上越教育大学研究紀要　第14巻　第2号　平成7年3月
Bull. Joetsu Univ. Educ., Vol. 14, No. 2, Mar. 1995

コンピュータ・プログラムの作成に関する教育的評価次元の構成

田　中　　敏*

（平成6年10月31日受理）

要　旨

コンピュータ・プログラムの作成が学習・問題解決スタイルをプランナーからマニピュレータのスタイルへ移行させていると A. Minskey は示唆している。後者のマニピュレータ・スタイルを評価するため，本研究はコンピュータ・プログラムの作成作業を分析することによっていくつかの評価次元を構成しようとした。166人の大学生にプログラミングまたはデバッギングの二者択一的やり方（プログラムの実行結果は変わらない）を提示し，両者の一方に対する好みを5ポイント尺度上に評定するよう求めた。この評定値について因子分析をおこない，次の4つの次元を抽出した。1．新奇事項への着手の回避（avoidance to do anything new），2．プログラムの表記と内容の明確化（keeping clear view of descripion and contents of program），3．その場しのぎ・不拡大方針（makeshift and inextension policy），4．探求・発展の志向（intention to progress)。引き続き分析した結果，プログラム読み書き能力の高い者ほど次元2と次元3に大きなマイナスの負荷量を示した。以上の結果はマニピュレータの学習スタイルという観点から考察された。マニピュレータとは活動の「正しさ」よりも「うまくゆくこと」に関心をもつ者である。

KEY WORDS
computer programming　コンピュータ・プログラムの作成
planner and manipulator　プランナーとマニピュレータ
educational evaluation　教育評価　　factor analysis　因子分析

問　題

コンピュータの利用者は2層に分かれる。ひとつはアプリケーション・ユーザーの層であり，もうひとつはプログラマーの層である。前者は既製のプログラムを使用する。後者はそのプログラムを作成する。

情報教育と情報産業の動向は，アプリケーション・ユーザーに特別のリテラシーを求めることを極力避けようとしている。しかし，あいかわらずプログラマーには，人間の知性ではなくハードウェアのアーキテクチャーに対して特殊化された高コストのリテラシーを要求する。

ここに，それまでの言語的リテラシーを用いる学習者とはまったく異なる，新しいタイプの

* 教育方法講座

学習者が見いだされることになった。その典型はミンスキー（Minsky, A., in ブランド，1988）が従来のブルーナー（Bruner, J. S.）流の学習者像である「プランナー」（planner）に対比させた「マニピュレータ」（manipulator）である。学習場面において，プランナーは仮説を立て，それを検証し，修正し，だんだんと「正しいこと」に接近してゆく。これに対して，マニピュレータは正解を発見しようとするよりも誤りを発見しようとする。そして，その誤りを取り除くことによって「うまくゆくこと」に接近してゆく。さらに対比的にいえば，プランナーは自分の仮説にしたがって計画的に環境にはたらきかけ，現実を正しく認識し，支配しようとするが，マニピュレータはその時々の事情に応じて臨機に予定を変更し，現実との妥協をはかるにもやぶさかではない。要するに，マニピュレータは問題の最終的・決定的な解決を意図していない。それに代えて，その場面に問題が起こらないようにし，起こったら即時対応するというようなプラグマティックな処理方針をとる。

「このマニピュレータの発想はコンピュータ・プログラムの作成における誤り除去の作業に由来している。コンピュータ・プログラムのどこかに誤りがあることがわかると慣用的に『プログラムのなかにバグ（虫）がいる』と表現するが，このバグに『除去』の意味をもつ接頭辞“de-”を付加してプログラム中の誤りを除去することを『デバッギング』（debugging：虫取り）と呼んでいる。デバッギングはプログラムの作成に宿命的に付いて回る面倒な作業であり，プログラミングとはデバッギングの別の名前である。…（中略）…プログラムの作成に誤りがつきものであり，デバッギングが不可避であれば，最初から正しいプログラムを作ろうなどと考えないほうがよい。正しいかどうかということよりも，その時々の状況に応じて少しずつ良くしてゆこうと考えるほうが適応的である。このデバッギングの作業方針がマニピュレータとしての学習者像の基礎にある。マニピュレータの原意は『手作業する人』であるが，デバッギングは不良箇所を見つけてはひとつひとつ繕ってゆくという，まさに手作業である。』（田中，1994，p.94）

