PRAXIS: A Program for Reproducing Proforma Design Calculations

D. ALCOCK AND D. BROWN

PRAXIS is a computer program for reproducing design calculations in a paginated and tidy layout suitable for submission to a checking authority. The program is particularly suited to meeting structural engineers' calculations comprising formulae and table look-up; it reproduces calculations under control of a proforma which is analogous to a computer program. A growing library exists of proformas from various branches of structural engineering. The paper describes PRAXIS and the language in which its proformas are composed. Examples of typical calculations are provided. The current scope and possible future of PRAXIS is discussed. PRAXIS is distributed free for educational and non-commercial use.

This is a pre-copy-editing, author-produced PDF of an article accepted for publication in The Computer Journal following peer review. The definitive publisher-authenticated version (Vol 33, No. 4, 1990, pp.337-343) is available online at http://comjnl.oxfordjournals.org/cgi/content/abstract/33/4/337

1. INTRODUCTION

PRAXIS embodies a computer language for reproducing design calculations in a paginated and tidy layout suitable for submission to a checking authority. Some calculations reproduced by PRAXIS are illustrated in Fig. 1. From the calculations alone it would be impossible to tell whether the page came from a typewriter (being copied from a designer's hand-written calculations) or from a computer, because the calculation is self-descriptive. Every line is checkable. Every formula is shown in symbolic form before specific values are substituted. The only thing suggesting the involvement of a computer is the consistent neatness, and absence of arithmetical mistakes.

Although Fig. 1 shows a calculation for a brick wall, PRAXIS itself knows nothing about walls; it is independent of any particular engineering discipline. In order to reproduce calculations for a wall PRAXIS must be given a 'proforma' calculation for a wall. Such a proforma is shown in Fig. 2. This is the proforma from which the calculations shown in Fig. 1 were generated. The proforma corresponds directly to the finished calculations, groups of question marks in the proforma being replaced by parameters of a particular wall. The function of PRAXIS is to substitute these parameters, then resolve arithmetically the expressions in the proforma which were given algebraically.

Fig. 3a illustrates another proforma, details of which are explained more fully later in this paper. The resulting set of calculations in Fig. 3c shows extra lines generated from those in the proforma. Nevertheless the calculations in Fig. 3c remain self-descriptive and not evidently the product of a computer program.

During the last twenty-five years many design programs have been written to produce, as a by-product, calculations similar to those illustrated in Figs 1 and 3c. But these programs were written specifically to design beams or slabs or whatever the structural entities concerned. PRAXIS is a program which knows nothing about beams or slabs; such knowledge is embodied only in the proforma. Thus the proforma is analogous to a computer program for which PRAXIS is the interpreter (or processor). In this respect PRAXIS offers a novel approach to the reproduction of designers' calculations - at least as far as the authors are aware.

A proforma, being analogous to a computer program, need be composed only once for each problem. When a satisfactory proforma has been developed for the design of a rectangular column, say, then that proforma may be stored on disc to be recalled whenever a rectangular column is to be designed. An ever-growing library of proformas may be stored on disc in this way.

The rest of this paper explains the principles of processing a proforma, concentrates on certain features of the 'language' in which proformas are composed and describes the operation of PRAXIS on a computer. Finally this paper discusses the current scope and possible future of PRAXIS in structural engineering.

2. TYPEWRITER CONVENTIONS

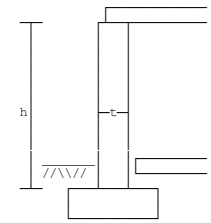
Design calculations are submitted to checking authorities either as a facsimile of the designer's handwritten work or as typescript. In the case of typescript there is the problem of how to represent the multiplication sign. Using x simply transposes the problem into one of uniquely representing the letter x. The designers of the FORTRAN programming language faced this problem years ago and adopted the asterisk as a multiplication sign. This convention has been long accepted among engineers and so has been adopted in PRAXIS.

It would be wrong to blame 'computers' for the lack of a proper multiplication sign; the problem existed with typewriters long before computers were invented. Those who wrote programs for the Ferranti Pagasus computer in the 1950s may remember the genuine multiplication sign we then took for granted. Such is progress!

