

AutoLISP

Module 12

Measurement

Why is it **Important** for you to learn this task?

RATIONALE:

A large amount of your time programming with AutoLISP is spent measuring distances and angles as well as finding XY coordinate locations. This is especially important because of the graphic nature of CADD models and drawings.

Here is what you will be able to do when you complete each **Step** of this learning activity package:

OBJECTIVE(S):

1. Describe and apply the DISTANCE function.
Describe and apply the ANGLE function.
Describe and apply the POLAR function.
Describe and apply the INTERS function

To show that you have **Mastered** this task, here is what you will be asked to do:

PERFORMANCE EVALUATION:

1. Complete Self-Test No. 1 with 100% accuracy.
2. Complete Lab Exercise L1200-01 with 100% accuracy.
3. Complete Lab Exercise L1200-02 with 100% accuracy.
4. Complete Lab Exercise L1200-03 with 100% accuracy.

OBJECTIVE NO. 1

When you complete this objective, you will be able to:

Describe and apply the DISTANCE function.
Describe and apply the ANGLE function.
Describe and apply the POLAR function.
Describe and apply the INTERS function

Complete each of the learning activities listed below.

LEARNING ACTIVITIES

DO the following things:

USE the following resources:

1. Read Information Sheet No. 1.
2. Complete Self-Test No. 1.
3. Check your answers to Self-Test No. 1 and correct any errors.
4. Complete Lab Exercise L1200-01.
5. Check your answers to Lab Exercise L1200-01 and correct any errors.
6. Complete Lab Exercise L1200-02.
7. Check your answers to Lab Exercise L1200-02 and correct any errors.
8. Complete Lab Exercise L1200-03.
9. Check your answers to Lab Exercise L1200-03 and correct any errors.

INFORMATION SHEET NO. 1

AutoLISP Function - DISTANCE

AutoCAD Release: All

DESCRIPTION

The DISTANCE function is used to find the distance between two coordinate points.

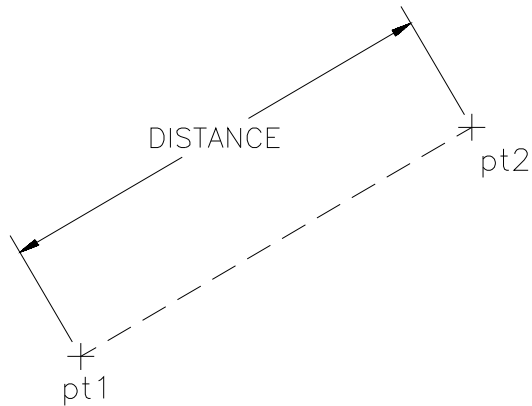
If both coordinate points are 3D points, DISTANCE will return the 3D distance between the points. If either or both of the coordinate points are 2D points, DISTANCE will return the 2D distance between the points, projected onto the current construction plane.

RETURNS

A real number.

FORMAT

(distance *pt1 pt2*)



EXAMPLES

Example No. 1

```
Command: (setq dist1 (distance (list 2.0 2.0 0.0) (list 4.0 2.0 2.0)))
```

```
2.82843
```

```
Command:
```

{In this example, the DISTANCE function is supplied with two 3D points. It returns a real number which is the 3D distance between the points.}

Example No. 2

```
Command: (setq dist2 (distance (list 2.0 2.0 0.0) (list 4.0 2.0)))
```

```
2.0
```

```
Command:
```

{In this example, the points are the same as in Example No. 1 except that one of the points was changed to a 2D point. Notice how the distance returned has changed to the 2D distance between the points.}

Example No. 3

```
(setq pnt1 (getpoint "\nEnter the first point: "))
```

```
(setq pnt2 (getpoint pnt1 "\nEnter the second point: "))
```

```
(setq dist3 (distance pnt1 pnt2))
```

{This example shows a portion of a program that can be used to obtain two points from the user and then find the distance between them.}

AutoLISP Function - ANGLE

AutoCAD Release: All

DESCRIPTION

The ANGLE function is used to find the angle of an imaginary line between two coordinate points.

