理言語)がSFCにおいて自然言語と並ぶ車の両輪の一つとして重視されている。SFCにおいては、一般教育科目と専門教育を分水嶺的に分断するのではなく、くさび型に専門が入り込む形をとる。一般教育科目、専門教育科目とも選択科目を多くして、必修科目を減らしている。コンピュータ教育の理念として、

- (1) コンピュータを電子文房具として手軽に使うことを学ぶ
- (2) プログラミングの基本概念や技法を知り、人工言語を用いてコンピュータを自分の目的のために利用することを学ぶ
- (3) 以上を通じて、コンピュータリテラシーという基礎的能力を培うことになっている。

この理念の具体化として、1年生で、総合政策、環境情報の両学部の必修科目として、情報処理言語Iがあり、2年時からは、両学部共通の選択科目として情報処理言語IIが始まる。専門教育は、2、3、4年時に学部専門科目が始まる。さらに、SFCでは、キャンパスネットワークを整備して、ワークステーションあるいは、ラップトップから世界中の研究者・学生といつでも情報交換ができるようになっている。

### 3. コンピュータリテラシー教育の実態

以上述べてきた SFC における情報処理教育の理念に基づいて、最初にSFCにおけるコンピュータの環境(インフラストラクチャ)がどのようなものを述べた後、情報処理言語Iでは、実際にどのようなカリキュラムを組んで授業を行っているかを述べる。

#### 3.1 SFC キャンパスネットワーク

SFC では、FDDI 100 Mbps の基幹デジタルネットワークの下に、各種の分散サーバ(ネットワークサーバ、ファイルサーバ、メールサーバ、プリントサーバ、アーカイブファイルサーバ、データベースサーバ、など)と、教育用、研究用、図書館用、および事務用の4つの(論理的な)サブネットワークがつながっている。教育用サブネットワークには、一般教育用、専門教育用、言語教育用のワークステーションなどがつながっており、研究用のサブネットワークには、一般研究用、あるいは、機能別のワークステーションがつながっている(図-1参照)。各種のワークステーション、サーバなどは、すべてマルチベンダ方式をとり、したがって各社の製品が入っている<sup>2)</sup>。

各ワークステーションはほとんどが、UNIXワークステーションであり、標準的には、次のよ

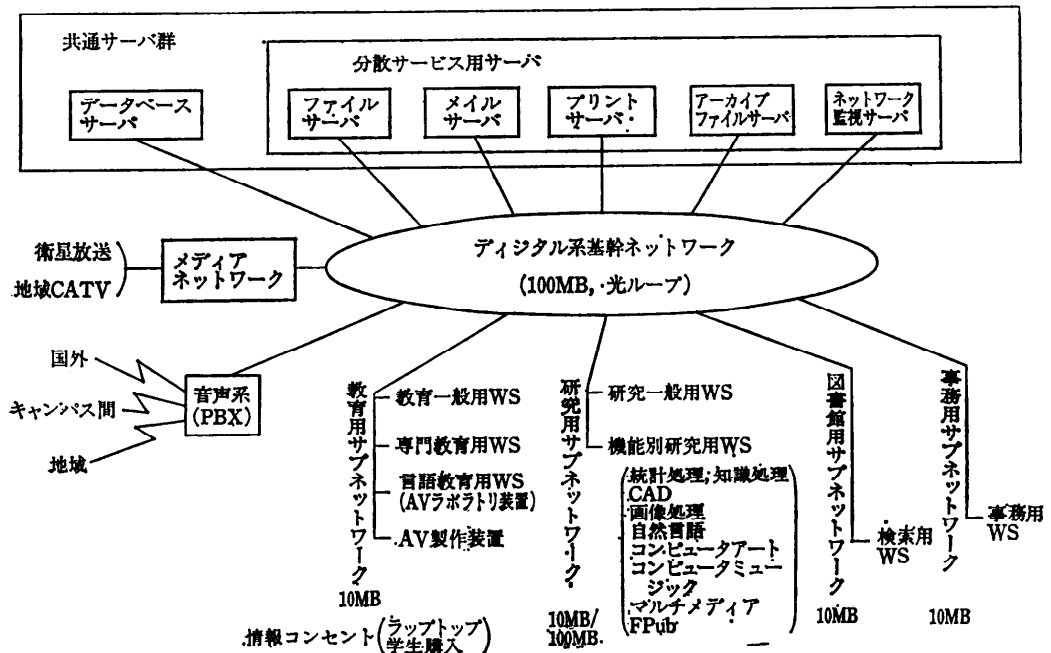


図-1 湘南藤沢キャンパスネットワーク構成

うな環境で動かしている<sup>3)</sup>。

- システム: UNIX 4.3 BSD 相当
- ユーザインタフェース: X window
- 分散ファイルシステム: NFS
- 日本語エディタ: Emacs,  
仮名漢字変換: EGG (Wnn)
- 電子メール: Emacs (mhe),  
電子ニュース: Emacs (gnus)
- 清書システム: 日本語 LaTeX,  
図形編集: TeXDraw

