FŌRTRAN 言語の難易性に関する統計分析*

松田栄子

1. software の発展とその問題点
2. FŌRTRAN program の error の傾向：従来の報告の要約
3. data の性質と統計的分析方法の概要
4. 統計的分析の結果
5. 残された問題と 2・3 の示唆

1. software の発展とその問題点

電子計算機は，所詮，Mälzel が創ったという象棋指しの自動人形以上のものではないと思われる。かつて E. A. Poe が，この人形の中に人間が隠れていたことを推理によって見破っているが，これはある意味で今日の電子計算機を考えるうえに，きわめて示唆的な episode である。もちろん，電子計算機といえば，いわゆる hardware そのものをいみするが，これをある job を処理しうるひとつの system とみる場合，周辺機械はいうに及ばず，さらに重要な factor として，software 類も含めた総合的全体として把握されなければならない。上の Mälzel の人形に即していえば，人形自体が hardware で，その中に入って人形を動かしている人間が software ということになろう。実際，software が隠れて電子計算機を動かすもの，あるいはそれ以上のものであることを，多少でも電子計算機と接触をもったことの

* 本稿執筆にあたり，多大の御指導と御助言を賜わった小樽商科大学古瀬大六教授，戸島熙助教授，清水川緋紗子助教授に深く感謝の意を表する。なお，本稿の統計計算は，ALGŌL-H で筆者自身が組んだ program によって，小樽商科大学計算センターの ŌKITAC-5090H を使用して行われた。

ある者ならば，深く認識せざるをえないのである。この意味で software を hardware に比べて軽視することは，software を電子計算機の system の中に，正しい perspective で位置づけることを拒否することになり，正しい態度とはいわれない。

さて電子計算機の hardware の技術の歴史的発展を回顧することは，これまでにもしばしば行なわれ，その発展過程はかなり細部まで明らかにされていると思われるが，これに反して，software 技術の発展に関しては必ずしもその全貌が明らかになっているわけではなく，詳細な記録としてそれらがまとめられるためには，まだ多少の時間が必要であろう。しかしある程度の総括的回顧を行なうことは，現在の時点に於いても不可能ではないし，いく人かの著者もすでにそれをこころみている(1)。以下われわれも，software の発展について簡単に要約し，その歴史が現在のわれわれにどの様な問題をなげかけているかを吟味することにしたい。

電子計算機が本格的に普及し始めたのは50年代の後半からである。そのひとつの指標として，transistor 化された最初の計算機は1959年までには市場に出現していたといわれている。計算機に所望の仕事を行なわせるには，計算機に対して問題を coding して与えてやらなければならない（いわゆる programming)。初期の計算機に対して，この coding はすべて machine language で行なわれていた。例えば，計算機が binary machine ならば，coding の結果は1と0の羅列となり，それに対してあとからいろいろ check を行なうことはきわめて困難であった。そのため，program を何らかの symbolic language でかき，計算機を使ってそれを machine language にかきかえるという工夫が行なわれた。これが assembler の登場に他ならない。初期の assembler として著名なものに IBM 704 に対して SHARE group が開発した UASAP がある。最初の UASAP は非常に簡単なもので，その機能は，まさに one-to-one translation のかきかえ以外のなにものでもな

(1) 例えば Rosen [1] の論文 IA, IB などをみよ。

かった。やがて，assembler はこうした単純なかきかえ機能の他にもっと programmer の負担を軽減させる機能をもちうることがわかってきた。すなわち，番地割付の機能がそれである。まず programmer は番地を絶対番地でかく必要がなくなり（symbolic address の導入），さらに，各 instruction の location をひとつひとつ指定することも必要でなくなった。

しかし，この様ないわば bookkeeping 的な機能が assembler に具っても，assembler language と machine lenguage は依然として 1－1 に対応し，その意味で両者は 'isomorphic' であり，そのため assembler language による programming にもかなりの熟練が要求された。

この様な assembler の次にくるのが，当然，one-to-many translation の考え方である。これによって，programmer は計算機に対する命令を非常に簡潔な形でかくことが出来る様になった。計算機はこの簡潔な命令を machine language による幾命令かに translate しそれを実行する。通常この過程には 2 種類のものが区別されて考えられている。1 つは interpretive mode といわれるもので，1 つの命令の translation と execution がつづいて行なわれる。他の 1 つは translative mode といわれるもので，すべての命令の translation のみが行なわれ，execution は translation の過程とは独立に行なわれる。現在では後者の mode が一般に普及しているが，前者も全くすたれてしまったわけでなく，例えば入出力命令は interpretive mode で処理されることが多い。

ところで，ここで注意しなければならないのは，計算機に対する命令を簡潔にかき表すことができるということが，次に，それが特定の計算機のみに限られる必要がないという認識を生んだことである。適当な translation program さえあれば，同じかき方をした簡潔な命令を，異った計算機が input data としてうけつけ，それを machine language に translate して実行することが出来るのである。これは結局，計算機と language の分離ということである。こうした認識の後には programming language は計算機

とは一応独立に扱われることになり，programming language の design などが新たな問題領域を形づくるにいたった。すなわち，computer-oriented language にかわる problem-oriented language の誕生である。

software の歴史の上で，この種の language として一期を画するほどに成功したのは FŌRTRAN language が最初であった。これは IBM 704 に対して適当な translation program（すなわち compiler）を開発するという J. Backus などによる IBM の2年半にもわたる project から生れたものである。FŌRTRAN compiler は遅くとも 1957 年までには basic なものが完成していたと推定される。もちろん，50年代後半の compiler としては，この FŌRTRAN compiler が唯一のものではなかった。例えば Burroughs の Datatron 205 に対して Purdue 大学でつくられた compiler がある。その後 IBM 650 に対しても Carnegie 工科大学でも同じ様な compiler がつくられ IT と名付けられていた。また，こうしたものの中で，後の compiler の発展に多大の影響を及ぼした，IBM 701 に対する PACT system の名前も逸することが出来ない。

IBM 704 と共に FŌRTRAN language の普及が本格的に始ったのは，1958年における FŌRTRAN II の出現以降である。今日，FŌRTRAN は version V まで出ており，programming language としてはもっとも普及したもののひとつとなっている。

FŌRTRAN language がいわば暗黙に公認される様になったのに対して，ALGŌL language が，1958年に ACM と GAMM の共同委員会で検討されて，計算機に対する国際共通言語として承認されてできたものであることは周知の通りである。1958年の ALGŌL は後に 1960年に revised report が出て，一般には ALGŌL 60 として知られている。ところで，ここで，広く普及し始めていた FŌRTRAN をさておいて，ALGŌL が recommend されたのは何故か，という疑問がおきるかもしれないが，それに対しては次の

様な解答が用意されている。[(2)] まず第1は，FŌRTRAN language 自体がもつ種々の制約である。これは上に述べたように，FŌRTRAN が，元来 IBM 704 という特定の機種に対して implement されたものであるという事情が強く働いている。従って FŌRTRAN language そのものには IBM 704 の性質がかなり反映していて，必ずしも計算機から完全に独立した language とはいいえないのである。第2の理由は IBM の独占的な販売成績に関係している。57年から58年の computer field において IBM に対立する有力な競争者は殆んどなかったといわれている。そこで ACM の member の多くは，もし FŌRTRAN language を計算機の国際共通言語として認めるならば，IBM はその独占的体制をますます強めるであろうと考えたのである。こうした理由によって，ACM は IBM の強力な競争者を育てるために GAMM と共同して，国際共通言語としての ALGŌL の制定にふみきったのであった。

さて，以上の説明から problem-oriented language の出現が次の様な利点をもたらすことはあきらかである。まず，それは計算機を使って問題をとこうとする者に特定の計算機に対する習熟を要求しない。しかも彼は同時に programmer でもありうるので，多くの場合，彼の問題に関しては全くの素人でしかない専門の programmer に問題とその解法を理解させる手間を省くことが出来るのである。しかし，便利さの裏には大低不便利さが同居しているのであって，この場合も例外ではない。すなわち problem-oriented language は新たに次の様な2つの問題を提起することになったのである。

その第1は，problem-oriented language を処理する translation program をつくる労力をいかにして軽減するかということである。現在でこそ，FŌRTRAN compiler, ALGŌL compiler などの作成に関しては標準的といっていい程の手法が開発されていて，比較的容易にそれらをつくることが出来るようになっているが，かつては，かなりの日時と労力を要する仕事であ

(2) Rosen [1, p. 10] 前掲書。

った。ちなみに50年代後半に FŌRTRAN compiler を作成した時には，J. Buckus を chief とする25人の staff が約2年半の月日を費したのである。現在でもその事情は新しい language の processor をつくる時には同様である。これをどの様にして解決するかという問題は，現代の computer science に対する1つの大きな挑戦であったが，今日では一応の解決をみようとしている。(3) しかしこの theme について語ることは本稿の範囲を逸脱すると思われるので，これ以上は立ち入らない。

もう1つの問題とは problem-oriented language の習得に関するものである。problem-oriented language はたしかに programming 人口を急増させた。それは problem-oriented language の使い易さに起因するものであることは疑うことが出来ないが，ここではさらに1歩進んでその「使い易さ」とは一体何に比べてのことか，という問を発してみれば，その答は machine language またはそれと isomorphic な language ということになろう。実際，machine language 時代からの長い経験をもつ者であればある程，problem-oriented language の有難さが身にしみて判るという傾向が一般的に見られる様である。しかし計算機との接触が最初から problem-oriented language である者には，当然のことだが，その有難さが実はよく判らないのであって，彼は FŌRTRAN, ALGŌL という様な人工の language を使って，自分のとこうとしている問題の計算の procedure を表現することにかなりの困難を感ずるのである。もちろんその困難は僅かな努力によって急速に減少して行くべき性質のものであるが，それでも困難はやはり困難である。この困難は現在の programming language が problem-oriented であるという，まさにその人工性それ自身に由来するものである。

problem-oriented language が人間にとって一層 natural であるといわれるのは，あくまでも比較の対象が machine language であることを忘れてはならない。いま natural language と比較するならば，problem-oriented

(3) 例えば，syntax-oriented compiler などがその答のひとつである。

language はきわめて制約の多い，その意味では，上に述べたこととは反対に，かなり「使いにくい」language というべきであろう。そこで，これを克服するにはどうしたらよいであろうか。そのひとつの方法は，いわば 'human-oriented programming language' への道である。すなわち，現在の programming language をもっと **user の立場から**使い易くすることである。これを別な言葉でいえば，programming language の設計に人間工学的な思想をもっともり込むべきであるという様にも表現できる。実はこのような考え方はすでに PL/1 に多少あらわれ始めているので，必ずしも新しい思想というわけではないが，従来それほど強調されなかった様に思われるので，ここでとくにこの点を現在の programming language における「問題点」として注意をよびおこしておくことにしたい。

さて，一口に 'human-oriented programming language' といっても，一挙に到達できるものでないことは容易に予想される。従って我々は，まず手始めに従来の problem-oriented language のどの点が使いにくいのか，又は，どこが理解しにくいかという様な検討を徹底的に行なうべきであろうと思われる。その様な作業のつみ重ねが新しい道の開拓のひとつの重要な point となることは疑いのない所である。

以下，本稿では，problem-oriented language の代表的なもののひとつである FŌRTRAN language を対象として，それを学習者の立場からみた場合どんな問題点が潜在しているかを定量的に分析することをこころみる。**2** では，従来 FŌRTRAN language の困難な点に関してどんなことがいわれていたかについて略述する。**3** では，我々の採用した統計的分析方法と data に簡単にふれる。**4** では，我々の統計的分析の結果を呈示する。**5** では，我々の分析が示唆するものを展望して稿を閉じる。

