С Ъ Д Ъ Р Ж А Н И Е

Забележка: Командите (nentsel <въпрос>) и (findfile <име на файл>) не са описани тук !!!

Глава 1

Функции в AutoLISP

AutoLISP предоставя многобройни, предварително дефинирани функции. Всяка функция се извиква чрез въвеждане на името на функцията (с малки или големи букви) като първи елемент на списък, следван от аргументите на тази функция (ако има такива).

В тази глава ще намерите пълно представяне на функциите на AutoLISP по азбучен ред. При много функции се касае за стандартни функции, които се съдържат във всяка реализация на програмния език AutoLISP. Други функции са характерни за графичните приложения на AutoCAD. По-новите функции са описани в последните глави.

1.1 (+ <число> <число> ...)

Тази функция дава като резултат сумата от всички числа. Тя може да се използва с реални и цели числа. Когато всички числа са цели, резултатът също е цяло число (за реалните е аналогично).

Пример:

(+ 1 2) резултат 3

(+ 1 2 3 4.5) резултат 10.500000

(+ 1 2 3 4.0) резултат 10.000000

1.2 (- <число> <число> ...)

Тази функция изважда второто число от първото и дава резултата. Ако са дадени повече от две числа, сумата на второто до последното се изважда от първото число и се дава резултата. При въвеждане само на едно число, то се изважда от нула. Тази функция се използва с реални и цели числа.

Пример:

(- 50 40) резултат 10

(- 50 40.0 2) резултат 8.000000

(- 50 40.0 2.5) резултат 7.500000

(- 8) резултат -8

1.3 (* <число> <число> ...)

Тази функция дава като резултат произведението от всички числа. Може да се използва с реални и цели числа.

Пример:

(* 2 3) резултат 6

(* 2 3 4.0) резултат 24.000000

(* 3 -4.5) резултат -13.500000

1.4 (/ <число> <число> ...)

Тази функция дели първото число с второто и дава частното като резултат. Ако са дадени повече от две числа, първото число се дели на произведението от второто до последното дава резултата. Тази функция се използва с реални и цели числа.

Пример:

(/ 100 2) резултат 50

(/ 100 2.0) резултат 50.000000

(/ 100 20 2.0) резултат 2.500000

(/ 100 20.0 2) резултат 2.500000

(/ 100.0 20 2) резултат 2.500000

(/ 100 20 2) резултат 2

(/ 135 360) резултат 0

(/ 135 360.0) резултат 0.375000

1.5 (= <атом> <атом> ...)

Касае се за функцията "equal to" (равно). Ако всички <атоми> са числово идентични, резултатът е T (true = истина), в противен случай - Nil. Тази функция е в сила за числа и символни низове.

Пример:

(= 4 4.0) резултат T

(= 20 388) резултат Nil

(= 2.4 2.4 2.4) резултат T

(= 499 499 500) резултат Nil

(= "i" "i") резултат T

(= "i" "you") резултат Nil

1.6 (/= <атом1> <атом2> ...)

Касае се за функцията "not equal to" (не е равно). Ако първият <атом1> числово не е равен на втория <атом2>, резултатът е T, в противен случай - Nil. Ако са дадени повече от два <атома>, преглежда се цялата серия. Тази функция е в сила за числа и символни низове.

Пример:

(/= 10 20) резултат T

(/= "you" "you") резултат Nil

(/= 5.43 5.44) резултат T

1.7 (< <атом> <атом> ...)

Касае се за функцията "less than" (по-малко от). Ако първият <атом> е по-малък от втория, резултатът е T, в противен случай - Nil. Ако са дадени повече от два <атома>, резултатът е T, ако всеки <атом> е по-малък от стоящия отдясно <атом>.

Пример:

(< 10 20) резултат T

(< "b" "c") резултат T

(< 357 33.2) резултат Nil

(< 2 3 88) резултат T

(< 2 3 4 4) резултат Nil

1.8 (<= <атом> <атом> ...)

Това е функцията "less than or equal" (по-малко или равно). Ако първият <атом> е по-малък или равен от втория, резултатът е T, в противен случай - Nil. Ако са дадени повече от два <атома>, резултатът е T, ако всеки <атом> е по-малък или равен на <атома> отдясно.

Пример:

(<= 10 20) резултат T

(<= "b" "b") резултат T

(<= 357 33.2) резултат Nil

(<= 2 9 9) резултат T

(<= 2 9 4 5) резултат Nil

1.9 (> <атом> <атом> ...)

Това е функцията "greater than" (по-голямо от). Ако първият <атом> е по-голям от втория, резултатът е T, в противен случай - Nil. Ако са дадени повече от два <атома>, резултатът е T, ако всеки <атом> е по-голям от <атома> отдясно.

Пример:

(> 120 17) резултат T

(> "b" "c") резултат T

(> 3.5 1792) резултат Nil

(> 77 4 2) резултат T

(> 77 4 4) резултат Nil

1.10 (>= <атом> <атом> ...)

Това е функцията "greater than or equal" (по-голямо или равно). Ако първият <атом> е по-голям или равен от втория, резултатът е T, в противен случай - Nil. Ако са дадени повече от два <атома>, резултатът е T, ако всеки <атом> е по- голям или равен на <атома> отдясно.

Пример:

(>= 120 17) резултат T

(>= "b" "b") резултат T

(>= 3.5 1792) резултат Nil

(>= 77 4 4) резултат T

(>= 77 4 9) резултат Nil

1.11 (~ <число>)

Функцията дава битовия NOT-член на <число>. Работи с цели числа.

Пример:

(~ 3) резултат -4

(~ 100) резултат -101

(~ -4) резултат 3

1.12 (1+ <число>)

Функцията дава като резултат <число> плюс 1 (инкремента). Работи с цели и реални числа.

Пример:

(1+ 5) резултат 6

(1+ -17.5) резултат -16.500000

1.13 (1- <число>)

Функцията намалява <число> с 1 (декремента). Работи с цели и реални числа.

Пример:

(1- 5) резултат 4

(1- -17.5) резултат -18.500000

1.14 (abs <число>)

Функцията дава абсолютната стойност на <число>. Работи с цели и реални числа.

Пример:

(abs 100) резултат 100

(abs -100) резултат 100

(abs -99.25) резултат 99.250000

1.15 (and <израз> ...)

Тази функция дава като резултат логическото AND на списък от изрази. Ако един от изразите се оценява на Nil, оценяването се прекратява и резултатът е Nil, в противен случай - T.

Пример:

```
(setq a 103)
(setq b Nil)
(setq c "text")
```

Резултатите ще са следните:

(and 1.4 a c) резултат T

(and 1.4 a b c) резултат Nil

1.16 (angle <точка1> <точка2>)

Тази функция дава като резултат ъгъла (в радиани) между <точка1> и <точка2>, при което една <точка> е списък от две реални числа.

Пример:

(angle '(5.0 1.33) '(2.4 1.33)) резултат 3.141593

1.17 (angtos <ъгъл> [<режим> [<точност>]])

Тази функция взема един <ъгъл> (реално число в радиани) и го връща редактирано в символен низ. Аргументът <режим> е цяло число, което определя редакционния формат.

<Режим> РЕДАКЦИОНЕН формат .

0 - градуси

1 - градуси/минути/секунди

2 - гради

3 - радиани

4 - геодезически единици

Аргументът <точност> е цяло число, което определя желаната точност на позициите след десетичната точка. <Режим> и <точност> съответстват н а системните променливи от AutoCAD - AUNITS и AUPREC (ако се пропуснат, се използват актуалните им стойности).

Пример:

(setq pt1 '(5.0 1.33))

(setq pt2 '(2.4 1.33))

(setq a (angle pt1 pt2))

От това следва:

(angtos a 0 0) резултат "180"

(angtos a 0 4) резултат "180.0000"

(angtos a 1 4) резултат "180d0'0""

(angtos a 3 4) резултат "3.1416r"

(angtos a 4 2) резултат "W"

1.18 (append <израз> ...)

Тази функция взема произволно много списъци (<израз>) и ги свързва в списък.

Пример:

(append '(a b) '(c d)) резултат (A B C D)

(append '((a) (b)) '((c) (d))) резултат ((A) (B) (C) (D))

1.19 (apply <функция> <списък>)

Изпълнява <функция> с дадени в <списък> аргументи.

Пример:

(apply ‘+ ‘(1 2 3)) резултат 6

(apply ‘strcat ‘("a" "b" "c")) резултат "abc"

1.20 (ascii <символен низ>)

Тази функция преобразува първия знак от <символен низ> като ASCII-знаков код (цяло число) - подобно на функцията ASC в BASIC.

Пример:

(ascii "A") резултат 65

(ascii "a") резултат 97

(ascii "BIG") резултат 66

1.21 (assoc <елемент> <асоциативен списък>)

Тази функция претърсва <асоциативен списък> за <елемент> като ключ и дава като резултат асоциативния списък за съответния ключ. Ако <елемент> не е намерен като ключ в <асоциативен списък>, ASSOC връща Nil.

Пример:

ако списъкът "a1" е дефиниран като

((name box) (width 3) (size 4.7263) (depth 5))

от това следва:

(assoc ‘size a1) резултат (size 4.7263)

(assoc ‘weight a1) резултат Nil

Асоциативните списъци често се използват за записване на данни, към които е възможно обръщение с помощта на "ключ". Функцията SUBST, която е описана по-надолу в тази глава, е практично средство да се замени стойността, която е асоциирана с ключ в <асоциативен списък>.

1.22 (atan <число1> [<число2>])

Ако не е дадено <число2>, ATAN дава аркустангенса на <число1> в радиани. <Число1> може да е отрицателно; областта на ъгъла, който се дава като резултат, е от -Pi до +Pi.

Пример 1:

(atan 0.5) резултат 0.463647

(atan 1.0) резултат 0.785398

(atan -1.0) резултат -0.785398

(angtos (atan -1.0) 0 4) резултат "-45.0000"

Ако са дадени <число1> и <число2>, резултатът е аркустангенса на <число1>/<число2> в радиани. Ако <число1> е 0, резултатът е ъгъл от 1.570796rad (90 или -90 градуса), в зависимост от знака на <число1>.

Пример 2:

(atan 2.0 3.0) резултат 0.588002

(angtos (atan 2.0 3.0) 0 4) резултат "33.6901"

(atan 2.0 -3.0) резултат 2.553590

(angtos (atan 2.0 -3.0) 0 4) резултат "146.3099"

(atan -2.0 3.0) резултат -0.588002

(atan -2.0 -3.0) резултат -2.553590

(atan 1.0 0.0) резултат 1.570796

(angtos (atan 1.0 0.0) 0 4) резултат "90.0000"

(atan -0.5 0.0) резултат -1.570796

(angtos (atan -0.5 0.0) 0 2) резултат "-90.00"

1.23 (atof <символен низ>)

Тази функция преобразува <символен низ> в реално число.

Пример:

(atof "97.1") резултат 97.100000

(atof "3") резултат 3.000000

1.24 (atoi <символен низ>)

Тази функция преобразува <символен низ> в цяло число.

Пример:

(atoi "97") резултат 97

(atoi "3") резултат 3

(atoi "3.9") резултат 3

1.25 (atom <елемент>)

Ако <елемент> е списък, тази функция дава като резултат Nil, в противен случай - T. Това, което не е списък, се разглежда като ATOM.

Пример:

(setq a ‘(x y z))

(setq b ‘a)

От това следва:

(atom ‘a) резултат T

(atom a) резултат Nil

(atom ‘b) резултат T

(atom b) резултат T

(atom ‘(a b c)) резултат Nil

ЗАБЕЛЕЖКА: Не всички LISP-езици интерпретират еднакво един ATOM, затова проверете преобразуващите кодове.

1.26 (boole <функция> <цяло число1> <цяло число2> ...)

Това е обща булева функция, която се прилага бит по бит на две цели числа. <Функция> е цяло число между 0 и 15, съответстваща на една от 16-те възможни булеви функции на две променливи. Целочислените аргументи се комбинират бит по бит (логически) в зависимост от избраната булева функция и въз основа на таблица за истинност:

цяло число 1	цяло число 2	функция	резултат
0	0	8	1
0	1	4	1
1	0	2	1
1	1	1	1

Всеки бит от <цяло число 1> се комбинира със съответния бит от <цяло число 2> чрез избор на хоризонтален ред от таблицата за истинност. Резултиращият бит е или 0, или 1, в зависимост от стойността (0 или 1) на бита на <функция> от съответния ред от таблицата за истинност. Когато съответния бит във <функция> е зареден, резултиращият бит е 1, в противен случай - 0.

Някои стойности на <функция> съответстват на стандартните булеви операции AND, OR, XOR и NOT:

функция	операция	резултиращият бит е 1 ако
1	AND	двата бита са 1
6	XOR	само единия от двата бита е 1
7	OR	единия или двата бита е/са 1
8	NOT	двата бита са 0 (комплемент на 1)

Пример 1:

(boole 1 12 5) резултат 4 при AND на 12 и 5

(boole 6 6 5) резултат 3 при XOR на 6 и 5

Можете да използвате други стойности на <функция> за изпълнение на други булеви операции, за които няма стандартни имена.

Пример 2:

Ако <функция> е 4, резултиращите битове ще са 1, ако съответните битове в <цяло число 2> (но не и в <цяло число 2>) са 1:

(boole 4 3 14) резултат 12

1.27 (boundp <атом>)

Тази функция дава като резултат T, ако <атом> притежава присвоена стойност (независимо за каква цел). Ако на <атом> не е присвоена стойност или е присвоена Nil, резултатът е Nil.

Пример:

(setq a 1)

(setq b Nil)

От това следва:

(boundp ‘a) резултат T

(boundp ‘b) резултат Nil

1.28 caar, cadar, caddr, cadr, cdar и cddr

AutoLISP поддържа свързване на CAR и CDR до 4 нива.

Пример 1:

(setq x ‘((a b) c d)

ще даде:

(caar x) еквивал.на (car (car x)) резултат A

(cdar x) еквивал.на (cdr (car x)) резултат (B)

(cadar x) еквивал.на (car (cdr (car x))) резултат B

(cadr x) еквивал.на (car (cdr x)) резултат C

(cddr x) еквивал.на (cdr (cdr x)) резултат (D)

(caddr x) еквивал.на (car (cdr (cdr x))) резултат D

В AutoLISP CADR често се използва за получаване на Y-стойността (втори елемент на списък от 2 реални числа). Също така CADDR може да се използва за получаване на Z-координатата на 3D-точка.