こうしたマニピュレータの実務的な作業方針は，プランナーの計画的な作業方針に比べて首尾一貫性に欠けるが，その分，不測の事態によく対処することができる。そして，現実に計画外の不測の事態というものは必ず生じるものである。それなら，計画とは必ず失敗するものなのだと最初から決めてかかるほうがよい。究極の計画，万能の知識，最終の解決は存在しないか，存在しても，今すぐわれわれの手の届くところにあるものではない，と。

マニピュレータはひとつの新しい学習者像を示すにとどまらず，もっと一般的な活動スタイルの変化を表している。この変化は時代的・社会的に見れば，「システマティク処理からプラグマティク処理へ」と定式化することができる（田中，1994，pp.116-120参照）。この後者への大局的変遷の結果をとらえるのに，従来の達成型・問題解決型の学習者像に関する観点では不十分であることが容易に予想されるであろう。そこで，本研究は，マニピュレータとしての学習者における活動特性をとらえるための基礎的アプローチを試行しようとする。

このため，マニピュレータの考え方が端を発するコンピュータ・プログラムの作成作業を取り上げ，その潜在次元を抽出してみる。対象は複数のコンピュータ学習の授業を修了したか，受講中の大学１年生であり，実習内容がプログラミングに統一されていないためコンピュータの利用経験はあってもプログラムの作成に関してはまったくの初心者から一応の理解者まで連続的分布が期待できるサンプルとした。したがってまた制約があるとすれば，本研究が明らかにしようとするプログラムの作成の諸次元は初学者に限って存在するものであり，その意味においてコンピュータ教育上の初期の評価次元として適用されるものである。

調　　査

目　的

コンピュータ・プログラムの作成作業を支配する次元を抽出する。大学1年生の初学者を対象に，質問紙において作業場面を例示して当該作業への同調傾向を調べ，因子分析法を用いて妥当な次元を探索する。

方　法

対象者　コンピュータ実習の授業（半期）を修了したか受講中の大学1年生175人。実習内容は各授業によってプログラミングをはじめとして，ワープロ，グラフィック，表計算ソフトの利用，キーボードのタイプ練習など多岐にわたるので，履歴ではなくプログラムの理解度によって既存適性を把握することにした（以下の『質問紙』の項参照）。

質問紙　BASICのプログラムの作成を課題として仮定した（表1参照）。そして，このプログラムの作成時に生じる作業場面を15個想定した（表2参照）。これらの作業場面は，今回の対象者とは異なる初学者が実際にプログラムの作成を実習したときの「キータッチ・キャリア」[1]の記録を再生検討して構成されたものである。教示文は次のとおりであった。「あなた自身が上のようなプログラム（表1の解答例）を作るとします。そのときに，あなたならば『こうするだろうな』ということを想像して，以下の質問に回答してください。なお，質問はひとつひとつ調査者が読み上げて説明します。その説明が終わってから回答するようにしてください。また，『正しい回答』『間違った回答』はありませんので，思ったとおりを答えるようにしてください。」

表1　プログラム作成の問題と解答例

［問　題］
X＋Y　の足し算をするプログラムを作り，
3＋5　の答えを画面に表示せよ。

［解答例］

```
10  INPUT  X
20  INPUT  Y
30  PRINT  X+Y
```

この教示の後，表2のそれぞれの作業場面について「あなただったら，そうしますか」とたずねた。対象者の回答は，「ぜったいそうしないと思う」から「必ずそうすると思う」まで5段階尺度において求めた。なお，以上の作業場面のほかに，表1解答例のプログラムが理解できるかどうかを「ぜんぜんわからない」「少しだけわかる」「だいたいわかる」「完全にわかる」の4段階においてたずね，また，コンピュータを使う作業が好きかどうかを「ぜんぜん好きでない」から「たいへん好きである」まで5段階においてたずねた。