A similar problem arises with subscripts and exponentiation. On a typewriter the drum may be given a halfturn to elevate the exponent or depress the subscript, but this is no solution for a printer connected to a computer. The designers of FORTRAN adopted a double asterisk for exponentiation (E**X to mean E to the power X) but today's preferred solution, and the one adopted in PRAXIS, is the up-arrow or caret sign (E^X to mean E to the power X). Subscripts in PRAXIS are simply left on the same line.

```
N G NEERS & R K TECTS CO-PARTNERSHIP                    Page:    1
                                                     Made by:  DWB/DGA
Job: NEW PREMISES for G CHISHOLM at                     Date:  3.3.87
     ECCLEFECHAN and AUCHTERMUCHTY                    Ref No:  87687
```

Location: External wall of garage



Single leaf wall

This calculation is in accordance with BS5628-1:1978 being the Code of practice for STRUCTURAL USE OF MASONRY Part 1. Unreinforced masonry.

The form of this calculation closely follows that given In Example 5.1 of 'Structural Design of Manonry' by Andrew Orton.

```
Height of wall                  : h=2.2 m
Thickness of wall               : t=100 mm
Eccen. of loads at top of wall : ecc=20 mm

A long external single leaf wall has a vertical load on it from the
roof and a horizontal load from wind as follows:

Characteristic loads:
Dead load                       : Gk=1.20 kN/m
Live load                       : Qk=1.60 kN/m
Horizontal wind load            : Wk=0.25 kK/m2

There is assumed to be no uplift on the roof.  The roof construction
provides simple resistance to lateral movement.

Char comp strength of masonry  : fk=6.00 N/mm2
Char flex strength of masonry  : fka:0.40 N/mm2
Part safety factor materials   : gammam=2.5

Vertical loading
Slenderness ratio               : SR=h*1000/t
                                    =2.2*1000/100
                                    =22
At top of wall:
Design vert load                : g=1.4*Gk+1.6*Qk
                                   =1.4*1.2+1.6*1.6
                                   =4.24 kN/m
Design bending moment           : M=g*ecc/1000
                                   =4.24*20/1000
                                   =0.0848 kNm
```

```
N G NEERS & R K TECTS CO-PARTNERSHIP                    Page:    2
                                                     Made by:  DWB/DGA
Job: NEW PREMISES for G CHISHOLM at                     Date:  3.3.87
     ECCLEFECHAN and AUCHTERMUCHTY                    Ref No:  87687
```

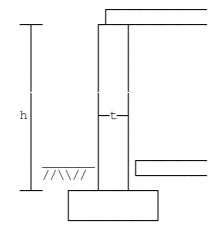
```
Resultant eccentricity          : ex=M/g
                                    =0.0848/4.24
                                    =0.02 m
Eccentricity ratio              : e=ex*1000/t
                                   =0.02*1000/100
                                   =0.2
Capacity reduction factor       : beta=TABLE 7 for SR=22, e=0.2
                                     =0.43
Design vert load resist of wall: gd=beta*(t/1000)*(fk*1000)/gammam
                                    =0.43*(100/1000)*(6*1000)/2.5
                                    =103.2 kN/m
Design vertical load within vertical load resistance of wall,
therefore OK.

Horizontal loading
Limiting dimensions of panel    : lim=40*t/1000
                                     =40*100/1000
                                     =4 m
Height of wall less than limiting dimension, therefore OK.
Section modulus                 : Z=1000*t^2/(6*10^9)
                                   =1000*100^2/(6*10^9)
                                   =0.0016667 m3
At middle of wall elastic design moment of resistance of wall ignoring
effect of vertical load         : Mr=fka*1000*Z/gammam
                                    =0.4*1000*0.0016667/2.5
                                    =0.26667 kNm/m
Elastic design moment           : M=1.4*Wk*h^2/8
                                   =1.4*0.25*2.2^2/8
                                   =0.21175 kNm/m
Elastic design moment within design moment of resistance, therefore OK.
```