Пример 2:

(setq pt2 ‘(5.2 1.0)) това е 2D-точка

(setq pt3 ‘(5.2 1.0 3.0)) това е 3D-точка

тогава:

(car pt2) резултат 5.200000

(cadr pt2) резултат 1.000000

(caddr pt2) резултат Nil

(car pt3) резултат 5.200000

(cadr pt3) резултат 1.000000

(caddr pt3) резултат 3.000000

1.29 (car <списък>)

Тази функция дава като резултат първия елемент на <списък>. Ако <списък> е празен, резултатът е Nil.

Пример:

(car ‘(a b c)) резултат A

(car ‘((a b) c)) резултат (A B)

(car ‘()) резултат Nil

1.30 (cdr <списък>)

Тази функция дава списък, съдържащ всички елементи на <списък>, с изключение на първия. Ако <списък> е празен, резултатът е Nil.

Пример 1:

(cdr ‘(a b c)) резултат (B C)

(cdr ‘((a b) c)) резултат (C)

(cdr ‘()) резултат Nil

Ако аргументът <списък> е точкова двойка(виж CONS по-долу), CDR дава като резултат втория елемент, без да го включва в списък.

Пример 2:

(cdr ‘(a . b)) резултат B

(cdr ‘(1 . "Text")) резултат "Text"

1.31 (chr <число>)

Тази функция преобразува цяло число, представляващо ASCII-знаков код, в символен низ, състоящ се от един символ(като функцията CHR$ в BASIC).

Пример:

(chr 65) резултат "A"

(chr 66) резултат "B"

(chr 97) резултат "a"

1.32 (close <променлива, указваща файл>)

Тази функция затваря файл и дава Nil. <Променлива, указваща файл> предварително е зададена с функцията OPEN. След CLOSE променливата остава непроменена, обаче вече не е валидна.

Пример:

Ако X е валидна променлива, указваща файл, тогава

(close X)

затваря асоциирания файл и дава Nil.

1.33 (command <аргументи> ...)

Тази функция изпълнява AutoCAD-команди от AutoLISP и винаги дава Nil. Аргументите са за AutoCAD-командите и техните подкоманди; всеки аргумент се оценява и се предава като отговор на следващите запитвания на AutoCAD. Имената на командите и опциите се дават като символни низове: 2D-точките като списък от 2 реални числа, а 3D-точките като списък от 3 реални числа.AutoCAD познава имената на командите само при запитване "Command:".

Пример 1:

```
(setq pt1 ‘(1.45 3.23))
(setq pt2 (getpoint "Укажете точка: "))
(command "line" pt1 pt2)
(command "")
```

След запитването на AutoCAD - "Command:", горедадената поредица от изрази поставя стойност на точка pt1, пита за точка pt2 и накрая изпълнява AutoCAD-командата LINE с двете точки. Аргументите на COMMAND могат да са символни низове, реални числа, цели числа или точки (според командата). Празен символен низ "" съответства на натискане на "Space" от клавиатурата. Извикването на COMMAND без аргументи съответства на въвеждане на CTRL+C от клавиатурата и прекъсва повечето команди на AutoCAD.

Командите, изпълнявани от COMMAND, не се пренасят на екрана, ако системната променлива на AutoCAD CMDECHO е 0 (достъпно чрез SETVAR и GETVAR). Функцията COMMAND е най-простия метод за извикване на AutoCAD команди чрез AutoLISP.

ЗАБЕЛЕЖКА: GETxxx-функциите, въвеждани от потребителя (например GETSTRING, GETINT, GETPOINT и т.н.) не могат да се използват във функцията COMMAND. Ако въпреки това се направи опит, ще се появи съобщение "error" и функцията ще бъде прекъсната. Но ако са необходими данни от потребителя, извикайте функцията GETxxx преди това, както е показано горе, или я поставете между следващите "Command:".

Командите DTEXT и SKETCH четат клавиатурата и дигитализиращото устройство и затова не могат да се използват с функцията COMMAND. Също така COMMAND не може да се използва за изпълнение на командите PLOT, PRPLOT и SCRIPT.

Ако една AutoCAD-команда е активна и символът DELAY е аргумент на COMMAND, последната се прекъсва и позволява директно въвеждане на данни от потребителя (или видимо водене на буксир). Това съответства на механизма на обратната наклонена черта, предназначен за менютата.

Ако извикате прозрачна команда, когато функцията COMMAND е прекъсната, последната остава такава. Затова например ‘ZOOM и ‘PAN могат да бъдат използвани произволно често в такава ситуация. Паузата се задържа, докато AutoCAD получи валидни данни и никаква прозрачна команда не е активна.

Пример 2:

```
(command "circle" "5,5" delay "5,5" "7,5" "")
```

Тази последователност активира командата CIRCLE, определя център с координати 5,5 и след това прави пауза, за да може потребителят видимо да води радиуса на окръжността. След като потребителят посочи желаната точка (или набере радиуса от клавиатурата), функцията се взема отново и се изчертава линия от 5,5 до 7,5.

Въвеждане на меню не се прекъсва от AutoLISP-пауза. Ако е активна една меню-функция по времето, когато COMMAND е в пауза и чака данни от потребителя, тези данни могат да се въведат също и чрез менюто. Ако искате да прекъснете меню-функцията, трябва да ú зададете обратна наклонена черта. След като са дадени валидни данни, отново се вземат както функцията COMMAND, така и меню-функцията.

ЗАБЕЛЕЖКИ :

1. Символът DELAY е символен низ, който се състои само от една единствена наклонена черта. Вместо него бихте могли да използвате директно обратна наклонена черта. Когато обаче COMMAND се извика от меню-функция, обратната наклонена черта прекъсва меню-функцията, а не функцията COMMAND. Но за всеки случай Ви препоръчваме да използвате символа DELAY вместо обратната наклонена черта.

2. Ако DELAY е предварително указана, когато една команда очаква въвеждане на текстов символен низ или атрибутна стойност, AutoCAD очаква данни от потребителя само тогава, когато системната променлива TEXTEVAL е със стойност, различна от 0. В противен случай стойността на символа DELAY (една единствена наклонена черта) се приема като действителен текст и не прави пауза за данни от потребителя.

3. Ако функцията COMMAND е в пауза за данни от потребителя, тя се разглежда като все още активна, така че потребителят не може да зададе друг AutoLISP-израз за оценяване.

1.34 (cond (<тест 1> <резултат 1> ...) ...)

Тази функция приема произволно много списъци като аргументи. Тя оценява първия елемент във всеки списък (в дадения ред), докато един от тези елементи покаже стойност, различна от Nil. След това се оценяват следващите от успешния тест изрази и като резултат се дава стойността на последния израз от подсписъка. Ако подсписъка съдържа само един израз (когато напр. Липсва <резултат>), като резултат се дава стойността на израза <тест>. **COND е най-важната условна функция на AutoLISP.**

Пример 1:

В следващия пример COND се използва за изчисление на абсолютна стойност.

```
(cond ((minusp a) (- a))
      (t a)
)
```

Ако "a" има стойност -10, резултатът ще е 10. Обикновено T се използва като предварително зададен <тест>-израз.

Пример 2:

В символа "s" е даден отговор на потребителя във формата на символен низ. Тази функция проверява отговора и дава 1, ако той е "Y" или 0, ако е "N", в противен случай - Nil.

```
(cond ((=s "Y") 1)
      ((=s "y") 1)
      ((=s "N") 0)
      ((=s "n") 0)
      (t Nil)
)
```

1.35 (cons <нов първи елемент> <списък>)

Това е основната функция за конструиране на списъци. Тя взема един елемент (<нов първи елемент>) и един <списък> и прибавя този елемент в началото на <списък>.

Пример 1:

(cons ‘a ‘(b c d)) резултат (A B C D)

(cons ‘(a) ‘(b c d)) резултат ((A) B C D)

*** Първи елемент може да е АТОМ или СПИСЪК.

CONS приема също вместо аргумент <списък> 1 атом и конструира структура, която се нарича точкова двойка (dotted pair). При изобразяването на точкова двойка, AutoLISP поставя

точка между първия и втория елементи. Точковите двойки се нуждаят от по-малко памет в сравнение с обикновените списъци и втория атом може да се изведе с функцията CDR.

Пример 2:

(cons ‘a ‘b) резултат (A B)

(car (cons ‘a ‘b)) резултат A

(cdr (cons ‘a ‘b)) резултат B

1.36 (cos <ъгъл>)

Тази функция дава като резултат косинуса на <ъгъл> в радиани.

Пример:

(cos 0.0) резултат 1.000000

(cos Pi) резултат -1.000000

1.37 (defun <име на функция> <списък на аргументи> <изрази> ...)

DEFUN дефинира функция с име <име на функция> (появява се автоматично в кавички). След името на функцията следва <списък на аргументи> (евентуално празен), следван от факултативна наклонена черта и името на 1 или няколко локални символи за функцията. Наклонената черта трябва да е отделена от първия локален символ и от последния аргумент (ако има такъв) най-малко с един интервал. Ако липсват локални символи или аргументи, след името на функцията трябва да има един чифт скоби.

Пример 1:

(defun myfunc (x y) ...) има 2 аргумента

(defun myfunc (/ a b) ...) има 2 локални символа

(defun myfunc (x / temp) ...) има 1 арг. и 1 лок.символ

(defun myfunc () ...) няма аргументи и локални символи

След списъка от аргументи и локални символи следват 1 или няколко израза, които се оценяват при изпълнение на функцията.

DEFUN сама дава името на дефинираната функция. При извикване на така дефинираната функция нейните аргументи се оценяват и се присвояват на символи. Локалните символи могат да се използват вътре във функцията, без да се променят при това техните връзки на външни нива. Функцията дава резултата на последния оценен израз. Всички предшестващи изрази имат само вторично действие.

Пример 2:

```
(defun add10 (x)
            (+ 10 x)
)                          резултат ADD10
(add10 5)                  резултат 15
(add10 -7.4)          резултат 2.600000
```

Пример 3:

(defun points (x y / temp)

```
                (setq temp (strcat x "..."))
                (strcat temp y)
        )                       резултат points
        (points "a" "b")        резултат "a...b"
```

ЗАБЕЛЕЖКА: За <име на функция> никога не използвайте името на системна функция или системен символ, защото така ще се лишите от достъп до тази функция или символ.

***** 1. Библиотеки с функции и автоматично зареждане.**

Дефинициите на функции могат да се запишат във файлове с AutoLISP-функцията LOAD, която е описана по-долу(1.63). Ако файлът "acad.lsp" вече съществува, той ще се зарежда автоматично всеки път при извикване на чертожния редактор. Благодарение на това могат да се създават библиотеки с често използвани функции. Така Вие ще сте сигурен, че те са налице при необходимост.

Освен DEFUN, всеки файл от типа ".lsp" може да съдържа изрази. Тъй като зареждането на файл предизвиква оценяване на съдържащите се в него изрази, може да бъде включено извикването на функции, които автоматично ще бъдат изпълнявани при всяко зареждане на този файл. Файлът "acad.lsp" обаче се зарежда преди да бъде напълно инициализиран графичния редактор на AutoCAD. Затова не трябва да използвате функцията COMMAND вътре във файла "acad.lsp".

***** 2. Функции C:XXX - вмъкване на команди в AutoCAD.**

С DEFUN могат да се дефинират функции, които внедряват нови команди в AutoCAD. За да могат такива функции да се използват като команди на AutoCAD, трябва да се спазват следните ПРАВИЛА:

¹ 1. Името на функцията трябва да има формат "C:XXX", при което всички букви са главни. "C:" е задължително; "XXX" може да е име на команда по Ваш избор, при условие че то не се среща като вътрешна или външна команда на AutoCAD.

¹ 2. Функцията трябва да се дефинира със списък на Nil-аргументи (разрешени са локални променливи). По-долу дефинираната функция чертае квадрат с полилиния.

Пример 4:

```
(defun C:PQUADRAT ()
        (setq pt1 (detpoint "Долен ляв ъгъл: "))
        (setq l (detdist "Дължина на едната страна: "))
        (setq pt2 (polar pt1 0.0))
        (setq pt3 (polar pt2 (/ Pi 2.0) l)
        (setq pt4 (polar pt3 Pi l))
        (command "PLINE" pt1 pt2 pt3 pt4 "S")
)
```

По този начин създадените функции могат да се извикват с въвеждане на компонентата "XXX" от името на функцията, когато се появи запитването на AutoCAD "Command:". Ако "XXX" е непозната команда, AutoCAD опитва да извика AutoLISP-функция "XXX" без параметри. За примера по-горе диалогът би изглеждал така:

```
Command: PQUADRAT
```

Долен ляв ъгъл: (въвеждане на точка)

Дължина на едната страна: (въвеждане на разстояние)

Функцията извиква командата на AutoCAD PLINE и отговаря на нейните въпроси така, че да начертае желания квадрат.

Вмъкването на команди по този начин е важна функция на AutoLISP. Веднъж дефинирана, новата команда може да използва всички свойства на AutoLISP. Новата команда не трябва да се въвежда в скоби, поради което тази, реализирана с AutoLISP команда, може да се използва като всяка друга команда на AutoCAD.

1.38 (distance <точка1> <точка2>)

Тази функция дава разстоянието между две 2D-точки <точка1> и <точка2>, при което една 2D-точка е списък от 2 реални числа.

Пример:

(distance '(1.0 2.5) '(7.7 2.5)) резултат 6.700000

(distance '(1.0 2.0) '(3.0 4.0)) резултат 2.828427

1.39 (eq <израз1> <израз2>)

Тази функция проверява дали <израз1> е идентичен с <израз2>, т.е. дали са присвоени на един и същи обект (напр.със SETQ). EQ дава като резултат T, ако двата израза са идентични, в противен случай - Nil. Тази функция е въведена преди всичко за да установи, дали при 2 списъка се касае за идентични списъци.

