

RhinoScript 101 Creativity

"or how to do interesting things that are not easy to do with the mouse"

Nevermind the code...

' Copy And paste this code In your RhinoScript Editor (Tools □ RhinoScript □ Edit...)

' This Is a basic script To draw a curve with fixed coordinates (Not very useful, but a good starting point)

Option Explicit ' nevermind this, just make sure that your scripts always start With it

DrawCurve ' this tells the program what subroutine To run

Sub DrawCurve ' this Is the code To Execute when “DrawCurve” Is called above

Call Rhino.enableRedraw(False) ' nevermind this, it speeds up the execution of the code

Dim controlpoints(1) ' controlpoints is an array of 3-D points (see next slide)

controlpoints(0) = Array(0,0,0) ' x = 0, y = 0, z = 0

controlpoints(1) = Array(10,5,15) ' x = 10, y = 5, z = 15

Rhino.Print ("Curve ID: " + Rhino.AddCurve(controlpoints)) ' this draws the curve and prints its I.D.

Rhino.Print ("Sphere ID: " + Rhino.AddSphere (controlpoints(1), 1)) ' this draws a sphere and prints its I.D.

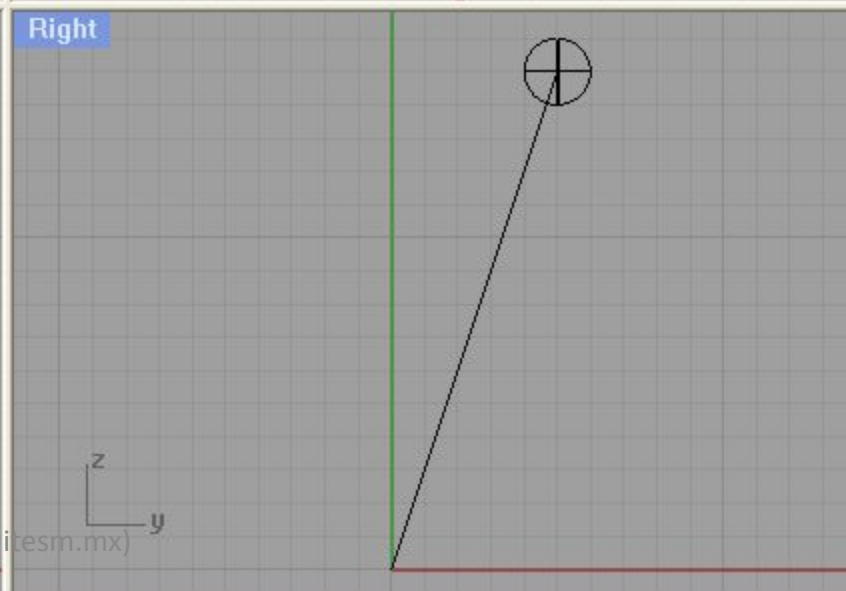
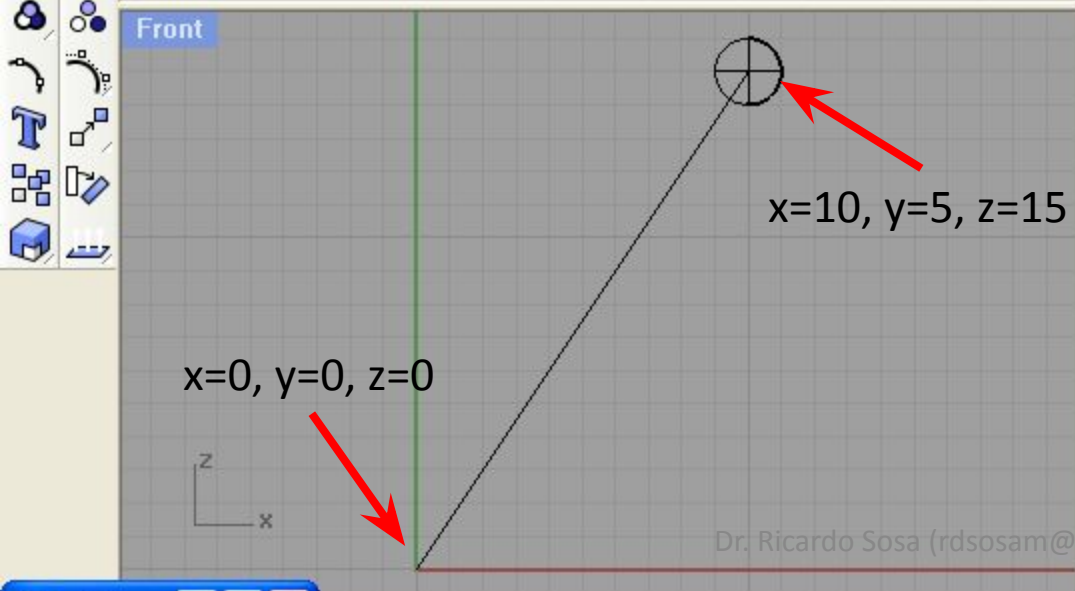
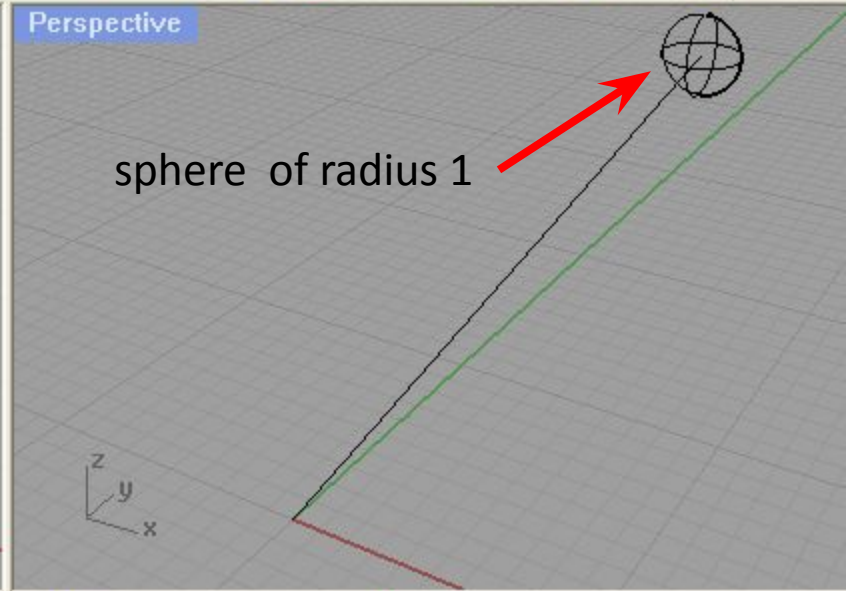
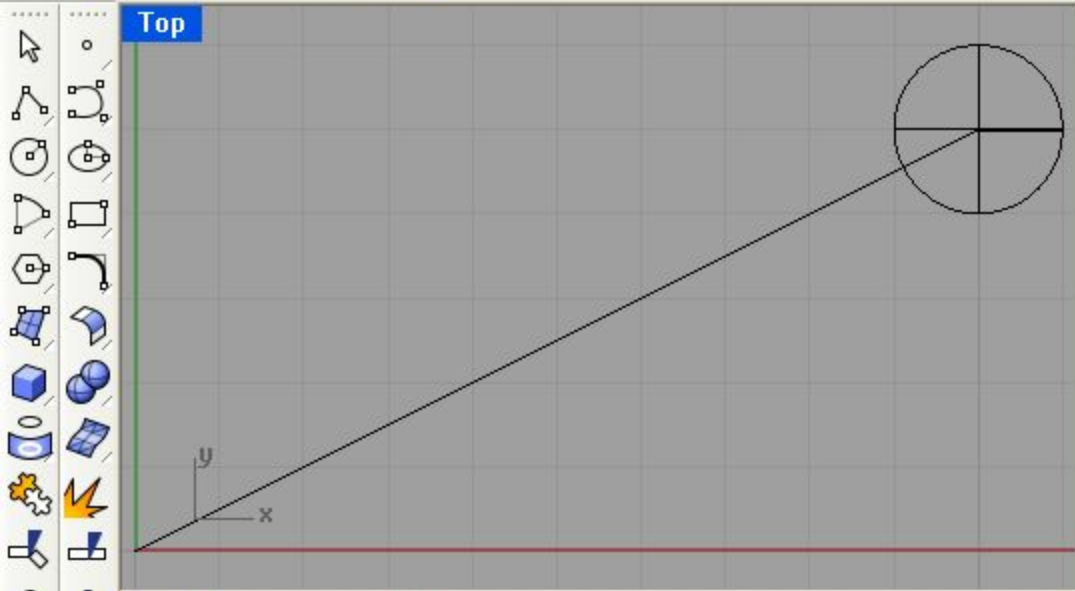
Call Rhino.enableRedraw(True) ' nevermind this, it refreshes the screen

