鳥取大学教育学部
研究報告（自然科学）
40（1991），33-45

BASIC 言語実習用エラー自動記録システム

西　田　英　樹*

Hideki NISHIDA: Automated Error Acquisition System for Training the Programming Language, BASIC

（1991年8月31日　受理）

1　は　じ　め　に

本学部では，教員養成コースの学生に対する情報処理入門教育として「コンピュータ実習」「情報基礎実習」「家庭情報処理実習」などいくつかの授業科目が開設されている。これらの授業では主にプログラミング言語 BASIC を用いた教育がなされている[1)-4)]。入門教育としてプログラミング言語を教えることの是非や，BASIC 言語が適切かと言う問題も無いわけではないが，種々のアプリケーションソフトの入手が予算的に困難な現状では，やむを得ないことである。

BASIC 言語はインタープリタ方式であるため，作成したプログラムが編集画面から直ちに実行でき，しかも細かい約束が少ないので，本学部の学生のように将来情報処理関係の技術者になる可能性は低く本質的にはいわゆる文科系の学生にとって馴染みやすいといえる。しかも言語を用いた簡単なプログラミング技術を修得させることはコンピュータによる処理の仕組み即ち分岐や繰り返し，ジャンプといった単純な機能の組合せが複雑かつ高度な処理の基本になっていることを理解させる上で効果的であり，またコンピュータに限らず一般の問題解決においてもその手順を論理的，体系的に組み立てるための訓練にもなる。

このようにこれらの授業科目では BASIC 言語を使ってはいても言語そのものを教えることが主目的ではなく，BASIC 言語を手段として情報処理の仕組みを教えることを目的としていると考えることができ，現状ではそれなりに効果をあげている。

BASIC 言語をクラス単位で実習するに当たって，本学部ではすでに独自の CAI 方式が考えられている[4)]。この方式はそれぞれの学習者が理解力に応じて個別の進度で実習を進めることが出来る点で優れている。一方，この方式は学習者が課題の意味を十分には理解・消化できなくてもファンクションキーを押せば先へ進むことができるという欠点を持っており，エラーの原因や対処の仕方が判らないまま次々と先に進んでしまう光景も時折見受けられる。このように多人数相手のしかも実習形式の授業の場合，実習補助者が授業中絶えず巡回して質問やエラー等に対応していても全ての学習者が確実に理解しているかどうかを把握することは困難であり，つまづきを見落としていることが多いと思われる。

本報告では，学習者がプログラムを編集，実行する過程で発生させたエラーの内容や頻度を

* 技術教室

逐一自動記録する常駐型システムを開発した。このシステムを用いれば内容の理解度，定着の程度を把握でき，実習方法や指導内容の改善に資することができる。

2 システムの概要

本システムの開発に使用したパソコンは NEC PC-9801E であり，その機器構成は標準的なものである。PC-9801 シリーズであればこれ以外の機種でも動作可能である。

PC-9801 シリーズで使われている N_{88} BASIC には，NEC 社のパソコン用の標準 OS として準備されている N_{88} 日本語 BASIC (86)（以下 DISK-BASIC と略す）と，MS-DOS 上で動くインタープリタシステム N_{88} 日本語 BASIC (86)（MS-DOS 版）（以下 DOS-BASIC と略す）とがあるが，本システムでは実際に授業で使われている DOS-BASIC を対象とした。開発に実際に使用した OS は MS-DOS V3.1，BASIC インタープリタは N_{88} 日本語 BASIC (86)（MS-DOS 版）3.0 である。

発生エラーに関する情報を随時ディスクに記録するためには，記録プログラムは勿論の事，そのプログラムをインタープリタの機能や動作を損なう事なくメモリー中へ常駐させる方法，エラーを検出して常駐プログラムへ制御を渡す方法などを工夫しなければならない。以下にこれらについて述べる。

2-1 エラー検出の方法

BASIC では，エラーが発生するとそれまでにおこなっていた処理を中断して，警告音を鳴らし，ディスプレイ画面にエラーメッセージを表示してコマンドレベルに復帰する。エラーメッセージの表示はエラー発生時にのみ起こる特異的な現象であり，これを検出して本システムの中心となるエラー情報記録プログラムを実行させることにより，エラー発生直後の様々な情報を得ることが出来る。

