A2. プログラミング学習システム
CLASS

鍛治　勝三，佐竹　紀男（(財) 日本情報処理開発センター）

1. まえがき

Computer Assisted Instruction （CAI)が世間の注目をあびるようになってからすでに久しいが，現在まだその実用効果の確証が得られていないというのが実情であろう．特に，価格に見合う成果という点ではかなり悲観的な見方も多い．

この様な実情の中で，比較的その効果を評価されつつあるのがコンピュータのプログラミングの学習であるという．これは,その目的と道具が一致したという強味と同時に非常に問題が大きいといわれる学習プログラムそのものに専門的知識の介入が可能であるからであろう．即ち学習の内容を100％理解しているものがシステムを作る強味である．

CLASS(Conversational Language ASisted System) は，これに更に次のような2つの要素をつけ加えてシステムを設計した．

第一は，TSSの会話型言語CPL[1] (Conversational PL/I) の学習をおこなうシステムであるため，プログラミング言語の学習に不可欠な練習問題として作成されたプログラムを実際にCPLのプロセッサで処理して，エラーの指摘や結果の表示をおこなわせることができる．

第二は，方式の比較という目的から2つのシステムを作成したことである．一つは，CRTキャラクタ・デイスプレイ装置とタイプライタ装置（TSS用端末）を学習端末とし，CPLによるプログラムの練習問題を中心に学習者の作成したプログラムの正否および誤りの指摘をし，また，CPLシステムそのものへのガイダンスだけでなく，ディバック方法のガイダンスの要素も含めたユニークなものである．他の一つは，標準CAI端末（スライド・プロジェクタ，キイ・セット，タイプライタ）を学習端末としたオーソドックスなものであり，前者と同様に各セクションのまとめとして，併用されているTSS用端末からCPLの実習を実際におこなうことができる．この2つのシステムは，それぞれFACOM 230−60, NEAC 2200/500を用いて開発したが，対象とするコンピュータの名称から各々CLASS−FシステムおよびCLASS−Nシステムと呼んで区別している．

本論文では，CLASS−Fシステムについてのみ報告する．

2. システムの概要

現在開発されているプログラム教育用のCAIシステムは，プログラム学習の方式にもとずいて作成され，プログラムの個々のステートメントのみを対象にして，対話的に学習させているものが多い．

コンピュータのプログラミング教育においては，この種のCAIシステムでは不十分であり，学習者自身が自分でプログラムを組み，直接コンピュータにかけ，その結果に対して適切な診断（エラー・メッセージ）や治療が行なわれることが望ましい．

CLASS−Fシステムは，FACOM 230−60タイム・シェアリング・システムの元で使用される会話型言語CPLの学習をおこなうシステムであり，学習自身もキャラクタ・ディスプレイ装置を使用して会話型式でおこなうことができる．すなわち，会話型言語CPLによるプログラムの作成，実行あるいは修正の各段階において，十分なガイダンス機能を与えると共に，シンタックスの誤りの指摘は当然のこと，ロジカルな誤りに対しても診断をおこな

い学習者のディバック技術も併せて習得させる．

一般に，学習者の作成したプログラムは，バラエティに富み，学習者の犯す誤りも千差万別であることから，考えられるすべての誤りに対して適切な診断や治療を与えることはきわめて難かしいことである．

しかし，ある問題に対するプログラムを学習者に自由に作らせ，そのプログラムを分析して統計をとってみると，アルゴリズムのパターンはある程度限られており，そのアルゴリズムの誤りやすい点がわかる．

そこで，本システムでは，提示された問題について学習者に自由にプログラミングさせ，そのプログラムの文法エラー（シンタックス・エラー）を即座に指摘し，すぐ訂正させるとともに，実際のデータを使ってそのプログラムを実行させ，実行結果を診断する．この場合，最終結果だけでなく中間結果についてもシステム側でチエックし，どのあたりに誤りがあるかの見当をつけ，学習者との対話によって誤りを発見させるという方法を採用した．

プログラム診断の学習アルゴリズムは次の通りである．

(1) まず，学習者は氏名，コードと学習すべき問題の名称を入力する．システムは，指定された問題の本文をディスプレイ装置に表示する．

(2) 指定された問題に対するあらかじめ用意されている標準プログラムをＣＰＬプロセッサでコンパイルし，標準データ（チエック・データ）を使って実行させ，その実行結果を保存する．

(3) 学習者は，その問題に対するプログラムをディスプレイ装置を経由してＣＰＬプロセッサと会話しながら作成する（システムは，学習経過記録（logging）を調べすでに学習者が途中まで作成したプログラムがあればローディングする）．

