Stołeczny Ośrodek
Elektronicznej
Techniki Obliczeniowej

INFORMATYKA mikrokomputerowa

STEFAN NAWROCKI

GFA BASIC

PROGRAMOWANIE ATARI ST
W GFA-BASIC'u

ATARI

Warszawa 1988

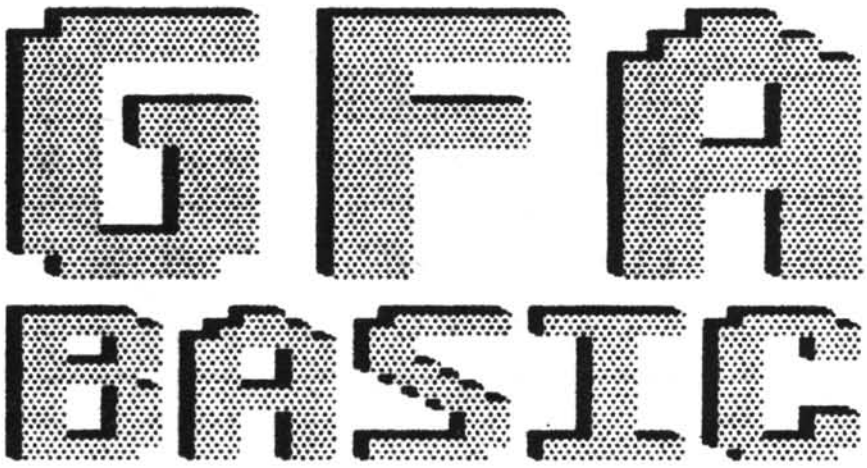
Książka przeznaczona jest dla użytkowników ATARI ST. Komputer ten nie posiada wbudowanego języka programowania dzięki czemu istnieje duża swoboda w jego wyborze. Jedną z alternatyw jest opisane w niniejszej książce GFA-BASIC, produkt zachodnioniemieckiej firmy GFA SYSTEMTECHNIK GmbH /twórcą tego języka jest Frank Ostrowski/.

GFA-BASIC posiada wiele atrakcyjnych cech, dzięki którym z przyjemnością sięgamy po to narzędzie. Wspomniane wyżej cechy to:

- możliwość pracy z interpreterem
- możliwość kompilacji programu
- programowanie strukturalne /linie nienumerowane, procedury z parametrami, zmienne lokalne/
- wygodny edytor /operacje na blokach/
- duża szybkość wykonywania programu
- łatwy dostęp do GEM-u /alert, menu/
- możliwość wywołań funkcji systemowych /GEM dos/, BIOS, XBIOS, Gemsys, Vdisys
- bogaty zestaw operacji graficznych a zwłaszcza grafika rastrowa /GET, IPUT/
- obsługa SPRITE'ów

Książka zawiera opis klawiszy funkcyjnych i specjalnych obsługujących edytor, wykaz operatorów i słów zarezerwowanych, opis wszystkich komend i funkcji języka a także krótki słowniczek wyjaśniający pojęcia obcoję - zyczne oraz kilka przykładowych programów.

STEFAN NAWROCKI



PROGRAMOWANIE ATARI ST

W GFA-BASIC'u

SPIS TREŚCI

1. WPROWADZENIE

GFA-BASIC jest językiem strukturalnym posiadającym szybki interpreter oraz kompilator tworzący programy z rozszerzeniem .PRG, które można uruchamiać bezpośrednio z TOS-u.

Niniejsza instrukcja dotyczy zestawu: Edytor-interpreter V 2.0 US Copyright 1986 GFA Systemtechnik GmBH oraz kompilator V 1.71 1987 tej samej firmy (oferowany przez producenta zestaw zawiera ponadto GFABASRO.PRG - GFA BASIC RUN ONLY służący wyłącznie do uruchamiania nieskompilowanych programów).

Edytor GFA-BASICA pracuje w trybie 80 kolumn (w każdej z rozdzielczości) i jest edytorem pełnoekranowym, posiadającym możliwość akceptacji linii w dowolnym miejscu.
Kursor można przesuwać klawiszami sterującymi lub może być on przesunięty w dowolne miejsce za pomocą myszy.
Linie programu nie są numerowane, wywoływanie podprogramów (procedur) ewentualnie skoki odbywają się według etykiet.
Procedury mogą zawierać zmienne lokalne oraz mogą być wywoływane z parametrami.

Edytor posiada wiele funkcji wspomagających edycję programu:
- przesuwanie i kopiowanie zaznaczonego fragmentu programu,
- wyszukiwanie dowolnego łańcucha,
- wymiana dowolnego łańcucha na inny.

W kolejnych rozdziałach opisano: klawisze specjalne i funkcyjne, operatory i słowa zarezerwowane, instrukcje edytora, błędy interpretera i kompilatora, opis kompilatora, kody ASCII znaków dostępnych z klawiatury oraz sposób ich uzyskiwania, przykładowe programy oraz wykaz skrótów i pojęć obcojęzycznych.

2. KLAWISZE SPECJALNE I FUNKCYJNE

Interpreter GFA-BASIC'a może pracować w dwóch trybach: edycji i komend. Po załadowaniu GFA-BASIC'a znajdujemy się w trybie edycji. W trybie tym klawisze specjalne pełnią następujące funkcje: (^ - oznacza CONTROL, + - oznacza SHIFT)

→	- ruch kursora w prawo
←	- ruch kursora w lewo
↑	- ruch kursora w górę
↓	- ruch kursora w dół
^ →	- przesuwa kursor na koniec linii
^ ←	- przesuwa kursor na początek linii
^ ↑	- strona w górę
^ ↓	- strona w dół
Clr Home	- przesuwa kursor na początek strony
Esc	- przejście do trybu komend
Backspace	- usuwa znak z lewej strony kursora
Delete	- usuwa znak spod kursora
Insert	- wstawia linię powyżej kursora
Undo	- przywraca ostatnią zaakceptowaną postać linii
Tab	- przesuwa kursor do następnej pozycji tabulacji w prawo
^Delete	- usuwa cała linię
^Tab	- przesuwa kursor do następnej pozycji tabulacji w lewo
^Clr Home	- przesuwa kursor na początek programu
^F	- przesuwa kursor do linii zawierającej łańcuch podany w opcji Find
^R	- wymienia łańcuch w przypadku aktywnej opcji Replace
^Z	- przesuwa kursor na koniec programu
^S	- przełącza klawiaturę na drugi zestaw znaków
^Shift Alt	- przerywa wykonywanie programu

Klawisze funkcyjne opisane są na górze ekranu:

```
Save¦Save,A¦Quit ¦New  ¦Blk S¦Replace¦ Pg up ¦Text 8¦Direct¦Run
Load¦Merge ¦Llist¦Block¦Blk E¦ Find  ¦Pg down¦Insert¦ Flip ¦Test
```

F1 (Load)	- patrz komenda Load
F2 (Merge)	- dołącza zbiory typu ASCII (zapisane komendą Save,A)
F3 (Llist)	- patrz komenda Llist
F4 (Block)	- wywołuje następujące menu:

"C"opy "M"ove "W"rite "L"list "S"tart "E"nd "^D"el "H"ide

Copy	- kopiuje blok w miejsce zaznaczone kursorem

Move - przesuwa blok w miejsce zaznaczone kursorem
Write - zapisuje blok na dysku w kodzie ASCII
Llist - drukuje blok na drukarce
Start - działa jak Blk Sta
End - działa jak Blk End
Del - usuwa blok
Hide - usuwa linie oznaczające blok
Funkcje te można wywoływać wyróżnionymi klawiszami.
F5 (Blk End)- miejsce oznaczone kursorem zostanie przyjęte jako koniec bloku
F6 (Find) - wyszukuje zadany łańcuch w programie (szukany łańcuch wpisujemy w miejsce wskazywane kursorem)
F7 (Pg down)- przesuwa stronę w dół
F8 (Insert) - zmiana trybu Insert (dopisywanie) na tryb Overwrite (usuwanie poprzedniego tekstu) i odwrotnie
F9 (Flip) - wywołanie strony komend (nie trybu komend)
F10(Test) - sprawdzenie prawidłowości zapisu programu bez jego uruchamiania
+F1 (Save) - patrz komenda Save
+F2 (Save,A)- zapisuje program na dysku jako zbiór ASCII
+F3 (Quit) - powrót do TOS-u
+F4 (New) - patrz komenda New
+F5 (Bl Sta)- miejsce oznaczone kursorem zostanie przyjęte jako początek bloku
+F6 (Repl.) - wymienia zadany łańcuch na inny (patrz ^R)
+F7 (Pg up) - przesuwa stronę w górę
+F8 (Text 8)- przełącznik 23/46 wierszy (tylko w trybie High)
+F9 (Direct)- przejście do trybu komend
+F10(Run) - uruchamia program

Wszystkie pozycje Menu można wywołać myszą.

W trybie komend klawisze specjalne pełnią następujące funkcje:
→ - ruch kursora w prawo
← - ruch kursora w lewo
↑ - przesunięcie kursora na początek linii
↓ - przesunięcie kursora na koniec linii
Esc+ Return - przejście do trybu edycji
Backspace - usuwa znak z lewej strony kursora
Delete - usuwa znak spod kursora
Undo - przywraca ostatnią zaakceptowaną postać linii
^Shift Alt - przejście do trybu edycji

Klawisze funkcyjne w trybie komend nie działają.

3. OPERATORY I SŁOWA ZAREZERWOWANE

Operatory matematyczne:

Operator	Funkcja	Priorytet	Przykład
+	dodawanie	6	3+2=5
-	odejmowanie	6	3-2=1
-	zmiana znaku	2	-(-2)=2
*	mnożenie	3	3*2=6
/	dzielenie	3	3/2=1.5
^	potęgowanie	1	3^2=9
DIV	dzielenie całkowite	4	3 DIV 2=1
MOD	reszta z dzielenia całk.	5	3 MOD 2=1

Znaki relacji:

=	- równy	<>	- różny
>	- większy	>=	- większy równy
<	- mniejszy	<=	- mniejszy równy

Stałe:

True=-1
False=0

Pi=3.1415926536

Typy zmiennych:

bez znaku - zmienna rzeczywista (dwanaście miejsc znaczących)
% - zmienna całkowita
$ - zmienna tekstowa
! - zmienna logiczna (przyjmuje tylko wartości 0 i -1)
E - oznacza postać wykładniczą (2E3=2000)

Znaki specjalne:

' - komentarz (od początku linii np: ' To jest komentarz)
! - komentarz (wewnątrz linii np:A=2 ! To jest komentarz)
@ - wywołanie procedury (@ Proc = Gosub Proc)
? - zastępuje Print
==) - znakiem tym opatrywane są teksty wczytane komendą Merge a nie będące programem
* - zapis X=*Y jest równoważny zapisowi X=Arrptr(Y)

Operatory logiczne:

Operator	Funkcja	X	Y	Wynik
NOT	negacja	1	-	0
		0	-	1
AND	iloczyn logiczny	1	1	1
		1	0	0
		0	1	0
		0	0	0
OR	suma logiczna	1	1	1
		1	0	1
		0	1	1
		0	0	0
XOR	wyłączne lub	1	1	0
		1	0	1
		0	1	1
		0	0	0
EQV	nie (wyłączne lub)	1	1	1
		1	0	0
		0	1	0
		0	0	1
IMP	implikacja	1	1	1
		1	0	0
		0	1	1
		0	0	1

Słowa zarezerwowane:

ADDRIN
ADDROUT
CONTRL
FALSE
GB
INTIN
INTOUT
GINTIN
GINTOUT
PI
PTSIN
PTSOUT
TRUE

oraz nazwy wszystkich komend i funkcji.

4. FUNKCJE I KOMENDY GFA-BASIC'a

W rozdziale tym omówiono w porządku alfabetycznym funkcje i komendy GFA-BASIC'a według następującego schematu:

```
  .....................................................Numer kolejny
  :
  :       .............................................Rodzaj instrukcji
  :       :
  :       :          ..................................Nazwa
  :       :          :
  :       :          :          .......................Pełna nazwa
  :       :          :          :
13____KOMENDA____Bload__(binary load)_____________________________B

Komenda ładuje (zapisany wcześniej komendą 'Bsave').....opis
obszar pamięci z dysku do RAM-u.                        instrukcji

* Bsave.....................................................instrukcje
                                                           związane

Składnia: Bload filename[,address].....................składnia

Skrót   : Bl filename[,address].........................skrót

filename: nazwa pliku - stała lub zmienna tekstowa.....opis
address: stała,zmienna lub wyrażenie numeryczne        parametrów
określające adres ładowania

 Box 0,0,639,199
 Bsave "Screen",Xbios(2),32000.........................przykład
 Cls
 Bload "Screen",Xbios(2)
 Pause 100
-------------------------------------------------------------------

Parametry lub fragmenty instrukcji ujęte w nawiasy kwadratowe [ ]
są opcjonalne, tzn. mogą być pominięte.
```

Alfabetyczny spis funkcji i komend:

1____FUNKCJA____Abs__(absolute)_____A

Funkcja oblicza wartość bezwzględną argumentu.

Składnia: Y=Abs(X)
X: stała, zmienna lub wyrażenie numeryczne dowolnego typu

```
X=-5
Print Abs(-5)'Abs(0)'Abs(5)'Abs(X*5)
Pause 100
```

2____KOMENDA____Add__(addition)_____A

Komenda dodaje do zmiennej 'var' wartość 'n'.
* Inc

Składnia: Add var,n
Skrót   : Ad var,n
var: zmienna numeryczna dowolnego typu
n : stała, zmienna lub wyrażenie numeryczne dowolnego typu

```
T=Timer
For I%=1 To 10000
 Add A%,5
Next I%
Print (Timer-T)/200
'
A%=0
T=Timer
For I%=1 To 10000
 A%=A%+5
Next I%
Print (Timer-T)/200
```

3____KOMENDA____Alert__(alert)_____A

Komenda wywołuje skrzynkę alarmową.

Składnia: Alert a,mstring,b,bstring,var
Skrót   : A a,mstring,b,bstring,var
a:stała, zmienna lub wyrażenie numeryczne określające rodzaj znaku informacyjnego:

0-bez znaku informacyjnego
1-wykrzyknik
2-znak zapytania
3-stop

mstring: zmienna tekstowa opisująca komunikat (max. 5 linii po 40 znaków)
b:stała, zmienna lub wyrażenie numeryczne o wartości z przedziału 0-3 określające, który z wariantów zostanie wybrany po wciśnięciu RETURN
bstring: zmienna tekstowa opisująca warianty
var: zmienna numeryczna dowolnego typu (przyjmuje wartość odpowiadającą numerowi wariantu)
Znak "!" (Chr$(124)) powoduje przeniesienie dalszej części tekstu do nowej linii.

```
M$="   CZY CHCESZ   !SKASOWAĆ TEN PLIK ?"
Alert 2,M$,1,"TAK!NIE",B
Print B
```

4____KOMENDA____Arrayfill__(array fill)_____A

Komenda wypełnia tablicę 'field()' wartością 'n'.
* Dim, Dim?, Erase

Składnia: Arrayfill field(),n
Skrót : Ar field(),n
field(): tablica numeryczna dowolnych rozmiarów
n: stała, zmienna lub wyrażenie numeryczne tego samego typu co tablica

```
Dim A%(4,5,6),B(100)
C=7
Arrayfill A%(),12
Arrayfill B(),C
Print A%(0,0,0),A%(3,4,5),A%(4,5,6),B(80)
```

5____FUNKCJA____Arrptr__(array pointer)_____A

Funkcja podaje adres tablicy opisującej zmienną tekstową.
* Varptr

Składnia: Y=Arrptr(A$)
A$: zmienna tekstowa

```
A$="BASIC"
Print Arrptr(A$)
Print Lpeek(Arrptr(A$)),Varptr(A$)
Print Dpeek(Arrptr(A$)+4),Len(a$)
```

6____FUNKCJA____Asc__(ascii)_____A

Funkcja podaje kod ascii pierwszego znaku argumentu.
* Chr$

Składnia: Y=Asc(A$)
A$: zmienna tekstowa

```
A$="GfA"
Print Asc(A$)
Print Asc("!")
Print Asc("ABC!")
```

7____FUNKCJA____Atn__(arcus tangent)_____A

Funkcja oblicza arcus tangens argumentu.
* Tan

Składnia: Y=Atn(X)
X: stała, zmienna lub wyrażenie numeryczne dowolnego typu
Y: wartość arctn(x) wyrażona w radianach

```
Print Pi
X=1
Rad=Atn(X)
Grad=Rad*180/Pi
Print Rad
Print Grad
```

8____FUNKCJA_____Basepage__(base page)_____B

Funkcja podaje wartość równą adresowi strony bazowej programu.

Składnia: Y=Basepage

Strona bazowa programu to obszar pamięci o długości 256 bajtów zawierający dane dotyczące wykonywanego programu.

Patrz: Exec

STRUKTURA STRONY BAZOWEJ

Adres	Długość	Zawartość
Basepage+&h00	4	Adres bazowy TPA
&h04	4	Najwyższy adres TPA+1
&h08	4	Adres bazowy tekstu programu
&h0C	4	Długość tekstu programu
&h10	4	Adres bazowy danych początkowych
&h14	4	Długość danych początkowych
&h18	4	Adres bazowy BSS
&h1C	4	Długość BSS
&h2C	4	Wskaźnik łańcucha opisującego środowisko programowe
&h80	&h80	Początek linii komend

9____KOMENDA____Bget#__(binary get number)_____B

Komenda pobiera ze stosu binarny blok danych i umieszcza go w zadanym miejscu pamięci.
* Bput

Składnia: Bget #n,A,L
Skrót : Bg #n,A,L
n,A,L : stałe, zmienne lub wyrażenia numeryczne określające odpowiednio: numer pliku, adres przeznaczenia i długośc bloku

```
Circle 300,100,100
Box 0,0,639,199
Open "R",#99,"TEST"
Bput #99,Xbios(2),32000
Close
Cls
Open "R",#99,"TEST"
Bget #99,Xbios(2),32000
Close
```

10___FUNKCJA____Bin$__(binary dollar)_____________________________B

Funkcja tworzy łańcuch będący zapisem argumentu w układzie binarnym.
* Hex$, Oct$

Składnia: X$=Bin$(X)
X: stała, zmienna lub wyrażenie numeryczne dowolnego typu

```
A=-1
B=&022
Print Bin$(A)
Print Bin$(234)
Print Bin$(B)
```

11___FUNKCJA____Bios__(basic input / output system)_____________B

Funkcja wywołuje zadaną procedurę WE/WY z ROM-u.

Składnia :Y=BIOS(opc,list of parameters)
opc: stała, zmienna lub wyrażenie numeryczne określające numer funkcji
list of parameters: lista parametrów

WYKAZ WAZNIEJSZYCH FUNKCJI BIOS-u:

Opc(hex) Nazwa	Opis funkcji
01 bcinstat(a:I;)	a=0/1/2/3/4/5:PRT/AUX/CON/MIDI/KBD, sprawdza stan wybranego urządzenia wejściowego; urządzenie gotowe:-1, przeciwnie:0
02 bconin(a:I)	a:j/w, funkcja przyjmuje bajt z wybranego urządzenia
03 bconout (a:I;b:C)	a:j/w,funkcja wysyła bajt do wybranego urządzenia
06 tickcal()	funkcja podaje stan zegara systemowego
08 bcostat(a:I;)	jak dla opc=1,ale dla urządzenia wyjściowego
09 mediach(a:I;)	a:numer napędu, funkcja sprawdza stan nośnika, nośnik bez zmian:0, może być zmieniony:1 definitywnie zmieniony:2
0A drvmap()	funkcja podaje bitową mapę dostępnych napędów (bit 0-napęd A, bit 15-napęd P)

```
Print "Wcisnij dowolny klawisz"
Do
 Print Bios(2,2)
Loop
```

12____KOMENDA____Bmove__(binary move)_____________________________B

Komenda przesuwa zadany obszar pamięci w określone miejsce.

Składnia: Bmove S,D,L
S,D,L: stałe, zmienne lub wyrażenia numeryczne określające odpowiednio: adres startu, adres przeznaczenia, długość bloku

```
A$="test Bmove"
B$="          "
Adr=Varptr(A$)
Bmove Adr,Varptr(B$),Len(B$)
Print A$'B$
```

13____KOMENDA____Bload__(binary load)_____________________________B

Komenda ładuje (zapisany wcześniej komendą 'Bsave') obszar pamięci z dysku do RAM-u.
* Bsave

Składnia: Bload filename[,address]
Skrót   : Bl filename[,address]

filename: nazwa pliku - stała lub zmienna tekstowa
address: stała,zmienna lub wyrażenie numeryczne określające adres ładowania (W przypadku nie podania adresu przyjmuje on automatycznie wartość taką jak podczas zapisu pliku o tej samej nazwie komendą Bsave).

```
Box 0,0,639,199
Bsave "Screen",Xbios(2),32000
Cls
Bload "Screen",Xbios(2)
Pause 100
```

14____KOMENDA____Box__(box)_____________________________________B

Komenda rysuje prostokąt o wierzchołkach w punktach X0,Y0,X1,Y1.
* Pbox, Prbox

Składnia: Box X0,Y0,X1,Y1
Skrót : B X0,Y0,X1,Y1
X0,Y0,X1,Y1: stałe, zmienne lub wyrażenia numeryczne

Za punkt o współrzędnych 0,0 przyjęto we wszystkich komendach graficznych lewy górny punkt ekranu.

```
Box 10,10,100,10
Box 0,0,300,100
Box 10,10,305,105
Box 400,200,133,90
```

15____KOMENDA____Bput#__(binary put number)_____________________B

Komenda odkłada na stos blok pamięci o określonej długości począwszy od zadanego adresu.
* Bget

Składnia: Bput #n,A,L
Skrót : Bp #n,A,L
n,A,L: stałe, zmienne lub wyrażenia numeryczne określające odpowiednio: numer pliku, adres i długość bloku

Patrz: Bget

16____KOMENDA____Bsave__(binary save)___________________________B

Komenda zapisuje na dysku zadany obszar pamięci.
* Bload

Składnia: Bsave filename,address,len
Skrót : Bs filename,address,len
filename: nazwa pliku - stała lub zmienna tekstowa
address,len: stałe, zmienne lub wyrażenia numeryczne określające odpowiednio: adres początkowy obszaru pamięci, jego długość

Patrz: Bload

17____KOMENDA____C__(c)_____C

Komenda wywołuje skompilowany program w jezyku C.

Składnia: C:adr(list of parameters)
adr: stała, zmienna lub wyrażenie numeryczne określajace adres podprogramu maszynowego

```
Q%=Varptr(Q$)
A=C:Q%(17,L:0,W:-1)
```

18____KOMENDA____Call__(call)_____C

Komenda wywołuje podprogram maszynowy.

Składnia: Call adr [(list of parameters)]
Skrót : Ca adr [(list of parameters)]
adr: stała, zmienna lub wyrażenie numeryczne określajace adres podprogramu maszynowego

19____KOMENDA____Chain__(chain)_____C

Komenda wczytuje program o zadanej nazwie oraz uruchamia go automatycznie.
* Load

Składnia: Chain filespec
Skrót : Ch filespec
filespec: stała lub zmienna tekstowa

20____KOMENDA____Chdir__(change directory)_____C

Komenda ustala nazwę katalogu, który będzie aktualnie dostępny.
* Chdrive, Dir, Mkdir, Files

Składnia: Chdir folder name
Skrót : Chd folder name
folder name: stała lub zmienna tekstowa (nazwa katalogu istniejacego na dysku)

```
Mkdir "\Dir_1"
Mkdir "\Dir_1\Dir_2"
Chdir "\"
Files
Chdir "\Dir_1"
Files
```

21____KOMENDA____Chdrive__(change drive)_____C

Komenda powoduje przełączenie napędów.
* Chdir, Dir, Files, Load, Save

Składnia: Chdrive n
Skrót   : Chdr n
n: stała, zmienna lub wyrażenie numeryczne określające numer napędu: (1-15) 1-Floppy 1, 2-Floppy 2

```
Chdrive 1
Files
Chdrive 2
Files
```

22____FUNKCJA____Chr$__(character dollar)_____C

Funkcja wywołuje znak o kodzie X.
* Asc

Składnia: Y$=Chr$(X)
X: stała, zmienna lub wyrażenie numeryczne dowolnego typu

```
For I=32 To 255
 Print Chr$(I);" ";
Next I
Pause 100
Cls
'
For I=32 To 255
 Print Chr$((I+2147483648) Mod 256);" ";
Next I
End
```

23____KOMENDA____Circle__(circle)_____C

Komenda rysuje łuk o współrzędnych środka X,Y promieniu R, kącie początkowym Fi1 oraz kącie końcowym Fi2.
* Pcircle, Ellipse, Pellipse

Składnia: Circle X,Y,R(,Fi1,Fi2)
Skrót : C X,Y,R(,Fi1,Fi2)
X,Y,R,Fi1,Fi2: stałe, zmienne lub wyrażenia numeryczne dowolnego typu. W przypadku niepodania wartości Fi1 i Fi2 są one automatycznie przyjmowane jako 0 i 3600.

```
Circle 160,100,100,450,2700
Circle 100,50,45
```

24____KOMENDA____Clear__(clear)_____C

Komenda powoduje nadanie wszystkim zmiennym numerycznym wartości 0, a zmiennym tekstowym wartości "".
* Clr

Składnia: Clear
Skrót : Cle

```
A=17
B$="BASIC"
Clear
Print A,B$+"**"
```

25____KOMENDA____Clearw__(clear window)_____C

Komenda czyści okno o numerze n.
* Cls

Składnia: Clearw n
Skrót : Cle w n
n: stała, zmienna lub wyrażenie numeryczne określające numer okna

```
Openw 2
Circle 160,100,100,450,2700
Pause 100
Clearw 2
```

26____KOMENDA____Close__(close)

Komenda powoduje zamknięcie pliku o numerze n.

