プログラミング教育を振り返って

Looking back on Programming Education

森田　彦

1．はじめに

筆者は 1991 年の社会情報学部開設以来，情報処理基礎教育，特にプログラミング教育に携わって来た．本学着任までは，物理学の研究者や学生の集団に所属していたので，それまでとは異なる学生の気質や興味・関心に戸惑いながらも，試行錯誤を続けながら教育を行って来た．そこで試行した教育上の様々な試みは折に触れて本学部紀要にまとめて来たが，今振り返ってみると，それらにはプログラミング教育に特有なもののみならず，今の大学教育全般に通底する普遍的な内容も少なからず含まれているように感ずる．そこで，この機会にプログラミング教育に関して行って来た取り組みを俯瞰し，それらを現在の視点から総括しておくことは一定の意味があるのでと考え，本稿をまとめることにした．以下では，着任時から現在に至るまでのプログラミング教育上の取り組みを経年的に辿る形をとりながら，これまでの取り組みの要点を提示して行くことにする．

2．文系学生はプログラミングに不向きか？

本学部で筆者が最初に担当した科目は，１年次学生対象の「情報処理」という科目であった．これは，コンピュータを使用する演習科目で，ワードプロセッサを用いた文書処理，表計算ソフトを用いたデータ処理，そして BASIC 言語を用いたプログラミング学習の３つの部分から構成されていた．この科目に関する全般的な成果や課題については文献（森田･新國，1992）に譲り，ここでは，その中の BASIC 言語によるプログラミング学習について振り返ってみる．

社会情報学部は，文理融合型の教育を目指していたので，文系総合大学である本学にあって，理系指向の学生も一定数入学していた．初年度は，文系･理系･その他が５：４：１の割合であった．この数値は，高校時に所属していた進路別クラスを尋ねたアンケート調査の結果である．以下，前二者をそれぞれ理系学生，文系学生と便宜的に呼ぶことにする．我々教員側としては，基礎的な学習を行う限り，習熟度や理解度にこれら理系・文系志向の違いはあまり影響しないものと見込んでいた．実際，本科目でも文書作成や表計算処理の単元では，あまり文系・理系の別を気にすることはなかった．しかし，BASIC プログラミング学習に入った段階で，文系学生達の間で「僕は文系だからプログラミングには向かない….」などというような発言を頻繁に耳にするようになった．この点，気にはなったものの，単なる先入観だろうと当初はあまり深刻には受け止めていなかった．

ところが，学期末に，アンケート調査を実施して学習単元毎の理解度を尋ねたところ，文書作成や表計算処理の単元では，理解できたと回答した学生の割合が理系・文系で大きな差はなかったのに対して，プログラミング

Morita Hiko　札幌学院大学社会情報学部

の単元では，理系学生が 72.2%だったのに対し，文系グループでは 45.9%と差が大きく開いている事が分かった．(ibid., 1992)やはり，文系志向の学生にはプログラミングは難しいのだろうか…？　これが，プログラミング教育に際して我々が最初に抱いた不安である．その翌年度には本格的なプログラミング科目の開講が控えていたので，何としてもこの〝開き〟の原因を捉えておく必要があった．

そこで，アンケートの結果をもう少し詳細に分析してみることにした．アンケート調査では，BASIC の学習単元毎に理解度を尋ねていたので，それら単元毎の理解度を理系・文系別に分析してみることにしたのである．その結果，次のような，顕著な傾向があることが分かった．まず，基本的な約束事や書式等の習得を目指す導入的な第 1 ・ 2 章では理系・文系で理解度に大きな差はなかった．ところが，分岐処理や反復処理など制御構造を学習する第 3 章に進んだ段階で理解・文系で大きな差が生まれていたのである．これはアルゴリズムの理解にも関わるところなので，もし，文系学生にとってアルゴリズムの理解が難しいという傾向があるのであれば，理解できる結果ではある．ところが，応用編として行った，第 4 章のグラフィックスの章では，両者の格差が大きく縮まっていたのである．プログラミングする量や複雑さという点では後者のグラフィックスの部分の方が増大しているにもかかわらず，である（森田・新國・原田，1993）．