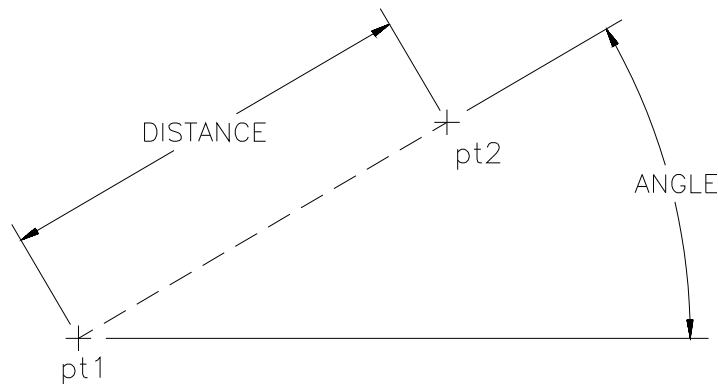
The angle is measured from the X axis, on the active construction plane, counterclockwise. If either or both of the coordinate locations is a 3D coordinate, the points are projected onto the active construction plane.

RETURNS

The angle in radians.

FORMAT

(angle *pt1 pt2*)



EXAMPLES

Example No. 1

```
Command: (setq ang1 (angle (list 2.0 2.0 2.0) (list 4.0 4.0 4.0)))
```

```
0.785398
```

```
Command:
```

{This example supplies two 3D points and AutoLISP returns the angle 0.785398 radians which is 45 degrees.}

Example No. 2

```
Command: (setq ang2 (angle (list 2.0 2.0 ) (list 4.0 4.0 4.0)))
```

```
0.785398
```

```
Command:
```

{The ANGLE function in this example returns the angle 0.785398 radians. One of the points is a 2D point and the other is a 3D point. Note how it still returns the same angle 45 degrees.}

Example No. 3

```
(setq pnt1 (getpoint "\nEnter the first point: "))
```

```
(setq pnt2 (getpoint pnt1 "\nEnter the second point: "))
```

```
(setq ang3 (angle pnt1 pnt2))
```

{This example shows a portion of a program that obtains two points from the user and then calculates the angle between the two points.}

AutoLISP Function - POLAR

AutoCAD Release: All

DESCRIPTION

The POLAR function is used to find a coordinate point using a known coordinate point *pt* plus the *angle* and the *distance* from that point.

The known coordinate point *pt* can be a 2D or 3D point. If it is a 3D point, it is projected onto the active construction plane. If the known point is a 2D point, POLAR returns a 2D point, but, if the point is 3D point, POLAR returns a 3D point.

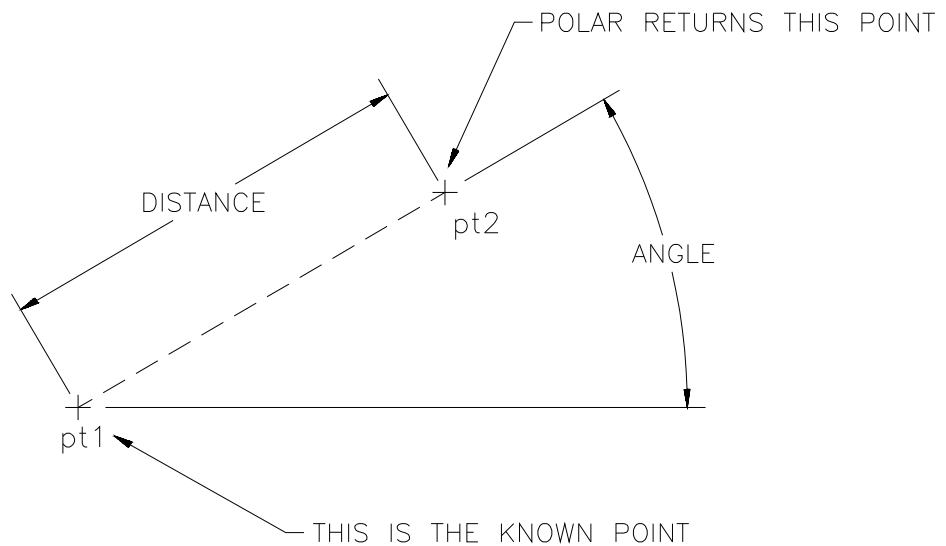
The *angle* must be expressed in radians from the X axis counterclockwise.