Rhino.ZoomExtents ' and this adjusts the zoom level

End Sub ' this is the end of the "DrawCurve" subroutine

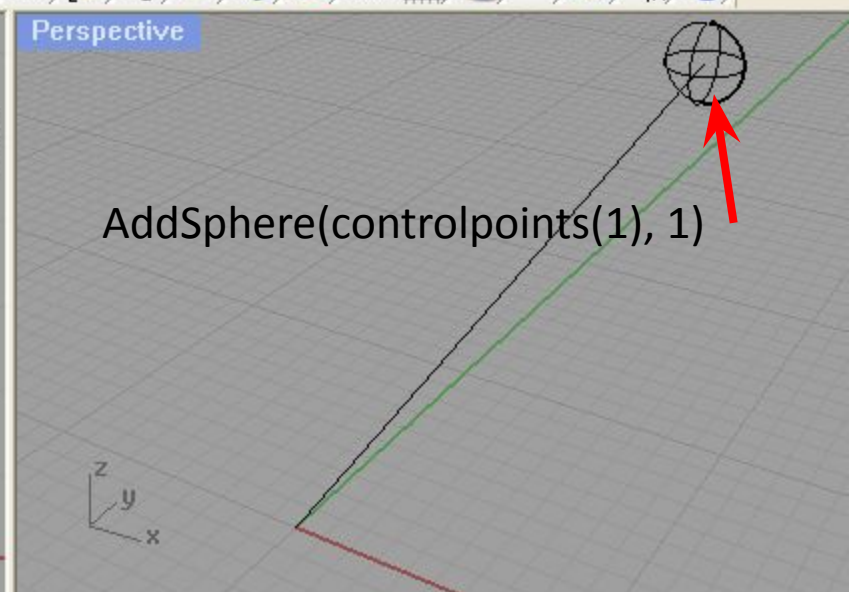
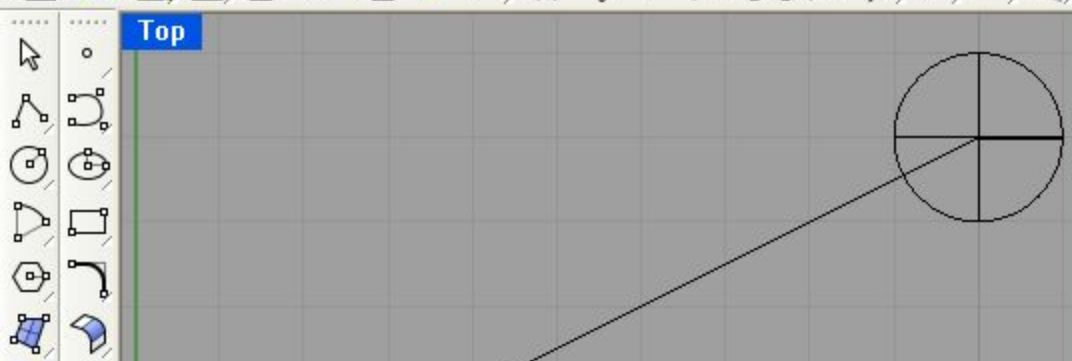
Drag a window to zoom (All Dynamic Extents Factor In Out Selected Target 1To1): _All

Command:



Drag a window to zoom (All Dynamic Extents Factor In Out Selected Target 1To1): _All

Command:

