[08_02]情報処理教育広報表紙奥付等

出版情報：情報処理教育広報. 8 (2), 1985-12. Educational Center For Information Processing, Kyushu University
バージョン：
権利関係：

あなたはコンピュータに遊ばれている

広重　法道*

1. はじめに

私がはじめて、情報処理教育センターのコンピュータを使用したのは丸々6年も昔のことでした。教養部のころで、たしか火曜日の午後に林先生の情報科学の講義があり、そのときに利用して以来、情報処理教育センターのコンピュータにはいろいろとお世話になりました。

当時の教養部には今みたいな端局はありませんでした。我々学生はＴＳＳなどという高級なものを使えるはずもなくカードによるバッチ処理を利用していました。我々はマークカードと呼ばれるカードを買って来て、必死にマークを塗りつぶしてはカードリーダーを操作する人に渡し、出力される結果を長いときには３０分以上もひたすら廊下で待っていたものでした。そして、返ってきた出力には当然のごとくエラーがあったのですが、エラーにどういう種類があってこのエラーがどこで起きたかなどさっぱりわからず、友人とあれやこれやと推測しながらカードを鉛筆と消しゴムで書きなおして、またカードリーダーにかけてもらうということの繰り返しでした。今思えばあきれるばかりですが、それでも当時は〝これがコンピュータなんだ〟と感動していました。

それから2年の後期に本学に来て物理学科の情報講義で半年間、また4年になってから研究室の北村先生が情報講義を担当している関係で情報処理教育センターを利用させてもらいました。この間、生徒として先生の講義を受けたり、また逆に授業補佐として講義をしたり、またセンターの補佐としてプログラム相談員をしたりいろいろな立場で、いろいろな人を見、いろいろなことを感じてきました。そこで今回は情報処理教育センターのことを中心にコンピュータに関することについて、今感じていることを学生という身分を顧みずに率直に書きたいと思います。

2. コンピュータ教育とは何を目指すのか

ここ数年のコンピュータの一般会社への広がり方には目を見はるものがありますし、また学問の分野でもコンピュータを使った研究が増えているようです。こういう状況の中でコンピュータを使った情報処理教育が注目を集めていることはあたりまえです。しかしどういうことを目標にすればよいか、先生方も学生の方もどのように考えていらっしゃるのでしょうか。大部分の人は、〝プログラムを

* 大学院理学研究科（プログラム相談員）

自分で組んでコンピュータを操作できるようになる' ということを目標にしているのではないでしょうか。確かにもっともな目標だとは思いますが、果たしてこれが妥当な目標なのでしょうか。

教養部にいたころ法律学の講義を受けたことがありますが「法律をたくさん覚えないといけないだろうな。」と思っていた私は単位のためだと渋々1回目の講義にでました。その時間に先生が「法律学とは法律を覚えるための学問ではなく、ＬＡＷ　ＭＩＮＤを知るための学問である。」と言われて、なるほどと心のわだかまりがとれたようなことがありました。これと同じことがコンピュータ教育にもあてはまると思うのです。つまり 'コンピュータに何ができるのか、そして何ができないのか' を判断する能力をつけることが情報処理教育において一番重要なことではないでしょうか。これさえできればプログラミングなどできなくてもいっこうに構わないと思うのです。ただこの判断能力を身につけるためには、ある程度のプログラミング経験がないと話にならないでしょうが、少なくとも学生が半年か1年で身につけたプログラミング能力など何の役にも立たないでしょう。なぜならコンピュータが一般会社に広まったとはいっても、その使用方法は多種多様であり、かつ専門的だからです。実際、コンピュータ関係の会社の多くは新入社員にコンピュータに関する能力はゼロとみなした教育をしているようです。また私が会社訪問に行ったとき「ソフトとハードとどちらがやりたいか。」という質問に「私は3年間いろいろなプログラムを組んできたのでソフトに自信がある。」と答えたら、「君たち学生がたとえ3、4年プログラムを組んできても、我々の世界では通用しない。」と言って怒られました（会社訪問に行って怒られるとは思ってもみませんでした）。

では学校のコンピュータ教育は何の役にも立たない無意味なものでしょうか。そうではないはずです。プログラミング能力そのものは役に立たなくても 'コンピュータを操作したことがある' という経験は非常に重要だと思います。コンピュータを使い始めてしばらくすると大部分の人は良い意味にも悪い意味にも「コンピュータとはこんなものか。」という印象を持たれたことでしょう。この「こんなものか。」という印象こそ、 'コンピュータに何ができ、何ができないか' の判断を感覚的にくだしている点で一番大切だと思います。