手続き　調査は集団調査であり，300人規模の大教室において実施された。対象者に質問紙を配布し，そこに書かれた課題を実験者が教室前面の2台のTV画面において実演して見せた。実演は表1解答例のプログラムを作成し，実行して，数値を入力し，演算結果を得るというものであった。その後，質問紙の教示文を読み上げ，実験者がひとつずつ作業場面を説明しながら，そのたびごとに対象者の回答をうながし，質問紙の評定尺度上に書き込ませた。

結　果

回答不備の9人のデータを削除したので，最終的に分析対象は166人になった。

1．作業場面の項目に対する評定得点についての因子分析

表2 質問紙の項目（プログラムの作成作業場面）

［項目1］
プログラムを書くとき，1行ずつ書いては，すぐに実行してみた。

［項目2］

```
10 INPUT X
20 INPUT Y
30 PRINT X+Y
```

の10行と20行を一緒にし

```
10 INPUT X: INPUT Y
20 PRINT X+Y
```

と書いてもよいと言われたので，そう書くことにした。

［項目3］

```
10 INPUT X
20 INPUT Y
30 PRINT X+Y
```

いったんX+Yの答えを単独で求めておきたい。その行をKOTAE=X+Yとして途中に加えることにした。

```
10 INPUT X
20 INPUT Y
30 KOTAE=X+Y
40 PRINT KOTAE
```

［項目4］

```
10 INPUT X
20 INPUT Y
30 PRINT X+Y
```

の途中にKOTAE=X+Yを加えるとき，新たに25行として加えることもできた(右)。

```
10 INPUT X
20 INPUT Y
25 KOTAE=X+Y
30 PRINT KOTAE
```

しかし，すべての行番号を10飛びに付け替えることにした（下）。

```
10 INPUT X
20 INPUT Y
30 KOTAE=X+Y
40 PRINT KOTAE
```

［項目5］
KOTAE=X+Yと書くところは，単にA=X+Yでもいいはず（左）。
けれども，やはり
KOTAE=X+Yと書いた（右の枠）。

```
10 INPUT X
20 INPUT Y
30 A=X+Y
40 PRINT A
```

```
10 INPUT X
20 INPUT Y
30 KOTAE=X+Y
40 PRINT KOTAE
```

［項目6］
一応，プログラムの全体を書いた。すぐに実行してみた。途中でエラーが出ても，そのとき直せばいいや，と考えて，実行前に1行ずつ見直してみたりすることはしなかった。

［項目7］
PRINTと書くところをPRNITと打ち間違えていた。
NIだけを直せばいいのだが，頭のPから全部打ち直した。

［項目8］
INPUTと書くところをINPTとしか打ってなかった。Tの直前にIを挿入すればいいし，挿入のキーがあることもわかっていたが，結局，最初のIからその行全部を打ち直した。