この傾向を，演習指導時の学生の反応等を思い起こしながら考察した結果，第 3 章では，数学的な題材を主に使用していた点が文系学生の理解度低下につながった可能性があることに気づいた．例えば，分岐処理の例題として 2 次方程式の解を判別式が正負の場合に分けて表示する課題，反復処理の例題としてユークリッドの互除法を用いて最大公約数を求める課題などである．もちろん，数学に苦手意識を持つ学生がいることは想定してはいた．そこで，テキストには式と手順を図解も交えながら具体的に示し，言わばそのままプログラムとして記述すれば良い様に配慮していた．そのため，筆者としては数学の理解に多少の難があっても問題ないと捉えていたのである．ところが，「わぁー，数学の問題だ！」とため息を吐きながら課題を行っていた文系学生の反応を思い起こすと，その嫌悪感から，プログラミング課題そのものの理解に至らなかった可能性が高いのでは，と思い至った．一方，第 4 章のグラフィックスの課題では，画面に表示した適当な数の小円の色を，ランダムに白色と背景色（黒色）に切り替えることで，沖の漁り火の様子を再現するものや，円と円弧を描く命令を駆使して，ドラえもんの顔を描画するなどの，視覚的楽しさを前面に出した題材を用意した．ただし，描画のための新しい命令の使用が必要になり，また，指定通り描くためのアルゴリズムも第 3 章のものよりはかなり複雑になる．実際，理系学生は，この複雑さを嫌い，第 3 章から 4 章にかけて理解度が低下している．ところが，興味深いことに，文系学生は，数学的な要素が含まれていないこの単元で逆に理解度が上がり，結果として第 4 章では文系・理系別で理解度の差がほとんどなくなっていたのである（ibid., 1993）．

この分析から，題材とした問題内容そのものに嫌悪感を持っている，あるいは関心を持てない場合は，プログラミングそのものの理解度も低下してしまうという結論に達した．上の場合は，理解の阻害要因として，数学的要素が考えられ，これが，第 3 章で理系・文系学生間の理解度の差を広げた主原因であると理解した．そして，数学要素を廃した題材にすると，第 4 章のように，文系学生と理系学生の間でプログラミングの理解度に大きな差は生じないという，言わば後の学習指導上のヒントを得ることができた．

これ以降，プログラミングの課題を与える際，必要性がある場合を除いて数学的な題材は用いないようにした．そうすれば，プログラミング学習の本質である，アルゴリズムに対する理解については，文系・理系の区別なく習得できると考えたからである．そして，以降の学習では実際その通りに進んだ．

3．アルゴリズムを考える際のつまずきとは？

1992 年度から，「プログラミング」という科目が開講された．これは，講義と演習が 2 コマ続きで行われる科目で筆者は演習を担当した．

この科目で，我々はアルゴリズムの理解に重点を置いた教育を目指し，上で経験したことを生かして，採り上げる素材としては数学的な知識を要する内容は極力廃し，日常的な題材を扱うことにした．これにより，文系学生の戸惑いは見られなくなり，順調に講義・演習は進んでいるように見えた．しかし，学期の中間で実施したアルゴリズム理解度チェックテストで，学生が示した思わぬつまずきに遭遇することになる．

このテストは，150 円の入場切符を売る自動販売機の処理を，流れ図の一つである PAD（Problem Analysis Diagram）で記述させるというものである．我々の意図としては，金額が 150 円に達しない間は料金の投入を求め，150 円を超えたら，切符販売完了のメッセージとおつりを表示する，という反復処理の理解度をチェックするために課した課題であった．この出題は 2 段階で行い，第 1 段階では具体的なメッセージの表示の仕方等は学生の裁量に任せる形で行った．続いてその後に，第 2 段階として今度は不足金額の投入を促すメッセージや金額投入後に切符を出す際のメッセージなど，いわゆる仕様の詳細を指定して，それに沿ったアルゴリズムを解答させた．ここでは，便宜的に前者をテスト 1，後者をテスト 2 と呼ぶことにする．

類似したアルゴリズムに関する課題は講義・演習でやっており，それらはほとんどの学生がこなしていたので，大方の学生が解答できるものと見込んでいた．ところが，テスト 1 では正答は 2 割程度で，一部不完全ながら基本部分は記述できている解答を含めても約半数に留まった．一方，テスト 2 では，基本部分の記述ができた解答は 8 割に上った．もちろん，後者の方が高成績になるのは当たり前だとしても，この変化の大きさは我々の予想を超えたものであった．