2. FŌRTRAN program の error の傾向：従来の報告の要約

FŌRTRAN language の難易性に関しては，従来，実際の programming

の段階で犯される error の傾向という形で追求されてきた。この節では，この FŌRTRAN program の error の傾向について，どの様なことがいわれているかを，のちの我々の結果と比較するために簡単に要約しておきたい。program に何らかの文法的 error があった場合，それは compiler によって検出され，error message として計算機操作のための console typewriter に打たれたり，LP (line printer) 用紙に印字されるのである。この error message の頻出度がどのようなものであるかを知る事は，FŌRTRAN language の学習上の躓石がどこにあるかを知る上に，大いに役立つであろう。

さて，この種の研究では，東京大学大型計算センター（以下東大センターとする）からの諸報告がある(4)。それによれば，まず初心者向けの講習会において，理解され難いのは，FŌRMAT statement である。FŌRTRAN language の FŌRMAT statement には，整数 field の I-type，指数部のない実数 field の F-type，指数部つき実数 fieldの E-type，空白の field の X-type，Hollerith field の H-type，alphameric field の A-type，論理 field の L-type，8進数整数 field の Ō-type の8種類がある。これは programmer が入出力 data の仕様を自由に決定出来るものであり，非常に便利なものであるが，初心者にとってはこの FŌRMAT statement の自由さがかえって FŌRMAT statement を理解し難いものにしているのである。また東大センターの program 相談室にもち込まれる error に関する相談のうち，最も件数の多いものは I/Ō (FŌRMAT, DATA) に関するもので，それには field に関する誤りがあり，FŌRMAT にとってある field と list の対応の悪いものがあり，さらに制御用文字の使用上の error と種々である。次に多いのが，subscript, DIMENSIŌN に関するものである。配列の宣言を忘れたり，変数名と重複して使用した為におびただしい error が生ずる事があ

(4) 数理科学 [4, p. 4～p. 9, p. 14～p. 17)]，東京大学大型計算センター広報 [5, p. 45] 等にみられる。

る。また subscript の形が許されているかどうか，subscript が宣言された領域外に出ていないかどうかは綿密に検討されなければならない。3番目に多いのが DŌ loop であり，これには，parameter を loop の中で変更する誤りが一番多いという報告があった。これらの error は，FŌRTRAN language で programming することにかなり習熟した人でも，応々にして犯してしまう error の様である。次に，我々にとって，実際の program における error の表われ方，error の傾向が関心事となる。これについては東大センターにおいて，一般計算依頼1170件の program を無作為に抽出し，compile の際に出された error と job-monitor から出された error message の度数分布の調査が成され，program miss にかなりはっきりした傾向がある事が明らかにされている。以下はその結果を本稿のために適宜まとめたものである。((1), (2)…は東大センター報告の error の頻度の順位を表わす)

I. 変数に関する error (31.7％)

(1) 変数 □ は＝の左辺，READ statement に現われているが，それを使用している部分がない——21.3％

(2) 変数 □ は実行可能な statement に現われなかった——5.4％

(3) 変数 □ の値を使っているところはあるが，その値を定義している所がない——5.0％

II. FŌRTRAN の statement に関する error (11.2％)

(10) この statement は program の中で実行されることがない——2.6％

(12) 1個の statement で左カッコの数が右カッコの数よりも多い——2.4％

(13) 最後の実行可能な statement が GŌ TŌ 型か STŌP statement でない——2.3％

(15) 1個の statement で右カッコの数が左カッコの数よりも多い——2.1％

(17) HITAC 5020 の FŌRTRAN で許されていない statement が現われ

た——1.8％

III. statement number に関する error (10.4％)[5]

(4) statement number |___| が未定義である——4.4％

(5) statement number |___| が使われているが，それが定義されていない——3.4％

(11) statement number を2重に定義している——2.6％

IV. 算術式に関する error (8.1％)

(7) 算術式の中に許されない組合せ（2つの演算数の型に関して）がある——3.2％

(7) 算術式，論理式の中の記号の位置がおかしい——3.2％

(16) 算術 statement の左辺の名前に誤りがある——2.0％

V. FŌRMAT statement に関する error (8.1％)

(6) この statement は FŌRMAT statement として用いられなければならないが，正しい FŌRMAT statement ではない——3.4％

(9) この FŌRMAT statement は入出力 statement で利用されていない——3.1％

(21) FŌRMAT statement の仕様の書き方に誤りがある——1.6％

VI. DŌ loopに関する error (5.6％)

(14) DŌ の指標が loop の中で変えられている——2.2％

(18) DŌ loop の中の statement に函数を定義する statement がある——1.7％

(18) DŌ loop の重なり方に誤りがある——1.7％

VII. subscript に関する error (1.7％)

(18) 添字付き変数に添字がついていない（配列名だけでは使えないとき）——1.7％

---

(5) (4)は level number 2, (5)は level number 4 と level number が異ることが HITAC [6] でしられる。

以上の諸報告から我々は，FŌRTRAN language を用いて programming する際に頻出する error の傾向を知る事が出来た。そしてこれらの結果は coding の段階で program を check する為に，「算術式の中で実数と整数と四則演算をしていませんか？」，「IF の中で実数と整数とを比較していませんか？」，「配列はすべて忘れずに宣言してありますか」等々の check list として活用されている。ちなみに Southworth and Deleeuw[(6)] の FŌRTRAN check list の項目は，1．一般（statement の左右のカッコ数，statement number, Key word の spelling 等々），2．定数と変数，3．算術 statement, 4．入出力 statement, 5．制御 statement となっている。

しかし，我々が本稿で取り扱おうとしている FŌRTRAN language の理解に関する，すなわち，FŌRTRAN language そのものに潜んでいる難しさと，program 上の error の頻度が符合するかどうかは速断を許さないところである。例えば，初期の段階に於いて FŌRMAT statement は非常に理解され難いものであるが，FŌRTRAN language を学習し，かなり自由に使いこなせる様になった段階においては，相対的に error の頻度は低くなり，これとは別に，変数や算術式に関する error が多くなってくるという傾向がみられるのである。

このような事柄を考慮しつつ，以下我々の data の分析とその結果の解釈を行ってみよう。

3．data の性質と統計的分析方法の概要

小樽商科大学では，昭和 38 年に電子計算機が導入せられ，翌年秋から OKITAC-5090 H が稼動し始め，教官の研究，学生の教育，学内事務の機械化などのために使用されている。このうち，学生の教育としては，昭和 40 年度から管理科学（management science）科が正式に発足し，翌 41 年から，主として同科 2 年度生を対象に「計算機 program」という講義が開か

(6) Southworth & Deleeuw [2] の 3-14・2 をみよ。

れている。昭和41年度の同講義の内容は，前半に FŌRTRAN, ALGŌL などの compiler language の解説と演習を行い，後半では assembler language の解説と演習を行うと共に，数式の翻訳など system programming のごく初歩的な事柄を説明した。前半の compiler language の解説では，こころみに，毎時間の一定時間をさいて，前の時間の講義の内容に関する achievement test を行って採点をし，それを受講者に戻し，各自に check させると共に，それから得られた情報を講義の解説の内容に反映させる様にしていた。これはある程度の効果をあげた様であるが，10回の講義（1回は100分）で，FŌRTRAN, ALGŌL をすましてしまうのは多少無理でもあり，また，かなりの脱落者が出たことを考慮して，昭和42年度は進行の pace をすこしおとし，compiler language の講義に15回程度をあて，ことに FŌRTRAN の講義にその2/3以上をさいて，解説と演習を念入りに行い，しかも受講者の理解の程度を解説に feed back させるために，昭和41年度に行った achievement test をふたたび毎時間連続して行った。この年度は成績の一覧表を毎回掲示するのみで，答案そのものは受講者に返却しなかった。

本稿で分析の対象となった data は，それらのうちの FŌRTRAN に関する答案である。test の問題のいくつかを例として附録にかかげておいた。そこで見られる通り，大部分の問題の難易の level は森口［7］のものと大差がなく，また実際に，いくつかのものは森口の問題を多少かきかえただけで使用している。しかし，大部分の問題は講義にあわせて，その都度作成したものである。なお，問題の作成時には，必ずしも本稿でこころみた様な統計的分析を行うことを，問題の作成者は考えていたわけではなかったことを，特にここで述べておいた方がよいと思われる。従って，例えば FŌRMAT 関係の事柄に対して解答を要求する様な問題でも，それを解答者に積極的に要求する場合（すなわち，ある statement をかきこむ）と，それを比較的消極的に要求する場合（すなわち，ある statement が正しいかどうかの判

断をする）の両方が同時に存在している。[(7)] このことは後にふれるように多少の問題を孕んでいるのであるが，ここでは，このような意味で上述の答案から得られるものが一概に管理された実験の data とはみなしえないものであることをあらかじめ指摘しておくことにしよう。

さて，これら個々の問題は FŌRTRAN のどの statement と関係しているかということによって分類することが出来る。とくに FŌRMAT 関係の問題は一層詳しく分類される。その分類は附録第 1 表に記されている通りである。問題によっては，複数の statement に関係するものもあって，必ずしも一意に分類しえない場合もあったが，その時は何が主として質問されているかによって分類した。この様にして，個々の問題はいくつかの group にわかれる。その group のことを以下では「問題群」ということにする。すなわち，我々はいくつかの問題群をうることが出来るのである。各問題群にはいくつかの問題が属しており，その個数は，最低で 1 個，最高で 17 個である。

次に，この分析のためには，受験者の中に test を全部うけていないものもある等の理由で，test data 全部を使用することはできないので，その中から test の成績を参考として，比較的成績のよいものと悪いものの比率が 2：1 になる様にして，とくに 30 名を選び出した。この意味で，受験者 sample の抽出は random には行っていない。これで実験 data を 2 方向に分類して並べることが出来る（附録第 1 表参照）。実際の test では，各問題に問題作成者の主観的な判断による評価（配点）が与えられていたが，ここでは，その主観的評価を一切やめて，各問題に対する解答が正しいか（その時は 1 とする），誤っているか（その時は 0 とする）のいずれかであるとした。ここで正答率を次の様に定義する。

---

(7) しかし，出題形式についてはそれ程極端な差があるわけでなく，恐らく少数個の pattern に分類されてしまうであろう。このことは，本稿でこころみた様な分析をやるにいたった 1 つの理由である。

$$正答率=\frac{1の個数}{1の個数+0の個数}\times 100$$

以上によって，各問題群毎の正答率と各受験者毎の正答率を計算することが出来る（附録第1表及び附表1）。なお，2方向に分類した0と1からなるdataと上の正答率は，本稿の分析にとっては基本的なdataであって，以下の分析はこれらから，さまざまの情報を引き出すために行われる。

1で述べた様に，本稿の我々の目的はFŌRTRAN languageの難易性を検出することにあり，しかも，上述した様な受験者sampleのえらび方によって，ここで受験者間のFŌRTRAN languageに対する理解度の相違という問題は一応考慮しないこととする。従って，dataを2方向に分類してはいるが，以下，実際には，問題別の分類が主要な関心の対象となる。

それでは，問題毎の正答率から我々は一体何を判断することが出来るであろうか。附録第1表によって容易にみられる通り，ある問題では正答率がかなり低く，他の問題では正答率が相当高い。このことは，直ちにその問題（関連するFŌRTRAN statement）についての難易性を反映するであろうか。勿論，その様に判断するのがnonsenseであることはいう迄もない。そこで，各問題群毎に平均正答率を計算して，それを比較してみることにしたならばどうであろうか。これによって各問題群毎の難易性がある程度迄は明らかになるであろう。しかし，各問題群はいくつかの問題，従ってそれらに対応する正答率をもち，しかも，それらはバラついているのであるから，単に平均正答率を比較することだけでは十分とはいえない。結局，この様なsituationに有効で，かつconventionalなひとつの統計的分析方法として，ひとは容易に分散分析を思いつくであろう。そこで，我々は，以上に述べた基本的dataに対して，まず，分散分析をこころみることにした。