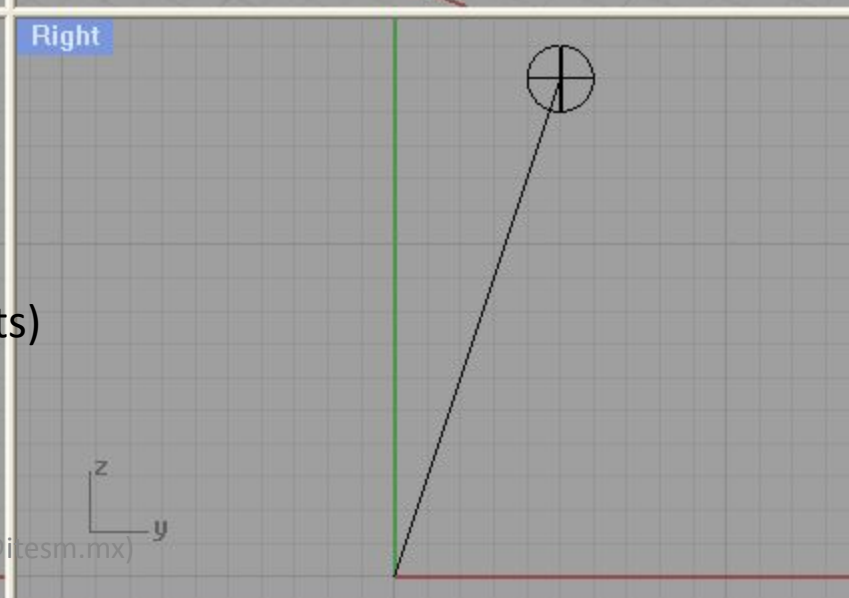
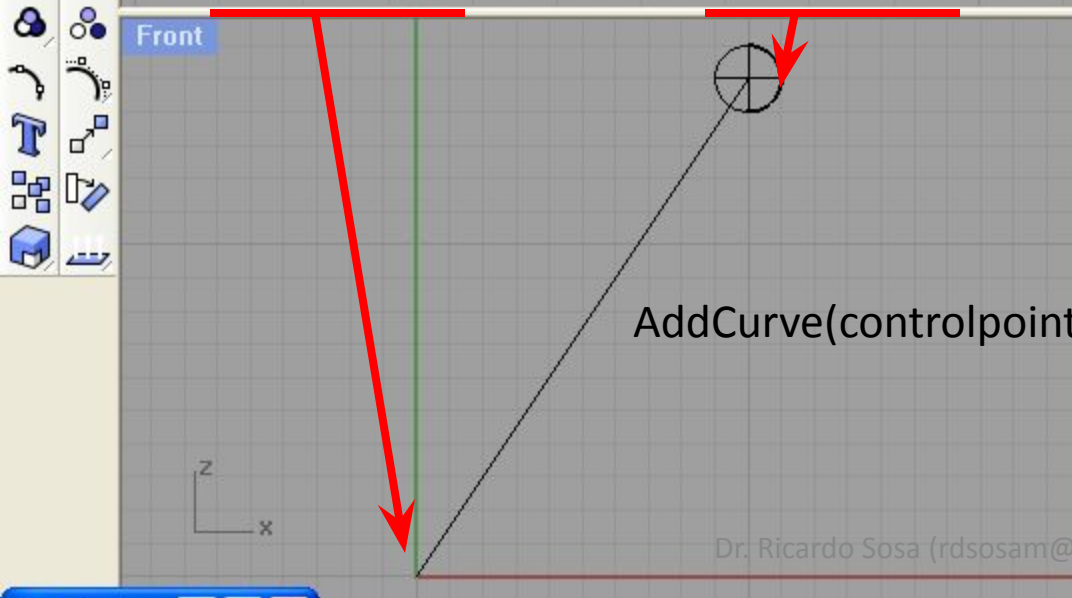


controlpoints()

controlpoints(0)

controlpoints(1)

