福山大学工学部紀要
第11号　1989年3月

建築教育におけるパーソナル・コンピューター利用の可能性について(5)（入力形式における標準形について）

谷口　興紀*・横井　友幸*

A Note on CAI in the Architectural Education (5) (Standardization of Input Formats)

Okinori Taniguchi and Tomoyuki Yokoi

ABSTRACT

We discuss the standardization of input formats in CAI program. In CAI input behavior is a bottle neck for students and in order to break through this, CAI program must be checked by following points,

(1) adjustment of hardware environment
(2) one character input without hitting carriage return key
(3) guide message with blinking : ' hit the return key at the end of input ! '
(4) controlled input guide message as prompt
(5) correction of input error
(6) undo-passing of input command, if necessary

If these conditions are satisfied, the success probability of CAI program is high.

キーワード：建築教育、パソコン、ＣＡＩ、入力形式、成功確率、標準化

1. はじめに

前稿（参考文献１）で述べた開発・試用経験から、ＣＡＩシステム開発のポイントの一つは、ユーザー（学生）との接点である。この点をクリアーして置かないと、ＣＡＩの成功は、望めないと感じる。本稿では、この点に焦点を当てて論じる。２、３では、入力形式という点から述べ、４において、入力形式の標準化について述べる。なお、ここで述べられる事柄が展開するハードウェア環境としては、本学７号館のパソコン室（NEC PC-9801 M２／30台、同VX21／30台、プリンター30台＝パソコン２台に１台）の状態を想定している。

2. 入力形式等の問題点

コンピューターとの応答は、キーボードからの入力によるとして、学習現場で、どのようなことが、実際に起こり得るかを見てみよう。

先ず、＜キーボード入力＞と＜入力コンテクスト＞とに分けて考えることが有効であろう。＜キーボード入力＞とは、キーボードを押すことによって、目的の応答をすることである。＜入力コンテクスト＞とは、学生が応答する場面を取り巻く環境条件、言い換えれば、時空的状況と学生の内的状況の二つを含めた状況である。この状況を状態空間的に述べる。

状態Ｓ１　CAPSが押し下げてない（ある）
状態Ｓ２　カナが押し下げてない（ある）
状態Ｓ３　プリンターのスイッチが点灯している（ない）

*建築学科

状態Ｓ４　プリンターが切り替わっている（ない）
状態Ｓ５　リターンキーを押す（押さない）
状態Ｓ６　ピリオッドを付ける（ない）
状態Ｓ７　プログラム上は正しいが、ユーザーの意図に反していない（いる）

ＣＡＩプログラムを実行したとき、これらの状態のどれかが、継起的に生起している。もし、プログラムを作るときにこれらの状態のどれが生じても、進行するように手当されていなければ、成功しないことになる。その手当が、教師による教示ということだと学生は、先生がいなければそのプログラムを使えないことになる。これを、ここではスタンドアロンではないと呼ぶ。

各状態に１／２の確率を割り当てると、（１／２）の７乗で、成功の確率は、Ｐsuc.＝0.0008であり、全く白紙の1,000人の学生がおれば、多目に見積もって、そのうち１人だけが、そのプログラムを使いこなすことが出来るが、999人は、「先生！　動かない。」と訴えて来ることになる。ユーザーには、パソコンのことを良く知っている者から、今まで触ったこともないという者まで含まれている。その間に、生半可に知っている者もいて、適切でない操作を、良く知らない友達に教えることもある。すなわち、状態８として、適切でないキー(stopキーやリセットキー等）を押すことを加えねばならないが、状態１から状態７までが充分に手当されておれば、状態８の生起の確率は、充分小さいと考えられるのでここでは、無視する。しかし、使用者が千人を越えるような大量の場合には、無視できなくなり、何等かの手当てが必要になる。（例えば、本学の付属図書館の文献検索システムでは、検索に不必要なキーは、アルミ板でカバーされていた。）

ＣＡＩクラスの自然状態は、これらの組み合わせ事象が起こっていることになり、良いＣＡＩプログラムの開発方向は、いかにそれらの生起をコントロールし、成功の確率を高めるかであろう。