現状 (1991年4月時点) では、一般教育用として、UNIX ワークステーションが 40 台×5 教室と、一般研究用として、UNIX ワークステーションが各教員の個室に入っている。

### 3.2 情報処理言語 I の構成

情報処理言語 I は、両学部の 1 年生に共通の必修科目であり、1 年間で 4 単位が与えられる。SFC は、春学期と秋学期との 2 学期制をとっているが、自然言語 (語学) の教育が秋学期から、本格的に始まることもあり、人工言語の教育は春学期のほうに重みがある。すなわち、各学期とも、講義が週 1 回 90 分が 15 回分あるが、このほかに、春学期は週 2 回の自習 (講師と TA\* がつく) と秋学期は週 1 回の自習 (やはり講師と TA つき) がある。

両学部合わせて、約 1000 人の学生をまったく区別せずに、約 35 人程度を 1 クラスとして編成している。講義は 4 クラスないし 8 クラス程度まとめて大教室で行い、自習はワークステーションを設置した特別教室で約 35 人単位で行っている。

### 3.3 情報処理言語 I のカリキュラム

初年度の情報処理言語 I では、春学期は、(狭い意味での) コンピュータリテラシーとプログラミング (Pascal を使用) の初歩を学ぶ。秋学期は、プログラミングを通じた問題解決を学び、ミニプロジェクトを行う。すなわち、各学期をさらに半分のクォーターに分け、各クォーターごとの講義と自習を次のように割り当てた:

クォーター	講義/自習
1 (春・前半)	情報社会とコンピュータ/ コンピュータリテラシー
2 (春・後半)	プログラミング入門/ プログラミング入門

- 3 (秋・前半) プログラミングによる問題解決/  
プログラミングによる問題解決
- 4 (秋・後半) コンピュータの利用環境/  
ミニプロジェクト

春学期の前半 (第 1 クォーター) は、講義として、情報社会とは、から始まって、コンピュータとは何かについて概要を学ぶ。自習のほうは、その間、コンピュータを道具として使えるように一連の基礎的な訓練を積み、日本語の文書が作れて、メールが出せるようになるまで覚える。春学期の後半 (第 2 クォーター) は、講義、自習ともプログラミング (Pascal) の初歩を学ぶ。

秋学期の前半 (第 3 クォーター) は、まず講義は、情報処理の初歩として、アルゴリズムなどを学び、自習のほうは、Pascal が実際に書けるようになることを目指す。後半 (第 4 クォーター) は、講義では、コンピュータの応用分野や先端分野などを学び、自習のほうでは、グループでのミニプロジェクトを体験する。

### 3.4 情報処理言語 I の実際

第 1 クォーターで特徴的なのは、まず、タッチタイプによるキーボード教育と、Emacs と LaTeX の教育である。タッチタイプを、5 時間程度時間をかけて訓練させることにより、後の、Emacs エディタや仮名漢字変換、あるいは、プログラミング教育がスムーズにいく。また、LaTeX はかならずしも学生にとって易しくはないが、他の科目などで LaTeX でのレポート課題を出されると、学生は必死になって覚え、春学期の後半にはほぼ全員が一応使えるレベルに達した。

第 2 クォーターでは、Pascal によるプログラミングで、Pascal 言語の概要を理解させ、簡単なプログラムの修正や拡張ができることを目標にしたが、第 1 クォーターのコンピュータリテラシーに比べて、学生がなぜプログラミングをやらなければならないのかなど目的意識の点で戸惑いをみせる場合もあった。

第 3 クォーターでは、自習では Pascal の簡単なプログラムが書けるようになることが目標であり、講義では、コンピュータサイエンスの入門編ともいべきデータ構造や再帰、アルゴリズムの初歩などを教えた。プログラミングでは、クラス分けをしていないせいもあるが、できる学生とできない学生の差が大きい。また、自習の教え方に