DOS-BASIC の内部構造については一般には公開されていないが，すでに内容が公開されている DISK-BASIC と共通する部分が多く，ある程度は推測することが出来る[5]。DISK-BASIC ではこのエラーメッセージの表示にソフトウェア割り込み $C4_H$ 番が使われている。DOS-BASIC でのエラーメッセージ表示には本研究における解析の結果 $C4_H$ 番の割り込みは使用可能であるにも関わらず使われておらず，$C6_H$ 番の割り込みが使われていることが明かとなった。

本システムではこの割り込み $C6_H$ のジャンプ先アドレスをエラー記録プログラムの先頭アドレスに変更することでエラーが発生する度にインタープリタから記録プログラムに一旦制御を移すこととしている。しかし，この割り込みはエラーメッセージ表示以外にも使われているので，記録プログラムの入口で，エラーかどうかの判定が必要である。

2-2 エラー情報の取得方法

DOS-BASIC 上でエラーが発生した場合，エラーコード，エラー発生行番号，エラーの起こったテキストの行内容，直前にダイレクトモードで入力した文字列などをワークエリアから得ることが出来る。またエラーが発生した日時は MS-DOS のシステムコール[6] を利用して知ることが出来る。記録プログラムでそれらのデータの取得に利用した機能やアドレスを表1に

表1　エラーに関する各種情報の取得方法

内　　容	取　得　方　法	取得時のデータの表現形式
年　月　日	MS-DOS システムコール ファンクション $2A_H$	16進数値 年2バイト，月1バイト，日1バイト
時　分　秒	MS-DOS システムコール ファンクション $2C_H$	16進数値 時1バイト，分1バイト，秒1バイト
エラーコード	BASIC ワークエリア　システム制御情報領域 オフセットアドレス $6E2_H$	16進数値，1バイト
エラー発生行番号	BASIC ワークエリア　システム制御情報領域 オフセットアドレス $6E0_H$	16進数値，2バイト
テキスト内容	BASIC ワークエリア テキスト領域	中間コードによる内部形式
ダイレクト入力内容	BASIC ワークエリア　システム制御情報領域 オフセットアドレス $1C00_H$	アスキー形式

システム制御情報領域のセグメントアドレスは，インタープリタ起動後に調べてデータ収録プログラム中に追記する。

一覧にして示す。表中のシステム制御情報領域およびテキスト領域のセグメントアドレスは COMMAND. COM のバージョンやデバイスドライバ使用の有無，DOS-BASIC のバージョンなどによって異なる。これら値は DOS-BASIC の関数 SEGPTR (7)，SEGPTR (6) によって知ることが出来る[7]。

2-3　システムのソフトウェア構成と組込み方法

本システムではインタープリタ起動に引き続きエラー情報の記録プログラムをメモリー中に組み込んで常駐させておき，実習中にエラーの発生を検出した場合ソフトウェア割り込みによってこのプログラムに制御を移し，情報収集作業を実行した後再びインタープリタに制御を渡す事としている。また集められたデータは後日まとめて解析される。このように本エラー記録・解析システムに必要な内容は大きく分けると組込み，記録，解析という三つの機能に分けることが出来る。これらの目的に使用するプログラムはそれぞれ機能と使用環境が大きく異なるため，

(1)　ERR1000. BAS　：システムの組込みプログラム
(2)　ERROR. COM　：エラー情報記録プログラム
(3)　CNVTOASC. BAS ：記録されたデータの解析，表示プログラム

の三つに分けて個別に作成されている。各プログラムの具体的な働きは次章に述べる。

機械語で記述された記録プログラムのメモリーへの書き込みは，

```
N88BASIC /T:RUN "ERR1000. BAS"
```

という内容のバッチファイル BASIC. BAT によってインタープリタ起動に引き続き自動的に行われ，組込み用プログラム ERR1000. BAS によってメモリーのフリーエリア上限付近に格納 (BLOAD) される。この記録プログラムは実習者がプログラムを作成したり実行したりする間もメモリー中に常駐し，BASIC の CLEAR 文を実行しない限り消えることはない。一方組込みに用いたプログラム ERR1000. BAS は必要な処理を行った後自ら消滅し，通常のコマンドレベルで実習者による使用を待つ状態になる。