［項目9］
先生から，「実行する前には必ずセーブせよ」（プログラムをディスクに保存せよ）と言われていた。思い出して，実行前にセーブした。

［項目10］
プログラムを実行した。3と5を入力したら8と出た。
これで完成と思ったが，ほかにも，−3と−5とか，0.3と0.5とか，いろいろと試してみた。

（表2の続き）

[項目11]
プログラムを実行したら，

```
? ■
```

と聞いてきた。
わざとエラーを出させてみたくなり，数字ではなく，なにか文字を入力することにした。そこで，でたらめにABCと入力してみた。

[項目12]

```
10 INPUT X
20 INPUT Y
30 KOTAE=X+Y

40 PRINT KOTAE
```

の30行と40行を一緒にして

```
10 INPUT X
20 INPUT Y
30 PRINT KOTAE=X+Y
```

とすれば，KOTAEも計算できて，プリントもできるのではないかと思い，やってみた。

[項目13]

```
10 INPUT X
20 INPUT   Y
30 KOTAE=X+Y
40 PRINT KOTAE
```

の10行のXと，20行のYの位置がずれていた。
実行上はまったく支障がなかったが，そろえることにした。

[項目14]
30行めは，純粋な計算ステップであるから，ほかの行と区別して，段を付けてみた。右のように2字分，下げた。

```
10 INPUT X
20 INPUT Y
30   KOTAE=X+Y
40 PRINT KOTAE
```

[項目15] [1)]
このプログラムは3と5をひとつずつ入力していたが，3＋5といっぺんに入力できるプログラムが出来ればいいのにな，と思った。
そう先生に言ったら，「足し算だけなら，これで充分じゃないか」と言われた。しかし自分自身は，充分だという気はしなかった。

1) 項目15は「あなただったら，充分だという気がしますか。」というたずね方をした。

作業場面の項目に対する評定は，それぞれの例示された作業について（自分ならば）「ぜったいそうしないと思う」を1点，「たぶんそうしないと思う」を2点，「どちらともいえない」を3点，「たぶんそうすると思う」を4点，「必ずそうすると思う」を5点として得点化した。この評定得点の平均と標準偏差を表3に示す。

ここで，項目9「プログラムを実行前にセーブする」は平均＋標準偏差の値（5.03）が上限値の5点を越えていたので天井効果が生じたとみなし，以下の因子分析の対象からは除外することにした。

表3　各項目の評定得点の平均と標準偏差（N＝166）

項目	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
M	2.23	2.96	2.70	3.36	2.49	3.23	2.49	2.39	4.22	3.44	2.55	2.57	3.65	2.17	2.91
SD	0.86	1.01	0.96	1.07	0.99	1.05	1.14	1.13	0.81	1.01	1.01	0.92	1.03	0.75	0.94

主成分分析の結果，4因子を適当とみなして抽出した。バリマクス回転後，表4の因子パターンを得た。

表4 バリマクス回転後の因子パターン[1)]

項目番号	因子I	因子II	因子III	因子IV	共通性
7[2)]	0.832[3)]	0.130	0.114	0.063	0.726
8	0.819	0.078	0.022	0.002	0.677
14	-0.049	0.665	0.104	-0.040	0.457
5	-0.001	0.585	-0.132	-0.190	0.396
3	0.230	0.511	0.050	0.174	0.347
12	-0.397	0.446	0.001	0.345	0.476
4	0.145	0.327	0.131	-0.163	0.172
6	0.108	-0.046	0.733	0.051	0.554
1	0.007	0.156	0.679	-0.054	0.488
2	-0.387	0.153	0.423	0.306	0.446
10	-0.157	0.395	-0.426	0.394	0.517
11	-0.004	0.100	-0.366	0.684	0.612
13	0.027	0.283	-0.241	-0.422	0.318
15	-0.042	0.112	-0.094	-0.578	0.357
固有値	1.785	1.683	1.621	1.452	

1) 全分散に対する4因子の累積説明率は46.7%。
2) 項目内容については表2参照。
3) 下線を付した負荷量は.350以上。

±.35以上の因子負荷量を示した項目の内容を参考に各因子の解釈を試みた。

因子Iは項目7・項目8（誤って綴られた語を初頭からすべて打ち直す）より，挿入キーや削除キーなどによる他の機能を使わずに少数の単一のキー操作だけで済ませようとする傾向ではないかと考えられる。それを裏づけるように因子Iは項目2（先生にいわれた工夫をやってみる）と項目12（文法に違反するかもしれない書き方を試みる）にはマイナスの負荷をあたえている。つまり「新奇な，不慣れなことには手を出さない」「いま覚えようとしている範囲内で手一杯である」というような，習いたての初心者に特有の余裕のない状態であると思われる。したがって因子Iは『新奇事項への着手の回避』(avoidance to do anything new) と命名した。

