Roland G. Hülsmann
Fraunhoferstraße 20
6800 Mannheim 31

0621/733850

Programmieren für Anwender

P R O F A N 1.4

Die einfache Programmiersprache unter Windows 3.x

0 INHALT

1 EINFÜHRUNG

Gewiß, es gibt schon Batchprogrammiersprachen unter Windows. Die bekanntesten sind "WinBatch" und "Oriel". "WinBatch" ist eine herausragende Sprache, aber leider völlig ohne Grafik.
"WinBatch" von "WilsonWare" (USA) ist sowohl als Shareware oder Vollversion (mit deutschem Handbuch) erhältlich, als auch als Batchsprache (unter anderem Namen) im Norton Desktop für Windows integriert.
"Oriel" hingegen hat umfangreiche Grafikbefehle, aber doch einen sehr eingeschränkten Befehlsumfang. So gibt es da z.B. keine Möglichkeit Eingaben zu machen oder Dateien von Diskette oder Festplatte zu lesen.
Beide Sprachen enthalten keinen Compiler. Anwendungen können daher nur an Leute weitergeben werden, die Ihrerseits diese Sprache gekauft haben.

Ziele der Entwicklung von Profan waren:

* Eine einfache - an BASIC angelehnte - Syntax für den Anfänger
* Alle Grafikmöglichlkeiten, die "Oriel" bietet ... und noch mehr
* umfangreiche Datei- und Verwaltungsfunktionen
* der komplette Interpreter kleiner als 100 kB, keine weiteren DLLs oder sonstige Dateien
* Möglichkeit, Anwendungen ohne weitere Kosten weiterzugeben.

Herausgekommen ist "Profan". Eine komplette Programmiersprache. Der Anfänger wird viele BASIC-Befehle in gewohnter Form wiederfinden, u.a. PRINT, LOCATE, CLS, INPUT, GOTO, GOSUB, RETURN, IF, WHILE, WEND, SOUND, PRINT #n, INPUT #n, END, LET, ...
Auch die Variablen und Konstanten werden weitestgehend wie in BASIC gehandhabt.
Bei den Dateioperationen wurde allerdings das etwas vielseitigere Konzept von PASCAL übernommen:
ASSIGN, RESET, REWRITE, APPEND, RENAME, ERASE, CLOSE, ...
Dazu kamen dann noch WINDOWS-spezifische Dinge, wie Menüs, Listboxen, Inputboxen. Load- und Save-Dialoge und vieles mehr.
Und die Krönung sind natürlich die umfangreichen Grafikbefehle, die direkt auf den Grafik-Kern von Windows zugreifen: USEBRUSH, USEFONT, USEPEN,
USEICON, COPYBMP, RECTANGLE, ROUNDRECT, ELLIPSE, ARC, PIE, DRAWTEXT, ...
Der fortgeschrittene Anwender wird vermutlich zur Bildschirmausgabe die textorientierten Ausgabebefehle des BASIC weniger verwenden, als vielmehr die neuen und mächtigen Windows-Befehle.
Strukturierte und übersichtliche Programmierung wird ermöglicht, indem nur ein Befehl pro Zeile erlaubt ist und zahlreiche Kontrollstrukturen verfügbar sind: IF, CASE, WHILE, GOSUB. Außerdem ist beliebiges Einrücken ebenso statthaft, wie komplett leere Zeilen.

Im Laufe der Entwicklung kam dann auch noch ein Compiler dazu, der einen sehr kompakten Zwischencode erzeugt, der mit einem Runtime-Modul gestartet werden kann. Das letzte Glied in der Kette zur kompletten Programmiersprache war dann der Linker, der Runtimemodul und Zwischencode
zu einer eigenständigen EXE-Datei linkt. Und da das Runtime-Modul nur wenig

mehr als 80 kB groß ist, sind recht aufwendige Windowsaplikationen unter 100 kB möglich. Da die fertige PROFAN-Applikation immer noch als Runtime-Modul für weitere Zwischencode-Dateien dienen kann und sogar eine Übergabe
von Parametern möglich ist, steht der modularisierten Programmierung nichts im Wege. Die Sahnehaube aber ist PROFED, die Entwicklungsumgebung ...

Ach ja: Das berühmte "Hallo Welt"-Programm in Profan:

```
Print "Hallo Welt"
WaitKey
End
```

So einfach ist das. Das "WaitKey" steht nur deshalb da, damit der Anwender Gelegenheit hat, das Ergebnis zu betrachten: Ein Programm in einem Windows-Fenster mit allem was dazugehört. Sozusagen eine vollständige Windows-Applikation. Mit einem Tastendruck oder über die entsprechende Fensterfunktion wird das Programm beendet.

2 INSTALLATION

PROFAN benötigt zum optimalen Funktionieren mindestens einen 286er mit 2 MB RAM und VGA-Grafik. Bei weniger RAM kann ein Profan-Programm u.U. kein anderes Programm mehr aufrufen. Bei EGA- oder HERCULES-Grafik wird das DEMO nicht ordentlich angezeigt und bei der Programmierung ist darauf zu achten, daß mit WINDOW die richtige Fensterschirmgröße eingestellt wird.

HINWEIS: Wenn beim DEMO nicht alles funktioniert, wie erwartet (also z.B. DRAW nicht startet, etc.), können die einzelnen Programmteile über die Zeile PROFRUN <name>.PRC problemlos aufgerufen werden. Das spart Speicherplatz. (Shareware: SHPRRUN <name>)

PROFAN ist vom Installationsprogramm auf dem von Ihnen gewählten Datenträger im gewählten Verzeichnis installiert worden.

(Die Sharewareversion ist gemäß Anleitung in der Datei LIESMICH.TXT zu installieren.)

Damit PROFAN-Programme, auch wenn sie nicht zur EXE-Datei gelinkt sind, durch Doppelklick gestartet werden können, sind folgende Zeilen in die Datei WIN.INI unter dem Abschnitt [Extensions] einzufügen:

prf=c:\profan\profan.exe ^.prf
prc=c:\profan\profrun.exe ^.prc

(Besitzer der Shareware-Version geben statt "profan.exe" und "profrun.exe" bitte "shprofan.exe" und "shprrun.exe" an. Sollten Sie PROFAN nicht auf Laufwerk C: installiert haben, ändern Sie die Zeilen entsprechend ab.)

Nur die wenigsten Dateien werden wirklich gebraucht. Die meisten Dateien gehören zum Demonstrationsprogramm und dienen Ihrer Information.

DEMOPROGRAMM:

Das DEMO-Programm ist komplett in PROFAN geschrieben. Die Listings sind in der registrierten Vollversion mit auf der Diskette.

DEMO.EXE - Hauptprogramm
*.PRC - kompilierte Programmodule. Beachten Sie die Kompaktheit selbst so komplexer Programme, wie RGH-DRAW.
*.BLD - Beispielbilder zum Malprogramm
*.PLT - Beispiel-Farbpaletten für Malprogramm und Farbkasten.
*.BMP - Bilder für das Hypertext-Beispiel
*.TXT - Texte für den Schnelleinstieg
*.PRF - Listings des DEMO-Programmes (nur Vollversion) und Beispielprogramme

PROFAN (Entwicklung):

Diese Dateien sind unbedingt notwendig:

(In Klammern steht bei abweichendem Dateinamen der Name der Datei in der Shareware-Version.)

PROFAN.EXE (SHPROFAN.EXE) - Der Interpreter
PROFED.EXE (SHPROFED.EXE) - Die Entwicklungsumgebung
*.HLP - Hilfstexte für die Entwicklungsumgebung (nur Vollversion)
*.CRD - Hilskartei zur PROFAN-Syntax
DRUCK.EXE - (nur Vollversion) Druck-Programm
ANSI.EXE - (nur Vollversion) ANSI-Tabelle
PROFCOMP.EXE (SHPRCOMP.EXE) - Der Compiler
PROFLINK.EXE (SHPRLINK.EXE) - Der Linker
PROFRUN.EXE (SHPRRUN.EXE) - Das Runtime-Modul

(Wollen Sie PROFAN ausschließlich als Batchsprache nutzen, können Sie auf die letzten drei Programm-Dateien verzichten. Compilieren und Linken ist dann weder notwendig noch sinnvoll.)

PROFAN (Anwendung als Batchsprache für Windows):

PROFAN.EXE (SHPROFAN.EXE) - Der Interpreter
*.PRF - Das Batchprogramm (+ Module) in Profan

PROFAN (compilierte Anwendung):

PROFRUN.EXE (SHPRRUN.EXE) - Das Runtime-Modul oder ein zur EXE-Datei gelinktes Profanprogramm
*.PRC - Programm, bzw. Programmmodule

Ein zur EXE-Datei gelinktes Profanprogramm ist ohne weitere Dateien eigenständig lauffähig.

3 "RGH-PROFAN-Editor 1.4"

Der Editor ist eigentlich eine komplette Entwicklungsumgebung für RGH-PROFAN.

Der Editor ist ein "Multi-File"-Editor, das heißt: Es können mehrere Dateien gleichzeitig editiert werden. Das ermöglicht einfaches Kopieren und Verschieben auch zwischen verschiedenen Programmen über die Zwischenablage. Um ein anderes Programm oder einen anderen Text einzusehen, muß das aktuelle Programm nicht verlassen werden.

Die Menüpunkte im Einzelnen:

DATEI - Neu: Ein neues RGH-PROFAN-Programm schreiben. Es bekommt zunächst den Namen "OHNENAME.PRF". Existiert bereits ein Programm "OHNENAME.PRF", so wird es geladen. Es empfiehlt sich, das Programm sogleich mit "Speichern als" abzuspeichern und ihm einen neuen Namen zu geben.

- Öffnen: Eine bereits bestehende PROFAN-Datei wird zum Bearbeiten geöffnet. (Durch Ändern der Dateimaske können beliebige Texte geladen werden.)
- Speichern: Die aktuell aktive PROFAN-Datei wird unter ihrem Namen gespeichert.
- Speichern als: Die aktuell aktive PROFAN-Datei kann unter einem neuen Namen/Pfad abgespeichert werden.
- Druckprogramm: Das Druckprogramm "RGH-DRUCK" wird aufgerufen. Da das Druckprogramm die zu druckende Datei von der Festplatte liest und nicht aus dem Editor, überprüft das Programm zunächst ob geänderte Dateien etwa noch nicht abgespeichert wurden und fragt nach, ob man dies nachholen möchte.
- Systemsteuerung: Die Systemsteuerung wird aufgerufen. Hier kann zum Beispiel die Druckereinstellung geändert werden.
- DOS-Shell: Die von Windows 2.x her bekannte Shell wird aufgerufen. Um mal rasch ein anderes Programm aufzurufen oder etwas zu kopieren/löschen ist es dieses Programm schneller, als der Filemanager von Windows 3.0. (Bei Windows 3.1 ist dieser Menüpunkt nicht verfügbar, es sei denn, Sie haben das Programm MSDOS.EXE aus Ihrer 3.0-Version hinübergerettet. Wenn Sie den Filemanager zu MSDOS.EXE umkopieren, können Sie auch diesen nutzen. In 3.1 ist er ja wirklich gut!)
- Ende: Der Editor wird beendet. Sollten Programme nach der Veränderung noch nicht abgespeichert sein, wird jeweils nachgefragt, ob man dies nun möchte.

EDITIEREN - Rückgängig [Alt-BS]: Der letzte Vorgang kann evtl. noch rückgängig gemacht werden. Hierzu ist gleichzeitig die ALT- und Rückschritt-Taste zu drücken. Der Umweg über das Menü ist nicht immer empfehlenswert.
- Ausschneiden [Shift-Entf]: Der markierte Text wird in die Zwischenablage kopiert und aus dem Programm entfernt.
- Kopie [Strg-Einfg]: Der markierte Text wird in die Zwischenablage kopiert, bleibt aber im Programm.
- Einfügen [Shift-Einfg]: Der Text der Zwischenablage wird an der Kursorposition eingefügt.
- Löschen [Strg-Entf]: Der markierte Text wird gelöscht.
- Ablage: Die Windows-Zwischenablage wird aufgerufen. So kann man nachsehen, was da gerade drin ist.

SUCHEN - Finden: Sucht im aktuellen Programmfenster nach einem Text. Es kann angegeben werden, ob zwischen Groß- und Kleinschreibung zu

unterscheiden ist. Mit <F3> kann die Suche wiederholt werden.

- Ersetzen: Sucht im aktuellen Programmfenster nach einem Text und ersetzt ihn durch einen anderen. Es kann angegeben werden, ob Groß- und Kleinschreibung zu unterscheiden ist, ob im ganzen Text gesucht werden soll und ob vor jedem Ersetzen nachgefragt werden soll.
- Weitersuchen [F3]: Wiederholt die Suche.

PROFAN
- Ausführen: Führt ein PROFAN-Programm aus. Das Programm wird von der Festplatte gelesen. Ist ein noch nicht in der aktuellen Version abgespeichertes Programm im Editor, wird nachgefragt, ob man es zuvor abspeichern will. Das gilt auch für die nächsten zwei Menüpunkte.
- PRC-Datei erzeugen: Es wird ein PROFAN-Programm (PRF) compiliert. Der Compiler PROFCOMP.EXE muß sich im gleichen Verzeichnis wie der Editor befinden. Das Ergebnis ist eine Datei mit der Endung PRC, die vom Runtime-Modul PROFRUN oder jeder PROFAN-EXE-Datei ausgeführt werden kann. Die PRC-Datei ist in der Regel nur halb so groß, wie die PRF-Datei, dafür aber deutlich (!) schneller.
- EXE-Datei erzeugen: Eine PRC-Datei wird mit dem Runtime-Modul zu einer EXE-Datei gelinkt. Der Linker PROFLINK.EXE muß sich im gleichen verzeichnis wie der Editor befinden. Die EXE-Datei kann unter Windows selbstständig ausgeführt werden. Da sie das komplette RUNTIME-Modul enthält, kann sie für PRC-Dateien auch als Runtime-Modul dienen.

FENSTER
- Nebeneinander: Die verschiedenen Programmfenster im Editor werden nebeneinander, bzw. übereinander angeordnet, sodaß sie sich nicht überschneiden.
- Hintereinander: Die Programmfenster werden versetzt hintereinander angeordnet, sodaß von allen die Titelseite zu sehen ist.
- Symbole anordnen: Die Icons der verkleinerten Programmfenster werden in Reih' und Glied gebracht.
- Alle löschen: Alle Programmfenster werden geschlossen.
- Rechner: Der Windows-Taschenrechner wird für Berechnungen zwischendurch aufgerufen.
- Ansi-Tabelle: Eine Ansi-Tabelle wird aufgerufen. Zeichen können aus der Ansi-Tabelle übernommen werden. (Siehe Hilfetext der ANSI-Tabelle)
- Kalender: Der Windows-Kalender wird aufgerufen.
- 1 ... : Hier sind alle geöffneten Programme aufgelistet. Wenn man die Editorfenster auf auf optimale Größe vergrößert hat, kann diese Menüpunkte einfach zwischen den

Fenstern umgeschaltet werden.

HILFE - PROFAN-Hilfekartei: Der Windows-Karteikasten wird mit einer ausführlichen Hilfekartei aufgerufen. CARDFILE.EXE muß im Windows-Verzeichnis sein. Jeder Befehl, jede Funktion und jede Systemvariable wird ausführlich beschrieben. Gesucht werden kann mit <F4>. Die weiteren Funktionen der Hilfe-Kartei sind dem Windows-Handbuch (bzw. der Hypertext-Hilfe bei Multimediaversionen) zu entnehmen.
- Editor: Hilfetext zur Entwicklungsumgebung.
- Druckprogramm: Ein Hilfetext zu allen Menüpunkten des Druckprogrammes.
- Über ... : Copyright-Vermerk.

