SCRIPT EXAMPLES

all_dimensions

all_dimensions offset small_dimension textHeight

Description:

Adds all the dimensions of the drawing

Arguments:

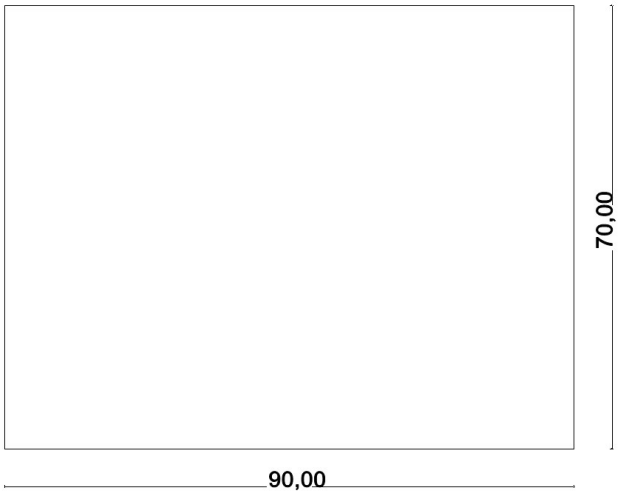
• offset = distance from geometry
• small_dimension = minimum dimension size to ignore
• textHeight = text size

Script:

line 10 10 100 10
line 100 10 100 80
line 100 80 10 80
line 10 80 10 10

all_dimensions 6 1 2

Result:



angular_dimension

angular_dimension x1 y1 x2 y2 x3 y3 x4 y4 tx ty textHeight

Description:

Adds an angular dimension between two lines

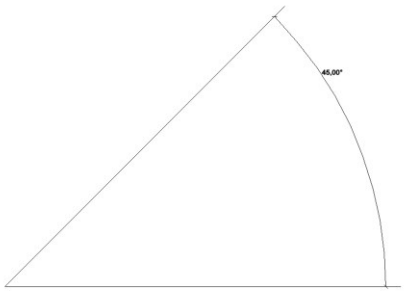
Arguments:

• x1 = first line start X
• y1 = first line start Y
• x2 = first line end X
• y2 = first line end Y
• x3 = second line start X
• y3 = second line start Y
• x4 = second line end X
• y4 = second line end Y
• tx = text position X
• ty = text position Y
• textHeight = text size

Script:

```
dim cx = 50
dim cy = 50
dim r = 15

' horizontal
line cx cy cx+r cy

' angled
line cx cy cx+r*cos(pi/4) cy+r*sin(pi/4)

' place dimension
angular_dimension cx cy cx+r cy cx cy cx+r*cos(pi/4) cy+r*sin(pi/4) cx+12 cy+8 5
```

Result:



arc

arc cx cy x1 y1 x2 y2

Description:

Draws an arc using center and two points

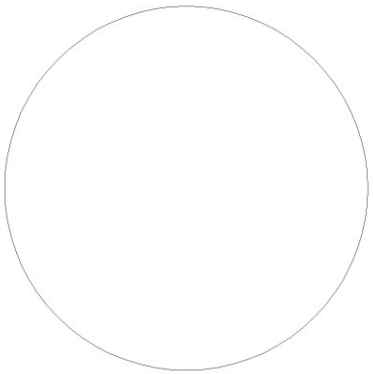
Arguments:

- *cx = center X*
- *cy = center Y*
- *x1 = start point X*
- *y1 = start point Y*
- *x2 = end point X*
- *y2 = end point Y*

Script:

```
dim cx = 50 : dim cy = 50 : dim r = 30
' 0 deg (right)
dim x0 = cx + r : dim y0 = cy
' 90 deg (top)
dim x1 = cx : dim y1 = cy + r
' 180 deg (left)
dim x2 = cx – r : dim y2 = cy
' 270 deg (bottom)
dim x3 = cx : dim y3 = cy - r

' back to 0 deg
dim x4 = x0 : dim y4 = y0

' draw 4 arcs
arc cx cy x0 y0 x1 y1
arc cx cy x1 y1 x2 y2
arc cx cy x2 y2 x3 y3
arc cx cy x3 y3 x4 y4
```