Пример:

(setq f1 '(a b c))

(setq f2 '(a b c))

(setq f3 f2)

От това следва:

(eq f1 f3) резултат Nil (f1 и f3 не са идентични)

(eq f3 f2) резултат T (f3 и f2 са един и същ списък)

1.40 (equal <израз1> <израз2>)

Тази функция проверява дали <израз1> е еднакъв с <израз2>, т.е. дали резултатите от оценяването са еднакви.

Пример:

(setq f1 '(a b c))

(setq f2 '(a b c))

(setq f3 f2)

От това следва:

(equal f1 f3) резултат T (стойн.на f1 и f3 са еднакви)

(equal f3 f2) резултат T (f3 и f2 са един и същ списък)

1.41 (eval <израз>)

Тази функция дава резултата от оценяването на <израз>, при което <израз> може да е някакъв AutoLISP-израз.

Пример:

(setq a 123)

(setq b ‘a)

От това следва:

(eval 4.0) резултат 4.000000

(eval (abs -10)) резултат 10

(eval a) резултат 123

(eval b) резултат 123

1.42 (exp <число>)

Тази функция дава "***e***" с експонента <число> (естествен логаритъм). Резултатът е реално число.

Пример:

(exp 1.0) резултат 2.718282

(exp 2.2) резултат 9.025013

(exp -0.4) резултат 0.670320

1.43 (expt <база> <експонента>)

Тази функция дава <база> със специфицираната <експонента>. Ако двата аргумента са цели числа, резултатът е цяло число, иначе - реално.

Пример:

(expt 2 4) резултат 16

(expt 3.0 2.0) резултат 9.000000

1.44 (fix <число>)

Тази функция дава като резултат преобразуваното в цяло число <число>. <ЧИсло> може да е реално или цяло. Реалните числа се закръгляват, като позициите след десетичната точка се игнорират.

Пример:

(fix 3) резултат 3

(fix 3.7) резултат 3.000000

1.45 (float <число>)

Тази функция дава като резултат преобразуваното в реално число <число>. <Число> може да е реално или цяло.

Пример:

(float 3) резултат 3.000000

(float 3.7) резултат 3.700000

1.46 (foreach <име> <списък> <израз> ...)

Тази функция проверява <списък>, дава на всеки елемент <име> и оценява всеки <израз> за всеки елемент от списъка. Могат да се специфицират произволно много изрази. FOREACH дава резултата от последния оценен израз.

Пример:

(foreach n '(a b c) (print n))

съответства на:

(print a)

(print b)

(print c)

1.47 (gcd <число1> <число2>)

Тази функция дава като резултат най-големия общ делител на <число1> и <число2>. Двете числа трябва да са цели.

Пример:

(gcd 81 57) резултат 3

(gcd 12 20) резултат 4

1.48 (getangle [<точка>] [<въпрос>])

Тази функция очаква въвеждането на ъгъл от потребителя. <Въпрос> е текст, който се указва по желание, а <точка> е 2D-основна точка, която също се избира по желание. Ъгълът може да се специфицира чрез въвеждане на един от форматите за ъглови мерни единици на AutoCAD. Тази функция винаги дава ъгъла в радиани, независимо от това, че актуалният формат за изобразяване на ъгъл в AutoCAD е в градуси, гради или геодезически единици.

Ъгълът може да се покаже на графичния екран с две 2D-точки. След това ъгълът се визуализира динамично на екрана. Избираемият аргумент <точка> се приема като първата от тези 2 точки, чрез което можете да специфицирате ъгъла с "показване" на другата точка.

На въпроса на GETANGLE не може да се въведе AutoLISP-израз. В противен случай се появява съобщение "Нямам достъп до AutoLISP".(Виж също GETORIENT и INITGET.)

Пример:

(setq win (getangle))

(setq win (getangle '(1.0 3.5)))

(setq win (getangle "В каква посока? "))

(setq win (getangle '(1.0 3.5) "В каква посока? "))

1.49 (getcorner <точка> [<въпрос>])

Тази функция дава като резултат точка, както GETPOINT. Обаче GETCORNER има нужда от базова точка като аргумент и чертае правоъгълник с базовата точка като изходна и следва указаното движение на потребителя на екрана. Като отговор на GETCORNER не може да се въведе друг AutoLISP-израз. (Виж също GETPOINT и INITGET.)

1.50 (getdist [<точка>] [<въпрос>])

Тази функция очаква въвеждането на разстояние от потребителя. <Въпрос> е произволен текст, който се указва по желание, а <точка> е базова точка, която също се избира по желание. Разстоянието може да се зададе чрез въвеждане на един от действащите формати на AutoCAD за разстояние. Тази функция винаги дава разстоянието като реално число, независимо от вида на въведените данни.

Разстоянието може да се покаже на графичния екран с две точки. Визуализира се динамично на екрана. Избираемият аргумент <точка> се приема като първата от тези 2 точки, чрез което можете да специфицирате втората точка с "показване".

На въпроса на GETDIST не може да се отговори с друг AutoLISP-израз. (Виж също INITGET.)

Пример:

```
(setq abst (getdist))
(setq abst (getdist '(1.0 3.5)))
(setq abst (getdist "В каква посока? "))
(setq abst (getdist '(1.0 3.5) "В каква посока? "))
```

1.51 (getint [<въпрос>])

Тази функция очаква въвеждане на цяло число. След това дава цяло число. <Въпрос> е текст по желание. На въпроса на GETINT не може да се отговори с друг AutoLISP-израз. (Виж също INITGET.)

Пример:

```
(setq ZZ (getint))
(setq ZZ (getint "Задайте число: "))
```

1.52 (getkword [<въпрос>])

Функцията GETKWORD пита потребителя за ключова дума. Списъкът на действащите ключови думи се определя преди извикването на GETKWORD с новата функция INITGET (описана по-долу). GETKWORD връща ключова дума, която съответства на данните от потребителя, като символен низ. AutoCAD стартира нов опит, ако данните не съответстват на нито една от ключовите думи. Отговор с интервали връща Nil (ако е позволен такъв). Резултатът е Nil и когато не е дефинирана ключова дума като символен низ.

Пример:

```
(initget 1 "Y N")
(setq x (detkword "Сигурен ли сте? (Y или N)"))
```

Тази последователност пита потребителя и поставя X или на "Y", или на "N", според отговора на потребителя. В случай, че отговорът не съвпада с никоя от ключовите думи,

AutoCAD изисква от потребителя друг опит. На въпроса на GETKWORD не може да се отговори с друг AutoLISP-израз. (Виж също INITGET.)

1.53 (getorient [<точка>] [<въпрос>])

В AutoLISP ъглите се представят винаги в радиани, при което посоката на ъгъл 0 е винаги надясно и нарастването е срещу часовниковата стрелка. Може да се направи преобразуване, ако потребителя е избрал друга посока на ъгъла 0 или друга посока на нарастване. Това става с командата UNITS или със системните променливи ANGBASE и ANGDIR.

GETORIENT е подобна на GETANGLE, обаче се отнася по различен начин по посоката на ъгъла 0 и посоката на нарастване. GETANGLE трябваше да се използва, когато е необходима величина на завъртане (относителен ъгъл). Противно на това, GETORIENT Ви дава ориентиране (абсолютен ъгъл).

Да приемем, че с командата UNITS посоката на ъгъла 0 е поставена на 90 градуса (север) и посоката на нарастване е по часовниковата стрелка. Таблицата по-долу Ви показва какво връщат GETANGLE и GETORIENT (в радиани) за въведените от потребителя данни (в градуси).

Данни(градуси)	GETANGLE	GETORIENT
0	0.000000	1.570796
-90	1.570796	3.141593
180	3.141593	4.712389
90	4.712389	0.000000

Както се вижда от таблицата, GETANGLE взема под внимание посоката на нарастване на ъгъла, обаче игнорира посоката на ъгъла 0. Можете да използвате GETANGLE, за да получите стойността на завъртане при вмъкване на блок, тъй като въвеждането на 0 градуса връща винаги 0 в радиани. От друга страна GETORIENT взема под внимание посоката на ъгъла 0, както и посоката на нарастване на ъгъла. Бихте могли да използвате GETORIENT например за получаване на ъгъла на основната линия на текст. На въпроса на GETORIENT не може да се отговори с друг AutoLISP-израз. (Виж също INITGET и GETANGLE.)

1.54 (getpoint [<точка>] [<въпрос>])

Тази функция очаква въвеждане на точка. <Точка> е избираема базова точка и <въпрос> е текст, който се указва по желание, и може да се изобрази като въпрос. Точката може да се специфицира чрез показване или чрез задаване на координати в актуални мерни единици. Ако е налице избираемата базова точка <точка> (като аргумент), се появява динамична линия от тази точка към визирния кръст. GETPOINT дава като резултат точка, която е списък от 2 реални числа.

Пример:

```
(setq p (getpoint))
(setq p (getpoint "Къде? "))
(setq p (getpoint '(1.5 2.0) "Втора точка: "))
```

С GETPOINT може да се зададе 3D-точка, ако функцията INITGET се използва за поставяне на "3D-точков" управляващ флаг. На въпроса на GETPOINT не може да се отговори с друг AutoLISP-израз. (Виж също INITGET и GETCORNER.)

1.55 (getreal [<въпрос>])

Тази функция очаква въвеждане на реално число. След това дава реално число. <Въпрос> е текст по желание. На въпроса на GETREAL не може да се отговори с друг AutoLISP-израз. (Виж също INITGET.)

Пример:

(setq ZZ (getreal))

(setq ZZ (getreal "Мащабен фактор: "))

1.56 (getstring [<cr>] [<въпрос>])

Тази функция очаква въвеждане на символен низ и връща като резултат символен низ. Ако <cr> е налице и не е равно на Nil, зададеният символен низ може да съдържа интервали (и трябва в такъв случай да бъде завършен с RETURN). В противен случай символният низ може да бъде завършен с интервал или с RETURN. <Въпрос> е текст, който се указва по желание и може да се изобрази като въпрос.

Ако заданието от потребителя трябва да е една от няколко опции (ключови думи), може вместо горепосочената функция да се използва GETKWORD. На въпроса на GETSTRING не може да се отговори с друг AutoLISP-израз. (Виж също INITGET.)

Пример:

(setq s (getstring))

(setq s (getstring "Как е Вашето име? "))

1.57 (getvar <име променлива>)

Тази функция търси стойността на системна променлива от AutoCAD. Името на променливата трябва да е записано в кавички. Ако търсите стойността на системна променлива, която не е позната на AutoCAD, резултатът е Nil. (Виж също функцията SETVAR.)

Пример:

Ако последният специфициран радиус е бил 0.25 единици:

(getvar "FILLETRAD") резултат 0.250000

1.58 (graphscr)

GRAPHSCR превключва на конфигурация от единичен екран, от текстов в графичен екран (както функционалния клавиш "FLIP SCREEN" на AutoCAD. GRAPHSCR винаги дава Nil. (Виж също функцията SETVAR.)

1.59 (if <тестов израз> <израз - ако ДА> [<израз - ако НЕ>])

Тази функция оценява изрази обусловено. Ако <тестов израз> не е равен на Nil, оценява се <израз - ако ДА>, иначе - <израз - ако НЕ>. Последния израз може да се указва по желание. IF дава като резултат стойността на избрания израз. Ако <израз - ако НЕ> липсва и <тестов израз> е Nil, IF дава като резултат Nil.

Пример:

(if (= 1 3) "Yes!!""N") резултат "N"

(if (= 2 (+ 1 1)) "Yes!!") резултат "Yes!!"

(if (= 2 (+ 3 4)) "Yes!!") резултат Nil

1.60 (initget [<битове>] [<символен низ>])

Тази функция дефинира опции за следващата GETxxx-функция (изключения: GETSTRING и GETVAR). INITGET винаги връща Nil. Избираемият аргумент <битове> е цяло число със следните стойности:

INITGET-Bits	значение
1	интервали не са позволени
2	нулеви стойности не са позволени
4	отрицателни стойности не са позволени
8	границите се проверяват само ако е включен LIMCHECK
16	дава като резултат 3D-точки вместо 2D-точки
32	за Box или за динамична линия използва пунктирна линия
64	игнорира координатата Z на въведената точка

Битовете могат да се събират в произволна комбинация и да дават стойности от 1 до 127. Ако данните от потребителя не изпълняват едно или няколко специфицирани условия (напр.въвеждане на 0, ако не са позволени нулеви стойности), AutoCAD показва съобщение и изисква от потребителя нов опит.

Пример 1:

```
(initget (+ 1 2 4))
(setq A (getint "На колко години сте? "))
```

Тази поредица пита за възрастта. При отговори с 0, интервали или отрицателни стойности, въпросите биха се повторили. Ако <битове> не е даден, се приема 0 (няма условие). Специалните управляващи стойности се вземат предвид само от GETxxx-функциите, за които те са разумни. Това ще видите от долустоящата таблица.

Стойността на управляващия бит 32 се взема под внимание от функциите GETPOINT, GETCORNER, GETDIST, GETANGLE и GETORIENT, ако е дадена базова точка. Тогава тази функция използва пунктирни линии (или като повдигнати) за динамичната линия или за прозореца на курсора от указаната базова точка. Ако системната променлива POPUPS е 0 (екранът не поддържа новия потребителски интерфейс), AutoCAD игнорира този INITGET-бит.

Функция	Управляващи битове						
име	"1"	"2"	"4"	"8"	"16"	"32"	"64"
GETINIT	Y	Y	Y	-	-	-	-
GETREAL	Y	Y	Y	-	-	-	-
GETDIST	Y	Y	Y	-	-	Y	Y
GETANGLE	Y	Y	-	-	-	Y	-
GETORIENT	Y	Y	-	-	-	Y	-
GETPOINT	Y	-	-	Y	Y	Y	-
GETCORNER	Y	-	-	Y	Y	Y	-
GETKWORD	Y	-	-	-	-	-	-
GETSTRING	-	-	-	-	-	-	-
GETVAR	-	-	-	-	-	-	-

Избираемият аргумент <символен низ> на функцията INITGET дефинира списък от ключови думи, които трябва да се проверят от следващия въпрос на GETxxx, в случай, че потребителят не въведе очакваните данни (напр. Ако за GETPOINT не е въведена точка). Ако данните от потребителя съвпадат с една от ключовите думи в списъка, тази ключова дума е резултат от функцията GETxxx като символен низ. Потребителската програма проверява ключовите думи и изпълнява за всяка желаното действие. Ако данните от потребителя не са от очаквания тип и не съответстват на никоя от ключовите думи, AutoCAD изисква от потребителя да направи нов опит.