このとき，プログラムとして入力された各ステートメントに文法エラーがあればすぐさま指摘され，直ちに修正することができる．またプログラムの作成の途中でステートメントやコマンドの使い方について疑問が生じた場合は随時，コンピュータに質問することができる．また，学習時間がなくなった場合は，途中で学習を中断することができる．

(4) 学習者のプログラムが完成（未完成でもよい）すると，それを標準データを使って実行させる．その実行結果と前もって実行して得られている標準プログラムの実行結果とを診断プログラムの手順に従ってくらべ，評価し，適切な診断メッセージをディスプレイ装置に表示する．

(5) 正しければ，ここで学習を終了するか，又は次の新しい問題を学習するため(1)に戻る．誤りならば，(2)からやり直す．何回も誤りをくり返している場合は，その都度適切なメッセージやヒントを与えるが，場合によっては，その問題に対する正答プログラムをディスプレイ装置に表示して学習を打切ることもある．

これらの処理過程中，常時，学習者の入力，システムからの重要なメッセージは学習経過記録ファイルに記録（log）される．本システムは，ＦＡＣＯＭ　２３０－６０モニタＶのＴＳＳ管理の元で働く会話型処理プログラムとして作成されている．

3. 学習プログラム

3.1 学習プログラムとは

学習者にプログラミングの問題を提示して自由に作らせたプログラムを自動診断するには，同じ問題を解く標準プログラムと同じデータを使って実行させ，両者の実行結果を比較するのが最も容易な方式の一つである．

標準プログラムには，その問題に対する正しいプログラムだけでなく，学習者がプログラム作成上まちがいやすいと思われる所を診断するための情報として，それらの箇所が実際にまちがっているプログラムもあわせて入れて

おく．このプログラムをサンプル・プログラムと呼ぶ．

両者のプログラムを実行するためのデータは，プログラムの診断に適したチエック・データが用いられる．このデータをサンプル・データと呼ぶ．

さらに，学習者のプログラムとサンプル・プログラムの実行結果の比較（診断）手続きを表わすプログラムが必要である．このプログラムは，診断プログラムと呼ばれ，ＤＩＡＬ言語（DIAgnostic Language，後述）でコーディングされている．

以上をまとめると学習プログラムは，次の４つの部分からなっている．

(1) 問題集

(2) サンプル・プログラム

(3) サンプル・データ

(4) 診断プログラム

3.2 診断方法

プログラムの自動診断を行なう１つの方法として，ＣＰＬ言語で記述された学習者のプログラムとすでにシステムで準備してあるサンプル・プログラムを同じサンプル・データで実行し，両者の実行結果を比較する．

実行結果の比較を容易にするためにまず，両方のプログラムで使用する主な変数の名前の付け方を統一しておく．ＣＰＬ言語では変数に対してバラエティに富んだ属性を与えることができるため変数の属性の比較をするが，

たとえば，DECIMALとＢＩＮＡＲＹの内どちらの属性を指定してもよい場合があるので，必ずしもすべての属性が一致する必要はない．

次に変数の内容の比較を行うとき，一致している場合は問題がないが，一致しない場合は，なぜ一致しなかったかの診断を行う必要がある．サンプル・プログラムの中には，誤りやすいと思われる部分（アルゴリズムの誤り）についてわざと間違ったコーディングを情報として含めているので，その中のどの部分と一致するかを調べてコーディング（アルゴリズム）の誤りを指摘することができる．

たとえば，プログラム・ループで回数のカウントが１だけ多かったり，少なかったりすることがよくある．学習者のプログラムとサンプル・プログラムの中で用いられている．

回数のカウント用の変数の内容を比較して１だけ異なる場合は，初期値の誤り，回数のカウントの終りの判定位置の誤り等がプログラム・ミスとして考えられるのでそれらを診断結果として学習者に知らせる．

3.3 学習プログラムの改良

学習プログラムの作成者は，学習プログラムを作成する前に問題に対して十分な分析を行なって作成するが，やはり不十分なことが多い．作成者はときどき，自分の作った学習プログラムに問題点がないかどうかを学習経過記録を評価し，学習者にとってどこがむずかしいのか，学習プログラムの間違いはどこかを発見し，学習プログラムを改良しなければならない（システム側では，常に，学習者の経過記録だけではなく，システムの評価，改良に必要な情報も収集しておかなければならない）．

作成者は，常に「診断されているのは学習者プログラムではなく，学習プログラムなのである．正しく診断できないのは，学習者の間違いではなく，学習プログラムの欠点なのだ．もし，学習者が正しくプログラムを修正することができなくなれば，学習プログラムを改良するつもりだ」という心がまえが必要である．