状態１から７は、非常に初歩的なことで、ちょっとパソコンに慣れた人にとっては、殆ど問題にならないことである。したがって、最初の時間に充分教示をして置けば済むことであるとも考えられる。しかし、このような考え方は、学生を＜未熟なオペレーター＞と考え、やがては、＜熟練したオペレーター＞となるように訓練することを意味する。しかし、これは、学生に教育本来の目標と独立な、余分の負荷をかけることになる。もし、訓練の必要のないプログラムがあれば、誰でもそちらの方を選ぶであろう。筆者のクラス試用の経験でも、プログラムの使い方を習得させるのに時間を取られて、本来の教育目標を達成しない内に時間が尽きてしまい、教育効果が上がるかどうかを検討するとこまで行かなかった。建築的発想として、器械に人間を合わすのではなく、人間に器械を合わせるという方を選びたい。

このような観点から、前稿に挙げた11本のプログラムをチェックして見ると表－１のようになる。成功確率がどれぐらいあれば、クラス（100人）試用に耐えるだろうか。「先生！　動かない。」を、５～７人を許容範囲とすると93～95%ということになり、表－１では、どれも基準に達していないことになる。

表－１　既存プログラムの使用成功確率チェック表

prog_no.	1	2	3	4	5	6	7	8	9	10	11
状態Ｓ１	1.0	1.0	1.0	0.5	0.5	1.0	1.0	1.0	1.0	0.5	0.5
状態Ｓ２	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	1.0	0.5	0.5
状態Ｓ３	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
状態Ｓ４	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
状態Ｓ５	0.5	0.5	0.5	0.5	0.5	0.7	0.5	0.5	0.5	0.9	0.9
状態Ｓ６	1.0	1.0	1.0	0.5	1.0	1.0	1.0	1.0	1.0	1.0	1.0
状態Ｓ７	0.5	0.5	0.9	0.5	0.7	0.5	0.7	0.9	1.0	0.7	0.5
成功確率	0.1	0.1	0.2	0.0	0.1	0.2	0.2	0.2	0.5	0.2	0.1

(注) 横の番号は前稿表－３で示された既開発プログラムの番号である。
(注) 小数点以下２桁目四捨五入。

3. ＜ユーザー＞一般から見た入力形式

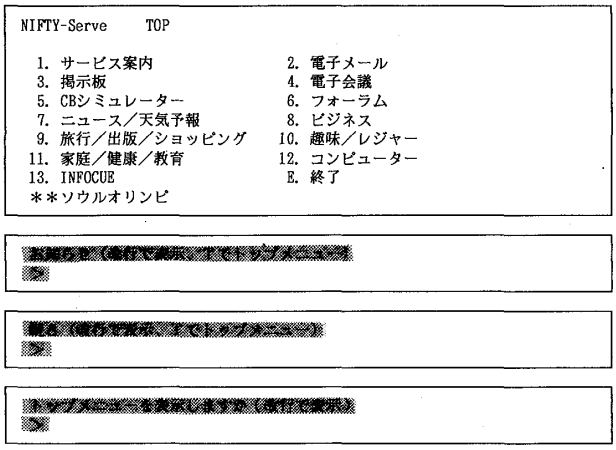
教室での多人数による試用のデータが不十分なので、別の観点――パソコン（ワープロ）通信の観点――から上述の事柄を検討して見たい。

パソコン通信の世界は、一般の人が、＜オペレーター＞としてではなく＜ユーザー＞としてパソコン(ワープロ)に接している世界である。現在、わが国では、13万人（1988年現在）の人が使っている。そこには、教師なしで、興味をもつ人が、スタンドアロンでワープロなり、パソコンを媒介して通信を楽しんでいる。ユーザーの意欲から考えて、そこで生じるトラブルは、クラス授業で生じるトラブルよりも少ないであろう。逆に、もし、そこでミスフィットが生じておれば、必ず、同様のミスフィットが教室内で生じると言えるであろう。

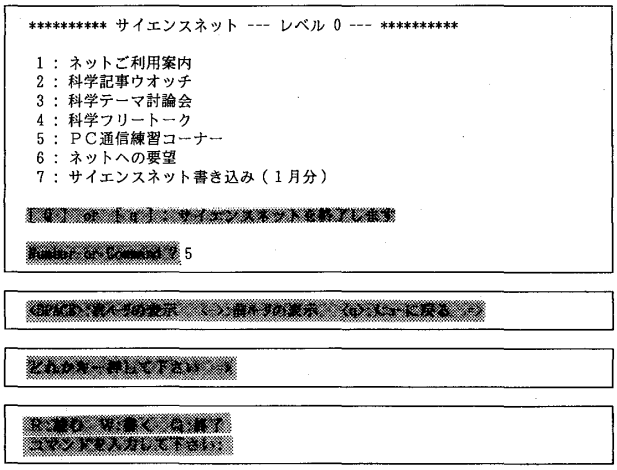
また、そこでの入力形式は、現実に、何千人という人によって使い込まれ、現実のコンテクストにおけるテスト結果を十分に反映していると考えて良いであろう。そこで、筆者は、二三のネットワークに実際にアクセスしてみた。ユーザー識別番号、パスワード入力後のトップメニュー画面と入力を求めてくる部分を抜き出したものを図－１、２、３、４に示す。但し、ここでは、ユーザ