Result:



circle

circle cx cy r

Description:

Draws a circle

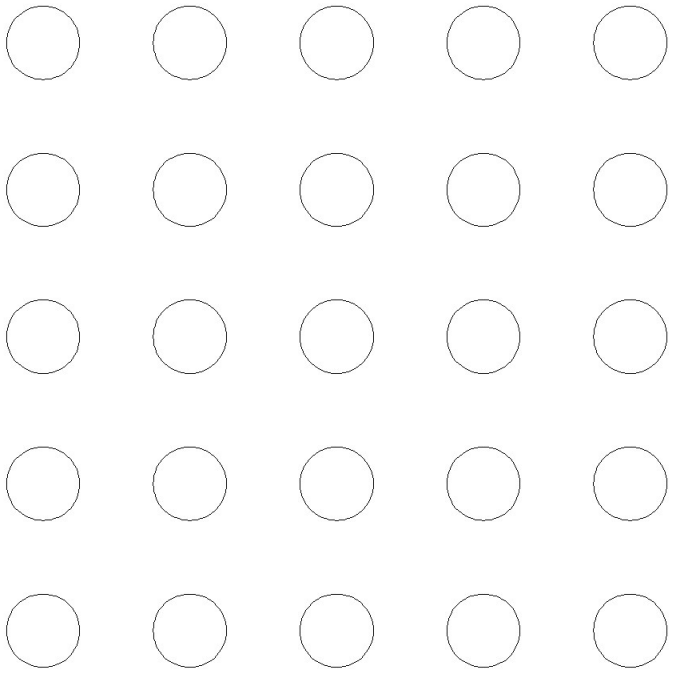
Arguments:

- *cx = center X*
- *cy = center Y*
- *r = radius*

Script:

```
for i = 0 to 4 step 1
    for j = 0 to 4 step 1
        circle i*20 j*20 5
    next j
next i
```

Result:



ellipse

ellipse cx cy rx ry

Description:

Draws an ellipse

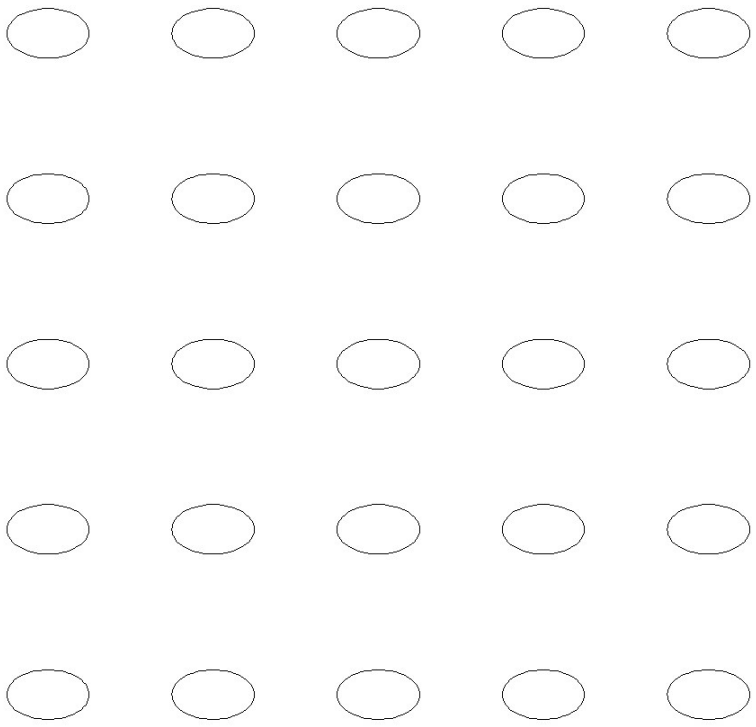
Arguments:

- *cx = center X*
- *cy = center Y*
- *rx = radius X*
- *ry = radius Y*

Script:

```
for i = 0 to 4 step 1
      for j = 0 to 4 step 1
            ellipse i*20 j*20 5 3
      next j
next i
```

Result:



point

point x y

Description:

Draws a point

Arguments:

- *x = X position*
- *y = Y position*

Script:

```
for i = 0 to 4 step 1
      for j = 0 to 4 step 1
            point i*20 j*20
      next j
next i
```

Result:

rectangle

rectangle x y w h

Description:

Draws a rectangle

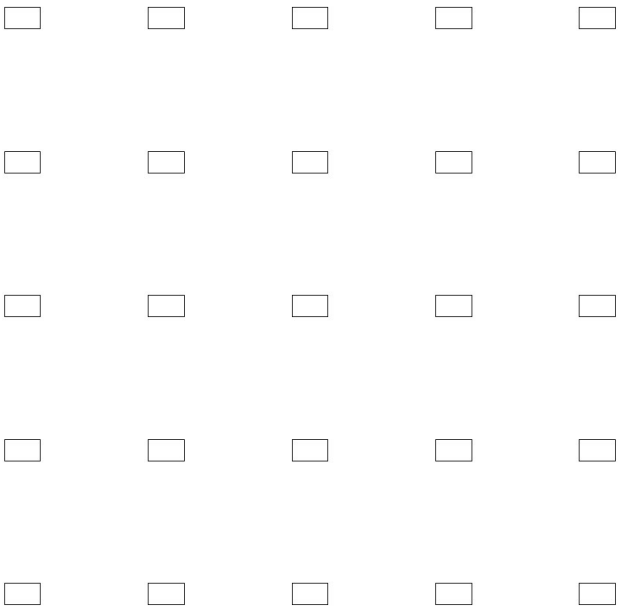
Arguments:

- *x = bottom-left X*
- *y = bottom-left Y*
- *w = width*
- *h = height*

Script:

```
for i = 0 to 4 step 1
      for j = 0 to 4 step 1
            rectangle i*20 j*20 5 3
      next j
next i
```

Result:



hatch

hatch x[] y[] distance angle outside border add_the_polyline

Description:

Applies a linear hatch to a closed shape

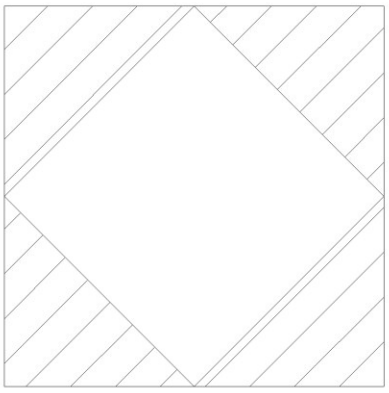
Arguments:

- *x[] = array of X points*
- *y[] = array of Y points*
- *distance = spacing between hatch lines*
- *angle = hatch angle (degrees)*
- *outside = 1 = outside / 0 = inside*
- *border = 1 = draw border / 0 = no border*
- *add_the_polyline = 1 = yes / 0 = no*

Script:

```
dim cx = 50
dim cy = 50
dim d = 30

dim x(4)
dim y(4)

' diamond points (top, right, bottom, left, close)
x(0)=cx          : y(0)=cy + d
x(1)=cx + d      : y(1)=cy
x(2)=cx          : y(2)=cy - d
x(3)=cx - d      : y(3)=cy
x(4)=x(0)        : y(4)=y(0)

' hatch it
hatch x y 5 45 1 1 1
```

Result:



linear_dimension

linear_dimension x1 y1 x2 y2 offsetX offsetY align textHeight

Description:

Adds a linear dimension between two points

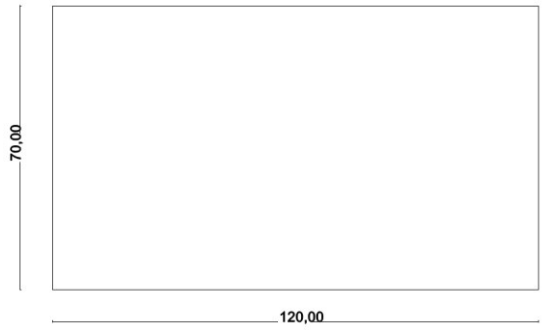
Arguments:

- *x1 = first point X*
- *y1 = first point Y*
- *x2 = second point X*
- *y2 = second point Y*
- *offsetX = dimension offset in X*
- *offsetY = dimension offset in Y*
- *align = alignment (-1, 0, 1)*
- *textHeight = text size*

Script:

```
dim w = 120
dim h = 70

rectangle 0 0 w h

linear_dimension 0 0 w 0 8 0 -1 2
linear_dimension 0 0 0 h 8 -1 0 2
```

Result:



line

line x1 y1 x2 y2

Description:

Draws a line from one point to another

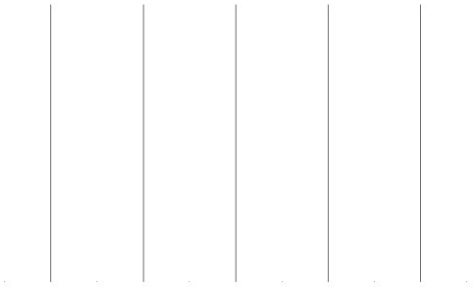
Arguments:

• x1 = start X
• y1 = start Y
• x2 = end X
• y2 = end Y

Script:

```
dim w = 100
dim h = 60
dim step = 10

for i = 0 to w step step

    ' alternate direction using mod
    dim y1 = 0
    dim y2 = h

    if i mod (2*step) = 0 then y2 = 0

    line i y1 i y2

next i
```

Result:

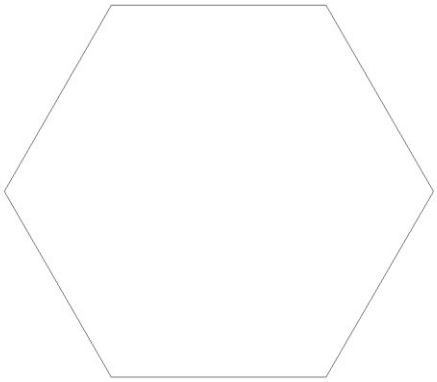


polygon

Script:

```
dim cx = 50
dim cy = 50
dim r = 30
dim sides = 6

dim prevx = cx + r
dim prevy = cy

for i = 1 to sides step 1
    dim a = 2 * pi * i / sides
    dim x = cx + r * cos(a)
    dim y = cy + r * sin(a)

    line prevx prevy x y

    prevx = x
    prevy = y
next i
```

Result:



polyline

polyline x[] y[]

Description:

Draws connected line segments using arrays of points

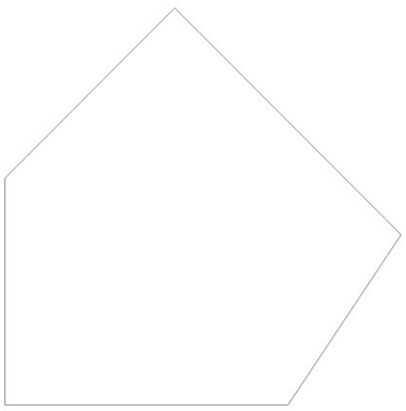
Arguments:

- *x[] = array of X points*
- *y[] = array of Y points*

Script:

```
dim x(5)
dim y(5)

x(0) = 10
y(0) = 10

x(1) = 60
y(1) = 10

x(2) = 80
y(2) = 40

x(3) = 40
y(3) = 80

x(4) = 10
y(4) = 50

x(5) = 10
y(5) = 10

polyline x y
```

Result:

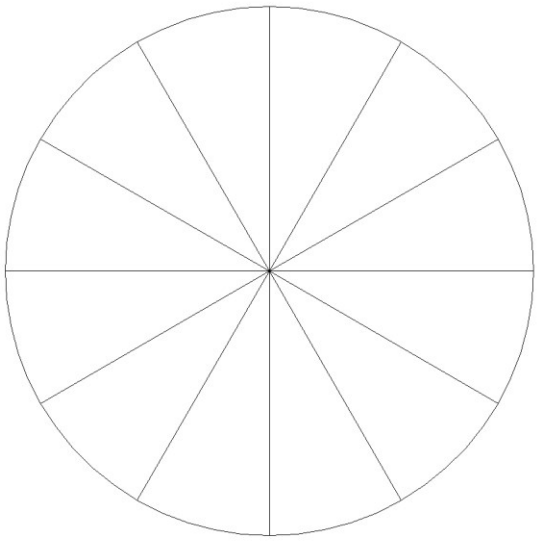


radial pattern

Script:

```
dim cx = 50
dim cy = 50
dim r = 30
dim steps = 12

' draw the circle
circle cx cy r

for i = 1 to steps step 1
    dim angle = 2 * pi * i / steps
    dim x = cx + r * cos(angle)
    dim y = cy + r * sin(angle)

    line cx cy x y
next i
```

Result:



rectangle with lines

Script:

```
dim w = 100
dim h = 60

line 0 0 w 0
line w 0 w h
line w h 0 h
line 0 h 0 0
```

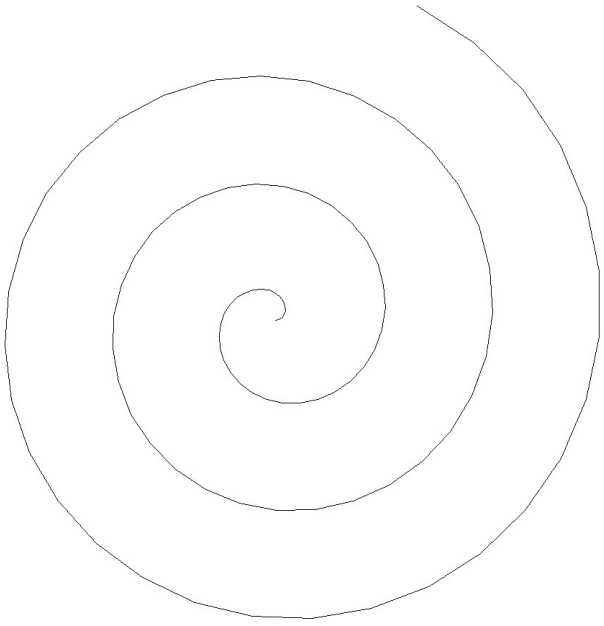
Result:

spiral

Script:

```
dim px = 50
dim py = 50

for i = 1 to 100 step 1
    dim a = i * 0.2
    dim r = i * 0.5

    dim x = 50 + r * cos(a)
    dim y = 50 + r * sin(a)

    line px py x y

    px = x
    py = y
next i
```

Result:



text

text "message" x y size

Description:

Draws text at a specific position

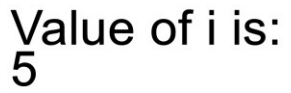
Arguments:

• message = the text to display (inside quotes)
• x = position X
• y = position Y
• size = text height

Script:

```
dim x = 20
dim y = 80
dim size = 10
dim i = 5

dim text = "Value of i is:"
text text x y size
text "5" x (y-1.5*size) size
```

Result:



triangle

triangle x1 y1 x2 y2 x3 y3

Description:

Draws a triangle using three points.