分散分析はどの統計学のtextにものっているきわめてfamiliarな統計的分析であるので，ここで改めて述べるまでもないと思われるが，論文のself-containednessを期するために，ごく簡単にその outlineにふれておこ

う。

いま，問題群が k 個あり，各問題群が n_i 個 $(i=1, \ldots, k)$ の正答率をもっているとする。この時，問題毎の正答率を次の様に並べて考えることが出来る。

$$x_{11},\ x_{12},\ x_{13},\ \ldots,\ x_{1n_1}$$

$$x_{21},\ x_{22},\ x_{23},\ \ldots,\ x_{2n_2}$$

$$\vdots \quad \vdots$$

$$x_{k1},\ x_{k2},\ x_{k3},\ \ldots,\ x_{kn_k}\ .$$

任意の正答率 x_{ij} は次の様に分解することが出来ると考える。

$$x_{ij}=b+a_i+\varepsilon_{ij}.$$

ここで，a_i は第 i 番目の問題群の難易性を反映する parameter で

$$\sum_{i=1}^{k} n_i a_i=0$$

を満足し，ε_{ij} は平均値が0で，i, j に depend しないある未知の分散をもって正規分布し，かつ，すべての i, j について互に独立である様な確率変数である。我々の分散分析の目的は各問題群の正答率の間に差異がみとめられるか，どうかをみることである。すなわち，parameter a_i が各問題群毎に等しいか，どうかを判定することである。さらに詳しくいえば，$\sum_{i=1}^{k} n_i a_i=0$ であるから

帰無仮設　H: $a_1=a_2=\ldots a_k=0$

を統計的に検定することである。

さて，

$$\bar{x}_{\cdot}=\frac{1}{n_i}\sum_{j=1}^{ni} x_{ij},$$

$$\bar{x}=\left(\sum_{i=1}^{k}\sum_{j=1}^{ni} x_{ij}\right)\Big/\sum_{i=1}^{k} n_i$$

とおく。この時，分析の結果は次の様な分散分析表にまとめることが出来る。

変動因	自由度	平方和	平均平方和
問題群間	$k-1$	$\sum_{i=1}^{k} n_i(\bar{x}_{i\cdot}-\bar{x})^2 \equiv S_1$	$S_1/k-1$
問題群内	$\sum_{i=1}^{k}(n_i-1)$	$\sum_{i=1}^{k}\sum_{j=1}^{ni}(x_{ij}-\bar{x}_i)^2 \equiv S_2$	$S_2 \Big/ \sum_{i=1}^{k}(n_i-1)$
全体	$\sum_{i=1}^{k} n_i-1$	S_1+S_2	

第 1 表　**分 散 分 析 表**

この分散分析表から

$$F=\frac{S_1}{k-1}\Big/\frac{S_2}{\sum_{i=1}^{k}(n_i-1)}$$

を求めることが出来るが，この統計量は，帰無仮設 H の下では自由度 $f_1=k-1$, $f_2=\sum_{i=1}^{k}(n-1)$ の F-分布をすることが判っているから，これを帰無仮設 H の検定基準にとることが出来る。すなわち，実際の data を使って，上の分散分析表の各欄の具体的数値を計算し，それから F の実現値 $\bar{F}$ を求めて

$\bar{F}\geqq F_{1-\alpha}[k-1, \sum_{i=1}^{k}(n_i-1)]$ なら H を reject し，$\bar{F}<F_{1-\alpha}[k-1, \sum_{i=1}^{k}(n_i-1)]$ なら H を accept するという方針で検定を行えばよい。ただし，$F_{1-\alpha}(f_1, f_2)$ は自由度 f_1, f_2 の F 分布の $((1-\alpha)\times 100)$ %点の値である。

なお，以上の説明は，全ての問題群が同じ正答率をもつとみなしてよいか，どうかの検定であるが，任意の問題群をふたつづつ組みあわせることによって，そのふたつの間の差異を検定することができる。これは上で，とくに k を 2 と考えた場合に他ならない。ところで，我々はさらにもうひとつの統計的分析をこころみた。実は，これは，結果からいえば，次節と附録第3, 4表にみられる様に，必ずしも，かんばしい finding をもたらすものではなかったが，とにかく，簡単に説明しておこう。それは，「計算機 program」の正答率と他学科の成績との間の関連をとらえ様とするこころみである。

具体的には，我々の sample となった30名の受験者の正答率と他の学科の成績との間の相関と回帰を求めるということである。これは直接には FŌRTRAN language の難易性を検出するものではないが，電子計算機の programming の能力が他のどんな能力を相伴っているかを知る上の1つの資料となりうるという意味で興味があると思われる。我々は30名の sample 受験者の大部分が受講していた「他学科目」として次のものを選び出した。

1　理　　科（化学，生物，物理）
2　経済学概論
3　統　　計
4　商学概論
5　簿　　記
6　数　　学
7　管理科学概論
8　文　　学（日本文学，ロシア文学，中国文学，英文学）
9　経済史概論
10　経済原論

第 2 表　**他学科目一覧**

なお，相関分析としては

「計算機 program」正答率（y）：$y_1, y_2 \cdots, y_n$,

他学科目の成績　　（x）：$x_1, x_2, \ldots, x_n$

から計算される相関係数 r：

$$r=\sqrt{\frac{\left(\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})\right)^2}{\sum_{i=1}^{n}(x_i-\bar{x})^2\sum_{i=1}^{n}(y_i-\bar{y})^2}}$$

の他に，

「計算機 program」の順位（R_0）：$R_{01}, R_{02}, \ldots, R_{0k}$,

他学科目の順位　　（R_{1i}）：$R_{11}, R_{12}, \ldots, R_{1k}$

により求められるスピアマンの順位相関係数 r'：

$$r'=1-\frac{6\sum_{i=1}^{k}(R_{0i}-R_{1i})}{k(k^2-1)}$$

も計算した。回帰分析は通常の最小二乗法によって

$$y=\alpha x+\beta$$

の α, β の値を推定した。よく知られている様に，それらの推定値 $\hat{\alpha}$, $\hat{\beta}$ は

$$\hat{\beta}=\frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sum_{i=1}^{n}(x_i-\bar{x})^2}$$

$$\hat{\alpha}=\bar{y}-\hat{\beta}\bar{x}$$

によってあたえられる。

最後に，この節をおえるにあたって，我々が分析の対象とした data に関して分散分析を適用することに伴う問題点にすこしふれておこう。一般に test 問題に対する解答の出来具合は次の様な要因に依存していると考えてもよいであろう。

1. 受験者の condition
2. 問題の出し方
3. その問題に関連する分野の難易度

我々の分析は，少し極端にいえば，1, 2 の要因を無視するか，あるいは一定とみなして，3 の要因のみが主として正答率を動かすものである，と解釈しようとしているのであるが，果して，これは第 1 次接近としても，みとめられるであろうか。すなわち，もし 3 以外の要因が大きく作用する様なことがあれば，問題群別に分類した data が群間ばかりでなく各群内でかなり大きく変動するため，必ずしも分散分析が適用可能な data ではなくなる。このことは，正答率 data を発生させている実際の test が，問題作成その他に管理された実験であるとみなせるか，どうかということに強く依存しているが，この点に関しては，すでに述べた様に多少疑問のあるところである。しかし，我々は 2 の要因の影響を出来るだけ除くために，個別問題を問題群に

ふりわける際にかなり慎重にふるまうと共に，極端な値をもつとみなされるdata は予めこれを除く様にした。また，1 の要因については，前述した様な受験者の選び方以外には特に考慮していないが，経験的にいって，受験者の成績は平均点によって順位をつけていくと，平均点そのものはかなり変動するが，順位は毎回かなり安定したものとなる様である。従って，我々は以上の様な data からでも，3 の要因の効果を分散分析によってある程度は区別することが出来るのではないかと考えている。

4. 統計的分析の結果

我々は **3** で述べた様な data と分析方法により，附録の表に示した結果をえることが出来た。これらの結果は，FŌRTRAN language の難易性に関して，また，FŌRTRAN language の理解度に関して，我々に何を呈示してくれるであろうか。本節ではそれらについて考察する。

まず sample 受験者 30 名の test の解答を data として，各問題群に分類し正答率を算出した結果の，各問題群の平均正答率についてである。附録第 1 表をみると，どの問題群の平均正答率が低いかがわかる。この平均正答率の順位（附録第 1 表－附表 2）は，問題群の難易性に関してのある information を我々に与えてくれる。それによれば，SUBRŌUTINE subprogram に関するもの，formatted READ statement，また，declaration，配列に関するもの，Record に関するもの等は，80％以上の高い平均正答率を示している。それに比較して，Arithmetic Assignment statement，subscript に関するもの，DŌ statement，そして，FŌRMAT の反復指定に関するもの，formatted WRITE statement 等は 50％以下の低い値である。さらにこの順位表では，FŌRMAT statement の field description に関する問題群は，ほぼ 60～70％の平均正答率を示しかなりの理解度を表わしている。すなわち，平均正答率のみで FŌRTRAN language の難易性を断定することは出来ないが，大体において平均正答率の高いものは理解され易いもの，平均正

答率の低いものは理解され難いものということが出来る。この様に平均正答率の順位表が我々に与えてくれる information は，FŌRTRAN language の学習において理解されづらい点はどこか，容易に理解される statement は何かということの他に，FŌRTRAN language が compiler language として使用される際，欠陥となり error を犯しやすいのはどの部分か，不便なのはどの部分かという，FŌRTRAN language そのものの weak-point についての若干の示唆である。すでに我々は，東大センターの諸報告から FŌRTRAN language で programming する時の error の頻度の分布をみてきた。東大センターの報告が program の error message の度数分布の調査であり，我々の平均正答率の順位表が学習の効果をみるために用いられた test の data による調査であるという概念上の相違はあるけれども，programming の段階で犯されやすい error は，恐らく，学習上の理解と密接に関連すると思われるので，以下両者の比較をこころみることとする。東大センターの集計報告による error 頻度の高い「変数に関する error」や「FŌRTRAN の statement に関する error」また「statement number に関する error」は，実際の programming の段階で表われるものであるため，我々の data には含まれていなかった。我々の結果で平均正答率 45.29 %の Arithmetic Assignment statement は東大センターの集計「IV. 算術式に関する error (8.1 %)」と調和している。すなわち，Arithmetic Assignment statement は program error も多く，学習上の理解度も悪いということになる。同様に，平均正答率 41.70 %の DŌ statement も「VI. DŌ loop に関する error (5.6 %)」と調和している。FŌRMAT statement については，東大センター集計の「IV. FŌRMAT statement に関する error (8.1 %)」の「㉑ FŌRMAT statement の仕様の書き方に誤りがある (1.6 %)」が，field description として我々の data の問題群 6 (I-type), 7 (F-type), 8 (E-type), 9 (A-type), 10 (L-type), 11 (H-type) と対応する。東大 program 相談室にもち込まれる error の件数が 1 番のわりには，我々の

field description の平均正答率はそれ程低い値を示してはいない。むしろ，「FŌRMAT の反復指定に関するもの」，「formatted WRITE statement」が平均正答率30%台であることから，FŌRMAT statement においては，field description そのものよりは，list と data と field description の各々の対応関係の方に問題があるのではなかろうかと推察される。東大センターの報告にはない部分で，我々の結果が示していることとしては subprogram についてである。すなわち，subprogram の中でも SUBRŌUTINE subprogram の方が FUNCTION subprogram よりも高い理解度を示し，SUBRŌUTINE subprogram の中でも argument のない方が，argument を含んでいるものよりも易しいのである。また，declaration, 配列に比べて subscript の概念が理解し難いことが，平均正答率の開きに示されている。