RETURNS

A list of three reals or a 3D point.

FORMAT

(polar *pt angle distance*)



EXAMPLES

Example No. 1

Command: (setq pnt1 (polar (list 2.0 2.0) 0.785398 1.414214))

(3.0 3.0)

Command:

{In this example, the known point is a 2D point, the angle is 45 degrees, and the distance is 1.414214. POLAR returns the 2D point '(3.0 3.0)'.}

Example No. 2

Command: (setq pnt1 (polar (list 2.0 2.0 2.0) 0.785398 1.414214))

(3.0 3.0 2.0)

Command:

{In this example, the known point is a 3D point, the angle is 45 degrees, and the distance is 1.414214. POLAR returns the 3D point '(3.0 3.0 2.0)'.}

Example No. 3

```
; ***** AutoCAD Modules *****
; A1200-01.LSP (v12)      Written by: J. Smith      040215
; *****
; This program will draw a line one-half as long as the endpoint
; supplied by the user.
;
; ***** Function - HALF *****
(defun C:HALF (/ svcm pnt1 pnt2 pnt3 ang1 dist1 dist2)
  (setq svcm (getvar "cmdecho"))
  (setvar "cmdecho" 0)
  (setq pnt1 (getpoint "\nEnter start point of line: "))
  (setq pnt2 (getpoint pnt1 "\nEnter end point of line: "))
  (setq dist1 (distance pnt1 pnt2))
  (setq ang1 (angle pnt1 pnt2))
  (setq dist2 (/ dist1 2))
  (setq pnt3 (polar pnt1 ang1 dist2))
  (command "LINE" pnt1 pnt3 "")
  (setvar "cmdecho" svcm)
  (princ)
)
; ***** The End *****
```

{Notice how this program uses the DISTANCE, ANGLE, and POLAR functions to find a point that is half-way along the line specified by the user.}

Example No. 4

```
; ***** AutoCAD Modules *****
; A1200-02.LSP (v12)      Written by: J. Smith      040215
; *****
; This program will ask the user to draw a line and then it draws a line parallel to that line.
; It does not use the OFFSET command.
; ***** Function - d2r *****
(defun d2r (anga)
  (* pi (/ anga 180.0))
)
; ***** Function - PARALINE *****
(defun C:PARALINE (/ svcm pnt1 pnt2 pnt3 pnt4 ang1 dist1)
  (setq svcm (getvar "cmdecho"))
  (setvar "cmdecho" 0)
  (setq pnt1 (getpoint "\nEnter start point of line: "))
  (setq pnt2 (getpoint "\nEnter end point of line: "))
  (setq dist1 (distance pnt1 pnt2))
  (setq ang1 (angle pnt1 pnt2))
  (command "LINE" pnt1 pnt2 "")
  (setq pnt3 (polar pnt1 (+ ang1 (d2r 90)) 1))
  (setq pnt4 (polar pnt2 (+ ang1 (d2r 90)) 1))
  (command "LINE" pnt3 pnt4 "")
  (setvar "cmdecho" svcm)
  (princ)
)
; ***** The End *****
```

{There is an important principle in this program. Pay particular attention to the way POLAR is used to find pnt3 and pnt4.}

AutoLISP Function - INTERS

AutoCAD Release: All

DESCRIPTION

The INTERS function is used to find the real or the imaginary intersection point of two lines.

The *pt1* and the *pt2* arguments are endpoints of the first line and the *pt3* and *pt4* arguments are the endpoints of the second line.