Figure 1. Typical output from PRAXIS.

```
STORE 7 13 4 ! Table 7 from BS5628 having 13 rows and 4 columns
    0.05  0.1   0.2   0.3
0   1.00  0.88  0.66  0.44
6   1.00  0.88  0.66  0.44
8   1.00  0.88  0.66  0.44
10  0.97  0.88  0.66  0.44
12  0.93  0.87  0.66  0.44
14  0.89  0.83  0.66  0.44
16  0.83  0.77  0.64  0.44
18  0.77  0.70  0.57  0.44
20  0.70  0.64  0.51  0.37
22  0.62  0.56  0.43  0.30
24  0.53  0.47  0.34  0.00
26  0.45  0.38  0.00  0.00
27  0.40  0.33  0.00  0.00

START
Location: ????
```



Single leaf wall

This calculation is in accordance with BS5628-1:1978 being the Code of practice for STRUCTURAL USE OF MASONRY Part 1. Unreinforced masonry.

The form of this calculation closely follows that given In Example 5.1 of 'Structural Design of Manonry' by Andrew Orton.

```
Height of wall                  : +h=???? m
Thickness of wall               : +t=???? mm
Eccen. of loads at top of wall : +ecc=???? mm

A long external single leaf wall has a vertical load on it from the
roof and a horizontal load from wind as follows:

Characteristic loads:
Dead load                       : +Gk=???? kN/m
Live load                       : +Qk=???? kN/m
Horizontal wind load            : +Wk=???? kK/m2

There is assumed to be no uplift on the roof.  The roof construction
provides simple resistance to lateral movement.
```

```
Char comp strength of masonry   : +fk=???? N/mm2
Char flex strength of masonry   : +fka:???? N/mm2
Part safety factor materials    : +gammam=????
IF gammam<2.5
Partial safety factor out of range (Table 4).
STOP
ENDIF
II gammam>3.5
Partial safety factor our of range (Table 4)
STOP
ENDIF

/5
Vertical loading
Slenderness ratio               : +SR=h*lOOO/t
At top of wall:
Design vert load                : +g=1.4*Gk+1.6*Qk kN/m
Design bending moment           : +M=g*ecc/1000 kNm
Resultant eccentricity          : +ex=M/g m
Eccentricity ratio              : +e=ex*1000/t
IF e<0.05
Capacity reduction factor       : +beta=1
ELSE
Capacity reduction factor       : beta=TABLE(7,SR,e)
ENDIF
Design vert load resist of wall: +gd=beta*(t/1000)*(fk*1000)/gammam kN/m
IF g>gd
Design vertical load exceeds vertical load resistance of wall
STOP
ELSE
Design vertical load within vertical load resistance of wall, therefore OK.
ENDIF

/5
Horitontal loading
Limiting dimensions of panel    : +lim=40*t/1000 m
IF h<lim
Height of wall less than limiting dimension, therefore OK.
ELSE
Height of wall exceeds limiting dimension.
STOP
ENDIF
Section modulus                 : +Z=1000*t^2/(6*10^9) m3
At middle of wall elastic design moment of resistance of wall ignoring
effect of vertical load         : +Mr=fka*1000*Z/gammam kNm/m
Elastic design moment           : +M=1.4*Wk*h^2/8 kNm/m
IP M<=Mr
Elastic design moment within design moment of resistance, therefore OK
ELSE
Elastic design moment exceeds design moment of resistance.
STOP
ENDIF
FINISH
```

Figure 2. A proforma for a brick wall.

```
START
Natural frequency of multi-storey frames
```

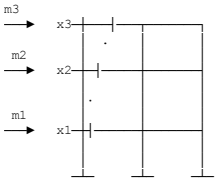
For frames in which the mass can be assumed to be lumped at floor levels, and where compression of columns and rotation of column heads may be considered negligible in comparison to sway effects, the period of the fundamental mode is given by:

$$T = 2.PI \sqrt{\frac{sigma(mi.xi^2)}{g.sigma(mi.xi)}}$$ Raleigh's formula

Where: T is the period of the fundamental mode

mi and xi are the i'th mass and its displacement for the set of masses m1,m2,... ,mn and corresponding static displacements under gravity forces x1,x2,... ,xn