Składnia: Close [[#]n]
Skrót : Cl [[#]n]
n: stała,zmienna lub wyrażenie numeryczne określające numer pliku
Opuszczenie parametru n powoduje zamknięcie wszystkich otwartych na dysku plików.

```
Open "O",#1,"PRO"
Print #1,"GfA-BASIC"
Close #1
Open "I",#1,"PRO"
Input #1,A$
Close
Print A$
```

27____KOMENDA____Closew__(close window)

Komenda zamyka okno o podanym numerze.
* Openw, Clearw

Składnia: Closew n
n: stała, zmienna lub wyrażenie numeryczne określające numer okna

Patrz: Openw

28____KOMENDA____Clr__(clear)

Komenda nadaje wyszczególnionym zmiennym wartość zero.
* Clear

Składnia: Clr X,[Y,...]
X..Y: zmienne dowolnego typu za wyjątkiem zmiennych tablicowych

```
A=5
B=10
Print A*B,A+B
Clr A
Print A*B,A+B
```

29____KOMENDA____Cls__(clear screen)_____C

Komenda powoduje wyczyszczenie ekranu.
* Clearw

Składnia: Cls

```
For I=0 To 1999
 Print "A";
Next I
Print At(32,13);" Press a key ! "
A=Inp(2)
Cls
Pause 100
```

30____KOMENDA____Cls #__(clear screen number)_____C

Komenda powoduje zapisanie w zadanym pliku informacji powodujacej czyszczenie ekranu, każdorazowo po wyprowadzeniu na ekran danej zapisanej w pliku bezpośrednio po komendzie Cls #.

Składnia: Cls #n
n: stała,zmienna lub wyrażenie numeryczne określające numer pliku

```
A$="abcdefg"
Open "O",#1,"test1"
Open "O",#2,"test2"
Cls #1
Print #1,A$
Print #1,"123456"
Print #2,A$
Print #1,"123456"
Close
Open "I",#1,"test1"
Input #1,B$,C$
Close
Print At(35,10);"PAUSE 100"
Pause 100
Print B$
Pause 100
Print C$
Pause 100
Cls
```

```
Open "I",#2,"test2"
Input #2,B$,C$
Close
Print At(35,10);"PAUSE 100"
Pause 100
Print B$
Pause 100
Print C$
Pause 100
```

31____KOMENDA____Color__(color)_____C

Komenda ustala kolor, który będzie użyty w następnej instrukcji graficznej.
* Setcolor

Składnia: Color C
Skrót : Co C
C:stała,zmienna lub wyrażenie numeryczne określające numer koloru

```
Tryb LOW   :C=0-15
Tryb MEDIUM:C=0-3
Tryb HIGH  :C=0-1

Pcircle 100,100,100
Color 0
Line 0,0,200,200
Pcircle 300,100,100
Color 2
Line 200,0,400,200
Pcircle 500,100,100
Color 3
Line 400,0,600,200
Print At(35,16);"Press a key"
A=Inp(2)
```

32____KOMENDA____Cont__(continue)_____C

Komenda powoduje wznowienie pracy programu po komendzie Stop.
* Stop

Składnia: Cont
Skrót   : Con

33___FUNKCJA____Cos__(cosine)_____________________________C

Funkcja oblicza wartość cosinus(x).

Składnia: Y=Cos(X)
X: stała, zmienna lub wyrażenie numeryczne dowolnego typu

```
 Plot 320,100
 For I=1 To 5400
  X=I/24*Cos(I*Pi/180)
  Y=I/48*Sin(I*Pi/180)
  Draw  To 320+X,100+Y
 Next I
```

34____FUNKCJA____Crscol__(cursor columne)_____________________C

Funkcja podaje numer kolumny, w której znajduje się kursor.
* Crslin, Pos

Składnia: Y=Crscol
Kolumny liczone są od lewej strony od 1 do 40 w trybie LOW oraz od 1 do 80 w trybach: MEDIUM i HIGH.

```
 Print At(10,15);
 X=Crscol
 Y=Crslin
 Print X,Y
```

35____FUNKCJA____Crslin__(cursor line)_____________________C

Funkcja podaje numer wiersza w którym znajduje się kursor.
* Crscol

Składnia: Y=Crslin
Wiersze liczone są od góry od 1 do 25 w każdym z trybów graficznych.

Patrz: Crscol

36____FUNKCJA____Cv*__(convert)_____C

Funkcja zamienia zmienną tekstową na numeryczną.
* Mk*$

	INTEGER		REAL		
Składnia:	X=Cvi(X$)	Cvl(X$)	Cvs(X$)	Cvf(X$)	Cvd(X$)
Dług.łańcucha:	2 bajty	4 bajty	4 bajty	6 bajtów	8 bajtów

X$: stała lub zmienna tekstowa

```
A=65
A$=Mki$(65)
Print A$
Print Cvi(A$)
```

37____KOMENDA____Data__(data)_____D

Instrukcja Data umożliwia umieszczenie zbioru danych w programie.
* Read

Składnia: Data [const[,const]...]
Skrót : D [const[,const]...]
const: stała dowolnego typu

```
Read A,B$,C$,D,E!
Print A;B$;C$;D;E!
'
Data 234,"G.f.A",BASIC,&HFF,56
```

38____KOMENDA____Date$__(date)_____D

Komenda powoduje wywołanie daty w postaci miesiąc / dzień / rok.
* Settime

Składnia: X$=Date$

```
X$=Date$
Print X$
Print Date$
Deftext 1,4,0,12
Text 240,180,X$
```

39____KOMENDA____Dec__(decrease)_____D

Komenda powoduje odjęcie od zmiennej 'var' liczby 1.
* Sub

Składnia: Dec var
var: zmienna numeryczna dowolnego typu

```
T=Timer
For I%=1 To 10000
 Dec A%
Next I%
Print (Timer-T)/200
'
A%=0
T=Timer
For I%=1 To 10000
 A%=A%-1
Next I%
Print (Timer-T)/200
```

40____KOMENDA____Deffill__(define fill)_____D

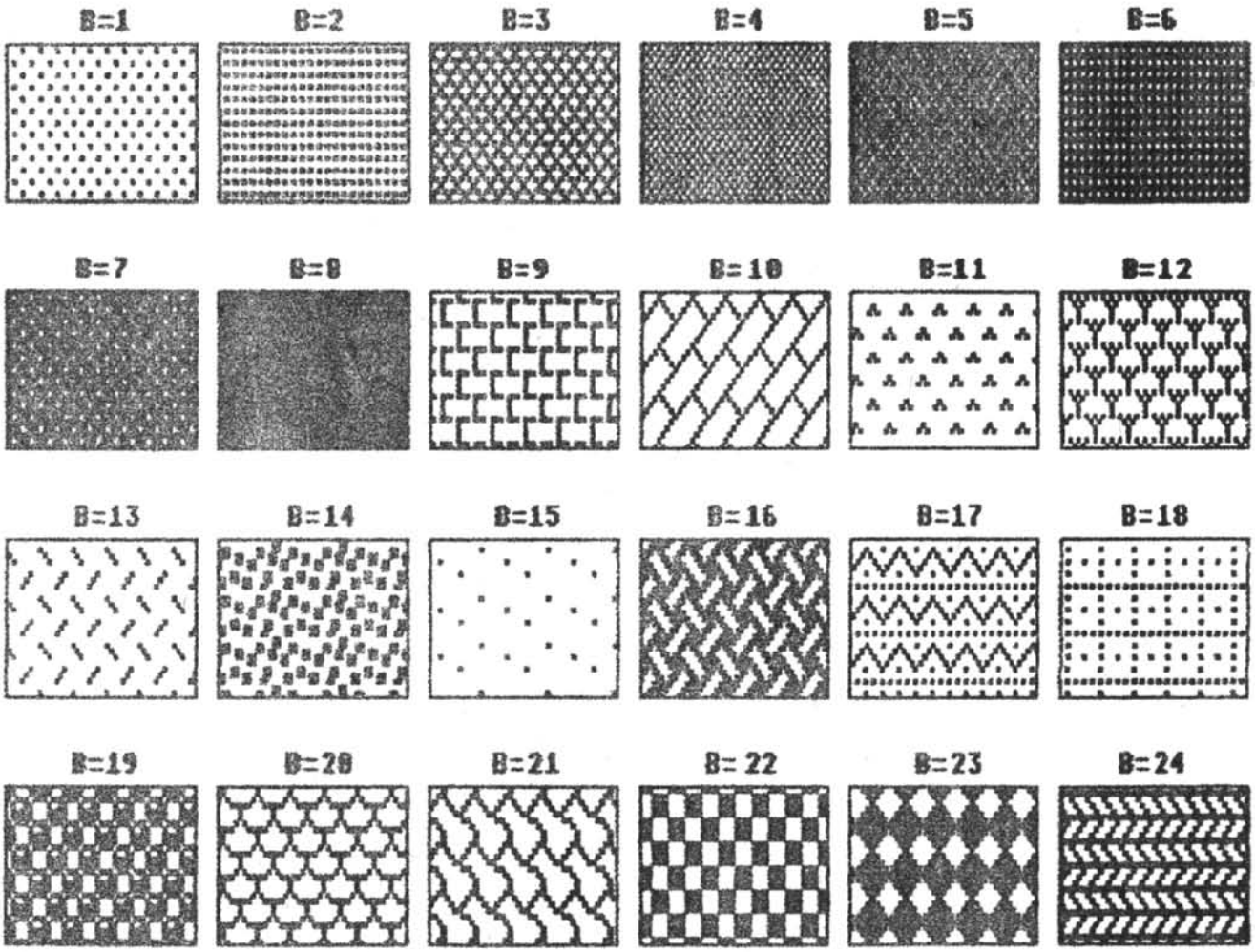
Komenda ustala rodzaj deseniu, którym będzie wypełniony obszar w następnych instrukcjach graficznych oraz pozwala zdefiniować własny deseń.
* Fill, Pcircle, Pellipse, Pbox, Prbox

Składnia: Deffill [C],[A],[B]
Skrót : Deff [C],[A],[B]
C: stała, zmienna lub wyrażenie numeryczne określające kolor deseniu
A,B: stałe, zmienne lub wyrażenia numeryczne określające rodzaj deseniu: dla A=2 B=1-24, dla A=3 B=1-12

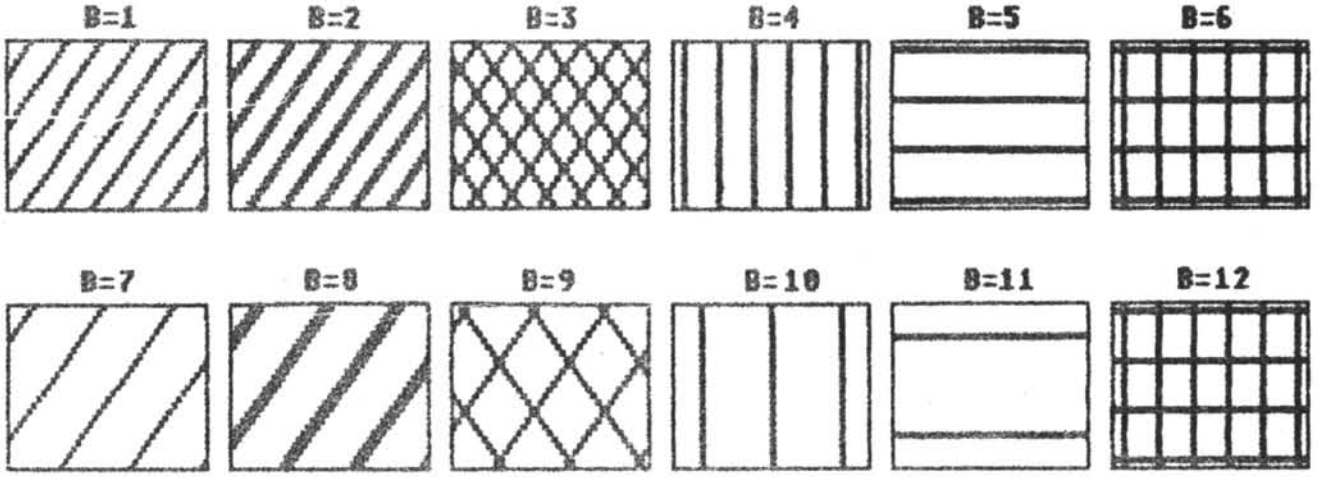
```
Box 50,50,200,150
Deffill 1,2,9
Fill 25,100
Deffill ,,10
Fill 160,100
```

Desenie zaprogramowane w ROM-ie. (Komenda Deffill)

A=2



A=3



Definiowanie własnego deseniu.

Składnia: Deffill [C],A$
Skrót   : Deff [C],A$

```
 Data &x0000000000010000
 Data &x0000000001100000
 Data &x0000000011000000
 Data &x0000000011000000
 Data &x0000000010000000
 Data &x0000000000000000
 Data &x0000111001110000
 Data &x0011111111111000
 Data &x0011111111111000
 Data &x0011111111100000
 Data &x0011111111000000
 Data &x0011111111100000
 Data &x0011111111111000
 Data &x0001111111110000
 Data &x0000111011100000
 Data &x0000000000000000
 For I=1 To 16
  Read Desen
  Desen$=Desen$+Mki$(Desen)
 Next I
 Deffill 1,Desen$
 Pbox 100,0,300,100
```

41____KOMENDA____Deffn__(define function)_____________________D

Komenda definiuje funkcje użytkownika.

Składnia: Deffn name[(varliste)]=expression
name: nazwa funkcji
varliste: lista parametrów
expression: dowolne wyrażenie numeryczne

```
 Do
  Input X
  Print X,Fn Three,Fn Mult(10)
 Loop
 Deffn Three=3*X
 Deffn Mult(Y)=Y*Fn Three
```

42____KOMENDA____Defline__(define line)_____D

Komenda definiuje rodzaj linii, który będzie użyty w następnych instrukcjach graficznych.
* Box, Circle, Ellipse, Line, Draw, Plot

Składnia: Defline [S],[W],[E1],[E2]
Skrót : De [C],[W],[E1],[E2]
S: stała, zmienna lub wyrażenie numeryczne określające rodzaj linii (1-6); dla linii definiowanej przez użytkownika - S=7
W: stała,zmienna lub wyrażenie numeryczne określające grubość linii (0-40)
E1,E2: stałe, zmienne lub wyrażenia numeryczne określające sposób zakończenia linii (0-pod kątem prostym, 1-strzałka, 2-półkole)

```
Defline 1,10,2,1
Line 140,170,350,170
'
For S=1 To 6
 Print "S=";S;":"
Defline S,1,0,0
Line 50,S*8-5,350,S*8-5
Next S
```

43____KOMENDA____Deflist__(define list)_____D

Komenda ustala sposób edycji programu.

Składnia: Deflist X
Skrót : Deflis X

X=0 - EDYCJA DUŻYMI LITERAMI
X=1 - Edycja małymi literami

```
Deflist 0
List
Pause 100
Deflist 1
List
```

44____KOMENDA____Defmark__(define marker)_____D

Komenda definiuje kolor, rodzaj oraz wielkość znacznika.
* Polymark

Składnia: Defmark [C],[X],[D]
Skrót   : Defm [C],[X],[D]

C: stała, zmienna lub wyrażenie numeryczne określające kolor
X: stała,zmienna lub wyrażenie num. określające kształt znacznika
D: stała, zmienna lub wyrażenie numeryczne określające wielkość znacznika

```
Dim X(5),Y(5)
Do
 For I=0 To 5
  X(I)=Random(640)
  Y(I)=Random(200)
 Next I
 Defmark 1,Random(7),0
 Polymark 6,X(),Y()
Loop
```

45____KOMENDA____Defmouse__(define mouse)_____D

Komenda ustala rodzaj wskaźnika myszy oraz pozwala zdefiniować jego własny kształt.

Składnia: Defmouse N
Skrót   : Defmo N

N: stała, zmienna lub wyrażenie numeryczne określające rodzaj wskaźnika

N=0 - strzałka
  1 - kursor
  2 - trzmiel
  3 - palec wskazujący
  4 - otwarta dłoń
  5 - krzyż
  6 - krzyż wytłuszczony
  7 - krzyż wytłuszczony ze szczeliną

```
For I=0 To 7
 Defmouse I
 Pause 100
Next I
```

Definiowanie własnego kształtu wskaźnika myszy.

Składnia: Defmouse A$
Skrót   : Defmo A$

A$=Mki$(Dx) + Mki$(Dy) + Mki$(0) + Mki$(Color 1) + Mki$(Color 2)+
16*Mki$(postać 1) + 16*Mki$(postać 2)
Dx,Dy: przesunięcia wskaźnika względem punktu o współrzędnych:
Mousex,Mousey

```
 Data &x0000001110000000
 Data &x0000001010000000
 Data &x0000001010000000
 Data &x0000001010000000
 Data &x0000001010000000
 Data &x0000001010000000
 Data &x1111110001111110
 Data &x1000000000000010
 Data &x1111110001111110
 Data &x0000001010000000
 Data &x0000001010000000
 Data &x0000001010000000
 Data &x0000001010000000
 Data &x0000001010000000
 Data &x0000001110000000
 Data &x0000000000000000
 ,
 Data &x0000000000000000
 Data &x0000000100000000
 Data &x0000000100000000
 Data &x0000000100000000
 Data &x0000000100000000
 Data &x0000000100000000
 Data &x0000000000000000
 Data &x0111110001111100
 Data &x0000000000000000
 Data &x0000000100000000
 Data &x0000000100000000
 Data &x0000000100000000
```

Tekst. (Komenda Deftext)

X=0 TEXT

X=1 TEXT

X=2 TEXT

X=4 TEXT

X=8 TEXT

X=16 TEXT

D=0 TEXT

D=450 TEXT

D=1350 TEXT

D=2250 TEXT

N=3 TEXT

N=4 TEXT

N=6 TEXT

N=7 TEXT

N=9 TEXT

N=12 TEXT

Znaczniki. (Komenda Defmark)

X=0 X=1 X=2 X=3 X=4 X=5 X=6

Linie zaprogramowane w ROM-ie. (Komenda Defline)

S=1

S=2

S=3

S=4

S=5

S=6

```
Data &x0000000100000000
Data &x0000000100000000
Data &x0000000000000000
Data &x0000000000000000
'
Mf$=Mki$(7)+Mki$(7)+Mki$(0)+Mki$(0)+Mki$(1)
'
For I%=1 To 32
 Read A
 Mf$=Mf$+Mki$(A)
Next I%
Defmouse Mf$
'
Do
 If Mousek
 Plot Mousex,Mousey
Endif
Loop
```

46____KOMENDA____Deftext__(define text)_____D

Komenda definiuje kolor,czcionkę, kierunek oraz wielkość napisów.
* Text

Składnia: Deftext [C],[X],[D],[H]
Skrót : Deft [C],[X],[D],[H]
C: stała, zmienna lub wyrażenie numeryczne określające kolor
X: stała, zmienna lub wyrażenie numeryczne określające czcionkę:

Wartość bitu	Waga bitu					
	32	16	8	4	2	1
0	normal	normal	normal	normal	normal	normal
1	shadow	outline	under	italic	light	bold

D: stała, zmienna lub wyrażenie numeryczne określające kierunek pisma:

0 -poziomo w prawo
450 -pionowo w górę
1350-poziomo w lewo
2250-pionowo w dół

H: stała, zmienna lub wyrażenie numeryczne określające rozmiar czcionki (pismo normalne-H=6)
Deftext wpływa zawsze na tekst wyprowadzany komendą Text. W przypadku otwartego okna (np.:Openw 0),wpływa również na tekst wyprowadzany komendą Print.

```
Deftext 1,4,450,9
Text 100,180,"pionowa kursywa"
Deftext 1,1,0,7
Text 200,20,"WYTŁUSZCZENIE"
```

47____FUNKCJA____Dfree__(disk free)_____D

Funkcja podaje ilość dostępnej pamięci na dysku (w bajtach).
* Fre

Składnia: Y=Dfree(X)
X: stała, zmienna lub wyrażenie numeryczne określające numer napędu

```
Print Dfree(1)
Y=Dfree(2)
Print Y
```

48____KOMENDA____Dim__(dimension)_____D

Komenda deklaruje tablicę o zadanej nazwie i zadanym rozmiarze.
* Arrayfill, Dim?, Erase

Składnia: Dim var(size)[,var(size),...]
Skrót : Di var(size)[,var(size),...]
var: zmienna dowolnego typu (możliwe jest istnienie zmiennej prostej i tablicy o tej samej nazwie)
size: stała, zmienna lub wyrażenie numeryczne określające rozmiar tablicy

```
Dim A(1000),B(4,5,3)
Dim N$(20,5)
Print A(10)
Arrayfill A(),3
Print A(10)
```

49____FUNKCJA____Dim?__(dimension)_____D

Funkcja podaje rozmiar tablicy o zadanej nazwie.
* Dim, Erase

Składnia: Y=Dim?(var())
var: zmienna określająca tablicę, której rozmiar chcemy otrzymać

```
Dim A$(3,4,5)
Dim N%(12,12)
Print Dim?(A$())
Print Dim?(N%())
```

50____KOMENDA____Dir__(directory)_____D

Komenda wywołuje katalog dysku oraz umożliwia zapisanie go w pliku o zadanej nazwie (katalog nie uwzględnia folderów).
* Chdir, Chdrive, Dir$, Files, Fileselect

Składnia: Dir[filespec [To file]]
filespec: stała lub zmienna tekstowa określająca napęd oraz typ plików, które mają być wyświetlone (* - oznacza dowolny)
file: nazwa pliku, w którym zostanie zapisany wywołany katalog

```
Dir "A:*.*" To "LST"
Dir "A:*.BAS"
Dir
```

51____FUNKCJA____Dir$__(directory dollar)_____D

Funkcja podaje nazwę katalogu, która została ustalona komendą Chdir.
* Mkdir

Składnia: Y$=Dir$(N)
N: stała, zmienna lub wyrażenie numeryczne określające numer napędu

```
Mkdir "\TEST"
Chdir "\TEST"
Print Dir$(1)
```

52____KOMENDA____Div__(division)_____D

Komenda powoduje podzielenie zmiennej 'var' przez wartość n.

Składnia: Div var,n
var: zmienna numeryczna
n: stała, zmienna lub wyrażenie numeryczne dowolnego typu

```
Dim AX(10000)
For I=1 To 10000
 AX(I)=5
Next I
T=Timer
For I=1 To 10000
 Div AX(I),5
Next I
Print (Timer-T)/200
T=Timer
For I=1 To 10000
 AX(I)=AX(I)/5
Next I
Print (Timer-T)/200
```

53____KOMENDA____Do__(do)_____D

Komenda powoduje otwarcie pętli Do-Loop.
* Loop

Składnia: Do

```
Do
 Inc I
 Print I
Loop
```

54____FUNKCJA____Dpeek__(double peek)_____D

Funkcja podaje zawartość dwóch kolejnych komórek pamięci począwszy od adresu równego argumentowi jako zmienną 16 - bitową.
* Lpeek, Peek

Składnia: Y=Dpeek(X)
X: stała, zmienna lub dowolne wyrażenie numeryczne określające adres komórki

Patrz: Peek

55____KOMENDA____Dpoke__(double poke)_____________________________D

Komenda wpisuje do dwóch komórek począwszy od wskazanej, zadaną wartość 16-bitową.
* Lpoke, Poke

Składnia: Dpoke X,Y
X,Y: stałe, zmienne lub dowolne wyrażenia numeryczne określające odpowiednio: adres komórki, zadaną wartość (Y<65535)

Patrz: Poke

56____KOMENDA____Draw__(draw)_____________________________________D

Komenda rysuje linię między punktami o współrzędnych X0,Y0 i X1,Y1.
* Defline

Składnia: Draw X0,Y0 To X1,Y1 [To X2,Y2...]
Skrót : Dr X0,Y0 To X1,Y1 [To X2,Y2...]

```
Draw 10,10
Draw  To 100,10
Draw 100,10 To 100,200
Draw 100,200 To 10,150 To 10,10
Draw 40,40 To 50,50
```

57____KOMENDA____Edit__(edit)_____________________________________E

Komenda powoduje przejście do trybu edycji.