All points are considered to be in the active construction plane. If all the points are 3D points, INTERS will find the 3D intersections on the two lines. If any of the points are 2D, INTERS will project all the points onto the active construction plane and then find the 2D intersection of the two lines.

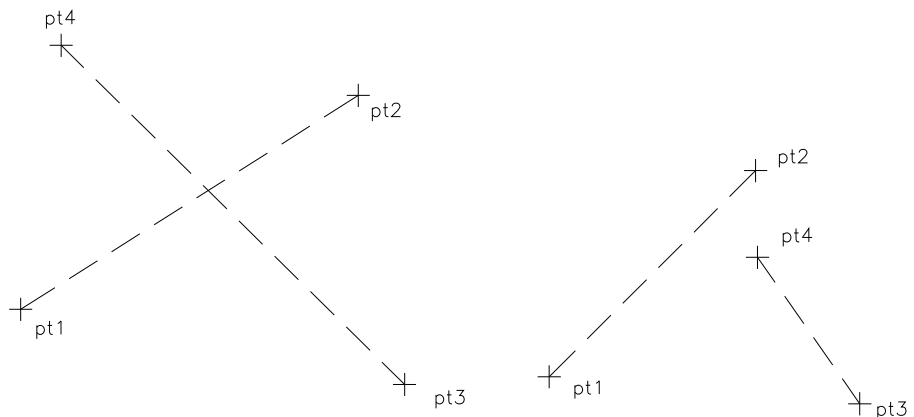
If the option *onseg* is present and is nil, the lines defined by the four points are considered to be indefinite in length and INTERS will find the imaginary intersection point if there is no actual intersection point. If *onseg* is omitted or is not nil, the lines must actually intersect or INTERS will return nil.

RETURNS

The 2D point, list of 2 reals, or a 3D point, list of 3 reals, or nil if the lines do not actually intersect or do not potentially intersect.

FORMAT

(inters *pt1 pt2 pt3 pt4*
[*onseg*])



EXAMPLES

Example No. 1

Command: (setq pnt5 (inters (list 2.0 2.0 2.0) (list 4.0 4.0 4.0) (list 4.0 2.0 2.0) (list 2.0 4.0 4.0)))
(3.0 3.0 3.0)

{Used this way, INTERS returns the 3D coordinate point '(3.0 3.0 3.0)'. Both lines are defined by 3D coordinate locations, therefore, INTERS will return a 3D point. Since onseg is omitted, the lines actually intersect.}

Example No. 2

Command: (setq pnt6 (inters (list 2.0 2.0) (list 4.0 4.0) (list 4.0 2.0 2.0) (list 2.0 4.0 4.0)))
(3.0 3.0)

{In this example, one line is defined by 2D points, therefore, INTERS will return the intersection point as a 2D point. Both lines will be projected onto the active construction plane to calculate the intersection point. Since onseg is omitted, the lines actually intersect.}

Example No. 3

```
Command: (setq pnt7 (inters (list 2.0 2.0) (list 4.0 4.0) (list 2.0 4.0) (list 4.0 5.0)))
nil
Command:
```

{In this example, the two lines do not actually intersect, therefore, INTERS returns nil.}

Example No. 4

```
Command: (setq pnt7 (inters (list 2.0 2.0) (list 4.0 4.0) (list 2.0 4.0) (list 4.0 5.0) nil ))
(6.0 6.0)
Command:
```

{In this example, the lines do not actually intersect (same lines as Example No. 3) but since *onseg* is present, and is nil, INTERS finds the intersection point as if the lines were indefinite length.}

Example No. 5

```
Command: (setq pnt7 (inters (list 2.0 2.0) (list 4.0 4.0) (list 2.0 4.0) (list 4.0 5.0) T ))
nil
Command:
```

{In this example, the *onseg* is present but is not nil, therefore, INTERS returns the actual intersection point. In this case, the lines do not actually intersect.}