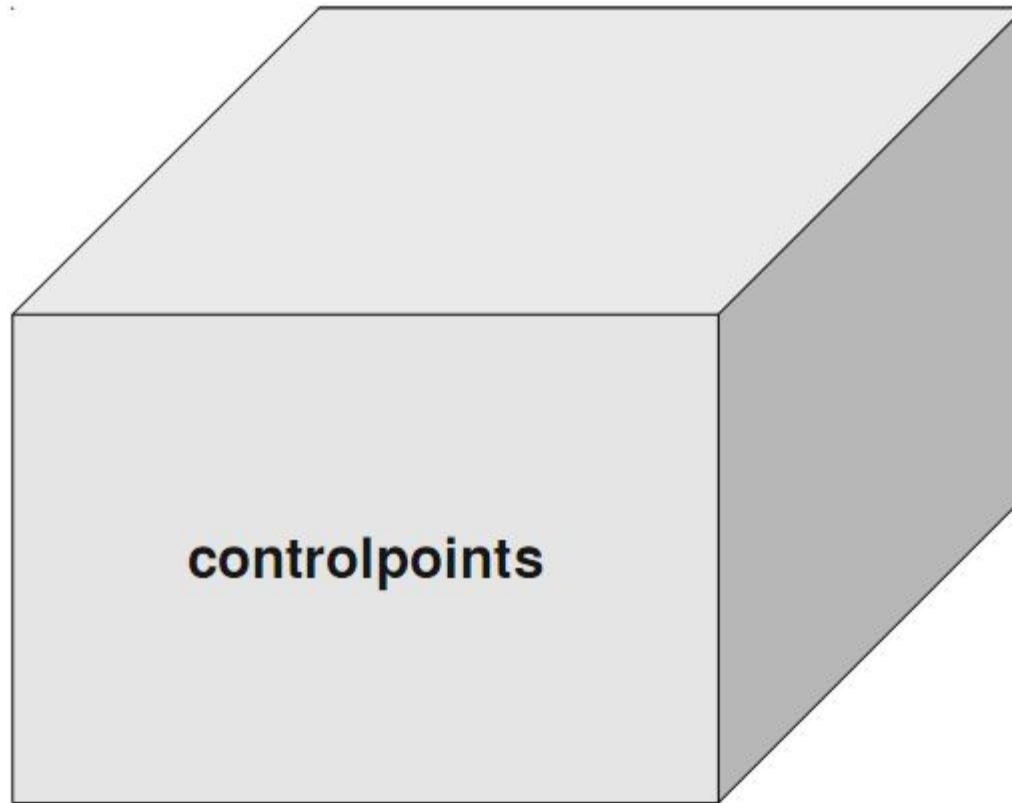
x0	y0	z0	x1	y1	z1
0	0	0	10	5	15



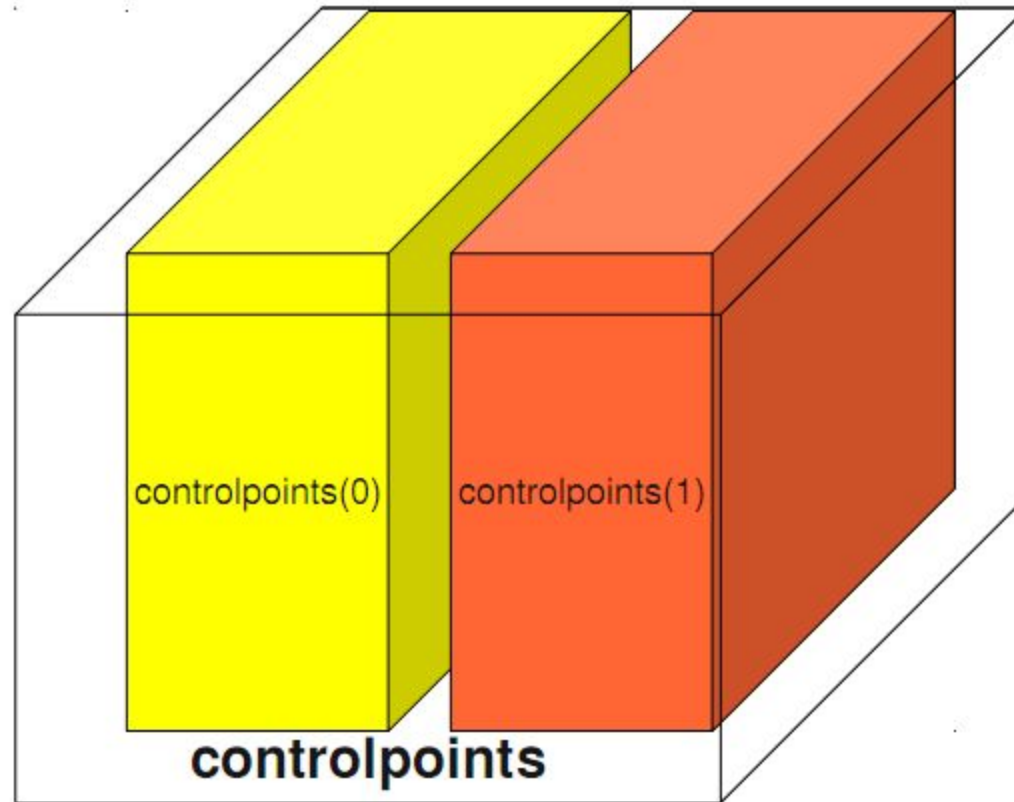
AddSphere(controlpoints(1), 1)