Składnia: Edit
Skrót : Ed

```
Print "Test"
Pause 100
Edit
```

58____KOMENDA____Ellipse__(ellipse)_____E

Komenda rysuje wycinek elipsy o współrzędnych środka X,Y, promieniach RX,RY, kącie początkowym Fi1 oraz kącie końcowym Fi2.
* Pellipse

Składnia: Ellipse X,Y,RX,RY(,Fi1,Fi2)
Skrót : Ell X,Y,RX,RY(,Fi1,Fi2)
X,Y,RX,RY,Fi1,Fi2: stałe, zmienne lub wyrażenia numeryczne dowolnego typu
W przypadku niepodania wartości Fi1 i Fi2 są one automatycznie przyjmowane jako 0 i 3600.

```
Ellipse 320,100,300,75
Ellipse 320,0,320,150,1800,3600
```

59____KOMENDA____Else__(else)_____E

Komenda powoduje wykonanie następującego po niej bloku programowego w przypadku nie spełnienia warunku występującego po instrukcji If.
* If, Endif

Patrz: If

60____KOMENDA____End__(end)_____E

Komenda wstrzymuje wykonywanie programu.

Składnia: End

```
For I=1 To 10000
 If I=1000
  Print I
  End
 Endif
Next I
```

61____KOMENDA____Endif__(end if)_____E

Komenda kończy strukturę If...Else...Endif.
* If, Else

Składnia: Endif
Skrót   : En

Patrz: If

62____FUNKCJA____Eof__(end of file)_____E

Funkcja przyjmuje wartość -1 po odczytaniu ostatniej danej z pliku.

Składnia: X=Eof([#]n)
n: stała,zmienna lub wyrażenie numeryczne określające numer pliku

```
Open "O",#1,"DAT"
Print #1,"123456"
Close #1
Open "I",#1,"DAT"
Do
 Print Inp(#1),Eof(#1)
 Exit If Eof(#1)
Loop
```

63____KOMENDA____Erase__(erase)_____E

Komenda usuwa tablicę o zadanej nazwie.
* Dim, Dim?

Składnia: Erase field()
Skrót   : Er field()
field: nazwa tablicy, która chcemy usunąć

```
Print Fre(0)
Dim A(100,50,5)
Print Fre(0)
Erase A()
Print Fre(0)
```

64____FUNKCJA____Err__(error)_____E

Funkcja podaje numer błędu w przypadku, gdy taki wystąpi.
* Error

Składnia: Y=Err
Err: numer błędu (patrz: Wykaz Błędów)

```
On Error Gosub errorroutine
Print Sqr(-1)
'
Procedure errorroutine
 Print " error nr.:";Err
Return
```

65____KOMENDA____Error__(error)_____E

Komenda wywołuje błąd o zadanym numerze.
* On error

Składnia: Error n
Skrót   : Err n
n: numer błędu (patrz: Wykaz Błędów)

```
Input "Podaj numer bledu ";F
Error F
```

66____FUNKCJA____Even__(even)_____E

Funkcja przyjmuje wartość -1 dla argumentów będących liczbami parzystymi.
* Odd

Składnia: Y=Even(X)
X: stała, zmienna lub dowolne wyrażenie numeryczne

```
For I=0 To 10
 Print I,Even(I)
Next I
```

67____KOMENDA____Exec__(execute)_____E

Komenda powoduje wczytanie z dysku skompilowanego programu oraz umożliwia jego uruchomienie.

Składnia: Exec mode,filename,cmd,env
mode: 0 - powoduje wczytanie programu oraz jego natychmiastowe uruchomienie
3 - powoduje wczytanie programu bez jego uruchomienia
4 - uruchamia program wczytany za pomocą mode 3
filename, cmd, env: stałe lub zmienne tekstowe określające odpowiednio: nazwę programu, linię komend, łańcuch opisujący konfigurację środowiska

Skompiluj poniższy program i nadaj mu nazwę "TESTEXEC.PRG"

```
For I=0 To 30
 Print Chr$(Peek(Basepage+&H80+I));
Next I
Print At(1,10);"Wcisnij dowolny klawisz"
While Inkey$=""
Wend
Quit
End
```

po czym wykonaj nastepujący program pamiętając by dysk z programem "TESTEXEC.PRG" był w aktywnym napedzie

```
If Exist("TESTEXEC.PRG")
 Reserve Fre(0)-50000
 Exec 0,"TESTEXEC.PRG","TU MOZESZ UMIESCIC PARAMETRY",""
 Reserve Fre(0)+50000
Endif
```

Zwróć uwagę na sposób przekazania parametrów poprzez Basepage oraz cmd.

68____FUNKCJA____Exist__(existence)_____E

Funkcja przyjmuje wartość -1, jeżeli dany plik istnieje.

Składnia: X=Exist(filespec)
filespec: nazwa pliku (stała lub zmienna tekstowa)

```
Open "O",#1,"DATE.DOC"
Close #1
Print Exist("DATE")
Print Exist("DAT*.DOC")
Print Exist("*.*")
```

69____KOMENDA____Exit__(exit)_____E

Komenda powoduje opuszczenie pętli Do-Loop w przypadku spełnienia zadanego warunku. Może być również stosowana w pętli For-Next.
* Do, Loop

Składnia: Exit If condition
Skrót   : E if condition

```
Do
 A=A+1
 Print A
 Exit If A=20
Loop
```

70____FUNKCJA____Exp__(exponential)_____E

Funkcja oblicza exponent argumentu.

Składnia: Y=Exp(X)

```
Print Exp(1)
Print Exp(0.5),Exp(-2)
```

71____KOMENDA____Field__(field)_____F

Komenda otwiera stos,na który możemy odkładać dane tekstowe o zadeklarowanej długości.
* Get #n, Put #n

Składnia: Field [#]n,len As var1$[,len As var2$,...]
Skrót   : Fie [#]n,len As var1$...
n: stała,zmienna lub wyrażenie numeryczne określające numer pliku
len: długość zmiennej odkładanej na stos
var$: zmienne tekstowe

```
Open "R",#1,"TEST",50
Field #1,20 As A$,30 As B$
For I=1 To 4
 Input "IMIE,NAZWISKO: ",V$,N$
 Lset A$=V$
 Lset B$=N$
 Put #1,I
Next I
For I=4 Downto 1
 Get #1,I
 Print A$'B$
Next I
Close #1
```

72____KOMENDA____Files__(files)_____F

Komenda powoduje wyświetlenie nazw plików znajdujących się na dysku oraz umożliwia zapisanie ich w nowym pliku.
* Chdir, Chdrive, Dir, Dir$, Fileselect

Składnia: Files [filespec [To file]]
Skrót : File [filespec [To file]]
filespec: określenie typu plików
file: nazwa pliku w którym zostanie zapisany wywołany zbiór plików

```
Files "A:*.*"
Files "A:*.*" To "Spis"
Files "A:*.bas" To "A:BASIC"
```

73____KOMENDA____Fileselect__(file selection)_____F

Komenda powoduje wywołanie skrzynki dialogowej zawierającej katalog dysku.
* Dir, Files

Składnia: Fileselect filespec,filename,X$
Skrót : Filese filespec,filename,X$
filespec: określenie typu plików
filename: proponowana nazwa pliku
X$: zmienna X$ przyjmie nazwę wybranego pliku

```
Do
 Fileselect "\*.*",B$,A$
 Exit If A$=""
 Print A$
 B$=Right$(A$,Len(A$)-1)
Loop
```

74____KOMENDA____Fill__(fill)_____F

Komenda wypełnia zamknięty kontur zadanym deseniem.
* Deffill

Składnia: Fill X,Y
Skrót   : Fi X,Y
X,Y: stałe, zmienne lub wyrażenia numeryczne określające współrzędne X, Y punktu, od którego rozpocznie się wypełnianie.

```
 Box 10,10,100,100
 Draw 400,10 To 500,10 To 500,150 To 400,150 To 400,11
 Deffill 1,2,9
 Fill 50,25
 Deffill 1,2,8
 Fill 450,140
```

75____FUNKCJA____Fix__(fix)_____F

Wynikiem działania funkcji Fix jest część całkowita argumentu.
* Int

Składnia: Y=Fix(X)
X: stała, zmienna lub dowolne wyrażenie numeryczne

```
 A=3.1415
 Print Fix(A),Fix(3.99),Fix(-1.99)
```

76____FUNKCJA____Fn__(function)_____F

Fn oblicza wartość funkcji definiowanej przez użytkownika.
* Deffn

```
Składnia: Y=Fn name(list of parameters)
Skrót   : Y=@ name(list of parameters)
name: nazwa funkcji

 Do
  Input X
  Print X,Fn Three,Fn Mult(10)
 Loop
 Deffn Three=3*X
 Deffn Mult(Y)=Y*Fn Three
```

77____KOMENDA____Form Input__(form input)_____F

Instrukcja wstrzymuje działanie programu i oczekuje na wprowadzenie z klawiatury zadanej liczby znaków.
* Input, Input$

```
Składnia: Form Input n,var$
Skrót   : F Input n,var$
```
n: stała, zmienna lub wyrażenie numeryczne określające ilość znaków, które mają być wprowadzone

```
 Do
  Form Input 10,Name$
  Print Name$
 Loop
```

78____KOMENDA____For__(for)_____F

Komenda powoduje otwarcie petli For...Next.
* Next

```
Składnia: For Var=A [Down]To B [Step S]
Skrót   : F Var=A [Down]To B [Step S]
```
Var: zmienna numeryczna dowolnego typu
A: stała, zmienna lub wyrażenie numeryczne określające wartość początkową zmiennej sterującej
B: stała, zmienna lub wyrażenie numeryczne określające wartość końcową zmiennej sterującej
S: stała, zmienna lub wyrażenie numeryczne określające krok zmiennej sterującej (w przypadku niepodania tej wartości - step = 1)

```
For I=3 To 4
 For J=5 Downto 3
  For K=1 To 7 Step 6
   For L=3 To 0 Step -1
    Print I,J,K,L
   Next L
  Next K
 Next J
Next I
Pause 50
'
T=Timer
For I=1 To 10000
Next I
Print (Timer-T)/200
'
T=Timer
For I%=1 To 10000
Next I%
Print (Timer-T)/200
```

79____FUNKCJA____Fre__(free)_____F

Funkcja podaje ilość wolnej pamięci RAM (w bajtach).
* Dfree

Składnia: Y=Fre(0)

```
Print Fre(0)
```

80____FUNKCJA____Frac__(fraction)_____F

Funkcja podaje część ułamkową liczby dziesiętnej.

Składnia: Y=Frac(X)

```
Print Frac(1.28)
Print Frac(-3.002)
Print Frac(Pi)
```

81____KOMENDA____Fullw__(full window)_____________________F

Komenda powoduje powiększenie okna o numerze n do pełnego ekranu.
* Openw

Składnia: Fullw n
Skrót   : Fu n
n: stała, zmienna lub wyrażenie numeryczne określające numer okna

```
Openw 2
Pause 100
Fullw 2
Pause 100
Closew 2
End
```

82____FUNKCJA____Gemdos__(gem disk operating system)___________G

Funkcja wywołuje zadaną procedurę GEMDOS-u.

Składnia: Y=Gemdos(opc,list of parameters)
opc: numer procedury
list of parameters: lista parmetrów

WYKAZ WAZNIEJSZYCH FUNKCJI GEMDOSU-u:

Opc(hex) Nazwa	Opis funkcji
00 Pterm()	opuszczenie wykonywanego programu
19 Dgetdrv()	funkcja podaje numer bieżącego napędu; 0=A, 1=B...,15=P
31 Ptermres(a,b:Li)	a=długość programu, b=adres powrotu; opuszczenie wykonywanego programu i pozostawienie go rezydentnym
36 Dfree(a:ALI,b:I)	a=adres bufora wyników, b=numer napędu; funkcja umieszcza w buforze wyników cztery długie słowa stanowiące ilość:wolnych klastrów, klastrów/dysk, bajtów/sektor, sektorów/na klaster
48 Malloc(a:Li;)	a=-1:podaje wolną pamięć; a=n bajtów: rezerwuje n bajtów, podaje adres zarezerwowanego obszaru

```
49 Mfree(a%Li%)      a=adres  obszaru pamieci otrzymany z Malloc;
                     funkcja zwalnia zarezerwowany  wcześniej ob-
                     szar pamieci

Print Fre(0)                        !wolna pamieć GFA
X=Gemdos(&h48,L%-1)                 !wolna pamięć systemu
Print X
If X<10000
 Reserve Fre(0)-10000
Endif
X=Gemdos(&h48,L%10000)              !rezerwacja 10000 bajtów
Print X                             !adres zarezerwowanego bloku
Print Gemdos(&h48,L%-1)
Void Gemdos(&h49,X)                 !zwolnienie pamięci
Print Gemdos(&h48,L%-1)
```

83____KOMENDA____Gemsys__(gem system)_____________________________G

Komenda wywołuje zadaną procedurę GEM-u (z biblioteki AES).

Składnia: Gemsys opc
opc: stała lub zmienna numeryczna dowolnego typu określająca kod procedury (numer procedury)

Pełny wykaz procedur GEM-u zawarty jest w książce: " ATARI ST GEM PROGRAMMER'S REFERENCE " Norberta Szczepanowskiego i Bernd'a Gunther'a wydanej przez Abacus Software w 1985.

```
Dpoke Gintin,320
Dpoke Gintin+2,100
Dpoke Gintin+4,5
Dpoke Gintin+6,5
Dpoke Gintin+8,0
Dpoke Gintin+10,0
Dpoke Gintin+12,639
Dpoke Gintin+14,199
'
Do
 Gemsys 73
 Pause 10
 Gemsys 74
 Pause 10
Loop
```

Komenda powoduje zapamiętanie prostokątnego fragmentu ekranu (bloku) w postaci zmiennej tekstowej.
* Put, Sget, Sput

Składnia: Get X0,Y0,X1,Y1,X$
X0,Y0,X1,Y1: stałe, zmienne lub wyrażenia numeryczne określające współrzędne wierzchołków prostokąta
X$: zmienna tekstowa

STRUKTURA ZMIENNEJ X$

```
Varptr(X$)     - Starszy bajt szerokości bloku
Varptr(X$)+1   - Młodszy bajt szerokości bloku
Varptr(X$)+2   - Starszy bajt wysokości bloku
Varptr(X$)+3   - Młodszy bajt wysokości bloku
Varptr(X$)+4   - 0
Varptr(X$)+5   - Rozdzielczość (4-LOW,2-MEDIUM,1-HIGH)
Varptr(X$)+6   - Treść obrazu
Varptr(X$)+7   - Treść obrazu
.............................
.............................
Varptr(X$) + Len(X$)
```

Wartość Len(X$) możemy obliczyć z zależności:

```
Len(X$)=6+((Dpeek(Varptr(X$))+15) Div 16)*2*(1+Dpeek(Varptr(X$) +
+ 2))*Dpeek(Varptr(X$)+4)
```

```
For I=1 To 5
 Circle 320,100,I*30
Next I
Get 0,0,320,199,A$
Put 320,0,A$,3
Pause 300
'
Cls
For I=1 To 5
 Box 20,10,280,80
Next I
'
Get 0,0,320,199,B$
Put 320,0,A$,3
```

```
Pause 300
'
Cls
Do
 For I=0 To 300
  Put I,100,A$
 Next I
Loop
```

85____KOMENDA____Get $__(get number)_____________________________G

Komenda powoduje zdjecie ze stosu zmiennej tekstowej.
* Field, Put $
Składnia: Get [$]n[,i]
n:stała, zmienna lub wyrażenie numeryczne określające numer pliku

Patrz: Field

86____KOMENDA____Gosub__(go to subroutine)_____________________G

Komenda powoduje wywołanie podprogramu.
* Procedure, Return

Składnia: Gosub name[(list of parameters)]
Skrót   : @ name [(list of parameters)]
name: nazwa procedury - dowolna nazwa nie zawierająca spacji
list of parameters: lista parametrów

```
Print "Main"
Gosub 1sub
@2sub(0,0)
End
'
Procedure 1sub
 Print "Procedure 1"
 @2sub(3,2)
 Print A,B
Return
'
Procedure 2sub(A,B)
 Print A,B
```

```
 Print "Procedure 2"
Return
```

87____KOMENDA____Goto__(go to)_____G

Komenda powoduje przekazanie sterowania do linii nastepnej po etykiecie.

Składnia: Goto Label
Skrót   : Got Label
Label: etykieta

```
Start:
Print I
Inc I
If I=100
 End
Endif
Goto Start
```

88____KOMENDA____Graphmode__(graphic mode)_____G

Komenda powoduje ustawienie zadanego trybu graficznego.

Składnia: Graphmode n
Skrót   : G n
n: stała, zmienna lub wyrażenie numeryczne określające tryb graficzny

CHARAKTERYSTYKA TRYBOW GRAFICZNYCH

n	Charakterystyka	Nazwa trybu
1	new=col AND obj	Tryb wymiany
2	new=(col And obj) OR (old AND NOT obj)	Tryb sumowania
3	new=obj XOR old	Tryb XOR
4	new=(old AND obj) OR (col AND NOT obj)	Tryb 2 z negacja

obj - objekt graficzny (linia, punkt, itp.)
col - kolor objektu
old - kolor tła
new - rezultat

```
Deffill 1,3,4
Pbox 10,50,150,125
Graphmode 4
Deffill 1,3,5
Pbox 120,100,200,150
Graphmode 3
Pbox 120,90,200,25
Graphmode 2
Deffill 3,3,9
Pbox 0,0,639,199
'
Do
Loop
```

89____KOMENDA____Hardcopy__(hard copy)_____________________________H

Komenda powoduje wykonanie kopii ekranu na drukarce.

Składnia: Hardcopy
Skrót   : H

```
Graphmode 3
For I=1 To 800 Step 8
 Box I Mod 640,I Mod 400,639-I Mod 640,399-I Mod 400
Next I
'
Hardcopy
For I=1 To 800 Step 8
 Box I Mod 640,I Mod 200,639-I Mod 640,199-I Mod 200
Next I
'
Hardcopy
```

90____FUNKCJA____Hex$__(hex dollar)_____________________________H

Funkcja tworzy łańcuch będący zapisem argumentu w układzie sze-
snastkowym.
* Bin$, Oct$

Składnia: A$=Hex$(X)
X: stała, zmienna lub wyrażenie numeryczne dowolnego typu

```
A=-1
B=&022
Print Hex$(A)
Print Hex$(234)
Print Hex$(B)
```

91____KOMENDA____Hidem__(hide mouse)_____H

Komenda powoduje wygaszenie wskaźnika myszy.
* Showm

Składnia: Hidem

```
Do
 Hidem
 Pause 100
 Showm
 Pause 100
Loop
```

92____FUNKCJA____Himem__(high memory)_____H

Funkcja podaje najwyższy adres dostępny dla Basic'a.

Składnia: Y=Himem

```
Print Fre(0),Himem
Print Lpeek(Basepage+&h4)
Reserve Fre(0)-1024
Print Fre(0),Himem
Reserve Fre(0)+1024
```

93____KOMENDA____If__(if)_____I

Komenda powoduje otwarcie struktury programowej If..Else..Endif.
* Else, Endif

Składnia:If condition [Then] programmblock [Else] [programmblock]
Endif
Skrót :I condition programmblock [El] [programmblock] En

condition:porównanie dwóch stałych, zmiennych lub wyrażeń numerycznych, tekstowych lub logicznych
programmblock: dowolny blok programowy

```
Do
 Input A,B
 If A=B
  Print "A=B"
 Else
  Print "A()B"
 Endif
Loop
```

94____KOMENDA____Inc__(increase)_____I

Komenda powoduje dodanie do zmiennej 'var' liczby 1.
* Add

Składnia: Inc var
Skrót : In var
var: zmienna numeryczna

```
T=Timer
For I%=1 To 10000
 Inc A%
Next I%
Print (Timer-T)/200
A%=0
'
T=Timer
For I%=1 To 10000
 A%=A%+1
Next I%
Print (Timer-T)/200
```

95____KOMENDA____Infow__(information line of window)_____I

Komenda wywołuje linię informacyjną w oknie o zadanym numerze.
* Openw

Składnia: Infow n,X$
n: stała, zmienna lub dowolne wyrażenie numeryczne określające

numer okna
X$: stała lub zmienna tekstowa stanowiąca treść linii informacyjnej

Patrz: Openw

96____FUNKCJA____Inkey$__(in key dollar)_____I

Funkcja przypisuje zmiennej tekstowej znak odpowiadający aktualnie wciśniętemu klawiszowi.

Składnia: X$=Inkey$

```
Do
 A$=Inkey$
 Print A$
Loop
```

97____FUNKCJA____Inp__(input)_____I

Instrukcja odczytuje bajt z urządzenia zewnętrznego.
* Out
Składnia: Y=Inp(X)
X: stała, zmienna lub wyrażenie numeryczne określające numer urządzenia zewnętrznego

```
Do
 Print Inp(2)
Loop
```

STANDARDOWE URZADZENIA WE/WY

n	Urządzenie zewnętrzne
0	CENTRONICS
1	RS-232
2	CONSOLA
3	MIDI
4	KLAWIATURA

98____KOMENDA____Input__(input)_____I

Instrukcja wstrzymuje wykonywanie programu i przypisuje zmiennej 'var' stałą wprowadzoną z klawiatury. Wprowadzanie danej należy zakończyć wciśnięciem klawisza RETURN.
* Inp

Składnia: Input ["text";(or,)]var[,var...]
Skrót : Inp ["text";(or,)]var[,var...]
text: dowolny tekst lub zmienna tekstowa opisująca wprowadzane dane
var: zmienne dowolnego typu

```
Do
 Input K
 Input "Twoje Imie";N$
 Input "",A,W$
 Print K,N$,A,W$
 Exit if K=0
Loop
```

99____FUNKCJA____Input$__(input dollar)_____I

Instrukcja wstrzymuje wykonywanie programu i przypisuje zmiennej 'var$' n znaków wprowadzonych z klawiatury.
Składnia: var$=Input$(n)
n: stała, zmienna lub dowolne wyrażenie numeryczne określające ilość wprowadzanych znaków

```
Print "Wprowadz 10 dowolnych znakow"
X$=Input$(10)
Print X$
```

100____KOMENDA____Inp #__(input number)_____I

Instrukcja odczytuje bajt z zadanego pliku.

Składnia: X=Inp(#n)
n: stała,zmienna lub wyrażenie numeryczne określające numer pliku

```
Open "O",#1,"DAT"
Print #1,"ABC"
```

```
Close #1
'
Open "I",#1,"DAT"
Print Inp(#1)
Close
```

101____FUNKCJA____Input$ #__(input dollar number)_____I

Instrukcja wstrzymuje wykonywanie programu i przypisuje zmiennej 'var$' n znaków wprowadzonych z zadanego pliku.

Składnia: X$=Input$(n,#p)
n: stała, zmienna lub dowolne wyrażenie numeryczne określajace ilość wprowadzanych znaków
p: stała, zmienna lub dowolne wyrażenie numeryczne określajace numer pliku

```
Open "O",#1,"DAT"
Print #1,"GfA-BASIC"
Close #1
'
Open "I",#1,"DAT"
Print Input$(3,#1)
Close
```

102____FUNKCJA____Inp?__(Input ?)_____I

Funkcja sprawdza możliwość przyjecia informacji z zadanego urządzenia zewnętrznego.

Składnia: Y=Inp?(X)
X: stała, zmienna lub dowolne wyrażenie numeryczne określajace numer urządzenia zewnętrznego

```
For I=0 To 4
 Print "Numer urzadzenia"'I,
 Print Inp?(I),
 Print Out?(I)
Next I
```

103____FUNKCJA____Instr__(in string)_____________________________I

Funkcja podaje miejsce wystąpienia danego znaku lub zestawu znaków w łańcuchu.

Składnia: Y=Instr([n,]A$,B$)
n: stała, zmienna lub dowolne wyrażenie numeryczne określające ilość znaków,które mają być pominięte przy porównywaniu (od lewej strony łańcucha)
A$: stała lub zmienna tekstowa stanowiąca łańcuch przeszukiwany
B$: stała lub zmienna tekstowa stanowiąca łańcuch poszukiwany

```
N$="GfA-BASIC"
S$="BASIC"
Print Instr(N$,"A")
Print Instr(4,N$,"A")
Print Instr("GfA-BASIC","fB")
Print Instr(N$,S$)
Print Instr(N$,"G")
```

104____FUNKCJA____Int__(integer)_____________________________I

Funkcja zaokrągla argument do najbliższej liczby całkowitej mniejszej od argumentu.
* Fix

Składnia: Y=Int(X)
X: stała, zmienna lub dowolne wyrażenie numeryczne

```
A=3.1415
Print Int(A)
Print Int(-A)
Print Int(678)
Print Int(-1.001)
Print Int(3/4)
```

105____KOMENDA____Kill__(kill)_____________________________K

Komenda powoduje usunięcie z dysku pliku o zadanej nazwie.

Składnia: Kill filespec
Skrót   : K filespec

filespec: stała lub zmienna tekstowa będąca nazwa pliku istniejącego na dysku

```
Open "O",#1,"TEST"
Close
Files
Kill "TEST"
Files
```

106____FUNKCJA____Left$__(left dollar)_____L

Funkcja wycina z lewej strony danego łańcucha określoną ilość znaków.

Składnia: X$=Left$(string[,n])
string: stała lub zmienna tekstowa
n: stała, zmienna lub wyrażenie numeryczne określające ilość znaków, które mają zostać wycięte

```
N$="GFA-BASIC"
Print Left$(N$)
Print Left$(N$,3)
Print Left$(N$,12)
Print Left$("GFA_BASIC",5)
```

107____FUNKCJA____Len__(lenght)_____L

Funkcja podaje długość łańcucha.