すなわち，こういったシステムは，自己増殖的（自己修正的）でなければいけないのである．

3.4 ＤＩＡＬ言語

ＤＩＡＬは診断プログラムを記述する言語であり，一般のＣＡＩのＣＡＩ言語に相当する．

ＤＩＡＬはＰＬ／Ｉとほぼ同じ仕様と機能を持っているが，このＣＬＡＳＳ－Ｆシステムでプログラム診断を行うために便利な機能が若干つけ加えられている．特にＣＰＬプロセッサの中の記号表を直接参照したり，変数の属性の宣言の正否を診断したりする特殊目的があるため，変数名のつけ方に工夫をしている．

たとえば，診断すべき属性については，その属性の直前に“？”記号を書くことにより，システムが自動的に診断し，適切な診断メッセージを出す．

〔例〕 ＤＥＣＬＡＲＥ ＠Ａ　　？ＦＩＸＥＤ（？１０）；

この例では，変数＠Ａの属性が固定小数点であるか，その桁数が１０桁かどうかを診断すべきことを宣言している．

変数としては４種類あり，次のように区別している．すなわち，学習者のプログラム中で使われている変数を参照するための変数には先頭に“＠”記号をつける（スチューデント変数と呼ぶ）．サンプル・プログラム中で使われている変数を参照するための変数には先頭に“＃”記号をつける（サンプル変数と呼ぶ）．診断プログラム中で特殊な機能をもつ変数には，先頭に“￥”記号を付ける（システム変数と呼ぶ）．診断プログラム中だけで有効な変数には先頭に特殊文字を付けないことで区別している（ローカル変数と呼ぶ）．

たとえば，学習者のプログラム中の変数ＡＭＩＮに正しい結果が求まっているかどうかを診断するためには，サンプル・プログラム中の変数ＡＭＩＮに正しい結果が入っていることから，次のステートメントで診断することができる．

ＩＦ ＠ＡＭＩＮ ＝ ＃ＡＭＩＮ　ＴＨＥＮ ……；　　ＥＬＳＥ……；

4. システム構成

システムは，大別して次の９つの部分から構成されている．

(1) システム全体を管理する制御プログラム …… モニタ

モニタは，システムの初期設定を行ない，学習者の入力したシステム制御コマンド（後述）と内部的に発生したシステム制御コマンドに従ったコマンド処理プログラムを主記憶装置にローディングし，制御権を渡し，各コマンド処理プログラムの状態（モード）に遷移する．

(2) ＣＰＬ言語で書かれたプログラムを会話的に実行するプロセッサ …… ＣＰＬプロセッサ

(3) 学習プログラム

(4) 診断プログラムをコンパイルするコンパイラ …… ＤＩＡＬコンパイラ

(5) 診断プログラムを実行させるインタプリタ …… ＣＡＬＬ

ＣＡＬＬはＣＡpable Ｌinkage Ｌoaderの略で，コンパイルされた診断プログラムを主記憶装置にローディングし，診断プログラム，学習者の作成したプログラムとサンプル・プログラムの３者を結合し，診断プログラムの手続に従ってプログラムの診断を行なう．

(6) 学習者とコンピュータとが対話するために使用される学習端末…… キャラクタ・ディスプレイ装置と補助的なタイプライタ装置

本システムの主要な学習端末はキャラクタ・ディスプレイ装置である．キャラクタ・ディスプレイ装置はタイプライタ装置とくらべて騒音もなく，表示速度も非常に早く，特に入力の修正や追加には威力を発揮する．しかし，文字がブラウン管上に表示されるので，ハード・コピー（印刷物）がとれない．この欠点を補うため，本システムでは学習者が要求すれば，いつでもプログラムのリストを補助タイプライタ装置にタイプ・アウトすることができる．

(7) 学習端末を管理するプログラム …… SCROLL

本システムでは，キャラクタ・ディスプレイ装置の画面を2つに分割（スプリット・スクリーン）して，上側の画面（50字×15行）には，常時，学習者の作成したプログラムを表示し，下側の画面（50字×4行）には，学習者の入力，システムからの診断メッセージ（エラー・メッセージ）を出力するために使っている．学習者の作成したプログラムを全部一度には表示することができないので，学習者の希望するプログラムの任意の部分を自由に表示する機能を有する画面制御を行なっているが，そのプログラムがSCROLL である．

(8) 学習経過記録 …… LOG

学習者の入力したメッセージやシステムからの重要なメッセージ（診断メッセージ等）を自動的に学習経過記録ファイルに記録する．又この記録をもとにして学習の中断，再開の処理をも行なう．