ーに入力を求める形式に注目するので、その部分に筆者が網かけ加工した。実際の画面では、入力を求める部分以下は、空白であり、一目瞭然である。

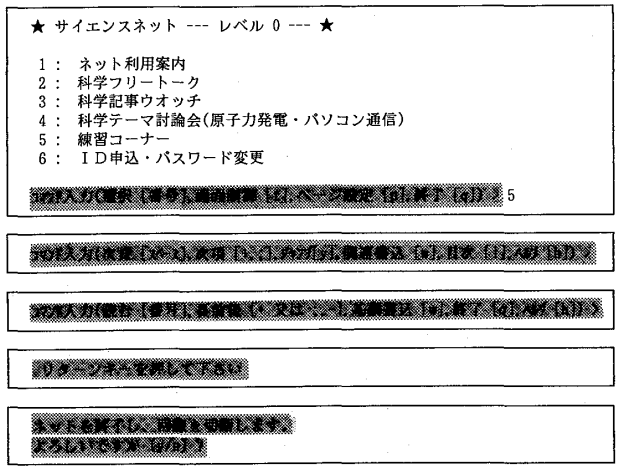
図－1　NIFTY



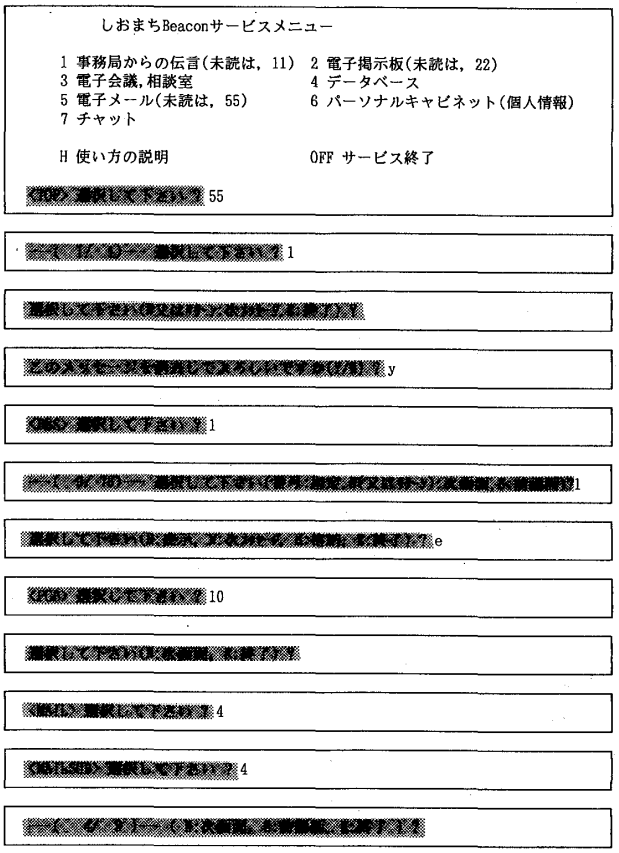
図－2　サイエンスネット1



図－3　サイエンスネット2



図－4　しおまちビーコン



これらのネットワークについて、上記の状態がどうなっているかを表－2に示した。

表－2　パソコン通信ネットの使用成功確率チェック表

各ﾊﾟｿｺﾝﾈｯﾄ	NIFTY	ｻｲｴﾝｽﾈｯﾄ1	ｻｲｴﾝｽﾈｯﾄ2	しおまちﾋﾞｰｺﾝ	
状態Ｓ１	1.0	1.0	1.0	1.0	
状態Ｓ２	0.5	0.5	0.5	0.5	
状態Ｓ３	1.0	1.0	1.0	1.0	
状態Ｓ４	1.0	1.0	1.0	1.0	
状態Ｓ５	0.5	0.5	0.7	0.5	
状態Ｓ６	1.0	1.0	1.0	1.0	
状態Ｓ７	1.0	1.0	1.0	1.0	
成功確率	0.3	0.3	0.4	0.3	