\* Teaching Assistant

関しては、担当者の自由裁量を大きくした。

第4クォーターは、自習では、グループごとにプログラミングや、調査などのミニプロジェクトを行ったが、同時に報告書の書き方や発表の仕方でも学んだ。講義のほうでは、コンピュータがどのように使われている、また、どのような可能性をもっているか学んだ。

### 3.5 情報処理言語 I の教材

情報処理言語 I の教材として、タッチタイプの練習用として、タイピングプログラムを用いたほか、(1)SFC-CNS ローカルガイド<sup>3)</sup>と、(2)SFC Pascal ガイド<sup>4)</sup>の二つの独自教材を新たに作った。

タイピングのプログラムは、各ステップごとに練習問題が出てきて、タイプの打鍵速度と正確さを出力するようになっている。CNS ローカルガイドは、A4 287 頁のレファレンスガイドであり、UNIX の基本から、ウィンドウシステム、Emacs エディタの使い方、LaTeX の使い方、電子メールの使い方などが書かれている。Pascal ガイドは、A4 137 頁の Pascal レファレンスガイドで、Pascal 言語の解説(講義ノット形式)、Pascal 言語のレファレンス、およびシステムガイドからなる。

また、6月末に、SFC 全体で統一したラップトップ(モデムカード付きのブック型の MS-DOS パソコン)の一括購入を決め、夏休み前に約 6 割の学生が購入した。このラップトップは日本語辞書を内蔵しているので、大学側では日本語 Emacs と通信ソフト(Kermit)の入ったフロッピーを同時に配った。このラップトップは、当初は UNIX の動くものという要求があったが、大学までの携帯に便利な重さと経済性を考えて、初年度は MS-DOS 型のパソコンに落ち着いた。使い方としては、自宅でワープロとして文書を作ったり、プログラムを作ってきたり、あるいは、モデムで大学のコンピュータを使ったり、さらに、大学にもってきて、ネットワークに直接接続して使ったりなどが可能である。

### 3.6 情報処理言語 I の評価

初年度の講義と自習は、第1クォーターと第4クォーターは、互いにほぼ独立しており、第2、第3クォーターは、関連づけられている。講義と自習の関連をもっと密にすべきだという意見もあるが、自習と講義が別の曜日のため、進度が必ず

しもうまく合わず、せいぜい緩い関連づけをするのが精いっぱいであった。

第2クォーターと第3クォーターの間に夏休みが入るために、せっかく覚えた Pascal を多くの学生は忘れてしまうようである。また、第1クォーターは、電子文房具ということで、学生にとっては直接的な学習の目標が分かりやすいが、第2クォーターから始まるプログラミングは、なぜこれを学ぶかという目的がいま一つ、学生に十分理解されなかったようである。春学期の終りには、大部分の学生が電子メールを読んだり書いたりし、LaTeX で文書が作れるようになったのは予想を越えるものであった。プログラミングのほうは第2クォーターの間は、難しいとかよく分からないとかの声が大きかったが、第3クォーターに入ると、レポートや練習問題がたくさん出されるようになると、否応なくプログラミングに励むようになり、できる学生とできない学生の間のギャップが埋まってきた。さらに、ミニプロジェクトが始まると、急にプログラミングの重要性が分かったようである。

以上の評価を踏まえ、開設2年目の情報処理言語 I は、プログラミングを秋学期に回し、春学期はコンピュータリテラシーを十分時間をとってやる予定である。すなわち、1991年度の情報処理言語 I のカリキュラムは、次のように組み替えた：

クォーター	講義/自習
1(春・前半)	情報社会とコンピュータ/ コンピュータリテラシー入門編
2(春・後半)	コンピュータの利用環境/ コンピュータリテラシー発展編
3(秋・前半)	プログラミング入門/ プログラミング入門
4(秋・後半)	プログラミングによる問題解決/ ミニプロジェクト

また、プログラミングに用いる言語は、1年目は教育的配慮から Pascal を用いたが、2年目は、グラフィックスをもっと利用したいとの学生の要望を入れて、Cを用いる予定である。

ラップトップについては、ワープロとしてどの程度使われているかは確認できていないが、少なくともモデムを通した大学のコンピュータへのアクセスは土曜の真夜中でも回線がいっぱいになるほどよく使われている。また、現在、フリーソフ

トウェアの Pascal や C がラップトップ上に欲しいという声も強く、なんらかの対応が必要である。ネットワークとの (Annex による) 直接接続は、最近できたばかりであり、今後利用が増えるものと思われる。今年度の情報処理言語Ⅱの授業などで、ラップトップを教室に持ち込んで、講義と実習が一体となった学習も始まっている。