学習経過記録は後で学習過程や学習プログラムを評価するためバッチ処理で分析が行なわれる．

(9) ユーテイリティ・プログラム …… LIBE

LIBE はLIBrary Editor の略で，学習プログラムをシステムに登録する場合に使われるプログラムである．

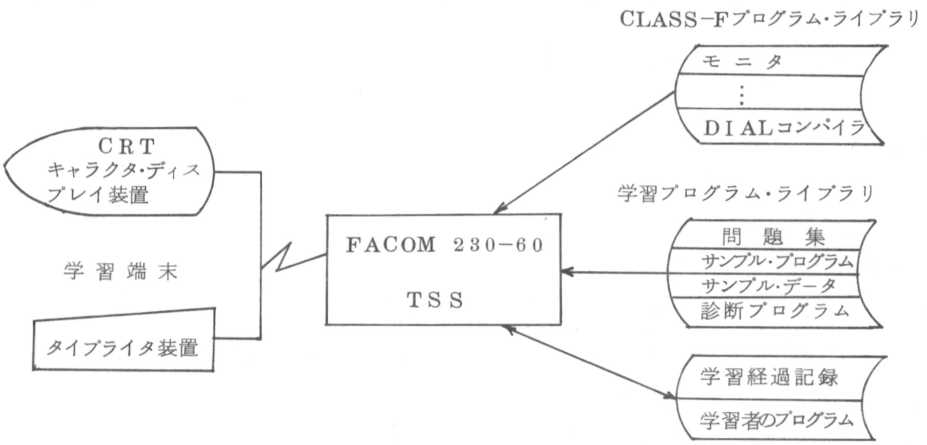

図1 CLASS−F システム構成図

5. 処理モード

本システムでは，処理モードという概念を導入してシステムを管理している．

処理モードには次の8種類あり，処理モードの遷移はシステム制御コマンドを使って指令する．

(1) OFF モード

OFF モードとは，システムが主記憶装置に存在しない状態のことである．この状態でシステム制御コマンド¥CLASSFを入力するとFACOM230−60モニタVによってCLASS−Fシステムのモニタが主記憶装置にロ

ーディングされ，自動的に初期モードに遷移する．

(2) 初期モード

初期モードとは，システムの初期化を行なっている状態である．ここでは，学習者に名前，コード，問題名を問合わせる．初期化が終了すると次のサンプル・モードへ自動的に遷移する．

(3) サンプル・モード

サンプル・モードとは，指定された問題に対するサンプル・プログラムをローディングし，ＣＰＬプロセッサでコンパイルし，実行し，実行結果を主記憶装置の一部に保存する処理過程である．このモードが終了すると学習モードへ自動的に遷移する．

(4) 学習モード

学習モードとは，提示された問題に対して学習者がディスプレイ装置を経由してＣＰＬプロセッサと会話しながらＣＰＬプログラムを作成している状態である．

学習者はプログラムの作成が終ると(未完成でもよい)．このプログラムをシステムに診断してもらうため，システム制御コマンド¥ＲＵＮを入力する．このコマンドが入力されると次の診断モードに遷移する．又，ここで学習を中断したい場合は，システム制御コマンド¥ＢＹＥを入力する．このコマンドが入力されるとＯＦＦモードに遷移する．

又，学習の途中で疑問が生じたならば，システム制御コマンド¥Ｑ入力すればよい．

(5) 診断モード

診断モードとは，学習者の作成したプログラムの実行結果とサンプル・プログラムの実行結果を診断プログラムの手続きに従って診断している状態である．診断結果が重傷であるとプログラムを修正させるため自動的にサンプル・モードへ遷移する．正しい場合は自動的に終了モードへ遷移する．