以上大まかに我々の平均正答率の順位表の意味することを，東大センターの報告との関連で述べてきた。しかし，各問題群の中で正答率がバラついているために，平均正答率だけで FŌRTRAN language の難易を測る事は不十分である。そこで我々は **3** で述べた様な分散分析を行うことにより，より正確な information をえることをこころみた。平均正答率が著しく異なる2つの問題群の間でも，問題群内のバラつきが激しい場合には有意差が表われない事を考慮して，全ての問題群相互の分散分析を行った（附録第2表参照)。その結果，2つの問題群の間で1%で有意のもの27組，5%で有意のもの33組となった。このことは各問題群の間に明らかな有意差の認められるものがあることを示している。我々は，20の問題群の中からこれら分散分析の結果と平均正答率から，以下の6つの問題群が FŌRTRAN language の中でも比較的理解困難な部分であると解釈することが出来るのである。次にそれらの問題群ひとつひとつについて，具体的に問題群のどの部分が FŌRTRAN language の理解を妨げているかについて吟味することにする。

Arithmetic Assignment statement：FŌRTRAN language の学習の最初に現われる実数型（real type）と整数型（integer type）の概念は仲々理解され

にくい。それは，数学で用いられる“整数”“実数”の概念が，program language 独特の type の概念に change するには，少なからぬ抵抗がある為と思われる。そして算術式の中で整数型と実数型を混ぜて用いることが禁じられていることは，Arithmetic Assignment statement を誤りやすいものとしている。これは注意深く check すると容易に除かれる性質のものであるが，それでもなお，programming したことのある者なら経験したであろうが，program の中にこの種の error が必ずあるといってよい程犯しやすいものである。また，FŌRTRAN では最初の文字が I, J, K, L, M, N の6文字で始まる変数は integer type であるということが，暗黙に指定されているが，そのことがかえって，その暗黙の指定を不注意に忘れた時に，error を発生させる原因となる場合もかなりある。もし，FŌRTRAN language において，算術式の中での real type と integer type の混合が許される様になったならば，Arithmetic Assignment statement の大部分の error は防げることであろう。

DŌ statement：DŌ statement に関しては，出題がわずか2題であるので，一般性は期待しえないかも知れないが，我々の結果では，DŌ loop の loop 内の index や parameter を変更している program を訂正させる問題（附録—試験問題例［3］の5－2）について，完全に解答しているのは全体の23.3％である。これは，DŌ loop の index はその範囲に入る前に完全に set されていなければならない，すなわち，DŌ の範囲の中では，index や parameter の値を再び定義する如何なる statement も許されないという文法が理解されていないためであると思われる。

subprogram：FŌRTRAN language の subprogram には SUBRŌUTINE subprogram と FUNCTIŌN subprogram がある。分散分析の結果この2つの subprogram の間には明らかな有意差がみられた（附録第2表—附表1）。有意差がもたらされた原因と思われるものに，argument の概念の理解の難しさがある。argument そのものの概念，dummy argument と actual

argument の個数・順序・型の対応，dummy argument, actual argument 各々に付加されている制約等々は subprogram の難点である。SUBRŌUTINE subprogram の中でも，argument のあるものと，ないものとの間で理解度が異なっているのも，この argument の理解の困難さに起因するものと思われる。

GŌ TŌ statement：GŌ TŌ statement について述べる前に，3 でふれた様な理由で，GŌ TŌ statement の出題が，computed GŌ TŌ statement に関するものばかりで，unconditional GŌ TŌ statement を含んでいないことを注記しておく必要があろう。この statement を集計しながら気付いたことは，computed GŌ TŌ statement である GŌ TŌ $(k_1, k_2, \ldots, k_n)$, i の statement lebel $(k_1, k_2, \ldots, k)$ には誤りがないが，i が添字のない整数型変数であるという文法が忘れられていることである。GŌ TŌ statement は，programming 上 statement number と結びつくが，ここではそのことに関する data はなかった。

declaration, 配列と subscript：東大センターの program 相談室にもち込まれる相談の中で比較的件数の多い「subscript, DIMENSIŌN に関するもの」を，我々は，問題群 4「declaration, 配列に関するもの」と問題群 17「subscript に関するもの」に 2 分してとらえた。その結果，"DIMENSIŌN" を "DIMENTIŌN" "DEMENSIŌN" 等とする spelling の careless miss を無視した場合，問題群 4 は平均正答率 88.30 %の高い値を示している。それに比べて，subscript に関する問題群17の方は，非常に低い平均正答率を示しているが，それは，subscript の形が，C, I±C, C*I±C′, C*I (C, C′ は整数定数，I は整数変数）の 7 つ以外は許されないということが，不用意に誤られるからである。

FŌRMAT statement：FŌRMAT statement は，FŌRTRAN language の最大の難関といわれてきた。我々はこの FŌRMAT statement をいくつかに分類してみることで，FŌRMAT statement の中にも比較的理解され易いも

のと，仲々理解されないものがあるという finding をしたのである。一括して I/Ō statement として考えられている formatted READ statement と formatted WRITE statement は，我々の分析結果では，その理解度にかなりの差があることになっているが，それは，formatted WRITE statement の出題形式が，formatted READ statement のそれに比べてはるかに積極的な解答を要求しているということが，大きな原因として結果に作用していると思われるので，これは apriori には言いえない。だが，input の時には data と list と field description の3つの対応関係が問題となり，output の時には list と field description の2つが対応しなければならず，さらに，input, output の field description において，小数点のとり扱いが必ずしも対称的でないなど，formatted READ statement と formatted WRITE statement は，それぞれ別の性格をもっていると思われる。それ故，それらについては，今後，I/Ō statement としてまとめて考えられるよりは，各々の特色の中にある困難性についての追求がなされるべきではなかろうか。FŌRMAT statement の field description に関する問題群6，7，8，9，10，11は，positive な解答を要求する問題形式ではなかったが，formatted READ statement, formatted WRITE statement に含まれた field description に関する部分と考え合わせてみても，理解するのにそれ程困難な statement ではない様に思われる。FŌRMAT statement の中で理解され難いと思われるものは，FŌRMAT の反復指定に関するものである。これは，FŌRMAT の中の括弧や / (slant), , (comma) の意味が正しく把握されていないためであろう。また，前述の data と list と field description の個数の対応の問題も，FŌRMAT の反復にかかわってくるものである。

以上，我々の結果が呈示してくれると思われる種々の推論を展開したが，これらは，我々が本稿で使用した data そのものの性質からも，あくまでも，FŌRTRAN language の難易性や理解度に関しての1つの提言である事を改めて断っておかなければなるまい。

本稿のもう1つの作業は，「計算機 program」と他学科目との相関関係に関する統計的分析である。我々の目的は，電子計算機の programming の能力がどの様な能力を相伴うものであるかということを検討するところにあった。現在では，電子計算機という字面から programmer を数学専攻者ないしは理工科系出身者に求めようとする傾向が著しい。これは，計算機 programming の能力という観点から妥当なのであろうか。programming 能力の基本的なものとは数学的な能力だけなのであろうか。この様な疑問に対して，我々は，「計算機 program」と他学科目の成績との関連を分析することで何らかの解答がえられることを期待したのである。

この分析にあたり，2つの場合について考察した。1つは，受験者全員についてであり，もう1つは，受験者のうち平均正答率 60.0％以上の者についてである。また sample 受験者全員が10学科全てを履習していなかったため，次の様に，各学科目の sample 受験者数に違いがあった。

	学科目		全員	平均正答率60％以上
1	理科	（化学，生物，物理）	30 人	20 人
2	経済学概論		30	20
3	統計		30	20
4	商学概論		30	20
5	簿記		30	20
6	数学		28	18
7	管理科学概論		25	17
8	文学	（日本文学，ロシア文学，中国文学，英文学）	24	16
9	経済史概論		29	19
10	経済原論		28	18

第 3 表　学科目別 sample 受験者数

「計算機 program」と他学科目の相関関係についてみると，受験者全員の場合も，平均正答率 60.0％以上の者の場合も共に，相関係数は非常に低く，殆んど相関関係が成立しない程であった。しかし，その中でも相対的に高い平行的相関を示している学科目は，受験者全員を対象とした場合は，経済学

概論の +0.27645 を始めとする数学・文学・統計等である。受験者を選択した場合（平均正答率 60.0％ 以上）は，文学が +0.44212 と最も高い相関関係を示し，経済学概論・数学・統計とつづいている。「計算機 program」と minus の逆行的相関関係を呈しているのは，どちらの場合も，管理科学概論・経済史概論であり，理科・商学概論は殆んど「0」に等しい。スピアマンの順位相関係数についても同じ様な結果がえられた。順位相関係数の高いものの順は，全員を対象とした時は，数学・商学概論・文学・簿記・経済学概論・理科・経済原論・経済史概論・管理科学概論・統計となっており，平均正答率 60.0％ 以上のものについては，数学・文学・商学概論・理科・簿記・経済原論・経済史概論・管理科学概論・経済学概論・統計の順となっている。しかし，危険率 2％で有意なものは，前者の数学（0.47838）のみ，危険率 5％で有意なものは後者の数学（0.44662），危険率 10％ で有意なものは前者の商学概論（0.33838），後者の文学（0.43653）だけである。すなわち，スピアマンの順位相関係数から「計算機 program」と平行的相関関係をもつのは，数学・文学・商学概論の 3 学科という結果になった。回帰分析の結果は附録第 3 表—附表 2 に示した様に殆んどその意味をなしていないが，参考までに掲げておいた。理科・経済学概論・統計・商学概論・簿記の 5 学科は同時に回帰分析を行い，他学科は別々に回帰分析を行った。

以上，相関係数，順位相関係数，回帰分析の結果を呈示したが，これらは必ずしも好ましい結果ということが出来ない。その原因は我々が使用した data にあると思われる。すなわち，data としての「計算機 program」の成績と，他学科目の成績との間に，比較検討するには不適当な factor があったと思われるのである。その factor としては，さしあたって 3 つ程あげられる。1 つは，「計算機 program」の成績は毎授業時間のあとに行った 12 回の test の結果をもとにしたものであり，成績にある程度の平均化がなされていると思われるが，他学科目の data は，少数回の test の結果によるために，様々の偶然的な事柄が，「計算機 program」の成績に比べて強く作用

しているのではなかろうかということである。2つめは，「計算機 program」と他学科目の test の形式の相違である。「計算機 program」は客観的な achievement 形式の問題が大部分であるが，他学科目は大体において「……について述べよ」式の論文形式の出題であったと考えられる。その相違は，当然，採点，配点の相違を導くのであり，こそが3つめの factor である。「計算機 program」は「1」か「0」かの客観的採点方法を採った。しかし，他学科目は主観的要素をかなり含んだ採点，配点となるため，成績には幅があると思われる。我々が使用した data の中に含まれていると思われる問題点は，大体以上の如くである。なお，我々が意図したことを，成分分析法を用いて行った相良［3］の論文があるので，参照されたい。

5. 残された問題と2・3の示唆

前節で述べた我々の分析結果は，FŌRTRAN language に関するものであったが，今後はこの種の分析を ALGŌL language, CŌBŌL language に関しても行う必要があると思われる。本稿の最初の節でも指摘した様に，FŌRTRAN と ALGŌL はまったく違った背景から出来上ってきたので programming language としては，両者はかなり異ったものである。従って，ALGŌL languageは FŌRTRAN language とはまた違った困難性を初学者に与えるであろう。実際，本稿では分析の対象にはしなかったが，ALGŌL language に関する achievement test の結果を簡単に検討しただけでも十分にそのことを予想することが出来る。

これに関連して，初学者を対象に programming language による電子計算機の programming を instruct する際に，FŌRTRAN language から始めるのが適当か，ALGŌL language から始めるのが適当かという問題がおこってくるが，いま，にわかにそのどちらがよいと判定を下すことが出来ないにしても，我々のごく僅かな経験的事実から推量するならば，入門用の language としては FŌRTRAN language の方が適当ではないかと考えられ