Списъкът от ключови думи трябва да притежава следната форма "КД1 КД2 КД3". Отделните ключови думи са разделени с интервал. Съкращенията се указват по желание. Има 2 начина за съкращения:

*необходимата част се пише с главни букви и остатъкът с малки;

"LTYP,LT"

**необходимата част се разделя със запетая от ключовата дума.

"Ltyp"

Тези 2 спецификации са еднакви. Двете позволяват въвеждането на "LTYP", "LTY" или "LT", но не биха приели "L" (недостатъчна спецификация), "LTFAKTOR" и "LTYPX" (не съвпадат).

Пример 2:

```
(defun getnum ( / x)
        (initget 1 "Pi Zwei-Pi")
        (setq x (getreal "Pi/Zwei-Pi/<число>: "))
        (cond ((eq x "Pi") Pi)
              ((eq x "Zwei-Pi") (* 2.0 Pi))
              (T x)
        )
```

Тази INIGET-функция предпазва от въвеждане на интервали и дефинира списък от 2 ключови думи, а именно "Pi" и "Zwei-Pi". С GETREAL се получава реално число, което извиква въпроса "Pi/Zwei-Pi/<число>:". Резултатът се записва в локалната променлива X. Когато потребителят въведе число, то се дава като резултат от GETNUM. Ако обаче потребителят зададе "Pi" като ключова дума (или просто "P"), GETPOINT ще даде като резултат ключовата дума "Pi". Функцията COND отбелязва това и в този случай връща стойността Pi. Ключовата дума "Zwei-Pi" се управлява по същия начин.

ЗАБЕЛЕЖКА: Дефинираните от INITGET управляващи флагове и списъци от ключови думи се използват при следващото извикване на GETxxx-функцията и тогава автоматично се изтриват. По този начин се спестява втора функция, която да изтрива отделните условия.

1.61 (inters <точка 1> <точка 2> <точка 3> <точка 4> [<onseg>])

Функцията INTERS проверява две линии и дава тяхната пресечна точка, или Nil, ако линиите не се пресичат. <Точка1> и <точка2> означават крайните точки на първата линия; <точка3> и <точка4> означават крайните точки на втората линия. Ако е налице избираемия аргумент <onseg> и е Nil, дължината на двете линии се приема като безкрайна. INTERS дава като резултат пресечната точка, ако тази точка се намира от страната на крайната точка на едната или двете линии. Ако аргументът <onseg> е пропуснат или не е Nil, пресечната точка трябва да се намира на двете линии, в противен случай INTERS дава Nil.

Пример:

(setq a ‘(1.0 1.0) b ‘(9.0 9.0))

(setq c ‘(4.0 1.0) d ‘(4.0 2.0))

От това следва:

(inters a b c d) резултат Nil

(inters a b c d T) резултат Nil

(inters a b c d Nil) резултат (4.000000 4.000000)

1.62 (itoa <цяло число>)

Тази функция преобразува цяло число в символен низ.

Пример:

(itoa 33) резултат "33"

(itoa -17) резултат "-17"

1.63 (lambda <аргументи> <израз> ...)

LAMBDA дефинира "анонимна" функция. Тя се използва особено когато не е оправдано новото дефиниране на функция. Също и намерението на програмиста се представя ясно по този начин. LAMBDA дава стойността на нейния последен <израз> и често се използва заедно с APPLY и/или MAPCAR за изпълнение на функция по списък.

Примери:

```
(apply ‘(lambda (x y z)
                    (* x (- y z))
        )
        ‘(5 20 14)
)                                   резултат 30
```

и още

```
(mapcar ‘(lambda (x)
                    (setq counter (1+ counter))
                    (* x 5)
         )
         ‘(2 4 -6 10.2)
)                                   резултат (10 20 -30 51.000000)
```

1.64 (last <списък>)

Тази функция дава като резултат последния елемент на <списък>. <Списък> не трябва да е Nil. LAST може да даде резултат, който е атом или списък.

Пример:

(last ‘(a b c d e)) резултат E

(last ‘(a b c (d e))) резултат (D E)

На пръв поглед изглежда, че LAST е идеалната възможност за получаване на Y-координатата на точка. Това е в сила за 2D-точки, но при 3D-точките тя дава Z-координатата. За да действат идентично Вашите функции при 2D и 3D-точките, препоръчваме Ви да използвате CADR за изчисление на Y-координати и CADDR - за изчисление на Z-координати.

1.65 (length <списък>)

Тази функция дава като резултат цяло число, което съответства на броя елементи в <списък>.

Пример:

(length ‘(a b c d)) резултат 4

(length ‘(a b (c d))) резултат 3

(length ‘()) резултат 0

1.66 (list <израз> ...)

Тази функция приема произволно много изрази, свързва ги и дава като резултат 1 списък. В AutoLISP тя често се използва за дефиниране на 2D и 3D-точкови променливи (списък от 2 или 3 реални числа).

Пример:

(list ‘a ‘b ‘c) резултат (A B C)

(list ‘a ‘(b c) ‘d) резултат (A (B C) D)

(list ‘3.9 ‘6.7) резултат (3.900000 6.700000)

1.67 (listp <елемент>)

Тази функция дава T, ако <елемент> е списък, ако не - Nil.

Пример:

(listp ‘(a b c)) резултат T

(listp ‘a) резултат Nil

(listp 4.343) резултат Nil

1.68 (load <име на файл>)

Тази функция зарежда файл с AutoLISP-изрази и ги оценява. <Име на файл> е символен низ, който е без тип на файл (автоматично ".lsp"). Пред името на файла може да се зададе указателен префикс (напр. "/Funktion/test1"). При MS-DOS / PC-DOS системите е позволена и буква за директорията и може да се използва обратна наклонена черта, вместо обикновената (обратна наклонена черта се задава в символен низ "\\").

Ако операцията протече успешно, LOAD ще даде името на последната функция, дефинирана във файла. Ако операцията е неуспешна, LOAD ще даде името на файла като символен низ.

Пример:

Ако файлът "test1.lsp" съдържа DEFUN на функцията MYFUNC:

(load "test1") резултат MYFUNC

Ако файлът "test2.lsp" не съществува:

(load "test2") резултат "test2"

Функцията LOAD не може да се извика от друга AutoLISP-функция. Тя трябва да се въведе директно от клавиатурата, от меню-файл или от команден(.scr) файл, докато не е активна друга AutoLISP-функция.

Всеки път, когато извикате чертожния редактор, AutoLISP зарежда файла "acad.lsp" (ако съществува). Този файл може да съдържа дефиниции на функции, които се оценяват автоматично в началото на редактирането.

1.69 (log <число>)

Функцията дава натурален логаритъм на <число> като реално число.

Пример:

(log 4.5) резултат 1.504077

(log 1.22) резултат 0.198850

1.70 (logand <число> <число> ...)

Тази функция дава резултата на логическото AND по битове на списък от <числа>. <Числа> трябва да са цели и резултатът също е цяло число.

Пример:

(logand 7 15 3) резултат 3

(logand 2 3 15) резултат 2

(logand 8 3 4) резултат 0

1.71 (logior <число> <число> ...)

Тази функция дава резултата на логическото OR по битове на списък от <числа>. <Числа> трябва да са цели и резултатът също е цяло число.

Пример:

(logior 1 2 3) резултат 7

(logand 9 3) резултат 11

1.72 (lsh <число1> <брой битове>)

Тази функция дава резултата от логическото преместване на битовете на <число1>. <Число1> и <брой битова> трябва да са цели числа и резултатът също е цяло число. Ако <брой битове> е положителен, <число1> се премества наляво, ако е отрицателен - надясно. И в двата случая се допълват "нулеви" битове, а изместените битове се изключват. Когато в първия бит (16-та позиция) на едно цяло число се появи единица или бъде изместена оттам - знакът се променя.

Пример:

(lsh 2 1) резултат 4

(lsh 2 -1) резултат 1

(lsh 40 2) резултат 160

(lsh 16384 1) резултат -32768

1.73 (mapcar <функция> <списък 1> ... <списък N>)

MAPCAR дава резултата от изпълнението на <функция> с отделните елементи от <списък 1> до <списък N>, които са аргументи на <функция>. Броят списъци трябва да съответства на броя аргументи на <функция>.

Пример 1:

(mapcar '1+ '(10 20 30)) резултат (11 21 31)

което съответства на:

(1+ 10)

(1+ 20)

(1+ 30)

като MAPCAR дава резултатите във вид на списък.

Пример 2:

(mapcar '+ '(10 20 30) '(4 3 2)) резултат (14 23 32)

което съответства на:

(+ 10 4)

(+ 20 3)

(+ 30 2)

Функцията LAMBDA може да специфицира анонимна функция, която може да се изпълни от MAPCAR. Това е особено практично, когато някои аргументи на функцията са постоянни или са предоставени по друг начин.

Пример 3:

(mapcar '(lambda (x) (+ x 3)) '(10 20 30)) рез.(13 23 33)

и:

(mapcar '(lambda (x y z) (* x (- y z)))

'(5 6) '(20 30) '(14 5.0)) резултат (30 150.000000)

1.74 (max <число> <число> ...)

Тази функция дава като резултат най-голямото от зададените <число>. Всяко <число> може да е реално или цяло.

Пример:

(max 4.07 -144) резултат 4.070000

(max -88 19 5 2) резултат 19

1.75 (member <израз> <списък>)

Тази функция претърсва <списък> за съответен <израз> и като резултат дава остатъка от <списък>, започващ с първия <израз>. Ако в <списък> не е намерен <израз>, MEMBER дава Nil.

Пример:

(member ‘c ‘(a b c d e)) резултат (C D E)

(member ‘q ‘(a b c d e)) резултат Nil

1.76 (menucmd <символен низ>)

Тази функция позволява на AutoLISP-програмите превключване между областите на менютата на AutoCAD. Поради това една AutoLISP-програма може да работи с асоцииран меню-файл, като като съответното подменю се изобразява при всякакви задания на потребителя. Аргументите <символен низ> имат формата:

Секция = Подменю

при което:

Секцията специфицира меню-секцията.

Валидни имена (типове на менюта) са:

S	екранно меню
K	бутонно меню
B	картинно меню
A1 - A10	горни екранни менюта от 1 до 10
T1 - T4	таблетни менюта от 1 до 4
Z1	AUX-менюта

Подменю специфицира активираното подменю. Името трябва да съответства или на едно от имената на подменютата (без "**") в настоящия зареден меню-файл, или на името на един от типовете менюта, дефинирани по-горе.

Стоящия отпред знак "$", който е необходим за информация за подменютата в меню-файла, тук не се появява.

Пример 1:

(menucmd "S=OFANG")

Така се извиква подменюто "OFANG" на екрана (ако такова подменю съществува в активния меню-файл).

Пример 2:

(menucmd "K=MY-BUTTON")

Така на бутонното меню се присвоява подменюто "MY-BUTTON".

За картинните и горните екранни менюта - "*" е валидно име на подменю и предизвиква изобразяването на подменюто, което е присвоено понастоящем на специфицираната меню-секция.

Пример 3:

(menucmd "A1=ЧИСЛОВО")

(menucmd "A1=*")

Тази последователност от команди би присвоила на горното картинно меню 1 - подменюто "ЧИСЛОВО" и непосредствено след това би го изобразила на екрана.

1.77 (min <число> <число> ...)

Тази функция дава като резултат най-малкото от зададените <число>. Всяко <число> може да е реално или цяло.

Пример:

(min 683 -10.0) резултат -10.000000

(min 73 2 48 5) резултат 2

1.78 (minusp <елемент>)

Тази функция дава T, ако <елемент> е реално или цяло число и се оценява като отрицателна стойност, ако не - Nil. Тази функция не е дефинирана за други типове <елемент>.

Пример:

(minusp -1) резултат T

(minusp -4.293) резултат T

(minusp 830.2) резултат Nil

1.79 (not <елемент>)

Тази функция дава T, ако изразът е Nil, в противен случай дава Nil. Функцията NULL често се използва за списъци, а NOT - за други типове данни, заедно с някаква управляваща функция.

Пример:

(setq a 123)

(setq b "Text")

(setq c Nil)

От това следва:

(not a) резултат Nil

(not b) резултат Nil

(not c) резултат T

(not '()) резултат T

1.80 (nth <N> <списък>)

Тази функция дава като резултат "N-тия" елемент на <списък>, при което <N> е позицията на елемента, който е резултат (първият елемент Nil). Ако <N> е по-голямо от максималния брой елементи на <списък>, резултатът е Nil.

Пример:

(nth 3 ‘(a b c d e)) резултат D

(nth 0 ‘(a b c d e)) резултат A

(nth 5 ‘(a b c d e)) резултат Nil

1.81 (null <елемент>)

Тази функция дава като резултат T, ако <елемент> е присвоен на Nil, в противен случай резултатът е Nil.

Пример:

(setq a 123)

(setq b "Text")

(setq c Nil)

От това следва:

(null a) резултат Nil

(null b) резултат Nil

(null c) резултат T

(null ‘()) резултат T

1.82 (numberp <елемент>)

Тази функция дава като резултат T, ако <елемент> е реално или цяло число, в противен случай - Nil.

Пример:

(setq a 123)

(setq b ‘a)

От това следва:

(numberp 4) резултат T

(numberp 3.8348) резултат T

(numberp "Juhui") резултат Nil

(numberp ‘a) резултат Nil

(numberp a) резултат T

(numberp b) резултат Nil

(numberp (eval b)) резултат T

1.83 (open <име на файл> <режим>)

Тази функция отваря файл, до който I/O-функциите на AutoLISP имат достъп. Функцията дава като резултат файлов дескриптор, който може да се използва от другите I/O-функции. Затова трябва да ѝ се присвои една променлива със SETQ. <Име на файл> е текст, който определя името и типа на отворения файл. <Режим> е read/write флаг. Той трябва да е символен низ с една малка буква. Валидните букви за <режим> са следните:

OPEN-режим	О п и с а н и е .