(6) 終了モード

終了モードとは，学習者と会話して，次の問題をやるかどうか問合せたり，システムの後始末をやっている状態である．

学習者は，次の問題をやりたければシステム制御コマンド¥ＣＬＡＳＳＦ，ここで完全に終了しなければシステム制御コマンド¥ＢＹＥをそれぞれ入力する．

(7) 問合モード

問合モードとは，学習中，学習者がＣＰＬの文法等について疑問が生じた場合，それに対して会話的に応答する処理過程である．

これらのモードおよびシステム制御コマンドの関連を図２に示してある．

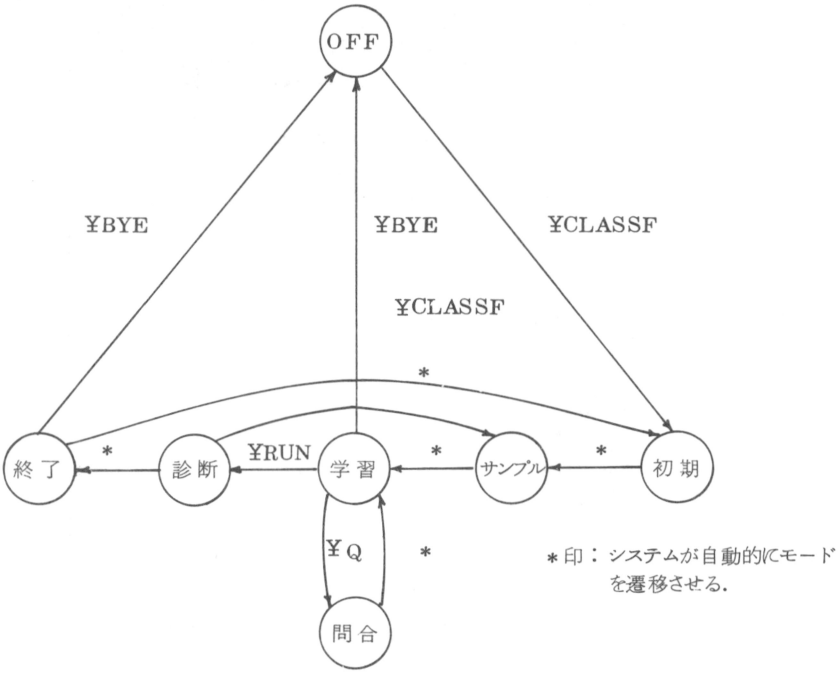

図2　処理モードとコマンドの関連図（状態遷移図）

6. 実　例

データを読み込んで小さい順に並べる問題に対する学習プログラムの例を図3，図4，図5に示す.

参考文献

1）川島，佐竹，山本　PL／I会話型言語プロセッサCPL　（第11回　情報処理学会大会講演予稿集）

```
**********   10ケタ ノ モジ゛ デ゛ータ オ タンマツ カラ ヨミコミ ナカ゛ラ
*ASNDSORT*  コレラ オ インサツ シ, カツ, ワーク エリア ニ チイサイシ゛ュン
**********  (ASCENDING ORDER) ニ ナラヘ゛, ソノ ケツカ ト
            デ゛ータ ノ コスウ オ インサツ スル フ゜ログ゛ラム オ ツクレ.
  ココニ, デ゛ータ ノ オワリ トシテ '**********' オ ニュ-リョク スル モノ
ト スル.
 タタ゛シ, デ゛ータ ノ コスウ ハ 20コ イカ トシ, チイサイシ゛ュン ニ ナラヘ゛タ
ケツカ オ イレル ワーク エリア ノ ハイレツメイ オ 'WORK', デ゛ータ ノ コスウ オ
カウント スル エリア ノ ヘンスウメイ オ 'N' トセヨ.
 デ゛ータ ノ ニュ-リョク ハ ツキ゛ ノ カタチ トスル.
   GET LIST(デ゛ータ_メイ) ;
                                                          A-1
```

```
***  デ゛ータ ノ レイ  ***

WADA
ISHII
SATO
TANAKA
SHIRAKAWA
TAKASU
NAKAZAWA
MIYAZAWA
KUSAKARI
TAKADA
NAGANUMA
**********
                                                          A-2
```

```
          ***  キオク エリア  ***

     N                WORK
 +-----------+     +-----------+
 I         3 I    1IISHII      I
 +-----------+     +-----------+
                   2ISATO       I   /* ヘンスウ ノ ナカミ ハ
                    +-----------+      3コメ ノ デ゛ータ オ
                   3IWADA       I      ショリ シ オワッタ トコロ
                    +-----------+      ノ シ゛ヨウタイ デアル.
                   ::           :                          */
                   ::           :
                    +-----------+
                  20I           I
                    +-----------+                          A-3
```

図3　提示問題の例

```
0001.00 ASNDSØRT: PRØCEDURE ØPTIØNS(MAIN);
0002.00 DECLARE NAMAE CHARACTER(10),
                WØRK(20) CHARACTER(10),
                N ;
0003.00 DECLARE SEQ(20) CHAR(10); /* FØR DIAGNØSTIC */
0004.00 N=0;
0005.00 LØØP: GET LIST(NAMAE) ;
0006.00 IF NAMAE = '**********'
          THEN GØ TØ FINISH;/* DATA END ? */
0007.00 IF N = 0 THEN GØ TØ FIRST ;
0008.00 DØ I = 1 TØ N ;  /*  ASCENDING SØRT  */
0009.00 II = N-I+1 ;
0010.00 IF NAMAE >= WØRK(II) THEN GØ TØ INSERT ;
0011.00 WØRK(II+1) = WØRK(II) ;
0012.00 END ;
0013.00 FIRST: II = 0;
0014.00 INSERT: WØRK(II+1) = NAMAE ;
0015.00 N = N+1;
0016.00 SEQ(N) = NAMAE ;   /*  FØR DIAGNØSTIC  */
0017.00 GØ TØ LØØP ;
0018.00 FINISH: END ASNDSØRT ;
```