自習 (特に春学期) の場合、専任の教員だけではまかない切れないので、学外から非常勤講師をお願いする一方で、理工学部の大学院生に TA として活躍していただいた。アメリカではごくふつうの TA 制度が日本で普及していないのは、予算の問題その他いろいろな課題があるからだろうが、別のアルバイトで時間を費やすよりは、自分の専門を活かした TA 制度は、今後日本でももっと広く採用されてしかるべきである。

なお、SFC では、学生による授業の評価制度を新年度から全科目にわたって始めている。これは、アメリカなどではごく普通に行われていることではあるが、我が国ではまだその例はほとんどない。評価は、13項目にわたる5段階の評点と、自由記入欄とからなり、各先生は、自分の授業の評価結果と全体の平均とを受け取り、次の学期での授業の改善の参考とする。情報処理言語Ⅰでも、講義と自習のそれぞれについて、学生の評価を受けた。

### 3.7 情報処理言語Ⅱの予定

2年次より、両学部の2年生向けに選択科目として情報処理言語Ⅱの講義が始まる。これは、情報処理言語Ⅰでのコンピュータリテラシー教育を踏まえて、「電子文房具を越えたハイパーリテラシーを学ぶ」という共通のテーマの下に、次のようなコースなどを用意している:

- 統計コース
- プログラミングコース
- システムコース
- 応用ソフトウェアコース
- アートコース
- グラフィックスコース
- ミュージックコース

各コースとも半年間に自習を含めて15回程度学ぶが、受講者の数は春秋合わせて全学生の過半数〜2/3弱程度となるものと見込んでいる。なお、学生はこれらのコースの一つ以上を選択すること

ができるが、二つ目の科目から卒業・進級の単位とはならない (自由科目となる)。

## 4. いくつかの経験と反省

以上、慶應大学湘南藤沢キャンパス (SFC) におけるコンピュータ教育の概要を述べてきたが、実際に、このカリキュラムを設計し、実行していく過程でいくつかの問題点が浮かび上がってきた。これらの問題点を手がかりに、コンピュータリテラシー教育の課題とその解決策を検討する。

### 4.1 コンピュータリテラシーとは何か

自然言語におけるリテラシーは、字や文章を読んだり書いたりする能力であり、比較の意味ははっきりしている。コンピュータにおけるリテラシーは必ずしも確立した概念ではない。しかし、21世紀において、情報がコンピュータを用いて水や電気などと同じように日常的に、かつ簡便に利用されるようになるとすると、そのとき自然語と同じようにその読み書き能力が問われるようになる。

そこで、まずコンピュータリテラシーとは何かを定義しておこう。コンピュータリテラシーといったとき、広い意味では、コンピュータを用いた情報のアクセス能力をいい、狭い意味では、コンピュータを道具として利用できる能力をいう。(コンピュータリテラシーの定義をここではやや抽象的に与えたが、具体的にコンピュータリテラシーが何を意味するかは、対象となるユーザにより変わってくる。)

まず、このうち、道具としてコンピュータを用いるというコンピュータリテラシーの教育の必要性は現在のワープロや電子メールの効用と利用のされ方を考えることで、ほぼ理解できるだろう。この場合の道具は、電子文房具とよぶことができる。典型的な電子文房具としては、ワープロ (欧文/日本語)、清書システム、作図・作表システム、電子メール (一般に通信ソフト)、データベースなどがある。

狭い意味でのコンピュータリテラシー教育、すなわち、コンピュータを道具として使うための教育は今後大いに普及するであろう。

### 4.2 プログラミング教育は必要か

現在でもコンピュータはある意味では、日常的に使われている。たとえば、いろいろな家電製品

などには、マイクロコンピュータが埋め込まれていることが多いし、ワープロやゲームマシンでは、単機能とはいえ、立派にソフトウェアつきでコンピュータが動作している。また、パソコンなどでも、実際の使われ方からすると、ワープロとしてとか、ゲームのプログラムとしてとか、あるいは、表計算だけといった、使われ方も少なくない。

ところで、コンピュータが一般の機械と異なる点は、ソフトウェアを変えることによって、さまざまな動作をさせることができる点にある。単一の動作をさせるだけなら、普通の機械と変わりがない。また、将来において、今とまったく同じようにコンピュータが使われるというよりも、今はかなり違った使われ方が現れる可能性のほうがはるかに高い。たとえば、企業におけるコンピュータの利用法も、かつての EDP の時代から、ネットワークの時代、さらに、戦略情報システム (SIS\*) といったように、大きく変化している。