4 DRUCKPROGRAMM "RGH-DRUCK 1.3"

RGH-DRUCK ist ein eigenständiges Programm zum Drucken beliebiger Texte. Es enthält weitreichende Formatierungsmöglichkeiten.

Die Menüpunkte:

DATEI - Öffnen : Öffnet die zu druckende Datei. Diese wird in den Speicher gelesen und kann auch betrachtet werden. Änderungen sind jedoch nicht möglich. Lange Texte werden auch nur teilweise angezeigt, jedoch vollständig gedruckt.
- Drucken: Der Text wird im eingetellten Format mit der eingestellten Schrift gedruckt.
- Seitenformat: Zahlreiche Formatierungen können eingestellt werden: Ränder, Zeilenlineal, Zeilennummern (bei PROFAN-Listings sinnvoll), ANSI-CC-Byte (in der Regel nicht anzukreuzen), Datum, Zeit, Tabulatorgröße. Die Tabulatorgröße des PROFAN-Editors ist 8.
- Ende: Beendet das Druckprogramm.

TEXT - Schrift: Es kann eine Schriftart und Schriftgröße gewählt werden. Es ist ratsam eine schmale Schrift zu verwenden, da Zeilen, die nicht ganz in eine Druckzeile passen, einfach abgeschnitten werden.

Alle diese Funktionen können auch über die entsprechenden Buttons direkt angewählt werden. Die Bilder sind selbsterklärend.

Im System-Menü von RGH-DRUCK finden sich noch zwei weitere Menüpunkte:

Systemsteuerung - Da kann der Drucker (und anderes) eingestellt werden.
Über ... - Copyright-Hinweis. Hinweis auf das amerikanische Programm, dessen Quellcode Grund-

lage von RGH-Druck ist.

5 Operatoren

Diese gibt es in PROFAN nicht. Alles wird mittels Funktionen erledigt. Es gibt für alle gewohnten Operatoren entsprechende Funktionen. Statt "LET A%=A%+3" muß z.B. geschrieben werden: "LET A%=@ADD(A%,3)". Übrigens: Das "LET" darf nicht weggelassen werden.

Diese Philosophie wird auch konsequent bei Vergleichen eingesetzt, etwa: "IF A%=B%" wird zu "IF @EQU(A%,B%)".

6 Datentypen

Die vorliegende 1. Version von PROFAN kennt nur die Datentypen String (maximale Länge 255 Zeichen) und Integer (-32768 bis 32767). Für die Zwecke, für die PROFAN entwickelt wurde, ist dies völlig ausreichend. Stringkonstanten werden in " eingeschlossen: LET TEXT$="Mauer".
Möglichweise werden künftige Versionen weitere Datentypen aufweisen, etwa Long-Integer und Fließkommazahlen.

7 Arrays

Diese gibt es in PROFAN 1.x nicht in der z.B. von BASIC gewohnten Form. Es gibt lediglich ein Integer-Array mit max. 1000 Elementen (0 .. 999) und ein String-Array mit max. 1000 Elementen (0 .. 999). Dimensioniert werden die Arrays mit DIM bzw. DIM$. Gefüllt werden diese Strukturen mit dem LIST-Befehl und ausgelesen mit der @LIST$- bzw. @LIST-Funktion.

8 Programmaufbau

Alle Variablen in PROFAN müssen vorher (wie etwa in PASCAL) declariert werden. Die DECLARE- und DIM-Befehle sollten am Anfang des Programmes stehen. Vor der ersten Bildschirmausgabe sollte ein CLS stehen, um das Bildschirmfenster zu öffnen. (Nicht alle Befehle nehmen sich die Zeit, dies zu überprüfen, um dann selbst das Fenster zu öffnen.)

BEISPIEL:

```
DECLARE ...
DIM ...
DIM$ ...

WINDOWTITLE ...
CLS

...

END
```

9 Befehle über mehrere Zeilen

Der Übersichtlichkeit wegen können Befehle (Programmzeilen) auch über mehrere Bildschirmzeilen geschrieben werden. Ist das letzte Zeichen einer Zeile ein \ , so wird die nächste Zeile ab dem ersten Zeichen, das kein Leerzeichen ist, angehängt. Beispiele:

```
MESSAGEBOX "Überschrift",\
    "Das ist ein Testtext, der \
    etwas länger ist!",16

PRINT "Das ist ein ziemlich lang\
      er Text!"

COPYSIZEDBMP 34,34-16-16 > \
             12,12-32,32
```

Beachten Sie: Auch eine Programmzeile über mehrere Zeilen darf nicht länger als 127 Buchstaben sein. Führende und folgende Leerzeichen werden nicht gezählt.

10 Fehlermeldungen

Diese werden in deutsch in einer Messagebox auf den Bildschirm gebracht. Neben der Fehlerbeschreibung wird die fehlerhafte Zeile angegeben, sodaß der Fehler genau lokalisiert werden kann. ACHTUNG: Bei Dateioperationen wird die Systemvariable %IORESULT gesetzt, das Programm aber nicht abgebrochen. Das Programm muß diese Variable auswerten, da ansonsten Abstürze möglich sind.

11 Text- und Grafikmodus

PROFAN kennt (für Windows unüblich) einen Text- und einen Grafikmodus.

Der Grafikmodus ist der Windowseigene Modus, der nahezu all die Möglichkeiten bietet, die das Windows-Grafik-Interface beeinhaltet.

Der Textmodus simuliert im Hauptfenster einen DOS-Bildschirm, der zeichenorientiert ist.

Im Textmodus funktionieren die von BASIC her bekannten Befehle, wie z.B. PRINT, LOCATE, INPUT, COLOR und CLS in gewohnter Weise. Zusätzlich sind in diesem Modus Befehle wie TBOX, FONT und @TMOUSE verfügbar.

Es gibt keinen SCREEN-Befehl, um zwischen den Modi umzuschalten, da immer beide gleichzeitig verfügbar sind.

12 SYSTEMVARIABLEN

Es gibt zwei Typen von Systemvariablen:

$Name - Der Ergebnistyp ist ein String

%Name - Der Ergebnistyp ist ein Integer

$GETINPUT

Der aktuelle Text in der InputBox.
Beispiel:

```
InputBox,"Text",20,20,150
WaitMouse
Let Text$ = $GetInput
DestroyInputBox
```

Um die InputBox zu verlassen muß der MausKlick außerhalb der InputBox erfolgen.

$GETTEXT

Der angewählte Eintrag in einer ListBox. Ist kein Eintrag angewählt, ist das Ergebnis ein Leerstring. Die ListBox wird mir LISTBOX erzeugt und ADDFILES bzw. ADDSTRING gefüllt. Beispiel:

```
ListBox 20,20 - 100,150
AddFiles "C:\WINDOWS\*.EXE"
WaitMouse
Print "Ausgewählt: ";$GetText
```

$INPUT

Der zuletzt mit INPUT eingegebene String.

%BUTTON

Der in einer MESSAGEBOX gedrückte Knopf:

1 - OK
2 - Abbrechen (Cancel)
3 - Abbrechen (Abort)
4 - Wiederholen

5 - Ignorieren
6 - Ja
7 - Nein

%GETCOUNT

Die Anzahl der Einträge in der ListBox.
Beispiel:

```
ListBox 20,20 - 100,150
AddFiles "C:\WINDOWS\*:EXE"
Print %GetCount;" EXE-Dateien"
WaitMouse
Print "Ausgewählt: ";$GetText
DestroyListbox
```

%INPUT

Die zuletzt mit INPUT eingegebene Zahl.

%IORESULT

Nach einer Dateioperation oder einer
Directory-Suche enthält diese Variable
den entsprechenden Wert:

0 - kein Fehler aufgetreten

Die übrigen Werte entsprechenden den von
Turbo-Pascal her bekannten Ergebnissen:

2 - Datei nicht gefunden
3 - Pfad nicht gefunden
5 - Zugriff verweigert (ReadOnly?)
12 - Ungültiger Dateimodus
15 - Laufwerksnummer unzuzulässig
16 - Verzeichnis kann nicht gelöscht werden (noch Dateien drin?)
17 - RENAME nicht über Laufwerks-grenzen möglich (siehe: RENAME)
18 - Kein weiterer Eintrag (bei @FINDFIRST/@FINDNEXT)

100 - Lesefehler von Diskette/Platte
101 - Schreibfehler auf Diskette/Platte
102 - Dateinummer ist keiner Datei mit ASSIGN zugeordnet (siehe ASSIGN)
103 - Datei nicht offen (RESET, REWRITE oder APPEND fehlt)
104 - Datei nicht zum Lesen geöffnet (RESET fehlt)
105 - Datei nicht zum Schreiben geöffnet (REWRITE/APPEND fehlt)
106 - Falsches Format (bei INPUT #N,..)

%KEY

Der ANSI-Code der zuletzt gedrückten Taste.
Anwendung nach den Befehlen WAITKEY und WAITINPUT:

```
WaitInput

If @Equ(Key,90)
  Print "Du hast 'Z' gedrückt"
EndIf
```

%MENUITEM

Die Identifikationsnummer des zuletzt angewählten Menüpunktes. Wurde das Copyright-Zeichen angeklickt, ist die Nummer 254, wurde kein Menüpunkt angeklickt ist sie 0. Die Nummern 1 .. 253 können vom Programmierer verwandt werden.

%MOUSEKEY

Die zuletzt gedrückte Maustaste:

0 - keine Taste gedrückt
1 - linke Maustaste
2 - rechte Maustaste

%MOUSEX

Die aktuelle X-Position der Maus.
Anwendung besonders nach den Befehlen
WAITMOUSE und WAITINPUT.

%MOUSEY

Die aktuelle Y-Position der Maus.
Anwendung besonders nach den Befehlen
WAITMOUSE und WAITINPUT.

%PARCOUNT

Die Anzahl der beim Programmaufruf
übergebenen Kommandozeilen-Parameter.
Parameter 0 ist beim Interpreter der
Interpreter selber, bei der Runtime das
Runtime-Modul und bei einer EXE-Datei
die EXE-Datei. Parameter 1 ist bei
Interpreter und Runtime das ausgeführte
Programm, bei einer EXE-Datei nicht
vorhanden. Er muß das "!" sein. Weiter
Hinweise siehe unter @PAR$.

13 FUNKTIONEN

Alle Funktionen in PROFAN beginnen mit
einem @.

Funktionen, deren Funktionsnamen mit dem
$ endet, geben einen String zurück, die
übrigen Funktionen einen Integerwert.

@ABS(N)

N : Integer

Ergebnis: Integer

Absolutwert der Zahl N.

@ADD$(S1,S2)

S1 : String
S2 : String

Ergebnis: String

Verknüpfung der Strings S1 und S2. S2 wird an S1 angehängt.

@ADD(N1,N2)

N1 : Integer - Summand
N2 : Integer - Summand

Ergebnis: Integer

Die Summe von N1 und N2.

@AND(N1,N2)

N1 : Integer
N2 : Integer

Ergebnis: Integer

Die Werte von N1 und N2 werden mit der AND-Funktion verknüpft. Das Ergebnis ist 0 wenn ein oder beide Werte 0 sind.

@ANSITOOEM$(S)

S - String

Ergebnis: String

Der String S wird von ANSI-Code in den ASCII-Code (OEM) umgewandelt. Das betrifft insbesondere die deutschen Sonderzeichen.

@CHR$(N)

N - Integer (0 .. 255)

Ergebnis: String (1 Zeichen)

Das Zeichen mit dem ANSI-Code N.

@DEL$(S,N1,N2)

S : String
N1 : Integer - Position
N2 : Integer - Zeichenzahl

Ergebnis: String

Ab der Position N1 werden N2 Zeichen aus dem String entfernt.

@DISKFREE(S)

S : String - Laufwerksbezeichnung

Ergebnis: Integer

Das Ergebnis ist der freie Speicher auf Laufwerk S in kB dividiert durch 10. Ist das Ergebnis z.B. 43, so sind noch 430 kB frei. Ist das Laufwerk nicht vorhanden oder nicht lesbar, ist das Ergebnis 0. Beispiel:

```
Print @DISKFREE("C:");"0 kB frei"
```

@DISKSIZE(S)

S : String - Laufwerksbezeichnung

Ergebnis: Integer

Das Ergebnis ist der Gesamtspeicher auf Laufwerk S in kB dividiert durch 10. Ist das Ergebnis z.B. 4300, so hat das Laufwerk 43000 kB bzw. 43 MB. Ist das Laufwerk nicht vorhanden oder nicht lesbar, ist das Ergebnis 0. Beispiel:

```
Print @DISKSIZE("C:");"0 kB"
```

@DIV(N1,N2)

N1 : Integer
N2 : Integer

Ergebnis: Integer

Ganzzahliges Ergebnis der Division von N1 durch N2.

@EOF(N)

N : Integer - Dateinummer (1..8)

Ergebnis: Integer (0 oder 1)

Wenn mit dem letzten Lesen aus der Datei das Ende der Datei erreicht wurde, ergibt @EOF(N) den Wert 1, ansonsten ist es 0.

@EQU(N1,N2)

N1 : Integer - Wert 1
N2 : Integer - Wert 2

Ergebnis: Integer (0 oder 1)

Das Ergebnis ist WAHR (1), wenn beide Werte gleich sind, UNWAHR (0), wenn die Werte verschieden sind.

@FINDFIRST$(S)

S : String - Datei-Maske

Ergebnis: String

Die Funktion sucht die erste Datei im aktuellen Verzeichnis, die der Dateimaske entspricht. Das Ergebnis ist die gefundene Datei. Handelt es sich um ein Verzeichnis, steht es in eckigen Klammern. Beispiel:

```
Let Datei$ = @FindFirst$("*.PRF")
```

@FINDNEXT$()

Ergebnis: String

Findet die nächste Datei, die zur mit @FINDFIRST$(S) gegebenen Maske paßt. War die Suche erfolgreich, ist %IORESULT = 0! Beispiel:

```
Print @FindFirst$(*.PRF)
WhileNot %IOResult
  Print @FindNext$()
Wend
```

@GETKEY$()

Ergebnis: String (1 Zeichen)

Wartet auf einen Tastendruck und gibt das Ergebnis zurück.

@GT(N1,N2)

N1 : Integer - Wert 1
N2 : Integer - Wert 2

Ergebnis: Integer (0 oder 1)

Das Ergebnis ist WAHR (1), wenn Wert 1 größer als (Greater Than) Wert 2 ist.

@INKEY$()

Ergebnis: String (1 Zeichen)

Gibt als Ergebnis das Zeichen der aktuell gedrückten Taste zurück.

@INS$(S1,S2,N)

S1 : String
S2 : String
N : Integer

Ergerbnis: String

Der String S1 wird in S2 an Position N eingegefügt.

@INSTR(S1,S2)

S1 : String
S2 : String

Ergebnis: Integer

Das Ergebnis gibt an, an welcher Position S1 in S2 vorkommt. Kommt S1 in S2 nicht vor, ist das Ergebnis 0.

@KEYIN(S)

S : String

Ergebnis: Integer (0 oder 1)

Das Ergebnis ist dann 1, wenn die zuletzt gedrückte Taste im String S vorkommt. Beispiel:

```
WaitKey
Case @KeyIn("AaBb"):Print "A oder B"
```

@LEN(S)

S : String

Ergebnis: Integer

Länge des Strings S

@LIST$(N)

N : Integer - Index (0 .. 999)

Ergebnis: String

Das Ergebnis ist der N. Eintrag der Stringliste. Die Liste wird mit dem Befehl LIST$ gefüllt (siehe dort).