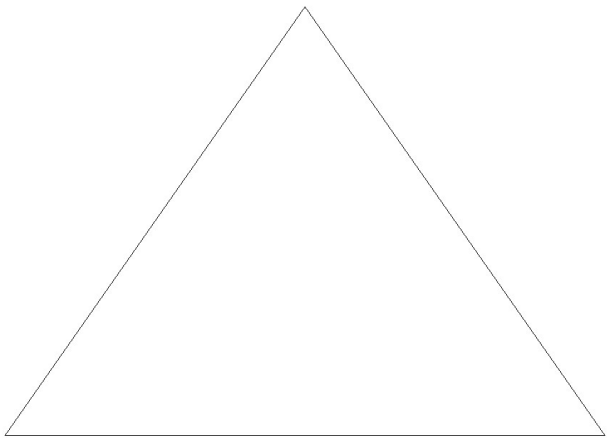
Arguments:

- x1 = first point X
- y1 = first point Y
- x2 = second point X
- y2 = second point Y
- x3 = third point X
- y3 = third point Y

Script:

triangle 10 10 80 10 45 60

Result:



rounded_rectangle

rounded_rectangle x y w h r

Description:

Draws a rectangle with rounded corners.

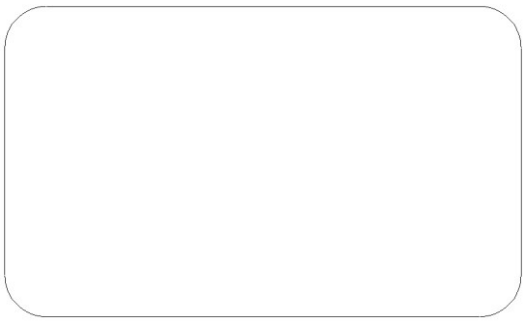
Arguments:

- x = bottom-left X
- y = bottom-left Y
- w = width
- h = height
- r = corner radius

Script:

rounded_rectangle 10 10 100 60 8

Result:



slot

slot x1 y1 x2 y2 r

Description:

Draws a slot shape between two points.

Arguments:

- x1 = start X
- y1 = start Y
- x2 = end X
- y2 = end Y
- r = slot radius

Script:

slot 10 20 100 20 5

Result:

star

star cx cy r1 r2 points

Description:

Draws a star polygon.

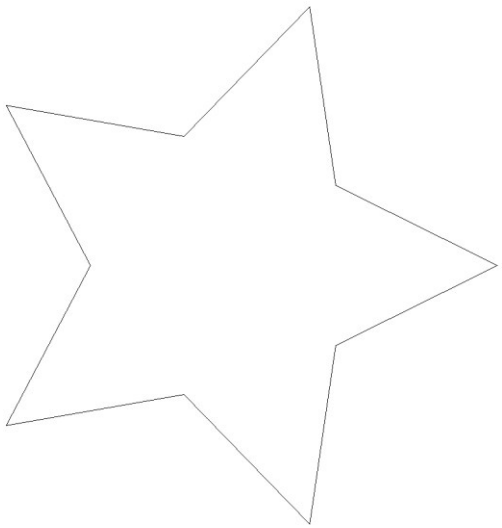
Arguments:

- cx = center X
- cy = center Y
- r1 = outer radius
- r2 = inner radius
- points = number of star points

Script:

star 50 50 30 15 5

Result:



grid

grid x y w h step

Description:

Draws a rectangular grid.

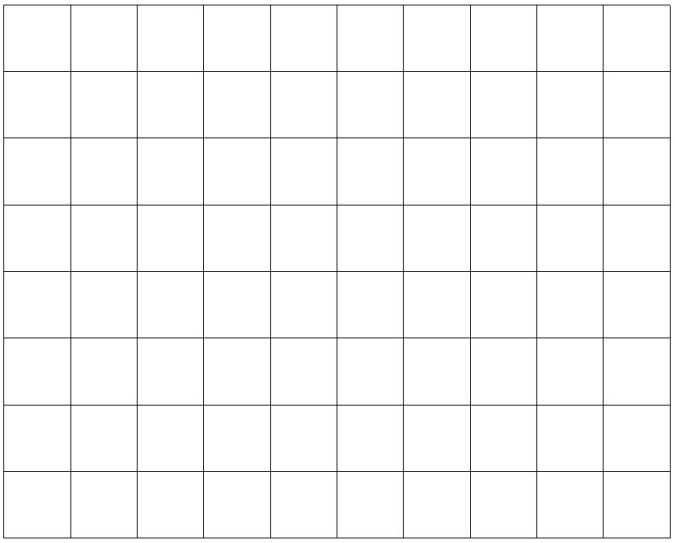
Arguments:

- x = start X
- y = start Y
- w = grid width
- h = grid height
- step = spacing between grid lines

Script:

grid 0 0 100 80 10

Result:



donut

donut cx cy outerRadius innerRadius

Description:

Draws two concentric circles forming a donut/ring shape.