Example No. 6

```
; ***** AutoCAD Modules *****
; A1200-03.LSP (v12)      Written by: J. Smith                      040216
; *****
; This program will ask the user to draw two lines and then find the
; actual or imaginary intersection point between the lines. It will
; insert a point at the intersection location.
;
; ***** Function - INTPNT *****
(defun C:INTPNT (/ svcm svpm pnt1 pnt2 pnt3 pnt4 pnt5)
  (setq svcm (getvar "cmdecho"))
  (setvar "cmdecho" 0)
  (setq svpm (getvar "pdmode"))
  (setvar "pdmode" 34)
  (setq pnt1 (getpoint "\nEnter start point of first line: "))
  (setq pnt2 (getpoint pnt1 "\nEnter end point of first line: "))
  (command "LINE" pnt1 pnt2 "")
  (setq pnt3 (getpoint "\nEnter start point of second line: "))
  (setq pnt4 (getpoint pnt3 "\nEnter end point of second line: "))
  (command "LINE" pnt3 pnt4 "")
  (setq pnt5 (inters pnt1 pnt2 pnt3 pnt4 nil))
  (command "POINT" pnt5 )
  (setvar "cmdecho" svcm)
  (princ)
)
; ***** The End *****
```

SELF-TEST NO. 1

DIRECTIONS

1. Answer the following questions.
2. Compare your answers to the enclosed answer key.
3. If you disagree with any of the answers, review the learning material and/or check with your instructor.
4. If no problems arise, continue with the next step.

1. If either or both of the coordinate points are 2D points, the DISTANCE function will return the _____ distance between the points.
2. The _____ function is used to find the real or the imaginary intersection point between two lines.
3. The _____ function is used to find the angle of an imaginary line between two coordinate points.
4. The POLAR function is used to find a coordinate point, using a known _____ location plus the _____ and the _____ from that point.
5. If all the given points are 3D points, _____ will find the 3D intersections of the two lines.
6. The _____ function is used to find the distance between two coordinate points.

LAB EXERCISE NO. 1

Lab Exercise L1200-01

Description

Program Name: L1200-01.LSP

1. Function 1 Name: RANGLE

Purpose: To draw a rectangle or square by allowing the user to select two coordinate locations on the drawing. The coordinate locations are on the center line of the rectangle or square. The rectangle or square must be able to be placed at any angle.

- a) Change the screen to a graphic screen.
- b) Prompt the user to select or enter the first point.
- c) Prompt the user to select or enter the second point.
- d) Rubber-band the center line of the rectangle or square.
- e) Use the LINE command to draw the lines.
- f) Redraw the screen.
- g) Print the message RECTANGLE COMPLETE on the screen.

Program Example

Command: **RANGLE**

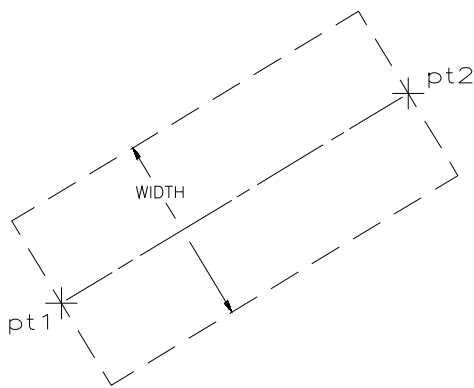
First point:

Second point:

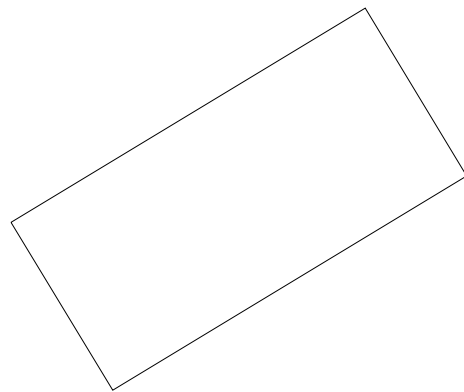
Enter width: 2.5

RECTANGLE COMPLETE

Command:



STEP 1



STEP 2

LAB EXERCISE NO. 2

Lab Exercise L1200-02

Description

Program Name: L1200-02.LSP

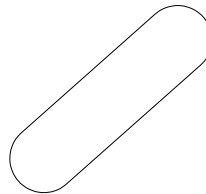
1. Function 1 Name: SLOT

Purpose: To draw a slot after allowing the user to enter two points that are the center locations for a slot. The program should also ask the user for the width of the slot. The user should be able to draw the slot at any angle.