交通機関をたとえにとり、比喩的にいえば、コンピュータを道具として使うのは、バスや電車を利用することであり、本当に行きたいところに行けるようになるには、やはり自分で運転できる、すなわち、プログラミングができるようになる必要がある。

#### 4.3 プログラミングとして何を教えるか

プログラミングというと、すぐにプログラミング言語の教育になり、それだけで終わりがちであるが、ここでは、できるだけプログラミングを通じて問題解決をする、ということを意識的に行う必要がある。つまり、問題を定式化して、それを解くというプロセスを理解させるわけである。このとき大事なのは、かならずしも既存の言語にこだわらない、という点である。問題を明確に定義し、その解法を論理的に (日常言語でも良いから) 書き表すことができるようになる、ことを目標にする。

実際には、コンピュータ上でそれを確かめる必要があるから、そのときプログラミング言語で表せれば良い、とする。ただし、ある程度はプログラミング言語を知り、使えるようになることも大事である。しかし、これが最終的な目的でないことにいつも注意を払う必要がある。

プログラミングやプログラミング言語を教える際、われわれ、コンピュータサイエンスの人間は、ついつい、数学的な例題を使いがちであるが、これは十分注意してできるだけ数学的知識を必要としない例題に選ぶ必要がある。しかも、抽象的な例よりもできるだけ具体的な例が良い。たとえば、ただ平均を求めろ、というのではなく、野球チームの打撃の平均を求めよとか、条件にしても、結婚の条件を判断させるとかすることにより、学生の興味のもち方がかなり変わってくる。

もう一つ注意すべきことは、できるだけゆっくり教えることと、すぐにできないからといってあきらめずに十分時間をかけて覚えさせる、ということである。この点は、いまの受験制度や教育制度にも問題があり、大学1年生は、いままで、できるだけ短い時間で解答を出すことを繰り返し教育されてきたため、じっくり考えて答を出すことに、慣れていないようである。自分の知っているキーワードを並べて文章を作ったり、連想で解答を選ぶ能力は大変発達しているが、それだけでは困るわけで、プログラミングに限らず論理的な思考力を養うにはじっくり時間をかけて考える訓練を重ねるしかない。われわれのモットーは、「ゆっくり、楽しくプログラミングを学ぼう」ということになっている。

#### 4.4 コンピュータサイエンスは教えるべきか

コンピュータを道具として使い、簡単なプログラミング言語が使えればそれでいいではないか、という意見もある。つまり、コンピュータサイエンスは非専門家には教えるべきでないという意見である。自然言語 (語学) でも、言葉を操れば良いのであって、文化とか、文学論は展開すべきでない、という考えに基づくようである。

これはおそらく、技術とか技法だけを教える専門学校の場合であれば、正しいであろう。しかし、大学におけるコンピュータ教育の場合、将来、コンピュータの新しい利用法が現れたときでも、大学で学んだことの応用がきき、さらには、自ら新しい応用を切り開く、そういった学生を育てるためには、コンピュータを使った情報処理の非常に基本的な考え方は伝えておく必要がある。

\* Strategic Information System

#### 4.5 なぜワークステーションか

一般教育におけるコンピュータの教育では、パソコンを用いた教育が（少なくとも我が国では）圧倒的に多い。確かに、パソコンのほうが手軽で、初期コストも安く、使い勝手もワークステーションよりも悪くはない。逆に、ワークステーションは初期コストもかかるし、使い勝手も必ずしも良くない。にもかかわらず、われわれがあえてUNIXワークステーションで一般教育を行おうとしたのは、(1)事実上の標準になりつつある、(2)パソコンに比べ速度が速く、メモリ空間も広い、(3)標準的なソフトウェアが多く、しかも比較的廉価（場合により無料）、(4)分散システムに必須のネットワークに強い、などの理由による。

実際には、システムのセットアップや保守など問題がないわけではないが、いまのところ、学生も教職員も徐々にワークステーションに慣れつつある。SFCでは、学生のほうが先に覚えて、先生が学生から教えてもらうという、ほほえましい風景も見られる。

#### 5. おわりに

KEIO SFCにおけるコンピュータリテラシー教育はまだ始まったばかりであり、試行錯誤の点がないとはいえない。しかし、現状でもSFCの学生はサークルの連絡などを電子メールで日常的に行い、LaTeXで政治学や文化人類学のレポートを出し、苦勞しつつもPascalのプログラミングを覚えつつある。また、ミニプロジェクトでは、プログラミング能力としては稚拙でも、自分たちの問題意識に基づいたプロジェクトをやり遂げている。