では現在の九州大学情報処理教育センターでの学習でこういう判断能力を学生が十分に身につけているでしょうか。私はＮＯだと思います。なぜなら大部分の人にとって「こんなに目と頭と手が疲れるものなのか。」とか「こんなに融通がきかないものなのか。」とかの印象しか残っていないようだからです。そこでこの判断能力をもっともっと身につけてもらうためにはどうしたらよいか、その問題点も含めもう少し具体的に書いてみます。おおまかに学生の方へと先生方へと2項目に分けています。

3. 学生の方に

ここ数年マイコンが一般的なものになってきて理系文系問わず研究室ではほとんどのところで少なくとも1台ぐらいは置いてあるでしょう。また個人で買っていろいろと操作している人も多いことでしょう。そういう状況ですから、「コンピュータは情報処理教育センターのＦＡＣＯＭを触ったのが

初めてです。」という人はそれほど多くないのではないかと思います。それでもいわゆる'大型機'というものを操作するということは、大部分の人が初めてのことでしょう。マイコンと比べるといいところもあれば扱いにくいところもあります。いろいろ苦労もするでしょう。しかし、なによりもまず"大型機をただでしかもこんなすばらしい環境で使える"ということに感謝しなければなりません。教育用専用にコンピュータマシーンを独立して持っている大学というのは、日本広しといえどそんなにないのです。ましてや最新のＦＡＣＯＭをＴＳＳをベースとし、その他いろいろのアプリケーションソフトもそろえ、これほど多くの学生にその使用を許しているシステムは旧帝大の中でもトップクラスなのです。卒業後そういうシステムを利用しようとしても、まずそんな贅沢は許されないでしょう。もし、マイコンをベースとしプリンター、ディスク等を一式そろえて、ＦＯＲＴＲＡＮ、Ｐａｓｃａｌ等を走らせようとするならば１００万円を越えるお金が必要なのです。それにアプリケーションソフトを付け加えようものなら１０万単位でお金が飛んで行くでしょう。あなたは本当にすばらしい環境の中にいることを忘れてはいけません。

とはいっても初心者にとってコンピュータとは、目と指と頭の単なる苦痛でしかないという場合が多いようです。かくいう私も２年後期のときに物理情報処理の講義で、当時のＡＣＯＳ（ＮＥＣの大型機）を使っていた頃は、タイプが打てなかったので一文字一文字キーボードからひろって打ち込むという状態で、目は疲れ、頭の中はパニック、とても"プログラミング"どころではありませんでした。またＲＵＮしてもエラーばかりで、そのエラーも何のエラーかどうしたらなおるのか全く分からないような有様で、半年たった頃は"物理に数値解析は邪道だ"といきがって完全にコンピュータ嫌いになってしまいました。その後情報処理教育センターでいろいろな学生を見てきましたが、大部分の人が私と同じようなことをしていて本当に気の毒になってきます。またそういう教育は、情報処理とはとても言えないと思うのです。

情報処理教育またはコンピュータ教育というのは難しい問題ですが、少なくとも一文字一文字キーボードをたたいてその苦労を知るというのではないはずです。なのに大部分の人は全然関係ないところで苦労していて、挙句の果てに「これがコンピュータか」と思い込んでコンピュータ嫌いになっているようです。そこで初心者の人を対象にどうしたらいらない苦労を取り除けるか考えてみたいと思います。それには、タイピング、エディタ、オペレーティングシステム、エラーの４点について学習することが大事です。

① タイピング

人間がコンピュータに何か情報を伝えるためのインターフェイスとして、先端技術には音声入力とか筆跡入力とかあるようですが、現在のところキーボードを使わざるをえません。そのためプログラムを打ち込んだりコンピュータを操作するには、どうしてもタイピングということがからんできます。しかしほとんどの学生の人は、タイプを打つどころかどの文字がどこにあるのかということも知らないので、当然ながらキーボードと睨めっこをし、ひどい人はキーボードと格闘し探すキーを見つけようものなら鬼の首を取るかのように凄まじい力でキーをたたくという状態のようです。

目は疲れ頭の中は文字を追うことで必死で、本質であるプログラムのことなど全く考える状態どころではない。時間をかける割には全然進まない。キーボードから一文字一文字探して入力するという、このようなやり方がどんなにアホらしいか、みんなわかっていると思うのです。タイピングは情報処理の本質ではないのにこういう状態では非能率です。そこでこの際タイピングも少し勉強してはどうでしょうか。パチパチ流れるように打つ必要はまったくありません。まずホームポジションと指の動かし方を知っている人から教えてもらいます。 'THIS IS A PEN.' をキーボードを見ないで打てるようになって下さい。次にこれができたら、あなたの好きな文を同じように練習して下さい。こうしているうちにキーボードの文字の配列が頭の中に入ってきます。くれぐれもAの右にSがあるなどとキーボードを見て覚えるようなことはしないで下さい。文字ではなく単語を、目ではなく指で覚えて下さい。これでもいやだという人は、READ、WRITE、FORMAT、DO、CONTINUEの5語だけでもキーボードを見ないで打てるようにして下さい（FORTRANの場合）。これだけで、目と頭の疲れがずいぶんとれます。またプログラミングにかかる時間がぐっと減り、それに伴ってエラーも減ります。初心者がコンピュータをうまく操作できるための最良の方法というのは、タイピングとエディタの使用方法をマスターすることです。