Składnia: Y=Len(X$)
X$: stała lub zmienna tekstowa

```
A$="GFA-"
Print Len(A$)
Print Len(A$+"BASIC")
Print Len(Left$(A$,2))
```

108____KOMENDA____Let__(let)

Komenda nadaje zmiennej 'var' zadana wartość.

Składnia: [Let] var=expression
var: zmienna dowolnego typu
expression: stała, zmienna lub wyrażenie dowolnego typu

```
Dim A(18)
Let A(15)=2*993
Let M$=" GfA "
B=A(15)
N$=M$
Print N$;B
```

109____KOMENDA____Line__(line)

Komenda rysuje linię łączącą dwa punkty.

Składnia: Line X0,Y0,X1,Y1
Skrót : Li X0,Y0,X1,Y1
X0,Y0,X1,Y1: stałe, zmienne lub wyrażenia numeryczne określające współrzędne łączonych punktów

```
Line 0,0,639,199
Line 639,0,0,199
Box 0,0,639,199
Pause 200
```

110____KOMENDA____Line Input____(line input)

Komenda wstrzymuje wykonywanie programu i przypisuje zmiennej 'var$' tekst wprowadzony z klawiatury (max. 256 znaków) łącznie ze znakami separującymi.

Składnia: Input var$
var$: zmienna tekstowa

```
Input "Wprowadz:A,B",A$
Print A$
Line Input "Wprowadz:A,B",A$
Print A$
```

111____KOMENDA____Line Input$__(line input number)_____L

Komenda wstrzymuje wykonywanie programu i przypisuje zmiennej 'var$' tekst (max.256 znakow) wprowadzony z pliku o zadanym numerze łącznie ze znakami separujacymi. Wprowadzanie danych jest zakończone w przypadku wczytania 256 znaków lub kodu &H0D (ENTER).

Składnia: Line Input #1,var$

112____KOMENDA____List__(list out)_____L

Komenda powoduje utworzenie na dysku pliku zawierającego listing programu.
* Llist

Składnia: List filename
Skrót : Lis filename
filename: stała lub zmienna tekstowa stanowiąca nazwę tworzonego pliku. Jeżeli filename="" otrzymamy listing programu na ekranie.

```
Print "Listing w formacie ASCII"
List "PROGR"
Print "Listing na ekranie"
List ""
```

113____KOMENDA____Llist__(line printer list out)_____L

Komenda powoduje wydruk listingu programu na drukarce.
* List

Składnia: Llist
Skrót : Ll

```
Print "Listing"
Llist
```

114____KOMENDA____Load__(load)_____L

Komenda powoduje wczytanie z dysku do pamięci programu o zadanej nazwie.
* Save

Składnia: Load filespec
Skrót   : Loa filespec
filespec: stała lub zmienna tekstowa określająca nazwę programu

```
Save "PROGRAM"
Load "PRO*"
```

115____FUNKCJA____Loc__(locate)_____L

Funkcja podaje położenie głowicy w napędzie względem początku zbioru.
* Seek

Składnia: Loc([#]n)
n:stała, zmienna lub wyrażenie numeryczne określające numer pliku

```
Open "O",#1,"DAT"
Print #1,"123456789"
Print Loc(#1)
Seek#1,4
Print Loc(#1)
A$=Input$(2,#1)
Print A$
Seek #1,0
Print Loc(#1)
Close
```

116____KOMENDA____Local__(local)_____L

Komenda pozwala zadeklarować zmienne lokalne wewnątrz procedury.

Składnia: Local var[,var...]
Skrót   : Loc var[,var...]
var: zmienne dowolnego typu

```
A=1000
Print A
Gosub P_1
Print A
End
```

```
Procedure P_1
 Local A
 I=I+1
 A=I
 Print A
 If I=10
 Else
  Gosub P_1
 Endif
Return
```

117____FUNKCJA____Lof__(lenght of file)_____L

Funkcja podaje długość zadanego pliku (w bajtach).

Składnia: X=Lof([#]n)
n:stała, zmienna lub wyrażenie numeryczne określające numer pliku

```
Open "O",#1,"DAT"
Print Lof(#1)
Print #1,"1234567"
Print Lof(#1)
```

118____FUNKCJA____Log__(logarithm)_____L

Funkcja oblicza logarytm naturalny argumentu.

Składnia: Log(X)
X: stała, zmienna lub dowolne wyrażenie numeryczne

```
A=4
Print Log(A)
Print Log(A*2)
```

119____FUNKCJA____Log10__(logarithm)_____L

Funkcja oblicza logarytm dziesiętny argumentu.

Składnia: Log10(X)
X: stała, zmienna lub dowolne wyrażenie numeryczne

```
Print Log10(10*10*10)
Print Log10(2.456)
```

120____KOMENDA____Loop__(loop)_____L

Komenda kończy pętle Do...Loop.
* Do

Składnia: Loop

Patrz: Do

121____FUNKCJA____Lpeek__(long peek)_____L

Funkcja podaje zawartość czterech kolejnych komórek pamięci począwszy od adresu równego argumentowi jako zmienną 32-bitową.
* Dpeek, Peek

Składnia: Y=Lpeek(X)
X: stała, zmienna lub dowolne wyrażenie numeryczne określające adres komórki

Patrz: Peek

122____KOMENDA____Lpoke__(long poke)_____L

Komenda wpisuje do czterech kolejnych komórek począwszy od wskanej, zadaną wartość traktowaną jako 32-bitową.
* Dpoke, Poke

Składnia: Lpoke X,Y
X,Y: stałe, zmienne lub dowolne wyrażenia numeryczne określające odpowiednio: adres komórki, zadaną wartość

Patrz: Poke

123____FUNKCJA____Lpos__(line printer position)_____L

Funkcja podaje położenie głowicy drukującej drukarki.

Składnia: Y=Lpos(n)
n: stała, zmienna lub dowolne wyrażenie numeryczne określające numer drukarki

```
For I=1 To 600
 Lprint "A";"
 If Lpos(1)=30 Then
  Lprint
 Endif
Next I
```

124____KOMENDA____Lprint__(line print)_____L

Komenda powoduje wydruk znaku na drukarce.
* Print

Składnia: Lprint [var[,][;]['][var...]
Skrót : Lpr [var[,][;]['][var...]
var: stała, zmienna lub wyrażenie dowolnego typu
[,][;]['] : patrz Print

```
Lprint "To jest test"
A=Sqr(3)
Lprint "A=";A'"A^2=";A^2
```

125____KOMENDA____Lset__(left set)_____L

Komenda wycina z lewej strony zmiennej tekstowej 'var$' tyle znaków, ile wynosi długość zmiennej 'set$'.
* Rset

Składnia: Lset set$=var$
Skrót : Ls set$=var$

```
A$="AAAAAAAAA"
B$=Space$(7)
C$="GfA"
Lset A$=C$
Lset B$="GfA-BASIC"
Print A$;B$
```

126____FUNKCJA____Max__(maximum)_____________________________M

Funkcja wybiera element o maksymalnej wartości z danego zbioru.
* Min

Składnia: Y=Max(var,[var,]...)
var: stała, zmienna lub wyrażenie dowolnego typu

```
A=17
B=3
A$="aa"
Print Max(A,B)
Print Max(A$,"b")
```

127____KOMENDA____Menu array$__(menu)_____________________________M

Komenda wyprowadza na ekran menu.

Składnia: Menu Array$()
Skrót : Me Array$()
Array$: tablica tekstowa zawierająca nazwy kolejnych pozycji menu
Ostatnie dwie pozycje Menu muszą być puste ("").

```
Dim Men$(15)
For I=0 To 13
 Read Men$(I)
Next I
Men$(I)=""
Men$(I+1)=""
Data Desk,  About,--------------------,1,2,3,4,5,6,""
Data File ,  Load ,  Save,""
Menu Men$()
Menu 11,1
Menu 12,2
Do
 On Menu
Loop
```

128____FUNKCJA____Menu__(menu)_____________________________M

Funkcja przekazuje informacje dotyczące obsługi programu za pomocą GEM-u.

Składnia: Y=Menu(X)
X: stała, zmienna lub dowolne wyrażenie numeryczne określające rodzaj informacji

ZNACZENIE FUNKCJI MENU(X)

1. Wybrano określoną pozycję Menu
   - Menu(1)=10
   - Menu(4)=numer pozycji+13
   - Menu(5)=numer grupy+2
2. Użytkownik chce uaktywnić okno
   - Menu(1)=20
   - Menu(4)=handle
3. Użytkownik chce zamknąć okno
   - Menu(1)=22
   - Menu(4)=handle
4. Użytkownik chce powiększyć okno do pełnego ekranu
   - Menu(1)=23
   - Menu(4)=handle
5. Użytkownik chce przesunąć zawartość okna
   - Menu(1)=24
   - Menu(4)=handle
   - Menu(5)=rodzaj przesunięcia
     - 0-strona do góry
     - 1-strona w dół
     - 2-linia do góry
     - 3-linia w dół
     - 4-strona w lewo
     - 5-strona w prawo
     - 6-kolumna w lewo
     - 7-kolumna w prawo
6. Użytkownik przesunął poziomy slider
   - Menu(1)=25
   - Menu(4)=handle
   - Menu(5)=względne położenie slider'a (0-1000)
7. Użytkownik przesunął pionowy slider
   - Menu(1)=26
   - Menu(4)=handle
   - Menu(5)=względne położenie slider'a (0-1000)
8. Użytkownik chce zmienić wymiary okna
   - Menu(1)=27
   - Menu(4)=handle
   - Menu(5)=bieżąca współrzędna X okna
   - Menu(6)=bieżąca współrzędna Y okna
   - Menu(7)=nowa szerokość okna
   - Menu(8)=nowa wysokość okna
9. Użytkownik chce przesunąć okno
   - Menu(1)=28

Menu(4)=handle
Menu(5)=nowa współrzędna X okna
Menu(6)=nowa współrzędna Y okna
Menu(7)=bieżąca szerokość okna
Menu(8)=bieżąca wysokość okna

Ponadto dostępne są następujące informacje:

Menu(0)=numer wybranej pozycji Menu
Menu(2)=numer aplikacji, której dotyczą pozostałe dane
Menu(9)=bit 0=1-użyto klawiatury
1=1-użyto klawiszy myszy
2=1-mysz jest wewnątrz 1-go zadanego prostokąta
3=1-mysz jest wewnątrz 2-go zadanego prostokąta
4=1-mysz użyto do obsługi okna
Menu(10)=współrzędna X myszy
Menu(11)=współrzędna Y myszy
Menu(12)=stan klawiszy myszy (1-lewy,2-prawy,3-oba klawisze)
Menu(13)=bit 0=1-wciśnięto prawy Shift
1=1-wciśnięto lewy Shift
2=1-wciśnięto Ctrl
3=1-wciśnięto Alt
Menu(14)=zawiera informacje o numerze wciśniętego klawisza
Menu(15)=informuje o ilości 'kliknięć' myszy

Patrz: On Menu Message, On Menu Key, On Menu Button

129____KOMENDA____Menu__(menu)_____M

Komenda ustala zadaną formę określonej pozycji Menu.
* Menu array$

Składnia: Menu X,Y
Skrót : Me X,Y
X,Y: stałe, zmienne lub wyrażenia numeryczne określające odpowiednio: numer pozycji Menu, formę tej pozycji

Y=0-kasuje znak odhaczenia
Y=1-pozycja odhaczona
Y=2-pozycja nieaktywna
Y=3-pozycja aktywna

130____KOMENDA____Menu Kill__(menu kill)_____________________M

Komenda dezaktywuje menu.
* On Menu

Składnia: Menu Kill

Patrz: On Menu

131____KOMENDA____Menu Off__(menu off)_____________________M

Komenda wygasza zapalone pozycje menu.
* On Menu

Składnia: Menu Off

Patrz: On Menu

132____Funkcja____Mid$__(middle dollar)_____________________M

Funkcja wycina ze zmiennej 'var$' zadaną ilość znaków poczynając od wskazanego miejsca.

Składnia: X$=Mid$(var$,A[,N])
var$= stała lub zmienna tekstowa
A: stała, zmienna lub dowolne wyrażenie numeryczne określające,od którego znaku ma nastąpić wycinanie
N: stała, zmienna lub dowolne wyrażenie numeryczne określające ilość znaków, które mają zostać wycięte (jeśli nie podamy to zostaną wycięte wszystkie znaki do końca zmiennej var$)

```
N$="GfA-BASIC"
Print Mid$(N$,5)
Print Mid$(N$,1,3)
Print Mid$(N$,0,3)
Print Mid$(N$,3,12)
```

133____KOMENDA____Mid$__(middle dollar)_____________________M

Komenda wymienia w zmiennej 'var$' zadaną ilość znaków poczynając od wskazanego miejsca.

Składnia: Mid$(var$,A[,N])=X$
var$= stała lub zmienna tekstowa
A: stała, zmienna lub dowolne wyrażenie numeryczne określające,od którego znaku ma nastąpić wymiana
N: stała, zmienna lub dowolne wyrażenie numeryczne określające ilość znaków, które mają zostać wymienione (jeśli nie podamy to zostaną wymienione wszystkie znaki do końca zmiennej var$)

```
N$="AAAAAAAAA"
Mid$(N$,5,2)="BB"
Print N$
```

134____FUNKCJA____Min__(minimum)_____M

Funkcja wybiera element o minimalnej wartości z danego zbioru.
* Max

Składnia: Y=Min(var,[var,]...)
var: stała, zmienna lub wyrażenie dowolnego typu

```
A=17
B=3
A$="aa"
Y=Min(A,B)
Print Y
Print Min(A$,"b")
```

135____KOMENDA____Mkdir__(make directory)_____M

Komenda tworzy folder na dysku.

Składnia: Mkdir foldername
Skrót : Mk foldername
foldername: stała lub zmienna określająca nazwę folderu

Patrz: Rmdir

136____FUNKCJA____Mk*$__(make dollar)_____M

Funkcja zamienia zmienne tekstowe na numeryczne.
* Cv*$

```
                 ________INTEGER_____   ___________REAL___________
Składnia:      X$=Mki$(X)   Mkl$(X)    Mks$(X)    Mkf$(X)    Mkd$(X)
Dług.łańcucha:2 bajty       4 bajty    4 bajty    6 bajtów   8 bajtów
X: stała lub zmienna numeryczna

 A=65
 A$=Mki$(65)
 Print A$
 X=Cvi(A$)
 Print X


137____KOMENDA____Mouse__(mouse)_____________________________________M

Komenda  podstawia  za zmienne  'varx','vary','vark' odpowiednio:
współrzędne X, Y, stan przycisków myszy.
* Mousex, Mousey, Mousek

Składnia: Mouse varx,vary,vark
Skrót   : M varx,vary,vark
varx,vary,vark: zmienne numeryczne dowolnego typu

 Do
  Mouse A,B,C
  Print At(1,1);A,B,C
 Loop


138____FUNKCJA____Mouse[x],[y],[k]__(mouse)_____________________M

Funkcja podaje współrzedną x,y oraz stan przycisków myszy.
* Mouse

Składnia: X=Mousex    Y=Mousey    K=Mousek
X,Y,K: zmienne numeryczne dowolnego typu
K=0-żaden przycisk nie wciśnięty
K=1-lewy przycisk
K=2-prawy przycisk
K=3-oba przyciski wciśnięte

 Do
  X=Mousex
  Print At(1,1);X,
 Loop
```

139____KOMENDA____Mul__(multiplication)_____M

Komenda powoduje mnożenie zmiennej 'var' przez wartość n.

Składnia: Mul var,n
var: zmienna numeryczna
n: stała, zmienna lub wyrażenie numeryczne dowolnego typu

```
Dim A%(10000)
T=Timer
For I=1 To 10000
 Mul A%(I),5
Next I
Print (Timer-T)/200
T=Timer
For I=1 To 10000
 A%(I)=A%(I)*5
Next I
Print (Timer-T)/200
```

140____KOMENDA____Name__(name)_____N

Komenda zmienia nazwę pliku.

Składnia: Name "oldfile" As "newfile"
Skrót   : Na "oldfile" As "newfile"
oldfile,newfile: stałe lub zmienne tekstowe

```
Open "O",#1,"Old"
Files
Name "Old" As "New"
Files
```

141____KOMENDA____New__(new)_____N

Komenda usuwa z pamięci aktualny program.

Składnia: New

```
Print "Test New"
List""
Pause 100
New
```

142____KOMENDA____Next__(next)_____N

Komenda kończy pętlę For...Next.

Składnia: Next var
Skrót   : N var
var: zmienna sterująca pętli

Patrz: For

143____FUNKCJA____Oct$__(octonary dollar)_____O

Funkcja tworzy łańcuch będący zapisem argumentu w układzie ósemkowym.
* Bin$, Hex$

Składnia: X$=Oct$(X)
X: stała, zmienna lub wyrażenie numeryczne dowolnego typu

```
A=-1
B=&H22
Print Oct$(A)
Print Oct$(234)
Print Oct$(B)
```

144____FUNKCJA____Odd__(odd)_____O

Funkcja przyjmuje wartość -1 dla argumentów będących liczbami nieparzystymi.

Składnia: Y=Odd(X)
X: stała, zmienna lub wyrażenie numeryczne dowolnego typu

```
For I=0 To 10
 Print I,Odd(I)
Next I
```

145____KOMENDA____On Break__(on break)_____O

Komenda powoduje wyłączenie On Break Cont oraz On Break Gosub.
* On Break Cont, On Break Gosub

Składnia: On Break

Patrz: On Break Gosub

146____KOMENDA____On Break Gosub__(on break go to subroutine)___0

Komenda powoduje wywołanie podprogramu w przypadku próby zatrzymania programu klawiszami Ctrl+Shift+Alt.

Składnia: On Break Gosub name
name: nazwa procedury (patrz Gosub)

```
On Break Gosub Pro_break
Print "zatrzymaj mnie"
'
Do
Loop
'
Procedure Pro_break
 Alert 3,"Chciales zatrzymac!   program",1," OK ",I%
 On Break
Return
```

147____KOMENDA____On Break Cont__(on break continue)_____________0

Komenda blokuje możliwość zatrzymania programu.

Składnia: On Break Cont

```
On Break Cont
Print At(1,1);"Zatrzymaj program. (Wcisnij przycisk myszy)"
Do
 If Mousek
  Stop
 Endif
Loop
```

148____KOMENDA____On Error__(on error)_____________________________0

Komenda wyłącza programową obsługę błędów.
* Error, Err

Składnia: On Error

Patrz: On Error Gosub

149____KOMENDA____On Error Gosub__(on error go to subroutine)___0

Komenda powoduje wywołanie podprogramu w przypadku wystąpienia błędu.
* Error, Err

Składnia: On Error Gosub name
name: nazwa procedury (patrz Gosub)

```
On Error Gosub errorroutine
Print Sqr(-1)
Print 3/0
'
Procedure errorroutine
 Print " error nr.:";Err
 On Error
 Resume Next
Return
```

150____KOMENDA____On Menu__(on menu)_____0

Komenda sprawdza występowania warunków niezbędnych do wywołania podprogramów poprzez instrukcje typu On Menu... .
* On Menu...

Składnia: On Menu

Patrz: On Menu Gosub

151____KOMENDA____On Menu Button__(on menu button)_____0

Komenda powoduje wywołanie podprogramu w przypadku użycia klawiszy myszy.

Składnia: On Menu Button 0,X,Y Gosub name
X,Y: stałe, zmienne lub wyrażenia numeryczne określające kombinacje klawiszy, na które zareaguje program

X	Y	0	1	2	3
0	0	+	+	+	+
0	1	+	+	+	+
0	2	+	+	+	+
0	3	+	+	+	+
1	0	+	-	+	-
1	1	-	+	-	+
1	2	+	-	+	-
1	3	-	+	-	+
2	0	+	+	-	-
2	1	+	+	-	-
2	2	-	-	+	+
2	3	-	-	+	+
3	0	+	-	-	-
3	1	-	+	-	-
3	2	-	-	+	-
3	3	-	-	-	+

0-żaden klawisz nie wciśnięty
1-lewy klawisz wciśnięty
2-prawy klawisz wciśnięty
3-oba klawisze wciśnięte
+-oznacza reakcję programu

```
On Menu Button 0,3,1 Gosub A
Do
 On Menu
Loop
'
Procedure A
 Print Mousek,Menu(15)
Return
```

152____KOMENDA____On Menu Gosub__(on menu go to subroutine)_____0

Komenda powoduje wywołanie podprogramu w momencie wybrania określonej pozycji z menu.
* On Menu...

Składnia: On Menu Gosub name
name: nazwa procedury (patrz: Gosub)

```
Dim Men$(16)
For I=0 To 14
 Read Men$(I)
Next I
Men$(I)=""
Men$(I+1)=""
'
Data Desk,  About
Data ----------------------
Data 1,2,3,4,5,6,""
Data File , Load , Save, Kill,""
'
On Menu Gosub Proc1
Menu Men$()
'
Do
 On Menu
Loop
'
Procedure Proc1
 Menu Off
 M=Menu(0)
 If M=1
  Alert 0,"komenda On Menu Gosub",1," OK ",J%
 Else
  Print At(1,14);"Pozycja menu nr:"'M'Men$(M)
  If M=13
   Menu Kill
  Endif
 Endif
Return
```

153____KOMENDA____On Menu Key__(on menu key)_____________________0

Komenda powoduje wywołanie podprogramu w chwili wciśnięcia dowolnego klawisza.
* On Menu...

Składnia: On Menu Key Gosub name
name: nazwa procedury (patrz: Gosub)

```
On Menu Key Gosub Pro_key
Print "Wcisnij dowolny klawisz"
'
Do
 On Menu
Loop
'
Procedure Pro_key
 M=Menu(14)
 Print M
Return
```

154____KOMENDA____On Menu Ibox__(on menu inside box)_____________0

Komenda powoduje wywołanie podprogramu w chwili, gdy wskaźnik myszy znajdzie się wewnątrz zadanego prostokąta.
* On Menu...

Składnia: On Menu Ibox A,X,Y,W,H Gosub name
A,X,Y,W,H: stałe, zmienne lub wyrażenia numeryczne określające odpowiednio: numer podprogramú Ibox (Obox), współrzędne X,Y lewego górnego rogu prostokąta, szerokość i wysokość prostokąta

```
On Menu Ibox 1,100,50,440,100 Gosub Pro_1
On Menu Obox 1,100,50,440,100 Gosub Pro_2
Box 100,50,540,150
'
Do
 On Menu
Loop
'
Procedure Pro_1
 Defmouse 2
 On Menu Obox 1,100,50,440,100 Gosub Pro_2
Return
'
Procedure Pro_2
 Defmouse 0
 On Menu Ibox 1,100,50,440,100 Gosub Pro_1
Return
```

155____KOMENDA____On Menu Obox__(on menu outside box)__________0

Komenda powoduje wywołanie podprogramu w chwili, gdy wskaźnik myszy znajdzie się na zewnątrz zadanego prostokąta.
* On Menu...

Składnia: On Menu Ibox A,X,Y,W,H Gosub name
A,X,Y,W,H: jak w komendzie On Menu Ibox

Patrz: On Menu Ibox

156____KOMENDA____On Menu Message__(on menu message)__________0

Komenda powoduje wywołanie podprogramu w przypadku, gdy wskaźnik myszy znajdzie się wewnątrz pola służącego do obsługi okna.
* On Menu

Składnia: On Menu Message Gosub name
name: nazwa procedury (patrz: Gosub)

```
Menu Kill
On Menu Message Gosub Mess
Titlew 1," Test "
Infow 1," Linia informacyjna"
Handle=Windtab
Attr=Windtab+2
Xpos=Windtab+4
Ypos=Windtab+6
Width=Windtab+8
Hight=Windtab+10
Dpoke Attr,&X111111101111
Dpoke Xpos,160
Dpoke Ypos,50
Dpoke Width,300
Dpoke Hight,100
Openw 1
Clearw 1
'
Do
 On Menu
Loop
'
Procedure Mess
 M%=Menu(1)
```

```
If M%=23
 Fullw 1
 Clearw 1
Endif
If M%=27
 Closew 1
 Closew 0
 Cls
 Dpoke Width,Menu(7)
 Dpoke Hight,Menu(8)
 Openw 1
 Clearw 1
 @Slider(Vslider,9)
 @Slider(Hslider,8)
Endif
If M%=28
 Closew 1
 Closew 0
 Cls
 Dpoke Xpos,Menu(5)
 Dpoke Ypos,Menu(6)
 Openw 1
 Clearw 1
Endif
If M%=22
 Alert 3," Ta funkcja zamyka okno",2,"OK!CANCEL",J%
 If J%=1
  Closew 1
  End
 Endif
Endif
If M%=24 And Menu(5)=3
 If Vslider<1000
  Add Vslider,10
 Endif
 @Slider(Vslider,9)
Endif
If M%=24 And Menu(5)=2
 If Vslider>0
  Sub Vslider,10
 Endif
 @Slider(Vslider,9)
Endif
'
```

```
  If M%=26
   Vslider=Menu(5)
   @Slider(Vslider,9)
  Endif
  If M%=24 And Menu(5)=7
   If Hslider<1000
    Add Hslider,10
   Endif
   @Slider(Hslider,8)
  Endif
  If M%=24 And Menu(5)=6
   If Hslider>0
    Sub Hslider,10
   Endif
   @Slider(Hslider,8)
  Endif
  If M%=25
   Hslider=Menu(5)
   @Slider(Hslider,8)
  Endif
 Return
 '
 Procedure Slider(Slider,Mode)
  Dpoke Contrl+2,6
  Dpoke Contrl+4,1
  Dpoke Contrl+6,0
  Dpoke Contrl+8,0
  Dpoke Gintin,Dpeek(Handle)
  Dpoke Gintin+2,Mode
  Dpoke Gintin+4,Slider
  Dpoke Gintin+6,0
  Dpoke Gintin+8,0
  Dpoke Gintin+10,0
  Gemsys 105
 Return
```

157____KOMENDA____On X Gosub__(on x go to subroutine)__________0

W zależności od wartości X komenda wywołuje odpowiedni podprogram.