システム組込みの一連の流れを図1に示す。

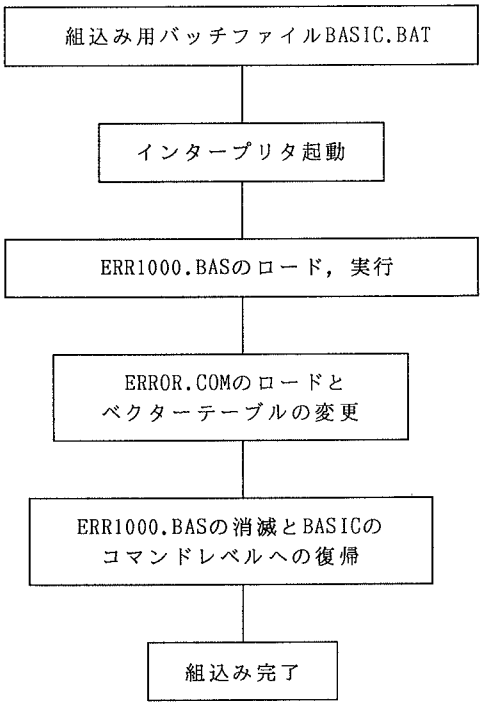

図1 システムの組込みと起動

3 プログラムの内容

前述のごとく本システムでは三つのプログラムが必要である。以下にこれらのプログラムの内容を詳しく述べる。

3-1 組込みプログラム (ERR1000.BAS)

このプログラムの働き（機能）は次の4つである。

(1) 機械語プログラム領域を64キロバイト確保し，そのセグメントアドレスをSEGPTR (2) 関数にて調べ，その領域のオフセットアドレス 100_H から記録プログラムを書き込む。

(2) エラー情報の取得に不可欠なシステム制御情報領域のセグメントアドレスをSEGPTR (7) 関数によって調べ，これを記録プログラムのオフセットアドレス $11B_H$，$11C_H$ に書き込む。

(3) ソフトウェア割り込み INT $C6_H$ によって記録プログラムがトリガされるよう，本来のソフトウェア割り込み $C6_H$ 番のジャンプ先アドレスを，通常は使われていないジャンプテーブルの FF_H 番の位置に移動しておき，$C6_H$ 番には新しいジャンプ先として記録プログラムが常駐しているアドレスを書き込む（ベクターテーブルの変更）。

(4) 以上の作業の後，BASIC 起動時の初期画面を再表示し，さらに組込みプログラムをメモリーから消去して BASIC のコマンドレベルに戻る。

これらの機能の一部には DOS-BASIC に固有の関数を利用しなければならないものがあるので，このプログラムは DOS-BASIC で記述されており，リストを末尾の付図1に掲載した。

3-2 記録プログラム (ERROR.COM)

記録プログラム ERROR.COM のおもな働きはエラーに関する各種の情報をディスクに記録することである。このプログラムは割り込み $C6_H$ が発生する度に稼働する常駐型プログラムとしなければならないので，アセンブリー言語 MASM の書式にしたがって記述されアセンブルされた機械語プログラムである。このプログラム全体の大まかな働きを図2にフローチャートで示す。

このプログラムは本来の割り込み $C6_H$ に寄生する形を取っているので，プログラムを終了した時点では各レジスタなどを完全にもとの状態の戻しておかなければならない。このためプログラムの先頭でスタックセグメントとスタックポイントを保存し，新たに独自のスタック領域を定義してこのプログラムで使用する全てのレジスタをそこに待避させ，プログラム終了直