- a) Change the screen to a graphic screen.
- b) Prompt the user to select the first center and then the second center of the slot.
- c) Prompt the user to enter the width of the slot.
- d) Rubber-band the center line of the slot.
- e) Use the LINE command to draw the lines and the ARC command to draw the arcs.
- f) Redraw the screen.
- g) Print the message SLOT COMPLETE on the screen.

Program Example

Command: **SLOT**
Center of first end of slot:
Center of other end of slot:
Enter width of slot: .2
SLOT COMPLETE
Command:



LAB EXERCISE NO. 3

Lab Exercise L1200-03

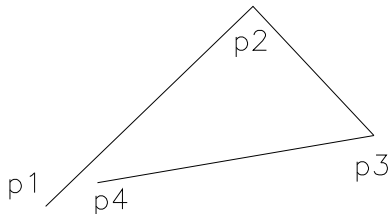
Description

Program Name: L1200-03.LSP

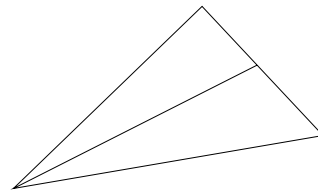
1. Function 1 Name: TRIANGLE

Purpose: To draw a triangle and a line inside the triangle that bisects the triangle into two equal parts.

- a) Change the screen to a graphic screen.
- b) Prompt the user to enter or select four coordinate locations. The first line ' p1 to p2 ' and a second line ' p3 to p4 ' may or may not intersect.



STEP 1



STEP 2

- c) Rubber-band and draw the lines as shown in STEP 1.
- d) Find the new ' p1 'and ' p4 ', the intersection point. Erase the original lines ' p2 ' to ' p1 ' and ' p3 ' to ' p4 ' and then redraw the new lines.
- e) Draw a third line that bisects the triangle.

Program Example

```
Command: TRIANGLE
Enter start point of triangle:
Enter second point of triangle:
Enter third point of triangle:
Enter fourth point of triangle:
Command:
```

ANSWER KEY

SELF-TEST # 1

1. If either or both of the coordinate points are 2D points, the **DISTANCE** function will return the **2D** distance between the points.
2. The **INTERS** function is used to find the real or the imaginary intersection point between two lines.
3. The **ANGLE** function is used to find the angle of an imaginary line between two coordinate points.
4. The **POLAR** function is used to find a coordinate point, using a known **coordinate** location plus the **angle** and the **distance** from that point.
5. If all the given points are 3D points, **INTERS** will find the 3D intersections of the two lines.
6. The **DISTANCE** function is used to find the distance between two coordinate points.

LAB EXERCISE No. 1 - L1200-01

```
; ***** AutoCAD Modules *****
; L1200-01.LSP (v12)      Written by: J. Smith      040217
; *****
; This program will draw a rectangle or square by asking the user for two points
; which if joined together would form the centerline of the rectangle or square.
; The centerline should rubber-band for the user.
; ***** Function - d2r *****
(defun d2r (angl)
  (* pi (/ angl 180.0))
)
; ***** Function - C:RANGLE *****
(defun C:RANGLE (/ svcm poi1 poi2 widt hwid lina cor1 cor2 cor3 cor4 )
  (setq svcm (getvar "cmdecho"))
  (setvar "cmdecho" 0)
  (graphscr)
  (setq poi1 (getpoint "\nFirst point: "))
  (setq poi2 (getpoint poi1 "\nSecond point: "))
  (setq widt (getreal "\nEnter width: "))
  (setq hwid (/ widt 2))
  (setq lina (angle poi1 poi2))
  (setq cor1 (polar poi1 (+ lina (d2r 90)) hwid))
  (setq cor2 (polar poi2 (+ lina (d2r 90)) hwid))
  (setq cor3 (polar poi1 (- lina (d2r 90)) hwid))
  (setq cor4 (polar poi2 (- lina (d2r 90)) hwid))
  (command "LINE" cor1 cor2 cor4 cor3 "c")
  (setvar "cmdecho" svcm)
  (redraw)
  (princ "RECTANGLE COMPLETE")
  (princ)
)
; ***** The End *****
```