Składnia: On X Gosub name1[,name2,....]
X: stała, zmienna lub wyrażenie numeryczne (dla X<1 i dla X> ilości procedur (name) komenda nie jest wykonywana)

name1,name2...: nazwy procedur (patrz Gosub)

```
Do
 Input A
 On A Gosub Pro1,Pro2
Loop
'
Procedure Pro1
 Print "1<=A<2"
Return
'
Procedure Pro2
 Print "2<=A<3"
Return
```

158____KOMENDA____Open__(open)_____0

Komenda otwiera plik na dysku.
* Close

Składnia: Open "Mode",[#]n,filename[,len]
Skrót : O "Mode",[#]n,filename[,len]
filename: stała lub zmienna tekstowa określająca nazwę pliku
len: stała, zmienna lub wyrażenie numeryczne określające długość pliku

Mode	Nazwa	Zastosowanie
I	INPUT	tylko do czytania
O	OUTPUT	tylko do zapisu
R	RANDOM	o swobodnym dostępie (do czytania i zapisu)
A	APPEND	tylko dla plików istniejących, dane są dopisywane
U	UPDATE	tylko dla plików istniejących, o swobodnym dostępie

```
Open "O",#1,"NAME"
Print #1,"GfA-BASIC"
Open "I",#2,"NAME"
Do
 Exit If Eof(#2)
 Print Input$(1,#2)
Loop
Close
```

159____KOMENDA____Openw__(open window)_____________________________0

Komenda powoduje otwarcie okna o zadanym numerze i określonych wymiarach.
* Fullw, Closew, Windtab

Składnia: Openw n[,X,Y]
Skrót : O w n[,X,Y]
n,X,Y: stałe, zmienne lub wyrażenia numeryczne określające odpowiednio: numer okna, szerokość i wysokość

Openw 0 otwiera okno o wymiarach pełnego ekranu pomniejszone o wysokość linii zajmowanej przez Menu. Punkt o współrzędnych 0,0, przesunięty jest wówczas do punktu 0,11.

```
Openw 1,200,100
Pause 250
Closew 1
'
Openw 2,110,150
Pause 250
Closew 2
```

Patrz Windtab

160____KOMENDA____Option Base__(option base)_____________________0

Komenda ustala bazę zmiennych tablicowych.
* Dim

Składnia: Option Base n
n: stała, zmienna lub dowolne wyrażenie numeryczne określające bazę (pierwszy element) tablicy.

```
Option Base 0
Dim A(10)
Print Dim?(A())
Erase A()
'
Option Base 1
Dim A(10)
Print Dim?(A())
```

161____KOMENDA____Out__(output)_____________________________0

Komenda wysyła bajt do urządzenia zewnętrznego.
* Inp

Składnia: Out X,Y
Skrót   : Ou X,Y
X: numer urządzenia zewnętrznego (patrz: Inp)
Y: bajt wysyłany do urządzenia zewnętrznego

```
Out 2,66
```

162____KOMENDA____Out #__(output number)_____________________0

Komenda wysyła bajt do zadanego pliku.
* Inp#

Składnia: Out #n,Y
n: stała,zmienna lub wyrażenie numeryczne określające numer pliku
Y: bajt wysyłany do urządzenia zewnetrznego

```
Open "O",#1,"DAT"
Out #1,65
Close #1
'
Open "I",#1,"DAT"
Print Inp(#1)
Close
```

163____FUNKCJA____Out?__(out ?)_____________________________0

Funkcja sprawdza możliwość wysłania bajtu do urządzenia zewnętrznego (przyjmuje wartość -1, gdy taka możliwość istnieje).

Składnia: Y=Out?(X)
X: numer urządzenia zewnętrznego (patrz: Inp)

```
For I=0 To 4
 Print I,Out?(I)
Next I
```

164____KOMENDA____Pause__(pause)_____P

Komenda powoduje wstrzymanie wykonywania programu na zadany okres czasu.

Składnia: Pause X
Skrót   : Pa X
X:stała,zmienna lub wyrażenie numeryczne określające czas wstrzymania wykonywania programu (jedna jednostka=1/50 sek)

```
Print "Czekaj 5 sekund"
Pause 250
Print "Minelo 5 sekund"
```

165____KOMENDA____Pbox__(plane box)_____P

Komenda rysuje wypełniony prostokat o wierzchołkach w punktach X0,Y0,X1,Y1.
* Box, Rbox

Składnia: Pbox X0,Y0,X1,Y1
Skrót   : Pb X0,Y0,X1,Y1
X0,Y0,X1,Y1:stałe,zmienne lub wyrażenia numeryczne dowolnego typu

```
Pbox 0,0,300,100
Deffill 1,2,2
Pbox 10,10,305,105
Pbox 400,200,133,90
```

166____KOMENDA____Pcircle__(plane circle)_____P

Komenda rysuje wypełniony łuk o współrzednych środka X,Y, promieniu R, kącie początkowym Fi1 oraz kącie końcowym Fi2.
* Circle, Ellipse, Pellipse

Składnia: Pcircle X,Y,R(,Fi1,Fi2)
Skrót   : Pc X,Y,R(,Fi1,Fi2)
X,Y,R,Fi1,Fi2: jak dla komendy Circle

```
Pcircle 160,100,100,450,2700
Deffil 1,2,4
Pcircle 100,50,45
```

```
Deffil 1,1,1
Pcircle 150,120,100,0,3500
```

167____FUNKCJA____Peek__(peek)_____P

Funkcja podaje zawartość komórki o adresie równym argumentowi.
* Dpeek, Lpeek

Składnia: Y=Peek(X)
X: stała, zmienna lub dowolne wyrażenie numeryczne określające adres komórki

```
A$="A"
D=Arrptr(A$)
A=Lpeek(D)
L=Dpeek(D+4)
C=Peek(A)
'
Print "ADDRESS"''A
Print "LENGHT"''L
Print "ASCII"''C
```

168____KOMENDA____Pellipse__(plane ellipse)_____P

Komenda rysuje wypełniony wycinek elipsy o współrzędnych środka X,Y, promieniach RX,RY, kącie początkowym Fi1 oraz kącie końcowym Fi2.
* Circle, Ellipse, Pcircle

Składnia: Pellipse X,Y,RX,RY(,Fi1,Fi2)
Skrót : Pe X,Y,RX,RY(,Fi1,Fi2)
X,Y,RX,RY,Fi1,Fi2: jak dla komendy Ellipse

```
  Pellipse 320,100,300,75
```

169____KOMENDA____Plot__(plot)_____P

Komenda rysuje punkt o zadanych współrzędnych.

Składnia: Plot X,Y
Skrót : Pl X,Y

X,Y: stałe, zmienne lub wyrażenia numeryczne dowolnego typu określające współrzędne punktu

```
For X=0 To 639
 Y=90*Sin(X*Pi/180)
 Plot X,Y+100
Next X
'
Do
Loop
'
End
```

170____FUNKCJA____Point__(point)_____P

Funkcja podaje kolor zadanego punktu.

Składnia: C=Point(X,Y)
X,Y: stałe, zmienne lub wyrażenia numeryczne dowolnego typu określające współrzędne punktu

```
Plot 100,100
Print Point(99,100)
Print Point(100,100)
Pause 150
```

171____KOMENDA____Poke__(poke)_____P

Komenda wpisuje do wskazanej komórki zadaną wartość.
* Dpoke, Lpoke

Składnia: Poke X,Y
X,Y: stałe, zmienne lub dowolne wyrażenia numeryczne określające odpowiednio: adres komórki, wartość która ma być do tej komórki wpisana (Y<256)

```
A$="A"
L=Arrptr(A$)
Z=Varptr(A$)
Dpoke L+4,4
Lpoke Z,1111638594
Print A$
```

```
Poke Z,57
Print A$
```

172____KOMENDA____Polyfill__(poly fill)_____________________P

Komenda łączy linią zadaną liczbę punktów o współrzędnych zapamiętanych w tablicach oraz wypełnia tak utworzoną figurę zadanym deseniem.

Składnia: Polyfill N,X(),Y() [Offset X0,Y0]
Skrót : Polyf N,X(),Y() [Offset X0,Y0]
N,X0,Y0: stałe,zmienne lub wyrażenia numeryczne określające odpowiednio: ilość punktów, przesunięcie X, przesunięcie Y
X(),Y(): nazwy tablic zawierających współrzędne kolejnych punktów

```
Dim X(8),Y(8)
For I=0 To 8
 X(I)=320+150*Sin(I*2*Pi/8)
 Y(I)=100+75*Cos(I*2*Pi/8)
Next I
Deffill 1,2,9
Polyfill 5,X(),Y()
'
Do
Loop
```

173____KOMENDA____Polyline__(poly line)_____________________P

Komenda łączy linią zadaną liczbę punktów o współrzędnych zapamiętanych w tablicach.

Składnia: Polyline N,X(),Y() [Offset X0,Y0]
Skrót : Pol N,X(),Y() [Offset X0,Y0]
N,X0,Y0: jak dla komendy Polyfill

```
Dim X(3),Y(3)
For I=0 To 3
 X(I)=20+100*I
 Y(I)=190-10*I^2
Next I
Polyline 4,X(),Y() Offset 100,-50
Pause200
```

174____KOMENDA____Polymark__(poly mark)_____P

Komenda rysuje zadaną liczbę znaczników w punktach o współrzędnych zapamiętanych w tablicach.

Składnia: Polymark N,X(),Y() [Offset X0,Y0]
Skrót : Polym N,X(),Y() [Offset X0,Y0]
N,X0,Y0: jak dla komendy Polyfill

```
Dim X(8),Y(8)
For I=0 To 8
 X(I)=320+150*Sin(I*2*Pi/8)
 Y(I)=100+75*Cos(I*2*Pi/8)
Next I
Defmark 1,3,30
Polymark 9,X(),Y() Offset 150,30
Do
Loop
```

175____FUNKCJA____Pos__(position)_____P

Funkcja podaje poziomą pozycję kursora względem punktu o pozycji podanej w opcji At.

Składnia: Y=Pos(0)

```
Print At(10,1);"XXX";
Y=Pos(0)
Print Y
```

176____KOMENDA____Prbox__(plane radius box)_____P

Komenda rysuje wypełniony prostokąt o zaokrąglonych rogach o wierzchołkach w punktach X0,Y0,X1,Y1.
* Pbox, Rbox

Składnia: Prbox X0,Y0,X1,Y1
Skrót : Prb X0,Y0,X1,Y1
X0,Y0,X1,Y1: stała,zmienna lub wyrażenie numeryczne dowolnego typu

```
Deffill 1,2,2
Prbox 400,200,133,90
```

177____KOMENDA____Print__(print)_____P

Komenda drukuje stałe lub zmienne dowolnego typu na ekranie.

Składnia: Print [At(X,Y)][Tab(X)][;][,][']var[;][,]['']var...]
Skrót   : P [At(X,Y)][Tab(X)][;][,][']var[;][,]['']var...]
X,Y: stałe, zmienne lub dowolne wyrażenia numeryczne określające
odpowiednio: numer kolumny, numer wiersza na ekranie
var: stałe, zmienne lub wyrażenia dowolnego typu

```
A$="GfA"
B=1986
Print A$'
Print B,B;Tab(20);A$
Print B;B
Print B'B
Print A$,B;"GfA"''B
Print At(77,25);A$;
```

Parametry dla opcji At liczone są w poziomie: od 1 do 40 dla rozdzielczości niskiej, od 1 do 80 dla rozdzielczości średniej i wysokiej; w pionie od 1 do 25 dla każdej z rozdzielczości.

178____KOMENDA____Print #__(print number)_____P

Komenda zapisuje stałe i zmienne dowolnego typu na dysku.

Składnia: Print #n,var[;var]
Skrót   : P #n,var[;var]
n: stała, zmienna lub wyrażenie numeryczne określające numer pliku
var: stałe lub zmienne dowolnego typu

Patrz: Open #

179____KOMENDA____Print Using__(print using)_____P

Komenda wyprowadza stałe, zmienne i wyrażenia dowolnego typu według zadanego formatu.
* Print # Using

Składnia: Print Using "format",liste[;]

Formaty dla zmiennych łańcuchowych:

Symbol	Działanie
!	Wyprowadza pierwszy znak Print Using "!","TEST" T
\AAAAAAAA\ n znaków	Wyprowadza n+2 znaki Print Using "\aaa\","TO JEST TEST" TO JE
& tekst	Wyprowadza wszystkie znaki + tekst Print Using "& $","1000" 1000 $

Formaty dla zmiennych numerycznych:

Symbol	Działanie
#	Ilość znaków '#' oznacza ilość wyprowadzonych cyfr. Znak '.' oznacza punkt dziesiętny. Print Using "###.#",123.48 123.5
+	Znaki '+'i'-' dodawane sa przed każdą zmienną num. Print Using "+####",100 +100
-	Znak '-' dodawany jest przed każdą zmienną ujemną Print Using "-####",-100 -100
*	Spacje miedzy danymi wypełniane są znakiem '*' Print Using "*####",10 ***10
$	Znak '$' dodawany jest przed każdą daną numeryczną Print Using "$####",100 $100
,	Znak ',' jest wstawiany w określonym miejscu. Print Using "#,####",10000 1,0000
^	Dana wyprowadzana jest w postaci wykładniczej. Print Using "##.##^^^^",100 10.00E+02

```
  Print Using "!","Test"
  Print Using "\aaa\","TO JEST TEST"
  Print Using "& $","1000"
  Print Using "##.##",2.5555
  Print Using "+###",10
  Print Using "##.##^^^^",1.32
  Print Using "###.##",Pi*100
  Print Using "A!b","lambda"
  Print Using "-####",-100
  Print Using "$####",100
  Print Using "#,####",10000
```

180____KOMENDA____Print # Using__(print number using)_____________P

Komenda zapisuje na dysku stałe i zmienne dowolnego typu według ustalonego formatu.

Składnia: Print #n,Using "format",var[;var]
Skrót : P #n,Using "format",var[;var]
n: stała,zmienna lub wyrażenie numeryczne określające numer pliku
format: patrz Print Using
var: stałe lub zmienne dowolnego typu

```
  A=10.123456
  Open "O",#1,"TEST"
  Print #1,Using "##.##",A
  Close
  '
  Open "I",#1,"TEST"
  Input #1,A
  Print A
  Close
```

181____KOMENDA____Procedure___(procedure)_____________________P

Komenda otwiera procedure.
* Return

Składnia: Procedure name[(list of parameters)]
Skrót : Pro name [(list of parameters)]
name: nazwa procedury - dowolna nazwa nie zawierająca spacji

```
Do
 Input"Podaj promien";A
 Gosub Kolo(A)
 Print "Pole="'Pole"
Loop
'
Procedure Kolo(R)
 Pole=2*Pi*R
Return
```

182____KOMENDA____Psave__(protect save)_____________________P

Komenda zapisuje program w formie zabezpieczonej na dysku.
* Save

Składnia: Psave filename
Skrót   : Ps filename
filename: stała lub zmienna tekstowa określająca nazwę programu.

```
Circle 100,100,50
Pause 200
Edit
```

W trybie bezpośrednim wydaj komendy:

```
Psave "Test"
New
Load "Test"
```

183____KOMENDA____Put__(put)_____________________P

Komenda powoduje wywołanie prostokątnego fragmentu ekranu (bloku) w zadanym miejscu.
* Get, Sget, Sput

Składnia: Put X0,Y0,X$[,M]
Skrót   : Pu X0,Y0,X$[,M]
X$: zmienna tekstowa
X0,Y0,M: stałe, zmienne lub wyrażenia numeryczne określające współrzędne wierzchołków prostokąta oraz sposób wklejenia bloku

M	Sposób wklejenia bloku
0	r = 0
1	r = q AND z
2	r = q AND (NOT z)
3	r = q
4	r = (NOT q) AND z
5	r = z
6	r = q XOR z
7	r = q OR z
8	r = NOT (q OR z)
9	r = NOT (q XOR z)
10	r = NOT z
11	r = q OR (NOT z)
12	r = NOT q
13	r = (NOT q) OR z
14	r = NOT (q AND z)
15	r = 1

q-blok źródłowy
z-blok przeznaczenia
r-blok wypadkowy

Patrz: Get

184____KOMENDA____Put #__(put number)_____P

Komenda powoduje zapis na stosie zmiennej tekstowej.
* Field, Get #

Składnia: Put [#]n[,i]
n: stała,zmienna lub wyrażenie numeryczne określające numer pliku

Patrz: Field

185____KOMENDA____Quit__(quit)_____Q

Komenda powoduje powrót do TOS-u.

Składnia: Quit
Skrót   : Q

Quit

186____FUNKCJA____Random__(random)_____R

Funkcja losowa.
* Rnd

Składnia: L=Random(X)
X: stała, zmienna lub wyrażenie określające przedział wartości funkcji
L: wartość funkcji-liczba całkowita zawierająca się w przedziale od 0 do X-1

```
Repeat
 L=Random(6)+1
 Print L
Until L=3
```

187____KOMENDA____Rbox__(radius box)_____R

Komenda rysuje prostokąt z zaokrąglonymi rogami o wierzchołkach w punktach X0,Y0,X1,Y1.
* Pbox, Prbox

Składnia: Rbox X0,Y0,X1,Y1
Skrót : Rb X0,Y0,X1,Y1
X0,Y0,X1,Y1: jak w komendzie Box

```
Rbox 0,0,300,100
Rbox 10,10,305,105
Rbox 400,200,133,90
Defline 3
Rbox 100,90,530,110
```

188____KOMENDA____Read__(read)_____R

Komenda czyta zmienną z linii Data i jej wartość przypisuje zmiennej'var'.
* Data, Restore

Składnia: Read var[,var]...
Skrót : Rea var[,var]...

Patrz: Data

189____KOMENDA____Relseek__(relative seek)_____R

Komenda przesuwa głowicę w dysku o określoną liczbę danych w prawo lub lewo od pozycji bieżącej.
* Seek

Składnia: Relseek [#]n,X
Skrót   : Rel [#]n,X
n,X: stałe, zmienne lub wyrażenia numeryczne określające odpowiednio: numer pliku, przesunięcie względne głowicy

```
Open "O",#1,"DAT"
Print #1,"1234567890"
Seek #1,8
Print Loc(#1)
Relseek #1,-5
Print Loc(#1)
```

190____KOMENDA____Rem__(remark)_____R

Komenda powoduje, że dalsza część linii nie jest wykonywana - stanowi komentarz.

Składnia: Rem text
Skrót   : R text
Skrót   : ' text

```
Do
 Rem TO JEST INSTRUKCJA REM
 Print I'
 Inc I
Loop
```

191____KOMENDA____Repeat__(repeat)_____R

Komenda powoduje otwarcie petli Repeat-Until.
* Until

Składnia: Repeat
Skrót   : Rep

Patrz: Until

192____KOMENDA____Reserve__(reserve)_____R

Komenda przesuwa Himem rezerwując w ten sposób obszar pamięci niedostępny dla Basica.

Składnia: Reserve Y
Skrót : Rese Y
Y: stała, zmienna lub wyrażenie numeryczne określające ilość bajtów pamięci pozostawionych dla Basic'a

```
Print "Himem","Fre"
'
Print Himem,Fre(0)
Reserve Fre(0)-10240
Print Himem,Fre(0)
Reserve Fre(0)+10240
Print Himem,Fre(0)
```

Patrz Exec

193____KOMENDA____Restore__(restore)_____R

Komenda powoduje przewijanie zbioru Data.
* Read, Data

Składnia: Restore [label]
Skrót : Res [label]

```
Read A,B,C,D,E
Restore
Read F,G,H,I
Restore label
Read J,K,L,M
Print A'B'C'D'E''F'G'H'I'J'K'L'M
'
Data 1,2,3
label:
Data 4,5,6,7
```

194____KOMENDA____Resume__(resume)_____R

Komenda wskazuje miejsce powrotu z podprogramu obsługi błędu.

* On Error
Składnia: Resume[Next][label]
Skrót   : Resu [Next][label]
Jeżeli zastosujemy Resume to powrót z podprogramu nastąpi do linii, w której wystąpił błąd, Resume Next - do linii następnej po tej, w której wystąpił błąd, Resume label - do linii następnej po etykiecie (label).

Patrz: On Error

195____KOMENDA____Return__(return)_____R

Komenda powoduje powrót z podprogramu.
* Procedure

Składnia: Return
Skrót   : Ret

Patrz: Gosub

196____FUNKCJA____Right$__(right dollar)_____R

Funkcja wycina z prawej strony danego łańcucha określoną ilość znaków.
* Mid$, Left$

Składnia: Right$(string[,n])
string: stała lub zmienna tekstowa
n: stała, zmienna lub wyrażenie numeryczne określające ilość znaków, które mają zostać wycięte

```
N$="GfA-BASIC"
Print Right$(N$)
Print Right$(N$,5)
Print Right$(N$,12)
```

197____KOMENDA____Rmdir__(remove directory)_____R

Komenda usuwa folder z dysku.
* Mkdir

Składnia: Rmdir foldername
Skrót   : Rm foldername
foldername: stała lub zmienna tekstowa określająca nazwę folderu

```
Mkdir "A:TEST"
Files "A:*.*"
Rmdir "A:TEST"
Files "A:*.*"
```

198____FUNKCJA____Rnd__(random)_____R

Funkcja losowa o wartościach z przedziału 0-1.
* Random

Składnia: Y=Rnd[(X)]

```
For I=1 To 20
 Print Rnd
Next I
```

199____KOMENDA____Rset__(right set)_____R

Komenda wycina z prawej strony zmiennej tekstowej 'var' tyle znaków, ile wynosi długość zmiennej 'set$'.
* Lset

Składnia: Rset set$=var$
Skrót   : Rs set$=var$

```
A$="AAAAAAAA"
B$="Space$(7)
C$="GFA"
Rset A$=C$
Rset B$="GFA-BASIC"
Print A$
Print B$
```

200____KOMENDA____Run__(run)_____R

Komenda uruchamia program.
* Chain, Psave

Składnia: Run

```
Print "TESTUJE KLAWIATURE"
If Upper$(Input$(1))="S"
 End
Else
 Run
Endif
```

201____KOMENDA____Save__(save)_____S

Komenda zapisuje program na dysku.
* Load, Psave

Składnia: Save filename
Skrót   : Sa filename
filename: stała lub zmienna tekstowa określająca nazwę programu.

```
Save "A:\PROG"
Files "A:\*.*"
```

202____KOMENDA____Seek__(seek)_____S

Komenda ustawia głowicę w napędzie dyskowym nad określoną daną.
* Loc, Relseek

Składnia: Seek [#]n,X
Skrót   : See [#]n,X
n,X: stałe, zmienne lub wyrażenia numeryczne określające odpowiednio: numer pliku, położenie głowicy

Patrz: Relseek

203____KOMENDA____Setcolor__(set color)_____S

Komenda definiuje nasycenie składowych R,G,B dla poszczególnych kolorów.

Składnia: Setcolor N,R,G,B lub Setcolor N,X
Skrót   : Se N,R,G,B   lub   Se N,X

N,R,G,B: stałe, zmienne lub wyrażenia numeryczne określające odpowiednio: numer koloru, nasycenie koloru R(0-7), G(0-7), B(0-7)
X=256*R+16*G+B

Występuje tu niekonsekwencja dotycząca numeracji kolorów w komendzie Setcolor i Color (oraz pozostałych komendach graficznych).

NUMERY KOLORÓW DLA INSTRUKCJI COLOR I SETCOLOR

Rozdzielczość średnia		Rozdzielczość niska	
Color	Setcolor	Color	Setcolor
0	0	0	0
1	3	1	15
2	1	2	1
3	2	3	2
		4	5
		5	6
		6	3
		7	5
		8	7
		9	8
		10	9
		11	10
		12	12
		13	13
		14	11
		15	13

```
Circle 300,100,50
Setcolor 0,0,0,0
Setcolor 3,7,7,7
Pause 100
Setcolor 0,7,7,7
Setcolor 3,0,0,0
```

204____KOMENDA____Settime__(set time)_____S

Komenda ustawia aktualny czas i datę.