"r" Отворен за четене ("read"). Ако <име на файл> не съществува, резултатът е Nil.

"w" Отворен за писане ("write"). Ако <име на файл> не съществува, създава се и се отваря нов файл. Ако <име на файл> вече съществува, съдържащите се в него данни се презаписват.

"a" Отворен за добавяне ("append"). Ако <име на файл> не съществува, създава се и се отваря нов файл. Ако <име на файл> вече съществува, той се отваря и се прибавя към края на вече съществуващите данни, така че данни, които са нови, въведени във файла, автоматично се добавят към вече съществуващите.

ЗАБЕЛЕЖКА: При DOS системите някои програми и текстови редактори пишат текстовите файлове със знак за край на файл в края на текста (CTRL+Z с десетичен ASCII код 26). При четене на текстов файл, DOS дава съобщение за край на файл, ако е намерил CTRL+Z, дори когато следват други данни. Следователно, когато използвате <режим> "a" на функцията OPEN за добавяне на данни към файлове, които са създадени от други програми, подсигурете се най-напред, че тази програма не вмъква CTRL+Z в края на текстовия файл.

Пример 1:

Да приемем, че имената на файловете в примера на съществуват

(setq f (open "new.tst" "w")) резултат <file #nnnn>

(setq f (open "kk.dat" "r")) резултат Nil

(setq f (open "logfile" "a")) резултат <file #nnnn>

На <име на файл> може да бъде зададен указващ префикс, като напр. "/test/func3". При MS-DOS / PC-DOS системите е позволена също буква за директорията и може да се използва обратна наклонена черта, вместо обикновената (обратна наклонена черта се задава в символен низ "\\").

Пример 2:

(setq f (open "/x/new.tst" "w")) резултат <file #nnnn>

(setq f (open "kk.dat" "r")) резултат Nil

1.84 (or <израз> ...)

Тази функция дава като резултат логическото OR на списък от изрази. OR оценява изразите отляво надясно и търси <израз>, различен от Nil. Когато бъде намерен такъв, OR прекъсва процеса на оценяване и дава T. Ако всички изрази са Nil, OR дава Nil.

Пример:

(or Nil 45 '()) резултат T

(or Nil '()) резултат Nil

1.85 (osnap <точка> <режим за текст>)

Тази функция дава точка (списък от 2 реални числа), която се получава от прилагането на режима на захват (описано чрез <режим за текст>) върху <точка>. <Режим за текст> описва един или повече от валидните режими на захват на обект, например средна точка, център и т.н., разделени със запетая.

Пример:

(setq pt2 (osnap pt1 "mid"))

(setq pt2 (osnap pt1 "mid,end,cen"))

Ако <точка> е 2D-точка, резултатът е 2D-точка. Ако <точка> е 3D-точка, резултатът е 3D-точка. Ако за <точка> не е намерена точка със захвата, която да съвпада със специфицирания <режим>, резултат - Nil.

1.86 (pi)

Това не е функция, а константата Pi. Тя се оценява на закръглената стойност 3.1415926.

1.87 (polar <точка> <ъгъл> <разстояние>)

Тази функция дава точката в <ъгъл> и <разстояние> от <точка>. Точката е списък от 2 реални числа, а <ъгъл> се въвежда в радиани.

Пример:

(polar ‘(1.0 1.0) 0.785398 1.414214) рез.(2.000000 2.000000)

1.88 (prin1 <израз> [<променлива, указваща файл>])

Тази функция изобразява <израз> на екрана и дава като резултат <израз>. <Израз> може да е произволен и не е задължително да е символен низ. Ако е налице <променлива, указваща файл> и се касае за файл, отворен за писане, <израз> се записва в този файл точно така, както се появява на екрана. Записва се само <израз> (без знак за нов ред или интервал).

Пример 1:

(setq a 123)

(setq b ‘a)

От това следва:

(prin1 ‘a) изобразява A и резултат A

(prin1 a) изобразява 123 и резултат 123

(prin1 b) изобразява (A) и резултат (A)

(prin1 "Hallo") изобразява "Hallo" и резултат "Hallo"

Всеки от горните примери се изобразява на екрана, тъй като не е специфицирана <променлива, указваща файл>. Ако **f** е валидна променлива, указваща отворен файл: (prin1 "Hallo" f), то "Hallo" се записва в определения файл и резултатът е "Hallo".

Ако <израз> съдържа управляващи символи, PRIN1 ги редактира с "\" пред тях:

\e	за Escape
\n	за знак на нов ред
\r	за Return
\t	за Tabulator
\nnn	за символа, чиито осмичен код е nnn

Пример 2:

(prin1 (chr 2)) изобразява "\002" и резултат "\002"

(prin1 (chr 10)) изобразява "\n" и резултат "\n"

PRIN1 може да се използва без аргументи и дава символ, чието име е нулев символен низ. Ако използвате PRIN1 без аргументи като последен израз в дефинирана от потребителя функция, след приключване на функцията се изобразява само празен ред.

Пример 3:

```
(defun C:AUF ()
        (setvar "Lunits" 4)
        (setvar "BLIPMODE" 0)
        (prin1)
)
```

Тогава, след задействане на дефинираната функция, ще се изпълнят зададените действия и без да се изобразяват други съобщения, се появява "Command:".

1.89 (princ <израз> [<променлива, указваща файл>])

Тази функция е еквивалентна на PRIN1, с изключение на това, че управляващите символи в <израз> се изобразяват без разширение. В общи линии PRIN1 изобразява изрази, съвместими с LOAD, докато PRINC изобразява онези, които могат да се четат от функции като READ-LINE.

1.90 (print <израз> [<променлива, указваща файл>])

Тази функция е еквивалентна на PRIN1, с изключение на това, че пред <израз> се пише знак за нов ред и след <израз> - празен ред.

1.91 (progn <израз> ...)

Тази функция оценява всеки <израз> по ред и дава стойността на последния израз. С PROGN могат да се оценяват повече изрази, особено при функции, определящи само един израз.

Пример:

```
(if (= a b) (progn  (setq a (+ a 10))
                                (setq b (- b 10)))
)
```

Функцията IF оценява в общия случай 1 израз-следствие, ако тестовия израз има стойност, различна от Nil. В горния пример сме оценили 2 израза с PROGN.

1.92 (prompt <символен низ>)

Тази функция изобразява <символен низ> в областта за въпроси на екрана. При двуекранни системи PROMPT изобразява <символен низ> на двата екрана и затова за предпочитане е PRINC.

Пример:

(prompt "Стойност:") резултат "Стойност:" и връща Nil

1.93 (quote <израз>)

Дава като резултат неоценения <израз>. Това би могло да се запише и така: ‘израз.

Пример:

(quote a) резултат A

(quote cat) резултат CAT

(quote (a b)) резултат (A B)

‘a резултат A

‘cat резултат CAT

‘(a b) резултат (A B)

Последните 3 примера не функционират, ако се набират директно от клавиатурата като отговор на "Command:" в AutoCAD. Не забравяйте, че такива данни трябва да започват с "(" или "!", за да се отбележат като AutoLISP-изрази.

1.94 (read <символен низ>)

Тази функция дава като резултат първия списък или атом, който се получава от <символен низ>. <Символен низ> не трябва да съдържа интервали.

Пример:

(read "Hallo") резултат HALLO

(read "Bye") резултат BYE

(read "(a)") резултат (A)

1.95 (read-char [<променлива, указваща файл>])

Тази функция чете само 1 символ от буфера за въвеждане от клавиатурата или от отворения файл, който е описан чрез <променлива, указваща файл>. Резултатът е цяло число, съответстващо на ASCII-кода на прочетения символ.

Ако не е специфицирана <променлива, указваща файл> и в буфера за въвеждане от клавиатурата няма символи, READ-CHAR очаква, че ще въведете данни от клавиатурата, последвани от RETURN. Да приемем, че този буфер е празен. READ-CHAR чака въвеждане на данни. Ако наберете "ABC", последвано от RETURN, READ-CHAR ще даде резултат 65 (ASCII-кода на буквата "A"). Следващите 3 извиквания на READ-CHAR дават съответно 66, 67 и 10 (нов ред). Ако още веднъж се извика READ-CHAR, тя ще очаква въвеждане на данни от потребителя.

1.96 (read-line [<променлива, указваща файл>])

Тази функция чете символен низ от клавиатурата или от отворен файл, който е описан от <променлива, указваща файл>. Ако е намерен края на файла, READ-LINE дава резултат Nil, в противен случай дава прочетения символен низ. Да приемем, че F е променлива, указваща отворен файл:

(read-line f) , резултатът би бил следващия ред за данни от файла или Nil, ако е намерен края на файла.

1.97 (redraw [<име на елемент> [<режим>]])

Ефектът на функцията зависи от броя на зададените аргументи.

(Redraw)	изчертава отново целия чертеж, като REDRAW на AutoCAD
(redraw <име на елемент>)	изчертава отново избрания елемент. Така един елемент може да бъде идентифициран, след използване на GRCLEAR (изтриване на екрана).
(redraw <име на елемент> <режим>)	осигурява пълен контрол на новото изчертаване на елементи. <Режим> е цяло число с една от следните стойности:

Режим-REDRAW	Д е й с т в и е .
1	Отново изчертава елемент на екрана
2	Изтрива елемент от екрана
3	Маркира елемент (ако се поддържа от дисплея)
4	Изключва маркирането

Ако <име на елемент> е Header на сложен елемент (полилиния или блок с атрибути), главният елемент и всички негови поделементи се преработват, ако <режим> е положителен. Ако последният е отрицателен, REDRAW обработва само Header-елемента.

REDRAW винаги дава Nil, ако не са намерени грешки.

1.98 (rem <число 1> <число 2> ...)

Тази функция разделя <число 1> от <число 2> и дава резултата (<число 1> MOD <число 2>). REM се използва с цели или реални числа.

Пример:

(rem 42 12) резултат 6

(rem 12.0 16) резултат 12.000000

(rem 60 3) резултат 0

1.99 (repeat <число> <израз> ...)

При тази функция <число> е цяло положително. Функцията оценява всеки <израз> всеки път, когато се укаже от <число> и резултатът е стойността на последния <израз>.

Пример:

(setq a 10)

(setq b 100)

От това следва:

(repeat 4 (setq a (+ a 10))

(setq b (+ b 10))

) резултат 140

1.100 (reverse <списък>)

Тази функция дава списък, чиито елементи са обърнати спрямо <списък>. Пример:

(reverse '((a) b c)) резултат (C B (A))

1.101 (rtos <число> [<режим> [<точност>]])

Тази функция дава като резултат символен низ, който представя <число>, съответно на стойностите, поставени от <режим>, <точност> и оразмерителната променлива на AutoCAD - DIMZIN. <Режим> и <точност> са цели числа, които определят режима на линейни единици и точността. Поддържани стойности на <режим>:

RTOS-режим	Формат на редактиране
1	Научен (десетични числа с експонента)
2	Десетичен (десетични числа)
3	Инженерен (инчове с десетична дробна част)
4	Архитектурен (инчове с дроби)
5	Дробен

<Режим> и <точност> съответстват на системните променливи на AutoCAD - LUNITS и LUPREC. Ако се пропуснат аргументите, използват се актуалните стойности на двете променливи.

Пример:

(rtos 17.5 1 4) резултат "1.7500E+01"

(rtos 17.5 2 2) резултат "17.50"

(rtos 17.5 3 2) резултат "1′-5.50""

(rtos 17.5 4 2) резултат "1′-5½""

(rtos 17.5 5 2) резултат "17½"

1.102 (set <символ> <израз>)

Присвоява на <израз> стойността на <символ>, при което <символ> е име на символ с quote и дава като резултат тази стойност.

Пример 1:

(setq 'a 5.0) резултат 5.000000 и дефинира символа A

(set (quote b) 'a) резултат A и дефинира символа B

Ако SET се използва с име на символ без quote, функцията може да присвои нова стойност индиректно на друг символ.

Пример 2:

(set b 640) резултат 640

Тази операция присвоява на символа A стойност 640, защото се съдържа в символа B.(Виж също SETQ.)

1.103 (setq <символ1> <израз1> [<символ2> <израз2>] ...)

Присвоява на <символ1> стойността на <израз1>, на <символ2> стойността на <израз2> и т.н. Това е основната присвояваща функция на AutoLISP. Дава се последния <израз>.

Пример 1:

(setq a 5.0) резултат 5.000000

(setq b 123 c 4.7) резултат 4.700000

(setq s "es") резултат "es"

(setq x ‘(a b)) резултат (A B)

Функциите SET и SETQ създават или модифицират глобални символи, с изключение на случаите, когато се използват вътре в DEFUN, за да присвоят стойност на аргумент на функцията или символ, който се използва локално за тази DEFUN.

Пример 2:

```
(setq glo1 123)              създава глобален символ
(defun demo (arg1 /loc1)
     (setq arg1 234)         присвоява локално нова стойност
     (setq loc1 345)         присвоява локално нова стойност
     (setq glo1 456)         присвоява глобално нова стойност
     (setq glo2 567)         създава глобален символ
)
```

Глобалните символи могат да се задействат и модифицират от всяка функция и да се използват в произволен израз. Локалните символи и аргументи на функцията притежават значение само по време на оценяването на функцията, от която са дефинирани или по време на функции, извикани от тази функция. ВНИМАВАЙТЕ, аргументите на функцията могат да се използват като локални символи; функцията може да променя стойностите им до прекъсването ú, след което тези промени не се запазват.

SET и SETQ могат да присвояват нови стойности на вградени символи и имена на функции, при което първоначалните стойности се изтриват или се правят недостъпни. Дори на опитни потребители могат да се случат следните грешки:

Г Р Е Ш К И:

```
(setq angle (...))
(setq length (...))
(setq max (...))
(setq t (...))
```

За да се избегнат необичайни грешки, бъдете внимателни при избора на имена на собствените Ви символи. Никога не използвайте името на системен символ или функция за Вашия собствен символ! За да получите списък на имената на символите, които не трябва да използвате, наберете "!ATOMLIST" след "Command:" в AutoCAD, но преди да е заредена някаква AutoLISP-функция.