@LIST(N)

N : Integer - Index (0 .. 999)

Ergebnis: Integer

Das Ergebnis ist der N. Eintrag der Integerliste. Die Liste wird mit dem Befehl LIST gefüllt (siehe dort).

@LOADFILE$(S1,S2)

S1 : String - Überschrift
S2 : String - Dateimaske (mit Pfad)

Ergebnis: Dateiname (mit Pfad)

Es wird eine Dateiauswahlbox zum Laden (Öffnen) mit der Überschrift S1 geöffnet. Das Ergebnis ist die gewählte Datei (mit Pfad) die geöffnet werden kann. Beispiel:

```
LET NAME$=@LOADFILE$("ÖFFNE:","*.EXE")
```

@LT(N1,N2)

N1 : Integer - Wert 1
N2 : Integer - Wert 2

Ergebnis: Integer (0 oder 1)

Das Ergebnis ist WAHR (1), wenn Wert 1 kleiner als (Less Than) Wert 2 ist.

@MENUITEM(N)

N : Integer - Menü-Kennzeichen

Ergebnis: Integer (1 oder 0)

Wenn der aktuell gewählte Menüpunkt mit N (0 bis 254) übereinstimmt, ist das Ergebnis 1, anderenfalls ist es 0. Ist N gleich 0, dann ist das Ergebnis 1, wenn kein Menüpunkt angewählt wurde. 254 steht für die Anwahl des Copyright-Zeichens.

@MID$(S,N1,N2)

S : String
N1 : Integer - Position
N2 : Integer - Anzahl

Ergebnis: String

Das Ergebnis ist ein String, der N2 Zeichen ab Position N1 aus String S enthält.

@MOD(N1,N2)

N1 : Integer
N2 : Integer

Ergebnis: Integer

Das Ergebnis ist N1 modulo N2.

@MOUSE(X1,Y1-X2,Y2)

X1,Y1 : Integer - linke obere Ecke
X2,Y2 : Integer - rechte untere Ecke

Ergebnis: Integer (0 oder 1)

Das Ergebnis ist 1, wenn beim letzten Mausklick die Maus im angegeben Bereich war. Beispiel:

```
WaitMouse
Case @Mouse(10,10-60,60):Gosub "Info"
```

@MUL(N1,N2)

N1 : Integer
N2 : Integer

Ergebnis: Integer

Multiplikation von N1 mit N2.

@NEQ(N1,N2)

N1 : Integer - Wert 1
N2 : Integer - Wert 2

Ergebnis: Integer (0 oder 1)

Das Ergebnis ist UNWAHR (0), wenn beide Werte gleich sind, WAHR (1), wenn die Werte verschieden sind.

@NOT(N)

N : Integer

Ergebnis: Integer (0 oder 1)

Das Ergebnis ist 1, wenn N den Wert 0 hat, ansonsten ist es 0.

@OEMTOANSI$(S)

S - String

Ergebnis: String

Der String S wird von ASCII-Code (OEM) in den Windows-ANSI-Code umgewandelt. Das betrifft insbesondere die deutschen Sonderzeichen.

@OR(N1,N2)

N1 : Integer
N2 : Integer

Ergebnis: Integer

ODER-Funktion: Das Ergerbnis ist ungleich 0, wenn N1 ODER N2 (oder beide) ungleich 0 sind. Anders ausgedrückt: Es ist nur dann 0, wenn beide Argumente 0 sind.

@ORD(S)

S : String

Ergebnis: Integer

Der ANSI-Code des ersten Zeichens von S.

@PAR$(N)

N : Integer - Parameter-Nummer

Ergebnis: String - Parameter

Die Kommandozeilenparameter werden ausgelesen. Die Anzahl wird mit %PARCOUNT festgestellt.

Aufrufmöglichkeiten:

Interpreter:

```
PROFAN <name.prf> <par2> <par3> ...
```

Runtime (bzw. .EXE-Datei als Runtime):

```
PROFRUN <name.prc> <par2> <par3> ...
```

.EXE-Datei:

```
<name.exe> ! <par2> <par3> ...
```

Bei der .EXE-Datei muß der erste Parameter das Ausrufezeichen sein.

@RGB(R,G,B)

R,G,B : Integer - Farbwerte (0 .. 31)

Ergebnis : Integer (0 .. 32767)

Die Funktion errechnet eine der 32767 in Profan möglichen Farben aus den Farbanteilen Rot, Grün und Blau:

@RGB(0,0,0) = 0 -> Schwarz
@RGB(31,31,31) = 32767 -> Weiß

@RND(N)

N : Integer

Ergebnis: Integer

Zufallszahl zwischen 0 und N-1.

@SAVEFILE$(S1,S2)

S1 : String - Überschrift
S2 : String - Namensvorschlag

Ergebnis: Dateiname (incl. Pfad)

Es wird eine Dateiauswahlbox zum Speichern mit der Überschrift S1 geöffnet. Der Dateiname und/oder Pfad kann geändert werden. Das Ergebnis ist der Name (mit Pfad) unter dem gespeichert werden kann.

@STR$(N)

N : Integer

S : String

Der Wert von N wird in einen String ohne führende oder folgende Leerzeichen umgewandelt.

@SUB(N1,N2)

N1 : Integer
N2 : Integer

Ergebnis: Integer

N2 wird von N1 abgezogen (subtrahiert).

@TMOUSE(X1,Y1 - X2,Y2)

X1,Y1 : Integer - links oben
X2,Y2 : Integer - rechts unten

Ergebnis: Integer (0 oder 1)

War beim letzten Mausklick die Maus innerhalb des Rechteckes, ist das Ergebnis 1. Die Koordinaten sind Textkoordinaten (wie bei LOCATE oder TBOX).

@TRIM$(S)

S : String

Ergebnis: String

Die führenden und endenden Leerzeichen eines Strings werden abgeschnitten.

@UPPER$(S)

S : String

Ergebnis: String

Der String S wird komplett in Großbuchstaben umgewandelt, wobei die deutschen Umlaute korrekt berücksichtigt werden.

@VAL(S)

S : String

Ergebnis: Integer

Der String S wird in einen numerischen Wert umgewandelt. Das Ergebnis ist 0, wenn der String nichtnumerische Zeichen enthält.

14 BEFEHLE

' ... (Kommentar)

Kommentarzeichen. Im Gegensatz zu REM kann (sollte) es auch hinter einem Befehl in einer Zeile stehen:

' Unterprogramm

```
Let X%=125  'X-Position zuweisen
```

AddFiles S

S : String - Dateimaske

Mit diesem Befehl wird eine Dateiliste der geöffneten Listbox hinzugefügt.
Beispiel:

```
ListBox 100,100-100,100
AddFiles "*.RGH"
WaitInput
Let Wahl$ = $GetText
DestroyListbox
```

AddString S

S : String

Mit diesem Befehl wird der String der geöffneten Listbox hinzugefügt.
Beispiel:

```
ListBox 100,100-100,100
AddString "Item 1"
AddString "Item 2"
WaitInput
Let Wahl$ = $GetText
DestroyListbox
```

Append #N

N: Integer - Dateinummer (1 .. 8)

Die Datei #N wird geööfnet, um Daten anzufügen. Tritt ein Fehler auf, ist %IoResult größer als 0. Beispiel:

```
Assign #1,"TEST.DAT"
Append #1
Case %IoResult:MessageBox "E","!",16
Print #1,"Testdaten",56,Hugo$
Close #1
```

AppendMenu N,S

N : Integer - Menünummer (1 .. 253)
S : String - Eintrag

Dem PopUp-Menü wird der Eintrag S unter der Nummer N hinzugefügt. Ein vorangestelltes & sorgt für eine Unterstreichung. Ein Menüeintrag mit der Nummer 254 bleibt wirkungslos, 255 erzeugt immer die PROFAN-Copyright-Meldung.
Beispiel:

```
CreateMenu
AppendMenu 100,"&Laden"
AppendMenu 101,"&Speichern"
AppendMenu 102,"Speichern &als"
TrackMenu 20,130
Case @MenuItem(100):Goto "Laden"
Case @MenuItem(101):Goto "Speichern"
...
```

AppendMenuBar N,S

N : Integer - Menünummer (1 .. 253)
S : String - Eintrag

Dem Haupt-Menü wird der Eintrag S unter der Nummer N hinzugefügt. Ein vorangestelltes & sorgt für eine Unterstreichung. Beispiel:

```
AppendMenuBar 10,"&Laden"
AppendMenuBar 11,"S&peichern"
WaitMouse
Case %MenuItem(10):Goto "Laden"
```

Arc X1,Y1-X2,Y2; X3,Y3; X4,Y4

X1,Y1 : Integer - Links oben
X2,Y2 : Integer - Rechts unten
X3,Y3 : Start
X4,X5 : Ende

Es wird ein Kreisbogen gezeichnet. X1,Y1 und X2,Y2 bezeichnen das Rechteck, indem der Kreis sich befindet, die anderen Koordinaten Start und Ende des Kreisbogens.

Assign #N,S

N : Integer - Dateinummer (1 .. 8)
S : String - Dateiname (+ Pfad)

Der Datei Nummer N wird die Datei S zugewiesen. Beispiel:

```
Assign #1,"DEMO.RGH"
Reset #1
WhileNot @Eof(1)
  Input #1,Zeile$
  Print Zeile$
Wend
```

Beep

Gibt einen Signalton aus. Zur Tonausgabe siehe auch unter SOUND und PLAY!

Case N:BEF

N : Integer - Bedingungsausdruck
BEF : Profan Befehlszeile

Hat der Ausdruck N einen Wert <> 0, wird BEF ausgeführt, ansonsten geht die Programmausführung in die nächste Zeile. CASE entspricht einem einzeiligem IF-Befehl. Siehe auch unter CASENOT.

CaseNot N:BEF

N : Integer - Bedingungsausdruck
BEF : Profan Befehlszeile

Hat der Ausdruck N den Wert 0, wird BEF ausgeführt, ansonsten geht die Programmausführung in die nächste Zeile. CASENOT entspricht einem einzeiligem IFNOT-Befehl. Siehe auch unter CASE.

ChDir S

S : String - Pfadangabe

Es wird zum Verzeichnis S gewechselt. Enthält S eine Laufwerksangabe, so wird auch das aktuelle Laufwerk gewechselt. Beispiel:

```
ChDir "D:\TABELLEN\GRAFIK"
LoadSizedBmp "Umsatz",10,30-400,600;1
```

Chord X1,Y1-X2,Y2; X3,Y3; X4,Y4

X1,Y1 : Integer - Links oben
X2,Y2 : Integer - Rechts unten
X3,Y3 : Start
X4,X5 : Ende

Es wird ein Kreisabschnitt gezeichnet. X1,Y1 und X2,Y2 bezeichnen das Rechteck, indem der Kreis sich befindet, die anderen Koordinaten Start und Ende des Kreisabschnittes.

Close #N

N : Integer - Dateinummer (1 .. 8)

Die Datei mit der Dateinummer N wird geschlossen. Beispiel:

```
Close #Nr%
```

Cls

Löscht den Bildschirm.

Color C1,C2

C1 : Integer - Textfarbe (0 .. 15)
C2 : Integer - Hintegrund (0 .. 15)

Die Farben des Textmodus (PRINT, TBOX, INPUT, CLS) werden festgelegt. Die Farben entsprechen den 16 Farben von MS-DOS im Textmodus. Beispiel:

```
Color 15,0
Cls
Print "Hugo was here!"
```

Copy S1 > S2

S1 : String - Quelldatei
S2 : String - Zieldatei

Die Quelldatei wird kopiert und erhälz
Zieldatei als Namen. Beide Strings
können auch Pfadangaben enthalten.
Wildcards sind jedoch nicht gestattet.
Beispiel:

```
COPY "C:\START.BAT" > "D:\BAK\START.BAK"
```

CopyBmp X1,Y1-X2,Y2 > X3,Y3;N

X1,Y1 : Integer : Quelle rechts oben
X2,Y2 : Integer : Breite, Höhe
X3,Y3 : Integer : Ziel rechts oben
N : Integer : Kopiermodus

Der Bildschirmausschnitt der beginnend bei X1,Y1 eine Breite von X2 Pixel und eine Höhe von Y2 Pixel hat, wird an die Position X3,Y3 kopiert.
ACHTUNG: X2,Y2 sind keine absoluten Koordinaten sondern Breite und Höhe!

Kopiermodus:

0 - normales Kopieren
1 - Quelle und Ziel mit UND verknüpft
2 - Quelle und Ziel mit ODER verknüpft
3 - Quelle und Ziel mit XOR verknüpft
4 - Zielbereich invertieren
(Quellbereich wird nicht berücksichtigt)

CopySizedBmp X1,Y1-X2,Y2>X3,Y3-X4,Y4;N

X1,Y1 : Integer : Quelle rechts oben
X2,Y2 : Integer : Breite, Höhe
X3,Y3 : Integer : Ziel rechts oben
X4,Y4 : Integer : Breite, Höhe
N : Integer : Kopiermodus (siehe COPYBMP)

Der Bildschirmausschnitt an X1,Y1 mit der Größe X2,Y2 wird an die Position X3,Y3 in der Größe X4,Y4 kopiert.
ACHTUNG: X2,Y2,X4,Y4 sind keine absoluten Koordinaten sondern die Größe!

CreateMenu

Erzeugt ein PopUp-Menü. Das Menü wird mit APPENDMENU gefüllt und mit TRACKMENU abgefragt. Das Ergebnis steht in der Systemvariablen %MENUITEM.
Beispiele siehe unter APPENDMENU und TRACKMENU.

Declare VAR [,VAR[,VAR]...]

VAR : Variablen-Name

In PROFAN müssen alle Variablen (zeitlich) vor der ersten Benutzung declariert werden. String-Variablen werden durch ein $ und Integer-Variablen durch ein % gekennzeichnet. Beispiel:

```
Decalare XPos%,YPos%,Text$
```

DestroyInputBox

Die InputBox wird vom Bildschirm und aus dem Speicher gelöscht.

DestroyListBox

Die ListBox wird vom Bildschirm und aus dem Speicher gelöscht.

Dim N

N : Integer - Anzahl 0 .. 999

Festlegung der Größe des Integerarrays. Der Befehl darf nur einmal im Programm vorkommen und muß vor dem ersten Gebrauch einer Integer-Array-Variablen

erfolgen.

Beispiel:

```
DIM 56
LIST 56 = 789
```

Dim$ N

N : Integer - Anzahl 0 .. 999

Festlegung der Größe des Stringarrays. Der Befehl darf nur einmal im Programm vorkommen und muß vor dem ersten Gebrauch einer String-Array-Variablen erfolgen.

Beispiel:

```
DIM$ 56
LIST$ 56 = "Teststring"
```

DrawIcon S,X,Y

S : String : Icon-Name
X,Y : Integer : Koordinaten

Das Icon S wird an Position X,Y gezeichnet. Beispiel:

```
DrawIcon "Knopf1",12,12
```

Die 19 Icons sind in PROFAN.EXE enthalten, können aber z.B. mit einem Resourcen-Toolkit verändert werden.

Die Icons, die in Profan vorhanden sind:

PROFAN KNOPF1 EIS KNOPF2
2121 BAUM COMPUTER
DOS WEG TEXT GESICHT
EDITOR EIMER MUELL MUENZE
SAND STEIN WINDOWS WASSER

Es gibt auch Icon-Editoren, mit denen sie verändert werden können.

DrawSysIcon N,X,Y

N : Integer - Icon-Nummer (0 .. 4)
X,Y : Integer - Koordionaten

Das Windowseigene Icon mit der Nummer N wird an Position X,Y gezeichnet.