Składnia: Settime timestring,datestring
Skrót : Sett timestring,datestring

timestring: lancuch o postaci:'gg:mm:ss'
datestring: lancuch o postaci:'dd:mm:rr'

```
 Tt$=Time$
 Dd$=Date$
 Print Time$,Date$
 Pause 200
 Settime "15:30:11","29.6.1986"
 Print Time$,Date$
 Pause 200
 Settime Tt$,Dd$
 Print Time$,Date$
 Pause 200
```

205____KOMENDA____Sget__(screen get)_____S

Komenda powoduje zapamietanie ekranu w postaci zmiennej tekstowej.
* Get, Put, Sput

Składnia: Sget X$
X$: zmienna tekstowa

```
 For I=1 To 5
  Circle 320,100,I*30
 Next I
 '
 Sget A$
 Cls
 Pause 50
 '
 Sput A$
 Pause 300
```

206____FUNKCJA____Sgn__(signum)_____S

Funkcja przyjmuje wartość -1 dla argumentów ujemnych, 0 dla argumentu równego zeru i 1 dla argumentów dodatnich.

Składnia: Y=Sgn(X)
X: stała, zmienna lub wyrażenie numeryczne dowolnego typu

```
 Do
  Input A
```

```
 Print Sgn(A)
Loop
```

207____FUNKCJA____Sin__(sinus)_____S

Funkcja oblicza wartość sinus(x).

Składnia: Y=Sin(X)
X: stała, zmienna lub wyrażenie numeryczne dowolnego typu

```
Plot 320,100
For I=1 To 5400
 X=I/24*Cos(I*Pi/180)
 Y=I/48*Sin(I*Pi/180)
 Draw  To 320+X,100+Y
Next I
```

208____KOMENDA____Sound__(sound)_____S

Komenda programuje generator akustyczny.
* Wave

Składnia: Sound Channel,Volume,Note,Octave[,Time]
Skrót   : So Channel,Volume,Note,Octave[,Time]
Channel-kanał (1-3)
Volume-głosność (0-15)
Note-nuta (1-12)
Octave-oktawa (1-10)
Time-czas po upływie którego wykonana zostanie nastepna instrukcja (w 1/50 sek)

```
For I=1 To 8
 Sound 1,15,1,I,40
 For J=1 To 6
  Read A
  Sound 1,15,A,I,20
 Next J
 Restore
Next I
'
Data 3,5,6,8,10,12,
```

209____FUNKCJA____Space$__(space dollar)_____S

Funkcja tworzy łańcuch spacji zadanej długości.

Składnia: X$=Space$(X)
X:stała,zmienna lub wyrażenie numeryczne określające ilość spacji

```
For I=0 To 20
 Print Space$(I);I
Next I
```

210____KOMENDA____Spc__(space)_____S

Komenda powoduje wydrukowanie zadanej ilości spacji.

Składnia: Print Spc(N)
N:stała,zmienna lub wyrażenie numeryczne określające ilość spacji

```
Print "GfA";Spc(10);"BASIC"
Print "gora";Spc(400);"dol"
```

211____KOMENDA____Spoke__Sdpoke__Slpoke__(supervisor poke)_____S

Ważne zmienne systemowe oraz niektóre rejestry komputera (np. rejestr Video) zostały zabezpieczone przed zmianą ich wartości komendą Poke. Dostęp do nich jest możliwy tylko w trybie Supervisora, poprzez komendy Spoke, Sdpoke i Slpoke.

Składnia i opis parametrów jak dla komend: Poke, Dpoke i Lpoke

```
Box 0,0,639,199
X=Peek(&HFF820A)          !Adres rejestru Video
Spoke &HFF820A,3
Pause 50
Spoke &HFF820A,254
```

212____KOMENDA____Sprite__(sprite)_____S

Komenda wywołuje sprite'a w punkcie o zadanych współrzędnych.

Składnia: Sprite A$[,X,Y]

Skrót : Spr A$[,X,Y]
Sprite jest obiektem o wymiarach 16x16 pikseli i składa się z dwóch rysunków (mogą być w różnych kolorach) nałożonych na siebie. Pominięcie parametrów X,Y powoduje 'schowanie' sprite'a.

A$: łańcuch definiujący postać sprite'a
A$=Mki$(1)+Mki$(Dx)+Mki$(Dy)+Mki$(color 1)+Mki$(color 2)+16*(Mki$(łańcuch 1)+Mki$(łańcuch 2))
Dx,Dy:przesunięcia x i y sprite'a względem punktu o współrzędnych X i Y
łańcuch 1:16 danych 16-bitowych definiujących rysunek w kolorze 1
łańcuch 2:16 danych 16-bitowych definiujących drugi rysunek w kolorze 2

```
Data &x0000000000000000,&x1111111111111111
Data &x0000000001100000,&x1000000000000001
Data &x0000000011000000,&x1000000000000001
Data &x0000000011000000,&x1000000000000001
Data &x0000000010000000,&x1000000000000001
Data &x0000000000000000,&x1000000000000001
Data &x0000111001110000,&x1000000000000001
Data &x0011111111111000,&x1000000000000001
Data &x0011111111111000,&x1000000000000001
Data &x0011111111100000,&x1000000000000001
Data &x0011111111000000,&x1000000000000001
Data &x0011111111100000,&x1000000000000001
Data &x0011111111111000,&x1000000000000001
Data &x0001111111110000,&x1000000000000001
Data &x0000111011100000,&x1000000000000001
Data &x0000000000000000,&x1111111111111111
'
A$=Mki$(1)+Mki$(8)+Mki$(8)
A$=A$+Mki$(2)+Mki$(3)
'
For I=1 To 16
 Read A,B
 A$=A$+Mki$(A)+Mki$(B)
Next I
'
Deffill 1,2,4
Pbox 150,50,400,150
For I=0 To 639 Step 2
 Vsync
```

```
 Sprite A$,I,I/639*199
Next I
'
B$=A$
For I=1 To 140
 Sprite B$,35+I,40+I
 Vsync
Next I
End
```

213____KOMENDA____Sput__(sput)_____S

Komenda powoduje wywołanie ekranu zapamietanego komendą Sget.
* Get, Sget, Put

Składnia: Sput X$[,M]
X$: zmienna tekstowa
M: patrz Put

Patrz: Sget

214____FUNKCJA____Sqr__(square root)_____S

Funkcja oblicza pierwiastek kwadratowy argumentu.

Składnia: Y=Sqr(X)

```
 Input X
 Input Y
 Przekatna=Sqr(X^2+Y^2)
 Print Przekatna
```

215____KOMENDA____Stop__(stop)_____S

Komenda wstrzymuje wykonywanie programu.

Składnia: Stop
Skrót   : St

```
 For I=1 To 10
  Print I
```

```
 If I=5
  Stop
 Endif
Next I
```

216____FUNKCJA____Str$__(string dollar)_____________________S

Funkcja przekształca zmienną numeryczną w tekstową.
* Val

Składnia: Y$=Str$(X)
X: stała lub zmienna numeryczna dowolnego typu

```
A=3.05E+58
B=&022
Print Str$(A)
Print Str$(234)
Print Str$(B)
Print Str$(1/3)
```

217____FUNKCJA____String$__(string dollar)_____________________S

Funkcja tworzy zmienną tekstową składającą się z n powtorzeń zadanego łańcucha.
* Space$

Składnia: String$(n,string) lub String$(n,c)
n,c: stałe, zmienne lub wyrażenia numeryczne określające odpowiednio: ilość powtorzeń, kod ASCII
string: dowolny lancuch

```
Z$="A"
Print String$(50,Z$)
Print String$(50,"A")
Print String$(50,65)
Print String$(25,"AA")
```

218____KOMENDA____Sub__(substraction)_____S

Komenda odejmuje od zmiennej 'var' wartość 'n'.
* Dec

Składnia: Sub var,n
Skrót   : S var,n
var: zmienna numeryczna dowolnego typu
n: stała, zmienna lub wyrażenie numeryczne dowolnego typu

```
T=Timer
For I%=1 To 10000
 Sub A%,5
Next I%
Print (Timer-T)/200
A%=0
T=Timer
For I%=1 To 10000
 A%=A%-5
Next I%
Print (Timer-T)/200
```

219____KOMENDA____Swap__(swap)_____S

Komenda powoduje zamianę wartości dwóch zmiennych.

Składnia: Swap var1,var2
Skrót   : Sw var1,var2
var1,var2: zmienne dowolnego typu

```
Dim A(3),B(8)
Arrayfill B(),8
'
A$="lewy"
B$="prawy"
'
Print A$,B$
Swap A$,B$
Swap A(),B()
Print A$,B$
Print A(8),B(2)
```

220____KOMENDA____System__(system)_____S

Komenda powoduje powrót do TOS-u.
* Quit

Składnia: System
Skrót   : Sy

```
Alert 3,"Czy napewno chcesz opuscic program",1," OK !CANCEL",J
If J=1
 System
Endif
```

221____FUNKCJA____Tan__(tangent)_____T

Funkcja oblicza wartość tangens(x).
* Atn

Składnia: Y=tan(X)
X: stała, zmienna lub wyrażenie numeryczne dowolnego typu

```
Input Rad
Print Tan(Rad)
Input Grad
Print Tan(Grad*Pi/180)
```

222____KOMENDA____Text__(text)_____T

Komenda wysyła tekst w określone miejsce ekranu.
* Deftext

Składnia: Text X,Y,[M,]string
Skrót   : T X,Y,[M,]string
X,Y,M: stałe, zmienne lub wyrażenia numeryczne określające odpowiednio: współrzędne X i Y początku wydruku oraz odległość miedzy pierwszą i ostatnią literą

```
Deftext 1,16,0,21
A$="ABC D"
Text 10,50,A$
Text 10,180,200,A$
```

```
Deftext 1,0,0,6
Do
 Input I
 Cls
 Text 300,160,I,Str$(I)+"ABC D"
Loop
```

223____FUNKCJA____Time$__(time dollar)_____T

Funkcja podaje aktualny czas.
* Settime, Date$

Składnia: X$=Time$

```
Deftext 1,16,0,32
Do
 Print At(1,1);Time$
 Text 400,40,Time$
 Text 240,180,Time$
Loop
```

224____FUNKCJA____Timer__(timer)_____T

Funkcja podaje stan licznika, którego zawartość zwiększa sie o 1 co 1/200 sekundy.

Składnia: Y=Timer

```
T=Timer
Repeat
 Z=Int((Timer-T)/2)/100
 Print At(38,3);Z
Until Inkey$()""
```

225____KOMENDA____Titlew__(title of window)_____

Komenda nadaje tytuł zadanemu oknu.
* Openw, Infow

Składnia: Titlew n,"titel"
Skrót : Tit n,"titel"

n: stała, zmienna lub wyrażenie numeryczne określające numer okna
titel: stała lub zmienna tekstowa określająca tytuł

```
Openw 1,100,50
Pause 100
Titlew 1,"Window 1"
Closew 1
Openw 2,200,100
Pause 100
Titlew 2,"Window 2"
Closew 2
Titlew 3,"   3   "
Titlew 4,"   4   "
Openw 3,110,150
Pause 100
Closew 3
```

226____KOMENDA____Tron__(trace on)_____T

Komenda włącza śledzenie programu.
* Tron #, Troff

Składnia: Tron
Skrót   : Tr

```
Tron
Do
Loop
```

227____KOMENDA____Tron#__(trace on number)_____T

Komenda włącza śledzenie programu i zapisuje treść śledzenia w pliku o zadanym numerze.
* Tron, Troff

Składnia: Tron #n
Skrót   : Tr [#]n
n:stała, zmienna lub wyrażenie numeryczne określające numer pliku

```
Open "R",#1,"TRON"
Tron #1
For I = 1 To 10
```

```
 Print I;
Next I
Troff
```

228____KOMENDA____Troff__(trace off)_____T

Komenda powoduje wyłaczenie śledzenia programu.
* Tron, Tron#

Składnia: Troff
Skrót   : Trof

Patrz: Tron#

229____KOMENDA____Until__(until)_____U

Dopóki nie zostanie spełniony zadany warunek komenda przekazuje sterowanie do linii Repeat.
* Repeat

Składnia: Until condition
Skrót   : U condition
condition: porównanie dwóch stałych lub zmiennych dowolnego typu

```
Repeat
 A=A+1
 Print A
Until A=20
```

230____FUNKCJA____Upper$__(upper dollar)_____U

Funkcja wymienia w zadanym łańcuchu małe litery na duże.

Składnia: X$=Upper$(string)
string: stała lub zmienna tekstowa

```
A$="basic"
Print Upper$(A$)
Print Upper$("1a")
Print Upper$("GfA")
```

231____FUNKCJA____Val__(value)_____V

Funkcja oblicza wartość łańcucha.
 * Str$

Składnia: X=Val(X$)
X$: stała lub zmienna tekstowa

```
A$="00950 Warszawa 5"
Print Val(A$)
Print Val("1.5e+05")
Print Val("GfA")
Print Val("&HFF")
```

232____FUNKCJA____Val?__(value)_____V

Funkcja podaje długość łancucha dającego wartość.

Składnia: X=Val?(X$)
X$: stała lub zmienna tekstowa

```
A$="22 DM"
Print Val?(A$)
Print Val?("1.0e+19")
Print Val?("1e19")"
Print Val?("&X1010011")
Print Val?("AB123")
```

233____FUNKCJA____Varptr__(variable pointer)_____V

Funkcja podaje adres komórki począwszy od której przechowywana jest zadana zmienna.

Składnia: X=Varptr(var)
var: stala lub zmienna dowolnego typu

```
A%=17
N$="GfA"
'
Print Varptr(A%)
Print Lpeek(Varptr(A%))
Print Peek(Varptr(N$))
```

```
Print Peek(Varptr(N$)+1)
Print Peek(Varptr(N$)+2)
```

234____KOMENDA____Vdisys__(vdi system)_____V

Komenda wywołuje zadaną procedurę GEM-u z biblioteki VDI.

Składnia: Vdisys

Wywołamy funkcję definiującą linię użytkownika (patrz: Defline)

```
Defline 7
Dpoke Contrl,113                  ! kod operacyjny funkcji
Dpoke Contrl+2,0
Dpoke Contrl+4,0
Dpoke Contrl+6,1
Dpoke Contrl+8,0
Dpoke Contrl+12,2
Dpoke Intin,&x1010101010101010    ! 16-to bitowe słowo stanowiące
'                                   wzór użytkownika
Vdisys
'
Box 100,100,200,150
```

235____Void__(void)_____V

Komenda pozwala wywołać funkcję bez wyprowadzenia jej wartości.

Składnia: Void [function]

Patrz: Gemdos

236____KOMENDA____Vsync__(vertical synchronization)_____V

Komenda powoduje wstrzymanie wykonywania programu do chwili pojawienia się impulsu synchronizacji pionowej.

Składnia: Vsync
Skrót : Vs

Patrz: Sprite

237____KOMENDA____Wave__(wave)_____W

Komenda ustawia rejestry generatora akustycznego.

Składnia: Wave channel,shape channel,shape,modulation,time
Skrót : Wa channel,shape channel,shape,modulation,time

Channel (kanał)					
szum			dźwięk		
C	B	A	C	B	A
32	16	8	4	2	1

Shape Channel (kanał regulowany)		
C	B	A
4	2	1

modulation: częstotliwość obwiedni (0-65535)
time: jak w komendzie Sound

Shape (obwiednia)	
_	0, 1, 2, 3, 9,
/_	4, 5, 6, 7, 15
\\\\\\\\\\	8
\/\/\/\/\/\	10
\¯¯¯¯¯¯¯¯	11
////////////	12
/¯¯¯¯¯¯¯¯¯¯	13
/\/\/\/\/\/\	14

```
Print "*"
Sound 1,15,1,4,20
Print "**"
Sound 2,16,4,4,20
Print "***"
Sound 3,15,8,4,20
Print "****"
Wave 7,7,0,65535,300
Print "*****"
Wave 0,0
```

238____KOMENDA____Wend__(while wend)_____W

Komenda kończy petle While-Wend.
* While

Składnia: Wend

Patrz: While

239____KOMENDA____While__(while)_____W

Komenda otwiera petle While-Wend. Petla jest wykonywana dopóki jest spełniony warunek w komendzie While.
* Wend

Składnia: While condition
Skrót : W condition
condition: porównanie stałych lub zmiennych dowolnego typu

```
While A<10
 A=A+1
 Print A
Wend
```

240____FUNKCJA____Windtab____(window table)_____W

Funkcja podaje adres tablicy opisujacej okna.
* Openw

Składnia: Y=Windtab

STRUKTURA TABLICY WINDTAB

Dane dla okna numer 1.

Windtab - handle-numer identyfikacyjny okna (dla potrzeb GEM-u patrz:Menu pkt.2.,On Menu Message,Procedure Slider), inny niż numer okna w GFA-BASICU;
Windtab+2 - składniki okna
Windtab+4 - współrzędna X lewej krawędzi okna
Windtab+6 - współrzędna Y górnej krawędzi okna
Windtab+8 - szerokość okna
Windtab+10 - wysokość okna

Dane dla okna numer 2

Windtab+12 - jak Windtab
Windtab+14 - jak Windtab+2
Windtab+16 - jak Windtab+4
Windtab+18 - jak Windtab+6
Windtab+20 - jak Windtab+8
Windtab+22 - jak Windtab+10

Dane dla następnych okien

Składniki okna określamy ustawieniem bitów. Znaczenie bitów jest następujące:

Bit 0	linia tytułowa okna
Bit 1	pole zamykające okno
Bit 2	pole powiększające okno do pełnych wymiarów
Bit 3	pole przesuwające okno
Bit 4	linia informacyjna okna
Bit 5	pole zmieniające wymiary okna
Bit 6	pole za strzałką w górę
Bit 7	pole ze strzałką w dół
Bit 8	slider pionowy
Bit 9	pole ze strzałką w lewo
Bit 10	pole ze strzałką w prawo
Bit 11	slider poziomy

241____KOMENDA____Write #__(write number)_____________________W

Komenda zapisuje stałe i zmienne dowolnego typu na dysku.

Składnia: Write #n,var[,var]
Skrót   : Wr #n,var[,var]
n: stała,zmienna lub wyrażenie numeryczne określające numer pliku
var: stałe lub zmienne dowolnego typu

```
Open "O",#1,"dat"
Write #1,"Karol","May",18
Close
Open "I",#1,"dat"
Input #1,Imie$,Nazwisko$,Wiek
Print Imie$'Nazwisko$'"lat"'Wiek
Close
```

242____FUNKCJA____Xbios__(extended basic input output system)___X

Funkcja wywołuje zadaną procedurę Xbios-u.

Składnia: Y=Xbios(opc, list of parameters)
opc:stała, zmienna lub dowolne wyrażenie określające numer procedury
list of parameters: lista parametrów

WYKAZ WAZNIEJSZYCH FUNKCJI XBIOS-u

Opc(hex)	Nazwa	Opis funkcji
02	physbase()	funkcja podaje adres pamięci obrazu (fizyczny)
03	logbase()	funkcja podaje adres pamięci obrazu (logiczny)
04	getres()	funkcja podaje aktualną rozdzielczość: 0-niska,1-średnia,2-wysoka
05	setscreen (a,b:LI;c:i)	ustawia parametry obrazu: a-adres logiczny, b-fizyczny, c-rozdzielczość
07	setcolor (a,b:I)	ustawia nasycenie (b) koloru o numerze a; dla b<0 podaje aktualne nasycenie
10	keytbl (a..c:LI;)	definiuje klawiaturę, a..c-128 bajtowe łańcuchy zawierające kody wysyłane przez poszczególne klawisze; a-bez SHIFT-u, b-z SHIFT-em, c-z CAPS LOCK
11	random()	podaje liczbę pseudolosową
23	kbrate (a,b:LI;)	definiuje opóźnienie (a) i czas repetycji (b) klawiatury

5. WYKAZ BŁĘDÓW

Błędy edycji:

- Editor-Internal Error. Błąd wewnętrzny edytora. Może wystąpić po użyciu instrukcji Poke w obszarze pamięci zajmowanym przez edytor. Należy wczytać ponownie edytor.
- While without Wend. Użyto While bez Wend.
- Repeat without Until. Użyto Repeat bez Until.
- Do without Loop. Użyto Do bez Loop.
- For without Next. Użyto For bez Next.
- Wend without While. Użyto Wend bez While.
- Until without Repeat. Użyto without bez Repeat.
- Loop without Do. Użyto Loop bez Do.
- Next without For. Użyto Next bez For.
- If without Endif. Użyto If bez Endif.
- Endif without If. Użyto Endif bez If.
- Else without If. Użyto Else bez If.
- Else without Endif. Użyto Else bez Endif.
- Exit without a loop. Użyto Exit poza petlą.
- Procedure without Return. Procedura nie ma Return.
- Procedure defined twice. Użyto dwukrotnie tej samej nazwy procedury
- Return without Procedure. Użyto Return bez Procedury.
- Label defined twice. Użyto dwukrotnie tej samej etykiety.
- Local only allowed in Procedure. Zmienne lokalne (Local) można definiować tylko wewnątrz Procedury.
- Local not allowed in a Loop. Zmiennych lokalnych nie można definiować wewnątrz petli.
- Function defined twice. Użyto dwukrotnie tej samej nazwy funkcji.
- Goto into/out of a For Next Loop or a Procedure. Użyto skoku do lub z petli For Next lub Procedury.
- Resume into a For Next Loop. Użyto Resume wewnątrz petli For.
- Resume without Procedure. Użyto Resume na zewnątrz Procedury.
- Syntax error. Błąd składni.
- Line too long. Linia za długa (max. 252 znaki).

Błędy wykonania:

Błędy systemu:

-1. * General error - Błąd ogólny.
-2. * Drive not ready - Napęd nie jest gotowy.
-3. * Unknown command - Nieznana komenda.
-4. * CRC error Disk check sum wrong - Zła suma kontrolna. Błąd zapisu lub czytania wykryty przez kontroler dysku.
-5. * Bad request - Złe polecenie.
-6. * Seek error. Track not found - Ścieżka nie znaleziona.
-7. * Unknown media.Boot sector wrong-Błąd sektora informacyjnego dysku.
-8. * Sector not found - Sektor nie znaleziony.
-9. * Out of paper - Brak papieru.
-10. * Write fault - Błąd zapisu.
-11. * Read fault - Błąd czytania.
-12. * General error 12 - Błąd ogólny 12.
-13. * Write protected - Zabezpieczony przed zapisem.
-14. * Media change detected - Wykryto zmiany nośnika.
-15. * Unknown device - Nieznane urządzenie (zewnętrzne).
-16. * Bad sector (verify) - Zły sektor.
-17. * Insert other disk (request) - Włóż jeszcze raz dysk.
-32. * Invalid function number - Zły numer funkcji. Błąd związany jest z procedurą &h42 GEMDOS-u odpowiedzialną za wykonywanie komend Seek i Relseek. Jednym z parametrów wejściowych tej procedury jest numer funkcji: 0-ustawia wskaźnik danych n bajtów od początku pliku, 1-względem położenia bieżącego, 2-od końca pliku.
-33. * File not found - Pliku nie znaleziony.
-34. * Path not found - Ścieżka (Path) nie znaleziona.
-35. * To many open files - Za dużo otwartych plików.
-36. * Acces denied - Odmowa dostępu. Występuje podczas otwierania nowego pliku, gdy katalog dysku jest zapełniony lub plik o zadanej nazwie już istnieje; podczas wykonywania komendy Name, gdy plik ma atrybut Read Only (tylko do czytania).
-37. * Invalid handle - Zły wskaźnik pliku. Występuje, między innymi, gdy wymienimy dysk z otwartym plikiem, po czym będziemy usiłowali z niego korzystać lub go zamknąć.
-39. * Out of memory - Brak pamięci. Występuje podczas ładowania programu, który nie mieści się w pamięci.

-40. * Invalid memory block adress - Zły adres bloku pamięci. Błąd związany jest z procedurami &h48,&H49 i &H4A GEMDOS-u odpowiedzialnymi za przydziały pamięci.
-46. * Invalid drive specification - Złe oznaczenie napędu. Może wystąpić, gdy żądamy danych (np. wolnej pamięci) z nieistniejącego napędu.
-49. * No more files - Nie ma więcej plików.
-64. * GEMDOS range error, Seek wrong? - Błędny zakres parametru komendy Seek
-65. * GEMDOS internal error - Błąd wewnętrzny Gemdos-u. Może być spowodowany brakiem komendy Close po komendach Bsave lub Bload (występuje po skompilowaniu programu)
-66. * Invalid executable file format - Zły format wykonywanego pliku. Może wystąpić podczas wykonywania komendy Exec.
-67. * Growth restriction failure. Błąd związany z procedurą GEMDOS-u Mshrink (&H4A) zmieniającą obszar przydzielonej pamięci; występuje, gdy zadana wielkość obszaru jest większa od obszaru dotychczasowego.

Błędy interpretera GFA-BASIC'a:

0. Division by zero - Dzielenie przez zero.
1. Overflow - Przepełnienie. Oprócz sytuacji 'normalnych' dla tego błędu tj. dzielenia przez liczby bliskie zeru, błąd ten występuje także w programach skompilowanych podczas podnoszenia liczb bliskich zeru do kwadratu.
2. Number not integer - Liczba nie jest całkowita.
3. Number not byte - Liczba nie jest bajtem.
4. Number not word - Liczba nie jest słowem (16-bitowym).
5. Square root only for positive numbers - Pierwiastek kwadratowy tylko dla liczby dodatniej.
6. Logarithm only for numbers greather then zero - Logarytm tylko dla liczb większych od zera.
7. Undefined error - Błąd niezdefiniowany.
8. Out of memory - Brak pamięci.
9. Function or command not yet implemented - Funkcja lub rozkaz jeszcze nie zaimplementowana.
10. String to loong max. 32767 charakters - Łańcuch za długi max. 32767 znaków.
11. Not GFA BASIC V1.0 program - Program nie jest napisany w GFA-BASIC-u.
12. Program to long. Memory full - Program za długi.Brak pamięci.