The method:
a) compute the sum of DL + 20% of imposed load for each floor and roof
b) apply these as a set of horizontal forces to the structure
c) substitute computed deflections and masses in Raleigh's formula to find T.



e.g. for a three storey frame:
x1 x2 & x3 are the displacements corresponding to the set of masses
m1 m2 & m3, then period is given by

$$T = 2 \times 3.14 \sqrt{\frac{m1.x1^2 + m2.x2^2 + m3.x3^2}{g(m1.x1 + m2.x2 + m3.x3)}}$$

In SI units, g = 9.81 m/s/s

Location: ????

```
Number of levels                : +n=????
! Initialise variables           +i=0
!                                +sigmx=0
!                                +sigmx2=0
REPEAT
!                                +i=i+1
Level i= +i
Lumped mass at this level       : +m=???? kN
Corresponding displacement      : +x=???? m
Running total of m.x product    : +sigmx=sigmx+m*x kNm
Running total of m.x^2          : +sigmx2=sigmx2+m*x^2 kNm2
UNTIL i=n
ENDREPEAT
Period                          : +T=2*PI*SQR(sigmx2/(9.81*sigmx)) sec
Natural frequency               : +f=1/T hertz
FINISH
```

```
STRUCTURE THEO D LITE & LEVEL ASSOCIATES
STRUCTURE
STRUCTURE Job: HIGH ENERGY PARTICLE ACCELERATOR
STRUCTURE      SUPPORT BENTS
REFNO 87787
DATE 4.3.87
MADEBY DWB/DGA
```

Figure 3. Full example. (a) Typical proforma. (b) Job data file.

Natural frequency of multi-storey frames

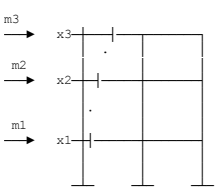
For frames in which the mass can be assumed to be lumped at floor levels, and where compression of columns and rotation of column heads may be considered negligible in comparison to sway effects, the period of the fundamental mode is given by:

$$T = 2.PI \sqrt{\frac{sigma(mi.xi^2)}{g.sigma(mi.xi)}}$$ Raleigh's formula

Where: T is the period of the fundamental mode

mi and xi are the i'th mass and its displacement for the set of masses m1,m2,... ,mn and corresponding static displacements under gravity forces x1,x2,... ,xn

The method:
a) compute the sum of DL + 20% of imposed load for each floor and roof
b) apply these as a set of horizontal forces to the structure
c) substitute computed deflections and masses in Raleigh's formula to find T.



e.g. for a three storey frame:
x1 x2 & x3 are the displacements corresponding to the set of masses
m1 m2 & m3, then period is given by

$$T = 2 \times 3.14 \sqrt{\frac{m1.x1^2 + m2.x2^2 + m3.x3^2}{g(m1.x1 + m2.x2 + m3.x3)}}$$

In SI units, g = 9.81 m/s/s

Location: Ex. 'Matrices for structural analysis' by McMinn

```
Number of levels                : n=3
Level 1
Lumped mass at this level       : m=25 kN
Corresponding displacement      : x=0.0175 m
Running total of m.x product    : sigmx=sigmx+m*x
                                        =0+25*0.0175
                                        =0.4375 kNm
Running total of m.x^2          : sigmx2=sigmx2+m*x^2 kNm2
                                        =0+25*0.0175^2
                                        =0.0076563 kNm2
```



```
Level 2
Lumped mass at this level       : m=23.3 kN
Corresponding displacement      : x=0.0796 m
Running total of m.x product    : sigmx=sigmx+m*x
                                        =0.4375+23.3*0.0796
                                        =2.2922 kNm
Running total of m.x^2          : sigmx2=sigmx2+m*x^2 kNm2
                                        =0.0076563+23.3*0.0796^2
                                        =0.15529 kNm2
Level 3
Lumped mass at this level       : m=4.2 kN
Corresponding displacement      : x=0.1191 m
Running total of m.x product    : sigmx=sigmx+m*x
                                        =2.2922+4.2*0.1191
                                        =2.7924 kNm
Running total of m.x^2          : sigmx2=sigmx2+m*x^2 kNm2
                                        =0.15529+4.2*0.1191^2
                                        =0.21486 kNm2
Period                          : T=2*PI*SQR(sigmx2/(g*sigmx))
                                  =2*3.1416*sqr(0.21486/(9.81*2.27924))
                                  =0.55647 sec
Natural frequency               : f=1/T
                                  =1/0.55647
                                  =1.7971 hertz
```

Figure 3(c). Resulting calculations.