図4　サンプル・プログラムの例

```
1       ASNDSORT:PROCEDURE OPTIONS(MAIN) ;
2       /*
          ** THE DIAGNOSTIC PROGRAM FOR
                THE STUDENT'S ASCENDING SORT PROGRAM **

          ** THE RULE FOR IDENTIFIERS WITH FIRST LETTER **
                @ ..... THE STUDENT VARIABLE
                # ..... THE SAMPLE VARIABLE
                ¥ ..... THE SYSTEM VARIABLE
                ELSE .. THE LOCAL VARIABLE
          ** THE MARK '?' IS THE DIAGNOSTIC ATTRIBUTE. **
                                                          */

         DECLARE @N ?FIXED(10,?0)
                @WORK?(?20) ?CHARACTER(?10) ;
3        DECLARE #N, #WORK(20) CHAR(10),
                #SE@(20) CHAR(10) ;
4        DCL SCORE FIXED, MSGSTK(5) ;
5        DECLARE MSGBLK(12) CHARCTER(50) INIT(
                'ﾃﾞｰﾀ ｶﾞ ﾀﾀﾞｼｸ ﾌﾞﾝﾙｲ ｻﾚﾃｲﾏｽ。                          ',  /*  1 */
                'ﾃﾞｰﾀ ｶﾞ ｵｵｷｲ ｼﾞｭﾝ ﾆ ﾅﾗﾝﾃﾞｲﾙ。                        ',  /*  2 */
                'ﾃﾞｰﾀ ﾉ ﾆｭｰﾘｮｸ ｼﾞｭﾝ ﾆ ﾅﾗﾝﾃﾞｲﾙ。                      ',  /*  3 */
                'ﾃﾞｰﾀ ﾉ ﾆｭｰﾘｮｸ ﾉ ｷﾞｬｸﾝｼﾞｭﾝ ﾆ ﾅﾗｲﾃﾞｲﾙ。                ',  /*  4 */
                'ﾃﾞｰﾀ ｵ ﾁｲｻｲ ｼﾞｭﾝ ﾆ ﾅﾗﾍﾞﾅｻｲ。                        ',  /*  5 */
                'ﾃﾞｰﾀ ﾉ ｲﾚｶｴ ﾉ ﾄｺﾛ ｵ ﾁｪｯｸ ｼﾅｻｲ。                      ',  /*  6 */
                'ﾃﾞｰﾀ ﾉ ﾋｶｸ ﾉ ﾄｺﾛ ｵ ﾁｪｯｸ ｼﾅｻｲ。                      ',  /*  7 */
                'ﾆｭｰﾘｮｸ ｻﾚﾀ ﾃﾞｰﾀ ｶﾞ ﾅｸﾅｯﾃｲﾙ。                        ',  /*  8 */
                'ﾆｭｰﾘｮｸ ｻﾚﾀ ﾃﾞｰﾀ ｲｶﾞｲ ﾉ ﾃﾞｰﾀ ｶﾞ ﾊｲｯﾃｲﾙ。              ',  /*  9 */
                'ﾆｭｰﾘｮｸ ｻﾚﾀ ﾃﾞｰﾀ ｶﾞ ｼﾞｭｰﾌｸ ｼﾃｲﾙ。                      ',  /* 10 */
                'ﾓﾝﾀﾞｲ ｵ ﾖｸ ﾖﾐﾅｻｲ。                                  ',  /* 11 */
                'ﾓｼ、ﾜｶﾗﾅｹﾚﾊﾞ ｻｲﾀﾞｲﾁ/ｻｲｼｮｳﾅ ﾓﾝﾀﾞｲ ｵ ｻｸｾｲｼﾅｻｲ。          ');  /* 12 */
6
         SCORE = 0 ;
7       /* THE DIAGNOSTIC FOR TESTING THE END OF DATA */
         IF ¥EOF ¬= 1 THEN GO TO LOOP_NORMAL ;
9                DISPLAY ('ﾆｭｰﾘｮｸ ﾃﾞｰﾀ ﾉ ｵﾜﾘ ﾉ ﾊﾝﾃｲ ｶﾞ ｵｶｼｲ。') ;
10               GO TO FINISH ;
11      LOOP_NORMAL:
                 SCORE = SCORE+10 ;
12               DISPLAY ('ﾆｭｰﾘｮｸ ﾃﾞｰﾀ ﾉ ｵﾜﾘ ﾉ ﾊﾝﾃｲ ｶﾞ ﾀﾀﾞｼｲ ﾃﾞｽ') ;
13
        /* THE DIAGNOSTIC FOR THE VARIABLE 'N' */
         IF @N = #N THEN GO TO  N_NORMAL ;
15               DISPLAY('''N'' ﾉ ｱﾀｲ ﾊ ', N, 'ﾃﾞｱﾙ。') ;
16       IF @N = #N+1 THEN GO TO N_ERR1 ;
18       IF @N = #N-1 THEN GO TO N_ERR2 ;
20       IF @N = 0 THEN GO TO N_ERR3 ;
22       IF @N = 1 THEN GO TO N_ERR4 ;
24       IF @N = -#N THEN GO TO N_ERR5 ;
26               DISPLAY ('ﾃﾞｰﾀ ﾉ ｺｽｳ ''N'' ﾉ ｶｳﾝﾄ ｶﾞ ｵｶｼｲ。') ;
27               GO TO FINISH ;
28       N_ERR1: SCORE = SCORE+15 ;
29               DISPLAY ('''N'' ﾉ ｱﾀｲ ｶﾞ 1 ﾀﾞｹ ｵｵｷｲ。ﾊﾝﾃｲ ﾉ ｲﾁ ｵ ﾁｪｯｸ ｼﾅｻｲ。');
30               GO TO FINISH ;
31       N_ERR2: SCORE = SCORE+15 ;
32               DISPLAY ('''N'' ﾉ ｱﾀｲ ｶﾞ 1 ﾀﾞｹ ﾁｲｻｲ。ﾊﾝﾃｲ ﾉ ｲﾁ ｵ ﾁｪｯｸ ｼﾅｻｲ。');
33               GO TO FINISH ;
34       N_ERR3: SCORE = SCORE+5 ;