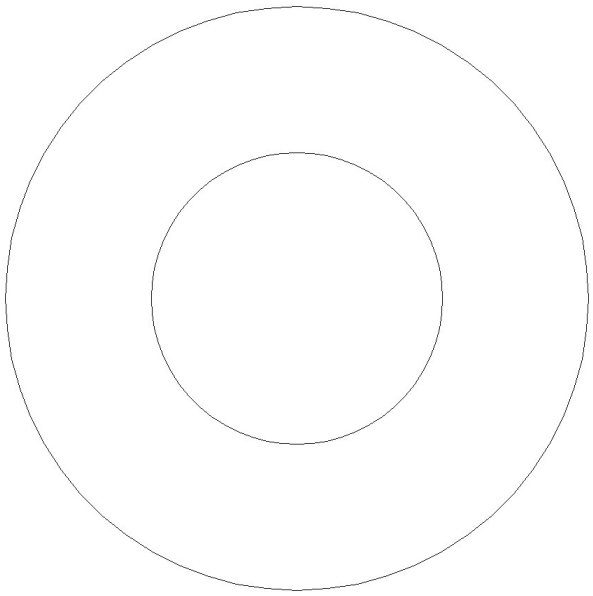
Arguments:

- cx = center X
- cy = center Y
- outerRadius = outside radius
- innerRadius = inside radius

Script:

donut 50 50 20 10

Result:



select_all

select_all

Description:

Selects all objects in the drawing.

Script:

```
rectangle 10 10 40 20
circle 80 40 15
select_all
move 20 0
```

Result:

All objects are selected and moved.

select_last

select_last

Description:

Selects the last created objects.

Script:

```
rectangle 10 10 40 20
circle 80 40 15
select_last
move 0 20
```

Result:

The last created object is moved.

select_window

select_window xMin yMin xMax yMax

Description:

Selects objects inside a rectangular selection area.

Arguments:

- xMin = minimum X
- yMin = minimum Y
- xMax = maximum X
- yMax = maximum Y

Script:

```
rectangle 10 10 40 20
circle 80 40 15
select_window 0 0 50 50
move 20 0
```

Result:

Only objects inside the selection window are selected.

delete

delete

Description:

Deletes the selected objects.

Script:

```
rectangle 10 10 40 20
select_all
delete
```

Result:

The selected objects are deleted.

move

move dx dy

Description:

Moves the selected objects.

Arguments:

- dx = movement in X direction
- dy = movement in Y direction

Script:

```
rectangle 10 10 40 20
select_all
move 30 10
```

Result:

The object is moved.

rotate

rotate centerX centerY angle

Description:

Rotates the selected objects around a base point.

Arguments:

- centerX = rotation center X
- centerY = rotation center Y
- angle = rotation angle in degrees

Script:

```
rectangle 20 20 40 20
select_all
rotate 40 30 45
```

Result:

The object is rotated by 45°.

mirror

mirror x1 y1 x2 y2

Description:

Mirrors the selected objects using a mirror axis.

Arguments:

- x1 = first axis point X
- y1 = first axis point Y
- x2 = second axis point X
- y2 = second axis point Y

Script:

```
triangle 10 10 60 10 35 50
select_all
mirror 35 0 35 100
```

Result:

The object is mirrored.

scale

scale baseX baseY scaleX scaleY

Description:

Scales the selected objects.

Arguments:

- baseX = base point X
- baseY = base point Y
- scaleX = scale factor in X direction
- scaleY = scale factor in Y direction

Script:

```
rectangle 20 20 40 20
select_all
scale 40 30 1.5 1.5
```

Result:

The object is scaled by 1.5 times.

set_color

set_color index

Description:

Sets the current drawing color.