る。なぜならば，初学者にとって ALGŌL の BLŌCK structure とそれに関連する事項などは正確な理解と応用が殆んど不可能であると思われるのに反して，FŌRTRAN には，その様な複雑な構造が許されていないので割合に理解と応用が容易だからである。少し digress すれば，ここで「初学者」にとってという限定はかなり重要であって，これがある程度迄 FŌRTRAN, ALGŌL を理解した者については，まさに，逆のことがいえる様に思われる。すなわち，彼らは科学＝技術計算に対しては FŌRTRAN language よりは ALGŌL language の方を使う機会が多いといえそうである。勿論，正確に data をとって比較検討したわけではないから断言は出来ないが，それにしても，この様な傾向は FŌRTRAN language が単に programming language であるのに対して，ALGŌL language は programming language であると同時に algorithm 用の language であるという性格を反映するものであろう。それはともかくとして，FŌRTRAN language と ALGŌL language を初学者の立場から比較することは興味深いことである。世界的にみてもこの2つの language が電子計算機の共通の language として通用している現在，両方の language を一定の時間内に教育することが望まれる場合が多いが，その際，上の様な比較研究は FŌRTRAN, ALGŌL をどの様に instruct すべきかという問題に1つの指針を与えるであろう。いいかえれば，2つの language を別々に切り離して instruct することはきわめて能率が悪いし時間もかかるので，これらの研究が，FŌRTRAN, ALGŌL をどの様に組み合わせて教育すればよいかに関する経験的事実を明らかにするならば有益であろう。

同様なことは CŌBŌL に関してもいえる。CŌBŌL 自身のもつ初学者に対する困難性の分析と他の language との比較研究も，必ずや何らかの finding をもたらすものと思われる。

ところで，我々の分析は次の様な1つの特殊な側面をもっている。それは，我々の分析の data は小樽商科大学という文科系大学の特定の学科（管

理科学科）の2年度生からえられたものが大部分であり，このことが，すでに data のある種の偏りを暗示しているということである。そこで，同様な分析をもっと一般的な集団からえられた data についてもこころみるべきであろう。勿論，電子計算機の programming に全く興味のない者に無理に programming language を教えるわけにもいかないので，例えばいろいろな機会に行われる programming language の講習会に出席した者から，出来るだけ多くの data をうる様にしたいものである。この様にして次第に分析の対象範囲を広げていくことによって，より信頼度の高い結論がえられるであろうことはいうまでもない。

なお，本稿で行った分析はいずれも「初学者」を対象としたものであったが，3節でもふれた様に，このことはみかけほど，この種の分析の結論に対する制約条件とはならないと我々は考えている。その理由の1つとしては，初学者の困難の大部分は次第に解消していくものではあるけれども，それは学習者が次第に programming language の種々の約束になれていくためであり，その意味では programming language の学習過程は人間の適応能力に依存しており，決して programming language の困難性が学習過程を通じて次第になくなるのではないことがあげられると思う。従って，初学者の困難の**あるもの**はそのまま language そのものの難点と考えてよいであろう。そのひとつの例として，我々が前節で指摘した arithmetic expression の困難性に関する問題をとりあげてみる。

今日の FŌRTRAN language のいくつかは依然として乗巾の場合を除いて数式の中で type の異なる定数，変数，配列の要素を使ってはならない様になっている。これは，実際に program をかく時には，一見した以上に厳しい制約であって，FŌRTRAN language では数式をほぼそのままかけるという利点も，この制約によって殆んど帳消しになってしまいかねないといっても過言ではないであろう。なお，同じことは数式に type の混合を許さない ALGŌL language についてもいえる。今日では，type を混合しても差

支えない様に作られた FŌRTRAN compiler, ALGŌL compiler もかなり存在しているが，もとより compiler はその様に作られるのが望ましいことはいう迄もない。この点，PL/1 は文法上もはっきり type の混合をみとめている。FŌRTRAN language が type の混合をみとめなかったのは，恐らく，最初に implementation が行われた 704 による object program の run time を短縮するためであったと思われるが，現在の電子計算機の事情は 704 の当時とは比べものにならない程改善されているから，type の混合をみとめて，その結果，例えば object program において，type を統一するために適当な変換を演算に先立って行う様にしても，そのために，とくに object program の run time が長くなって困るということはないであろう。従って，当然，今後の全ての FŌRTRAN language からは type の混合はみとめないという不必要な制約は除くべきで，type を混合した時は，実数型として演算が行われる様にすべきである。

次に，我々の分析でも明らかな様に，FŌRTRAN language の入出力＝FŌRMAT statement は初学者にとって殊に困難な様である。それはよくいわれている様に FŌRTRAN language の入出力＝FŌRMAT statement が machine language の要素をきわめて多く持っているからに他ならない。これは programming language としての FŌRTRAN のひとつの大きな難点であろう。これも最初に FŌRTRAN compiler が作られた 704 の制約に由来するものであると考えれる。しかし，今日の段階からみれば，入出力＝FŌRMAT statement がその様なものに限られる必要は少しもなく，もっと machine language 的な要素の少い入出力＝FŌRMAT statement が設計されなければならないであろう。このことは入出力が一定の format でしか行われないという犠牲を払っても入出力＝FŌRMAT statement をもっと簡略化するということを意味しているのであって，仮にそうなっても，以下の理由によって FŌRTRAN language が特に不自由になるとは考えられない。元来 FŌRTRAN language で電子計算機の全ての機能を働かせることは不

可能なことなのであり，その様な FŌRTRAN が入出力の形式においてかなりの自由が許される入出力＝FŌRMAT statement をもっていることは，いわば不釣合な「贅沢さ」であって，programming language としては思想的に統一がとれていないといえるであろう。しかも FŌRTRAN language を科学＝技術計算向きの language としてとらえる限り，入出力の形式が硬直的であってもそれ程の困難はおきないと考える。実際，FŌRTRAN 的入出力＝FŌRMAT statement が必要とされる場面は，科学＝技術計算におけるよりも，事務計算に多いのである。ところで，どの様に入出力＝FŌRMAT statement を簡略化したならばよいかという問題に，直ちに答えることは困難であるが，ひとつの方法は，FŌRMAT などでこまかく指定しないときはある標準の style での入出力しか行うことが出来ないが，その標準の指定は何らかの方法によって変更することが出来る様にすることであろう。この標準の指定という考え方は PL/1 にとくに顕著に表われている思想である。PL/1 の program ではこまかい指定がない時は，compiler の方で自動的に標準の場合とみなされる様になっている。これは，従来の programming language のわずらわしい rule に悩まされていた初学者（並びに一般の programmer）にとって困難性のかなりの逓減を意味するものである。

さて，最後に我々の分析が示唆するものについて簡単にふれておくことにしたい。我々の分析は全体として，FŌRTRAN language の習得が必ずしも容易でないことを示している。しかし，そのことは FŌRTRAN language の全てが理解困難であるということではないのであって，FŌRTRAN language の特定の個所が他の個所よりも理解しにくい傾向がみられるということである。このことは，我々の分析の結果が，ある程度迄，他の機会の別の分析の結果と調和する部分をもっているということからもうかがい知ることが出来る。そこで，その様な傾向がどの位一般的なものであるかということを，さらに範囲を広げた data の分析によってたしかめる必要があるのは勿論であるが，たとえ，この様な tentative な結論であっても，新しい

programming language の設計に際して1つの論点を提供するであろう。それと共に，これが FŌRTRAN language の instruction をより合理的に進める上に参考として役立つことは見逃せない。従って，ここでは experienced programmer は必ずしも good instructor ではありえないことを強調しておくのが適当であろう。すなわち，FŌRTRAN language の教育も，他の語学教育と同様に，初学者が何を困難に感ずるかという data に基いて，綿密に計画を立てて行うことが望まれるのである。しかも，FŌRTRAN language の目的から言って，文法の理解が一応出来ただけでは殆んど役に立たないのであって，実際に，その知識を応用して program がかけなければ何にもならない。この意味では森口[7]はきわめて優れた入門書であり，FŌRTRAN programming においては，座右の書として手放すことが出来ないが，我々の観点からは，前節に指摘した様な困難性のある statement に関してさらに多くの例題が補充されることを嘱望したいのである。

以上，きわめて常識的なことに紙面を費してきたが，本稿が電子計算機と programming language 普及の陰に user の立場から多少の問題点が潜んでおり，それらは組織的にのみ解決されるべき性質のものであることを指摘することに成功しているならば幸いである。

＜引 用 文 献＞

[1] S. Rosen : *Programming Systems and Languages*, New York, McGraw-Hill, 1967.

[2] Southworth, Raymond W. and Samuel L. Deleeuw : *Digital Computation and Numerical Methods*, New York, McGraw-Hill, 1965.