The absence of Greek letters is another problem with typewriters and computers alike. For the examples in this paper the Greek letters are spelled out (`alpha`, `beta` etc.).

Using the conventions described above it is possible to write any formula along the line. However, there is nothing to prevent the designer of a proforma from exercising ingenuity with formulae - or even with diagrams as can be seen in Fig. 3.

3. BASIC PRINCIPLES

To produce a set of calculations PRAXIS must be given a proforma, as already explained. The basic function of PRAXIS is then to copy the proforma line by line to the output. For some lines in a proforma this is all that happens; comparison of Figs 1 and 2 reveals several lines of the proforma copied to the output without transformation. Other lines of the proforma cause something else to happen, as explained separately below for each kind of happening.

3.1. Exclamation marks

When a line is about to be copied by PRAXIS it is checked for exclamation marks. When an exclamation mark is found it is not copied to the output, nor is anything to its right on the same line. In other words, an exclamation mark suppresses printing of the rest of its line. If the exclamation mark is the first character, the whole line is suppressed.

Suppression does not mean the line is ignored; if the exclamation mark precedes an assignment (see later) the assignment is still carried out. The line following `REPEAT` in Fig. 3a illustrates this case.

3.2. Keywords

In Fig. 3a the words in capitals are 'keywords'; for example `REPEAT`, `UNTIL` and `FINISH`. Lines beginning with keywords are not themselves copied to the output but determine which line is to be copied next. `FINISH` determines that no line is to be copied next. The effects of `REPEAT` and `UNTIL` should be apparent from Fig. 3; they are discussed in more detail later.

Keywords which precede brackets, such as `COS( )`, `ABS( )` and `SQR( )`, denote functions just as in FORTRAN or BASIC. The keyword `TABLE` invokes the special table look-up essential to PRAXIS and explained later.

3.3. Question marks

A line containing question marks is copied to the output but not immediately. The presence of one or more question marks (conventionally `????`) causes that line to be displayed on the screen as a prompt to the user of PRAXIS. For example, in Fig. 3a the line reading:

```
Number of levels : +n=????
```

is displayed on the screen leaving the user to type `3` (say) and press the RETURN key. The line is then copied to the output as though it had been given as:

```
Number of levels : n=3
```

Table 1. Examples of transformation from proforma to output

As on proforma	As copied to output page
`+a`	`67.3`
`+a*b`	`134.6`
`+b=12.5`	`b=12.5`
`+C=12.5/2`	`c=12.5/2` `=6.25`

`+M=1.4xWk+hA2/8 kNm/m`	`M=1.4*Wk*h^2/8`
	`=1.4*0.25x2.2^2/8`
	`=0.21175 kNm/m`

`+beta=TABLE(7,SR,e)`	`beta=TABLE 7 for`
	`SR=22, e=0.2`
	`=0.43`

in the proforma. This is what turns a generalised proforma into a particularised set of calculations. Figure 5 depicts the mechanism of response to prompts.

3.4. Equals signs

An equals sign in the proforma signifies 'assignment' to a 'variable'. This is a simple concept to computer programmers. In the line of a proforma reading:

```
Number of levels : +n=3
```

the equals sign tells PRAXIS to assign the number `3` to a variable named `n`. A variable is conceptually a little box which has a name and can hold a number. Subsequently, when the name `n` appears to the right of an equals sign it simply stands for the number currently held in the little box of that name.

3.5. Leading plus signs

A leading plus sign in the proforma tells PRAXIS to substitute numbers for names, also to do any arithmetic specified by the operators `^`, `*`, `/`, `+`, `-` or functions like `SQR( )`. Because output is to be self-descriptive the original expression is shown first, then the substitutions, then the result of the arithmetic. This three-line resolution is illustrated in Fig. 3c in the formula for `sigmx`. Notice in Fig. 3c how the `kNm` is displaced to the third line, all the signs are aligned, the leading plus sign is stripped off.