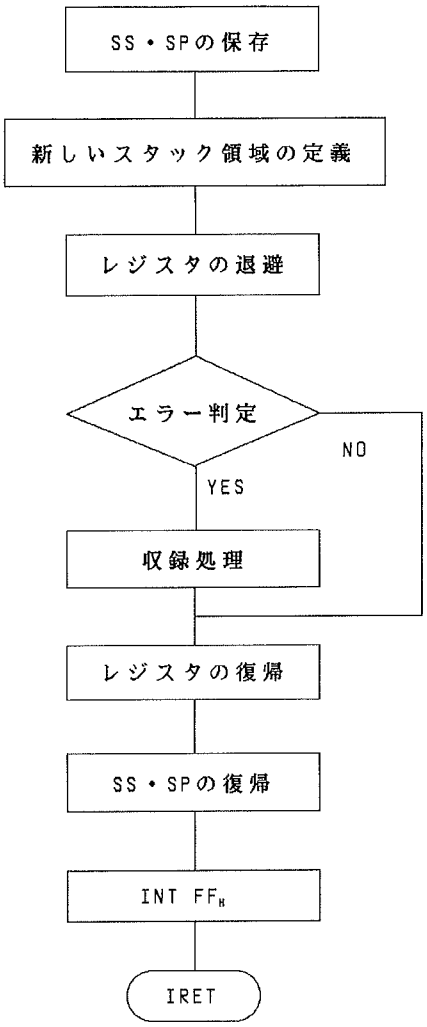

図2　ERROR. COM による処理の流れ

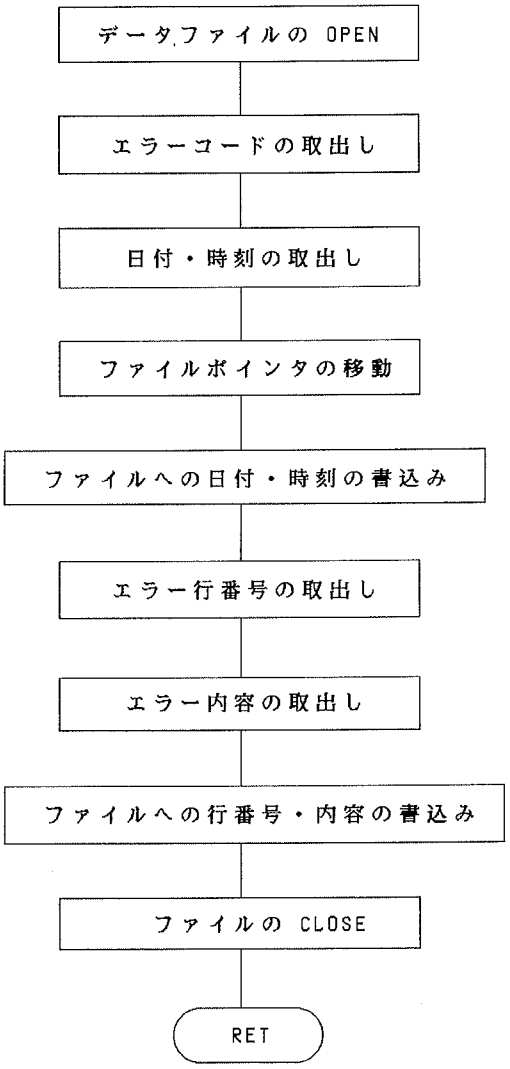

図3　データ収録サブルーチンの処理の流れ

前に各レジスタを復帰させている。割り込み INT C6$_H$ はエラーメッセージ表示以外にも様々な機能を持っており，エラー発生時以外にも度々呼ばれるので，記録プログラムの入口でまずBASIC ワークエリアのエラーコードを調べ，エラーで呼ばれたので無ければ直ちに本来の INT C6$_H$ 処理に分岐させている。一方エラーコードの値が 0 以外であればエラーが発生したものとみなして情報収集作業を実行した後，本来の INT C6$_H$ を実行することとしている。

図2中，収録処理となっている部分がエラー情報の記録をおこなうサブルーチンであり，この部分の内容は図3にフローチャートで示した。ドライブAのディスクには，予めデータ記録用のシーケンシャルファイル ERROR. DAT が作成されていなければならない。記録処理ではまず ERROR. DAT ファイルを追加モードでオープンし，エラーコード，年，月，日，時，分，

秒に関するデータをメモリから取り出してファイルに記録している。年データのみ2バイト他は1バイトであり，これらのデータはファイル中で特別の意味を持つ場合がある $1F_H$ 以下の数値をなくすために 20_H が本来の値に単純に加算されている。この後データの区切りとして改行コード $0D_H$, $0A_H$ を記入する。

