Wykorzystanie języka AutoLISP.

AutoLISP jest językiem programowania podobnie jak Basic, Pascal, C.
Różni się tym, że jest zintegrowany z AutoCADem i nie można go używać oddzielnie,
AutoLISP liczy w arytmetyce liczb rzeczywistych, obsługuje pliki, przetwarza zmienne tekstowe, wykonuje operacje logiczne i bitowe, przetwarza listy oraz pomaga w rysowaniu.
Translator AutoLISPu jest interpreterem. Programy są tłumaczone linia po linii.
Czym różni się AutoLSIP od skryptu?
Dzięki programowi AutoLisp można np. wykonać rysunki parametryczne, czego nie może zrobić skrypt.
AutoLISP pozwala tez na pisanie funkcji.

Obiekty języka AutoLISP
Typ INT – liczba całkowita -32768 do 32767
Typ REAL – liczba rzeczywista z dokładnością 8 cyfr znaczących
Typ STR – łańcuchy tekstowe, np.
„\n” - znak przejścia do nowej linii
„c\acad\\lisp\prog1.lsp”
Typ FILE – wartości sa deskryptorami plików
Typ SYM – symbole, np. 012abc, +, open, T
Typ SUBR – wbudowane funkcje programu
np. (setq plus +); (+ 5 4)
Typ LIST – listy
Np. (1 2 3) („abc” (12 2.0))
Typ PICKSET – zbiory wyboru AutoCADA, np.
(setq zbiorwyb_okno (setq „_w” ‘(0 0) ‘(10 10)))
Typ ENAME – nazwy entycji AutoCADa, np. linie, bloki itp.

Instrukcje AutoLISPu mają strukturę
(symbol_operatora argument1 argument2 …)

Funkcje
Schemat
((parametry_formalne /zmienne_lokalne) (cialo_funkcji))

Programy
Ciąg instrukcji zapisany w pliku o rozszerzeniu .LSP.

Wczytanie programu
(load „nazwa_pliku_lsp” [wyrazenie])
Nazwę można pominąć jeżeli jest to rozszerzenie **.lsp.**
Jeśli program ma inne rozszerzenie to trzeba podać pełną nazwę.
Uruchomienie
Wpisujemy nazwę funkcji w nawiasie lub bez nawiasu jeśli przed nazwą procedury występuje c:

Program napisany w AutoLISP składa się z procedur.

Przykładowy program – rysowanie odcinka PQ
```
; Program r1.lsp
```

```
; rysowanie odcinka
; definicja funkcji r1
defun r1()
; wczytanie punktow
(setq p (getpoint "\Podaj 1 punkt: "))
(setq q (getpoint "\Podaj 2 punkt: "))
; komenda rysowania linii
(command "_LINE" p q "")
)
```

Wczytanie programu:
(load „r1.lsp”) lub (load :”r1”) albo wczytaj aplikację z menu.
Uruchomienie: (r1)

Jeżeli nazwa funkcji jest poprzedzona C: to uruchamiamy ją wpisując nazwę bez nawiasów.

Przykładowe programy

; Rysowanie linii prostopadłych do innych linii

```
; This program permits you to draw lines perpendicular to
; other lines by changing the snap angle to the angle of the
; target line.  Be sure to execute perpdoff.lsp after you
; draw your perpendicular lines.
;
(defun c:perpdon (/ a b pnt1 pnt2 ang1)
    (graphscr)
    (setq a (entsel))
    (setq b (entget (car a)))
    (setq pnt1 (cdr (assoc 10 b)))
    (setq pnt2 (cdr (assoc 11 b)))
    (setq ang1 (angle pnt1 pnt2))
    (setvar "snapang" ang1)
)
(defun c:perpdoff ()
    (setvar "snapang" 0)
)
```

Zapis współrzędnych wskazanych punktów do pliku

```
(defun c:xyz ()
; Zapis wspolrzednych wskazanych pktow do pliku

(defun zapis ()
  (progn
    (print "Lp ")
    (princ sp)
    (princ i)
    (setq p1 (getpoint "    Punkt :  "))
```

```
    (if (= n 0) (princ n plik) (print n plik))
    (princ sp plik)
    (princ (car p1) plik)
    (princ sp plik)
    (princ (cadr p1) plik)
    (princ sp plik)
    (princ (caddr p1) plik)
    (terpri)
  ) ; progn
) ; zapis

 (graphscr)
 (setq sp "  ")
 (setq nazwa (getstring "Nazwa pliku do zapisu wspolrzed.: "))
 (terpri)
(setq plik (open nazwa "a"))

(if (eq plik nil)
  (*ERROR* "Zapis: Nie moge otworzyc pliku ")
)

(setq n 1 i 0)

(while (/= n 0)
 (progn
  (setq n (getint "Nr pktu lub 0 gdy koniec : "))
  (if (/= n 0) (zapis))
  (setq i (+ 1 i))
 ) ; progn
) ; while

(close plik)

) ; xyz
 (princ "\nW celu wywolania napisz XYZ")
(princ)
```

Strzałka

```
(defun C:STR ()
  (setq pp (getpoint"\nSTRZALA - Punkt poczatkowy strzalki :"))
  (setq pk (getpoint pp "\nPunkt strzalki z grotem: "))
  (COMMAND "_LINE" PP PK "")
  (setq p1 (geTpoint pk  "\nKoniec grotu strzalki: "))
  (COMMAND "_LINE" PK P1 "")
  (setq azp (angle pk pp))
  (setq az1 (angle pk p1))
  (setq alfa (- azp az1))
```

```
  (setq az2 (+ azp alfa))
  (setq a (distance pk p1))
  (setq p2 (polar pk az2 a))
  (command "_linE" pk p2 "")
)

(princ "\nWczytano STR.LSP - strza│ka. W celu wywolania napisz STR")
(princ)
```