0 : leeres Rechteck
1 : STOP-Zeichen
2 : Fragezeichen
3 : Ausrufezeichen
4 : Info-Zeichen

DrawText X,Y,S

X,Y : Integer - Koordinaten
S : String - Text

Der Text S wird an der Bildschirmposition X,Y mit dem durch USEFONT eingestellten Font in der durch TEXTCOLOR eingestellten Farbe dargestellt.

Ellipse X1,Y1-X2,Y2

X1,Y1 : Integer - Links oben
X2,Y2 : Integer - Rechts unten

Es wird ein Kreis, bzw. eine Ellipse gezeichnet. X1,Y1 und X2,Y2 bezeichnen das Rechteck, indem die Ellipse sich befindet.

End

Programmende. Ohne weitere Abfrage wird das Profan-Programm und der Profan-Interpreter verlassen.

Erase #N

N : Integer - Dateinummer (1 .. 8)

Die Datei mit dem Dateikennzeichen N wird gelöscht. Beispiel:

```
ASSIGN #1,"TEST.BAK"
ERASE #1
```

Fill X,Y,C

X,Y : Integer - Koordinaten
C : Integer - RGB-Farbwert

Ausgend von Punkt X,Y wird alles bis zu einem Rahmen, der die Farbe C hat, mit dem aktuellen Pinsel (USEBRUSH) gefüllt. Beispiel:

```
Fill 20,50,@RGB(0,0,31)
```

Font N

N : Integer - Font - Nummer (0 .. 2)

Der Font N für den Textmodus (Textausgabe mit PRINT) wird gewählt:

0 : System-Font (ANSI-Zeichensatz)
1 : OEM-Font (ASCII-Zeichensatz)
2 : ANSI-Font

Gosub S ... Return

S : String - Unterprogramm-Label

Mit GOSUB wird zum Label S verzweigt. Bei RETURN gehts in der Zeile nach dem GOSUB weiter. Beispiel:

```
  Gosub "Unterprogramm"
  End

Unterprogramm:
  Print "Unterprogramm"
  Return
```

Goto S

S : String - Label

Das Programm verzweigt zum Label S. Ein Label wird durch einen Doppelpunkt gekennzeichnet (der in S nicht enthalten ist). Beispiel:

```
Nochmal:
  Input S%
  IfNot S%
    Goto "Nochmal"
  EndIf
```

If N ... EndIf

N : Integer . Bedingungsausdruck

Die zwischen IF und ENDIF stehenden Zeilen werden nur dann ausgeführt, wenn N ungleich 0 ist. Beispiel:

```
If @Equ(Test%,78)
  Print "Test% ist 78!"
  Print "-------------"
EndIf
```

IfNot N ... EndIf

N : Integer . Bedingungsausdruck

Die zwischen IF und ENDIF stehenden
Zeilen werden nur dann ausgeführt, wenn
N gleich 0 ist. Beispiel:

```
IfNot @Equ(Test%,78)
  Print "Test% ist nicht 78!"
  Print "--------------------"
EndIf
```

Input #N,VAR

N : Integer - Dateikennzeichen (1..8)
VAR : Variable

Aus der Datei #N wird ein Wert in die
Variable gelesen. Beispiel:

```
Assign #1,"TEST.DAT"
Reset #1
Input #1,Titel$
Input #1,Anzahl%
Close #1
```

Input VAR

VAR : Variable

Von der Tastatur wird ein Wert in die
Variable gelesen. (Textmodus.) Beispiel:

```
Print "Geben Sie den Titel ein: ";
Input Titel$
Print "Geben Sie die Anzahl ein: ";
Input Anzahl%
```

InputBox S,X,Y,L

S : String - Vorgabewert
X,Y : Integer - Position
L : Integer - Breite in Pixel

Es wird ein PopUp-Fenster geöffnet, in dem Text eingegeben werden kann. Der aktuell eingegebene Text steht in der Systemvariablen $INPUTTEXT.
ACHTUNG: X und Y sind absolute (!) Bildschirmkoordinaten und nicht relativ zum Fenster-Ursprung!

Let VAR={N|S}

VAR : Variablenname
N : Integer
S : String

Einer zuvor mit DECLARE declarierten Variablen wird ein Wert zugewiesen.
Einer numerischen Variablen kann nur ein numerischer Wert zugewiesen werden, einer Stringvariablen nur ein String.
Das LET muß (!) da stehen!

Line X1,Y1-X2,Y2

X1,Y1 : Integer - Startkoordinaten
X2,Y2 : Integer - Endkoordinaten

Es wird eine Linie zwischen den angegebenen Punkten in der mit USEPEN eingestellten Linienart und Strichstärke gezeichnet.

LineTo X1,Y1

X1,Y1 : Integer - Endkoordinaten

Es wird eine Linie vom zuletzt
gezeichneten Punkt zum angegebenen Punkt
in der mit USEPEN eingestellten
Linienart und Strichstärke gezeichnet.

List N1 = N2

N1 : Integer - Index (0 .. 999)
N2 : Integer - Wert

Der Wert N2 wird an Position N1 in die
Integerliste übernommen. Beispiel:

```
List 1 = 567
List 2 = 1045
Print @List(1),@List(2)
```

List$ N = S

N : Integer - Index (0 .. 999)
S : String

Der String S wird wird an Position N in
die Stringliste übernommen. Beispiel:

```
Dim$ 12
List$ 0 = "JANUAR"
List$ 2 = "FEBRUAR"
Print @List$(1),@List$(2)
```

ListBox X1,Y1 - X2,Y2

X1,X2 : Integer - Koordinaten
X2,Y2 : Integer - Größe

An der Position X1,Y1 wird eine ListBox geöffnet. Diese kann mit ADDSTRING und/oder ADDFILES gefüllt werden. Das Ergebnis kann mit $GETTEXT abgefragt werden.
ACHTUNG: X1 und Y1 sind absolute (!) Bildschirmkoordinaten und nicht relativ zum Fenster-Ursprung!

LoadBmp S, X,Y ;N

S : String - Name der Bilddatei
X,Y : Integer - Koordinaten
N : Integer - Kopiermodus (siehe COPYBMP)

S kann auch Pfadangaben enthalten. Das Bild muß im BMP-Format vorliegen. Es wird an der Position X,Y in Originalgröße angezeigt. Beispiel:

```
LoadBmp "C:\WINDOWS\PAPER.BMP",10,10;0
```

LoadSizedBmp S, X1,Y1-X2,Y2;N

S : String - Name der Bilddatei
X1,Y1 : Integer - Koordinaten
X2,Y2 : Integer - Größe
N : Integer - Kopiermodus (siehe COPYBMP)

S kann auch Pfadangaben enthalten. Das Bild muß im BMP-Format vorliegen. Es wird an der Position X1,Y1 in der angegebenen Größe angezeigt.Durch negative Werte für X2 bzw. Y2 sind auch Spiegelbilder realisierbar.

Locate X,Y

X : Integer - Zeile
Y : Integer - Spalte

Der nächste PRINT- oder INPUT-Befehl
positioniert auf Zeile X, Spalte Y.
Wirkt nur für Textmodus-Befehle

MessageBox S1,S2,N

S1 : String - Meldungstext
S2 : String - Überschrift
N : Integer - Art der MessageBox

Eine MessageBox wird auf dem Bildschirm
gebracht. N setzt sich zusammen aus
BUTTONS + ICON + DEFAULT + FENSTERART.
Beispiel:

```
MessageBox "Abbruch!","FEHLER",32
```

Werte für BUTTONS:
0 - OK
1 - OK Abbrechen
2 - Abbrechen Wiederholen Ignorieren
3 - Ja Nein Abbrechen
4 - Ja Nein
5 - Wiederholen Abbrechen

Werte für ICON:
0 - Kein Icon 48 "!"
16 - STOP 64 "i"
32 - "?"

Werte für DEFAULT:
Der Wert gibt an, welcher Knopf
defaultmäßig angewählt ist:
0 - erster Knopf
256 - zweiter Knopf
512 - dritter Knopf

Werte für FENSTERART:
0 - "normales" Fenster.
4096 - großes, nicht verschiebbares
"Fehler"-Fenster.

MkDir S

S : String - Pfadangabe

Das Verzeichnis S wird angelegt. Im Falle eines Fehlers hat %IORESULT einen von 0 verschiedenen Wert. Beispiel:

```
MkDir "C:\WINDOWS\TEST"
```

MoveTo X1,Y1

X1,Y2 : Integer - Koordinaten

Der Grafik-Kursor wird zum angegebenen Punkt bewegt, ohne eine Linie zu zeichnen. Praktisch ist dieser Befehl nur im Zusammenhang mit nachfolgendem DRAWTO-Befehl sinnvoll.

Pie X1,Y1-X2,Y2; X3,Y3; X4,Y4

X1,Y1 : Integer - Links oben
X2,Y2 : Integer - Rechts unten
X3,Y3 : Start
X4,X5 : Ende

Es wird ein Kreisteil (Kuchenstück) gezeichnet. X1,Y1 und X2,Y2 bezeichnen das Rechteck, indem der Kreis sich befindet, die anderen Koordinaten Start und Ende des Kreisteiles.

Play N1,N2,N3

N1 : Integer - Notenwert (0 = Pause)
N2 : Integer - Dauer
N3 : Integer - Punkte

Die Notenwerte werden in Halbtonschritten hochgezählt. Die Dauer wird reziprok angegeben: 4 bedeutet z.B. 1/4 Note; 2 = 1/2 Note. Im Beispiel wird eine halbe punktierte Note gespielt:

```
Play 24,2,1
```

Print #N,f;f,f ...

N : Integer - Dateikennzeichen (1..8)
f : druckbarer Ausdruck

Der Ausdruck, bzw. die Ausdrücke werden in die mit N bezeichnete Datei geschrieben. Steht zwischen den Ausdrücken ein Semikolon, so werden sie direkt aneinander gehängt, bei einem Komma wird ein Leerzeichen eingefügt. Siehe auch unter ASSIGN, REWRITE, APPEND und CLOSE.

Print f;f,f...

f : druckbarer Ausdruck

Der Ausdruck, bzw. die Ausdrücke werden auf den Bildschirm geschrieben. Steht zwischen den Ausdrücken ein Semikolon, so werden sie direkt aneinander gehängt, bei einem Komma wird ein Leerzeichen eingefügt. Siehe auch unter LOCATE. Beispiel:

```
Print "Dies ist der",N%;". Test"
```

Randomize

Sorgt dafür, daß die Zufallszahlen (@RND) auch wirklich zufälligen Charakter haben.

Rectangle X1,Y1-X2,Y2

X1,Y1 : Integer - Links oben
X2,Y2 : Integer - Rechts unten

Es wird ein Rechteck innerhalb der angegebenen Koordinaten gezeichnet.

Rem ...

Alles was dahinter steht, ist Kommentar. REM steht immer am Anfang der Kommentarzeile. Für Kommentare hinter anderen Befehlen ist das ' zu benutzen.

Rename #N,S

N : Integer - Dateikennzeichen (1..8)
S : String - neuer Name (mit Pfad)

Die Datei mit dem Kennzeichen N wird in S umbenannt. Enthält S eine Pfadangabe (mit oder ohne Laufwerks- bezeichnung), so wird sie verschoben. Verschieben ist nur innerhalb eines Laufwerks möglich. Beispiel:

```
Assign #3,"C:\TEST.DAT"
Rename #3,"C:\UTILS\TOAST.DAT"
```

RePaint

Malt den Bildschirm neu. Ist nur dann zu benutzen wenn bestimmte Befehlskombinationen (u.U. in Zusammenhang mit PopUp-Windows) den Bildschirm etwas durcheinander bringen.

Reset #N

N : Integer - Dateikennzeichen (1..8)

Die Datei N wird zum Lesen geöffnet. Beispiel:

```
Assign #2,"TEST.DAT"
Reset #2
WhileNot @Eof(2)
  Input #2,Zeile$
  Print Zeile$
Wend
```

Rewrite #N

N : Integer - Dateikennzeichen (1..8)

Die Datei N wird zum Schreiben geöffnet. Existiert sie nicht, wird sie erzeugt. Beispiel:

```
Assign #2,"C:\UTILS\TEST.DAT"
Rewrite #2
Print #2,Zeile$
Close #2
```

RmDir S

S : String - Verzeichnisnamen

Das Verzeichnis S wird gelöscht. Enthält es noch Dateien oder Unterverzeichnisse, kann es nicht gelöscht werden. Bei erfolgreicher Operation hat %IORESULT den Wert 0.

RoundRect X1,Y1-X2,Y2; X3,Y3

X1,Y1 : Integer - Links oben
X2,Y2 : Integer - Rechts unten
X3,Y3 : Integer - Rundung

Es wird ein abgerundetes Rechteck gezeichnet. X1,Y1 und X2,Y2 bezeichnen das Rechteck, die anderen Parameter den Grad der Rundung in waagrechter und senkrechter Richtung.

Run S

S : String - Programmname

Das Programm S wird aufgerufen und das aufrufende Profanprogramm beendet. Es können alle Windowsprogramme - auch mit Parametern - aufgerufen werden.
Beispiele:

```
Run "NOTEPAD HILFE.TXT"
Run "E:\PROFAN\PROFAN DEMO.RGH"
Run "WRITE"
```

Separator

Fügt dem PopUp-Menü eine Trennungslinie hinzu. Beispiel:

```
CreateMenu
AppendMenu 100,"&Neu"
AppendMenu 101,"&Laden"
AppendMenu 102,"&Speichern"
AppendMenu 103,"Speichern &als"
Separator
AppendMenu 104,"&Ende"
TrackMenu 50,50
```

SetPixel X,Y,C

X,Y : Integer - Koordinaten
C : Integer - RGB-Farbewert

Setzt an die Position X,Y ein Pixel in der Farbe C. Beispiel:

```
SetPixel 10,10,@RGB(13,20,3)
SetPixel 11,11,0
```

Shell S

S : String - Programmname

Das Programm S wird aufgerufen und das aufrufende Profanprogramm läuft weiter. Es können alle Windows- programme - auch mit Parametern - aufgerufen werden. Beispiel:

```
Shell "NOTEPAD HILFE.TXT"
```

Sound N1,N2

N1 : Integer - Frequenz in Hertz
N2 : Integer - Dauer in 18tel Sekunden

Es wird ein Ton mit angegebener Frequenz und Dauer ezeugt. Während der Tonerzeugung wird das Programm angehalten.

TBox X1,Y1 - X2,Y2; N

X1,Y1 : Integer - linke obere Ecke
X2,Y2 : Integer - rechte untere Ecke
N : Rahmentyp

Es wird im Textmodus eine Box gezeichnet. Benutzt werden die Grafikzeichen des OEM-Zeichensatzes.