この点の分析の詳細は文献（森田・新國・原田，1993）に譲り，ここではその要点を整理しておく．まず，テスト 1 においては，不足金額を要求するメッセージやおつりの提示の仕方などの仕様を自分で決めなければならない，つまり設計しなければならなかったため，その考察範囲の広さ，およびどこまで考察範囲を限定するかの判断に戸惑った点に原因があったようである．例えば，お釣りの 50 円を 50 円玉 1 枚で出すのかあるいは 10 年玉 5 枚で出すのか，というこの問題の本質とはあまり関係のない部分で悩んでいた学生もいたようである．一方テスト 2 では，どういうメッセージをどういう流れで出すのかという仕様を指定したため，学生は反復処理の記述に専念する事ができた．それが出来た学生が 8 割程度だったということは，所定のアルゴリズムを理解していた学生はその程度いたことを示している．

このことは，所定のアルゴリズムを学習する際，その本質に当たる部分に集中できるように仕様あるいは設計部分を指定しなければ，学生達が本質部分以外で悩んで，結果的につまずいてしまう可能性があるということを示唆している．このことを教訓として，以降，課題を与える際には，学習ポイント以外の部分は具体的に説明あるいは指示して当該アルゴリズムの理解に集中できる様に配慮す

ることにした．この点については，後に一部の教員から「問題をモデル化してアルゴリズムの全体構造を設計する能力は重要であり，そのトレーニングが十分なされないのはプログラミング教育上好ましくない．」というご指摘を頂いた．もっともなご指摘である．しかし，我々としては，導入期の学習で多くの学生がつまずいては，それ以降の専門的学習の選択の幅も狭めてしまうと考え，できる限り多くの学生が一定のレベルまで到達できるようにすることを優先した．もとより，このような教育方針に関しては，どちらが正しいというような問題ではなく，学部全体の教育方針という観点から議論されるべきことである．その観点からすると，今から思えば，科目担当者の考えのみではなく，もっと関係者で広く意見交換して，どのような学生をどのように育成するか，という点を共有できたらより有益であったと考えている．

ともあれ，導入期の学習におけるつまずきが以降の学習に影響を与える可能性については，図らずも次節で述べる形で改めて考えさせられることになる．

4．コンピュータに対する苦手意識はプログラミングの理解に影響するか？

1997 年度から，高校で「情報基礎」を学習した学生が入学するようになった．また，本学情報処理実習室の PC 環境も同年度から OS として Windows NT を導入し，プログラミングを始めとする情報処理演習以外の幅広い教育で PC が活用されるようになった．このような環境変化の一方で，「今まであまり PC にさわったことはないが，将来必要だから学んでおいた方が良いのでは…．」という動機で本学部に入学して来る学生が増えたように感じた．つまり，入学時点で必ずしも PC 操作を得意としない学生が増えて来たのである．

プログラミング演習の現場においては，前節までに述べたような工夫や改善の成果もあって，この頃にはもはや「文系だからプログラミングに向いていない．」というような声は，ほとんど聴かれなくなっていた．しかし，上に述べた事情を反映して今度は，「自分はコンピュータが苦手だからプログラミングはできない…．」というような声が目立ち始めたのである．確かにプログラミングを行うためにはコンピュータの操作が前提ではあるものの，基本操作を身につければあとは，論理的処理手順であるアルゴリズムの理解に重点が置かれる．それ故，指導する側からみれば，このような〝嘆き〟は的外れに思えたのだが，少なからぬ受講生がそのような感覚を持っているように見える以上，何らかの分析をする必要があると考えていた．ちょうどその頃，私情協の研究会で，愛知教育大学でコンピュータに対する意識調査が行われていることを知った．そして，その中のコンピュータに対する不安尺度のアンケート項目（仲谷・金子，1995）と我々のアルゴリズム理解度チェックテストとの相関をとれば，何らかの傾向がつかめるものと考えた．

その分析の詳細は文献（森田，1998）に譲るとして，主要な分析結果は，６月に行った最初の理解度チェックテストの結果と，７月に行ったコンピュータに対する不安尺度アンケートの回答結果との間に最も強い相関があるという点である．ここに，調査を行ったプログラミング演習科目（２年次学生対象）では，テストを前期２回，後期２回の全４回，アンケートを学期開始時の４月とテスト時の計５回行った．当初は，４月時点の苦手意識が６月実施の第１回理解チェックテストに影響を与えるものと予想していた．しかし，分析結果は，６月の第１回テストの成績が良かった学生の苦手意識が７月時点で４月と比較して減少し，同時に，成績が悪かった学生の苦手意識が増大したことを示していた．そ