キーボードを見ないで打てるようになった人が、意識的に目をキーボードに向けると逆に時間もエラーも増えるようです。目からの情報が人間の感覚の中では、一番ウエイトを占めているという良い例でしょうか。

② エディタ

プログラミングする場合エディタというのは非常に大切です。エディタとは何か知らない人もいるでしょうが、プログラムの編集をするソフトだと思って下さい。特にフルスクリーンエディタが便利です。同じような文をコピーしたり、まとまった部分を移動したり、探したい文字列が入っている文だけを探してきたりとその機能は多才ですが基本的な操作だけを知っていても非常に便利です。

サブコマンド　：DOWN、UP、LIST

行サブコマンド　：COPY、MOVE、DELETE、INSERT（フルスクリーンのとき）

これだけで十分に能率がアップします。

「TSSによる情報処理」（藤村直美　著）のP62～P80に詳しく載っていますので、一度目を通して下さい。

なおFACOMではプログラムを入力する場合、EDITモードの他にINPUTモードというのがありますがこれは使わないで下さい。疲れるだけです。

③ オペレーティングシステム（OS）

'オペレーティングシステムとは何か'、私自身も詳しくは知りませんが、コンピュータを動かす一番基本のソフトとでも考えて下さい。 'LOGON TSS *******' と打ち込むとコンピュータがパスワードを聞いてきて、正しいパスワードを入力するとREADY状態になるというの

もＯＳの仕事ですし、 'ＬＩＳＴ データセット名' とすると、プログラムが画面にリストされるのもＯＳがあるからなのです。

ではなぜここでＯＳのことを取り上げるかというと、それはコンピュータというとすぐプログラミングということと結び付けられがちですが、そうではないということに気付いて欲しいからなのです。もちろんプログラミングというのは大きな要素です。しかし、「コンピュータを操作すること＝プログラミングではない」のです。具体的な例を上げると、ＦＯＲＴＲＡＮのプログラムのリストを出すときに、ＬＩＳＴコマンドを入力しますが、このＬＩＳＴという'コマンド'はＦＯＲＴＲＡＮの命令ではないのです。そういう'コマンド'をもつＯＳが走っているから実行されるのです。コンピュータが作動する上で一番根本にあるソフトというのはＯＳなのです。コマンドを実行するという意味でも基本的なのですが、それ以上にそのコンピュータの性格を決めてしまうのです。

皆さんが作ったプログラムはファイルという単位で区別されます。このこと自体がＯＳに関係しているのです。例えばＦＯＲＴＲＡＮが走る場合、ＦＯＲＴＲＡＮコンパイラというファイルがあなたの作ったプログラムファイルを入力し、機械語のファイルを出力します。そしてもし必要ならば、データの入ったファイルをセットして機械語のファイルにＯＳの制御が移るというふうになっているのです。ディスプレイもキーボードも一つのファイルとしてＯＳは処理しています。つまりファイルを基本単位とするようにＯＳが作られているのです。

そしてファイルには順データセットの他に、区分データセットといういくつかのファイルをまとめて一つのファイルとなっているようなものがありますが、これもＯＳがそのように作っているからなのです。

その他にもＯＳに関係することはたくさんあります。例えば、何十人もの人が一つのコンピュータを一度に使えるのもＯＳのおかげですし、他人のファイルを勝手にCOPYできないようになっているのもＯＳのためなのです。このようにいろいろとあるのですが、初心者にはこれ以上のことは必要ないし私自身分からないのでやめておきます。

ただ始めに言ったようにプログラミングとＯＳということをきちんと整理して理解して下さい。そうすることによって、ファイルの管理という考えやそれを実行するコマンド（CONDENSE、RELEASE、COPY、RENAME）の必要性が分かってもらえると思うし、コンピュータを操作する上での効果的な操作方法も分かってもらえると思います。

④ エラーについて

エラーといってもいろいろなエラーがあります。ここでは、プログラミングでのエラーについて書きたいと思います。まず大きく分けてエラーには３種類あります。

1） コンパイル中にでるエラー

2） 実行時にでるエラー

3） エラーメッセージの出ないエラー

コンパイル中にでるエラーは文法的に間違ったものです。下の例のようなエラーメッセージが出

ます。見方は　JZK***I　と出ますが、このJZKは文法的なエラーということで、***がエラー番号です。エラーは英文で出るのでどういう意味か分かりにくいと思いますから、そのときはFORTRANなら「FORTRAN77メッセージ説明書」で調べて下さい。