Not all substitutions involve three lines of output to every one of the proforma; Table 1 gives an example of each possibility.

4. ASPECTS OF PROGRAMMING

Developing a proforma is similar to programming in a structured language like Pascal. PRAXIS has the equivalent of read and write statements, structured control statements and assignment statements. Assignment statements may involve complicated expressions. The language also allows the definition of procedures. These aspects of programming are separately described below.

The table look-up facility in PRAXIS is not typical of programming languages and merits special attention.

4.1. Expressions

All arithmetic is floating-point arithmetic carried out to high precision (PRAXIS is written in FORTRAN and employs type DOUBLE PRECISION). The number of decimal digits to be sent to the output, however, may be set in the proforma as desired, five digits being typical.

An expression comprises numbers, names of variables, references to functions - all bound together by operators and brackets. The usual conventions for precedence apply in PRAXIS; for example the `*` is applied before `+` unless brackets override, as in `2*(3+4)`. Figure 3*a* shows several expressions assigned to variables, the expression assigned to `T` being the most complicated. Any reasonable level of complexity of expression is permissible in a PRAXIS proforma.

4.2. Functions

The proforma of Fig. 3*a* shows a line reading:

```
Period : +T=2*PI*SQR(sigmx2/
  (9.81*sigmx)) sec
```

In resolving this formula the term `sigmx2/(9.81*sigmx)` is first evaluated, then the square root of the resulting value found. `SQR( )` is an instruction to the computer to compute the square root of the value in brackets. The `SQR( )` is called a 'function'.

A comprehensive range of functions is provided in PRAXIS for dealing with logarithms, trigonometrical ratios, hyperbolic functions and miscellaneous arithmetical functions such as absolute value and integral part.

`PI` is a special variable known to PRAXIS and holding 3.14159.

4.3. Table look-up

Most design calculations are based on codes of practice. Codes of practice contain tables in which values have to be looked up by reading across from a row header and down from a column header. Because values have to be looked up so frequently a facility has been devised for doing so automatically, interpolating from the table wherever row and column headers are not matched precisely.

Figure 2 starts with a table from BS 5628. An item in a proforma reading:

```
+beta=TABLE(7,16,0.2)
```

would cause table `7` to be read across from `16` and down from `0.2`. In this case a value of `0.64` would be found and assigned to the variable named `beta`. A more general look-up might be expressed :

```
+beta = TABLE(7,SR,e)
```

where appropriate row and column headers are to be found in the variables named `SR` and `e`. If these values did not coincide precisely with row and column headers in table `7`, the value assigned to the variable `beta` would be interpolated (by two-way linear interpolation) from adjacent values. A need for extrapolation, on the other hand, would signal an error.

Tables consulted from a proforma must be given to PRAXIS at the beginning of a proforma and before the word `START`. The keyword `STORE` introduces: table number, the number of rows, number of columns, then the data for the table itself including row and column headers. As far as the authors are aware, the table lookup facility of PRAXIS is a novel development.

4.4. Conditions and loops

From the simplest proforma, lines are copied one by one to the output, numerical substitutions being made during the process. But it is possible to make the copying of a line conditional upon some calculated result. It is also possible to make PRAXIS copy a given set of lines again and again until some calculated condition is satisfied. In computer jargon, PRAXIS allows 'conditions' and 'loops'.

In the proforma of Fig. 2 is the conditional statement beginning

```
IF gammam<2.5
```

Conditions following IF or UNTIL may be far more general, involving arithmetic expressions separated by < or <= or = etc. A further example is

```
IF x+0.2>=SIN(RAD(a))
```

The `REPEAT` loop in PRAXIS is illustrated in Fig. 3*a*. The `UNTIL` statement may be placed anywhere between `REPEAT` and `ENDREPEAT`. By placing `UNTIL` just after `REPEAT` a 'while' loop can be constructed; by placing `UNTIL` just before `ENDREPEAT` a 'for' loop or 'repeat' loop can be constructed. In a Pascal program each of these types of loop would have a different syntax.