の結果，6月のテスト成績と7月の苦手意識の間に最も強い相関がみられるようになっていたのである．この結果は，6月の(最初の)テスト成績を反映する形で，学生の苦手意識が再構成されたことを示唆している．さらに，そこで形成された苦手意識の基本的傾向は学期末の12月まで維持されており，2回目以降のテスト成績は，あまり苦手意識の形成に影響しないようであった．

この分析結果が示唆しているのは，導入期のプログラミング学習の理解度がコンピュータに対する苦手意識を形成し，それが一定期間維持される，あるいは固定されてしまう，という点である．そうすると，再構成された苦手意識がその後の学習の理解度にも影響している可能性があり，我々はこの結果から，導入期の学習における理解度がその後のコンピュータに対する意識さらには学習姿勢にも影響を与える可能性があると考えた．当時はコンピュータを活用する科目が増えて行く時期だったので，プログラミング学習を通じて形成された苦手意識がそれらの科目の学習にも影響を与えることが危惧され，我々はこの結果を重く受け止めた．そうして，学習指導に当たっては導入期のハードルを下げて，ある程度の理解度あるいは理解できたという成功体験を獲得してから徐々に難易度を上げて行くという学習パターンが必要だと考えるようになった．以降，筆者は図1の実線のような学習曲線を念頭に，学習スケジュールを立てるようにしている．ここに，横軸は学習経過時間，縦軸は学習到達レベルである．意味するところは，時間経過と共に線形（単調）に学習レベルを上げるのではなく，当初はレベルを抑え気味にして，その分，学期後半にレベルを上げて目標レベルまでの到達を目指すというものである．模式図なので，定量性には乏しいが概念としては有効だと考えている．

なお，この傾向は，学生全体を見た場合の大まかな傾向であり，苦手意識が最初のテストで増大したグループ，変わらないグループ，そして低下したグループに分けて調べると，各々のグループで異なった特徴を持っていることが分かった．その詳細については，文献（ibid.，1998）を参照されたい．

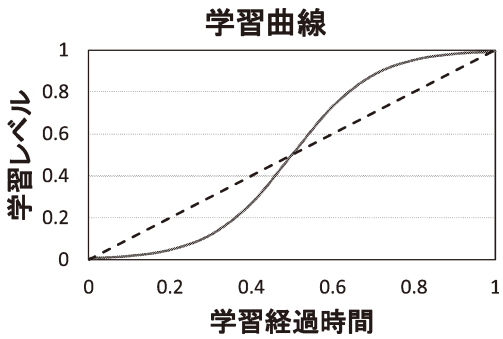

図1　目指すべき学習曲線

5．教室内の学生は三層状態？

演習時における学生の反応に関しては，もう一つ気になる点があった．それは，全員が課題に集中している活気に満ちた週と，過半の学生が中々課題学習に集中していない状態に陥る週とが，目まぐるしく移り変わる，という点である．具体的に言えば，ある週は8割程度の学生が予定より早めに課題を仕上げたので，全般的に軌道に乗って来たと思っていたら，翌週は過半の学生が予定進度に到達せず，あきらめムードが演習室に漂ってしまうというような具合である．学習内容に応じて進度状況にある程度の差があるのは当然だとしても，指導する側からすればその変化の幅が大きく，課題の分量や難易度のわずかな違いが演習時の雰囲気を大きく左右しているように見えた．一体，このような変化のメカニズムは何であろうか．当時は150名から200名近くの受講生を指導していたので，演習の円滑かつ安定的な運営を図る上でも，少し分析してみる必要性を感じた．

そこで，演習時の学生の課題の進み具合や

取り組む姿勢などを俯瞰的に観察し続けた結果，受講生は，周りに影響されず着実に課題を消化できる上位層（仮にこう呼ぶことにする），課題に集中できず結果として課題をこなせない下位層，そして学習進度が周囲の状況に影響される中位層の三層からなっているという考えに到達した．この中位層は浮動層とも言えるグループで，周りがやり出したらそれに引っ張られて課題に集中し，逆に周りの進度が思わしくないとそれに引きずられて出来なくても仕方ないという態度に陥ってしまう層である．集団を上位・中位・下位の三層に分類すること自体は一般的なことだが，ここで特徴的な点は，中間層が容易に上位層側あるいは下位層側に流れる傾向を持っているという点である．演習時の観察から，これらの割合が２：６：２程度の割合で構成されていると仮定すると，それまでの演習時の変化が理解しやすい事に気づいた．典型的な例で述べると次のようになる．まず，この中位層が課題をやらなければならないという雰囲気を感じ取った際には，上位層側にくっつき，８割程度の学生が課題を消化する．一方，課題が思った様に進まないという雰囲気が醸成されると中位層が下位層側に引きずられて，８割程度の学生が予定の課題をこなせない，という状態に陥ってしまう，という具合である．