[3] 相良信子「プログラマの適性について」，情報処理，1965年9月号 (Vol. 6, No. 5)。

[4] ダイヤモンド社「数理科学，特集 FŌRTRAN」，1968年2月号。

[5] 「東京大学大型計算機センター広報」，第8号，昭和41年5•6月，東大出版会。

[6] 日立製作所，HITAC 5020, 5020 E/F FŌRTRAN (HARP) 5020-3-029-01，昭和41年9月。

[7] 森口繁一「FŌRTRAN IV 入門」，東京大学出版会，1965年。

附録第 1 表

Data No.	問題群	問題の内容	試験問題	学生番号 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	正答率 %	平均正答率 %	標準偏差
1	1	Arithmetic	7-2(1)	0,	1,	0,	0,	0,	0,	0,	1,	1,	1,	0,	0,	0,	1,	0,	0,	1,	1,	1,	0,	0,	1,	1,	0,	1,	0,	1,	1,	0,	0,	43.3		0.495
2		Assignment	(2)	1,	0,	1,	0,	1,	0,	0,	1,	0,	0,	0,	0,	1,	1,	0,	0,	0,	0,	0,	0,	1,	0,	0,	1,	0,	0,	0,	0,	0,	1,	33.3		0.471
3		statement	(3)	0,	1,	1,	0,	1,	0,	0,	0,	0,	0,	0,	0,	0,	0,	1,	0,	0,	0,	0,	0,	0,	0,	1,	0,	1,	0,	0,	0,	1,	0,	23.3		0.422
4			(4)	1,	1,	1,	1,	0,	1,	0,	0,	1,	0,	0,	1,	0,	1,	0,	0,	1,	1,	0,	0,	0,	0,	0,	1,	1,	0,	0,	0,	0,	1,	43.3		0.495
5			7-4(1)	0,	0,	0,	0,	1,	0,	1,	0,	1,	1,	0,	0,	0,	1,	0,	0,	0,	1,	0,	1,	1,	0,	0,	1,	0,	0,	0,	1,	0,	1,	36.6	45.29	0.481
6			(2)	1,	0,	0,	0,	1,	1,	1,	0,	1,	1,	0,	0,	0,	1,	0,	0,	0,	1,	0,	1,	1,	0,	0,	1,	0,	0,	0,	1,	0,	1,	43.3		0.495
7			(3)	1,	0,	0,	1,	1,	1,	1,	0,	1,	1,	0,	0,	0,	1,	0,	0,	0,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	0,	1,	0,	1,	63.3		0.481
8			(4)	1,	0,	0,	1,	1,	1,	1,	0,	1,	1,	0,	0,	0,	1,	0,	0,	0,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	0,	1,	0,	1,	63.3		0.481
9			(5)	1,	0,	0,	0,	1,	1,	1,	0,	1,	0,	0,	0,	0,	1,	0,	0,	0,	0,	0,	0,	1,	1,	0,	0,	1,	0,	0,	0,	1,	0,	1,	36.6		0.481
10			11-4	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	0,	1,	0,	1,	0,	1,	0,	1,	0,	1,	1,	0,	0,	1,	0,	1,	1,	0,	66.6		0.471
11	2	GŌ TŌ statement	9-1	1,	0,	0,	0,	0,	1,	0,	0,	1,	0,	0,	0,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	0,	1,	1,	0,	0,	0,	0,	50.0		0.500
12			10-1(1)	0,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	0,	0,	0,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	80.0		0.400
13			(2)	1,	1,	0,	1,	0,	1,	1,	1,	1,	0,	0,	0,	1,	1,	1,	1,	0,	1,	0,	0,	1,	1,	0,	1,	0,	1,	0,	1,	0,	1,	1,	60.0	65.20	0.489
14			(3)	0,	0,	1,	1,	1,	1,	0,	0,	1,	0,	0,	0,	1,	1,	0,	1,	1,	1,	1,	1,	1,	0,	0,	1,	1,	0,	1,	1,	0,	1,	1,	60.0		0.489
15			(4)	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	0,	0,	0,	0,	1,	1,	1,	1,	1,	1,	0,	0,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	76.6		0.422
16	3	DŌ statement	4-4	1,	1,	0,	1,	1,	1,	1,	0,	0,	0,	1,	0,	1,	1,	1,	1,	1,	0,	1,	1,	1,	0,	1,	1,	0,	1,	1,	0,	1,	0,	0,	0,	0,	60.0	41.70	0.489
17			5-2	0,	0,	0,	0,	1,	0,	0,	1,	1,	0,	0,	0,	0,	0,	0,	0,	0,	0,	0,	0,	1,	0,	0,	0,	1,	0,	0,	1,	1,	0,	0,	0,	23.3		0.422
18	4	declaration, 配列に関するもの	4-5	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	90.0	88.30	0.300
19			5-1	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	86.6		0.339
20	5	(1) formatted READ statement	3-3	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	0,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	0,	0,	0,	1,	1,	1,	80.0	90.00	0.400
21			4-1	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	100.0		0.000
22		(2) formatted WRITE statement	5-3	1,	0,	0,	0,	0,	0,	1,	0,	0,	0,	0,	1,	0,	0,	0,	0,	0,	1,	0,	0,	0,	1,	1,	1,	0,	0,	0,	0,	1,	0,	0,		26.6	31.60	0.442
23			8-5	1,	0,	0,	1,	0,	0,	0,	0,	1,	0,	0,	0,	1,	0,	0,	1,	0,	0,	0,	0,	1,	0,	0,	0,	1,	1,	1,	0,	0,	1,	0,	0,	1,	36.6		0.481
24	6	FŌRMAT statement I-type	6-12	1,	1,	1,	1,	1,	1,	0,	1,	0,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	90.0	71.65	0.300
25			6-25	0,	1,	0,	0,	0,	1,	0,	1,	0,	0,	0,	1,	1,	1,	0,	0,	1,	1,	1,	0,	1,	1,	1,	0,	1,	1,	0,	1,	0,	1,	53.3		0.498
26	7	FŌRMAT statement F-type	6-5	1,	0,	1,	0,	0,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	0,	0,	1,	1,	0,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	0,		70.0		0.458
27			6-11	1,	1,	1,	1,	1,	1,	1,	1,	0,	0,	0,	0,	0,	1,	1,	1,	0,	1,	1,	0,	0,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	0,	1,	70.0	66.67	0.458
28			6-22	0,	0,	0,	1,	0,	0,	1,	1,	0,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	0,	1,	0,	0,	1,	0,	1,	1,	60.0		0.489
29	8	FŌRMAT statement E-type	6-4	0,	1,	0,	1,	1,	1,	1,	1,	0,	1,	1,	1,	1,	0,	1,	0,	1,	1,	1,	0,	1,	1,	1,	0,	0,	1,	1,	1,	1,	1,		73.3	73.30	0.442
30			6-24	0,	1,	0,	1,	1,	1,	0,	0,	0,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	0,	0,	1,	1,	1,	1,	1,	1,	1,		73.3		0.442
31	9	FŌRMAT statement A-type	12-1(1)	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	0,		86.6	69.57	0.339
32			(2)	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	0,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,		86.6		0.339
33			(3)	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,		90.0		0.300
34			(4)	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	0,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	0,	1,	86.6		0.339
35			(5)	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	90.0		0.300
36			(6)	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	86.6		0.339
37			(7)	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	0,	1,	1,	0,	0,	1,	0,	1,	1,	1,	1,	0,	1,	0,	1,	0,	1,	73.3		0.442
38			(8)	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	0,	1,	1,	0,	1,	1,	1,	0,	0,	1,	0,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	0,	1,	73.3		0.442
39			(9)	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	90.0		0.300
40			(10)	0,	0,	1,	1,	0,	1,	1,	1,	0,	0,	1,	0,	0,	1,	1,	0,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	63.3		0.481
41			(11)	0,	0,	0,	1,	0,	1,	1,	1,	0,	0,	0,	0,	0,	0,	1,	0,	0,	0,	1,	0,	0,	1,	0,	1,	0,	1,	0,	1,	0,	0,	1,	1,	36.6		0.481
42			(12)	1,	1,	1,	0,	0,	1,	1,	1,	0,	0,	0,	0,	0,	0,	0,	1,	0,	1,	1,	0,	1,	0,	1,	1,	1,	1,	1,	1,	0,	1,	0,	0,	50.0		0.500
43			(13)	1,	1,	1,	0,	1,	1,	1,	1,	0,	0,	0,	0,	0,	0,	0,	0,	0,	0,	1,	0,	1,	1,	1,	1,	1,	1,	0,	1,	1,	0,	0,	0,	46.6		0.498
44			(14)	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	0,	0,	1,	0,	0,	0,	0,	0,	0,	0,	1,	1,	1,	1,	1,	0,	0,	1,	1,	0,	1,	60.0		0.489
45			(15)	0,	0,	0,	1,	1,	1,	1,	1,	0,	0,	1,	1,	0,	0,	1,	0,	0,	1,	1,	0,	1,	1,	0,	1,	1,	0,	1,	0,	1,	1,	0,	0,	53.3		0.498
46			(16)	0,	1,	1,	1,	0,	1,	1,	0,	1,	1,	0,	0,	0,	0,	1,	1,	0,	1,	0,	1,	0,	1,	1,	1,	1,	0,	1,	0,	1,	1,	0,	0,	53.3		0.498
47			(17)	0,	0,	1,	1,	0,	1,	0,	1,	1,	0,	1,	1,	0,	1,	1,	0,	0,	0,	0,	1,	1,	1,	1,	1,	0,	1,	0,	0,	1,	1,	0,		56.6		0.495
48	10	FŌRMAT statement L-type	11-3(1)	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	0,	1,	1,	0,	1,	1,	0,	0,	1,	0,	1,	0,	0,	0,	0,	0,	1,	0,		56.6	50.00	0.495
49			(2)	1,	0,	1,	1,	1,	0,	1,	1,	1,	1,	0,	0,	1,	0,	1,	1,	0,	1,	1,	0,	0,	1,	0,	1,	0,	1,	0,	1,	0,	1,	0,		56.6		0.495
50			(3)	1,	0,	0,	0,	1,	0,	1,	0,	1,	0,	1,	0,	0,	1,	1,	1,	1,	0,	0,	0,	1,	0,	0,	0,	0,	0,	0,	1,	0,	1,		40.0		0.489
51			(4)	1,	0,	0,	1,	1,	0,	0,	1,	1,	0,	1,	0,	0,	1,	1,	0,	0,	0,	1,	0,	1,	1,	0,	0,	1,	0,	0,	0,	0,	1,		43.3		0.495
52			(5)	1,	0,	0,	1,	1,	0,	1,	1,	1,	1,	0,	1,	0,	0,	1,	1,	0,	1,	0,	1,	0,	1,	1,	1,	0,	0,	1,	0,	0,	1,	0,	1,	53.3		0.498
53	11	FŌRMAT statement H-type	6-13	0,	0,	1,	0,	0,	0,	1,	0,	1,	1,	0,	0,	0,	1,	1,	1,	1,	1,	1,	1,	0,	1,	0,	1,	0,	0,	0,	1,	1,	1,		53.3	59.96	0.498
54			6-21	1,	0,	0,	0,	0,	1,	1,	1,	0,	0,	0,	1,	1,	1,	0,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,		60.0		0.489
55			8-7	0,	1,	0,	1,	1,	1,	1,	1,	1,	1,	0,	0,	0,	1,	1,	1,	1,	0,	0,	1,	1,	1,	1,	1,	1,	1,	0,	1,	0,	1,	0,		66.6		0.471
56	12	Record に関するもの	2-5	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	0,	0,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,		86.6	86.65	0.339
57			6-7	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,		100.0		0.000
58			6-14	1,	1,	1,	1,	0,	0,	1,	1,	1,	0,	0,	0,	1,	1,	1,	1,	0,	1,	1,	0,	1,	0,	0,	1,	1,	0,	0,	0,	1,	1,	1,		60.0		0.489
59			6-17	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,		100.0		0.000
60	13	FŌRMAT の反復指定に関するもの	5-4	0,	1,	1,	1,	1,	1,	0,	1,	1,	1,	0,	1,	1,	1,	0,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	0,	0,	0,	0,	0,	1,		36.6	32.20	0.481
61			6-9	1,	1,	0,	1,	1,	1,	1,	0,	0,	1,	0,	0,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	0,	1,	0,	0,	1,	0,	0,	1,			33.3		0.471
62			6-16	0,	1,	0,	1,	1,	1,	0,	1,	1,	0,	0,	0,	1,	1,	0,	0,	0,	1,	1,	1,	0,	1,	0,	0,	0,	0,	0,	0,	0,	0,	0,		26.6		0.442
63	14	(1) SUBRŌUTINE 引数なし	8-1	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,		96.6	94.95	0.179
64			8-7	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,		93.3		0.249
65		(2) SUBRŌUTINE 引数あり	9-7	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	0,	0,	1,	0,	1,		83.3	83.30	0.372
66	15	FUNCTIŌN subprogram	10-4	1,	1,	1,	1,	1,	1,	0,	0,	1,	0,	1,	0,	1,	1,	0,	1,	1,	1,	1,	1,	1,	0,	0,	1,	1,	0,	0,	0,	0,	1,		63.3	56.65	0.481
67			10-6	0,	1,	1,	0,	1,	1,	0,	1,	1,	1,	0,	1,	0,	1,	0,	0,	1,	0,	0,	0,	1,	0,	1,	1,	0,	0,	0,	1,	0,	1,		50.0		0.500
68	16	標準関数	9-2	0,	1,	1,	1,	1,	1,	0,	1,	1,	1,	0,	1,	1,	1,	0,	1,	0,	1,	1,	1,	1,	1,	1,	1,	0,	0,	0,	0,	0,	1,		66.6	56.60	0.471
69			10-3	1,	1,	0,	1,	1,	1,	1,	0,	0,	1,	0,	0,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	0,	1,	0,	0,	1,	0,	0,	1,		63.3		0.481
70			9-4	0,	1,	0,	1,	1,	1,	0,	1,	1,	0,	0,	0,	1,	1,	0,	0,	0,	1,	1,	1,	0,	1,	0,	0,	0,	0,	0,	0,	0,	0,		40.0		0.489
71	17	subscript に関するもの	7-3(1)	1,	0,	0,	1,	0,	1,	0,	1,	1,	1,	1,	0,	0,	0,	1,	0,	1,	1,	0,	1,	0,	1,	0,	0,	1,	1,	1,	1,	1,	0,		56.6	42.45	0.495
72			(2)	1,	0,	0,	0,	0,	0,	0,	0,	1,	0,	1,	0,	0,	0,	0,	0,	0,	0,	0,	0,	0,	0,	0,	1,	0,	0,	0,	0,	0,	0,	0,		13.3		0.339
73			(3)	1,	1,	0,	0,	0,	0,	0,	0,	1,	0,	0,	0,	0,	0,	0,	0,	0,	0,	0,	0,	1,	1,	0,	1,	1,	0,	0,	0,	0,	0,	0,		23.3		0.422
74			(4)	1,	1,	1,	1,	0,	0,	1,	0,	1,	0,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	0,	1,	1,	1,		76.6		0.422
75	18	logical statement	11-1	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	1,	1,	1,	1,	0,	1,	1,	1,	1,	90.0	81.65	0.300
76			11-2	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	0,	0,	1,	1,	0,	1,	1,	0,	1,	1,	1,	1,	0,	1,	0,	0,	1,	1,	0,	1,		73.3		0.442