アクセスしてみた感じでは、NIFTYが完成度が高いようである。筆者が良くアクセスする「しおまちビーコン」は、NIFTYと同じメーカーのものである。サイエンスネットは、開局してから、まだ日が浅く、システムが試用段階であり、改良が加えられつつある。サイエンスネット1は、1988年7月、サイエンスネット2は、1989年2月の時点のものであり、一般の人々が使用することにより、半年程の間にどのように修正されたかが伺えて興味深い。

NIFTYに比べて、サイエンスネットで目立つことは英文キーに精通していることを前提にしているように思える点である。日本人一般は、英文キー配列も、かなキー配列にも不慣れである。例えば、「きゅう」を押して下さいと口頭で言うと、「Q」を押すことは、まず期待できない。「9」を殆どの人が押すであろう。また、「いー」を押して下さいと言うと、「E」を押すことは期待できなくて、「イ」または「い」が押されるであろう。ましてや、「;」等は、"セミコロン"と読むことも出来ないし、キーボード上の位置を知っていることはさらさら期待できない。一方、NIFTYは、数字を主に使うことにより、これらの点を回避している。また、初心者用のルートとコマンドを使用するベテラン用のバイパスルートとの二つが重ね合わされている。後者は、画面に直接表示されないがマニュアルまたは、ヘルプ機能で画面にあらわれる。

この工夫は、市販されている高機能なエディタ・ソフト等の使用方法が参考になろう。典型的なコマンド入力法として、

1. 同種のコマンド群の割り当てられたファンクションキーを押して、ポップアップメニューを表示させメニュー内のコマンドを
   a) カーソルキーで選びリターンキーを押して入力する。
   b) 表示されたコマンドに付けられた番号または記号を押して入力する。
2. CTRLキーを押しながらアルファベットキーを押して、各アルファベット記号に割り当てられたコマンドを入力する。

の2way（または3way）の仕方が挙げられる。この方法の利点は、a)はキー操作回数は多いが視覚的にコマンドを決めるので初心者に判り易く、b)になるとやや視覚的であるがコマンド選択までのキー操作回数がa）より少なく中級者向きとなり、2.はキーとコマンドの割り当てを覚えなければならないがコマンド選択が1回のキー操作ですみベテラン向きと言える様に使用経験に従って操作が換えられると言うことである。
最近の市販ソフトは殆どがこの様な入力方法を採っている。

4. 入力形式の標準化

既開発プログラムの入力形式を整理して見ると、計画系では、製図図形入力が主となり、構造系では、演習問題解答入力が主となる。前者はコマンド入力形式となり、後者は数値入力形式である。これらの入力をそのコンテクスト抜きで見ると、

1　画面上の指示文に従う一文字入力
2　画面上の指示文に従う文字列入力

の2つに分けられる。ここで、文字列とは広義に解釈して、数字列としての数値、ファイル名等の文字列、文章を表す文字列等を含む。
2節の状態空間は、

S1、2、3、4：ハードウェアの状態
（S1、2；キーボード、S3、4；プリンター）
S5：入力方法
S6：データの形
S7：入力ミス／エラー

に関わることと分類できる。S1～5が成功すれば物理的入力に成功したことになるが、S6、7をパスしなければ論理的に成功したことにならず、入力は失敗に終わる。とは言っても、まず物理的入力が成功しなければ論理的入力の成功は望むべくもないので、第一に物理的入力が成功すべき標準形を確立しよう。
S1～4をパスするにはハードウェア環境を調べ整える処理をプログラムすればよい。とりたててそれを行わずに、システム起動時のメッセージ表示から最初の処理に移るとき、例えば、Aのキーを押す様に指示し、入力されるコードが"A"か"a"か"チ"かによってキーボード状態が判定できニュートラルな状態を設定できる。更に、プリンターを使う場合はプリンターの状態を読み取って指示を表示するようにすればいい。これを標準化I－1：ハード環境調整とする。

次にS5であるが、前述の1ではリターンキー押下を必要とせず入力することが可能なので、一文字入力では常にリターンキー不要の形式にする。これを標準化I－2：一文字入力とする。

2の入力形では2文字以上の文字列を入力するので入力完了をコンピューターに知らせるリターンキー押下がどうしても必要になる。これをパスするには、

a) 簡単な方法として、入力時に入力欄近くに「入力終了時にリターンキーを押す」旨を表示して注意を促す。できればこのガイド欄はウインド形式で入力し終えれば消え、元の画面表示に復元される形式とする。
b) 入力時に打鍵がとだえて一定時間入力がないと入力欄近くに「入力終了時にリターンキーを押す」旨を表示して入力を促す、ような専用の入力処理