1.104 (setvar <име променлива> <стойност>)

Тази функция присвоява на системна променлива на AutoCAD <стойност> и резултатът е тази стойност. <Име на променлива> трябва да е в кавички.

Пример:

(setvar "FILLETRAD" 0.50) резултат 0.500000

Някои команди на AutoCAD приемат стойностите на системните променливи преди да са извикани някакви запитвания. Ако използвате SETVAR за поставяне на нова стойност по време, когато е активна една команда, може да стане така, че новата стойност да се взема предвид едва при следващото изпълнение на командата. (Виж също GETVAR.)

1.105 (sin <ъгъл>)

Тази функция дава синуса на <ъгъл> (в радиани).

Пример:

(sin 1.0) резултат 0.841471

(sin 0.0) резултат 1.414214

1.106 (sqrt <число>)

Тази функция дава квадратен корен на <число> като реално число.

Пример:

(sqrt 4) резултат 2.000000

(sqrt 2.0) резултат 1.414214

1.107 (strcase <символен низ> [<как?>])

STRCASE взема <символен низ> и дава като резултат копие, при което всички буквени символи се преобразуват съгласно <как?> в малки или главни букви. Ако <как?> е пропуснат или е оценен на Nil, всички съдържащи се в <символен низ> буквени символи се преобразуват в главни букви. Ако <как?> е зададен и не е равен на Nil, всички съдържащи се в <символен низ> буквени символи се преобразуват в малки букви.

Пример:

(strcase "Muster") резултат "MUSTER"

(strcase "Muster" T) резултат "muster"

1.108 (strcat <символен низ 1> <символен низ 2> ...)

Тази функция дава като резултат символен низ, който се състои от свързаните < символен низ 1> и < символен низ 2>.

Пример:

(strcat "cam" "era") резултат "camera"

(strcat "a" "b" "c") резултат "abc"

(strcat "a" "" "c") резултат "ac"

1.109 (strlen <символен низ>)

Тази функция дава дължината на <символен низ> в символи като цяло число.

Пример:

(strlen "abcd") резултат 4

(strlen "ab") резултат 2

(strlen "") резултат 0

1.110 (subst <нов елемент> <стар елемент> <списък>)

Тази функция претърсва <списък> за <стар елемент> и дава копие на <списък> с <нов елемент> за всеки намерен <стар елемент>. Ако в <списък> не е намерен <стар елемент>, SUBST дава непроменен <списък>.

Пример 1:

(setq Muster ‘(a b (c d) b))

От това следва:

(subst ‘qq ‘b Muster) резултат (A QQ (C D) QQ)

(subst ‘qq ‘z Muster) резултат (A B (C D) B)

(subst ‘qq ‘(c d) Muster) резултат (A B QQ B)

(subst ‘(qq rr) ‘(c d) Muster) резултат(A B (QQ RR) QQ)

(subst ‘(qq rr) ‘z Muster) резултат (A B (C D) B)

Като се прилага заедно с ASSOC, SUBST позволява да се замени стойността, която е присвоена на наименование от в асоциативен списък.

Пример 2:

(setq wer ‘((vorname karl) (bez der) (name kahle)))

От това следва:

(setq alt (assoc ‘vorname wer)) резултат (VORNAME KARL)

(setq neu ‘(erster k)) резултат (VORNAME K)

(subst neu alt wer) резул.((VORNAME K)(BEZ DER)(NAME KAHLE))

1.111 (substr <символен низ> <старт> [<дължина>])

Тази функция дава част от <символен низ>, започваща от <старт>-символа и е толкова знака, колкото са определени от <дължина>. Ако <дължина> не е определена, се връща останалото от <символен низ> до края. <Старт> и <дължина> трябва да са положителни числа. Първият знак от <символен низ> съответства на номер 1.

Пример:

(substr "abcde" 2) резултат "bcde"

(substr "abcde" 2 1) резултат "b"

(substr "abcde" 3 2) резултат "cd"

1.112 (terpri)

Тази функция пише нов ред на екрана. Като резултат се дава нов ред. TERPRI не се използва за I/O-файл. За да напишете нов ред във файл, можете да използвате PRINT или PRINC.

1.113 (taxtscr)

Тази функция превключва графичния монитор в текстов при системи с един екран (като клавиша "FLIP SCREEN" при AutoCAD). TEXTSCR винаги дава Nil. (Виж също GRAPHSCR.)

1.114 (trace <функция> ...)

Тази функция е в помощ при търсене на грешки. Тя маркира специфицираните <функции> и дава името на последната функция. Всеки път, когато се оценява <функция>, се появява индикация, която показва задаването на функцията (в областта за извикване на команди). След това на екрана се появява резултата от функцията.

Пример:

(trace my-func) резултат MY-FUNC и поставя маркирането на функцията. (Виж също UNTRACE.)

1.115 (type <елемент>)

Тази функция дава типа (TYPE) на <елемент>, при което TYPE е един от следните типове (като атом):

REAL	Числа с плаваща запетая
FILE	Файлови дескриптори (променливи, указващи файлове)
STR	Символни низове
INT	Цели числа
SYM	Символи
LIST	Списъци и потребителски функции
SUBR	Вътрешни AutoLISP-функции
PICKSET	AutoCAD-изборни записи
ENAME	AutoCAD-имена на елементи
PAGETB	Funktionspaginierungstabelle

Пример 1:

(setq a 123 r 3.45 s "Hallo" x ‘(a b c))

(setq f (open "Name" "r"))

От това следва:

(type ‘a) резултат SYM

(type a) резултат INT

(type f) резултат FILE

(type r) резултат REAL

(type s) резултат STR

(type x) резултат LIST

(type +) резултат SUBR

Пример 2:

```
(defun isint (a)
        (if (= (type a) 'INT)    дали типът е цяло число
                        T                       ако ДА - дава T
                        Nil                     ако НЕ - дава Nil
        )
)
```

1.116 (untrace <функция> ...)

Тази функция изтрива маркирането на специфицираните <функции> и дава името на последната функция. По този начин може да се изключи чрез избор възможността за търсене на грешки.

Пример:

(untrace my-func) резултат MY-FUNC и изтрива маркирането на функцията. (Виж също TRACE.)

1.117 (ver)

Тази функция дава символен низ, който съдържа номера на актуалната версия на AutoCAD. Може да се използва с EQUAL за проверка на програмната съвместимост.

Пример:

(ver) резултат "AutoLISP Releasse 9.0”

1.118 (while <тестов израз> <израз> ...)

Тази функция оценява <тестов израз> и (ако не е равен на Nil) оценява другите изрази и още веднъж <тестов израз>. Този процес продължава, докато <тестов израз> е Nil. WHILE дава като резултат последната стойност на последния <израз>.

Пример:

(setq a 1)

От това следва:

```
(while (<= a 10)
                (new-func a)
                (setq a (1+ a)
)
```

Потребителската функция NEW-FUNC ще бъде извикана 10 пъти, при което "a" ще получи стойности от 1 до 10. Резултатът, който ще се появи е 11, т.е. стойността на последния оценен <израз>.

1.119 (write-char <число> [<променлива, указваща файл>])

Тази функция пише символ на екрана или в отворен файл, който е описан чрез <променлива, указваща файл>. Число е ASCII-кода на символа и също е стойността, която се дава като резултат от функцията.

Пример 1:

(write-char 67) дава резултат 67 и пише "C" на екрана.

(write-char 67 f) резултат 67 и пише "C" във файл F.

1.120 (write-line <символен низ> [<променлива, указваща файл>])

Тази функция пише <символен низ> на екрана или в отворен файл, който е описан чрез <променлива, указваща файл>. <Символен низ> се дава обикновено с quote като резултат, при което се пропускат кавичките при записването във файла.

Пример:

(write-line "Test" f) резултат "Test" и записва Test

1.121 (zerop <елемент>)

Тази функция дава T, ако <елемент> е реално или цяло число и е оценено на 0, в противен случай се дава Nil. Тя не е дефинирана за други типове <елементи>.

Пример:

(zerop 0) резултат T

(zerop 0.0) резултат T

(zerop 0.0001) резултат Nil

1.122 (*error* <символен низ>)

Това е функция за отстраняване на грешки, която може да бъде дефинирана от потребителя. Ако не е равна на 0, тя се изпълнява като функция, докато се появи AutoLISP-грешка. Единственият аргумент съдържа текстово описание на грешката.

Пример:

```
(defun *error* (st)
                    (princ "Error: ")
                    (princ st)
                    (terpri)
)
```

Тази функция действа точно като стандартното търсене на грешки на AutoLISP. (За съжаление в превода липсва описание на грешките!!!)

Глава 2

Достъп до елементи и устройства

Голям избор от AutoLISP-функции позволява достъпа до елементи на AutoCAD, както и до графичния монитор и входните устройства. Можете да изберете елементи, чиито стойности се извикват и променят. LISP-овите променливи могат да управляват множества от избрани елементи, за да могат да се обработват групи елементи. Не са предоставени функции, които директно създават елементи, тъй като функцията COMMAND може да задейства и изпълнява обичайните за тази цел команди на AutoCAD.

Специални типове данни

Реализирани са два специални типа данни, които позволяват достъпа до елементите на AutoCAD: ИМЕ НА ЕЛЕМЕНТ и МНОЖЕСТВО ОТ ИЗБРАНИ ОБЕКТИ. Тези типове данни могат да бъдат манипулирани само чрез директно присвоени на тях функции и тяхната вътрешна структура е несъществена за програмиста на AutoLISP. ИМЕТО НА ЕЛЕМЕНТА показва все едно точка в управляван от графичния редактор на AutoCAD файл, в който AutoLISP може да намери информацията от базата данни и векторите (в случай, че тези се намират на екрана) на елемента. МНОЖЕСТВОТО ОТ ИЗБРАНИ ОБЕКТИ е просто група от имена на елементи.

Имената на елементи и множествата от избрани обекти са валидни само за действащото състояние на редактиране, в което те са били създадени от AutoCAD.

Функции за манипулация с множеството избрани обекти

2.2.1 (ssget [<режим>] [<точка1> [<точка2>]])

С функцията SSGET можете да образувате множество от избрани обекти. <Режим> може да се указва по желание и е символен низ, който определя типа на избора на обект. Може да се касае за "З", "С", "П" и "Н", аналогично на режимите за избор в AutoCAD - "Прозорец", "Пресичащ", "Последен" и "Набор". <Точка1> и <точка2> са списъци от точки, които указват основните точки на множеството от обекти. Ако една точка е специфицирана без <режим>, този избор на елемент става с посочване на точка. Ако се пропуснат всички аргументи, SSGET задава на потребителя въпроса на AutoCAD "Select objects:", чрез което интерактивно може да се създаде множеството от избрани елементи.

Пример 1:

(ssget)	Изисква от потребителя произволен избор на обекти
(ssget "V")	Избира последния избран запис
(ssget "L")	Избира елемент, който последен е вмъкнат в базата данни
(ssget '(2 2))	Избира елемент, преминаващи през точка 2,2

(ssget "F" ‘(0 0) ‘(5 5))	Избира елементи, намиращи се изцяло в прозореца, дефиниран чрез 0,0 и 5,5
(ssget "K" ‘(0 0) ‘(1 1))	Избира елементи, които се намират в или пресичат прозорец, дефиниран с 0,0 и 1,1
(ssget "X" <списък-филтър>)	Избира елементи, съдържащи се в <списък-филтър>

Избраните елементи се маркират само, ако SSGET се използва без аргументи. Не се записва информация за това с какъв метод е избран един елемент (вижте обаче ENTSEL за една алтернативна възможност). Множествата от избрани обекти използват временно файлово пространство на AutoCAD, затова AutoLISP не може да поддържа отворени повече от 6 множества едновременно. Ако се премине тази граница, AutoCAD отказва създаването на други множества от избрани обекти и дава Nil на въпросите на SSGET.

Опциите за избор на обекти могат да бъдат въвеждани след въпроса на AutoCAD "Select objects:" и то тогава, когато е възможен избор с "Последен". По този начин се избират всички обекти в LISP-множеството на избраните обекти.

(ssget "X" <списък-филтър)

Към функцията SSGET на AutoLISP беше допълнен новия режим "X", който прочита целия чертеж и създава едно множество от избрани обекти. Това множество съдържа имената на всички обекти, които изпълняват определени критерии. Например с този механизъм може да се състави множество от обекти, в което всички обекти са от определен тип и се намират на определен пласт или притежават определен цвят.

<Списък-филтър> е списък от точкови двойки, подобно на списъка, който се дава от функцията ENTDET. <Списък-филтър> определя признаците на проверяваните обекти и стойностите, които съответстват.

Пример 2:

(ssget "X" ‘((0 , "CIRCLE")))

Тази поредица ще даде множество, което съдържа всичките кръгове от чертежа (код на група 0 е типът на елемента).

Пример 3:

(ssget "X" ‘((8 , "ST3")))

Тази поредица ще даде множество, което съдържа всички обекти на слой "ST3" (код на група 8 е име на слой). Ако в <списък-филтър> е определена повече от една точкова двойка, обектът трябва да изпълнява всички условия, за да бъде включен в множеството.

Пример 4:

(ssget "X" ‘((0 , "CIRCLE") (8 , "ST3") (62 , 1)))

Тази поредица ще даде множество, което съдържа всички червени кръгове на слой "СТ3”.

Въпреки, че списъкът от филтрирани обекти притежава същия формат, както създадения с ENTGET списък, SSGET познава само определени кодове на групи. Функцията SSGET "X" прочита целия чертеж и сравнява полетата на всеки обект с определения <списък-филтър>. Ако признаците на един обект съвпадат с всички специфицирани полета на <списък-филтър>, този обект ще бъде включен в резултантното множество от избрани обекти. В противен случай обектът няма да бъде приет в множеството. Ако няма обекти, изпълняващи определените от филтъра критерии, SSGET дава Nil. Ако <списък-филтър> е празен или не съществува, SSGET избира всички обекти в базата данни.