続いてエラー行番号を取り出してダイレクト入力により発生したものかプログラム実行過程で発生したものかを判定し，ダイレクト入力であれば識別のためのコード 30_H と入力内容を，プログラムであればコード 31_H と，エラー行番号をもとにテキスト領域から対応する行を捜してリンクポインタ以下行末コード 00_H までを，そのままファイルに記録する。ダイレクト入力された内容はアスキー形式でファイルに記録され，プログラムのエラーは中間コードによる内部形式のまま記録される。中間コード形式で表されたテキスト内容の内，00_H, $0A_H$, $0D_H$, $1A_H$ は本システムで記録するデータファイル中で特別の意味を持つので，それぞれ EA_H, $E8_H$, CA_H, EF_H に変換している。このうち CA_H, EF_H はもともと中間コードとしては未定義である[7]。一方本来は EA_H は ISET 命令に，$E8_H$ は CMD 命令に対する中間コードとして定義されている[7]が，これらの命令は入門段階で使われることはないのでこれらの変更を行っても支障はないと考えられる。

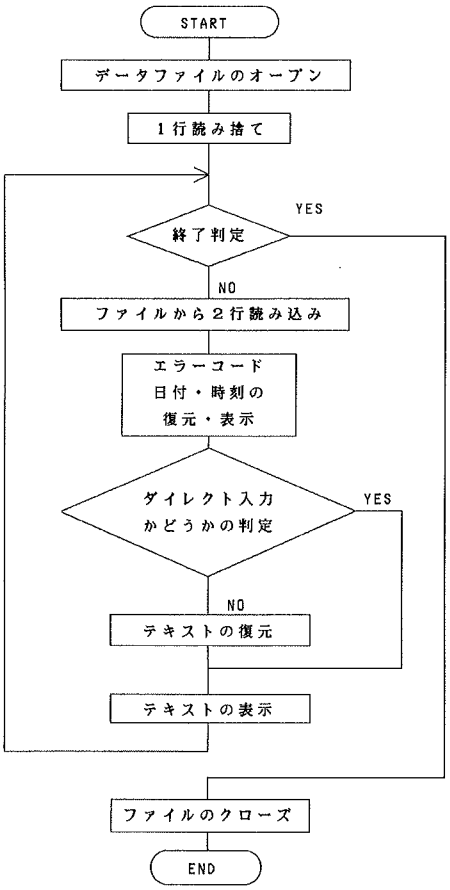

図4 CNVTOASC.BAS の処理の流れ

一件のエラーに関する全てのデータの記録が終わると，ファイルをクローズしてメインプログラムに復帰する。このプログラムのリストは末尾の付図2から付図6に掲載している。

3-3 解析・表示プログラム (CNVTOASC.BAS)

前述のように，記録する時点では処理時間を短縮するためエラー情報の表現形式が種々異なるので，これらをデータの区切りとなる改行コードやプログラムかダイレクトかの識別のためのコードなどを利用して解析・表示時に判別し，全てアスキー形式の文字列に変換して表示しなければならない。解析・表示プログラム CNVTOASC.BAS のフローチャートを図4に示す。

このプログラムではファイルから行単位で読み込んだデータを1バイトずつ調べ，エラーコード，年，月，日，時，分，秒のデータからは 20_H を引いた後十進形式に変換して表示している。一方中間コード形式で記録されているエラー部分のプログラムテキストは記録時に変換された EA_H, $E8_H$, CA_H, EF_H のコードを 00_H, $0A_H$, $0D_H$, $1A_H$ にそれぞれ復元した後，アスキー形式に直して表示している。エラーがダイレクト入力により発生したものについてはそ

の内容がアスキー形式で記録されているので，そのまま表示している。

データの記録解析の例として図5に試験的にエラーを発生させた後の ERROR. DAT ファイルのダンプ結果を，図6には CNVTOASC. BAS によるこれの解析表示結果を示した。行番号の無いものはダイレクト入力である。これらのデータは CRT 画面，プリンタ，ディスクファイルのいずれにも出力可能である。

このプログラムのリストは約200行あり，やや長いので掲載を割愛した。

```
Dump Version 2.1

00000000  54 45 53 54 20 44 41 54-41 0D 36 E7 07 28 38 26   TEST DATA.6..(8&
00000010  30 3B 22 0D 0A 30 41 3D-00 0D 0A 36 E7 07 28 38   0;"..0A=...6..(8
00000020  26 30 3B 2A 0D 0A 31 09-EA E8 EA 01 41 EA F1 00   &0;*..1..韭.A...
00000030  0D 0A 1A 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

図5　ERROR. DAT ファイルのダンプ例