35               DISPLAY ('''N'' ﾉ ｱﾀｲ ｶﾞ 0 ﾃﾞｱﾙ。ｺｽｳ ｵ ｶｳﾝﾄ ｼﾅｻｲ。') ;
36               GO TO FINISH ;
37       N_ERR4: SCORE = SCORE+10 ;
38               DISPLAY ('ｺｽｳ ｵ ｶｳﾝﾄ ｽﾙ ﾌﾞﾌﾞﾝ ﾉ ｽﾃｰﾄﾒﾝﾄ ｵ ﾁｪｯｸ ｼﾅｻｲ。') ;
39               GO TO FINISH ;
40       N_ERR5: SCORE = SCORE+15 ;
41               DISPLAY ('ｺｽｳ ｵ ｶｳﾝﾄ ｽﾙ ｽﾃｰﾄﾒﾝﾄ ｶﾞ ''N=N-1;'' ﾄ ﾅｯﾃｲﾅｲｶ ﾐﾖ。');
42               GO TO FINISH ;
43       N_NORMAL:
                 SCORE = SCORE+20 ;
44               DISPLAY ('ﾃﾞｰﾀ ﾉ ｺｽｳ ''N'' ｶﾞ ﾀﾀﾞｼｸ ｶｳﾝﾄ ｻﾚﾃｲﾏｽ。') ;
```

```
 45
        /* THE DIAGNOSTIC FOR THE VARIABLE 'WORK' */
        WORK_CHK1:  /* CHECK THE ASCENDING ORDER */
         DO I = 1 TO @N ;
 46        IF @WORK(I) ¬= #WORK(I) THEN GO TO WORK_CHK2 ;
 48       END ;
 49      /* THE CONTENT OF 'WORK' IS CORRECT. */
          SCORE = SCORE+70 ;
 50       NMSG = 1 ;  MSGSTK(1) = 1 ;
 52        GO TO DSPMSG ;
 53      WORK_CHK2:  /* CHECK THE DESCENDING ORDER */
         DO I = 1 TO @N ;
 54        J = @N-I+1 ;
 55        IF @WORK(I) ¬= #WORK(J) THEN GO TO WORK_CHK3 ;
 57       END ;
 58        SCORE = SCORE+60 ;
 59        NMSG = 2 ;  MSGSTK(1) = 2 ;  MSGSTK(2) = 7 ;
 62        GO TO DSPMSG ;
 63      WORK_CHK3:  /* CHECK THE SEQUENTIAL ORDER */
         DO I = 1 TO @N ;
 64        IF @WORK(I) ¬= #SEQ(I) THEN GO TO WORK_CHK4 ;
 66       END ;
 67        SCORE = SCORE+30 ;
 68        NMSG = 3 ;  MSGSTK(1) = 3 ;  MSGSTK(2) = 5 ;  MSGSTK(3) = 11 ;
 72        GO TO DSPMSG ;
 73      WORK_CHK4:  /* CHECK THE REVERSE ORDER */
         DO I = 1 TO @N ;
 74        J = @N-I+1 ;
 75        IF @WORK(I) ¬= #SEQ(J) THEN GO TO WORK_CHK5 ;
 77       END ;
 78        SCORE = SCORE+40 ;
 79        NMSG = 3 ;  MSGSTK(1) = 4 ;  MSGSTK(2) = 5 ;  MSGSTK(3) = 11 ;
 83        GO TO DSPMSG ;
 84      WORK_CHK5:  /* CHECK THE RANDAM ORDER */
         DO I = 1 TO @N ;
 85        DO J = 1 TO @N ;
 86         IF #SEQ(I) = @WORK(J) THEN GO TO NEXTDATA ;
 88        END ;
 89         GO TO WORK_CHK6 ;
 90       NEXTDATA: END ;
 91        SCORE = SCORE+45 ;
 92        NMSG = 3 ;  MSGSTK(1) = 5 ;  MSGSTK(2) = 6 ;  MSGSTK(3) = 7 ;
 96        GO TO DSPMSG ;
 97      WORK_CHK6:
          SCORE = SCORE+45 ;
 98        NMSG = 3 ;
 99        MSGSTK(1) = 8 ;  /* THE INPUT DATA IS LOST. */
100        MSGSTK(2) = 6 ;  /* CHECK THE EXCHANGE OF DATA */
101       DO I = 1 TO @N ;
102        DO J = 1 TO @N ;
103         IF @WORK(I) = #SEQ(J) THEN GO TO NEXTITEM ;
105        END ;
106         GO TO WORK_CHK7 ;
107       NEXTITEM: END ;
108        MSGSTK(3) = 10 ;  /* THE INPUT DATA IS USED MORE THAN TWICE. */
109        GO TO DSPMSG ;
110     WORK_CHK7:
          MSGSTK(3) = 9 ;  /* THE DATA EXCEPT INPUT DATA IS CONTAINED. */
111
        DSPMSG:  /* DISPLAY THE DIAGNOSTIC MESSAGE */
         DO I = 1 TO NMSG ;
112        J = MSGSTK(I) ;
113        DISPLAY ( MSGBLK(J) );
114      END ;
115
        FINISH:  /* INFORM THE SYSTEM OF THE SCORE */
         STOP (SCORE) ;
116     END ;
```