次に，因子IIは項目14（計算ステップの行を段を付けて書く）・項目5（変数名をAとするよりもKOTAEとする）・項目3（計算と表示を分けて独立の行とする）・項目12（文法に違反するかもしれない書き方を試みる）・項目10（課題の3と5以外にも－3と－5などを入力しプログラムの作動を点検する）より，『(プログラムの) 表記と内容の明確化』(keeping clear view of descripion and contents of program) と命名した。

続いて，因子IIIは項目6（プログラムを書き上げたらよく見直さずに即実行する）・項目1（1行ずつ書いては実行してみる）・項目2（先生にいわれた工夫をやってみる）より，場当たり的で，その場しのぎの操作をおこなっているように推測される。そのような場合は，課せられたもの以上に作業を拡大したり応用したりしないものである。それゆえに，おそらく因子IIIはマイナスの負荷を，項目10（課題の3と5以外にも－3と－5などを入力してみたくなる）と項目11（プログラムの実行中にわざとエラーを出させてみたくなる）にあたえるのであろう。したがって因子IIIは『その場しのぎ・不拡大方針』(makeshift and inextension policy) と命名した。

最後の因子IVは，上述の項目10と項目11にプラスの負荷をあたえ，項目13（同じ内容の行について形式的なケタ揃えをする）と項目15（プログラムの機能を課題の要求以上に高める必要はない）にマイナスの負荷をあたえることから，現状にとどまらずに探求・発展を進めようとする傾向が読み取れる。そこで因子IVは『発展・探究の志向』(intention to progress) と命名

した。なお，この因子IVは項目12（文法に違反するかもしれない書き方を試みる）にも，ある程度のプラスの負荷（.345）をあたえることからも，その内容が例証されるであろう。また，特に項目10と項目11に対しては，上掲の因子III『その場しのぎ・不拡大方針』と逆方向の負荷をあたえているので，因子IIIと部分的に相反するものとして因子IVを『発展・探究の志向』と解釈することは適当であろう。

総じて，多少の重構造を示したが因子の解釈は明快であり，妥当性は高いと思われる。

項目の側からいえば，項目4が他の項目と比べて著しく共通性が低く(.172)，独自に変動する項目であると考えられる。そこで項目4は単独で取り上げて分析することにした（後出3参照)。

2．プログラムの理解レベルとコンピュータ作業の好き嫌いによる因子得点の比較

本研究の目的からして，以上の4因子がコンピュータ・プログラムの作成に及ぼす影響は対象者の適性に応じて変化するものであることが期待される。このことを確かめるため，各因子得点について，適性の異なる対象者を比較してみることにする。

質問紙において対象者に，表1解答例のプログラムをどれくらい理解できるかを4段階でたずねたが，この回答をそのままコンピュータ・プログラムに関する理解レベルとし，低いほうから『レベル1群』『レベル2群』『レベル3群』『レベル4群』に対象者を分けた。同様に，コンピュータ作業に対する好き・嫌いの程度を5段階でたずねたが，その回答において「ぜんぜん好きでない」と「あまり好きでない」を『嫌い群』，「どちらともいえない」を『中立群』，「好きである」と「たいへん好きである」を『好き群』とした。なお，全体の平均は理解レベルが2.49（0.98)，好き嫌い程度が3.57（1.65）であった（カッコ内は標準偏差)。なお，両者のクロス集計の人数は表5のNに示されるが，特に有意な偏りは見いだされなかった($\chi^2_{(6)}=5.35$, n. s.)。このことから，コンピュータ・プログラムの理解とコンピュータ作業の好き嫌いは直接には関連しないといえる。