0 - Einfacher Rahmen
1 - Doppelter Rahmen
2 - Senkrecht doppelt/waagrecht einf.
3 - Ohne Rahmen

TextColor C1,C2

C1 : Integer - Vordergrundfarbe (RGB)
C2 : Integer - Hintergrundfarbe (RGB)

Die Farben für die Ausgabe mittels DRAWTEXT werden festgelegt. Bei -1 als Hintergrundfarbe, ist diese transparant, d.h. der vorhandene Hintergrund wird genommen. Beispiel:

```
TextColor @RGB(15,15,31),-1
DrawText 23,67,"Hugo was here!"
```

TrackMenu X,Y

X,Y : Integer - Position

Das Programm zeigt das mit CREATEMENU und APPENDMENU erzeugte Menü an der gewünschten Position an und erwartet die Auswahl. Das Ergebnis steht in %MENUITEM. Beispiel:

```
TrackMenu 20,130
Case %MenuItem(100):Gosub "Speichern"
Case %MenuItem(120):End
```

UseBrush N,C

N : Integer - Art des Pinsels
C : Integer - Farbe des Pinsels (RGB)

De Pinsel wird benutzt, um die Grafiken (RECTANGLE, ROUNDRECT, etc.) auszufüllen. C ist die benutzte RGB-Farbe (0 ... 32767), N bezeichnet die Art des Pinsels:

0 : Transparant (nicht ausgefüllt)
1 : Solid (voll gefüllt)
2 .. 8 : verschiedene Schraffuren

Schraffurem:

2 - waagrecht =====
3 - senkrecht |||||
4 - diagonal \\\\\
5 - diagonal /////
6 - kariert #####
7 - diag.kar. XXXXX

Die Schraffur wird in der Farbe C ausgeführt.

UseCursor N

N : Integer - Cursortyp (0 .. 10)

Ändert das Aussehen des Mauszeigers. Es sind alle 11 von Windows vorgegebenen Mauszeiger verwendbar. Der Sanduhr-Zeiger sollte immer dann eingesetzt werden, wenn der Anwender warten muß und eh' keine Eingaben machen kann, z.B. beim Speichern und Laden.

Die Bedeutung von N:

0 - Zeiger (Standard)
1 - Texteingabe (senkr. Strich)
2 - Sanduhr (Bitte warten ...)
3 - Fadenkreuz
4 - Großer senkrechter Pfeil
5 - Vierfach-Pfeil (+)
6 - Icon (kleines Quadrat in Größerem
7 - Doppel-Pfeil (\)
8 - Doppel-Pfeil (/)
9 - Doppel-Pfeil (-)
10 - Doppel-Pfeil (|)

UseFont S,N1,N2,N3,N4,N5

S : String - Fontname
N1 : Integer - Zeichenbreite
N2 : Integer - Zeichenhöhe
N3 : Integer - Unterstrichen? (0 .. 1)
N4 : Integer - Kursiv? (0 .. 1)
N5 : Integer - Fett? (0 .. 1)

Der Font für die Ausgabe von Text mittels DRAWTEXT wird festgelegt.

N1: Zeichenhöhe. Steht hier 0, wird ein Defaultwert für Breite und Höhe genommen.
N2: Zeichenbreite. Steht hier 0, wird ein Defaultwert genommen
N3: 1 = fett , 0 = normal
N4: 1 = kursiv, 0 = normal
N5: 1 = unterstrichen, 0 = normal

UseIcon S

S : String - Name des Icons

Es wird das Icon ausgewählt, das für das Programm verwandt wird, wenn es verkleinert wird. Standardmäßig ist es das Profan-Icon. Es kann aber jedes der 19 "eingebauten" Icons verwandt werden. Die Namen der 19 Icons in Profan siehe unter DRAWICON.

UsePen N1,N2,C

N1 : Integer - Linienart (0 .. 5)
N2 : Integer - Linienstärke
C : Integer - RGB-Farbe

Die hier eingestellten Werte werden von den Zeichenbefehlen für die Rahmen bzw. Linien benutzt. Beispiel:

```
UsePen 1,5,@RGB(0,0,31)
Rectangle 10,10 - 150,100
```

Linienart:

0 - durchgezogen _____
1 - gestrichelt _ _ _ _ _ _
2 - gepunktet .....
3 - Strich-Punkt _._._._._.
4 - Strich-Punkt-Punkt _.._
5 - keine Linie

Die Werte 1 bis 5 sind nur bei einer Strichstärke von 1 sinnvoll.

WaitInput

Wartet auf einen Mausklick oder Tastendruck im Profan-Fenster. Das Ergebnis steht in den Systemvariablen %MOUSEKEY, %MOUSEX, %MOUSEY und %KEY.

WaitKey

Wartet auf einen Tastendruck. Das Ergebnis steht in der Systemvariablen %KEY.

WaitMouse

Wartet, bis eine Maustaste (im Profan-Fenster) gedrückt wird. Welche Taste es war, und die Position der Maus steht in den Systemvariablen %MOUSEKEY, %MOUSEX und %MOUSEY.

While N ... Wend

N : Integer - Bedingungsausduck

Die zwischen WHILE und WEND stehenden Befehle werden solange wiederholt, solange N ungleich 0 ist. Beispiel:

```
While N%
  Print "Gebe eine Zahl ein: ";
  Input N%
Wend
```

WhileNot N ... Wend

N : Integer - Bedingungsausduck

Die zwischen WHILE und WEND stehenden Befehle werden solange wiederholt,

solange N gleich 0 ist. Beispiel:

```
WhileNot @Eof(2)
  Input #2,Zeile$
  Print Zeile$
Wend
```

Window X1,Y1-X2,Y2

X1,Y1 : Integer - linke obere Ecke
X2,Y2 : Integer - Größe in Pixeln

Das Fenster wird in der angegebenen Größe geöffnet. X1 und Y1 sind absolute Bildschirmkoordinaten.

Wenn nichts anderes angeben wird, benutzt Profan den ganzen Bildschirm (640 * 480 Pixel).

WindowTitle S

S : String

Die Überschrift des Programmfensters wird festgelegt. Beispiel:

```
WindowTitle "Mein erstes Programm"
```

Wenn im Programm keine Überschrift festgelegt wird, wird eine voreingestellte Überschrift benutzt.
(Der Befehl existiert nur in der Vollversion!)

Anhang: Icon-Manager "Phish"

Dieser Icon-Manager ist Shareware und muß bei regelmäßigem Gebrauch beim Autoren registriert werden! Ich habe ihn dem PROFAN-Paket beigepackt, weil er sehr gut geeignet ist, die Icons in PROFAN.EXE bzw. PROFRUN.EXE zu verändern. Beachten Sie bitte:

- Sie können auch Icons in der fertigen EXE-Datei verändern, dürfen diese aber nicht um ein Byte vergrößern oder verkleinern!

- Es ist sinnvoll, die Icons im Runtime-Modul PROFRUN.EXE vor dem Linken zu ändern. Vor dem Linken dürfen dieser Datei mittels entsprechender Werkzeuge auch weitere Icons zugefügt werden. Diese können dann auch im Programm verwandt werden.

PhishEdit version 1.3

(C) Copyright 1992 Philip B. Eskelin, Jr. All rights reserved.
Windows is a trademark of Microsoft Corp.
(blah blah blah...<g>)

I am not going to say that this is the hottest icon editor out. Instead, I will tell you that I tried to design it so that it had an easy, intitive, graphical interface. The tool bar and status bar were to aid in efficiency. I named the editor "PhishEdit" (I know you are probably wondering what relevance it has to drawing icons...) mainly because it would be a catchy name, and I knew no one else would ever dream in their pseudo-zen imaginations of the same name. Every where I look, there is a massive saturation of things like "Icondraw", "Icon Manager", "Icon Tamer"(where did they get that???), "Iconthis", "Iconthat", "Icon*.*", etc. So, maybe PhishEdit will add a little spice to the world of informationatyourfingertips-icon-insanity.

I have noticed that it is a matter of time before a fellowship will be formed. I can see it now: "Hi, my name is Phil, and I am an Iconoholic...<cringe>." "I have just realized that I am powereless over Icons, and understand that I am insane and powerless over any graphical user interface." Suddenly, an insane rumble becomes evident: "Hi Phil, we're here to help you."

The most righteous part of PhishEdit is the ability to edit an icon with the PhishEdit window maximized, and the drawing window maximized. After being burnt out with the Icondraw eye-strain, denial broke, and PhishEdit became my higher power <grin>.

On the serious note, I tested PhishEdit to the best of my ability, in all resolutions. I fully developed it under debug Windows. It should be very solid, robust, and machine independent. Notify me of any bugs over compuserve mail [76216,1410]. Please beware when changing icons in any Windows executable or library. I suggest you back up the executable you change until you get use to using the feature of saving to executables/libraries. Have Fun!!

Usage:

File Menu

New - Opens a new untitled icon. It is not possible to create a new icon library or Windows executable with this option. A maximum of eight icons can be opened at one time. The icon opended by this command must be saved to a file name by way of the "Save As..." command from the File Menu.

Open - A dialog box is displayed which allows you to find an icon, library, or Windows executable. If it is just an icon, then the image is displayed in a new window for editing. If the file is a Windows executable or library, another dialog box is shown, so that you can pick which icon inside the executable or library to edit. From there, the icon image in the file is displayed, and can be edited and saved back into the same executable. I recommend that only advanced users who are accustomed to using Windows open/manipulate executables and libaries, as it can mean disaster if not used correctly.

Close - closes the currently active icon, prompting to save changes if any changes have been made. The currently active icon is the icon window that is up front, or has the focus.

Save - saves an icon that has been opened under the same file name. If the icon image was opened from a Windows executable or library, then the "Save" command writes over the existing image (For Example, if you open PROGMAN.EXE, and edit the 8th icon, then make changes to the icon, choosing "Save" writes over the 8th icon in the Program Manager with the new image).

Save As - Displays a dialog box allowing you to specify which file name to save the image into. BEWARE: if you create a new icon image, then save to a Windows executable or library file, the icon image will be written over the first icon in the executable or library.

Exit - Exits PhishEdit, and asks you whether or not to save changes over unsaved icon images.

Edit Menu

Undo - Cancels the previous drawing operation.

Clear - Clears the drawing area of the currently active icon image.

Copy - Copies the currently active icon image to the clipboard in bitmap format. The image can then be pasted into other icon drawing programs with clipboard support, or pasted into PaintBrush.

Cut - Equivalent to choosing "Copy" then "Clear" from the edit menu.

Paste - Takes a bitmap image from the clipboard, and pastes it into the drawing area of the currently active icon in PhishEdit.

Tools Menu

Pixel - draws one pixel at a time in the currently active icon image with the current color.

Line - draws a straight line (click and drag with the mouse to draw a line) with the current color.

Hollow Rectangle - draws a hollow rectangle with the border being of the current drawing color.

Solid Rectangle - draws a solid rectangle with the current drawing color.

Hollow Ellipse - draws a hollow ellipse or circle with the border being of the current drawing color.

Solid Ellipse - draws a solid ellipse or circle with the current drawing color.

Rotate 90 Degrees - Rotates the currently active icon image 90 degrees in the clockwise direction. Undo can be used to cancel this.

Shift Right - Shifts the icon image to the right by one pixel column.

Shift Left - Shifts the icon image to the left by one pixel column.

Shift Up - Shifts the icon image up by one pixel row.

Shift Down - Shifts the icon image down by one pixel row.

32x32 Capture - PhishEdit minimizes, and the cursor changes to a hollow box. When you click down with the left mouse button, everything inside the box cursor is copied into the drawing area of the current icon. If you make a mistake, Undo can be used. This is especially useful if you want to capture text.

Variable Capture - PhishEdit minimizes. You then click and drag with the left mouse button. When you up-click with the left mouse button, the image inside of the rectangle is copied into the currently active icon window inside PhishEdit.

Window Menu

Cascade - Cascades all opened drawing windows so that each one is offset down and to the right of the previous window.

Tile - tiles all opened drawing windows so that all of them can be seen at once.

Arrange Icons - Arranges all of the minimized drawing windows within PhishEdit at the lower-left area of the PhishEdit window.

File Listings - Below the "Cascade", "Tile", and "Arrange Icons" menu options, you will see a list of the drawing windows. Picking one of them brings the drawing window up to the front of the other windows and makes it the active drawing window.

Color Menu

Colors - You can change the current color in any of the three drawing

modes. Choosing a color in here is equivalent to choosing a color in the drop-down box in the tool bar.

Color Mode - The drawing mode where drawing in a color will result in the color being displayed normally when the icon is displayed.

Screen Mode - The same thing as "Transparent" mode. Whatever color you choose for this, will be the "invisible" part of the icon.

Inverse Mode - Whatever color you choose for this, will be the inverse part of the icon, so that when the icon is displayed, the inverse part is just the colors behind the icon inverted. The Inverse color will always be inverse of the "Transparent" or "Screen Mode" color.

Window Grid - Displays a grid in the currently active drawing area. Especially useful when you are trying to be accurate.

Window Coordinates - displays the logical coordinates of the mouse cursor when the mouse cursor is inside of the drawing window. This also aids in accuracy. The Window coordinates are displayed in the status bar at the bottom of PhishEdit's window.

Help Menu

Help on PhishEdit - Runs notepad and displays the README.TXT file you are reading right now.

About PhishEdit - displays about information and shows where to send the shareware registration (also discussed below).

Changing Program Manager Icons

Edit an icon, and save it to a file name. In Program Manager, highlight the group item you wish to change. Then, choose "Properties" from the File menu. Then, click on the "Change Icon" button. Next, specify the full path and file name of the icon you just saved. Click on OK, then on OK again, and the highlighted icon in Progman should change.

Using FileDragger and PhishEdit

If you have my shareware utility FileDragger, which is a file management utility (also available on Compuserve and other random BBS's), then you can select icons from the FileDragger file list and drag/drop them into PhishEdit. The icon files are then opened.

Registration:

If you find this useful, send 25 dollars (suggested) as a donation to my work, in either a check or money form. If you are outside of the U.S., send either cash or a money order in US dollars.

Please mail along with your checks any suggestions or comments about its interface and its abilities.

Please send to:

Philip B. Eskelin, Jr.
P.O. Box 4010
Silverdale, WA 98383-4010

Shareware outlets have my permission to distribute this,
as long as this text file is distributed with it, and
the executable remains unchanged.

P.S. There is a story behind the name "Phish"...