ルーチンを作成する。

等の方法が考えられる。b)(参考文献2)が望ましいが、a)でもガイド表示をブリンクさせて注意を引き付けるようにすれば、効果を期待できよう。これを標準化Ⅰ－3：リターンキー警告とする。これは初心者対策なので入力方法が理解できれば警告表示しなく出来るように考えておくべきである。

次に論理的入力が成功するための手だてを述べる。このためには、S6、S7をパスすることを考えればよい。S6の対象になるのは文字列であり、個々のデータによって異なる形（文字の並びや位置関係等）の適合性を一般的に判定するのは不可能であり、この場合、標準化Ⅰ－3 a)と同じ様に、入力時にデータの形式をガイド表示することが考えられる。個々のデータ形入力指示解説が異なるので、入力時にガイド文をプロンプトとして与える形式となろう。これを標準化Ⅱ－1：データ入力ガイド表示とする。このような工夫は、パソコン通信でも見られる（図－1～4）。

S7に対しては、データ訂正機能を備えることで対処する。訂正処理は入力するデータの性質によって様々に考えられるが、標準化の方向として単純で一般的なものが望ましい。一案として「前回の入力データを訂正できる機能を備えた入力」が考えられる（参考文献3）。このような処理は手続き型言語では考えにくいことであるが、prolog 言語では repeat 述語を参考にして素直に実現できる。これを参考にすると手続き型言語でも、データ入力の後に入力されたものがデータか訂正指示かを判定して訂正の場合一回前の入力へジャンプさせることで、前回訂正機能付入力処理を実現できる。（入力命令の前にラベルを付け、後に訂正判断と前回入力ジャンプ処理を加えるというプログラミング時の煩わしさは我慢しなければならないが、却って入力の整理が出来ていいかも知れない。）訂正指示キーは文字列を割り当てることの出来るファンクションキーに統一しておくのがよい。前データ訂正を続ければ数回前の入力データも訂正できる。手続き型言語では訂正時に前回入力から今回の入力の間で値が変更され、且その前に値が参照されるような変数の内容を元に戻すことを忘れてはならない。また、prolog 言語ではカット述語の使用に注意を払わねばならない。不用意に用いると入力の選択子がカットされて訂正できず訂正機能付入力述語が失敗してしまうことになる。このように使用時に注意すべき点はあるが、単純かつ一般性があるのでS7対策の標準化案としよう。これを標準化Ⅱ－2：前データ訂正機能付入力とする。勿論、専用のデータ訂正機能を備えてS7をパスするものはそれでよい。

一文字入力時の訂正は、この案によるのもよいが、むしろ入力時のコンテクストの中で考慮したほうがよい。入力時のコンテクストを考えた場合のデータ扱いは

1）他のデータとの関連性のないもの：独立データ
2）他のデータと関連し合い、それ等一組で意味を持つもの：組データ

に分けられ、1）の独立データは演習問題解答の入力によく見られ、2）の組データはコマンド入力形式に現れる。個々のデータの入力に対しては既に述べた形式で十分であるが、組データとしてコマンド入力形式を考えると次の訂正法も実現した方がよい。コマンド入力形式では、メニューに示されるコマンドを入力して（大体一文字入力である）、選択された処理に必要なデータ入力へ進むが、コマンド誤選択時に、前述の前データ訂正によって1ステップ戻るのではなく、そのコマンドをパス(何もせず終了）して、次のコマンド選択へ進むと言う形式にしたほうが、画面表示などの処理上都合がよい。従って、これを標準化Ⅱ－3：コマンドパス機能とする。

以上を纏めると、

標準化Ⅰ－1：ハード環境調整
標準化Ⅰ－2：一文字入力
標準化Ⅰ－3：リターンキー警告
標準化Ⅱ－1：データ入力ガイド表示
標準化Ⅱ－2：前データ訂正機能付入力
標準化Ⅱ－3：コマンドパス機能

である。これらの実現は既に述べた様にプログラムすればよく、その成功確率は当然1になる、筈である。が、各標準化での対策が、「入力がうまく出来るようにその仕方を説明表示して、その指示を理解して初めて入力が成功する」と言うものであるから、実際に1になるか否かは、画面に現れた指示ガイドを教師が意図したように学生が読み取れるか否かにかかってくる。この値を1にするには、標準型の入力述語の試用を重ね、入力指示ガイドが教師の意図どおりに作用するかを調べ、作用するような指示ガイドの作り方を確立しなければならない。リスト1に prolog で書かれた標準型の訂正機能付入力述語の例とその使用法を示し、試用テスト時のパソコン室での一連の必要な処理を参考までに記す。