これはかなり単純化したモデルであるが，もしこれが一定程度学生の傾向を捉えているとすると，演習全体の雰囲気を良い方向に誘導するためには，この中位層に照準を当てて，彼らが上位層側に移動するように指導を行えば良いことになる．実は，それまで筆者は，下位層の学生を引き上げることが全体の底上げにつながる，という考えで，特に進度が遅れている学生を集中的に指導していた．こういった学生が伸びない限り演習はうまく行かない，という考えの上に立ってのことだったが，それが全体の底上げにつながることはなく，徒労感を感じていた．しかし，そういった学生の指導が必要だということは認めつつも，受講生全体の底上げには，中位層を念頭に置いた指導が必要だと切り分けて考える事で，筆者の場合気が楽になり，指導がしやすくなった．ただし，これは履修者が 100 名を超えるような場合のことで，履修者数がそれ程多くない場合は，下位層の底上げは効果的と考えている．

６．学生の２極分化の顕在化 ── その原因および有効な対策は？ ──

プログラミング演習では，当初は大学院生からなる TA（Teaching Assistant）を配置して演習指導を運営していたが，2002 年度より，本学部生からなる SA（Student Assistant）を採用･配置する事にした．そして，2006 年度からは TA の採用をやめ，SA のみからなる演習指導体制に移行した．また，2001 年度の学部カリキュラム改訂に連動して，2002 年度より全員がノートパソコンを所持し，それを教育に活用することになった．これを受けて，同年度からプログラミング演習は学生が所持する携帯 PC を用いて，一般教室で行う様になった．

このような大きな学習環境の変化があったものの，プログラミング演習は順調に進んでいた．しかし，2006 年度あたりから，学期中に行うテスト成績の分布が低得点側へかたより始めて，つまり低得点者の割合が増大して来て，2009 年度にはついに 50 点未満の割合が約半数に達するに至った．これは，前節で述べた三層モデルの観点からすると，中間層が大きく下位層に引きずられた状態になっていることを意味しており，このままでは一定の水準を維持することが困難になる恐れがあった．担当教員としては，学習内容のレベルを同水準に保っているので，これは学生側の変化と考えられる．そうすると，学生に適合するように教育指導方法を何らかの形で改

善しなければならないことは明らかだった．その問題意識から幾つかの試みを行ったが，その詳細は文献（森田，2011）で紹介している．ここでは，その後の視点も加えながら以下にそれらの取り組みを整理してみることにする．

まず，最初に行った対応は，相対的に成績が下位だった学生の動向を把握することである．幸い，プログラミング演習では，TA および SA（2006 年度からは SA のみ）を活用していたので，指示通りに課題をこなして行ける学生の指導は彼らに任せ，筆者は理解度が低いと思われる学生に目を配ることができた．詳細な観察の結果分かったことは，成績下位グループに位置すると思われた学生でも，個別に指導すると過半の学生は課題をこなすことができる，ということである．その意味で彼らには潜在的な能力はあると捉えた．しかし，少し目を離すとまた課題が滞るという事態に陥ってしまう．その原因が筆者には中々理解できなかったので，指導した学生に個別に尋ねてみたところ，みな一様に「自分１人ではうまく勉強できない．」という答が返って来た．要するに，学習の仕方が分からない，ということである．

筆者としては，この時点でようやく事態がつかめてきた．つまり，以前は，学習ポイント毎に「テキストあるいは配布プリントを理解できるまでよく読みなさい．」と指示することで，大半の学生が理解を進める事ができたのだが，そういう一般的な指示では的確に学習を進められない層が増えてきたということである．ただし，個別に学習ポイントを指示し，それに対する学習の進め方を具体的に指示すると大半の学生は理解を向上させる事ができるという実態も把握していた．そこで，改善の一つの方策は，学習内容を理解するための具体的な方法を指示あるいは提供する，ということであるという結論に到達した．