Anhang: PROFAN-Programme

A: DEMO.PRF

```
Declare x1%,x2%

WindowTitle "Profan für Windows - Demo"

AppendMenuBar 102,"&Aktivität"
AppendMenuBar 101,"&Ende"
Cls
UseBrush 1,@Rgb(0,31,15)
Rectangle 0,0-640,480
TextColor 0,-1
UseFont "Roman",48,0,1,0,0
DrawText 252,15,"Profan"
TextColor @Rgb(31,0,0),-1
DrawText 251,15,"Profan"
TextColor 0,-1
UseFont "System",0,0,0,0,0
DrawText 195,81,"Die Batchsprache mit Grafik \
                 und Text"
DrawText 300,96,"für"
DrawText 240,111,"Microsoft Windows 3.x"
DrawIcon "Knopf1",10,10
CopySizedBmp 10,10 - 32,32 > 10,10 - 54,32;0
DrawText 20,18,"Ende"
UseBrush 1,32767
Let x1% = 216
Let x2% = 435
RoundRect x1%,144 - x2%,180; 6,6
RoundRect x1%,189 - x2%,225; 6,6
RoundRect x1%,234 - x2%,270; 6,6
RoundRect x1%,279 - x2%,315; 6,6
RoundRect x1%,324 - x2%,360; 6,6
RoundRect x1%,369 - x2%,405; 6,6

UseFont "System",0,0,0,1,0
Let x1% = 249
Let x2% = 171
DrawIcon "PROFAN",x2%,144
DrawText x1%,153,"Der Schnelleinstieg"
DrawIcon "PROFAN",x2%,189
DrawText x1%,198,"Das Müllspiel"
DrawIcon "PROFAN",x2%,234
DrawText x1%,243,"'Hypertext'-Beispiel"
DrawIcon "PROFAN",x2%,279
DrawText x1%,288,"Farb-Palette"
DrawIcon "PROFAN",x2%,324
DrawText x1%,333,"Profan-Malprogramm"
DrawIcon "PROFAN",x2%,369
DrawText x1%,378,"Windows-Programme"

UseBrush 1,@RGB(31,31,16)
Rectangle 30,210 - 144,294
UseFont "System",0,0,0,0,0
DrawText 39,222,"Bitte einen"
DrawText 39,237,"Menüpunkt"
```

```
DrawText 39,252,"oder ein Icon"
DrawText 39,267,"anklicken!"
```

```
WaitForInput:

       WaitMouse

       Let x1% = 171
       Let x2% = 435
       Case @Mouse(x1%,144 - x2%,180):Goto "Intro"
       Case @Mouse(x1%,189 - x2%,225):Goto "Features"
       Case @Mouse(x1%,234 - x2%,270):Goto "Hyper"
       Case @Mouse(x1%,279 - x2%,315):Goto "Color"
       Case @Mouse(x1%,324 - x2%,360):Goto "Draw"
       Case @Mouse(x1%,369 - x2%,405):Goto "WinApps"
       Case @Mouse( 10, 10 -  65, 41):Goto "Ende"
       Case @MenuItem(0) :Gosub "NoHit"
       Case @MenuItem(101) :Goto "Ende"
       If @MenuItem(102)
         CreateMenu
         AppendMenu 200,"&Schnelleinstieg"
         AppendMenu 201,"&Müllspiel"
         AppendMenu 202,"&Hypertext"
         AppendMenu 203,"&Farb-Palette"
         AppendMenu 204,"Mal-&Programm"
         AppendMenu 205,"&Windows Programme"
         Separator
         AppendMenu 210,"&Ende"
         TrackMenu 20,42
         Case @MenuItem(200):Goto "Intro"
         Case @MenuItem(201):Goto "Features"
         Case @MenuItem(202):Goto "Hyper"
         Case @MenuItem(203):Goto "Color"
         Case @MenuItem(204):Goto "Draw"
         Case @MenuItem(205):Goto "WinApps"
         Case @MenuItem(210):Goto "Ende"
       EndIf
       Goto "WaitForInput"

NoHit:

       MessageBox "Menüpunkt oder Icon anklicken!",\
                  "HINWEIS:",16
       Return

Ende:

       MessageBox "Wollen Sie das Demo wirklich
beenden?",\
                  "FRAGE:",36
       If @Equ(%Button,6)
         End
       Endif
       Goto "WaitForInput"

Intro:
       Shell "DEMO.EXE INTRO.PRC"
       Goto "WaitForInput"

Features:

       Shell "DEMO.EXE MÜLLDEMO.PRC"
```

```
        Goto "WaitForInput"

Hyper:

        Shell "DEMO.EXE HYPER.PRC"
        Goto "WaitForInput"

Color:

        Shell "DEMO.EXE FARBEN.PRC"
        Goto  "WaitForInput"

Draw:

        Shell "DEMO DRAW.PRC"
        Goto "WaitForInput"

WinApps:

        Shell "DEMO WINAPPS.PRC"
        Goto "WaitForInput"
```

B: MÜLLSPIEL

```
windowtitle "DAS MÜLLSPIEL"
window 92,10-456,430

declare mx%,my%       'Mausposition
declare fx%,fy%       'Figurposition
declare px%,py%       'neue Müllposition
declare dx%,dy%       'Abstände
declare i%,j%         'Zählvariablen
declare anz%,vorg%    'Anzahl gesammelt/Vorgabe
declare ende%,ok%     'Endeschalter, Zug-Ok-Schalter
declare ton%          'Sound-Flag
declare f$,figur$
dim$ 12

appendmenubar 190,"&Ton"
appendmenubar 200,"&Ende"
appendmenubar 210,"&Hilfe"

list$  1="SSSSSSSSSSSSS"
list$  2="SBBBBBBBBBBBS"
list$  3="SBB   BB  GBBS"
list$  4="SBB M BB  MBBS"
list$  5="SBSS  BB    BS"
list$  6="SBB BBBBSSSSBS"
list$  7="SB     S    BS"
list$  8="SB S   S S  BS"
list$  9="SB SBM S BBBBS"
list$ 10="SB BB  S   EBS"
list$ 11="SBBBBBBBBBBBS"
list$ 12="SSSSSSSSSSSSS"

let i%=1
let j%=1
let ton%=1

gosub "SPIELFELD"

let ende% = 0
whilenot ende%
 waitinput
 repaint
 let mx%=@add(@div(%mousex,32),1)
 let my%=@add(@div(%mousey,32),1)
 case @menuitem(190):gosub "TON"
 case @menuitem(200):gosub "ENDE"
 case @menuitem(210):gosub "HILFE"
 case @menuitem(254):\
   messagebox "© 1992 Roland G. Hülsmann",\
              "DAS MÜLLSPIEL - DEMO",48
 case @menuitem(  0):gosub "SPIEL"
wend
usecursor 0
end

TON:

 let ton%=@sub(1,ton%)
```

```
 case ton%:messagebox "Der Ton ist EINgeschaltet","TON",48
 casenot ton%:messagebox "Der Ton ist \
                          AUSgeschaltet","TON",48
 return

SPIEL:

 let ok% = 1
 let dx% = @sub(mx%,fx%)
 let dy% = @sub(my%,fy%)
 let px% = @add(dx%,mx%)
 let py% = @add(dy%,my%)
 case @equ(@abs(dx%),@abs(dy%)):let ok% = 0
 case @gt(@add(@abs(dx%),@abs(dy%)),1):let ok% = 0
 ifnot ok%
   case ton%:beep
   messagebox "Da gehts nicht lang!","FEHLER!",16
   return
 endif
 let f$=@mid$(@list$(my%),mx%,1)
 case @instr(f$,"BE"):let ok% = 0
 if @instr(f$,"S")
   case @instr(@mid$(@list$(py%),px%,1),"BME"):let ok%=0
   if ok%
     case ton%:sound 170,7
     list$ py% = @del$(@list$(py%),px%,1)
     list$ py% = @ins$("S",@list$(py%),px%)
     list$ my% = @del$(@list$(my%),mx%,1)
     list$ my% = @ins$(" ",@list$(my%),mx%)
     drawicon "STEIN",@mul(@sub(px%,1),32),\
                      @mul(@sub(py%,1),32)
     drawicon "WEG",  @mul(@sub(mx%,1),32),\
                      @mul(@sub(my%,1),32)
   endif
 endif
 if @instr(f$,"M")
   case @instr(@mid$(@list$(py%),px%,1),"BSE"):let ok%=0
   if ok%
     case ton%:sound 70,7
     list$ py% = @del$(@list$(py%),px%,1)
     list$ py% = @ins$("M",@list$(py%),px%)
     list$ my% = @del$(@list$(my%),mx%,1)
     list$ my% = @ins$(" ",@list$(my%),mx%)
     drawicon "MUELL",@mul(@sub(px%,1),32),\
                      @mul(@sub(py%,1),32)
     drawicon "WEG",  @mul(@sub(mx%,1),32),\
                      @mul(@sub(my%,1),32)
   endif
   if @instr(@mid$(@list$(py%),px%,1),"E")
     if ton%
       sound 500,2
       sound 250,5
       sound 300,2
       sound 150,5
       sound 100,2
       sound  50,5
     endif
     let ok%=1
     let anz%=@add(anz%,1)
```

```
    list$ my% = @del$(@list$(my%),mx%,1)
    list$ my% = @ins$(" ",@list$(my%),mx%)
    drawicon "WEG", @mul(@sub(mx%,1),32),\
                    @mul(@sub(my%,1),32)
  endif
endif
```

```
 ifnot ok%
   case ton%:beep
   messagebox "Da kannst Du nicht hin!","FEHLER!",16
   return
 endif
 if ton%
   sound 200,2
   sound 100,2
 endif
 drawicon "GESICHT",@mul(@sub(mx%,1),32),\
                    @mul(@sub(my%,1),32)
 drawicon "WEG", @mul(@sub(fx%,1),32),\
                 @mul(@sub(fy%,1),32)
 let i%  = mx%
 let mx% = fx%
 let fx% = i%
 let i%  = my%
 let my% = fy%
 let fy% = i%
 if @equ(anz%,vorg%)
   copysizedbmp @mul(@sub(fx%,1),32),\
                @mul(@sub(fy%,1),32)-32,32\
                > 32,32-384,320;0
   if ton%
     play 16,2,0
     play 18,2,0
     play 20,2,0
     play 22,1,0
   endif
   messagebox "Du hast es geschafft!",\
              "G E W O N N E N !",48
   messagebox "Die Vollversion hat noch weitere 49
Level!",\
              "G E W O N N E N !",48
   let ende%=1
 endif
 return

ENDE:

 case ton%:beep
 messagebox "Willst Du wirklich aufhören?","SPIELENDE",36
 case @equ(%button,6):let ende%=1
 return

HILFE:

 let f$=@add$(@add$("DAS MÜLLSPIEL",@chr$(13)),@chr$(13))
 let f$=@add$(f$,"Ihre Aufgabe als Parkwächter ist es, den
\
                  Müll in den bereitstehenden Mülleimer zu
\
                  schieben. Der direkte Weg ")
 let f$=@add$(f$,"wird leider durch Bäume und Steine \
                  versperrt. Während ")
 let f$=@add$(f$,"die Bäume fest stehen, können die Steine
\
                  verschoben werden. ")
 messagebox f$,"ANLEITUNG 1/3",65
```

```
 case @equ(%button,2):return
 let f$=@add$(@add$("Bewegung:",@chr$(13)),@chr$(13))
 let f$=@add$(f$,"Sie können ein Feld waagrecht oder \
                  senkrecht laufen, nicht aber diagonal. \
                  Klicken Sie mit der Maus auf das ")
 let f$=@add$(f$,"Feld, auf das Sie wollen. Liegt dort \
                  Müll oder ein Stein, so wird dieser in \
                  Laufrichtung weitergeschoben. ")
 messagebox f$,"ANLEITUNG 2/3",65
 case @equ(%button,2):return
 let f$=@add$(@add$("Übrige
Funktionen:",@chr$(13)),@chr$(13))
 let f$=@add$(f$,"Über den Menüpunkt 'Ton' können Sie
den \
                  Sound an- und ausschalten. Mit 'Ende' \
                  können Sie das Spiel beenden. ")
 let f$=@add$(f$,@chr$(13))
 let f$=@add$(f$,"Das Spiel ist komplett in RGH-PROFAN \
                  geschrieben. ")
 messagebox f$,"ANLEITUNG 3/3",64
 return

SPIELFELD:

usecursor 2
let vorg%=0
let anz% =0
whilenot @gt(i%,12)
 whilenot @gt(j%,14)
   let f$=@mid$(@list$(i%),j%,1)
   case @instr(f$,"B"):let figur$="BAUM"
   case @instr(f$," "):let figur$="WEG"
   case @instr(f$,"S"):let figur$="STEIN"
   case @instr(f$,"E"):let figur$="EIMER"
   if @instr(f$,"M")
     let figur$="MUELL"
     let vorg%=@add(vorg%,1)
   endif
   if @instr(f$,"G")
     let figur$="GESICHT"
     let fx%=j%
     let fy%=i%
   endif
   drawicon figur$,@mul(@sub(j%,1),32),@mul(@sub(i%,1),32)
   let j%=@add(j%,1)
 wend
 let j%=1
 let i%=@add(i%,1)
wend
useicon "GESICHT"
usecursor 4
return
```