The control statements in PRAXIS may be nested to any reasonable depth. Being a 'fully structured' language, PRAXIS has no need of a `GOTO` statement.

4.5. Procedures

It may happen that a sequence of lines written once in a proforma is needed somewhere else in the same proforma. Rather than duplicate all the lines of that sequence it is possible to parcel them up as a 'procedure' and give the procedure a name. Then the name alone may be included wherever the full sequence would otherwise be required in the proforma. This concept is familiar to programmers.

An example of a procedure defined is:

```
DEFINE CEng
These calculations were all made
by competent Chartered Engineers
ENDDEFINE
```

Elsewhere in the proforma this procedure might be invoked several times as follows:

```
CEng
+area=PI*radius^2
CEng
+volume=area*height
CEng
```

The effect in the output would be a pious message about Chartered Engineers before and after every piece of calculation. Procedures do, however, find less banal application in complicated proformas.

5. OPERATION

PRAXIS is a program written in standard FORTRAN; it requires no particular computer or operating system. Anyone attempting to run a PRAXIS job should know enough about the computer being used to be able confidently to create, display, edit, rename and print data files simple character files stored on disc.

5.1. Data files

Figure 4 depicts the creation of files by entering data via the computer's keyboard. Before PRAXIS is started the following data files should be stored on disc: the proforma file - its name should end with the 'extension' .PRO (example: BEAM.PRO); a job data file with the few details to appear at the top of every page of results - firm's name, date, etc. The name of this file should have .DAT as its extension (example: CON123.DAT).

5.2. Calculation file

When PRAXIS is started, the screen asks for the names of the data and proforma files (CON123.DAT and BEAM.PRO in the examples above). After these names have been given PRAXIS starts copying the proforma file line by line to a calculation file created automatically and given the same name as the data file but with extension .CAL (example: CON123.CAL).

5.3. Interaction

Each time a line with question marks is copied from the proforma the user is required to enter an appropriate number, expression or character string to be substituted for the question marks. The execution stage is depicted in Fig. 5. As the calculation proceeds it is displayed on the screen. At the end of the display there is an invitation to the user to type:

C if changes are to be made to items given in response to question marks
P if the calculation is now to be printed
R to return to the start (to change to another proforma for example)
D to display the calculation
S to stop.

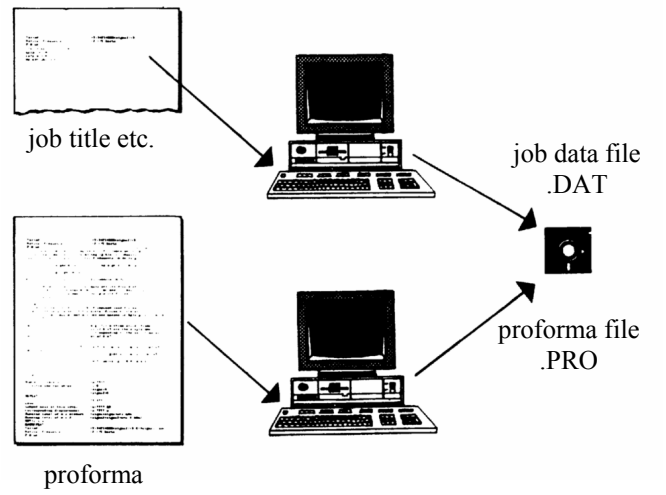


Figure 4. Creating files before execution.

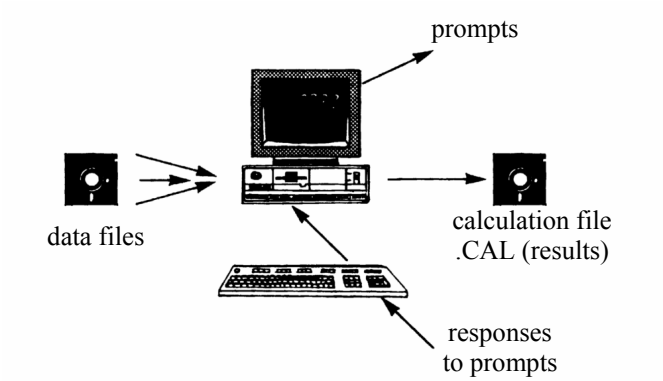


Figure 5. Execution of PRAXIS.