実行時のエラーとは0で割ったりオーバーフローしたりというようなエラーです。JZL***I と出ますが、これも「FORTRAN77メッセージ説明書」（FORTRANの場合）で調べて下さい。

たいがいのエラーはこのようなメッセージがでますが、中にはエラーメッセージが出ないエラーもあります。

この3種類のエラーの具体例を以下、説明します。

＜例1. コンパイル中に出るエラー（JZKで始まるメッセージ）＞

```
 READY
LIST LESSON1.FORT77(ER01)
 KEQ52800I  YZD2055.LESSON1.FORT77(ER01)
 00100 *      LESSON1.FORT77(ER01)
 00200 *
 00300 *                          -- 11/22/85 --
 00400 *                       MADE BY N.HIROSHIGE
 00500 *-------------------------------------------------*
 00600        REAL   BANK(3,10)
 00700 *
 00800        DTOR = 3.14159/180.0
 00900        DO 100 I=1,10
 01000           BANK(I)=SIN(I*DTOR)
 01100   100  CONTINUE
 01200 *
 01300        WRITE(6,600) ((BANK(I,J),I=1,3),J=1,10)
 01400   600  FORMAT(5X,10F7.3)
 01500 *
 01600        STP
 01700        END
 KEQ52802I  END OF DATA

 READY
RUN LESSON1.FORT77(ER01)

 FORTRAN 77 COMPILER ENTERED
 FORTRAN 77 ERROR MESSAGES: PROGRAM NAME(MAIN  ),FLAG(I),
  OPTIMIZE(2)
 JZK361I-S LNO:00001000 NAM:BANK      INVALID SUBSCRIPT
 NO.
 JZK276I-S LNO:00001300               INVALID DO VARIABLE
 JZK424I-S LNO:00001600               INVALID SYNTAX
 JZK524I-I NAM:J                      UNDEF.
 END OF COMPILATION

 READY
```

この例では 'JZK361I' と出ていますから、まずエラー番号361のコンパイルエラーということが分かります。詳しく知りたいときは「FORTRAN77メッセージ説明書」のJZK361Iのところを見て下さい。

次に 'LNO:00001000' というところからソースプログラムの行1000のところでエラーが起こっていることが、しかも 'NAM:BANK' というところからBANKという名前のもの

に関してのエラーだということが分かります。ソースリストを見てもらうと分かるように、2次元配列BANKが1次元として使われているところがエラーであることが分かります。

それから次のエラーメッセージでJZK424Iというエラーが行1600でおきています。 'STOP' を間違えています。

それから最後に'JZK524I'のエラーが出ていますがどこで起きているか書いてありません。実はこれは行1300に伴ったエラーで行1300をなおすとこちらのエラーメッセージも消えます。

この行1300のエラーのように1つのエラーで複数のエラーが出ることがあります。また、行1300のエラーメッセージの内容は 'DO変数がおかしい' という意味で '(が足りない' とは出ていませんがこのように少々的外れなメッセージを出すことがありますので、あまりこだわらないでその行をよく見てどこがエラーか考えて下さい。

この例では 'JZK****-S' と致命的なエラーが出ているので、実行を中断し 'READY' に戻ってしまいました。なおリターンコード (SEVERITY CODE) とエラーメッセージのハイフォンの後の文字との関係は次のようになっています。