附録第1表—附表1

学生番号	1	2	3	4	5	6	7	8	9	10
正答率(%)	69.7	63.1	61.8	71.0	59.3	72.3	69.7	67.1	76.3	53.9
標準偏差	0.459	0.482	0.485	0.453	0.490	0.447	0.459	0.469	0.425	0.498
学生番号	11	12	13	14	15	16	17	18	19	20
正答率(%)	52.6	50.0	42.1	80.2	63.1	34.2	68.4	69.7	60.0	53.9
標準偏差	0.499	0.500	0.493	0.398	0.482	0.474	0.464	0.459	0.491	0.498
学生番号	21	22	23	24	25	26	27	28	29	30
正答率(%)	80.2	72.3	68.4	68.4	60.5	44.7	53.9	64.4	46.0	65.7
標準偏差	0.398	0.447	0.464	0.464	0.488	0.497	0.498	0.478	0.498	0.474

附録第1表—附表2　**平均正答率の順位表**

順位	平均正答率	問題群	
1	94.95	14 (1)	SUBRŌUTINE 引数なし
2	90.00	5 (1)	formatted READ statement
3	88.30	4	declaration, 配列に関するもの
4	86.65	12	Record に関するもの
5	83.30	14 (2)	SUBRŌUTINE 引数あり
6	81.65	18	logical statement
7	73.30	8	FŌRMAT statement E-type
8	71.65	6	FŌRMAT statement I-type
9	69.57	9	FŌRMAT statement A-type
10	66.67	7	FŌRMAT statement F-type
11	65.20	2	GŌ TŌ statement
12	59.96	11	FŌRMAT statement H-type
13	56.65	15	FUNCTIŌN subprogram
14	56.60	16	標準関数
15	50.00	10	FŌRMAT statement L-type
16	45.29	1	Arithmetic Assignment statement
17	42.45	17	subscript に関するもの
18	41.70	3	DŌ statement
19	32.20	13	FŌRMAT の反復指定に関するもの
20	31.60	5 (2)	formatted WRITE statement

附録第2表

群＼群	1	2	3	4	5 (1)	5 (2)	6	7	8	9
1		6.88*	0.09	17.60*	15.91**	7.59*	4.51	5.92*	6.91*	12.96**
2			3.06	6.86*	5.22	11.98*	0.22	0.03	0.72	0.24
3				6.94	5.35	0.28	1.34	3.04	2.97	4.01
4					0.03	115.27**	0.82	23.26*	77.85*	2.02
5(1)						27.28*	0.77	7.35	2.79	2.32
5(2)							4.43	37.94**	69.56*	8.25*
6								0.12	0.01	0.02
7									2.37	0.07
8										0.08
9										
10										
11										
12										
13										
14(1)										
14(2)										
15										
16										
17										
18										

10	11	12	13	14 (1)	14 (2)	15	16	17	18
0.44	2.76	19.82**	2.25	21.65**	6.24*	3.33	1.41	0.06	10.84**
5.38*	0.45	4.16	18.03**	9.80*	1.70	0.01	0.81	2.53	2.46
0.54	1.59	6.21	0.45	8.37	1.71	1.51	0.74	0.01	3.94
42.22**	30.67*	0.01	196.57**	7.88	2.88	19.47*	8.47	4.67	0.61
25.86**	11.28*	0.05	47.80**	0.24	0.15	5.00	6.46	4.33	0.41
8.22*	20.92*	14.47*	0.01	144.76**	35.64	22.31*	4.80	0.24	26.45*
3.67	0.64	0.69	7.74	1.60	0.13	0.12	0.74	1.40	0.25
10.14*	1.73	3.02	60.22**	39.94**	6.23	0.09	1.24	1.90	3.92
16.02*	7.24	0.89	117.30**	172.17**	66.66	2.38	2.76	1.97	1.00
5.39*	0.79	2.83	12.07**	3.72	0.54	0.12	1.35	5.77*	0.82
	3.40	15.99**	12.08*	58.22**	15.25*	5.52	0.76	0.31	18.76**
		5.28	33.03**	46.92**	9.23	0.66	0.13	0.98	7.42
			22.75**	0.34	0.03	2.24	5.19	6.42*	0.11
				247.34**	75.53*	38.08**	7.60	0.34	46.06**
					16.62	32.35*	12.41*	5.68	2.44
						4.47	2.54	1.55	0.01
							0.54	1.03	2.93
								0.57	4.03
									3.01

〔注記〕　** は１％有意
　　* は５％有意

附録第2表—附表1　　分　散　分　析　表

1と2

変　動　因	自由度	平　方　和	平均平方和	F
問題群間	1	1337.34	1337.34	6.88
問題群内	13	2527.12	194.39	
全　　体	14	3864.45		

1と4

変　動　因	自由度	平　方　和	平均平方和	F
問題群間	1	3331.64	3331.64	17.60
問題群内	10	1893.07	189.31	
全　　体	11	5224.71		

1と5(1)

変　動　因	自由度	平　方　和	平均平方和	F
問題群間	1	3331.64	3331.64	15.91
問題群内	10	2093.07	209.31	
全　　体	11	5424.71		

1と5(2)

変　動　因	自由度	平　方　和	平均平方和	F
問題群間	1	1705.60	1705.60	7.59
問題群内	10	2246.85	224.68	
全　　体	11	3952.45		

1と7

変　動　因	自由度	平　方　和	平均平方和	F
問題群間	1	1054.53	1054.53	5.92
問題群内	11	1959.74	178.16	
全　　体	12	3014.26		

1と8

変動因	自由度	平方和	平均平方和	F
問題群間	1	1307.60	1307.60	6.91
問題群内	10	1893.07	189.31	
全体	11	3200.67		

1と9

変動因	自由度	平方和	平均平方和	F
問題群間	1	3711.96	3711.96	12.96
問題群内	25	7159.30	286.37	
全体	26	10871.26		

1と12

変動因	自由度	平方和	平均平方和	F
問題群間	1	4887.57	4887.57	19.82
問題群内	12	2959.74	246.64	
全体	13	7847.31		

1と14(1)

変動因	自由度	平方和	平均平方和	F
問題群間	1	4110.19	4110.19	21.65
問題群内	10	1898.51	189.85	
全体	11	6008.70		

1と14(2)

変動因	自由度	平方和	平均平方和	F
問題群間	1	1313.42	1313.42	6.24
問題群内	9	1893.07	210.34	
全体	10	3206.49		

1と18

変動因	自由度	平方和	平均平方和	F
問題群間	1	2203.42	2203.42	10.84
問題群内	10	2032.51	203.25	
全体	11	4235.93		

2と4

変動因	自由度	平方和	平均平方和	F
問題群間	1	870.15	870.15	6.86
問題群内	5	634.05	126.81	
全体	6	1504.20		

2と5(2)

変動因	自由度	平方和	平均平方和	F
問題群間	1	1653.37	1653.37	11.98
問題群内	5	690.23	138.05	
全体	6	2343.60		

2と10

変動因	自由度	平方和	平均平方和	F
問題群間	1	589.82	589.82	5.38
問題群内	8	876.94	109.62	
全体	9	1466.76		

2と13

変動因	自由度	平方和	平均平方和	F
問題群間	1	2060.89	2060.89	18.03
問題群内	6	685.97	114.33	
全体	7	2746.86		

2と14(1)

変動因	自由度	平方和	平均平方和	F
問題群間	1	1254.20	1254.20	9.80
問題群内	5	639.49	127.90	
全体	6	1893.69		

4と5(2)

変動因	自由度	平方和	平均平方和	F
問題群間	1	3214.89	3214.89	115.27
問題群内	2	55.78	27.89	
全体	3	3270.67		

4と7

変動因	自由度	平方和	平均平方和	F
問題群間	1	561.60	561.60	23.26
問題群内	3	72.45	24.15	
全体	4	634.05		

4と8

変動因	自由度	平方和	平均平方和	F
問題群間	1	224.99	224.99	77.85
問題群内	2	5.78	2.89	
全体	3	230.77		

4と10

変動因	自由度	平方和	平均平方和	F
問題群間	1	2099.94	2099.94	42.22
問題群内	5	248.67	49.73	
全体	6	2348.61		

4と11

変動因	自由度	平方和	平均平方和	F
問題群間	1	963.33	963.33	30.67
問題群内	3	94.23	31.41	
全体	4	1057.56		

4と13

変動因	自由度	平方和	平均平方和	F
問題群間	1	3781.14	3781.14	196.57
問題群内	3	57.71	19.24	
全体	4	3838.85		

4と15

変動因	自由度	平方和	平均平方和	F
問題群間	1	542.89	542.89	19.47
問題群内	2	55.78	27.89	
全体	3	598.67		

5(1)と5(2)

変動因	自由度	平方和	平均平方和	F
問題群間	1	3410.56	3410.56	27.28
問題群内	2	250.00	125.00	
全体	3	3660.56		

5(1)と10

変動因	自由度	平方和	平均平方和	F
問題群間	1	2290.29	2290.29	25.86
問題群内	5	442.89	88.58	
全体	6	2733.18		

5(1)と11

変動因	自由度	平方和	平均平方和	F
問題群間	1	1082.40	1082.40	11.28
問題群内	3	288.45	96.15	
全体	4	1370.85		

5(1)と13

変動因	自由度	平方和	平均平方和	F
問題群間	1	4013.63	4013.63	47.80
問題群内	3	251.93	83.98	
全体	4	4265.56		

5(2)と7

変動因	自由度	平方和	平均平方和	F
問題群間	1	1475.61	1475.61	37.94
問題群内	3	116.67	38.89	
全体	4	1592.28		

5(2)と8

変動因	自由度	平方和	平均平方和	F
問題群間	1	1738.89	1738.89	69.56
問題群内	2	50.00	25.00	
全体	3	1788.89		

5(2)と9

変動因	自由度	平方和	平均平方和	F
問題群間	1	2580.00	2580.00	8.25
問題群内	17	5316.24	312.72	
全体	18	7896.24		

5(2)と10

変動因	自由度	平方和	平均平方和	F
問題群間	1	481.56	481.56	8.22
問題群内	5	292.89	58.58	
全体	6	774.45		

5(2)と11

変動因	自由度	平方和	平均平方和	F
問題群間	1	965.60	965.60	20.92
問題群内	3	138.45	46.15	
全体	4	1104.05		

5(2)と12

変動因	自由度	平方和	平均平方和	F
問題群間	1	4040.67	4040.67	14.47
問題群内	4	1116.67	279.17	
全体	5	5157.34		

5(2)と14(1)

変動因	自由度	平方和	平均平方和	F
問題群間	1	4013.22	4013.22	144.76
問題群内	2	55.44	27.72	
全体	3	4068.66		

5(2)と15

変動因	自由度	平方和	平均平方和	F
問題群間	1	1115.56	1115.56	22.31
問題群内	2	100.00	50.00	
全体	3	1215.56		

5(2) と 18

変動因	自由度	平方和	平均平方和	F
問題群間	1	2505.00	2505.00	26.45
問題群内	2	189.45	94.73	
全体	3	2694.45		

7 と 10

変動因	自由度	平方和	平均平方和	F
問題群間	1	523.34	523.34	10.14
問題群内	6	309.56	51.59	
全体	7	832.90		

7 と 13

変動因	自由度	平方和	平均平方和	F
問題群間	1	1785.38	1785.38	60.22
問題群内	4	118.59	29.65	
全体	5	1903.97		

7 と 14(1)