LAB EXERCISE No. 2 - L1200-02

```
; ***** AutoCAD Modules *****
; L1200-02.LSP (v12)      Written by: J. Smith      040217
; *****
;
; This program will draw a slot. It will ask the user for the
; center of one end of the slot, center of the other end of the
; slot, and the width of the slot. The user must be able to
; draw the slot at any angle.
;
; ***** Function - d2r *****
; converts degrees to radians
;
(defun d2r (angl)
  (* pi (/ angl 180.0))
)
;
; ***** Function - C:SLOTS *****
;
(defun C:SLOT (/ svcm cen1 cen2 hwid lina end1 end2 end3 end4)
  (setq svcm (getvar "cmdecho"))
  (setvar "cmdecho" 0)
  (graphscr)
  (setq cen1 (getpoint "\nCenter of first end of slot: "))
  (setq cen2 (getpoint cen1 "\nCenter of other end of slot: "))
  (setq hwid (/ (getreal "\nEnter width of slot: ") 2))
  (setq lina (angle cen1 cen2))
  (setq end1 (polar cen1 (+ lina (d2r 90)) hwid))
  (setq end2 (polar cen2 (+ lina (d2r 90)) hwid))
  (setq end3 (polar cen1 (- lina (d2r 90)) hwid))
  (setq end4 (polar cen2 (- lina (d2r 90)) hwid))
  (command "LINE" end1 end2 "")
  (command "LINE" end3 end4 "")
  (command "ARC" "c" cen1 end1 end3)
  (command "ARC" "c" cen2 end4 end2)
  (redraw)
  (setvar "cmdecho" svcm)
  (princ "SLOT COMPLETE")
  (princ)
)
;
; ***** The End *****
```

EXERCISE No. 3 - L1200-03

```
; ***** AutoCAD Modules *****
; L1200-03.LSP (v12)           Written by: J. Smith           040215
; *****
; This program will prompt the user to draw a triangle which does not
; have to close. It will then draw a line that bisects the triangle.
;
; ***** Function - TRIANGLE *****
(defun C:TRIANGLE (/ svcm pnt1 pnt2 pnt3 pnt4 dist1 ang1 dist2)
  (setq svcm (getvar "cmdecho"))
  (setvar "cmdecho" 0)
  (setq pnt1 (getpoint "\nEnter start point of triangle: "))
  (setq pnt2 (getpoint pnt1 "\nEnter second point of triangle: "))
  (command "LINE" pnt1 pnt2 "")
  (setq pnt3 (getpoint pnt2 "\nEnter third point of triangle: "))
  (command "LINE" pnt2 pnt3 "")
  (setq pnt4 (getpoint pnt3 "\nEnter fourth point of triangle: "))
  (command "LINE" pnt3 pnt4 "")
  (setq pnt1 (inters pnt1 pnt2 pnt3 pnt4 nil))
  (command "ERASE" "1" "")
  (command "ERASE" "1" "")
  (command "ERASE" "1" "")
  (command "LINE" pnt1 pnt2 pnt3 pnt1 "")
  (setq dist1 (distance pnt2 pnt3))
  (setq ang1 (angle pnt2 pnt3))
  (setq dist2 (/ dist1 2))
  (setq pnt4 (polar pnt2 ang1 dist2))
  (command "LINE" pnt1 pnt4 "")
  (redraw)
  (setvar "cmdecho" svcm)
  (princ)
)
; ***** The End *****
```