AddCurve(controlpoints)

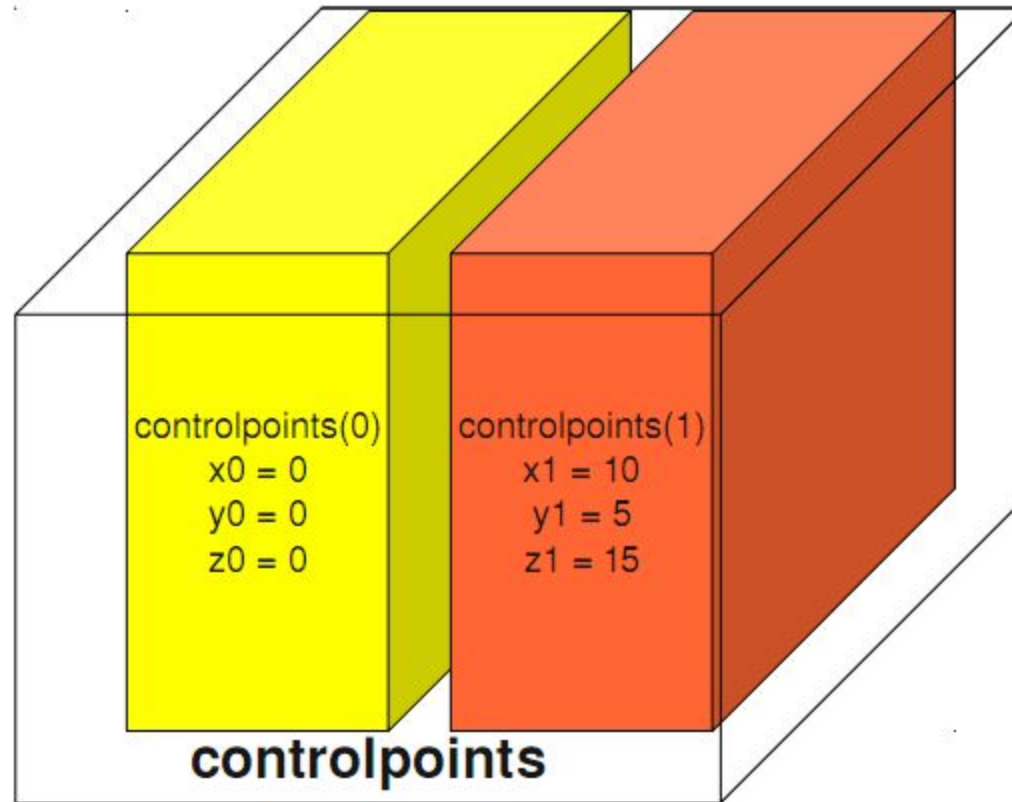
Hold on! What's an "array"?



An array is like a box



A box that holds 3D coordinates



' Copy and paste this code in your RhinoScript Editor (Tools ☐ RhinoScript ☐ Edit...)

' This is a basic script to draw a curve with fixed coordinates (Not very useful, but a good starting point)

Option Explicit ' nevermind this, just make sure that your scripts always start with it

DrawCurve ' this tells the program what subroutine to run

Sub DrawCurve ' this is the code to run when "DrawCurve" is called above

Call Rhino.enableRedraw(False) ' nevermind this, it speeds up the execution of the code

Dim controlpoints(1) ' controlpoints is an array of 3-D points (see next slide)

controlpoints(0) = Array(0,0,0) ' x = 0, y = 0, z = 0

controlpoints(1) = Array(10,5,15) ' x = 10, y = 5, z = 15

Rhino.Print ("Curve ID: " + **Rhino.AddCurve(controlpoints)**) ' this draws the curve and prints its I.D.

Rhino.Print ("Sphere ID: " + **Rhino.AddSphere (controlpoints(1), 1)**) ' this draws a sphere and its I.D.