Код	Значение
0	Тип на елемент
2	Име на блок за вмъкване на блокове (INSERT)
3	Име на шрифт за оразмеряване
6	Име на тип линия
7	Име на шрифт за текст, атрибути и дефиниране на атрибути
8	Име на слой
38	Елевация (реал.)
39	Дебелина на обект (реал.)
62	Номер на цвят (0="VONBLOCK", 256="VONLAYER")
66	Флаг (следват атрибути) за вмъкване на блок (INSERT)
210	3D вектор на посоката

За да тествате кодовете на групи, които са означени с (реал.), трябва за проверяваща стойност да се зададе реално число. Ако например трябва да се търсят обекти с елевация 2.0, първия вариант би бил грешен, но втория - верен:

(ssget "X" '((38 , 2))) грешно

(ssget "X" '((38 , 2.0))) вярно

SSGET "X" дава Nil, ако <списък-филтър> съдържа кодове на групи, които не са посочени в горната таблица. По този начин се осигурява AutoLISP-програмите с SSGET "X" да функционират и когато в бъдещите версии бъдат вмъкнати допълнителни кодове.

2.2.2 (sslength <избрани обекти>)

Тази функция дава като резултат цяло число, което обозначава броя елементи на едно множество от избрани обекти. Множествата никога не съдържат двоен избор на един и същ елемент.

Пример:

(setq a (ssget "L"))поставя в множество [a] последния обект

(sslength sset) дава 1 като резултат

2.2.3 (ssname <избрани обекти> <индекс>)

Тази функция дава името на елемента, който се намира в множеството на маркираната с <индекс> позиция. Ако <индекс> е отрицателен или по-голям от броя на елементите, резултатът е Nil. Първият елемент от множеството притежава индекс 0. Имената на елементи от множеството, които са извикани с SSGET, винаги са имена на главни елементи. Поделементи (например атрибути на блокове или върхове на полилиния) не се дават като резултат. (ENTNEXT позволява достъп до поделементите.)

Пример 1:

(setq a (ssget)) създава ново множество с име A

(setq ent1 (ssname a 0)) извиква името на I елемент в A

(setq ent4 (ssname a 3)) извиква името на IV елем. в A

За да е възможен достъпът до елементи, по-големи от 32767-мия, аргументът <индекс> трябва да се специфицира като реално число.

Пример 2:

(setq entx(ssname a 50843.0)) извиква името на 50844-ти ел.

2.2.4 (ssadd [<име на елемент> [<избрани обекти>]])

Ако се извика без аргументи, SSADD създава ново множество без членове. Ако се извика с аргумент - едно единствено име на елемент, SSADD създава ново множество с това единствено име на елемент. Ако се извика с едно име на елемент и едно множество, назованият елемент се прибавя към това множество. SSADD винаги дава новото или модифицирано множество от елементи. Ако назования елемент вече се намира в множеството, SSADD-операцията се игнорира и не се появява съобщение за грешка.

Пример:

(setq e1 (entnext)) дава E1 - име на I елемент на чертежа

(setq ss (ssadd)) поставя име "ss" на нулево множество

(ssadd e1 ss) дава множеството "ss" с вмъкнат елемент E1

(setq e2 (entnext e1)) извиква елементът след E1

(ssadd e2 ss) дава множеството "ss" с вмъкнат елемент E2

2.2.5 (ssdel <име на елемент> <избрани обекти>)

SSDEL изтрива <име на елемент> от множеството <избрани обекти> и дава като резултат името на множеството. Ако елементът не се намира в множеството, дава се Nil. В следващия пример приемаме, че елементът E1 принадлежи към множеството "ss", а елементът E2 - не.

Пример:

(ssdel e1 ss) дава множеството "ss", от което е изтрит E1

(ssdel e2 ss) дава Nil и не променя множеството "ss"

2.2.6 (ssmemb <име на елемент> <избрани обекти>)

Тази функция тества дали <име на елемент> е член на множеството <избрани обекти>. В положителния случай SSMEMB дава името на елемента; в противен случай - Nil. В следващия пример приемаме, че елементът E1 принадлежи на множеството "ss", а елементът E2 - не.

Пример:

(ssmemb e1 ss) дава името на елемента E1

(ssmemb e2 ss) дава Nil

Функции, свързани с имената на елементите

Описаните по-долу функции извършват операции, които са в директна връзка с имената на елементите. Имената на елементите могат да се въвеждат след въпроса на AutoCAD "Select objects:" и то тогава, когато е позволен избор "Последен". Назованите обекти се избират с прозорец.

2.3.1 (entnext [<име на елемент>])

Ако се извика без аргументи, тази функция дава името на първия изтрит елемент в базата данни. Ако ENTNEXT се извика с <име на елемент>, извежда се името на първия неизтрит елемент, който следва <име на елемент> в базата данни. Ако не съществува следващ елемент в базата данни, резултатът е Nil. ENTNEXT дава както главни, така и поделементи.

Избраните от SSGET елементи са главни, а не атрибути на блокове или върхове на полилинии. ENTNEXT позволява достъпа до вътрешната структура на тези комплексни елементи. Едва след като сте получили името на един поделемент, той може да се обработва като всеки друг. Ако сте получили името на поделемент с ENTNEXT, можете да намерите елемента на по-високо ниво, като продължите с ENTNEXT, докато се намери SEQEND-елемент. Извлечете сега групата "-2" на този елемент, която съответства на името на главния елемент.

Пример:

(setq e1 (entnext)) дава името на I елем.E1 на чертежа

(setq e2 (entnext e1)) дава името на елем.E2, следващ E1

2.3.2 (entlast)

Тази функция дава като резултат името на последния неизтрит главен елемент в базата данни. Често се използва за получаване името на нов елемент, който току-що е бил вмъкнат с функцията COMMAND. Елементът трябва да се намира или на монитора, или на размразен слой, за да може да бъде избран.

Пример:

(setq e1 (entlast)) дава името на посл.елем.E1 на чертежа

(setq e2 (entnext e1)) поставя E2 на Nil

2.3.3 (entsel [<въпрос>])

При операции, касаещи елементи, често е желателно да се избере елемент и едновременно да се специфицира точката, с която е избран елемента. Захватът на обект и командите BREAK, TRIM и EXTEND са примери от AutoCAD за това. Функцията ENTSEL позволява на AutoLISP-програмите да извършват тези операции. ENTSEL избира един единствен елемент чрез посочване на точка. Тя дава списък, чийто първи елемент е името на избрания елемент и втори елемент са координатите на точката, която е използвана за посочване на елемента. Ако за <въпрос> е специфициран символен низ, той пита потребителя за елемента. В противен случай се използва резервния въпрос "Select objects:".

Пример:

```
Command: LINE
From point: 1,1
To point: 6,6
To point (RETURN)
Command: (setq e (entsel "Моля, изберете елемент: "))
Моля, изберете елемент: 3,3
(<Entityname:60000014> (3.000000 3.000000))
```

Списък, който се дава от ENTSEL, може да бъде въведен при някакво запитване на AutoCAD за избор на обект. AutoCAD третира това въвеждане като избиране на елемент чрез посочване.

Функции по отношение данните за елементи

Следващите функции позволяват модификацията и достъпа до данни, които дефинират един елемент. Всички тези функции използват името на елемента, за да определят елемента, който ще се обработва.

2.4.1 (entdel <име на елемент>)

Специфицираният <име на елемент> се изтрива, ако се намира в действащия чертеж и се връща отново в чертежа, ако по-рано е бил изтрит в това работно състояние. Изтритите елементи се изчистват от чертежа, ако бъде напуснат чертожния редактор. Следователно ENTDEL може да връща само обекти, които са изтрити в актуалното работно състояние. ENTDEL може да се прилага само за главни елементи - атрибути и върхове на полилинии не могат да се изтриват, независимо от принадлежащите им елементи от по-високо ниво. (С функцията COMMAND за тази цел могат да се използват командите на AutoCAD ATTEDIT и PEDIT.)

Пример:

(setq e1 (entnext))	дава името на I елемент E1 на чертежа
(entdel e1)	изтрива елемента E1
(entdel e1)	връща изтрития елемент E1

2.4.2 (entget <име на елемент>)

Извиква <име на елемент> от базата данни и го представя като списък, който съдържа неговите дефиниционни данни. Резултатът е кодиран като асоциативен списък на AutoLISP, от който лесно могат да бъдат извикани съставните части с функцията ASSOC. Обектите от резултантния списък са кодирани с AutoCAD-DXF-код на група за всяка част от данните на елемента.

Пример 1:

```
Command: LINE
From point: 1,1
To point: 6,6
To point (RETURN)
Command: (setq a (entget (entlast)))  присвоява А на списъка
((-1 , <Entityname: 60000014>)
        (0 , "LINE")
        (8 , "0")
        (10 , 1.000000 2.000000)
        (11 , 6.000000 6.000000)
```

) отстъпът е даден за прегледност

Съставната част "-1" в началото на списъка съдържа името на елемента, който се представя чрез този списък. Функцията ENTMOD (описана по-долу) използва тази съставна част за да идентифицира модифицирания елемент.

Отделните точкови двойки, които представят стойностите, могат да се изведат лесни с ASSOC, при което стойностите могат да се извличат с CDR. Кодовете на компонентите на елемента са тези от DXF, които са документирани в Приложение С от РЪКОВОДСТВОТО на AutoCAD. Както при DXF съставните части на елемента от Header-секцията се извеждат само, ако не съответстват на резервните стойности. Противно на DXF, полетата за дефиниране на елементи, които се указват по желание, се извеждат независимо дали са равни или не на техните резервни стойности. ЧРез това правило обработването на програмата се улеснява.

Подсписъците от точки не са точкови двойки като остатъка. Условието определя, че CDR на подсписъка представя груповата стойност. Тъй като една точка е списък от 2 реални числа, цялата група е списък от 3 елемента. CDR на групата е списък, който представя точката, с което се запазва условието, че CDR винаги дава стойността.

Когато се пишат функции за обработване на списъци от елементи, те трябва да се правят в обратен ред на подсписъците. Това се осигурява с използването на ASSOC. Групата "-1", която съдържа името на елемента, позволява модификационни операции и списъкът да се приема лесно, чрез което се избягва управлението на името на елемента в паралелна структура. Еди SEQEND-елемент на края на полилиния или на атрибутен запис съдържа група "-2", чието CDR е името на елемента на Header-реда на този елемент. По този начин може да се намери Header-а вътре в поделемент, като този е стигнал до SEQEND, след което се използва CDR на група "-2" като име на елемент, за да се локализира съответния главен елемент.(???)

Пример 2:

Command: ELEV

New current elevation<0.000>: 3.5

New current thickness<0.000>: 0

Command: LINETYPE

?/Create/Load/Set: S

New entity linetype<BYLAYER>: GESTRICHELT

Command: COLOR

New entity color<BYLAYER>: BLUE

Command: LAYER

?/Make/Set/New/ON/OFF/Color/LType/Freeze/Thaw/LOck/Unlock: MAKE

New current layer<0>: AN

?/Make/Set/New/ON/OFF/Color/LType/Freeze/Thaw/LOck/Unlock: SET

New current layer<AN>: (RETURN)

?/Make/Set/New/ON/OFF/Color/LType/Freeze/Thaw/LOck/Unlock(RETURN)

Command: TEXT

Justify/SStyle/<Start point>: 2,2

Height<0.00>: 0.3

Rotation angle<0.00>: 30

Text: Текстът има печатни грешки !

Command: (setq e (entget (entlast))) Е се присвоява на списъкът:

```
((-1 , <Entityname: 6000003C>)
        (0 , "TEXT")
        (8 , "AN")
        (6 , "GESTRICHELT")
        (62 , 5)
        (38 , 3.500000)
        (10 2.000000 2.000000)
        (40 , 0.300000)
        (1 , "Текстът има печатни грешки !")
        (50 , 0.523598)
        (41 , 1.000000)
        (51 , 0.000000)
        (7 , "STANDARD")
        (71 , 0)
        (72 , 0)
        (11 2.000000 0.000000)
)
```

2.4.3 (entmake <Elist>)

Създава нови елементи.

Пример: (Нека да създадем окръжност с център (2,3,0) и радиус 1.5.)

```
(entmake '((0 , "CIRCLE")
           (10 2.0 3.0 0.0)
           (40 1.5))
)
```

2.4.4 (entmod <Elist>)

ENTMOD съдържа списък (<Elist>) във формат, който се дава като резултат от ENTGET и довежда информацията от базата данни за елемента, чието име е определени чрез група "-1" в <Elist>, в най-ново състояние. Процесът на направляване на базата данни в AutoLISP изглежда така:

с ENTGET се извикват елементи. Списъкът, който съдържа елемента се модифицира (SUBST служи за тази цел), след което елементът се въвежда в базата данни с ENTMOD.

Пример:

```
(setq en (entnext))          дава името на I елемент EN на чертежа
```

```
(setq ed (entget en))          присвоява ED на данните за елементи за име на
                               елемент EN
(setq ed (subst (cons 8 "0")
                (assoc 8 ed)   променя Layer-групата в ED
                ed))           на слой "0"
(entmod ed)    модифицира слоя на EN в чертежа
```

ENTMOD има някои ограничения по отношение на промените, които се предприемат. Типът на елемента не може да се променя. За тази цел трябва да изтриете елемента с ENTDEL и след това да създадете нов елемент с COMMAND. Всички обекти, за които се извлича информация от списъка, трябва да са познати на AutoCAD, преди да се изпълни ENTMOD. Имена на текстови шрифтове, типове линии, символи и блокове трябва да са модифицирани предварително в чертежа, преди да се използват в списък с ENTMOD. Изключение правят имената на слоеве - ENTMOD създава нов слой със стандартните - резервни стойности, както са използвани от командата LAYER "NEW", ако предварително се дава име на недефиниран слой в елементния списък.

Ако полетата на елемента се модифицират със стойности с плаваща запетая, както например групите "38"(елевация) и "39"(дебелина на обекта), трябва като нова стойност да се въведе реално число. **Ако специфицирате цяло число, ще получите грешни резултати**.

ENTMOD проверява списъка по същия начин, както DXF-данните от един DXF-файл. Ако бъде открита сериозна грешка, която пречи на направляването на базата данни, резултатът е Nil. В противен случай ENTMOD дава като резултат списъка. ENTMOD не променя вътрешни полета, като например името на елемента в група "-2" на един SEQEND-елемент. Опитите да се променят такива полета се игнорират.