C: MALPROGRAMM

```
Declare WZeug%,WAlt%,WNeu%      ' Werkzeug, Altes, Neues
Declare W%,Ende%                ' Hilfsw., EndeFlag
Declare X1%,Y1%,X2%,Y2%         ' Koordinaten
Declare Farbe%,Rand%            ' Farben
Declare Linie%,Fuell%,Muster%   ' Linie, Füllfarbe, -
muster
Declare Error%,Wert%            ' Fehler, Wert
Declare BMax%,B%,S%,P%,M%       ' maximale Befehle, ...
Let BMax%=199
Declare Pal$,Bild$,B$           '
Paletten-,Bildname,Befehl
Dim 16                          ' Farbpalette
Dim$ BMax%                      ' Befehlsspeicher

WindowTitle "RGH-DRAW 0.4"
AppendMenubar 10,"&Datei"
AppendMenubar 20,"&Füllmuster"
AppendMenubar 30,"&Werkzeuge"
AppendMenubar 90,"&Hilfe"
Let Pal$="DRAW.PLT"
Gosub "Init"

Let Bild$="OHNENAME.BLD"
Let B%=0

Let Linie%=2
Let Farbe%=0
Let Fuell%=@RGB(0,0,31)
Let Rand%=0
Gosub "FarbTest"

Let Ende%=0
WhileNot Ende%
 WaitMouse
 Case @Mouse(40,10-630,400)  :Gosub "Malen"
 Case @Mouse(150,402-630,430):Gosub "Farbwahl"
 Case @MenuItem(10)          :Gosub "Datei"
 Case @MenuItem(20)          :Gosub "Muster"
 Case @MenuItem(30)          :Gosub "Werkzeuge"
 Case @MenuItem(90)          :Gosub "Hilfe"
 If @Mouse(5,5-37,421)
   Let WNeu%=@Div(@Sub(%MouseY,5),32)
   Gosub "Auswahl"
 EndIf
Wend

End

Datei:
 CreateMenu
 AppendMenu 11,"&Neues Bild"
 AppendMenu 12,"Bild &laden"
 AppendMenu 13,"Bild &speichern"
 Separator
 AppendMenu 14,"&Farbpalette laden"
 Separator
 AppendMenu 15,"Programm&ende"
```

```
 TrackMenu 30,42
 Case @MenuItem(11):Gosub "Löschen"
 Case @MenuItem(12):Gosub "BildLaden"
 Case @MenuItem(13):Gosub "BildSpeichern"
 Case @MenuItem(14):Gosub "Palette"
 Case @MenuItem(15):Gosub "Ende"
 Return

Palette:
 Let Pal$=@LoadFile$("Farbpalette laden:","*.PLT")
 Case @Len(Pal$):Gosub "Laden"
 Return

Werkzeuge:
 CreateMenu
 AppendMenu 39,"Rückgängig"
 Separator
 AppendMenu 31,"Ausfüllen"
 AppendMenu 32,"Linie"
 AppendMenu 33,"Rechteck leer"
 AppendMenu 34,"Rundeck leer"
 AppendMenu 35,"Ellipse leer"
 AppendMenu 36,"Rechteck gefüllt"
 AppendMenu 37,"Rundeck gefüllt"
 AppendMenu 38,"Ellipse gefüllt"
 TrackMenu 165,42
 Case %MenuItem:Let WNeu%=@Sub(%MenuItem,31)
 CaseNot %MenuItem:Let WNeu%=WZeug%
 Gosub "Auswahl"
 Return

Muster:
 CreateMenu
 AppendMenu 21,"gefüllt"
 AppendMenu 22,"waagrecht [====]"
 AppendMenu 23,"senkrecht  [||||||||]"
 AppendMenu 24,"diagonal   [\\\\\\\\]"
 AppendMenu 25,"diagonal   [////////]"
 AppendMenu 26,"kariert     [#####]"
 AppendMenu 27,"diag.kar.  [XXXX]"
 TrackMenu 80,42
 Case %MenuItem:Let Muster%=@Sub(%MenuItem,20)
 Gosub "FarbTest"
 Return

Init:
 Cls
 UsePen 0,1,0
 UseBrush 1,@RGB(23,23,23)
 Rectangle 0,0-640,480
 UseBrush 1,32767
 Rectangle 40,10-630,400
 UsePen 0,1,0
 UseBrush 1,@RGB(31,31,31)
 Rectangle 10,403-105,432
 Let WZeug%=0
 While @LT(WZeug%,12)
   Gosub "WerkAus"
   Let WZeug%=@Add(WZeug%,1)
```

```
Wend
Let WZeug%=1
Let Muster%=1
Let WAlt%=1
Gosub "WerkEin"
            Gosub "Laden"
Return
```

```
Malen:
 Case @GT(WZeug%,7):Return    'Kein Malwerkzeug
 UseCursor 3
 Let W%=WZeug%
 Let X1%=%MouseX
 Let Y1%=%MouseY
 Let M%=Muster%
 If @Equ(W%,0)                'Sonderfall: FILL-Funktion
   UsePen 0,1,Rand%           'Kein zweiter Mausklick
   UseBrush 0,Fuell%          'erforderlich
   Rectangle 40,10-630,400
   UseBrush M%,Fuell%
   Fill X1%,Y1%,Rand%
   UsePen 0,1,0
   UseBrush 0,32767
   Rectangle 40,10-630,400
   Gosub "Befehl"
   UseCursor 0
   Return
 EndIf
 While %MouseKey
 Wend
 UsePen 5,1,0
 UseBrush 1,@RGB(31,31,31)
 UseFont "System",0,0,0,0,0
 WhileNot %MouseKey
   Let X2%=%MouseX
   Let Y2%=%MouseY
   Case @Equ(W%,1):\
     CopyBmp @Sub(X1%,2),@Sub(Y1%,2) - 4,4 > \
             @Sub(X1%,2),@Sub(Y1%,2) ; 4
   CaseNot @Equ(W%,1):\
     CopyBmp X1%,Y1% - @Sub(X2%,X1%),\
             @Sub(Y2%,Y1%) > X1%,Y1% ; 4
   Let B$=@Str$(@Sub(X2%,X1%))
   Let B$=@Add$(B$,"/")
   Let B$=@Add$(B$,@Str$(@Sub(Y2%,Y1%))
   Rectangle 11,404-104,431
   DrawText 13,410,B$
   Case @Equ(W%,1):\
     CopyBmp @Sub(X1%,2),@Sub(Y1%,2) - 4,4 >\
             @Sub(X1%,2),@Sub(Y1%,2) ; 4
   CaseNot @Equ(W%,1):\
     CopyBmp X1%,Y1% - @Sub(X2%,X1%),@Sub(Y2%,Y1%) > \
             X1%,Y1% ; 4
 Wend
 Rectangle 11,404-104,431
 If @Equ(%Mousekey,2)           'Rechte Taste gedrückt?
   Return
 Endif
 Let M%=0
 If @GT(W%,4)
   Let M%=Muster%
   Let W%=@Sub(W%,3)
 EndIf
 UseBrush M%,Fuell%
 UsePen 0,Linie%,Rand%
```

```
 If @Mouse(40,10-630,400)
   Case @Equ(W%,1):Line X1%,Y1%-%MouseX,%MouseY
   Case @Equ(W%,2):Rectangle X1%,Y1%-%MouseX,%MouseY
   Case @Equ(W%,3):RoundRect X1%,Y1%-%MouseX,%MouseY;20,20
   Case @Equ(W%,4):Ellipse X1%,Y1%-%MouseX,%MouseY
   Gosub "Befehl"
 EndIf
 UseCursor 0
 Return

Befehl:
 Let B$=""
 Let B$=@Add$(B$,@Str$(W%))
 Let B$=@Add$(B$,"|")
 Let B$=@Add$(B$,@Str$(X1%))
 Let B$=@Add$(B$,"|")
 Let B$=@Add$(B$,@Str$(Y1%))
 Let B$=@Add$(B$,"|")
 Let B$=@Add$(B$,@Str$(%MouseX))
 Let B$=@Add$(B$,"|")
 Let B$=@Add$(B$,@Str$(%MouseY))
 Let B$=@Add$(B$,"|")
 Let B$=@Add$(B$,@Str$(Fuell%))
 Let B$=@Add$(B$,"|")
 Let B$=@Add$(B$,@Str$(Rand%))
 Let B$=@Add$(B$,"|")
 Let B$=@Add$(B$,@Str$(M%))
 Let B$=@Add$(B$,"|")
 Let B%=@Add(B%,1)
 List$ B%=B$
 Return

Wiedergabe:
 UseCursor 2
 UsePen 0,1,0
 UseBrush 1,32767
 Rectangle 40,10-630,400
 Let S%=0
 While @LT(S%,B%)
   Let S%=@Add(S%,1)
   Let B$=@List$(S%)
   Gosub "LesePara"
   Let W%=Wert%
   Gosub "LesePara"
   Let X1%=Wert%
   Gosub "LesePara"
   Let Y1%=Wert%
   Gosub "LesePara"
   Let X2%=Wert%
   Gosub "LesePara"
   Let Y2%=Wert%
   Gosub "LesePara"
   Let Fuell%=Wert%
   Gosub "LesePara"
   Let Rand%=Wert%
   Gosub "LesePara"
   Let M%=Wert%
```

```
    If @Equ(W%,0)                'Sonderfall: FILL-Funktion
      UsePen 0,1,Rand%
      UseBrush 0,Fuell%
      Rectangle 40,10-630,400
      UseBrush M%,Fuell%
      Fill X1%,Y1%,Rand%
      UsePen 0,1,0
      UseBrush 0,32767
      Rectangle 40,10-630,400
    EndIf
    IfNot @Equ(W%,0)
      UseBrush M%,Fuell%
      UsePen 0,Linie%,Rand%
      Case @Equ(W%,1):Line X1%,Y1%-X2%,Y2%
      Case @Equ(W%,2):Rectangle X1%,Y1%-X2%,Y2%
      Case @Equ(W%,3):RoundRect X1%,Y1%-X2%,Y2%;20,20
      Case @Equ(W%,4):Ellipse X1%,Y1%-X2%,Y2%
    EndIf
  Wend
  Gosub "Farbtest"
  UseCursor 0
  Return

LesePara:
  Let P%=@Instr("|",B$)
  IfNot P%
    Messagebox "Keine DRAW-Datei","FEHLER",16
    Let S%=B%
  Endif
  If P%
    Let Wert%=@Val(@Mid$(B$,1,@Sub(P%,1)))
    Let B$=@Mid$(B$,@Add(P%,1),255)
  EndIf
  Return

Auswahl:
  Let WAlt%=WZeug%
  Gosub "WerkAus"
  Let WZeug%=WNeu%
  Gosub "WerkEin"
  Case @Equ(WZeug%,11):Gosub "BildLaden"
  Case @Equ(WZeug%,10):Gosub "BildSpeichern"
  Case @Equ(WZeug%, 9):Gosub "Löschen"
  If @Equ(WZeug%, 8)
    If @GT(B%,0))
      Let B%=@Sub(B%,1)
      Gosub "Wiedergabe"
    EndIf
    Gosub "WerkAus"
    Let WZeug%=WAlt%
    Gosub "WerkEin"
  EndIf
  Return

BildSpeichern:
  Let Bild$=@SaveFile$("Bild speichern:",Bild$)
```

```
 If @Len(Bild$)
   Let S%=0
   Assign #1,Bild$
   Rewrite #1
   IfNot %IoResult
     While @LT(S%,B%)
       Let S%=@Add(S%,1)
       Print #1,@List$(S%)
     Wend
   EndIf
   Close #1
 EndIf
 Gosub "WerkAus"
 Let WZeug%=WAlt%
 Gosub "WerkEin"
 Return

BildLaden:
 Let Bild$=@LoadFile$("Bild laden:","*.BLD")
 If @Len(Bild$)
   Let S%=0
   Assign #1,Bild$
   Reset #1
   IfNot %IoResult
     WhileNot @Eof(1)
       Input #1,B$
       Let S%=@Add(S%,1)
       List$ S%=B$
     Wend
     Close #1
     Let B%=S%
     RePaint
     Gosub "Wiedergabe"
   EndIf
 EndIf
 Gosub "WerkAus"
 Let WZeug%=WAlt%
 Gosub "WerkEin"
 Return

Ende:
 Messagebox "Willst Du wirklich aufhören?","RGH-DRAW",36
 Case @Equ(%Button,6):Let Ende%=1
 Return

Löschen:
 Messagebox "Willst Du das Bild wirklich löschen?",\
            "RGH-DRAW",36
 If @Equ(%Button,6)
   UsePen 0,1,0
   UseBrush 1,32767
   Rectangle 40,10-630,400
   Let B%=0
 EndIf
 Gosub "WerkAus"
 Let WZeug%=WAlt%
 Gosub "WerkEin"
 Return
```

```
WerkAus:
 DrawIcon "Knopf1",5,@Add(@Mul(WZeug%,32),5)
 Gosub "Werkzeug"
 Return

WerkEin:
 DrawIcon "Knopf2",5,@Add(@Mul(WZeug%,32),5)
 Gosub "Werkzeug"
 Return

Werkzeug:
 UseFont "System",0,0,0,0,0
 TextColor 0,-1
 UseBrush 0,0
 UsePen 0,1,0
 Case @Equ(WZeug%,0):DrawText 11,13,"Fill"
 Case @Equ(WZeug%,1):Line 12,43-30,61
 Case @Equ(WZeug%,2):Rectangle 12,75-30,93
 Case @Equ(WZeug%,3):RoundRect 12,107-30,125;10,10
 Case @Equ(WZeug%,4):Ellipse 12,139-30,157
 UseBrush 1,0
 Case @Equ(WZeug%,5):Rectangle 12,171-30,189
 Case @Equ(WZeug%,6):RoundRect 12,203-30,221;10,10
 Case @Equ(WZeug%,7):Ellipse 12,235-30,253
 Case @Equ(WZeug%,8):DrawText 8,270,"DEL"
 Case @Equ(WZeug%,9):DrawText 8,302,"CLS"
 Case @Equ(WZeug%,10):DrawText 10,334,"->O"
 Case @Equ(WZeug%,11):DrawText 10,367,"O->"
 Return

Laden:
 UseCursor 2
 Assign #1,Pal$
 Reset #1
 Let Error% = %IoResult
 Case Error%:\
   Messagebox @Add$("Palette: ",Pal$),"FARBPALETTE \
                    FEHLT!",16
 IfNot Error%
   Let Farbe%=0
   While @Lt(Farbe%,16)
     Input #1,WERT%
     List Farbe% = Wert%
     Let X1%=@Add(@Mul(Farbe%,30),150)
     UsePen 0,1,0
     UseBrush 1,Wert%
     Rectangle X1%,402 - @Add(X1%,30),430
     Let Farbe% = @Add(Farbe%,1)
   Wend
   Close #1
 EndIf
 UseCursor 0
 Return

Farbwahl:
 Let Wert% = @Div(@Sub(%MouseX,150),30)
 Case @Equ(%MouseKey,1):Let Fuell%=@List(Wert%)
 Case @Equ(%MouseKey,2):Let Rand%=@List(Wert%)
 Gosub "FarbTest"
```

```
 Return

FarbTest:
 UseBrush 1,32767
 UsePen 0,4,Rand%
 RoundRect 110,405-145,430;30,30
 UseBrush Muster%,Fuell%
 RoundRect 110,405-145,430;30,30
 Return

Hilfe:
 CreateMenu
 AppendMenu 91,"&Icon-Leiste"
 AppendMenu 92,"&Größen-Fenster"
 AppendMenu 93,"&Farben"
 Separator
 AppendMenu 94,"&Über..."
 TrackMenu 250,42
 Case @MenuItem(91):Gosub "Hilfe1"
 Case @MenuItem(92):Gosub "Hilfe2"
 Case @MenuItem(93):Gosub "Hilfe3"
 Case @MenuItem(94):Gosub "Hilfe4"
 Return

Hilfe1:
 Let B$="Die Bedeutung der Icons von oben nach unten: \
         Ausfüllen - Linie - leere Formen - gefüllte \
         Formen - Rückgängig - "
 Let B$=@Add$(B$,"Bild löschen - Bild speichern - Bild \
                  laden")
 MessageBox B$,"WERKZEUG-ICONS",64
 Return

Hilfe2:
 Let B$="Im Größenfenster wird die Größe des aktuell zu \
         zeichnenden Objektes angegeben, und zwar waag\
         rechte Ausdehnung/"
 Let B$=@Add$(B$,"senkrechte Ausdehnung. Positive Werte \
                  gehen nach rechts bzw. unten, negative \
                  nach links bzw. oben.")
 MessageBox B$,"GRÖSSEN-FENSTER",64
 Return

Hilfe3:
 Let B$="Die Farbleiste zeigt die zu Verfügung stehenden \
         Farben der geladenen Palette an, der Farbtopf die
\
         aktuelle Farb"
 Let B$=@Add$(B$,"kombination. Die Randfarbe wird mit
der \
                  rechten und die Füllfarbe mit der linken
\
                  Maustaste gewählt.")
 MessageBox B$,"FARBEN",64
 Return

Hilfe4:
 Let B$=@Add$("©1992 Roland G. Hülsmann",@Chr$(13))
 Let B$=@Add$(B$,@Chr$(13))
```

```
Let B$=@Add$(B$,"PROFAN-Demoprogramm:")
Let B$=@Add$(B$,@Chr$(13))
Let B$=@Add$(B$,"RGH-DRAW 0.4")
MessageBox B$,"RGH-DRAW - DEMO",64
Return
```