7．Web テストの導入

上に述べた様な考察は，学生の二極分化傾向が現れ出した 2007 年度あたりから進めていた．そうして，学生が理解度を主体的に向上できるようになるための方策を検討した結果，自動採点機能を持った Web テストを用意し，どの程度の理解度に達しているか，あるいはどこが理解できていないのかを各自が確認しながら学習を進められる環境の構築が望ましいという結論に達した．そこで，学習単元毎にそのポイントとなる部分の理解度を問う設問を Web 上に用意し，制限時間内にそれらに解答すると自動採点した結果を返してくれる，という Web アプリケーションシステムの構築に取りかかった．2007 年度に大枠部分を森田が作成し，詳細な制作は 2008 年度から森田ゼミ卒業研究生の原が行った．2008 年度に何度か試行した後に，2009 年度に本格的に受講生に公開した（原，2009）．

前節で述べた様に，2009 年度は成績の二極分化が顕著になった．当該科目では学期中に２回テストを実施しており，成績の二極分化が顕著に現れたのは１回目のテストである．また，当該科目では２回のテストの平均点で最終成績が決まるので，成績下位グループは単位取得のためには２回目のテストの挽回が必至である．そこで，それまでのように「プリントをよく読んで理解するように」ではなく，「用意した Web テストを，各単元で 80 点以上取るまで繰り返し受験すること．80 点以上とれたら，当該単元を理解できたと思って良い．誤答部分には Web テストでの採点後に表示される解説を読むこと．この解説が分からない場合は，プリントをよく読んでいないと思われるので，もう一度該当箇所を読むこと．」と，かなり具体的に学習方法を指示した．その結果，１回目のテストでの成績下位グループが徐々に Web テストに取り組むようになり，演習中にある学生がまだ一度も Web テストをやっていない友人に対して，

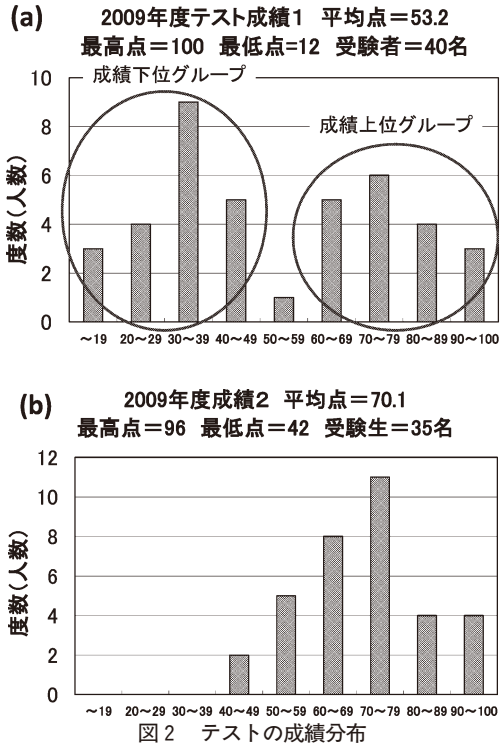

図２　テストの成績分布

「お前やった方がいいぞ.今のままだとやばいぞ.」と勧める光景も見受けられた．その他にも，当初はあまり課題が進まなかった友人がスケジュール通りに課題をこなせるようになったのを見て,Webテストを利用していない学生が焦り出した様子も見受けられた．それらを眺めながら，受講生の過半を占めると想定される中位層が上位層側にシフトして行っている感触を得た．

そうして，学期末の２回目のテストでは，図２に示すように大幅な向上が見られた．このグラフは文献（森田，2011）で公表したものを若干編集したものである．ここに，図２（ａ）１回目のテスト（テスト１）の成績分布で，（ｂ）は２回目のテスト（テスト２）のそれである．

図２（ａ）より，テスト１では平均点辺りを境にして成績が二極分化していることが確認できる．これに対して図２（ｂ）のテスト２では,成績が大幅に向上し,成績下位グループは消滅したことが分かる．三層状態モデルの観点からすると，恐らくテスト１では下位グループに沈んでいた潜在的中位層が上位層側に移動したことで演習時の雰囲気が変わり，さらに少数派となった下位層が中位層側に引き上げられたものと解釈できる．実際，このような成績の向上の主要因は，１回目のテストにおける成績下位グループの躍進であった．テスト成績を分析してみると，１回目の成績下位グループのテスト平均点は，１回目で34.0だったのに対し２回目では62.0まで向上している．一方，成績上位グループの平均は１回目が73.0そして２回目が76.0と言わば高止まりしている．この結果は，学習の仕方が分からないという層に対しては，ここまでやれば良いという目標とその具体的な達成方法を明示することで学習効果が上がることを示唆している．この事例は，プログラミング学習における一例に過ぎないが，同様なことは，他の科目の学習にも当てはまるのではないかと筆者は考えている．