Arguments:

- index = color index (0–255)

Script:

```
set_color 2
rectangle 10 10 40 20
set_color 4
circle 80 40 15
```

Result:

Objects are drawn using different colors.

set_width

set_width value

Description:

Sets the current line width.

Arguments:

- value = line width (1–255)

Script:

```
set_width 1
line 10 10 80 10
set_width 5
line 10 20 80 20
```

Result:

Lines are drawn with different widths.

set_style

set_style index

Description:

Sets the current line style.

Arguments:

- index = line style index

Script:

```
set_style 0
line 10 10 80 10
set_style 1
line 10 20 80 20
```

Result:

Lines are drawn using different styles.

set_layer

set_layer index

Description:

Sets the current drawing layer.

Arguments:

- index = layer number (0–255)

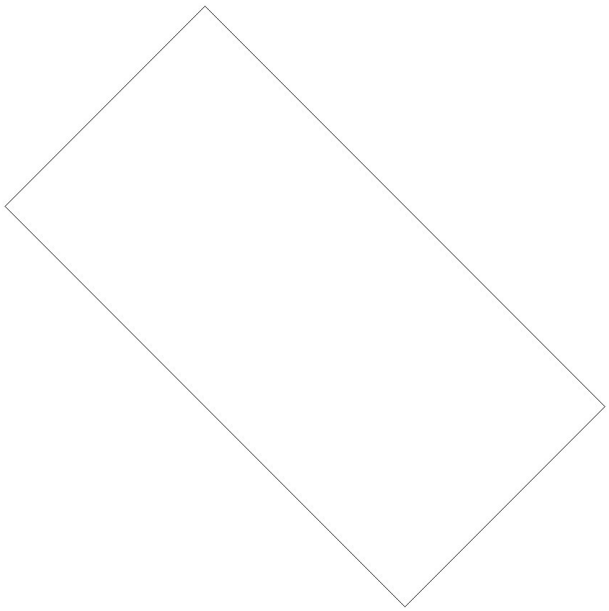
Script:

```
set_layer 1
rectangle 10 10 40 20
set_layer 2
circle 80 40 15
```

Result:

Objects are created on different layers.

```
' ================================================
' EXAMPLE 1
' ================================================

' create rectangle
rectangle 10 10 40 20

' select last created entities
select_last

' move by X=20 Y=10
move 20 10

' rotate around center (50,30) by 45 degrees
rotate 50 30 45

' mirror using vertical axis
mirror 50 0 50 100

' scale from center point
scale 50 30 1.5 1.5
```



```
' ================================================
' EXAMPLE 2
' ================================================

' SETTINGS

set_color 2
set_width 1
set_style 0
set_layer 1


' BASIC SHAPES


rectangle 10 10 60 40

circle 120 30 20

ellipse 180 30 25 12

triangle 240 10 300 10 270 60

rounded_rectangle 320 10 80 50 10

slot 430 35 520 35 8

star 600 35 30 15 5

donut 700 35 25 12


' GRID


set_color 8

grid 0 0 750 80 10


' POLYLINE + HATCH
```

```
set_color 4

dim x(4)
dim y(4)

x(0)=100
y(0)=120

x(1)=160
y(1)=150

x(2)=130
y(2)=210

x(3)=70
y(3)=180

x(4)=100
y(4)=120

polyline x y

hatch x y 5 45 0 1 0


' ARC


set_color 1

arc 250 160 300 160 250 210


' TEXT


text "SCRIPT EXAMPLE" 20 260 6


' DIMENSIONS


set_color 3
```

linear_dimension 10 10 70 10 8 0 -1 2

angular_dimension 250 160 300 160 250 160 250 210 285 195 3

all_dimensions 6 1 2

' SELECTION + TRANSFORMATIONS

select_window 300 0 750 100

move 0 120

rotate 500 150 15

scale 500 150 1.2 1.2

' MIRROR EXAMPLE

select_last

mirror 500 0 500 400