13.Not GFA BASIC program. File to short - Program nie jest napisany w GFA BASIC-u. Plik za krótki.
14.Array dimensioned twice - Tablica zadeklarowana powtórnie.
15.Array not dimensioned - Tablica nie zadeklarowana.
16.Array index to large - Wskaźnik tablicy za duży.
17.Dim index to large - Wymiar tablicy za duży.
18.Wrong number of indices. - Zła ilość wskaźników.
19.Procedure not found - Procedura nie znaleziona.
20.Label not found - Etykieta nie znaleziona.
21.On open only "I"nput "O"utput "R"andom "A"ppend "U"pdate allowed - Można otwierać tylko pliki I,O,R,A,U.
22.File already open - Plik już otwarty.
23.File # wrong - Zły numer pliku.
24.File not open - Plik nie otwarty.
25.Input wrong not numeric - Wprowadzona dana nie jest numeryczna.
26.End of file reached - Koniec zasobów pliku.
27.Too many points for Polyline/Polyfill/Polymark max. 128 - Za dużo punktów w instrukcjach Polyline/fill/mark (max.128).
28.Array must have one dimension-Tablica musi mieć jeden wymiar.
29.Number of points to large for array - Ilość punktów za duża dla tablicy.
30.Merge - not an ASCII file - Łączenie tylko dla plików ASCII.
31.Merge - line to loong - Łączenie - linia za długa.
32.==> syntax error program aborted - Syntax error.
33.Undefined label - Niezdefiniowana etykieta.
34.Out of data - Brak danych.
35.Data not numeric - Dana nie jest numeryczna.
36.Syntax error unpaired quotes - Syntax error, niesparowany cudzysłów.
37.Disk full - Dysk jest pełny.
38.Command not allowed in direct mode - Komenda jest niedostępna w trybie bezpośrednim.
39.Program error. Gosub not possible - Błąd programu. Gosub nie jest możliwe.
40.Clear not allowes in For-Next_Loops or Procedures - Clear nie jest dostepne wewnatrz For-Next lub wewnatrz Procedury.
41.Cont not possible - Cont nie jest możliwe.
42.Parameter missing - Opuszczony parametr.
43.Expression to complex - Wyrażenie zbyt złożone.
44.Undefined function - Niezdefiniowana funkcja.
45.Too many parameters - Zbyt wiele parametrów.
46.Parameter wrong must be a number - Zły parametr, musi być liczbą.

47.Parameter wrong must be a string - Zły parametr,musi być łańcuchem.
48.Open "R" record lenght wrong - Zła długość rekordu.
49.Undefined error 049 - Błąd niezdefiniowany.
50.Not an "R" file - Plik nie jest typu "R".
51.Only one Field per Open "R" allowed - Dostępne jest tylko jedno pole typu "R".
52.Fields larger then records lenght - Długość pola jest większa od długości rekordu.
53.Too many fields max. 19 - Zbyt wiele pól max.19.
54.GET/PUT Field string lenght changed -Długość pola w komendach GET/PUT uległa zmianie.
55.GET/PUT record number wrong - Ilość rekordów w komendach GET/PUT uległa zmianie.
60.Sprite. String lenght wrong - Zła długość łańcucha Sprite'a.
61.RESERVE error - Błąd RESERVE.
62.MENU error - Błąd MENU
63.RESERVE error - Błąd RESERVE.
64.Pointer(*x) error - Błąd wskaźnika *.
90.LOCAL error - Błąd LOCAL.
91.FOR error - Błąd pętli For.
92.Resume (Next) not possible Fatal, For oder Local - Resume (Next) nie jest mozliwe.
102.2 bombs-bus error. Peek or Poke possible wrong - Prawdopodobnie zły Peek lub Poke (patrz komenda Spoke).
103.3 bombs-adress error. Odd word adress! Possibly at Dpoke, Dpeek, Lpoke, Lpeek.- Błąd adresu. Adres jest liczbą nieparzystą. (Powinien być parzysty). Możliwy błąd instrukcji Dpoke, Dpeek, Lpoke, Lpeek.
104.4 bombs-illegal instruction executed in machine code. Nielegalna instrukcja wykonana w kodzie maszynowym.
105.5 bombs-divide by zero in 68000 Machine Code. Dzielenie przez zero w kodzie maszynowym

Błedy kompilacji:

- Can't compile. Check with Interpreter. - Kompilacja niemożliwa. Sprawdź z interpreterem.
- Compile error. No GFA-Basic File. - Program nie jest napisany w GFA-Basic'u.
- Compile error. Syntax Error in Programs. - Błąd syntaktyczny w programie.

- Compile error. Read Error. - Błąd czytania.
- Compile error. Write Error. - Błąd zapisu.
- Compile error. SAVEPROG-write error.- Błąd zapisu (pełny dysk).

Program kompilowany nie powinien zawierać następujących instrukcji:

- Cont
- Deflist
- List
- Llist
- Load
- Psave
- Save
- Stop
- Tron
- Troff

Wystąpienie jednej z powyższych komend powoduje wygenerowanie komunikatu: (np. dla komendy Stop)

Compile error. STOP??. Continue!Abort

Jeśli wybierzemy Continue, program będzie nadal kompilowany a wskazana komenda będzie ignorowana.

6. OPIS KOMPILATORA

Kompilator GFA-BASIC'a jest kompilatorem dwuprzebiegowym. Kompilacja przyspiesza w istotny sposób te programy, w których użyto wielu zmiennych typu INTEGER, w przeciwieństwie do programów używających zmienne rzeczywiste.
Poniższe zestawienie zawiera czasy wykonywania programu zapełniającego ekran komendą Poke (na podstawie ST COMPUTER 2/87).

Petla For...Next

	Interpreter	Kompilator
Integer:	10,96 sek	3,55 sek
Real :	13,27 sek	4,33 sek

Petla Repeat...Until

	Interpreter	Kompilator
Integer:	21,51 sek	3,38 sek
Real :	23,02 sek	6,03 sek

Długość programu skompilowanego jest wieksza od źródłowego, różnica ta zmniejsza sie w miarę wzrostu długości programu źródłowego. Przykładowe długości programów źródłowych i skompilowanych podano poniżej (ST COMPUTER 2/87).

Interpreter	Kompilator
264	4339
1370	8747
56409	86740

Z doświadczenia autora wynika, że nie zawsze krótszy program źródłowy daje krótszy kod wynikowy. Dwa poniższe programy dają ten sam efekt, a zestawienie ich długości wygląda następująco:

	Interpreter	Kompilator
Program 1	236	3776
Program 2	244	3160

```
'Program 1
Text 1,10,"Linia 1"
Text 1,20,"Linia 2"
Text 1,30,"Linia 3"
Text 1,40,"Linia 4"
Pause 100
'

'Program 2
For IX=1 To 4
  Read A$
  Text 1,10*iX,A$
Next IX
Pause 100
Data Linia 1,Linia 2,Linia 3,Linia 4
```

Czas kompilowania zależy również od rodzaju napędu dyskowego, dla przykładu podano czasy kompilacji programu o długości 56 kB.

Floppy	Ram-Disk	Hard-Disk
95 sek	40 sek	42 sek

Obsługa kompilatora.

Po załadowaniu kompilatora na ekranie pojawia się panel sterujący z czterema przełącznikami:

Stop:

- Ever - powoduje, że skompilowany program może być zatrzymany klawiszami: CTRL + Shift + Alt w dowolnym momencie,
- Loop - program może być zatrzymany jak wyżej,ale tylko wewnątrz pętli,
- Never- program nie może być zatrzymany.

Trapv:

- '+' - załącza śledzenie skompilowanego programu,
- '-' - wyłącza śledzenie.

Errors:

- Text - powoduje wyświetlanie standardowych komunikatów błędów,
- No - powoduje wyświetlanie błędów w postaci Error #...

Bombs (działa dla błędów powyżej 100):

- '+' - powoduje wyświetlenie komunikatu o błędzie,
- '-' - powoduje wyświetlanie 'bomb' zamiast komunikatu.

Po ustawieniu żądanych pozycji przełączników wybieramy opcję 'Compile', co spowoduje wywołanie katalogu dysku zawierającego programy z rozszerzeniem .BAS (możliwa jest kompilacja tylko programów zapisanych komendą Save). Wybranie programu rozpoczyna proces kompilacji, który przebiega do chwili ponownego wyświetlenia katalogu dysku. W tej fazie możliwa jest wymiana dysku oraz zmiana nazwy programu. Zapisanie programu skompilowanego kończy proces kompilacji i program jest gotowy do natychmiastowego uruchomienia z poziomu TOS-u.

Kompilator opuszczamy wybierając opcję 'Abort'.

7. ZESTAW ZNAKOW DOSTEPNYCH W GFA-BASIC'u.

HEX	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		^A	^B	^C	^D	^E	^F	^G	^H	^I	^J	^K	^L	^M	^N	^O
1	^P	^Q	^R	^S	^T	^U	^V	^W	^X	^Y	^Z	^[	^\	^]	^^	^_
2	SPACE	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	del
8	!^	!^A	At-	At=	!^D	!^E	!^F	!^G	!^H	!^I	!^J	!^K	!^L	!^M	!^N	!^O
9	At@	AtW	AtE	AtR	AtT	AtY	AtU	AtI	At0	AtP	!^Z	!^[	£	!^]	At^	At_
A	At0	AtF	At6	AtH	AtJ	AtK	AtL	!'	!(	!)	!=	!+	AtZ	AtX	AtC	AtV
B	AtB	AtN	AtM	!3	!4	!5	!6	!7	!8	!9	!:	!;	!<	!=	!>	!?
C	!@	!S	!B	!C	!D	!E	!F	!G	!H	!I	!J	!K	!L	!M	!N	!O
D	!P	!Q	!R	!S	!T	!U	!V	!W	!X	!Y	!Z	![	!\	!]	!^	!_
E	!`	!a	!b	!c	!d	!e	!f	!g	!h	!i	!j	!k	!l	!m	!n	!o
F	!p	!q	!r	!s	!t	!u	!v	!w	At1	At2	At3	At4	At5	At6	At7	At8

^ - oznacza Control At - oznacza Alternate

! - oznacza przejście do drugiego zestawu znaków poprzez Ctrl + S

8. PRZYKŁADOWE PROCEDURY I PROGRAMY

8.1. Procedura Chdrive sprawdza mape dostepnych napedów i pozwala uaktywnić jeden z nich.

```
@Chdrive
Print "Aktywnym napedem jest naped:";Chr$(Cdr%+65)
End
'
Procedure Chdrive
  A$=Right$("0000000000000000"+Bin$(Gemdos(&HE,1)),16)
  Do
    Alert 2,"|Current drive:",2,"      | "+Chr$(Cdr%+65)+" |      "
                                                          ,J%
    Exit If J%=2
    If J%=1
      Do
        Dec Cdr%
        If Cdr%<0
          Cdr%=15
        Endif
        Exit If Mid$(A$,16-Cdr%,1)="1"
      Loop
    Endif
    If J%=3
      Do
        Inc Cdr%
        If Cdr%>15
          Cdr%=0
        Endif
        Exit If Mid$(A$,16-Cdr%,1)="1"
      Loop
    Endif
  Loop
  Chdrive Cdr%+1
Return
```

8.2. Procedura Invers ustala sposób wyprowadzania znaków.

dla X=112 - znaki są zanegowane
dla X=113 - znaki są normalne

```
@Invers(112)
Print At(30,10);"AAAAAAAAAAA"
@Invers(113)
Print At(30,12);"AAAAAAAAAAA"
Do
Loop
'
Procedure Invers(X)
  Open "O",#1,"CON:"
  Print #1,Chr$(27);Chr$(X);
  Close #1
Return
```

8.3. Procedura obsługująca joysticki.

Peek(W1%) podaje następujące wartości w zależności od położenia joysticka:

5 1 9

4 8

6 2 10

Przycisk strzału zwiększa bieżącą wartość o 128.

```
Hidem
On Break Gosub Break
Tab%=Xbios(34)                    !adres tablicy zawierajacej dane
'                                  wyprowadzane z klawiatury
'
W1%=Tab%+60                       !adres danych joysticka 1
W2%=Tab%+61                       !adres danych joysticka 2
'
A$=Chr$(&H14)
B$=Chr$(&H15)
C$=Chr$(&H8)
'
```

```
X=Xbios(&H19,3,L:Varptr(A$))     !włącza joysticki
'
Do
  @P
Loop
'
Procedure P
  Print Peek(W1%),Peek(W2%)
Return
'
Procedure Break
  X=Xbios(&H19,3,L:Varptr(B$))   !wyłącza joysticki
  X=Xbios(&H19,3,L:Varptr(C$))   !włącza mysz
  Edit
Return
```

8.4. Program dokonuje konwersji obrazów między standardami:

DEGAS <> NEOCHROME <> DOODLE

```
Dim Col%(16)                     ! tablica kolorow
Screen$=Space$(32128)            ! rezerwacja obszaru dla obrazu
Res%=Xbios(4)                    ! rozdzielczosc
'
If Res%=2
  Xmax=639
  Ymax=399
  Res1%=1
Endif
'
If Res%=1
  Xmax=639
  Ymax=199
  Res1%=2
Endif
'
If Res%=0
  Xmax=319
  Ymax=199
  Res1%=4
Endif
```

```
Do
  @Czytaj
  Repeat
  Until Inkey$=" " Or Mousek
  Alert 2,"| Czy chcesz zapisac ten obraz ",2,"Tak|Nie",J%
  If J%=1
    @Przygotuj
  Endif
Loop
'
Procedure Czytaj
  Do
    Alert 2,"| Wybierz format wyjsciowy  ",1,
                                        " DEGAS | NEO | normal ",J%
    '
    Offset=0
    Extention$="DOO"
    '
    If J%=1 Then
      Extention$="PI"+Chr$(49+Res%)        ! Rozszerzenie
      Offset=34
    Else
      If J%=2
        Extention$="NEO"
        Offset=128
      Endif
    Endif
    W$="\*."+Extention$
    Fileselect W$,"",Name$
    If Exist(Name$)
      Print At(1,1);"Laduje ";Name$;" do bufora obrazu"
      Bload Name$,Varptr(Screen$)
      Close
      If Res%<2 Then
        If J%=1 Then
          @Czytaj_kolor(2)
        Endif
        If J%=2 Then
          @Czytaj_kolor(4)
        Endif
      Endif
      @Pokaz
      Wyj%=2
    Else
```

```
      Alert 3,"| Pliku nie znaleziono ",1," OK ",J%
      Alert 2,"| Czy chcesz opuscic program ",2," TAK : NIE ",Wyj%
      If Wyj%=1
        Stop
      Endif
    Endif
    Exit If Exist(Name$)
  Loop
Return
'
Procedure Czytaj_kolor(Offset)
  For Farba=0 To 15
    Col%(Farba)=Dpeek(Varptr(Screen$)+(Farba*2+Offset))
    Setcolor Farba,Col%(Farba)
  Next Farba
Return
'
Procedure Zapisz_color
  Farb$=""
  For Farba=0 To 15
    Col%(Farba)=Xbios(7,Farba,-1)
    Farb$=Farb$+Mki$(Col%(Farba))
  Next Farba
Return
'
Procedure Pokaz
  Screenp$=Mid$(Screen$,Offset+1,32000)
  Screen$=Mki$(Xmax)+Mki$(Ymax)+Mki$(Res1%)+Screenp$
  Put 0,0,Screen$
Return
'
Procedure Przygotuj
  Alert 2,"|   Wybierz format   ",3,"DEGAS|NEO|normal",J%
  If J%=1
    Extention$="PI"+Str$(Res%+1)
    Len=32034
    @Zapisz_color
    Screen_new$=Mki$(Xbios(4))+Farb$+Screenp$
    @Zapisz
  Endif
  If J%=2
    Extention$="NEO"
    Len=32128
    @Zapisz_color
```

```
    Screen_new$=Mki$(0)+Mki$(0)+Farb$+Space$(92)+Screenp$
    @Zapisz
  Endif
  If J%=3
    Extention$="DOO"
    Len=32000
    Screen_new$=Screenp$
    @Zapisz
  Endif
Return
'
Procedure Zapisz
  Oldname$=Right$(Name$,Len(Name$)-1)
  Oldname$=Left$(Oldname$,Len(Oldname$)-3)+Extention$
  Extention$="\*."+Extention$
  Fileselect Extention$,Oldname$,Newname$
  If Newname$<>"" Then
    Bsave Newname$,Varptr(Screen_new$),Len
  Endif
Return
```

8.5. Edytor fontów. Program pozwala przygotować zestaw dowolnych znaków, które można użyć w innych programach, np. procesorze tekstów. Program pracuje w średniej rozdzielczości.

```
Setcolor 0,0,0,0
Setcolor 3,7,7,7
Copy%=-1
Yt%=8
Xt%=6
Font$=Space$(4096)
F1$=Space$(4096)
'
Deffill 2,1,1
Defline 0,0,0,0
Deftext 1,0,0,6
Box 0,0,639,199
For I%=1 To 31
  Line 20*I%,0,20*I%,88
Next I%
For I%=1 To 8
  Line 0,I%*11,639,I%*11
Next I%
```

```
@Sysfn
For I%=0 To 8
  Line 20+20*I%,95,20+20*I%,183
  Line 20,95+11*I%,180,95+11*I%
Next I%
Box 200,95,620,183
Text 248,112,"            NEGAT         LOAD          COPY"
Text 250,132," "
Text 334,132,"CLEAR          SAVE"
Text 250,152," "
Text 334,152," SYM.          SYS.FN"
Text 250,172,Chr$(1)
Text 334,172,"ROTATE         QUIT"
Text 530,134,"HEX:"
Text 530,150,"DEC:"
Text 530,166,"VIEW:"
Text 20,194,"FONT  EDITOR  FOR  WORDPLUS   PROCESSOR      v 1.0
                                        1988     c by S. NAWROCKI"
'
For I%=0 To 3
  For J%=0 To 3
    If I%=3 And J%>0
    Else
      Box 220+102*I%,102+J%*20,290+102*I%,116+J%*20
    Endif
  Next J%
Next I%
'
Do
  Mouse X%,Y%,K%
  If Y%<88
    @Wybor
  Else
    If Asc%<>Ascw%
      Print At(72,17);Hex$(Ascw%);" "
      Print At(72,19);Str$(Ascw%);"  "
      Asc%=Ascw%
    Endif
  Endif
  If X%>21 And X%<178 And Y%>96 And Y%<180
    @Pixel
  Endif
  If X%>425 And X%<493
    @Ster
```

```
  Endif
  If X%)322 And X%(391
    @Ster1
  Endif
  If X%)221 And X%(289
    @Ster2
  Endif
  If X%)527 And X%(597 And Y%)103 And Y%(116 And K%
    @Copy
  Endif
Loop
'
Procedure Wybor
  Asc%=32*Int(Mousey/11)+Int(Mousex/(20))
  If Asc%()Ascs%
    Ascs%=Asc%
    Print At(72,17);Hex$(Asc%);" "
    Print At(72,19);Str$(Asc%);"  "
  Endif
  '
  If Mousek
    Yt%=8+Int(Asc%/32)*11
    Xt%=6+20*(Asc%-32*Int(Asc%/32))
    If Copy%)0
      For I%=0 To 7
        Poke Varptr(F1$)+Asc%+256*I%,Peek(Varptr(F1$)+Ascw%+256*I%)
      Next I%
      Lpoke &H41A6,Varptr(F1$)
      Text Xt%,Yt%,Chr$(Ascw%)
      @Copy
    Endif
    Text 576,166,Chr$(Asc%)
    Ascs%=Asc%
    Ascw%=Asc%
    For I%=0 To 7
      For J%=0 To 7
        Deffill Point(576+J%,160+I%)
        Pbox 21+20*J%,96+11*I%,39+20*J%,105+11*I%
      Next J%
    Next I%
  Endif
Return
'
```

```
Procedure Pixel
  Mouse X%,Y%,K%
  If K%
    Dx%=(X%-20)/(20)
    Dy%=(Y%-95)/11
    If Dx%()Dsx% Or Dys%()Dy%
      Dsx%=Dx%
      Dys%=Dy%
      Deffill 1,1,1
      Color 1
      If Point(24+20*Dx%,100+11*Dy%)=1
        Deffill 0,1,1
        Color 0
        Poke Varptr(F1$)+Ascw%+256*Dy%,
                        Peek(Varptr(F1$)+Ascw%+256*Dy%)-2^(7-Dx%)
      Else
        Poke Varptr(F1$)+Ascw%+256*Dy%,
                        Peek(Varptr(F1$)+Ascw%+256*Dy%)+2^(7-Dx%)
      Endi
      Text 576,166,Chr$(Asc%)
      Text Xt%,Yt%,Chr$(Ascw%)
      Pbox 21+20*Dx%,96+11*Dy%,39+20*Dx%,105+11*Dy%
    Endif
  Else
    Dsx%=10
    Dys%=10
  Endif
Return
'
Procedure Ster
  Mouse X%,Y%,K%
  If K%
    If Y%)103 And Y%(116
      Fileselect "\*.FNT","",W$
      If W$()""
        F1$=Space$(4096)
        Bload W$,Varptr(Font$)
        @Convert
        Lpoke &H41A8,Varptr(F1$)
        @Plansza
      Endif
    Endif
    If Y%)123 And Y%(136
      Fileselect "\*.FNT","",W$
```

```
      If W$<>""
        @Convert1
        If Right$(W$,4)<>".FNT"
          W$=W$+".FNT"
        Endif
        Bsave W$,Varptr(Font$),4096
      Endif
    Endif
    If Y%>163 And Y%<176
      Alert 3,"Are You sure You want! !        to Quit?",2,
                                                  "Yes!No",J%
      If J%=1
        Lpoke &H41A8,&HFD2F02
        Quit
      Endif
    Endif
    If Y%>143 And Y%<156
      @Sysfn
    Endif
  Endif
Return
'
Procedure Ster1
  Mouse X%,Y%,K%
  If K%
    If Y%>103 And Y%<116
      For I%=0 To 7
        Poke Varptr(F1$)+Ascw%+256*I%,
                                255-Peek(Varptr(F1$)+Ascw%+256*I%)
      Next I%
      Lpoke &H41A8,Varptr(F1$)
      Text 576,166,Chr$(Asc%)
      Text Xt%,Yt%,Chr$(Ascw%)
      For I%=0 To 7
        For J%=0 To 7
          Deffill Point(576+J%,160+I%)
          Pbox 21+20*J%,96+11*I%,39+20*J%,105+11*I%
        Next J%
      Next I%
    Endif
    If Y%>123 And Y%<136
      For I%=0 To 7
        Poke Varptr(F1$)+Ascw%+256*I%,0
      Next I%
```

```
  Lpoke &H41A8,Varptr(F1$)
  Text 576,166,Chr$(Asc%)
  Text Xt%,Yt%,Chr$(Ascw%)
  For I%=0 To 7
    For J%=0 To 7
      Deffill Point(576+J%,160+I%)
      Pbox 21+20*J%,96+11*I%,39+20*J%,105+11*I%
    Next J%
  Next I%
Endif
If Y%)143 And Y%(156
  A$=""
  For I%=0 To 7
    A$=A$+Mid$(F1$,Ascw%+1+256*I%,1)
  Next I%
  For I%=0 To 7
    Poke Varptr(F1$)+Ascw%+256*I%,Asc(Mid$(A$,8-I%,1))
  Next I%
  Lpoke &H41A8,Varptr(F1$)
  Text 576,166,Chr$(Asc%)
  Text Xt%,Yt%,Chr$(Ascw%)
  For I%=0 To 7
    For J%=0 To 7
      Deffill Point(576+J%,160+I%)
      Pbox 21+20*J%,96+11*I%,39+20*J%,105+11*I%
    Next J%
  Next I%
Endif
If Y%)163 And Y%(176
  For I%=0 To 7
    K%=0
    For J%=0 To 7
      Col%=Point(576+I%,160+J%)
      Color Col%
      Plot 7+Xt%-J%,Yt%-6+I%
      K%=K%+Col%*2^J%
    Next J%
    Poke Varptr(F1$)+Ascw%+256*I%,K%
  Next I%
  Lpoke &H41A8,Varptr(F1$)
  Text 576,166,Chr$(Asc%)
  For I%=0 To 7
    For J%=0 To 7
      Deffill Point(576+J%,160+I%)
```