(1) パソコン室へ学生を入れる。
(2) 操作方法の簡単な指示を与える。
　　パソコンの電源スイッチの位置、
　　ディスクドライブとロックレバーの操作法、
　　配布するディスケットの取り扱い方、
　　システム起動方法等。
(3) 学習する事柄を説明する。
(4) システムディスクと学習報告書用紙を配布して、ＣＡＩ学習を始めさせる。
(5) ディスケットをパソコンのディスクドライブ上段

にセットして電源を入れ、20秒以内にドライブをロックする。

(6) ＣＡＩシステム自動起動。

(6−1) ＣＡＩシステムタイトル表示。

(6−2) ハード環境調整　　<=

(6−3) キーボード説明（パス可能）　<=

(6−4) 学生番号、氏名の入力　　<===

(6−5) （パソコンからの応答）

(6−6) （キーボード説明の再学習（の有無）　<=）

(6−7) （ＣＡＩシステムメニュー／表示選択　<=）

(6−8) ＣＡＩプログラム

開　始　　<=

.....　　and／or

終　了　　<===

(6−9) （メニューに戻る）

(6−A) （学習報告書入力　<=，<===）

(6−B) ＣＡＩシステム終了

(7) ディスクドライブのランプ消灯を確かめて、ドライブのロックをはずし、電源を切り、ディスケットを取り出して、しまう。

(8) ディスケットを回収して、学習報告書を提出させＣＡＩ学習を終了する。

(1)−(4)、(8)が教師側の指示、(5)−(7)が学生側の学習であり、(6)がＣＡＩ部分、学生とプログラムによって指示されたパソコンとの応答である。<=と<===は一文字入力と文字列入力があることを示す。（ ）内は将来的な処理を記した。(6−3)／(6−5) では、ＣＡＩはキー操作が基本になるのであるから、グラフィックでキーボードを描き、操作に必要なキーの位置、名称等を視覚的に教えるプログラムを作成することにした。初心者向けにキーの図示を伴う入力を用意して、操作に関しては、質問レスを期待したい。

さて、次に考えるべき標準化は、如何に入力エラーを起こさないようにするか、如何に使い易くするか等に関することであるが、それは、現段階の標準型入力の試用結果を踏まえて論じるべきであり、ここでは３節での言及を参考にするに留める。

リスト１の標準入力形はArity Prologで記述されたが、手続き型言語で著す場合には、若干の言語間の整合性を考える必要があろう。特にハードウェアに関連するところでそれが生じる。

入力につまづくことなくＣＡＩ学習が進められるようになると、その学習効果の検証が問題として現れて来る。そのためには、ＣＡＩと一般講義等との関係を明らかにしなければならず、更に進んで、カリキュラム全体の再編成をも検討するようになろう。

これらを考えて行く為に、クラス規模での試用をしな

```
/*-------------------------------------------------------------------*/
/*    標準化案述語                          Arity Prolog Version     */
/*        キーボード／プリンタ環境調整述語、                          */
/*              前データ訂正機能付入力述語について                   */
/*      (入力ガイド・メッセージ表示機能、                            */
/*       リターンキー警告機能、入力データ記録機能付加述語を含む)     */
/*-------------------------------------------------------------------*/
環境調整(Prn):-    /* キーボード，プリンタ環境調整：標準化Ｉ-1 */
    repeat,
    write($ A のキーを押して下さい。$),
    get0_noecho(A),nl,nl,                    /* 標準化Ｉ-2：一文字入力 */
    case([[A]=:="A" -> (write($[CAPS]キーが押されています。$),
                        write($押して、戻して下さい。$)),
          [A]=:="a" ->  P=true,
          [A]=:="ﾁ" -> (write($[カナ]キーが押されています。$),
                        write($押して、戻して下さい。$),nl,nl,
                        write($出来たら、もう一度　$),
                        fail)
                       (write($A のキーですよ。左側の[CAPS]キーの隣です。$),
                        fail)
    ]),!,
    ifthen(var(P),(
           nl,nl,write($ 出来たら、どれか、キーを押して下さい。$),
           get0_noecho(_),nl,nl
    )),
    ifthen(Prn==ﾌﾟﾘﾝﾀ,ﾌﾟﾘﾝﾀ調査),
    nl,write($ ハードウエア環境調整終了。$),nl,!.
ﾌﾟﾘﾝﾀ調査:-
    ctr_set(0,0),
    repeat,
    ifthenelse(ﾌﾟﾘﾝﾀ準備,
    %%then
          ifthenelse(ctr_is(0,0),
          %%then
               (write($ プリンタの電源スイッチオンを確認して下さい。$),nl,
                write($ 電源が入っていなければ、付けて下さい。$)),
          %%else
                P=pas
          ),
    %%else
         (write($ プリンタのSELECTスイッチがオンになっていません。$),nl,
          write($ オンにして下さい。$),P=sel )
    ),
    ctr_inc(0,C),nl,nl,
    ifthen(P¥==pas,(
           write($ 出来たら、どれか、キーを押して下さい。$),
               get0_noecho(_),nl)
    ),
    ifthen((C=:=0 ; P==sel), fail).
ﾌﾟﾘﾝﾀ準備:- in(66,X),(X >> 2) /¥ 1 =¥= 0.
```