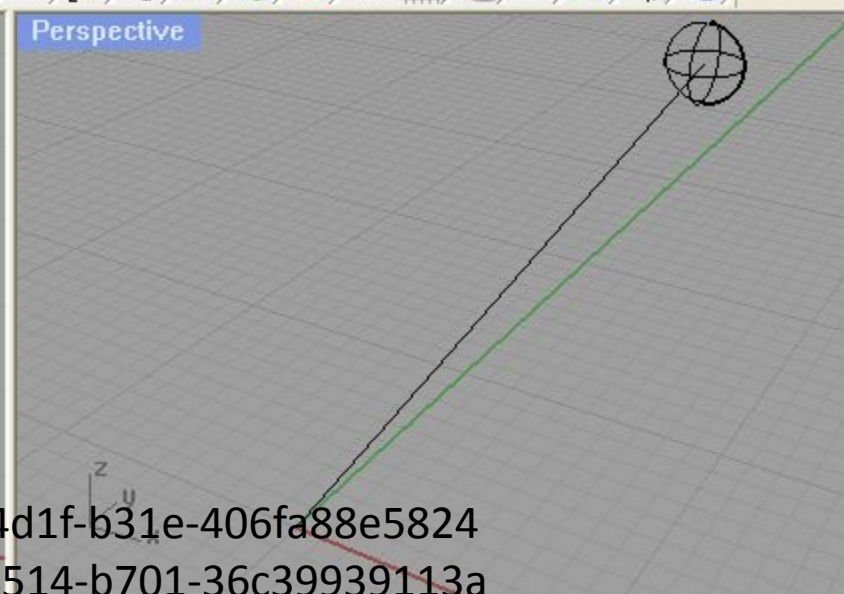
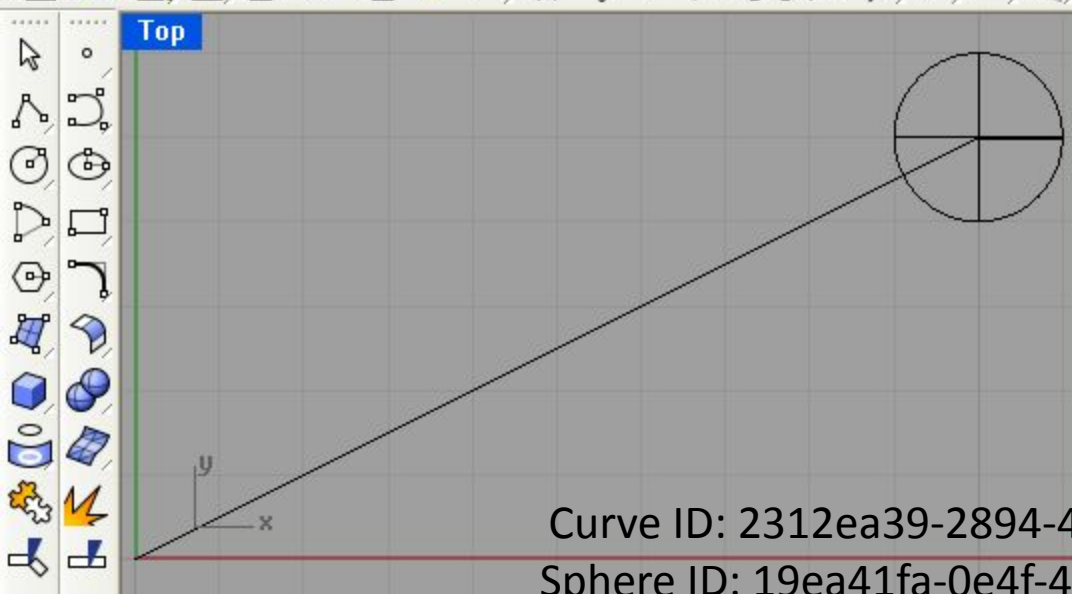
Call Rhino.enableRedraw(True) ' nevermind this, it refreshes the screen