I : 0 — エラーではないが注意。

W : 4 — 警告。実行は継続される。

E : 8 — 中度のエラー。修正解釈して実行は継続される。ただし10回を越えると打ち切られる。

S : 12 — 重度のエラー。実行はただちに打ち切られる。

U : 16 — 致命的なエラー。実行打切り。

<例2. 実行時のエラー (JZLで始まるメッセージ) >

```
 READY
LIST LESSON1.FORT77(ER02)
 KEQ52800I  YZD2055.LESSON1.FORT77(ER02)
 00100 *      LESSON1.FORT77(ER02)
 00200 *
 00300 *                          -- 11/22/85 --
 00400 *                       MADE BY N.HIROSHIGE
 00500 *-----------------------------------------------*
 00600        REAL   BANK(3,10)
 00700 *
 00900        DO 100 I=-10,10
 01000           BANK(1,I)=SIN(180.0/FLOAT(I))
 01100   100 CONTINUE
 01200 *
 01300        WRITE(6,600) ((BANK(I,J),I=1,3),J=1,10)
 01400   600 FORMAT(5X,10F7.3)
 01500 *
 01600        STOP
 01700        END
 KEQ52802I  END OF DATA

 READY
RUN LESSON1.FORT77(ER02)

 FORTRAN 77 COMPILER ENTERED
 END OF COMPILATION
 JZL209I-E FLOATING DIVIDE EXCEPTION IS DETECTED.
```

```
OLD PSW=078D000F800E22E0
    INSTRUCTION=7D00D22C          LOCATION=000E22DC
    GR0 =00679870 GR1 =000BC570 GR2 =000BC580
    GR3 =000E2000
    GR4 =000E2080 GR5 =000E22C0 GR6 =000ECFA0
    GR7 =000000FF
    GR8 =FFFFFF88 GR9 =00000015 GR10=0000000C
    GR11=00000000
    GR12=0000000B GR13=000E2080 GR14=400E2024
    GR15=000ECFA0
    FR0 =40C040B400EB8B00         FR2 =3F4D90DBE8919000
    FR4 =3F8E52CF40B94F00         FR6 =0400000000FFFFFF
  ERROR OCCURS AT MAIN       ISN 01000 LOC 000E22DC
  OFFSET 0002DC
  MAIN       AT LOC 400E2002 CALLED FROM (O.S)
TAKEN TO (STANDARD) CORRECTIVE ACTION, EXECUTION
CONTINUING

JZL254I-E IN SIN(X) OR COS(X), ABS(X).GE.2**18*PI(=0.82
3549E+06) (X= 0.723700515E+76)
  ERROR OCCURS AT SIN        LOC 600EF512 OFFSET ......
  SIN        AT LOC 000EF60C CALLED FROM LOC 600E239C IN
  MAIN       AT ISN 01000
  MAIN       AT LOC 400E2002 CALLED FROM (O.S)
TAKEN TO (STANDARD) CORRECTIVE ACTION, EXECUTION
CONTINUING
       0.005   0.0     0.0     0.006   0.0     0.0     0.007
       0.0     0.0     0.007
       0.0     0.0     0.007   0.0     0.0     0.007   0.0
       0.0     0.007   0.0
       0.0     0.007   0.0     0.0     0.007   0.0     0.0
       0.007   0.0     0.0
ERROR SUMMARY (FORTRAN77)
ERROR NUMBER ERROR COUNT
    209        001
    254        001
END OF GO,SEVERITY CODE=8
READY
```

この例では 'FORTRAN77 COMPILER ENTERED' と 'END OF COMPILATION' の間に何のエラーメッセージも出ていないので文法的エラーはないようです。

しかしそのあとに実行時のエラーが出ています。実行時のエラーというのはこのようにびっくりするほどの多量のメッセージが出てくるので特に初心者はめげてしまうようですが、どこを見ればいいかポイントさえ分かれば恐れることはありません。

まず 'GR0 =00679870' などの文はすべて無視して下さい。そしてそれらの下に' ERROR OCCURS AT …' という文がありますが、そこからMAINの行1000のところで起こっていることが分かります。FORTRANでのメインプログラムには 'MAIN' という名がつきます（ただし 'PROGRAM文' を使うとその名前になる）。もしサブルーチン内でエラーが起こればそのサブルーチンの名前が出ます。行1000のところをよく見るとI=0のときには、180.0/0.0となり0で割ることはできませんのでエラーになることが分かります。

次にJZL254Iのエラーメッセージが出ていますが、メッセージをよく見るとSINでエラーが起きているようです。そしてその問題のSINがあるのは行1000のようです。実はこのエラーは前のエラーに伴ったエラーです。始めのエラーのところで0で割っているので、数学的には∞です

が、計算機ではこの計算機で表わせる最大の数を代入してどうにかこうにか実行を続けたのです。しかしその数がSIN(X)での計算でその制限にひっかかったという次第です。

JZL209I-EのEと最後の 'SEVERITY CODE=8' の文より、重度のエラーであることを表わしています。このエラーは1回だけおきているので実行が継続されました。

<例3. エラーメッセージが出ないエラー>

```
 READY
LIST LESSON1.FORT77(ER03)
 KEQ52800I  YZD2055.LESSON1.FORT77(ER03)
 00100 *      LESSON1.FORT77(ER03)
 00200 *
 00300 *                          -- 11/22/85 --
 00400 *                       MADE BY N.HIROSHIGE
 00500 *-------------------------------------------------*
 00600        REAL   BANK(3,10)
 00700 *
 00800        DTOR = 3.14159/180.0
 00900        DO 100 I=-10,10
 01000           BANK(1,I)=SIN(I*DTOR)
 01100   100 CONTINUE
 01200 *
 01300        WRITE(6,600) ((BANK(I,J),I=1,3),J=1,10)
 01400   600 FORMAT(5X,10F7.3)
 01500 *
 01600        STOP
 01700        END
 KEQ52802I  END OF DATA
 READY
RUN LESSON1.FORT77(ER03)
 FORTRAN 77 COMPILER ENTERED
 END OF COMPILATION
      -0.017   0.0     0.0   -0.035   0.0     0.0    -0.052
       0.0     0.0    -0.070
       0.0     0.0    -0.087   0.0     0.0    -0.105   0.0
       0.0    -0.122   0.0
       0.0    -0.139   0.0     0.0    -0.156   0.0     0.0    -
       0.174   0.0     0.0
 END OF GO,SEVERITY CODE=00
 READY
```