コンピュータを専門とする学生に対する教育法だけでなく、コンピュータは専門ではないが、将来なんらかの形で、コンピュータに関わりをもつであろう学生に対する教育法も、そろそろ真剣に

考えてもいいのではないかと思う。このような観点で、慶應義塾大学湘南藤沢のコンピュータ教育を一つの例題として取り上げ、今後の議論の素材としてここに報告した。

**謝辞** SFCにおけるコンピュータ教育は多くの人の献身的な努力によって構築されつつある。ネットワークの構築に多大な努力をされた楠本、加藤、村井の各氏、講義や自習で指導されている徳田、千代倉、富田、萩野、立木、渡辺ほかの方々、非常勤講師の方々、TAとして参加されている大学院生の方々、相磯学部長はじめ、いろいろなサポートをしてくださっている方々に、深く感謝したい。

#### 参考文献

- 1) 慶應義塾大学: KEIO University Shonan Fujisawa Campus (1990).
- 2) 慶應義塾大学: KEIO SFC GUIDE (1990, 1991).
- 3) 慶應義塾大学: SFC-CNS WG, SFC-CNS ローカルガイド (1990, 1991).
- 4) 慶應義塾大学湘南藤沢キャンパス人工言語カリキュラム小委員会: SFC Pascal ガイド (1990).
- 5) 慶應義塾大学湘南藤沢キャンパス: 「慶應義塾大学湘南藤沢キャンパスに本格的キャンパスネットワーク構築」新聞発表 (1990. 9. 5).
- 6) Denning, P. J., Comer, D. E., Greis, D., Mulder, M. C., Tucker, A., Turner, A. J. and Young, P. R.: *Computing as a Discipline*, C. ACM, Vol. 32, No. 1 (Jan. 1989). 邦訳, 木村 泉訳, 学問としての計算機分野, 情報処理, Vol. 31, No. 10 (Oct. 1990).
- 7) 野口正一, 中森真理雄: 大学等における情報処理教育の諸問題, 情報処理, Vol. 31, No. 10 (Oct. 1990).
- 8) 安村通見, 有澤 誠, 斎藤信男: KEIO SFC におけるコンピュータリテラシー教育, 第 32 回プログラミングシンポジウム予稿集 (Jan. 1991).
- 9) Dijkstra, E. W. et al.: *A Debate on Teaching Computing Science*, C. ACM, Vol. 32, No. 12 (Dec. 1989). 邦訳, 安村通見訳: 計算機科学に関する討論, bit (Apr., May 1991).
- 10) 村岡他編: 知のキャンパス, 共立出版, bit 別冊 (Apr. 1991).

(平成 3 年 4 月 24 日受付)



**安村 通晃 (正会員)**

1947年生。1971年東京大学理学部物理学科卒業。1975~1977年UCLA留学。1978年東京大学理学系大学院博士課程(情報科学専攻)

満了。(株)日立製作所中央研究所主任研究員を経て、1990年4月より慶應義塾大学環境情報学部助教授。理学博士。ベクトルコンパイラ、Lisp処理系などの研究に従事。プログラミングシステム、並列ソフトウェア、ヒューマンインタフェース等に関心をもつ。日本ソフトウェア科学会、ACM、IEEE Computer Society各会員。



**有澤 誠 (正会員)**

1944年生。1967年東京大学工学部計数工学科卒業。電子技術総合研究所、山梨大学工学部計算機科学科を経て、現在慶應義塾大学環境情報

学部に勤務。工学博士。ソフトウェア工学、特にソフトウェアの評価、アルゴリズムの解析、オブジェクト指向システムなどに興味をもっている。



**斎藤 信男 (正会員)**

昭和15年生。昭和39年東京大学工学部計数工学科卒業。昭和41年同大学院修士課程修了。工学博士。

昭和41年電子技術総合研究所、昭和49年筑波大学電子情報工学系、昭和53年慶應義塾大学工学部数理科学科、同大学理工学部数理科学科教授を経て、現在同大学環境情報学部教授。昭和56年カーネギーメロン大学計算機科学科客員研究員。オペレーティングシステム、並列処理、分散処理、ソフトウェア開発環境、文書処理などに興味をもつ。電子情報通信学会、計測自動制御学会、日本ソフトウェア科学会、ACM、IEEE Computer Society各会員。