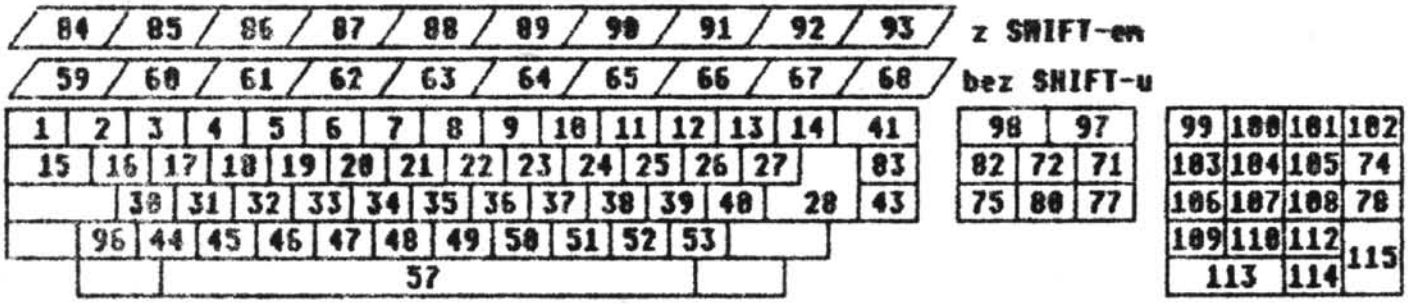
```
        Pbox 21+20*J%,96+11*I%,39+20*J%,105+11*I%
      Next J%
    Next I%
  Endif
Endif
Return
'
Procedure Ster2
  Mouse X%,Y%,K%
  If K%
    If Y%>103 And Y%<116
      Text 574,166," "
      Text 577,166,Chr$ Asc%)
      For I%=0 To 7
        K%=0
        For J%=0 To 7
          Col%=Point(576+J%,160+I%)
          Deffill Col%
          Pbox 21+20*J%,96+11*I%,39+20*J%,105+11*I%
          K%=K%+Col%*2^(7-J%)
        Next J%
        Poke Varptr(F1$)+Ascw%+256*I%,K%
      Next I%
      Lpoke &H41A8,Varptr(F1$)
      Text 576,166,Chr$(Asc%)+" "
      Text Xt%,Yt%,Chr$(Ascw%)
    Endif
    If Y%>123 And Y%<136
      Text 578,166," "
      Text 575,166,Chr$(Asc%)
      For I%=0 To 7
        K%=0
        For J%=0 To 7
          Col%=Point(576+J%,160+I%)
          Deffill Col%
          Pbox 21+20*J%,96+11*I%,39+20*J%,105+11*I%
          K%=K%+Col%*2^(7-J%)
        Next J%
        Poke Varptr(F1$)+Ascw%+256*I%,K%
      Next I%
      Lpoke &H41A8,Varptr(F1$)
      Text 575,166," "
      Text 576,166,Chr$(Asc%)+" "
      Text Xt%,Yt%,Chr$(Ascw%)
```

```
    Endif
    If Y%)143 And Y%(156
      For I%=7 Downto 1
        Poke Varptr(F1$)+Ascw%+256*I%,
                                  Peek(Varptr(F1$)+Ascw%+256*(I%-1))
      Next I%
      Poke Varptr(F1$)+Ascw%,0
      Lpoke &H41A8,Varptr(F1$)
      Text 576,166,Chr$(Asc%)
      Text Xt%,Yt%,Chr$(Ascw%)
      For I%=0 To 7
        For J%=0 To 7
          Deffill Point(576+J%,160+I%)
          Pbox 21+20*J%,96+11*I%,39+20*J%,105+11*I%
        Next J%
      Next I%
    Endif
    If Y%)163 And Y%(176
      For I%=0 To 6
        Poke Varptr(F1$)+Ascw%+256*I%,
                                  Peek(Varptr(F1$)+Ascw%+256*(I%+1))
      Next I%
      Poke Varptr(F1$)+Ascw%+256*7,0
      Lpoke &H41A8,Varptr(F1$)
      Text 576,166,Chr$(Asc%)
      Text Xt%,Yt%,Chr$(Ascw%)
      For I%=0 To 7
        For J%=0 To 7
          Deffill Point(576+J%,160+I%)
          Pbox 21+20*J%,96+11*I%,39+20*J%,105+11*I%
        Next J%
      Next I%
    Endif
  Endif
Return
'
Procedure Sysfn
  Bmove &HFD2F02,Varptr(F1$),2048
  Lpoke &H41A8,Varptr(F1$)
  @Plansza
Return
'
Procedure Plansza
  Deftext 1,0,0,6
```

```
  For I%=0 To 7
    For J%=0 To 31
      Text 6+J%*20,8+I%*11,Chr$(I%*32+J%)
    Next J%
  Next I%
Return
'
Procedure Convert
  I1%=0
  J1%=0
  For I%=1 To 16 Step 2
    J1%=I%-16
    For J%=1 To 256
      Add J1%,16
      Inc I1%
      Mid$(F1$,I1%,1)=Mid$(Font$,J1%,1)
    Next J%
  Next I%
Return
'
Procedure Convert1
  I1%=0
  J1%=0
  For I%=1 To 16 Step 2
    J1%=I%-16
    For J%=1 To 256
      Add J1%,16
      Inc I1%
      Mid$(Font$,J1%,1)=Mid$(F1$,I1%,1)
    Next J%
  Next I%
Return
'
Procedure Copy
  Graphmode 3
  Deffill 1,1,1
  Pbox 526,102,596,116
  Copy%=-Copy%
  While Mousek
  Wend
  Graphmode 1
Return
```

8.6. Definiowanie klawiatury.

Poniższy rysunek przedstawia klawiaturę ST.



Wciśnięcie któregokolwiek z opisanych klawiszy powoduje wysłanie z klawiatury odpowiedniej liczby (zgodnej z rysunkiem, a nie kodu ASCII) oraz informacji o stanie klawiszy Shift i Caps Lock. Kody ASCII zapamiętane są natomiast w trzech 128-elementowych tablicach. Wysłana z klawiatury liczba jest wskaźnikiem tablicy, zaś Shift i Caps Lock przełączają tablice. Pierwsza z nich zawarta w ROM-ie od adresu &hFC2034 odpowiada klawiaturze bez wciśniętego Shift-u i Caps Lock-a, druga (&hFC2034+128) odpowiada klawiaturze z wciśniętym Shift-em, trzecia (&hFC2034+256) z wciśniętym Caps Lock-iem.
Przedefiniowanie klawiatury polega więc na utworzeniu swoich tablic, umieszczeniu i zabezpieczeniu ich w RAM-ie oraz wywołaniu procedury Xbiosu(&h0F) ustalającej adresy tablic.

Poniższy program zawiera w liniach Data standardowe kody klawiatury. Użytkownik powinien wpisać w odpowiednie miejsca swoje kody i uruchomić program. Utworzony plik 'POLISH.KBD' będzie mógł być wczytany przez następny program (8.7.),ładujący polskie znaki i klawiaturę do popularnego procesora tekstów jakim jest Word Plus.

```
A$=String$(384,0)
For I%=0 To 383
  Read B$
  Poke Varptr(A$)+I%,Val("&H"+B$)
Next I%
Bsave "POLISH.KBD",Varptr(A$),384
End
'
```

```
Rem   Zestaw podstawowy
Rem    0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
Rem   ------------------------------------------------
Data 00,1B,31,32,33,34,35,36,37,38,39,30,2D,3D,08,09 ! 0
Data 71,77,65,72,74,79,75,69,6F,70,5B,5D,0D,00,61,73 ! 1
Data 64,66,67,68,6A,6B,6C,3B,27,60,00,23,7A,78,63,76 ! 2
Data 62,6E,6D,2C,2E,2F,00,00,00,20,00,00,00,00,00,00 ! 3
Data 00,00,00,00,00,00,00,00,00,00,2D,00,00,00,2B,00 ! 4
Data 00,00,00,7F,00,00,00,00,00,00,00,00,00,00,00,00 ! 5
Data 5C,00,00,28,29,2F,2A,37,38,39,34,35,36,31,32,33 ! 6
Data 30,2E,0D,00,00,00,00,00,00,00,00,00,00,00,00,00 ! 7
'
Rem   Z Shift-em
Rem    0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
Rem   ------------------------------------------------
Data 00,1B,21,22,9C,24,25,5E,26,2A,28,29,5F,2B,08,09 ! 0
Data 51,57,45,52,54,59,55,49,4F,50,7B,7D,0D,00,41,53 ! 1
Data 44,46,47,48,4A,4B,4C,3A,40,FF,00,7E,5A,58,43,56 ! 2
Data 42,4E,4D,3C,3E,3F,00,00,00,20,00,00,00,00,00,00 ! 3
Data 00,00,00,00,00,00,00,37,38,00,2D,34,00,36,2B,00 ! 4
Data 32,00,30,7F,00,00,00,00,00,00,00,00,00,00,00,00 ! 5
Data 7C,00,00,28,29,2F,2A,37,38,39,34,35,36,31,32,33 ! 6
Data 30,2E,0D,00,00,00,00,00,00,00,00,00,00,00,00,00 ! 7
'
Rem   Z Caps Lock-iem
Rem    0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
Rem   ------------------------------------------------
Data 00,1B,31,32,33,34,35,36,37,38,39,30,2D,3D,08,09 ! 0
Data 51,57,45,52,54,59,55,49,4F,50,5B,5D,0D,00,41,53 ! 1
Data 44,46,47,48,4A,4B,4C,3B,27,60,00,23,5A,58,43,56 ! 2
Data 42,4E,4D,2C,2E,2F,00,00,00,20,00,00,00,00,00,00 ! 3
Data 00,00,00,00,00,00,00,00,00,00,2D,00,00,2B,00,00 ! 4
Data 00,00,00,7F,00,00,00,00,00,00,00,00,00,00,00,00 ! 5
Data 5C,00,00,28,29,2F,2A,37,38,39,34,35,36,31,32,33 ! 6
Data 30,2E,0D,00,00,00,00,00,00,00,00,00,00,00,00,00 ! 7
```

8.7. Program ładuje polskie znaki i klawiaturę do Word Plus'a. Program należy skompilować i umieścić na dysku razem z plikami: 'POLISH.FNT' zawierającym polskie czcionki, 'POLISH.KBD' zawierającym definicje klawiatury oraz procesorem tekstów'WORDPLUS.PRG'. Plik 'POLISH.FNT' można przygotować np. za pomocą edytora fontów przedstawionego w pkt.8.5, zaś 'POLISH.KBD' za pomocą programu przedstawionego w pkt.8.6.

```
Chdir "\"
Reserve 5120
If Exist("POLISH.FNT")
  F1$=Space$(4096)                    !rezerwacja obszaru dla znaków
  X%=Varptr(F1$)
  Bload "POLISH.FNT",Himem
  @Convert
  Lpoke &H41A8,X%                   !zmiana adresu generatora znaków
  Text 150,100,"POLSKIE CZCIONKI ZAINSTALOWAŁ S.NAWROCKI"
  Text 250,130,"KOSZALIN 1988"
Else
  Alert 3,"!NO FILE: 'POLISH.FNT'",1," OK !QUIT",J%
  If J%=2
    @Quit
  Endif
Endif
If Exist("POLISH.KBD")
  F2$=String$(384,0)              !rezerwacja obszaru dla klawiatury
  X%=Varptr(F2$)
  Bload "POLISH.KBD",X%
  Void Xbios(16,L:X%,L:X%+128,L:X%+256)          !zmiana klawiatury
Else
  Alert 3,"!NO FILE: 'POLISH.KBD'",1," OK !QUIT",J%
  If J%=2
    @Quit
  Endif
Endif
Exec 0,"WORDPLUS.PRG","",""                    !wywołanie Word Plus-a
@Quit
End
'
Procedure Quit
  Lpoke &H41A8,&HFD2F02         !włączenie standardowego generatora
  X%=&HFC2034
  Void Xbios(16,L:X%,L:X%+128,L:X%+256)    !włączenie standardowej
```

```
  Quit
Return
'
Procedure Convert
  I1%=-1
  J1%=0
  For I%=1 To 16 Step 2
    J1%=I%-16
    For J%=1 To 256
      Add J1%,16
      Inc I1%
      Poke X%+I1%,Peek(Himem+J1%-1)
    Next J%
  Next I%
Return
'
```

9. SKROTY I POJECIA OBCOJEZYCZNE Z WYJASNIENIAMI.

AES - patrz: ROM

Address - wartość 32-bitowa określająca położenie komórki pamięci

ASCII (American Standard Code for Information Interchange - amerykański kod standardowy do wymiany informacji). Zestaw znaków, zawierający litery, cyfry, znaki specjalne oraz znaki sterujące, którym przyporządkowano określone liczby (patrz: Zestaw Znaków Dostępnych w GFA-BASIC'u).

AUX: (Auxiliary) - skrót,który użyty zamiast nazwy pliku w komendzie Open (np. Open "O",#1,"AUX:"), powoduje traktowanie łącza RS-232 jako pliku zewnętrznego

Basepage - patrz: TPA

BDOS - patrz: ROM

BIOS - patrz: ROM

BSS - patrz: TPA

Byte (bajt) - wartość 8-bitowa

CCP (Console Command Processor) - moduł GEMDOS-u odpowiedzialny za interpretację i wykonywanie poleceń wydawanych z konsoli

Character - znak

Cmd - (Command line - linia komend) - obszar Basepage, którego treść jest parametrem komendy Exec

CON: (Console) - skrót, który użyty zamiast nazwy pliku w komendzie Open (np. Open "O",#1,"CON:"), powoduje traktowanie konsoli jako pliku zewnętrznego

Directory - skorowidz dysku, spis jego zawartości

Disk (Dysk) - Atari ST wykorzystuje 3.5 calowe dyski jedno lub dwustronnie. Jedna strona dysku zawiera 80 ścieżek (track), numerowanych od 0 - ścieżka zewnętrzna do 79 - ścieżka wewnętrzna. Każda ścieżka zawiera 9 sektorów (sector) po 512 bajtów każdy. Tak sformatowany dysk ma pojemność 368640 bajtów.
Pierwszy sektor ścieżki zerowej zajęty jest przez Boot - zawiera on program ładujący system oraz informacje o formacie dysku. Następne 5 sektorów zajęte jest przez FAT (File Allocation Table-tablica rozmieszczenia plików) i dalej siedem sektorów przez katalog dysku. Ponieważ uszkodzenie FAT-u uniemożliwiłoby dostęp do całego dysku, jest on najczęściej kopiowany, stąd przestrzeń dostępna dla użytkownika wynosi 357 kB. Informacje te dotyczą standardowo sformatowanego dysku; istnieją programy formatujące dysk np. na 82 ścieżki na stronę.

FAT - patrz: Disk

File - plik, zbiór

File name - nazwa zbioru

GEM - patrz: ROM

GEMDOS - patrz: ROM

Handle - (dosłownie: rączka) - niektóre procedury GEM-u (np. otwarcie okna lub pliku) powodują wytworzenie identyfikatora, na który należy się powoływać podczas dalszych operacji na oknie lub pliku. Identyfikator ten nazwano 'handle'

I - (Integer) - wartość całkowita traktowana jako 16-bitowa

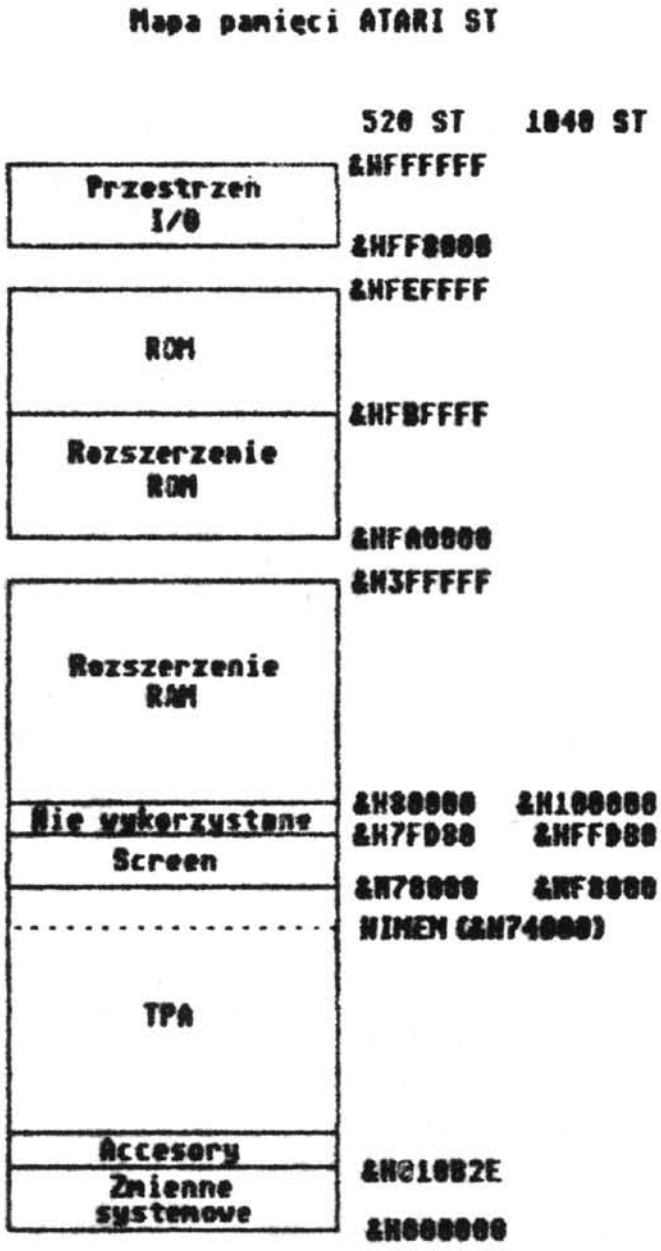
L: - zapis używany przy wywoływaniu procedur systemowych oznaczający, że występująca po nim dana będzie traktowana jako 32-bitowa

Length - długość (zbioru, zmiennej tekstowej)

LI - (Long Integer) - wartość całkowita traktowana jako 32-bitowa

Longword (długie słowo) - wartość 32-bitowa

Memory map (Mapa pamięci) - schematyczne przedstawienie pamięci komputera z opisem przeznaczenia poszczególnych obszarów

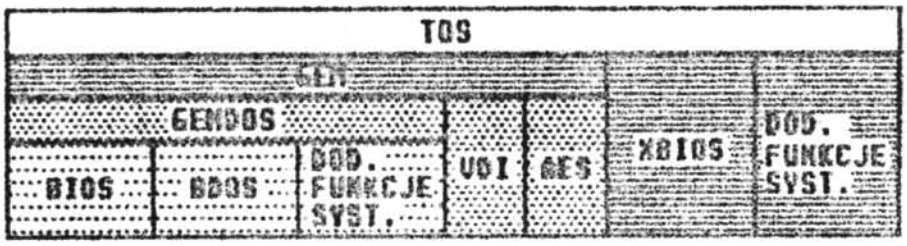


Nibble - wartość 4-bitowa

PRN: (Printer) - skrót, który użyty zamiast nazwy pliku w komendzie Open (np. Open "O",#1,"PRN:"), powoduje traktowanie drukarki jako zewnętrznego pliku wyjściowego

RAM (Random Access Memory - pamięć o swobodnym dostępie) - patrz: Memory Map

ROM (Read Only Memory - pamięć tylko do czytania) - pomięć stała zawierająca oprogramowanie systemowe. Struktura ROM-u ATARI ST jest nastepujaca:



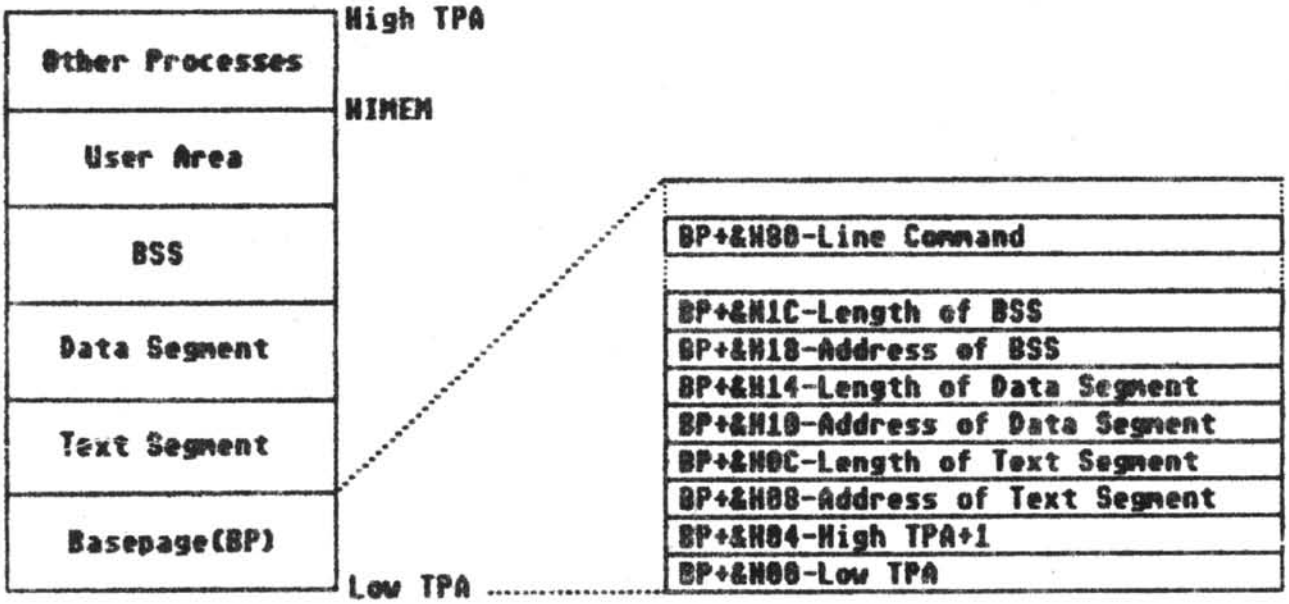
- TOS (Tramiel Operating System) - system operacyjny opracowany w firmie ATARI specjalnie dla serii ST, początkowo ładowany z dysku, obecnie zawiera się w 192 kB pamięci stałej.
  Zawiera trzy bloki:
  - GEM (Graphics Environment Manager) - system operacyjny pozwalający zarządzać wszystkimi zasobami komputera za pomocą symboli graficznych. Składa się z trzech modułów:
    - VDI (Virtual Device Interface) - zawiera programy wykonujące elementarne operacje graficzne jak np. rysowanie odcinka, okręgu, wydzielanie i przemieszczanie prostokątnego fragmentu ekranu,
    - AES (Application Environment Services) - zawiera program aktywujący procesy, programy zarządzające ekranem i akcesoriami oraz bibliotekę podprogramów obsługujących menu, okna i mysz,
    - GEMDOS (GEM Disk Operating System)- zawiera trzy składniki
      - BIOS (Basic Input Output System) - obsługuje drukarkę, interfejs szeregowy, monitor, port MIDI, klawiaturę oraz we/wy blokowe dla dysków elastycznych i twardych,
      - BDOS (Basic Disk Operating System) - wykonuje wszystkie operacje na plikach (tworzenie, otwieranie, zamykanie, zapis odczyt),
      - Dodatkowe funkcje systemowe - zarządzanie pamięcią,ustawianie czasu systemowego, zakończenie programu.
  - XBIOS (Extended Basic Input Output System) zapewnia obsługę funkcji związanych z architekturą sprzętową rodziny ST np. obsługę układu dźwiękowego.
  - Funkcje dodatkowe - zestaw procedur graficznych wspomagających GEM.

Sector - patrz Disk

String - łańcuch znaków, dana tekstowa

TOS - patrz: ROM

TPA (Transient Program Area - obszar programów przejściowych), obszar pamięci o następującej strukturze:



- Other Processes - Obszar o objętości X=Gemdos(&H48,L:-1) dostępny dla innych procesów,
- User Area - Obszar roboczy o objętości Fre(0),
- BSS - Block Storage Segment - Obszar Programu zawierający dane robocze programu,
- Data Segment - Obszar programu zawierający dane wejściowe programu,
- Text Segment - Obszar programu zawierający tekst programu,
- Basepage - Strona bazowa programu - obszar programu zawierający dane dotyczące wykonywanego programu.

Po załadowaniu GFA-Basic'a zastaje utworzony jego TPA taki, że dla innych procesów zarezerwowane jest 16384 bajtów (HIMEM=Lpeek (Basepage +&H04)-16384). Wielkość tego obszaru można zmieniać komendą Reserve, np.:Reserve Fre(0)+1024 przesuwa Himem 1024 bajty w górę zwiększając dostępną pamięć o tyleż bajtów. Zmniejszenie tego obszaru do zera prowadzi często do zakłóceń w pracy GFA-Basic'a, zwłaszcza podczas ładowania i zapisywania programów.

Track - patrz: Disk

Variable - zmienna

VDI - patrz: ROM

W: - zapis używany przy wywoływaniu procedur systemowych oznaczający, że występująca po nim dana będzie traktowana jako 16-bitowa

Wildcard character - oznacza znak * lub ?. Przy wywoływaniu plików każdy znak ? oznacza dowolną literę lub cyfrę, znak * oznacza dowolny łańcuch np. Dir *.* spowoduje wywołanie katalogu zawierającego wszystkie pliki na dysku.

XBIOS - patrz: ROM

Druk ZRiWDB Warszawa ul. Królewska 27
Format A-5 Nakład 5050 egz. Ark. 9,75
Zam. Nr 144 1988 V 11. U-65

Cena zł [redacted]-

SOETO BIT

Wydawca SOETO
ul. Hoża 50, 00-682 Warszawa
Tel. 21-64-01 w. 66 lub 29-18-64
Tlx. 81-47-86