この例を見てもらうと、エラーメッセージが1つもありませんし、SEVERITY CODE=00となっています。しかしソースリストを見て下さい。あきらかにエラーであることが分かるでしょう（BANKの添字が-10から始まっています）。なぜこうなるかというのは詳しくは分かりませんが、コンピュータがというよりＦＯＲＴＲＡＮ７７というコンパイラが無理矢理こじつけて実行しているのです。どうしてこんなことをするコンパイラを作るのかと皆さん思われるでしょう。これはコンパイラの融通性と関係があるのです。もしこの例でサブチェック（添字の上下限のチェック）をしたいならばプログラムの一番始めの行に次の文を入れて下さい。

```
*PROCESS DEBUG(SUBCHK)
```

ＦＯＲＴＲＡＮは高速に実行しようとして指定がない限り添字の上下限のエラーチェックはしません。SUBCHKオプションをつけると添字チェックをするため極端に遅くなります（場合によっては２０～３０倍違う）。始めはつけておいて後ではずすということをした方がよいと思います。

また普段皆さんは無意識のうちにX=2*3.14159などと書かれるでしょうが、これは厳密にいうと昔のＦＯＲＴＲＡＮではエラーなのです。2というのは整数型で3.14159というのは実数型なので、異なる型の演算は本来はエラーなのです。しかしこれでは少々不便なので、コンパイラが自動的にX=2.0*3.14159とするようにしたのです。このように融通性をコンパイラに持たせてきたために、コンパイラは少々のエラーがあってもどうにかこうにかこじつけて解釈し、実行するのです。エラーメッセージが出ないエラーというのはある点では非常に厄介な問題です。もし結果がおかしいのに、それを正常に作動して得られたものだと信じたらどうなるでしょうか。エラーメッセージが出ないエラーがあるということを知っておいて下さい。

4. 先生、センターの方に

センターでの講義の目標の１つとして 'その学科の特色を生かした情報処理教育' ということがよくいわれるようですが、私はこのことに疑問をもっています。はたしてズブの初心者が半年間でそういう講義をマスターできるでしょうか。またその目標があるために基本的なことからの学習がなおざりにされがちな感じがします。

もし先生方が「"日本の政治"という題のレポートをパプアニューギニア語で書け」という課題をだされた場合のことを考えて見ましょう。大部分の人は書けないと思いますがなぜでしょうか。

- パプアニューギニア語が分からない。
- 日本の政治についての理解がない。

の２通りの原因が考えられます。（パプアニューギニア語という言語が本当にあるのかどうか知りません。）

これと同じで初心者がプログラミングをマスターするという場合、その言語の基本的文法を学習するということと、解法を学習するということの２つの要素があると思うのです。しかし私が見るところではどうもこの２つのことが混同されているようです。基本文法を学習する途中で段々と難しい解法を必要とする例題が出されているようです。多分 'その学科の特色をいかすために' ということでしょうが、はっきりいって初心者にとっては非常にやる気を失う学習方法だと思います。まずは基本文法を一通り学習するまではあまり込み入った問題は出さない方がよいと思います。

それから言語の基本文法の学習方法として、例題を情報処理教育センターに登録するべきだと思います。生徒はタイピングがまるでダメなのですから、プログラムを全部打ち込ませるようなことは避けるべきです。そこでプログラムの例題として情報処理教育センターに登録されておく必要があるのです。ただし完全なプログラムを登録していてはあまり意味がないので、わざとエラーを入れてやり、'エラー探し' をさせるなどすれば十分な学習になると思います。１時間かかって１題を一文字一文字入力するより 'エラー探し' を６題解く方がよっぽどましだと思われます。私もそうでしたが大部分の人はＩＤをもらいたてのころは、端末に触りたくてたまらないくらいなのに、タイピングができないから疲れるし、本に載っている例題を必死に入力してもエラーがでるのでめげてしまうと

いうパターンでだんだんと離れていきます。もしＩＤをもらい立ての学習意欲旺盛のときに、段階的に学習できる例題、エラーを探すクイズみたいな例題などがセンターに登録されていたならば、もっともっと効果が上がると思います。

また初心者のプログラムをみると分かりますが、大部分の人がコメント行の空白行、段付け等をしないので、ものすごく見にくくなっています。そこで見やすく書いたプログラムを例題として入れておけば、見よう見まねでだいぶんきれいなプログラムを書くようになるのではないかと思います。プログラムの書き方というのは１行に一つの文などの２、３の決まりがあるだけであとは何の制限もありません。このことは一見気楽な感じがするのですが、長いプログラムになるような場合とか、他人のプログラムを見るような場合とかになってくると、このことがいかに不便なものか皆さん痛感されていることと思います。