表5は理解レベル群と好き嫌い群における標準因子得点の平均と標準偏差を示したものである。それぞれの因子について，4（理解レベル）×3（好き・中立・嫌い）の分散分析をおこなった。結果として，いずれの因子についても交互作用はまったく検出されなかった。なお，以下の分散分析において主効果が有意であった場合，平均間の多重比較にはテューキ法を用いた。

因子I『新奇事項への着手の回避』については，好き・嫌いの主効果が有意であった($F_{(2,154)}=4.78$, $p<.01$)。多重比較によれば，嫌い群の平均（0.42）が好き群の平均（−0.24）よりも有意に大きかった（$MSe=0.98$, $p<.05$)。中立群の平均（0.12）は両者の中間にあり，全体として直線的に変化していた。このことから，コンピュータ作業が嫌いであればあるほど，新奇の操作がたとえ合理的・節約的であると知っていても，その利用と学習を回避する傾向が強くなるといえる。

因子II『表記と内容の明確化』については，理解レベルの主効果が有意であった（$F_{(3,154)}=2.96$, $p<.05$)。各群の平均は大きい順に0.31（レベル1群），0.16（レベル3群），−0.08（レベル2群），−0.39（レベル4群）であり，多重比較によればレベル1群とレベル4群の間に有意差があった（$MSe=0.97$, $p<.05$)。したがって初級者は因子IIに強く支配される傾向があり，各行の内容をわかりやすく書き表し，よりよく把握しようとするが，これに対して，上級者は今回のプログラムのように全体の流れが明白な場合，あるいは実行上の誤作動が生じない

表5 プログラムの理解レベルとコンピュータ作業の好き嫌いによる標準因子得点の平均と標準偏差

レベル		1			2			3			4		
嫌・好		嫌	中	好	嫌	中	好	嫌	中	好	嫌	中	好
	N	7	8	13	17	17	25	9	11	29	6	5	19
因子I	M	0.39	0.20	-0.17	0.49	0.21	-0.20	0.41	-0.05	-0.32	0.23	0.04	-0.25
	SD	0.96	0.91	0.84	1.04	1.11	0.97	1.27	1.07	0.81	1.21	1.21	0.96
因子II	M	0.11	0.39	0.37	-0.10	-0.34	0.11	-0.21	0.30	0.21	-0.94	-0.37	-0.22
	SD	0.64	1.29	1.14	0.73	0.84	1.18	0.61	0.77	0.81	0.98	0.73	1.35
因子III	M	0.45	0.59	0.29	0.50	0.26	0.04	-0.23	0.14	-0.23	-0.18	-0.70	-0.73
	SD	1.20	0.96	0.85	0.78	0.99	1.09	0.89	0.78	1.14	0.60	0.87	0.64
因子IV	M	0.45	0.29	0.37	-0.52	-0.18	0.09	-0.12	-0.36	-0.04	-0.33	0.05	0.40
	SD	0.69	1.04	0.74	1.02	1.01	0.88	0.49	0.81	1.11	0.74	0.97	1.34

場合には，プログラムの表記と内容に特別の注意をはらう必要性を感じないのではないかと考えられる。

因子III『その場しのぎ・不拡大方針』についても理解レベルの主効果が有意であった（$F_{(3,154)}=5.52$, $p<.01$）。各群の平均は大きい順に，0.41（レベル1群），0.24（レベル2群），−0.15（レベル3群），−0.62（レベル4群）と直線的に並んだ。多重比較の結果，レベル1群とレベル4群との間に有意差があった（MSe=0.90, $p<.05$）。このことから，理解レベルの低い者の作業スタイルはいっそう場当たり的であり，課せられた問題だけに限定して作業を済ませてしまおうとする傾向が強いといえる。これと対照的に，理解レベルの高い者は，プログラムの実行に際していろいろなケースを試みたり，わざとエラーを求めてみたり，プログラムの適用と作動を試行・点検することに動機づけられるように思われる。