リスト１－ａ：キーボード／プリンター環境調整述語

```
/*-------------------------------------------------------*/
/*    入力ガイド表示機能，前データ訂正機能付入力述語     */
/*     (リターンキー警告付属)              標準化Ⅱ-2     */
/*-------------------------------------------------------*/
訂正入力(Prompt_p,Guide,Data):-
    [! Prompt_p,ガイド表(Guide,G),ガイド付行入力(G,Data) !],
    (Data==訂正,!,fail;true).
訂正入力(Prompt_p,Guide,Data):- 訂正入力(Prompt_p,Guide,Data).

訂正入力0(Prompt_p,Guide,Data):-訂正入力(Prompt_p,Guide,Data);
    (write($最初は訂正出来ません。$),nl,訂正入力0(Prompt_p,Guide,Data)).

訂正入力準備:- abolish(行窓控/5),abolish(行窓控表/9),d_setkey(10,$訂正^M$),
    開行窓(1,1),開行窓(2,1).
訂正入力終了:- 閉行窓(1),閉行窓(2),ctr_set(30,0),d_setkey(10,$ansi_ed.^M$).

/*-------------------------------------------------------*/
/*      前データ訂正機能付入力述語の使用方法            */
/*-------------------------------------------------------*
(1): 訂正入力を使用する前で、述語、訂正入力準備を実行する。
(2): 訂正入力の最初の使用する節では、述語、訂正入力_0/3を用いる。
(3): 次からの使用には、述語、訂正入力/3を用いる。
(4): 訂正入力を使用した最後で、述語、訂正入力終了を実行する。
(5): 入力ガイドを表示させるには、述語、ガイド表/2（第一引数：ガイド名／アトム、第二引数：ガイド文表示行番号とガイド文のリスト、第二引数を変数にすると入力ガイドを表示しない。また、ガイド文表示行番号を変数にすると、表示行は入力位置の２行上となる。）を作る。
(6): 述語、訂正入力/3　の引数は、順にプロンプト述語（入力の前に実行する述語でwrite($aa:$)　の様なもの）、入力のガイド表のガイド名（その名前のガイド文を表示する）と入力されるデータである。
(7): カウンタ述語の３０番が１のとき、リターンキー警告を表示し、１以外のときは表示しない。
(8): ファンクションキーf-10　に訂正時の指示データ(訂正)を割り付ける。
例：
    ..............,
    訂正入力準備,
      訂正入力_0(write($start  in :$),ini,D1),
      writef( / & $first input finished$ & $  , D1=$ & D1 & / ),
      訂正入力(write($second in :$),d1 ,D2),
      writef( / & $second input finished$ & $ , D2=$ & D2 & / ),
      訂正入力(write($third  in :$),d2 ,D3),
      writef( / & $third input finished$ & $  , D3=$ & D3 & / ),
      ..............,
    訂正入力終了,
    ..............
ガイド表(ini,
        [_,$入力ガイド：最後に"."を付けて下さい。 $]).
ガイド表(d1,
        [_,$入力ガイド：ファイル名を入力します。６文字までです。 $]).
ガイド表(d2,
        [_,$入力ガイド：文章で入力します。かな漢字変換の方法は[HELP]を
            押して下さい。$]).
***/
```