変動因	自由度	平方和	平均平方和	F
問題群間	1	959.94	959.94	39.94
問題群内	3	72.11	24.04	
全体	4	1032.05		

8 と 10

変動因	自由度	平方和	平均平方和	F
問題群間	1	778.22	778.22	16.02
問題群内	5	242.89	48.58	
全体	6	1021.11		

8と13

変動因	自由度	平方和	平均平方和	F
問題群間	1	2030.34	2030.34	117.30
問題群内	3	51.93	17.31	
全体	4	2082.27		

8と14(1)

変動因	自由度	平方和	平均平方和	F
問題群間	1	468.72	468.72	172.17
問題群内	2	5.44	2.72	
全体	3	474.16		

9と10

変動因	自由度	平方和	平均平方和	F
問題群間	1	1485.86	1485.86	5.39
問題群内	20	5509.13	275.46	
全体	21	6994.99		

9と13

変動因	自由度	平方和	平均平方和	F
問題群間	1	3567.59	3567.59	12.07
問題群内	18	5318.16	295.45	
全体	19	8885.75		

9と17

変動因	自由度	平方和	平均平方和	F
問題群間	1	2381.70	2381.70	5.77
問題群内	19	7849.13	413.11	
全体	20	10230.83		

10 と 12

変動因	自由度	平方和	平均平方和	F
問題群間	1	2991.46	2991.46	15.99
問題群内	7	1309.56	187.08	
全体	8	4301.02		

10 と 13

変動因	自由度	平方和	平均平方和	F
問題群間	1	593.63	593.63	12.08
問題群内	6	294.82	49.14	
全体	7	888.45		

10 と 14 (1)

変動因	自由度	平方和	平均平方和	F
問題群間	1	2891.57	2891.57	58.22
問題群内	5	248.34	49.67	
全体	6	3139.91		

10 と 14 (2)

変動因	自由度	平方和	平均平方和	F
問題群間	1	926.30	926.30	15.25
問題群内	4	242.89	60.72	
全体	5	1169.19		

10 と 18

変動因	自由度	平方和	平均平方和	F
問題群間	1	1434.65	1434.65	18.76
問題群内	5	382.34	76.47	
全体	6	1816.99		

11 と 13

変動因	自由度	平方和	平均平方和	F
問題群間	1	1159.26	1159.26	33.03
問題群内	4	140.37	35.09	
全体	5	1299.63		

11 と 14 (1)

変動因	自由度	平方和	平均平方和	F
問題群間	1	1468.60	1468.60	46.92
問題群内	3	93.89	31.30	
全体	4	1562.49		

12 と 13

変動因	自由度	平方和	平均平方和	F
問題群間	1	5088.74	5088.74	22.75
問題群内	5	1118.60	223.72	
全体	6	6207.34		

12 と 17

変動因	自由度	平方和	平均平方和	F
問題群間	1	3907.28	3907.28	6.42
問題群内	6	3649.56	608.26	
全体	7	7556.84		

13 と 14 (1)

変動因	自由度	平方和	平均平方和	F
問題群間	1	4730.10	4730.10	247.34
問題群内	3	57.37	19.12	
全体	4	4787.47		

13 と 14 (2)

変動因	自由度	平方和	平均平方和	F
問題群間	1	1960.96	1960.96	75.53
問題群内	2	51.93	25.96	
全体	3	2012.89		

13 と 15

変動因	自由度	平方和	平均平方和	F
問題群間	1	1293.63	1293.63	38.08
問題群内	3	101.93	33.98	
全体	4	1395.56		

13 と 18

変動因	自由度	平方和	平均平方和	F
問題群間	1	2938.32	2938.32	46.06
問題群内	3	191.37	63.79	
全体	4	3129.69		

14 (1) と 15

変動因	自由度	平方和	平均平方和	F
問題群間	1	897.00	897.00	32.35
問題群内	2	55.44	27.72	
全体	3	952.44		

14 (1) と 16

変動因	自由度	平方和	平均平方和	F
問題群間	1	1761.80	1761.80	12.41
問題群内	3	425.89	141.96	
全体	4	2187.69		

14 (1) (2) と 15

変動因	自由度	平方和	平均平方和	F
問題群間	1	1421.41	1421.41	23.13
問題群内	3	184.37	61.46	
全体	4	1605.78		

附録第3表　〔[計算機プログラム] に対する Correlation Matrix〕

学科目	理科	経済学概論	統計	商学概論	簿記
(1)	−0.00661	0.27645	0.18372	0.03921	0.16552
(2)	−0.06025	0.42737	0.24486	−0.05833	0.19343
学科目	数学	管理科学概論	文学	経済史概論	経済原論
(1)	0.26840	−0.24522	0.19852	−0.18864	−0.03889
(2)	0.29008	−0.09839	0.44212	−0.19643	0.15840

附録第3表—附表1　〔Regression Analysis〕

$Y={}_1X+B_2X+B_3X+B_4X+B_5X+A$

科目		理科	経済学概論	統計	商学概論	簿記
R. A.	A	B_1	B_2	B_3	B_4	B_5
(1)	20.05924	0.46671	0.27964	−0.29162	0.18153	−0.0254
(2)	50.30034	0.31802	−0.06085	−0.15545	0.15894	−0.00127

附録第3表—附表2　〔Regression Analysis〕

$Y=BX+A$

科目		数学	管理科学概論	文学	経済史概論	経済原論
(1)	A	39.47389	75.07915	40.24224	76.57908	63.79461
	B	0.30205	−0.16616	0.29632	−0.19344	−0.01535
(2)	A	54.39233	71.41799	38.29722	76.11794	67.13233
	B	0.18153	−0.03688	0.38020	−0.11646	0.04281

［注］ (1)は Sample 受験者全員について
(2)は Sample 受験者のうち平均正答率 60.0 %以上の者について

附録第4表　〔スピアマンの順位相関係数〕

科目	理科	経済学概論	統計	商学概論	簿記
(1)	0.16174	0.18354	−0.22492	0.33838	0.26897
(2)	0.17899	−0.17108	−0.24224	0.33371	0.10785
科目	数学	管理科学概論	文学	経済史概論	経済原論
(1)	0.47838	−0.18385	0.27826	0.02019	0.102354
(2)	0.44662	−0.07018	0.43653	0.02987	0.06767

［注］ (1)は Sample 受験者全員について
(2)は Sample 受験者のうち平均正答率 60.0 %以上の者について

附録——試験問題例

［1］ Arithmetic Assignment Statement

7-2 次の算術式の中で正しければ□の中に○を，正しくないものは×を記入し，正しくない理由を後に述べよ。

(1)□ $X+Y=A+B$ ＿＿＿＿＿＿＿＿＿＿

(2)□ $Z=A**3$ ＿＿＿＿＿＿＿＿＿＿

(3)□ $P=I*N**3$ ＿＿＿＿＿＿＿＿＿＿

(4)□ $J=K**A+17$ ＿＿＿＿＿＿＿＿＿＿

［2］ GŌ TŌ statement

9-1 KARE が1なら statement number が50の statement へ，2なら30へ，3なら20へ，4なら10へとぶには

とかく。

10-1 次の計算結果による GŌ TO statement が正しければ Y，誤りなら N を空白に入れよ。

(1)□ GŌ TŌ (31, 32, 33), M+N

(2)□ GŌ TŌ (31, 32, 33), 2

(3)□ GŌ TŌ (31, 32, 33), J(I)

(4)□ GŌ TŌ (31, 32, 33), M

［3］ DŌ statement

4-4 $N=0$

DŌ 10 $I=1, 10$

10 $N=N+5$

を実行したときの N の値は |＿＿| となる。

5-2 次のプログラムに文法上の誤りがあれば正せ。論理的な内容は問題にしていない。

16⊿DŌ 20 $I=1, K$

$K=I+K$

$IF(J. GT. K)$ GŌ TŌ 20

$S=S+I$

20⊿CŌNTINUE

[4] declaration, 配列に関するもの

4-5 X, Y, Z を整数型変数として宣言するには

|＿＿＿＿| X, |＿＿| Z

とかく。

5-1 DIMENSIŌN Q (4, 2) と宣言すればどのような順序で記憶場所が確保されるか。

[5(1)] formatted READ statement

4-1 A と B の値をカードからよみこむには

READ (|＿＿＿|) A, B

100⊿ |＿＿＿＿＿＿| (2F 12.0)

[5(2)] formatted WRITE statement

8-5 A=200.5

WRITE (6, 100) A

100⊿FŌRMAT (1H+, 10X, 9H THE⊿VALUE// 1H+, 5H ŌF⊿A=, F6.1)

という statement を実行したときどの様に印刷されるか，ブランクは b で示せ。

[6] FŌRMAT statement —— I type

6- FŌRTRAN の FŌRMAT statement に関する文章が正しければ Y，誤っていればNを□の中に入れよ。

-25□ 実数型データを I_w で WRITE すれば，整数部分だけ印刷される。

[7] FŌRMAT statement —— F type

6-5□ フィールド記述子が $F_{w \cdot d}$ であるときの出力で，もし指定した欄より

長いデータの時は欄の右側にはみ出した字が切りすてられる。

6-11□ $F_{w \cdot d}$ で $w=d+2$ でも差支えない。

[8] FŌRMATstatement —— E type

6-4□ 入力の FŌRMAT のフィールド記述子が $E_{w \cdot d}$ であるとき，カード上のフィールドに小数点があらわれないなら，指数の左へ d 桁の所に小数点があるものとみなされる。

[9] FŌRMAT statement —— A type

12-1 次の文章の空白に適当な文字又は数字を入れよ。

……。従って英数字フィールド ②□ w を使って，文字をカードからよむ時には $w>$③□ ならば，カード上のデータの ④□ の ⑤□ 文字がよみこまれ，逆に $w<$⑥□ ならば，1語の ⑦□ の ⑧□ 桁が ⑨□ になる。……

[10] FŌRMAT statement —— L type

11-3 FŌRTRAN で論理値をデータとしてよみこませるには，FŌRMAT 中のフィールド記述子を □ とかき，カードに □ 個のフィールドをとりカード上のそのフィールド内の □ の文字が □ 又は □ でありさえすれば，任意にパンチしておけばよい。

[11] FŌRMAT statement —— H type

6-13□ $mHh_1h_2 \ldots h_n$ で $m>n$ でも差支えない。

[12] Record に関するもの

2-5 データは card の 80 欄まで punch することができ □。

6-7□ 出力の 1 記録の最大は 1 行に印刷される 100 字である。

[13] FŌRMAT の反復指定に関するもの

5-4 FŌRMAT (1H1, I4, 2 (E15.7, 5X), 3F 12.3)

をこれと同じ内容をあらわす FŌRMAT に書き改めよ。

[14(1)] SUBRŌUTINE subprogram —— 引数なし

8-1 KEISAN という名前の SUBRŌUTINE subprogram を main program でよび出すには

□

とかく。

[14(2)] SUBRŌUTINE subprogram —— 引数あり

9-7 SUBRŌUTINE SUM (X, Y, Z)
Z=X+Y
RETURN
END
と定義されているとき，この SUBRŌUTINE subprogram を使って (T+2.5)+0.17 を計算して A(2) に格納するには
CALL SUM (□, □, □)
とかけばよい。

[15] FUNCTIŌN subprogram

10-4 関数 $F(x)=1/(1+x^2)$ を定義する FUNCTIŌN subprogram は

とかかれる。

[16] 標準関数

9-2 $\sqrt{2}$ は FŌRTRAN の statement では □ とかかれる。

[17] subscript に関するもの

7-3 次の添字を用いた例のうち正しいものは□の中に○，正しくないものは□の中に×を記入して正しくない理由を後に述べよ。

(1)□ ARRAY (LIST) ＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿

(2)□ A(M*N) ＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿

(3)□ ANSWER (K−25) ＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿

(4)□ BLŌCK (I, J) ＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿

[18] logical statement

11-2 A, B を論理変数として「A implies B」という命題に関する次の真理値表を完成せよ。

B \ A	真	偽
真		
偽		