D: FARBKASTEN

```
WindowTitle "Der PROFAN-FARBKASTEN V 1.0"
Cls

Declare Ende%,Farbe%,X%,Y%,R%,G%,B%,Wert%,Error%,Name$
Dim 16

AppendMenuBar 80,"&Laden"
AppendMenuBar 90,"&Speichern"
AppendMenuBar 99,"&Ende"

UseBrush 1,@RGB(0,0,15)
Rectangle 0,0 - 640,480
UseBrush 1,@RGB(15,15,31)
UsePen 0,4,0
RoundRect 20,20 - 620,200;20,20
UseBrush 1,32767
RoundRect 20,300 - 70,350;10,10
Let Farbe%=0
Let X%=25
Let Y%=28
Let R%=0
Let G%=0
Let B%=0
Gosub "Hilfetext"
Let Name$ = "STANDARD"
Gosub "Laden"

Let Ende% = 0
WhileNot Ende%
 UseFont "System",0,0,0,0,0
 TextColor 32767,-1
 UsePen 0,4,0
 UseBrush 1,@RGB(31,0,0)
 Rectangle 200,300 - 250,320
 UseBrush 1,@RGB(0,31,0)
 Rectangle 200,320 - 250,340
 UseBrush 1,@RGB(0,0,31)
 Rectangle 200,340 - 250,360
 DrawText 205,302,@Ins$(@Str$(R%),"-   +",3)
 DrawText 205,322,@Ins$(@Str$(G%),"-   +",3)
 DrawText 205,342,@Ins$(@Str$(B%),"-   +",3)
 UseBrush 1,@RGB(R%,G%,B%)
 RoundRect 20,300 - 70,350;10,10
 WaitMouse
 Case @MenuItem(80):Gosub "Dateiwahl"
 Case @MenuItem(90):Gosub "Speichern"
 Case @MenuItem(99):Let Ende% = 1
 Case @Mouse(200,300 - 215,320):Gosub "Rotminus"
 Case @Mouse(200,320 - 215,340):Gosub "Grünminus"
 Case @Mouse(200,340 - 215,360):Gosub "Blauminus"
 Case @Mouse(235,300 - 250,320):Gosub "Rotplus"
 Case @Mouse(235,320 - 250,340):Gosub "Grünplus"
 Case @Mouse(235,340 - 250,360):Gosub "Blauplus"
 Case @Mouse( 20, 20 - 620,200):Gosub "Plazieren"
Wend
End
```

```
Plazieren:

 Case @Mouse( 25, 28 - 95, 98):Let X% = 0
 Case @Mouse( 99, 28 -169, 98):Let X% = 1
 Case @Mouse(173, 28 -243, 98):Let X% = 2
 Case @Mouse(247, 28 -317, 98):Let X% = 3
 Case @Mouse(321, 28 -391, 98):Let X% = 4
 Case @Mouse(395, 28 -465, 98):Let X% = 5
 Case @Mouse(469, 28 -539, 98):Let X% = 6
 Case @Mouse(543, 28 -613, 98):Let X% = 7
 Case @Mouse( 25,120 - 95,190):Let X% = 8
 Case @Mouse( 99,120 -169,190):Let X% = 9
 Case @Mouse(173,120 -243,190):Let X% =10
 Case @Mouse(247,120 -317,190):Let X% =11
 Case @Mouse(321,120 -391,190):Let X% =12
 Case @Mouse(395,120 -465,190):Let X% =13
 Case @Mouse(469,120 -539,190):Let X% =14
 Case @Mouse(543,120 -613,190):Let X% =15
 Let Farbe%=X%
 Let Y%=28
 If @Gt(Farbe%,7)
   Let Y%=120
   Let X%=@Sub(X%,8)
 EndIf
 Let X%=@Add(@Mul(X%,74),25)
 UsePen 0,2,32767
 UseBrush 1,@RGB(R%,G%,B%)
 Ellipse X%,Y% - @Add(X%,70),@ADD(Y%,70)
 List Farbe% = @RGB(R%,G%,B%)
 UsePen 0,4,0
 Return

Speichern:

 UseFont "System",0,0,0,0,0
 TextColor 32767,-1
 DrawText 20,210,"Bitte geben Sie den Dateinamen ein:"
 InputBox Name$,20,280,100
 DrawIcon "KNOPF1",117,236
 DrawText 122,244,"Ok"
 WaitInput
 WhileNot @Mouse(115,236 - 146,267
   WaitInput
 Wend
 UseCursor 2
 UseFont "System",0,0,0,0,0
 TextColor 32767,-1
 DrawIcon "KNOPF2",117,236
 DrawText 120,244,"Ok"
 Let Name$ = $GetInput
 Assign #1,@Add$(Name$,".PLT")
 Let Error% = %IoResult
 Case Error%:\
   MessageBox "Ungültiger Dateiname!","FEHLER",16
 IfNot Error%
              Rewrite #1
              Let Farbe% = 0
              WhileNot @Equ(Farbe%,16)
                       Print #1,@List(Farbe%)
```

```
          Let Farbe% = @Add(Farbe%,1)
Wend
Close #1
```

```
 EndIf
 DestroyInputBox
 UsePen @RGB(0,0,15),0,0
 UseBrush 1,@RGB(0,0,15)
 Rectangle 15,210 - 300,270
 UseFont "System",0,0,0,0,0
 TextColor 32767,-1
 UseCursor 0
 Return

Dateiwahl:

 UseFont "System",0,0,0,0,0
 TextColor 32767,-1
 DrawText 20,210,"Bitte wählen Sie die Palette:"
 ListBox 20,280-100,140
 AddFiles "*.PLT"
 DrawIcon "KNOPF1",117,236
 DrawText 122,244,"Ok"
 WhileNot @Mouse(115,236 - 146,267)
   WaitInput
 Wend
 UseFont "System",0,0,0,0,0
 TextColor 32767,-1
 DrawIcon "KNOPF2",117,236
 DrawText 120,244,"Ok"
 Let Name$ = $GetText
 If @Instr(".",Name$)
   Let X% = @Instr(".",Name$)
   Let Name$ = @Del$(Name$,X%,4)
   Gosub "Laden"
 EndIf
 DestroyListbox
 UsePen @RGB(0,0,15),0,0
 UseBrush 1,@RGB(0,0,15)
 Rectangle 15,210 - 300,270
 UseFont "System",0,0,0,0,0
 TextColor 32767,-1
 Return

Laden:

 UseCursor 2
 Assign #1,@Add$(Name$,".PLT")
 Case %IoResult:Messagebox "Dateiname falsch!","FEHLER",16
 Reset #1
 Let Error% = %IoResult
 Case Error%:Messagebox "Datei nicht
gefunden!","FEHLER",16
 IfNot Error%
   Let Y%=28
   Let Farbe%=0
   While @Lt(Farbe%,16)
     Input #1,WERT%
     List Farbe% = Wert%
     Let X%=Farbe%
     If @Gt(Farbe%,7)
       Let X% = @Sub(Farbe%,8)
       Let Y% = 120
```

```
     EndIf
     Let X%=@Add(@Mul(X%,74),25)
     UsePen 0,2,32767
     UseBrush 1,Wert%
     Ellipse X%,Y% - @Add(X%,70),@ADD(Y%,70)
     Let Farbe% = @Add(Farbe%,1)
   Wend
   Close #1
 EndIf
 UseCursor 0
 Return

Rotminus:

 Case R%:Let R%=@Sub(R%,1)
 Return

Grünminus:

 Case G%:Let G%=@Sub(G%,1)
 Return

Blauminus:

 Case B%:Let B%=@Sub(B%,1)
 Return

Rotplus:

 IfNot @Equ(R%,31)
   Let R%=@Add(R%,1)
 Endif
 Return

Grünplus:

 IfNot @Equ(G%,31)
   Let G%=@Add(G%,1)
 EndIf
 Return

Blauplus:

 IfNot @Equ(B%,31)
   Let B%=@Add(B%,1)
 EndIf
 Return

Hilfetext:

 UseBrush 1,@RGB(31,31,15)
 Rectangle 300,220 - 620,400
 UseFont "System",0,0,0,0,0
 TextColor 0,-1
 DrawText 305,224,"1. Die gewünschte Farbe durch anklicken
\
                  von"
 DrawText 322,236,"[+] oder [-] auf nebenstehenden
Farbfeld"
```

```
 DrawText 322,248,"einstellen."
 DrawText 305,268,"2. Ist die Farbe optimal, den
Farbtopf \
                  für die"
 DrawText 322,280,"Farbe anklicken."
 DrawText 305,300,"3. Den fertigen Farbkasten abspeichern.
\
                  Namen"
```

```
DrawText 322,312,"ohne Erweiterung angeben und [Ok] \
                  anlicken."
DrawText 322,324,"Die Palette 'STANDARD' wird beim"
DrawText 322,336,"Programmstart automatisch geladen."
DrawText 305,356,"4. Um eine abgespeicherte Palette zu \
                  laden,"
DrawText 322,368,"den gewünschten Namen in der Liste"
DrawText 322,380,"anwählen und [Ok] anklicken."
Return
```

E: HYPERTEXT

```
        Declare Meldung$,Cr$
        Let Cr$=@Chr$(13)
        WindowTitle "HYPERTEXT"

Boat:

        Cls
        LoadBmp "BOAT1.BMP",2,20;0
        UseBrush 1,@RGB(22,22,22)
        TextColor @RGB(0,0,31),-1

        RoundRect 110,16-190,36;10,10
        UseFont "System",1,3,0,0,1
        DrawText 116,19,"A"
        UseFont "System",1,3,0,0,0
        DrawText 124,19,"bout..."

        RoundRect 20,270-70,290;10,10
        UseFont "System",1,3,0,0,1
        DrawText 26,273,"E"
        UseFont "System",1,3,0,0,0
        DrawText 34,273,"xit"

        RoundRect 20,300-180,320;10,10
        UseFont "System",1,3,0,0,1
        DrawText 26,303,"M"
        UseFont "System",1,3,0,0,0
        DrawText 34,303,"echanical Systems"

        RoundRect 20,330-210,350;10,10
        UseFont "System",1,3,0,0,1
        DrawText 26,333,"D"
        UseFont "System",1,3,0,0,0
        DrawText 34,333,"eck & Cockit Framing"

        RoundRect 20,360-240,380;10,10
        UseFont "System",1,3,0,0,1
        DrawText 26,363,"H"
        UseFont "System",1,3,0,0,0
        DrawText 34,363,"ull Framing and Planking"

        TextColor 0,-1
        UseFont "Helv",18,0,0,0,0
        DrawText 300,363,"©1991 Roland G. Hülsmann \
                          (Profan Version)"
        UseFont "System",1,3,0,0,0

WaitForInput:

        WaitInput
        Case @KeyIn("Aa"):Goto "About"
        Case @KeyIn("Ee"):Goto "Ende"
        Case @KeyIn("Mm"):Goto "Mechanics"
        Case @KeyIn("Dd"):Goto "Cockpit"
        Case @KeyIn("Hh"):Goto "Frame"
        Case @KeyIn("Bb"):Goto "Boat"
        Case @Mouse(110,16-190,36):Goto "About"
```

```
        Case @Mouse(20,270-70,290):Goto "Ende"
        Case @Mouse(20,300-180,320):Goto "Mechanics"
        Case @Mouse(20,330-210,350):Goto "Cockpit"
        Case @Mouse(20,360-240,380):Goto "Frame"
        Case @Mouse(520,330-570,350):Goto "Boat"
        Goto "WaitForInput"

Ende:

        Cls
        End

About:

        Let Meldung$=@Add$("IRENE",Cr$)
        Let Meldung$=@Add$(Meldung$,\
                           "Designed by John L. Hacker")
        Let Meldung$=@Add$(Meldung$,Cr$)
        Let Meldung$=@Add$(Meldung$,"Length: 28 feet")
        Let Meldung$=@Add$(Meldung$,Cr$)
        Let Meldung$=@Add$(Meldung$,"Draft: 2 feet")
        Let Meldung$=@Add$(Meldung$,Cr$)
        Let Meldung$=@Add$(Meldung$,"Beam: 6 feet 8
inches")
        Let Meldung$=@Add$(Meldung$,Cr$)
        Let Meldung$=@Add$(Meldung$,"Speed: 22 mph")
        MessageBox Meldung$,"About...",64
        Goto "WaitForInput"

Mechanics:

        Cls
        LoadBmp "MECH1.BMP",2,20;0
        UseBrush 1,@RGB(0,31,31)
        TextColor @RGB(0,0,31),-1

        RoundRect 520,330-570,350;10,10
        UseFont "System",1,3,0,0,1
        DrawText 526,333,"B"
        UseFont "System",1,3,0,0,0
        DrawText 534,333,"ack"

        TextColor 0,-1
        UseFont "Helv",18,0,0,0,0
        DrawText 300,363,"©1991 Roland G. Hülsmann \
                          (Profan Version)"
        UseFont "System",1,3,0,0,0

        Goto "WaitForInput"

Cockpit:

        Cls
        UseBrush 1,@RGB(0,31,31)
        LoadBmp "COCKPIT1.BMP",2,20;0
        TextColor @RGB(0,0,31),-1

        RoundRect 520,330-570,350;10,10
        UseFont "System",1,3,0,0,1
```

```
        DrawText 526,333,"B"
        UseFont "System",1,3,0,0,0
        DrawText 534,333,"ack"

        TextColor 0,-1
        UseFont "Helv",18,0,0,0,0
        DrawText 300,363,"©1991 Roland G. Hülsmann \
                         (Profan Version)"
        UseFont "System",1,3,0,0,0

        Goto "WaitForInput"

Frame:

        Cls
        UseBrush 1,@RGB(0,31,31)
        LoadBmp "FRAME1.BMP",0,0;0
        TextColor @RGB(0,0,31),-1

        RoundRect 520,330-570,350;10,10
        UseFont "System",1,3,0,0,1
        DrawText 526,333,"B"
        UseFont "System",1,3,0,0,0
        DrawText 534,333,"ack"

        TextColor 0,-1
        UseFont "Helv",18,0,0,0,0
        DrawText 300,363,"©1991 Roland G. Hülsmann \
                         (Profan Version)"
        UseFont "System",1,3,0,0,0

        Goto "WaitForInput"
```

F: WINDOWS-MENÜ

```
        Declare x1%,x2%
        WindowTitle "Starte Windows-Programme"

        Cls
        UseBrush 1,@Rgb(31,0,15)
        Rectangle 0,0-640,480
        TextColor @RGB(15,0,15),-1
        UseFont "Helv",48,0,1,1,0
        DrawText 251,30,"Starte"
        DrawText 132,75,"Windows-Programme"
        TextColor @RGB(7,0,0),-1
        UseFont "Helv",48,0,1,0,0
        DrawText 255,30,"Starte"
        DrawText 136,75,"Windows-Programme"
        TextColor 0,-1
        UseFont "System",0,0,0,0,0
        DrawIcon "Knopf1",10,10
        CopySizedBmp 10,10 - 32,32 > 10,10 - 64,32;0
        DrawText 20,18,"Zurück"

        UseBrush 1,32767
```

```
Let x1% = 216
Let x2% = 435
RoundRect x1%,144 - x2%,180; 6,6
RoundRect x1%,189 - x2%,225; 6,6
RoundRect x1%,234 - x2%,270; 6,6
RoundRect x1%,279 - x2%,315; 6,6
RoundRect x1%,324 - x2%,360; 6,6
RoundRect x1%,369 - x2%,405; 6,6
```

```
        UseFont "System",0,0,0,1,0
        Let x1% = 249
        Let x2% = 171
        DrawIcon "WINDOWS",x2%,144
        DrawText x1%,153,"Notizblock"
        DrawIcon "WINDOWS",x2%,189
        DrawText x1%,198,"Windows Write"
        DrawIcon "WINDOWS",x2%,234
        DrawText x1%,243,"Paintbrush"
        DrawIcon "WINDOWS",x2%,279
        DrawText x1%,288,"Profan Hilfe-Kartei"
        DrawIcon "WINDOWS",x2%,324
        DrawText x1%,333,"Rechner"
        DrawIcon "WINDOWS",x2%,369
        DrawText x1%,378,"System-Editor"

        UseBrush 1,@RGB(31,31,16)
        Rectangle 30,210 - 144,294
        UseFont "System",0,0,0,0,0
        DrawText 39,222,"Bitte einen"
        DrawText 39,237,"Menüpunkt"
        DrawText 39,252,"oder ein Icon"
        DrawText 39,267,"anklicken!"

WaitForInput:

        WaitMouse

        Let x1% = 171
        Let x2% = 435
        Case @Mouse(x1%,144 - x2%,180):Goto "Notepad"
        Case @Mouse(x1%,189 - x2%,225):Goto "Write"
        Case @Mouse(x1%,234 - x2%,270):Goto "Paintbrush"
        Case @Mouse(x1%,279 - x2%,315):Goto "Cardfile"
        Case @Mouse(x1%,324 - x2%,360):Goto "Calculator"
        Case @Mouse(x1%,369 - x2%,405):Goto "SysEdit"
        Case @Mouse( 10, 10 -  65, 41):Goto "Ende"
        Case @MenuItem(0)             :Gosub "NoHit"
        Goto "WaitForInput"

NoHit:

        MessageBox "Bitte einen Menüpunkt anwählen!",\
                   "HINWEIS:",16
        Return

Ende:

        End

Notepad:
        Shell "NOTEPAD.EXE"
        Goto "WaitForInput"

Write:
        Shell @Add$("WRITE.EXE ",\
              @LoadFile$("WRITE-Datei laden","*.WRI"))
        Goto "WaitForInput"
```

```
Paintbrush:

       Shell "PBRUSH.EXE"
       Goto "WaitForInput"

Cardfile:

       Shell "CARDFILE.EXE PROFAN.CRD"
       Goto  "WaitForInput"

Calculator:

       Shell "CALC.EXE"
       Goto "WaitForInput"

SysEdit:

       Shell "SYSEDIT.EXE"
       Goto "WaitForInput"
```