```
ERR= 22 1991: 8:24   16:27: 2          A=
ERR= 22 1991: 8:24   16:27:10           10 A=
```

図6　図5に示すデータの解析結果

4　使用結果と検討

本学技術専攻及び家庭専攻の2年次学生計20名に対する実習授業で本システムを利用してはじめの4週間エラーの記録テストをおこなった。記録された実際のデータの一例を図7に示す。

これらの試験的な利用と実際に集まった生のデータから，このシステムに関して以下の点が明らかとなった。

(1)　システムは意図した通り良好に作動する。

(2)　エラーが起こるとその都度データをディスクに記録するので，エラーメッセージの表示やビープ音の発生までの間に1秒程度の時間遅れが生ずる。
　この間ディスクのアクセス音が出る。

(3)　起動時にインタープリタの初期画面が二度表示されることになるので，画面が一瞬ちらつく。

(4)　インタープリタの起動方法が正規の場合と異なるので，指導がやや煩雑である。

(5)　データは内部表現のまま記録されるものが多いので，ファイルサイズがコンパクトである。

(6)　エラーの具体的な内容と秒単位までのエラー発生時刻が記録されるので，一つのエラー

```
ERR=  2 1991: 4:17   15:14:40      10 PRANT
ERR= 13 1991: 5:11    9:58:45     LOAD 2B:TEST1"
ERR= 53 1991: 5:11   10:25:11     FILES "B"
ERR= 53 1991: 5:11   10:31:57     LOAD "TEST1
ERR=  2 1991: 5:11   10:34:39     TEST
ERR=  2 1991: 5:11   10:35:10     LOAD "B:TEST2",A
ERR= 22 1991: 5:18    8:27:50     SAVE
ERR= 22 1991: 5:18    8:41: 1     SAVE
ERR=  2 1991: 5:18    8:45:44     OK
ERR=  2 1991: 5:18    8:47: 6     L
ERR=  2 1991: 5:18    8:54:41     FIKES
ERR= 13 1991: 5:18    8:56:35     LOAD SIMIZU-5
ERR= 51 1991: 5:18    9:24: 7     ¥AB=5
ERR=  2 1991: 5:18    9:27:27     A_B=5
ERR=  2 1991: 5:18    9:50: 7     LISTND
ERR= 11 1991: 5:18    9:53: 9      40 Y2=(X- 2)^ 2+ 1/(X- 2)
ERR=  2 1991: 5:25    8:26:58     OK
ERR=  2 1991: 5:25    8:32:28     OK
ERR=  2 1991: 5:25    8:33: 3      35 PRIN L;"ﾛｰﾚﾙ"
```

図7　授業で収録されたデータの一例

が発生してからそれを修正するのにどんな経過をたどったか，修正にどれだけの時間を要したか，等が手にとるようにわかる。

(7)　問題解決の手順における論理上の間違いはチェックできない。

(2)及び(3)は技術的な問題点であるが，実用上は問題ないと思われる。(4)については混乱を招く恐れがないとは言えないが，理由をはっきり説明することにより避けられる。(5)についてはデータが暗号化されて記録されていると考えることもでき，データの無断目的外使用に対する簡単な保護にもなっている。(6)については単に命令を理解しているかどうかでなく本人の思考能力の優劣までわかってしまうので，本システムの使用に当たっては本人の了解をとることが必要と考えられ，さらにデータの取扱には注意を要する。(7)は困難な課題で，このシステムの限界である。

この他，試験的に利用した期間の実習では BASIC の起動，プログラムの入力・編集・保存・呼出し・実行，入出力命令，代入あたりの範囲を扱ったのでエラーの一般的な傾向としてはダイレクト命令のエラーが意外に多かった。インタープリタ上であるにもかかわらず OS の命令を使用したり，プログラム実行中であるにも関わらず編集しようとしたりするなど，OS，インタープリタ，BASIC 言語で書いたプログラムの実行過程，というように使用環境にいくつかの階層がある中で自分のいる位置が把握できていない点が明らかに伺えた。

5 ま と め

本報告では，エラーの内容を逐一自動記録する常駐型システムを開発した。試験的にデータ収集を行ったところ内容の理解度，定着の程度を詳細に把握でき，きわめて有効であることが確かめられた。

本システムで考案したエラー検出方法は，エラー発生時にインタープリタが用意しているエラーメッセージ以外の詳しい説明や修正のヒントなどを画面に表示することにも利用でき，一斉授業におけるきめ細かな指導にも応用可能である。また，その場に居合わせなくても一種の学習履歴が残るので，後からでも指示，指導，助言が可能である。さらに学習者自身が，分からなかったところの復習に利用することもできる。このように様々な応用の可能性があるが，これを教育的にどう活用するかは今後の課題として残る。

引 用 文 献

1) 山岸正明：鳥取大学教育学部研究報告 教育科学，**27** (1985)，65.
2) 山岸正明：鳥取大学教育学部研究報告 教育科学，**30** (1988)，19.
3) 山岸正明：鳥取大学教育学部研究報告 教育科学，**32** (1990)，45.
4) 山岸正明，西田英樹，安藤由和，和泉澤正隆，清水寛厚，岡田昭明，大塚譲，浜崎修：鳥取大学教育学部研究報告 教育科学，**30** (1988)，39.
5) 浅野泰之，壁谷正洋，金磯善博，桑野雅彦：PC-9801 システム解析（上）（アスキー出版局，1983).
6) MS-DOS TM 3.1 プログラマーズリファレンスマニュアル.
7) N88-日本語 BASIC (86) (MS-DOS 版) 3.0 ユーザーマニュアル.

Abstract

A resident program to record contents of errors automatically which occur during training the programming language BASIC was built. This program can be triggered by every errors and can record the date, the time, the error code, the line number and contents of the line or characters directly typed. These data will be stored sequentially in a disk file. This program can be operated on the personal computer, PC9801 series.