Ако се направлява главен елемент, ENTMOD модифицира елемента и извежда неговия изглед на монитора (включително поделементи). Ако се използва ENTMOD за направляване на поделемент, неговите данни се въвеждат в базата данни, но на монитора не се появява нов изглед на елемента. След като е модифициран поделемент, долуописаната функция ENTUPD може да се използва за извеждане на изгледа на монитора.

2.4.5 (entupd <име на елемент>)

Както вече споменахме, модифициран с ENTMOD елемент няма да се изведе на екрана. Например, ако трябва да се модифицират 100 върхови точки на полилиния, при което ако след всяко ново изчисление трябваше да се изобразява полилинията, този процес би бил много бавен. Функцията ENTUPD може да се използва за извеждане на модифицирана полилиния или модифициран блок на монитора. ENTUPD се извиква с името на някаква част от полилиния или блок. При това не се касае за Header-елемента - ENTUPD сам търси този елемент. ENTUPD е замислен точно за полилинии и блокове с атрибути, но може да се използва за извикване на произволен елемент. Във всички случаи елементът ще се регенерира на екрана заедно с поделементите си. Да приемем, че първият елемент в чертежа е полилиния с различни върхове:

Пример:

```
(setq e1 (entnext))            дава E1 на името на елемент на полилинията
(setq e2 (entnext e1))         дава E2 на първата върхова точка
(setq ed (entget e2))          дава ED на данните за върхове
(subst '(10 1.0 2.0)
       (assoc 10 ed)   променя местоположението на върха в ED
```

```
        ed)                 на точка 1,2
    )
    (entmod ed)             премества върховата точка в чертежа
    (entmod e1)             регенерира елемента-полилиния E1
```

2.4.6 Ограничения

Имената на елементи и множествата от избрани обекти са действащи само за това състояние на редактиране, в което са били създадени от AutoCAD. Ако опитате да изпълните една от следните операции, докато са активни командите PLINE или ATTEDIT, резултатът ще е Nil и извиканата функция няма да се изпълни:

ENTMOD - модифициране на съществуващ елемент

ENTUPD - регенериране на модифициран комплексен обект

ENTDEL - извеждане и регенериране на изтрит елемент

Използване имената на елементи и множествата от избрани обекти в AutoCAD

Имената на елементи и множествата от избрани обекти се въвеждат като данни от AutoLISP на въпросите на AutoCAD за избор на обект. Затова назованите от AutpLISP елементи могат да се обработват от командите на AutoCAD. На въпроса "Select objects:" AutoLISP може да зададе името на елемента, с което се определя един единствен елемент или множество от избрани обекти, което определя всички елементи в този запис. Това е в сила винаги, когато се поддържа опцията "Last".

Винаги, когато AutoCAD допуска избор чрез посочване на обекти, от процедурата за избор се приемат списъци във формата, както се дават от ENTSEL. Елементът се избира от списъка, при което точката в списъка се определя като точка, с която е посочен елемента. AutoLISP може да задава такива точки на елементи на команди като BREAK, TRIM или EXTEND. Списъци с формата на ENTSEL могат да се използват за други видове избор, докато командата поддържа избор чрез посочване (обектът се посочва с точка). ENTSEL-списъци не могат да се използват с командите FILLET и CHAMFER, тъй като те изискват 2 елемента и точки.

Забележки при обработването на крива и сплайн за полилинии

Ако търсите върховите точки на полилиния с ENTNEXT, възможно е да се натъкнете на определен брой върхови точки, които не сте създали явно. Тези помощни точки се вмъкват автоматично от опциите "Крива" и "Сплайн" на командата PEDIT. Вероятно искате да игнорирате тези точки, тъй като промените на тези върхови точки ще бъдат загубени при следващото използване на опциите "Крива" и "Сплайн".

По флаговете на група "70" може да се познае дали полилинията е обработена с "Крива" (ст.на бит 2) или "Сплайн" (ст.на бит 4). В случай, че никой от тези битове на е зареден, всички върхови точки на полилинията са нормални. Ако обаче битът за "Крива" (2) е зареден, върховите точки на полилинията притежават редувайки се стойност на бит 1 в тяхната група "70" и с това показват, че те са вмъкнати с "Крива". Ако използвате ENTMOD за преместване на върховите точки на такава полилиния и имате намерение отново да закръглявате кривата с PEDIT, най-добре е да игнорирате тези върхови точки.

Същото е в сила, когато флаг-битът на една полилиния е зареден със "Сплайн" (4). Предварително ще намерите различни върхови точки, някои със стойност на флаг-бит 1 (вмъкнати с "Крива", когато системната променлива SPLINESEGS е отрицателна), някои със стойност на бит 8 (вмъкнати със "Сплайн") и други със стойност на бит 16 (контролни точки на рамката на кривата). Също и тук е в сила - ако искате да преместите върховите точки с ENTMOD и имате намерение да създадете нова крива, най-добре е да преместите върховите точки.

Достъп до таблицата със символи

За да се четат таблиците със символи на AutoCAD за слоеве, тъп линии, именовани сектори, шрифтове и дефиниции на блокове, са въведени новите AutoLISP-функции TBLNEXT и TBLSEARCH.

2.6.1 (tblnext <име на таблица> [<първи>])

Тази функция се използва за претърсване на цяла таблица от символи. Първият аргумент е символен низ, който идентифицира желаната таблица. Валидни имена на таблици са: "LAYER", "LTYPE", "VIEW", "STYLE" и "BLOCK". Символният низ не трябва да се записва с главни букви. Ако е налице втори аргумент и се оценява на стойност, различна от Nil, таблицата се навива в обратна посока и се намира първия запис. В противен случай се намира следващия запис. Ако няма други записи в таблицата, резултатът е Nil. Изтрити записи от таблицата никога не се връщат. Един намерен запис се представя като списък от точкови двойки, от кодове и стойности на типа DXF, подобно на тези, които се дават от функцията ENTGET.

Пример 1:

```
(tblnext "layer" T)              намиране на първи слой
```

Това въвеждане би довело до следното:

```
((0 , "LAYER")                   тип на символа
 (2 , "0")                 име на символа
 (70 , 0)                        флагове
 (62 , 7)                        номер на цвят (отрицателен, ако е off
 (6 , "CONTINUOUS")        име на тип линия
)
```

Забележете, че за разлика от списъците с данни за обекти, тук няма група "-1". AutoCAD записва от всяка таблица последния изведен запис и дава като резултат при извикване на TBLNEXT за тази таблица просто следващия запис. Преди да претърсвате една таблица, осигурете се, следващият аргумент да е различен от Nil, за да започне търсенето в таблицата отгоре и резултатът наистина да е първия запис.

Записи, намерени в таблицата "BLOCK", съдържат група "-2" с "име на обект" на първия обект в дефиницията на блока (ако изобщо има такава).

Пример 2 (за блок "BOX"):

```
(tblnext "block")                    намиране на дефиниция на блок
```

би дал следното:

```
((0 , "BLOCK")                              тип на символа
 (2 , "BOX")                          име на символа
 (70 , 0)                                   флагове
 (10 9.000000 2.000000 0.000000)    изходна точка X,Y,Z
```

(-2 , <OBJECTNAME: 40000126>) първи обект

)

Името на обекта в група "-2" се приема от ENTGET и ENTNEXT, обаче не и от други функции, които имат достъп до обекта. Те не могат да модифицират такъв обект с ENTMOD или с SSADD и ENTSEL да го вмъкнат в множеството от избрани обекти. От това, че името на обекта се задава в група "-2" на функцията ENTNEXT, могат да бъдат претърсвани обектите, които се съдържат в една дефиниция на блок. ENTNEXT дава Nil след последния обект в дефиницията на блока.

2.6.2 (tblsearch <име на таблица> <символ>)

Тази функция претърсва таблицата със символи, която се идентифицира от <име на таблица> (като при TBLNEXT) и търси името на символа, което е специфицирано чрез <символ>. Двете имена автоматично се превръща в главни букви. Ако се намери запис за дадено име на символ, този запис се дава във формат, който се описва от TBLNEXT. Ако не се намери такъв запис, резултатът е Nil.

Пример:

(tblsearch "style" "standard") намиране на шрифт

Това би довело до следното:

((0 , "STYLE")
(2 , "STANDARD")
(70 , 0)
(40 , 0.000000)
(41 , 1.000000)
(50 , 0.000000)
(71 , 0)
(3 , "txt")
(4 , "")
)

Последователността от намерени от TBLNEXT записи не се влияе от TBLSEARCH.

Достъп до графичния монитор и входните устройства

Описаните в този раздел AutoLISP-функции позволяват директен достъп от AutoLISP до графичния монитор на AutoCAD и входното устройство. Те позволяват на потребителя да използва внедрени AutoLISP-команди като AutoCAD-команди. Тези команди могат да предизвикат безредие на екрана. Евентуални смущения могат да се снижат със следната последователност от команди:

(grtext)

(redraw)

Тези функции могат да се прилагат само от опитни специалисти. Повечето приложения на AutoLISP не изискват тези функции. Искаме да обърнем внимание на потребителите, че начинът на работа на такива функции може да бъде променен в бъдещите версии на AutoCAD, затова фирмата "Autodesk-AG" не дава гаранция за съвместимостта на приложенията, които се ползват от тези функции. Приложения с функциите GRTEXT и GRREAD вероятно не работят еднакво на всички хардуерни конфигурации, ако системния програмист не се придържа плътно към следващите правила.

2.7.1 (grclear)

Тази функция изчиства графичния монитор на AutoCAD. При конфигурации с един екран най-напред се превключва от текстов на графичен екран. Областта за команди, както и областта на менюто и статусния ред остават непроменени. Първоначалното съдържание на екрана може да се върне с функцията REDRAW.

2.7.2 (grdraw <от> <до> <цвят> [<маркиране>])

GRDRAW чертае вектор между две точки. <От> и <до> специфицират двете крайни точки (списък от две реални числа) на вектора. Крайните точки се определят като чертожни координати с плаваща запетая и по необходимост се закръгляват за да се вместят на екрана. Векторът се чертае с аргумент <цвят> (цяло число), при което "-1" означава "XOR ink" и допълва всичко, над което чертае и изтрива себе си, ако е чертано там. Ако факултативният целочислен аргумент <маркиране> е даден и не е равен на Nil, векторът се маркира (в общи линии се щрихова). Ако <маркиране> е пропуснат или е равен на Nil се използва нормалния режим на изобразяване.

2.7.3 (grtext <поле> <текст> [<маркиране>])

GRTEXT позволява на AutoLISP да пише в текстовете на графичния монитор на AutoCAD. Ако функцията се извика с номер <поле> между 0 и най-високия номер на поле от екранното меню минус 1, AutoCAD изобразява аргумента <текст> в определеното поле на екранното меню. <Текст> се отрязва, ако не се вмести в полето или съдържа интервали, когато е прекалено къс. Ако е даден целочисления аргумент <маркиране>, който се указва по желание, или не е равен на Nil, текстът в определеното поле се маркира. Ако <маркиране> е равно на 0, текстът се появява в нормалното си състояние. Ако едно поле се маркира, това предизвиква изключване на вече маркирано поле. При писане в поле от меню, текстът трябва да се напише най-напред без аргумент <маркиране> и едва тогава да се маркира. Същият текст, който е написан първоначално в полето, трябва да се използва при прилагане на функцията за маркиране. При неспазване на тези правила може да се окаже, че AutoLISP-програмите работят различно на различни монитори.

Тази функция просто изобразява зададения текст в областта на менюто на екрана. Екранното меню не се преписва.

Ако GRTEXT се извиква с номер на поле "-1", текстът ще се изпише в статусния ред за режим на монитора. Дължината на статусния ред е различна за различните монитори. Текстът се вмества в съответното пространство и евентуално се отрязва.

Ако GRTEXT се извиква с номер на поле "-2", текстът се изписва в статусния ред за координати на монитора. Ако е включен режима за изобразяване на координати, координатите в това поле се преместват с всяко преместване на курсора. За номера на полета "-1" и "-2" аргументът <маркиране се игнорира.

GRTEXT може да се извика без аргументи, за да се върнат на всички текстове върху монитора техните стандартни стойности.

2.7.4 (grread [<drag>])

С GRREAD могат директно да бъдат четени входните устройства на AutoCAD, като при желание може да се следи всяко движение на указващото устройство. Тази функция се използва само при много специални команди. Повечето въвеждания на AutoLISP стават с GETxxx-функциите (GETSTRING, GETREAL и т.н.). Ако е зададен аргумента <drag> и не е равен на Nil, координатите се задават в процеса на движение на указващото устройство, без да се натискат клавиши. AutoCAD използва този механизъм за реализиране видимото водене на буксир на екрана.

GRREAD дава списък, чийто първи елемент е код, който определя типа на въвеждане. Вторият елемент на списъка е или цяло число, или списък от точки, според вида на въвеждане. Кодовете на първия елемент на списъка са:

2	знак от клавиатурата - ASCII-код като втори елемент
3	избрана точка - координати като списък
4	избор на поле на екранно меню - номер на поле като втори елемент
5	координати като втори елемент в режим DRAGMODE. Дава се само, ако е определен втория аргумент и не е равен на Nil
6	избор на меню-функция BUTTON - номер на бутон е втория елемент
7	дигитализиране на меню-функция TABLETT1 - номер на поле е втория елемент
8	дигитализиране на меню-функция TABLETT2 - номер на поле е втория елемент
9	дигитализиране на меню-функция TABLETT3 - номер на поле е втория елемент
10	дигитализиране на меню-функция TABLETT4 - номер на поле е втория елемент
11	дигитализиране на меню-функция AUX - номер на поле е втория елемент
12	задаване на асоциирани координати като втори елемент с бутона на указващото устройство. Винаги следва резултатния списък на тип 6
13	избор на функция от екранното меню, маркиран чрез въвеждане на данни от клавиатурата. Номерът на поле се дава като втори елемент

Въвеждането на CTRL+C по време на изпълнение на GRREAD прекъсва AutoLISP-програмата. Всяко друго на данни се направлява по-нататък от GRREAD, чрез което се осигурява постоянен контрол върху входните устройства.