Skarpy

```
; zamiana stopni na grady
(defun dtr (a)
 (* pi (/ a 180.0))
)

; pobranie informacji

(defun gp ()
 (setq sp (getpoint "\nPunkt pocz skarpy (kreski na prawo od linii): "))
 (setq ep (getpoint sp "\nPunkt konc skarpy : "))
 (setq ap (getpoint sp "\n1-szy punkt boczny skarpy (kolo pktu pocz) : "))
 (setq bp (getpoint ep "\n2-gi  punkt boczny skarpy (kolo pktu konc) : "))

 (setq tspac (getdist "\nPrzybl. odleglosc miedzy kreskami (bedzie dopasowana): " sp))
 (setq x (getint "\Rysowanie obrysow skarpy? 1-calosc, 2-gora i dol, 3-gora: "))

 (setq x1 (car sp))
 (setq y1 (cadr sp))
 (setq x2 (car ep))
 (setq y2 (cadr ep))
 (setq x3 (car ap))
 (setq y3 (cadr ap))

(setq A (- y2 y1))
 (setq B (- x1 x2))
 (setq C (- (- (* A x1)) (* B y1)))
 (setq d1 (/ (+ (* A x3) (* B y3) C) (sqrt (+ (* A A) (* B B)))))
 (setq d2 (/ (+ (* A x2) (* B y2) C) (sqrt (+ (* A A) (* B B)))))
 (setq pangle  (angle sp ep))
 (setq pang2 (angle ap bp))
 (setq alfa (- pangle pang2))
 (setq plength (distance sp ep))
 (setq angp90 (+ pangle (dtr 90))) ; kat skarpy + 90 stopni
 (setq angm90 (- pangle (dtr 90))) ; kat skarpy minus 90 stopni
 (setq s (distance sp ep))
 (setq n1 (/ s tspac))
 (setq n2 (fix n1))
 (setq tspac (/ s n2))
)
```

```
; Rysowanie obrysu skarpy
(defun drt ()
 (command "_line" sp ep ^C^C)
 (command "_line" ep bp ^C^C)
 (command "_line" bp ap ^C^C)
 (command "_line" ap sp ^C^C)
)

(defun drt2 ()
 (command "_line" sp ep ^C^C)
 (command "_line" ap bp ^C^C)
)

 (defun drt3 ()
 (command "_line" sp ep ^C^C)
)

; Rysowanie kresek skarpy
(defun drawk1 ()
    (setq p1 (polar sp pangle pdist))
    (setq p2 (polar p1 angm90 y1))
    (command "_line" p1 p2 ^C^C)
)


; Rysowanie kresek skarpy
(defun drawk2 ()
    (setq p3 (polar sp pangle pdist))
    (setq p4 (polar p3 angm90 y1))
    (command "_line" p3 p4 ^C^C)
)


; Rysowanie kresek skarpy
(defun draw1 ()
  (setq pdist  tspac)
  (while (< pdist plength)
    (setq y1 (* pdist (/ (sin alfa) (cos alfa))))
    (setq y1 (+ d1 y1))
    (drawk1)
    (setq pdist (+ pdist (* 2.0 tspac)))
   )
)


; Rysowanie kresek skarpy
(defun draw2 ()
  (setq pdist  (* 2 tspac))
  (while (< pdist plength)
```

```
    (setq y1 (* pdist (/ (sin alfa) (cos alfa))))
    (setq y1 (+ d1 y1))
    (setq y1 (/ y1 2.0))
    (drawk2)
    (setq pdist (+ pdist (* 2.0 tspac)))

  )
)

; Wykonanie polecenia przes wywolaqnie zdefiniowanych funkcji

 (defun C:skarpy ()
  (gp)
  (if (= x 1) (drt))
  (if (= x 2) (drt2))
  (if (= x 3) (drt3))
  (draw1)
  (draw2)
 )

 (princ "\nWczytano SK.LSP. W celu wywolania napisz SKARPY")
 (princ)
```

Czytanie danych z pliku. Rysowanie okręgów

```
; Program okrlnr.lsp
; Okregi, linie, numery
(defun c:okrlnr ()
  (graphscr)
  (prompt "\nFunkcja czyta dane z pliku (Nr X Y), rysuje okregi i numery")
  (terpri)

  (setq
      nazplik (getstring "\nNazwa pliku z danymi: ")
      f    (open nazplik  "r")
      r 1.0
      wiersz (read-line f)
      cr "\n"
   )

  (while wiersz
    (setq
        p0 p1
        na (substr wiersz  1 2)
        ya (substr wiersz  3 11)
        xa (substr wiersz 12 20)
        y (atof ya)
        x (atof xa)
        p1 (list x y)
```

```
        x2 (+ 6 x)
           y2 (+ 2 y)
           p2 (list x2 y2)

    );setq

    (princ na) (princ "   ")
    (princ ya) (princ "   ")
    (princ xa) (princ "   ")
    (terpri)

    (command "_circle" p1 1.0)

    (setq wiersz (read-line f))
;    (setq p0 p1)

    (command "_line" p0 p1 "" )
    (command "_text" p2 "2.5" "0" na)

   );while

  (close f)
)
```