リスト１－ｂ：前データ訂正機能付入力述語と使用法

がら、ＣＡＩ学習の応答データを蓄え分析していく必要がある。そのため、学生の応答と教師の応答とのログファイルを取ることが良かろう。

学生の入力データの記録と学習再現は入力部にフィルター的処理を施すことで簡単に実現できる（リスト１参照）。教師側の応答記録を採るためには、キー操作によって学生の疑問に答えられるようにプログラミングしておかねばならない。これは入力での一つのモード設定をすることであり、今後の課題としよう。また、再現性はＣＡＩ学習のデモに利用でき、操作法の解説が具体的にできて非常に有用な機能である。

```
/*-------------------------------------------------------*/%% 訂正入力述語
/*  入力ガイド表示機能付入力述語，リターンキー警告付属  */%% で参照される。
/*  標準化Ⅱ-1                    標準化Ⅰ-3             */
/*-------------------------------------------------------*/
ガイド付行入力(P,A) :- ガイド表示(P),記録付行入力(A),ガイド消去.
ガイド表示([Y,P])
     :- nonvar(P),!
      , ansi_cpr(Yc,Xc),ifthen(var(Y),Y is Yc-2)
      , string_length(P,L),ctr_set(5,1),退避行窓(1,Y,L )
          , ansi_cup(Y,1),ansi_sgr(41),write(P)
      , ifthen(ctr_is(30,1),リターンキー警告(Y))
      , ansi_sgr(0),ansi_cup(Yc,Xc)
        ;
      true,ctr_set(5,0).
ガイド消去 :-
    ctr_is(5,1),!
      ,ctr_set(5,0),復帰行窓(1),ifthen(ctr_is(30,1),復帰行窓(2))
    ;
        true.
リターンキー警告(Yc) :-
     リターンキー警告文(Y,P),string_length(P,L)
    ,ifthen(var(Y),Y is Yc+1),退避行窓(2,Y,L),ansi_cup(Y,1)
    ,put(27),write('[43;5m'),write(P).
リターンキー警告文(Y,'入力が終わったら、リターンキーを押して下さい。  ').

/*-------------------------------------------------------*/%% ガイド付行入力述語
/*  データ記録機能付キーボード；ファイル入力述語     */%% で参照される。
/*-------------------------------------------------------*/
記録付行入力(X):- filehandle(Fin,Fout),
    ifthenelse(var(Fin),
    /*then*/  read_line(0,X),
    /*else*/ (read_line(Fin,X),write(X),get0_noecho(_),nl)
    ),
    ifthen(nonvar(Fout),
    /*then*/ (write(Fout,X),put(Fout,13),put(Fout,10))
    ).
ファイルオープン:-
    abolish(filehandle/2),fileerrors(X,off),
       (入力モード(Min),!;asserta(入力モード(キー)),Min=キー),
    ifthen(Min\==キー,openl(Fin,Min,r)),
       (記録モード(Mrec),!;asserta(記録モード(off)),Mrec=off),
    ifthen(Mrec\==off,openl(Fout,Mrec,w)),
    asserta(filehandle(Fin,Fout)),fileerrors(_,X).
ファイルクローズ:- filehandle(Fin,Fout),ifthen(nonvar(Fin),close(Fin) ),
    ifthen(nonvar(Fout),close(Fout)),fileerrors(_,on),
    abolish(filehandle/2),abolish(入力モード/1),abolish(記録モード/1).

/*-------------------------*/%% ガイド付行入力述語定義節中の
/*    行窓操作述語他       */%% ガイド表示，ガイド消去やリターン警告
/*-------------------------*/%% で参照される。
          略
```

リスト１－ｃ：ガイド表示、データ記録述語他

5. 終わりに

入力形式というユーザーとコンピューターとの接点を形式的に考えて来たが、ここで得られた知見は、その形式性の故に学習内容、教材編成等を考えるのに役立つし「インテリジェント」部分を考えるのに役立つ。

【参考文献】

1. 谷口興紀、横井友幸「建築教育におけるパーソナルコンピューター利用の可能性について」 福山大学工学部紀要第６号、pp 58－65 ， 1984.

同(2) ―― prolog 言語と述語論理の可能性――
福山大学工学部紀要第７号、pp 135－142，1985.

同(3) ――ＣＡＩプログラムの一般的機能条件――
福山大学工学部紀要第８号、pp 48－54，1986.

同(4) ――建築分野におけるＣＡＩの現状――
福山大学工学部紀要第10号、pp 35－42，1988.

2. 谷口興紀「ＣＡＩプログラムのユーザーフレンドリー化に関する試論」 日本建築学会中国支部報第14巻、1988.

3. 谷口興紀，横井友幸「設計思考から見たＤＡＣシステムの性能仕様」 日本建築学会大会Ｓ 63.学術講演梗概集・計画系、1988.