```
1000 '      save "ERR1000.BAS",A
1010 CLS
1020 CLEAR &H1000
1030 DEF SEG=SEGPTR(2)
1040 B=SEGPTR(2)
1050 BH=INT(B/256)
1060 BL=B-BH*256
1070 '
1080 A=SEGPTR(7)
1090 AH=INT(A/256)
1100 AL=A-AH*256
1130 '---------------------------------------
1140 DEF SEG=SEGPTR(2)
1150 BLOAD "ERROR.COM",&H100
1160 POKE &H11B,AL
1170 POKE &H11C,AH
1180 DEF SEG=0
1190 'ADDRESS OF ORIGINAL INT C6
1200 C0=PEEK(&H318)
1210 C1=PEEK(&H319)
1220 C2=PEEK(&H31A)
1230 C3=PEEK(&H31B)
1240 '
1250 IF C0+C1*256=0 AND C2+C3*256=B THEN *STARTG
1260 '
1270 'ADDRESS OF NEW INT C6
1280 POKE &H318,0
1290 POKE &H319,1
1300 POKE &H31A,BL
1310 POKE &H31B,BH
1320 '
1330 'INT FF  AS  INT C6
1340 POKE &H3FC,C0
1350 POKE &H3FD,C1
1360 POKE &H3FE,C2
1370 POKE &H3FF,C3
1380 *STARTG
1390 PRINT "NEC N-88 BASIC(86) version 4.0"
1400 PRINT "Copyrite (C) 1984,85,86 by NEC Corporation"
1410 PRINT "453108 Bytes free"
1420 NEW
1430 END
```

付図1　組み込みプログラム ERR1000.BAS の全リスト

```
;---------------------------------
;ERROR.ASM
;---------------------------------
CODE SEGMENT
     ASSUME CS:CODE,DS:CODE
     ORG 100H
START:
     PUSH AX
     MOV CS:[0400H],SS
     MOV CS:[0402H],SP
     PUSH CS
     POP SS
     MOV SP,0400H
     PUSH DS
     PUSH CS
     POP DS
     PUSH BX
     PUSH CX
     PUSH DX
     PUSH SI
     PUSH ES
     PUSH DI
     PUSHF
     MOV AX,0000H
     MOV ES,AX
     MOV BH,ES:[06E2H]
     CMP BH,0
     JE P
     CALL MAIN
P:
     POPF
     POP DI
     POP ES
     POP SI
     POP DX
     POP CX
     POP BX
     POP DS
     MOV SS,CS:[0400H]
     MOV SP,CS:[0402H]
     POP AX
     INT 0FFH
     IRET
```

付図2　記録プログラム ERROR.ASM のリスト