If changes are requested, as is usual when trying different section sizes, the proforma is copied again. But when question marks are encountered the copying stops and the previously chosen response is displayed. The user may now type a new item or accept the previous one simply by pressing the RETURN key.

6. APPLICATIONS

6.1. Scope of application

PRAXIS has so far been used only in the field of structural engineering. That is not to suggest its scope is confined to structures, only that the originators of PRAXIS work in that field and have not looked for wider application. It is left to the reader to consider the use of PRAXIS in other fields. Such consideration might suggest extensions to the program; for example, electrical engineers may see the need for PRAXIS to handle the arithmetic of complex numbers. The authors welcome suggestions for extending the facilities of PRAXIS.

In the field of structural engineering PRAXIS has been applied in design work involving codes of practice for:

- selection of reinforcement in concrete beams, slabs and columns all to BS 8110;
- size selection of steel beams and stanchions and connections to BS 449 and BS 5950;
- design of masonry columns and walls to BS 5628;
- design of timber beams and columns to BS 5268.

PRAXIS proformas have also been written for miscellaneous calculations not directly involving a code of practice :

- properties of variously shaped cross-sections;
- fixed-end moments of a beam under various loading patterns (point load, udl, triangular);
- forces exerted by soil on face of a retaining wall;
- pad base with overturning moment;
- distribution of moments in subframes.

6.2. Embryonic library

Once a proforma has been developed it may be stored in a library for further use by the original developer or by others. The authors have started such a library, which is available to interested users of PRAXIS. Conversely, the authors would be interested to receive new or improved proformas for inclusion in this library.

6.3. Code revisions

Most proformas are abstracted from codes of practice; in other words a typical proforma embodies a design procedure specified by a code. Experience gained in transforming design procedures from words to proformas could be useful to those responsible for revising codes of practice.

Programmers of the 1960s used to complain that codes of practice containing tables and graphs did not always show the formulae from which the tabulated or plotted values had been derived. By the 1970s the message had seeped through to standards committees, with the result that all modern codes present such formulae explicitly.

Perhaps PRAXIS, or the ideas embodied in it, will motivate a similar cycle to that described above. In places where the wording of a code is vague or ambiguous it is difficult to devise a proforma, so perhaps the committee should attempt to do this before a new or revised code is published. Perhaps a proforma could be included with each published table or graph. Conversely, if a code committee found it too difficult to specify a design procedure as a PRAXIS proforma, would it be wise to include that procedure in a code of sound practice? Writing a PRAXIS proforma is one way to pare a verbal specification with Occam's razor.

6.4. Approval

A curious question arises from the use of PRAXIS in work undertaken for various clients. Because it is a computer program, should PRAXIS be put forward for 'approval' as required for other design programs? Earlier in this paper we showed that calculations produced by PRAXIS were self-descriptive; that there was no way of proving they were generated by computer. So the logical answer, perhaps not a diplomatic one, should be No! But as with all computer results, the engineer must check the calculations. As the engineer is now spared the slog of producing calculations he or she should be prepared to use some of the time saved to do a thorough check of every line of the calculations with the traditional red and yellow pencil.

Acknowledgements

The PRAXIS language was developed by both authors; the program was written by D. W. Brown. Examples in this paper were taken from the *PRAXIS User's Manual* by Alcock and Brown.

PRAXIS was developed on the IBM personal computer (PC) and is distributed free of charge for educational and non-commercial use on PCs and compatibles.

The control statements of PRAXIS were modelled on those in the programming language Super-BASIC, which runs on the Sinclair QL computer, the Thor and ICL One-per-desk.

The authors are grateful to Boyd Auger, who thoroughly tested the program during the production of proformas for the limit state design of structural steelwork, reinforced concrete and masonry structures.

To justify the name PRAXIS we quote from *The Pattern of English* by G. H. Vallins (André Deutsch, 1956) as follows: '...a Praxis, that is, a kind of analysing or parsing of a selected English passage, so that the principles and rules laid down in the text could be seen in action'. In our opinion no name would suit PRAXIS better than PRAXIS.