そこでそういう例題を登録してもらいたいのですが、センターの人も先生方も忙しいので簡単にはいかないでしょう。そこで一つ提案ですが、講義される先生方はその学期に一問だけ例題をセンターに提出していくというのはどうでしょう。どういう例題でもいいから、問題とそのソースプログラムを出してもらい、センターのライブラリに登録していけば、３、４年たてば立派なものになるはずです。

プログラムを組む場合よくアルゴリズムとか流れ図とかを書かせる先生がいらっしゃるようですが、こういうことをしても学生の役には立っていないような気がするのですが実際のところどうなのでしょうか。そもそも初心者にとって流れ図を書くというのはプログラム以上に難しいようです。その証拠に流れ図も必要とするレポートが出題された場合大部分の学生はプログラムを完成させて、それからそのプログラムを見て流れ図を書いているようです。もしくは流れ図を先に書いてもそれは全然参考にせずプログラムを考えるというような状態です。だいたい流れ図を書かせるぐらいならサブルーチンの使い方を十分に教え　”プログラムの内容をサブルーチンごとによくまとめメインルーチン中にはサブルーチンをＣＡＬＬするだけのプログラムを作成せよ”　という条件でレポートを出した方がずっとましだと思います。

それからコンピュータは道具であるからもっともっと使い方を勉強させるべきです。つまりプログラミングだけに集中するよりも、ＯＳ、エディタ、タイピング、エラーへの対処法、ファイルの管理等についての説明を十分にすべきだと思います。もし微分方程式のプログラムは学問的に価値があるが機械の使用法、操作法などというものには学問的価値がないとでも考えておられるのでしたら大間違いです。プログラムはコンピュータマシーンがあって初めて実行されるのですから。またコンピュータは道具であるといってもテレビとかラジオとかのように試行錯誤で使用法を身につけられるような道具ではありません。少々時間をくうかもしれませんが　'急がばまわれ'　というではありませんか、じっくり説明された方がよいでしょう。

5. コンピュータにできること、できないこと

「コンピュータに何ができるのか。また何ができないのか。」を判断する能力を身につけることが重要だと書きましたが、では具体的にいうとどういうことなのだろうかという疑問がわき、できればそのエッセンスだけを学習したいと思われるでしょう。しかしこれはあまりにも虫のいい話です。はっきりいってしまえば、皆さんが実際にコンピュータを操作してもらうしかありません。といってここで話をきってしまうのも寂しいので、私が経験的に「これがコンピュータか。」と思ったプログラムの中から2つの例を出したいと思います。

＜例1（演算精度について）＞

```
00100 *      LESSON1.FORT77(EX01)
00200 *
00300 *                              -- 11/28/85 --
00400 *                           MADE BY N.HIROSHIGE
00500 *-------------------------------------------------*
00600        X = 1.0/3.0*3.0
00700        Y = 1.0
00800 *
00900        IF (X .EQ. Y) THEN
01000            WRITE(6,600) X,Y
01100    600     FORMAT(5X,'X = Y ', -
01200                   /8X,'X=',F12.8,3X,'Y=',F12.8)
01300        ELSE
01400            WRITE(6,610) X,Y
01500    610     FORMAT(5X,'X <> Y ', -
01600                   /8X,'X=',F12.8,3X,'Y=',F12.8)
01700        END IF
01800 *
01900        STOP
02000        END

FORTRAN 77 COMPILER ENTERED
END OF COMPILATION
     X <> Y
       X=   0.99999994    Y=   1.00000000
END OF GO,SEVERITY CODE=00
```

このプログラムは簡単にいうと1.0÷3.0×3.0（1を3で割り3をかける）は1.0かという問題です。数学的にいうと当然のことですが、計算機では当然ではないのです。非常に簡単なことのようでこれくらいどうってことないのではとも思うのですが、コンピュータで数値計算を行なう場合には機械として本質的なことで、全てのコンピュータで避けられないことです。

もしあなたが作るプログラムの中で、X=Yであるかどうかを調べたいときに

```
IF (X .EQ. Y) THEN
```

というふうにするならばX≠Yとなることが目に見えています。あなたは数学的誤差以外に機械であるからこそ生じるこの誤差を考慮してプログラムを作る必要があるのです。この場合ですと、

```
EPS = 0.00001
IF(ABS(X-Y) .LE. EPS) THEN
```

とすることが多いようです。

それからこの１８行からなるプログラムを見てどういう印象をもたれるでしょうか。７行がコメント文で正味１１行のプログラムですがこの中で計算の中心となっているのは行番号が600と700の２行だけです。何かあっけない気がしますがこれから皆さんが作るプログラムの中で核となる部分は行数になおすとプログラム全体の中のほんの一部分にしかならないでしょう。