```
;---------------------------------
MAIN:
;---------------------------------
FCREAT:
     MOV AH,3DH
     MOV DX,OFFSET FNAME
     MOV AL,1
     INT 21H
     JC ERROR
     MOV FILE,AX
     MOV DI,OFFSET BUFF1
ERR_NO:
     MOV AH,ES:[06E2H]
     ADD AH,20H
     MOV [DI],AH
NENGETU:
     MOV AH,2AH
     INT 21H
     ADD CX,20H
     MOV [DI+1],CX
     ADD DH,20H
     MOV [DI+3],DH
     ADD DL,20H
     MOV [DI+4],DL
     ADD AL,20H
     MOV [DI+5],AL
JIKAN:
     MOV AH,2CH
     INT 21H
     ADD CH,20H
     MOV [DI+6],CH
     ADD CL,20H
     MOV [DI+7],CL
     ADD DH,20H
     MOV [DI+8],DH
     MOV BYTE PTR [DI+9],13
     MOV BYTE PTR [DI+10],10
FPMOV:
     MOV AH,42H
     MOV AL,2
     MOV BX,FILE
     MOV CX,0FFFFH
     MOV DX,0FFFFH
```

付図3　記録プログラム ERROR.ASM のリスト（続き）

```
      INT 21H
FWRITE:
      MOV AH,40H
      MOV BX,FILE
      MOV DX,OFFSET BUFF1
      MOV CX,11
      INT 21H
ER_CODE_CL:
      MOV BYTE PTR ES:[06E2H],00H
      JMP MDCODE
;--------------------------------
ERROR:
      MOV DX,OFFSET ER_MSG
      MOV AH,9
      INT 21H
      JMP FCLOSE
;--------------------------------
MDCODE:
      MOV SI,2280H
      MOV LPNEXT,SI
      MOV LNP,4
      MOV AX,ES:[06E0H]
      CMP AX,0FFFFH
      JNZ AAA
      MOV DI,OFFSET BUFF2
      MOV AH,30H
      MOV [DI],AH
      INC DI
      MOV SI,1C00H
DCM:
      INC LNP
      MOV AH,ES:[SI]
      CMP AH,0
      JE FCLOSE3
      CALL D_A_1A
      MOV [DI],AH
      INC DI
      INC SI
      JMP DCM
AAA:
      MOV SI,LPNEXT
      MOV DI,OFFSET BUFF2
      MOV CX,ES:[SI]
```

付図4　記録プログラム ERROR. ASM のリスト（続き）

```
      CMP CX,0
      MOV LNP,4
      JZ FCLOSE3
      MOV LNP,CX
      ADD LNP,3
      ADD LPNEXT,CX
      MOV BX,ES:[SI+2]
      CMP AX,BX
      JNZ AAA
      DEC CX
      MOV AH,31H
      MOV [DI],AH
      INC DI
      INC LNP
LP:
      MOV AH,ES:[SI]
      CALL D_A_1A
      MOV [DI],AH
      INC DI
      INC SI
      LOOP LP
FCLOSE3:
      MOV BYTE PTR [DI],0
      MOV BYTE PTR [DI+1],13
      MOV BYTE PTR [DI+2],10
      MOV BYTE PTR [DI+3],1AH
FWRITE2:
      MOV AH,40H
      MOV BX,FILE
      MOV DX,OFFSET BUFF2
      MOV CX,LNP
      INT 21H
FCLOSE:
      MOV AH,3EH
      INT 21H
ER_CODE_CL2:
      MOV BYTE PTR ES:[06E0H],00H
      RET
;--------------------------------
D_A_1A:
      CMP AH,0DH
      JNE PAS1
      MOV AH,0CAH
```

付図5　記録プログラム ERROR. ASM のリスト（続き）

```
PAS1:
     CMP AH,0AH
     JNE PAS2
     MOV AH,0E8H
PAS2:
     CMP AH,1AH
     JNE PAS3
     MOV AH,0EFH
PAS3:
     CMP AH,00H
     JNE PAS4
     MOV AH,0EAH
PAS4:
     RET
;---------------------------------
ER_MSG  DB "File can't open.$"
;---------------------------------
     FNAME   DB 'ERROR.DAT',0
     FILE    DW 1 (?)
     BUFF1   DB 20 (?)
     LNP     DW 1 (?)
     LPNEXT  DW 1 (?)
     BUFF2   DB 255 (?)
;---------------------------------
    CODE    ENDS
            END START
```

付図6　記録プログラム ERROR.ASM のリスト
(続き)