最後に，因子IV『探求・発展の志向』については，理解レベルの主効果が有意傾向であった（$F_{(3,154)}=2.29$, $p=0.081$）。しかしながら，多重比較の結果はいずれの群間にも有意差は見いだされず，実質的な差は生じていないと判断すべきであろう。ちなみに，平均の大きいほうから並べると，レベル1群（0.36），レベル4群（0.20），レベル3群（−0.13），レベル2群（−0.17）であり，群の順位にも取り立てて一貫した傾向を読み取ることはできない。

3. 項目4についての分析

項目4の内容は「新しい行を途中に設けるとき改めて行番号を10飛びに付け替える」という作業である。評定得点を従属変数とした4（理解レベル）×3（好き・中立・嫌い）の分散分析の結果，理解レベルの主効果のみが有意であった（$F_{(3,154)}=4.31$, $p<.01$）。テューキの多重比較によれば，レベルI群（3.51）・レベル2群（3.53）・レベル3群（3.45）と，レベル4群（2.77）との間に有意差があった（カッコ内は群平均，MSe=1.10, $p<.05$）。

基本的に，BASICのプログラミングでは最初の行番号はプログラムの作成が終了するまで付け替えないのが常識であり，まさにそのために最初から10ずつ飛び飛びに付け，新しい行は

1～9までの中間番号を付けるようにしている。また，たとえ，このことを知っても知らなくても，今回のプログラムに新たな1行を挿入する際の全行番号の付け替えはまったく不要であり，たんに冗長で面倒なだけである。したがって，理解レベルの低い者はプログラムの外見や表記の形式にとらわれる作業傾向が強いということが示唆される。このことの傍証として，項目4は低いながらも因子II『表記と内容の明確化』に対して相応のプラスの負荷量（.327）を示していることが注目される。

考　　察

以上の調査結果に基づいて，プログラム作成者の具体的な像を描き出すように考察を進めてみたい。そうすることは，プログラム作成者の諸作業に対する教育的な評価次元を構成することにもなるであろう。

まず，因子I『新奇事項への着手の回避』について，コンピュータ作業の嫌いな者がその傾向を強くもつことがわかった。一般的にいえば，この因子は学習領域や学習材料に対するネガティヴな態度であり，学習意欲・練習意欲・試行意欲の見られない活動を引き起こすことになる。その際，特に注意すべきは，因子Iについて好き・嫌い群の主効果のみが見られ，理解レベルは有意な変動を示さなかったということである。したがってコンピュータ作業が嫌いとなると，コンピュータ・プログラムに関する理解レベルが高かろうと低かろうと関係なく新奇事項に対する敬遠が起こりうる。教育的には，理解レベルの高い群が問題であり，プログラムの理解力や作成力が優れていても新奇事項の取り込みにおっくうな者が存在する。プログラムの作成作業に対する評価次元としては因子の名称を内容的に逆転させて『新奇事項への着手』とし，その程度を評価すればよい。

次に，因子II『(プログラムの) 表記と内容の明確化』と因子III『その場しのぎ・不拡大方針』については，両者とも理解レベルの主効果が有意であり，共にレベル1群とレベル4群との間に差異が見られたので一緒に検討すべきであろう。すなわち，理解レベルが上昇するにつれて，『表記と内容の明確化』と『その場しのぎ・不拡大方針』が並行して低下するのである。このことは統合的にみれば，プログラムの表記と内容と課題達成を重視する作業者像から，プログラムの機能と拡大適用を重視する作業者像への移行を示唆していて興味深い。この移行は，冒頭の『問題』において述べた，プランナーからマニピュレータへの学習者像の変遷と同質のものではないだろうか。