図5　診断プログラムの例

本 PDF ファイルは 1965 年発行の「第 6 回プログラミング―シンポジウム報告集」をスキャンし、項目ごとに整理して、情報処理学会電子図書館「情報学広場」に掲載するものです。

この出版物は情報処理学会への著作権譲渡がなされていませんが、情報処理学会公式 Web サイトの https://www.ipsj.or.jp/topics/Past_reports.html に下記「過去のプログラミング・シンポジウム報告集の利用許諾について」を掲載して、権利者の捜索をおこないました。そのうえで同意をいただいたもの、お申し出のなかったものを掲載しています。

過去のプログラミング・シンポジウム報告集の利用許諾について

情報処理学会発行の出版物著作権は平成 12 年から情報処理学会著作権規程に従い、学会に帰属することになっています。

プログラミング・シンポジウムの報告集は、情報処理学会と設立の事情が異なるため、この改訂がシンポジウム内部で徹底しておらず、情報処理学会の他の出版物が情報学広場 (=情報処理学会電子図書館) で公開されているにも拘らず、古い報告集には公開されていないものが少からずありました。

プログラミング・シンポジウムは昭和 59 年に情報処理学会の一部門になりましたが、それ以前の報告集も含め、この度学会の他の出版物と同様の扱いにしたいと考えます。過去のすべての報告集の論文について、著作権者（論文を執筆された故人の相続人）を探し出して利用許諾に関する同意を頂くことは困難ですので、一定期間の権利者捜索の努力をしたうえで、著作権者が見つからない場合も論文を情報学広場に掲載させていただきたいと思います。その後、著作権者が発見され、情報学広場への掲載の継続に同意が得られなかった場合には、当該論文については、掲載を停止致します。

この措置にご意見のある方は、プログラミング・シンポジウムの辻尚史運営委員長 (tsuji@math.s.chiba-u.ac.jp) までお申し出ください。

加えて、著作権者について情報をお持ちの方は事務局まで情報をお寄せくださいますようお願い申し上げます。

期間：2020 年 12 月 18 日 〜 2021 年 3 月 19 日
掲載日：2020 年 12 月 18 日

プログラミング・シンポジウム委員会

情報処理学会著作権規程
https://www.ipsj.or.jp/copyright/ronbun/copyright.html