その他，Webテストに続いて，学生のPCからの解答をリアルタイムでWeb上で集計･表示するシステムを卒業研究生の三浦(三浦，2009)と一緒に開発し，それを活用して，学生の理解度をリアルタイムで学生側へのフィードバックすることを試みた．それらの効果や,Webテスト活用における課題などについては，文献（森田，2011）に譲る．

８．おわりに

筆者は，1991年度の社会情報学部開設以，主にプログラミング科目の教育に携わってきた．その間，プログラミングに関する技術や開発環境は大きく変わって来ており，それらの動向を把握・分析しながら，それをいかに教育に反映させるか，ということに腐心してきた．それと同時に，プログラミングを題材として，いかに学生に適合した教育を実現することができるか，という問題意識を持って

教育に携わって来た．もしかしたら，筆者にとっては，プログラミング教育は教育上の題材の一つに過ぎず，後者の問題意識の方が大きかったかも知れない．ともかく，教育の実践に当たっては，学生の学習動向を把握し，それに適合した教育を工夫するという試行錯誤の連続であった．それらの取り組みをこの機会に振り返ってみたのが本稿である．

そういった試行錯誤を重ねる内に，プログラミング演習は，学生の動向を鋭敏につかむことができ，さらにそれに対するこちらの取り組みに対する学生側の反応もキャッチしやすい，言わば教育的試行を試すことが出来る格好の実験場だと捉えるようになった．その観点から，そこで得た学生の学習動向，そして教育的取り組みに対する彼らの反応等は他の科目でも活用し得る普遍的な要素を提供できるのではないか，という問題意識も併せ持つようになった．ここでまとめた様な取り組みがどれだけ，それに応えているかは甚だ心許ないが，少なくとも学習指導を考える際の一つのヒントになっていることを期待したい．

最後に当事者として，これまでの取り組みを振り返ってみて，学生の反応を把握しながら逐次教育方法を工夫して来たことは，一定程度評価できると捉えている．しかし，内容として，どの程度学生の関心を惹くプログラミング学習になっていたか，という点については，その時々で工夫はしたものの，教育方法の工夫の二の次になっていたように思う．現在経済学部と法学部と共同で開講しているCUP（Career Up Program）コースの情報プログラムでは，学生の興味関心を惹きつけるという観点からもう少し自由な発想で学習内容を展開してみたい．

謝　辞

学部のプログラミング科目立ち上げ時に共に科目を担当し，学生の動向に関して貴重なデータを提供してくれた新國教授を始め，種々の助言を頂いた学部の関係教員の方々に感謝したい．又，プログラミング演習の指導に当たってくれたSA（Student Assistant）達は，演習時の受講生の反応について気のついた点を適宜報告してくれた．彼らの貴重な情報がなければ学生の学習動向に関する分析は決して進まなかったであろう．記して感謝したい．

参考文献

森田　彦・新國三千代（1992）「社会情報学部における情報処理基礎教育」『社会情報』Vol. 1, No. 2：35-50

森田　彦・新國三千代・原田　融（1993）「アルゴリズム理解能力の分析」『社会情報』Vol. 2, No. 2：87-99

森田　彦（1995）「アルゴリズム理解能力の分析 II」『社会情報』Vol. 4, No. 2：95-104

仲谷祥子・金子　栄（1995）「文系学生のコンピュータ利用に関する意識調査」『平成 7 年度情報処理教育研究集会講演論文集』：331-334

森田　彦（1998）「コンピュータ苦手意識の影響の分析 ── プログラミング教育の観点から ──」『社会情報』Vol. 7, No. 2：47-62

原　正樹（2009）「「プログラミング」学習用 e-learning システムの開発・及びその運用」『2009 年度札幌学院大学社会情報学部卒業研究』

三浦雄哉（2009）「リアルタイムアンケート集計システムおよびレポート受理システムの開発とその運用『2009 年度札幌学院大学社会情報学部卒業研究』

森田　彦（2011）「ICT を活用したプログラミング教育の実践」『ICT 活用教育方法研究』第 14 巻，第 1 号：26-30