＜例２＞

```
00100 *      LESSON1.FORT77(EX02)
00200 *
00300 *                          -- 11/28/85 --
00400 *                       MADE BY N.HIROSHIGE
00500 *-------------------------------------------*
00600        REAL   IP
00700        COMPLEX Z(2)
00800 *--------------------*
00900        A = 1.0
01000        B = 2.0
01100        C = -2.0
01200 *--------------------*
01300        D = B*B - 4.0*A*C
01400        IF (D .GE. 0.0) THEN
01500             X1 = (-B+SQRT(D))/(2.0*A)
01600             X2 = (-B-SQRT(D))/(2.0*A)
01700             WRITE(6,600) X1,X2
01800        ELSE
01900             RP = (-B)/(2.0*A)
02000             IP = SQRT(-D)/(2.0*A)
02100             WRITE(6,610) RP,IP
02200        END IF
02300 *-----------< USE SSL 2 SUBROUTINE>-----------*
02400        CALL RQDR(A,B,C,Z,ILL)
02500        WRITE(6,*) Z(1),Z(2)
02600 *
02700        STOP
02800   600  FORMAT(/,5X,'X1=',F14.9, -
02900               /,5X,'X2=',F14.9/)
03000   610  FORMAT(/,5X,'REAL PART = ',F14.9,/5X,  -
03100               /,5X,'IMA. PART = ',F14.9/)
03200        END

FORTRAN 77 COMPILER ENTERED
END OF COMPILATION

     X1=    0.732050896
     X2=   -2.73205090

(-2.73205090,0.0) (0.732050776,0.0)
END OF GO,SEVERITY CODE=00
```

さてこのプログラムは何を解くプログラムでしょうか。 '二次方程式の解を求める' プログラムですか。厳密に言うと違います。もしウソだと思うならばこのプログラムで解いた答えを中学校の先生に見せて下さい。$x^2 + 2x - 2 = 0$の解が0.732050896と-2.73205090でしょうか。正解は$x = -1 \pm 2\sqrt{3}$です。

このことからわかるのはコンピュータができるのは "数値計算" だと言うことです。人口知能というものができるとどうなるかわかりませんが少なくとも現在のコンピュータができることは " 数値計算" だけです。もし ' $x = -1 \pm 2\sqrt{3}$ ' という解を出すプログラムが欲しければ作れ

ないことはありませんが、そのためには約数を求め約分したりルートの中から出せるものだけを出すなりするような手続きを自分で考える必要があります。

ワープロだってそうです。コンピュータが漢字を理解するわけはないのです。漢字にコードを対応させ、そのコードをコンピュータの内部でいろいろ計算しているだけなのです。

それからもう一つこの例題で言いたいことはコンピュータのアルゴリズムは数学的なアルゴリズムとは異なるということです。私がこの例題を6年前はじめて学習したときに「コンピュータはサギだ。」と思いました。なぜなら二次方程式の解の公式を求めるのは人間だからです。コンピュータが二次方程式を解くといってもそれは人間が解いた式に数値を入れて計算するだけなのです。

このことはとても大切です。コンピュータで問題が解けるかどうかということはそれ以前にその問題をプログラムできるかということによるのです。もし二次方程式の解の公式が数学的に求められないならばコンピュータは二次方程式が解けないことになるのです（近似的な解き方はできます）。実際にコンピュータが小説を書けないのは、小説を書くということを定式化できないからです。コンピュータで大切なのはハードよりソフトだとよくいわれますが、このソフトということをプログラミングにしぼって考えた場合、数学的なアルゴリズムの定式化がいかに大切であるかということを覚えておいて下さい。

6. 最後に

今までプログラムの相談を受けた学生の中に「ボクはコンピュータを信用しません。」という人が何人かいました。 "コンピュータは万能ではない" と言う意味で言っているのであればなかなかいい心がけなのですが、そういう人に限ってとんでもないプログラムを書いているのです。

コンピュータは人間が作ったものですからそれを操作するためには約束ごと（操作方法）があるのです。それから、それら約束ごとを守った上で操作してもコンピュータにはできないことがたくさんあります。この2つのことを理解してもらえればコンピュータはあなたの強力な道具となるはずです。

本当はもう少し書きたいことがあるのですが筆が遅いもので、これでも原稿の締め切りをのばしてもらった結果どうにか書けたという具合でした。そのため内容がごちゃごちゃとなってしまいましたし、また少々横着なことも書きましたが、もし皆さんがコンピュータを学習する上で何かのきっかけになったら最高だなと思っています。大学にはいって初めてこのような文章を書く機会を与えて下さった情報処理教育センターの方々に感謝の意を表してこのへんでおわりたいとおもいます。