ある意味で，プログラムの表記と内容を明確に書き表すことは処理の「正しさ」を認識しようとすることである。それはプランナーが正しい仮説を求めようとすることと同じである。しかし，そのような作業にこだわらなくなること（因子IIの負荷の低下）はプランナーからマニピュレータへの移行を示しているといえる。マニピュレータは，プログラムの中身の「正しさ」を認識することよりも，プログラムの機能（うまくゆくかどうか）を点検するほうにずっと関心がある。このためまたマニピュレータは，因子IIIによる項目11（プログラムの実行中にわざとエラーを出させてみたくなる）の負荷に典型的に表されているように故意に失敗を求めてみたり，当面の課題達成にとどまらずプログラムの適用を広げてみたりする。

かくして，理解レベルの高い者に見られる因子負荷特性はマニピュレータの像を描き出すよ

うに思われる。

この点，最後の因子IVについて理解レベルの効果が見いだせなかったことも示唆的である。通常の学習の進歩は因子IV『探求・発展の志向』と正の相関を示し，問題が解決されても，さらに解決の改善や応用へと向かうであろうが，コンピュータ・プログラムの作成は当面の問題が解消されれば特にそれ以上の問題を（欲してもよいが）欲しなくてもよい。プログラムに多少の改善の余地があっても，重大な問題を起こさなければ十分であるといえる。なぜなら，マニピュレータの活動方針と同様にコンピュータ・プログラムの作成も，（起こりうるすべてのバグの発生を予知することが不可能であるゆえに）問題がとことん解決された状態を最良とすることがむずかしく，たとえ問題が残っていても現実に問題が起こっていない状態をもって「よし」とせざるをえないような性質の作業にほかならないからである。

以上，因子IIと因子IIIは，マニピュレータという観点に立った特有の評価次元を構成することができるが，最後の因子IVはコンピュータ・プログラムの作成に限定されない資質を評価するものと考えるべきであろう。

そこで，因子IIと因子IIIの内容を，コンピュータ・プログラムの作成力・理解力とポジティヴな関係をもつように書き換えて教育的な評価次元を構成すれば，『プログラムの表記と内容に対する機能の優先』と『適用の試行と点検』となる。因子IVはこのまま『探究・発展の志向』でよいと思われるが，おそらく，その評価次元はコンピュータ・プログラムの作成作業を弁別的に価値づけることができないであろう。

なお，今後の課題として次の点を考慮した追試が必要である。①質問紙において指示する作業場面の種類と個数を増やすこと。②対象者にもっと高度の能力をもつプログラマーを含めること。③対象者の通常領域の問題解決力と活動方針を調べて，それがコンピュータ・プログラムの作成力と作業方針に関連するかどうかを分析すること。以上である。

注

1) 「キータッチ・キャリア」の資料は中野靖夫氏（上越教育大学学校教育研究センター教授）より提供していただいた。

文 献

ブランド，S. 室謙二・麻生九美（訳） 1988 メディア・ラボ 福武書店
田中 敏 1994 心のプログラム 啓文社

付 記

本研究は文部省科学研究費補助金一般研究C（No.05680172）「プログラム作成過程に視点を置いた評価法」の研究成果の一部である。

Construction of Dimensions for Educational Evaluation in Computer Programming Practice

Satoshi TANAKA*

ABSTRACT

Computer programming practices has been opening a possibility that, suggested by A. Minsky, the style of learning or problem solving would shift from planner's to manipulator's. In order to evaluate the latter style, this study tried to construct some evaluational dimensions by analyzing various ways of computer programming practice. One hundred and sixty-six undergraduate students were presented two alternative ways of programming or debugging that did not change program outputs, and asked to rate their preference to either of the two ways on five-points scale. On these ratings factor analysis was conduced and four dimensions were identified as (1) avoidance to do anything new, (2) keeping clear view of descripion and contents of program, (3) makeshift and inextension policy and (4) intention to progress. Post-hoc analysis revealed that those students who had higher programming literacy had more negative loadings of Dimension 2 and Dimension 3. The results were discussed in terms of the learning style of manipulator that would be concerned with "success" of activity rather than "correctness".

* Divison of Method and Evaluation