Rhino.ZoomExtents ' and this adjusts the zoom level

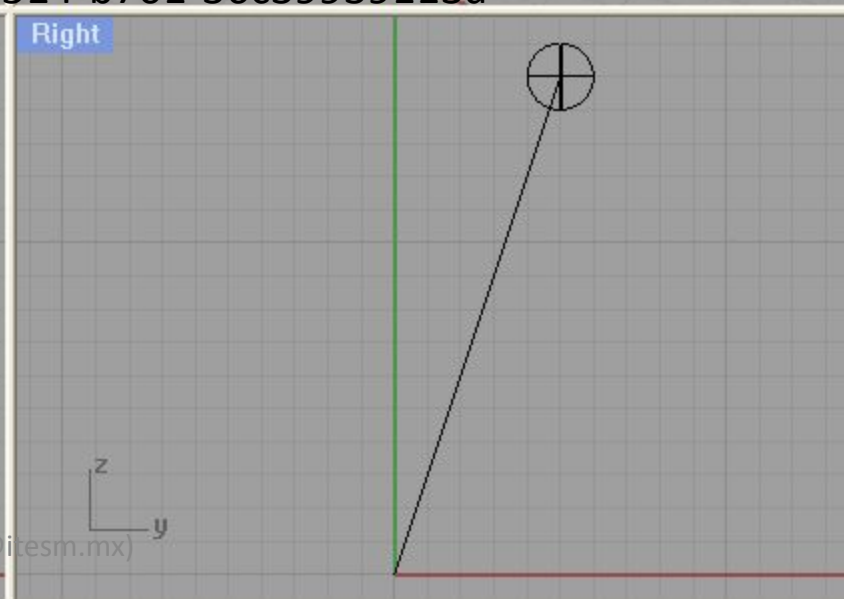
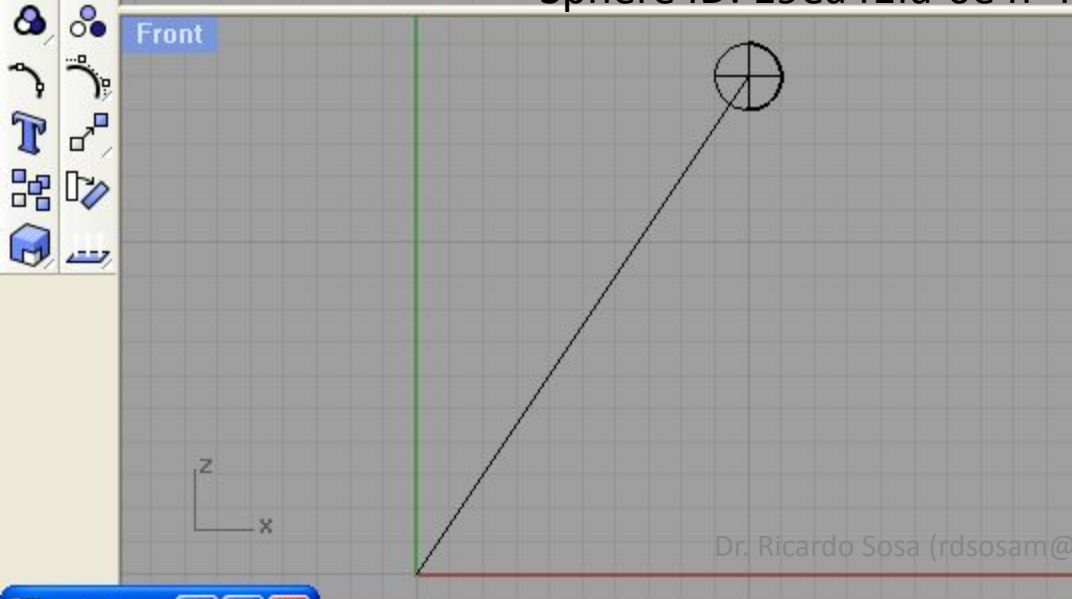
End Sub ' this is the end of the "DrawCurve" subroutine

Drag a window to zoom (All Dynamic Extents Factor In Out Selected Target 1To1): _All

Command:



Curve ID: 2312ea39-2894-4d1f-b31e-406fa88e5824
Sphere ID: 19ea41fa-0e4f-4514-b701-36c39939113a



Now some randomness...

' Copy and paste this code in your RhinoScript Editor (Tools ☐ RhinoScript ☐ Edit...)

' This is a basic script to draw a curve with fixed coordinates (Not very useful, but a good starting point)

Option Explicit ' nevermind this, just make sure that your scripts always start with it

DrawCurve ' this tells the program what subroutine to run

Sub DrawCurve ' this is the code to run when "DrawCurve" is called above

Call Rhino.enableRedraw(False) ' nevermind this, it speeds up the execution of the code

Dim controlpoints(1) ' controlpoints is an array of 3-D points (see next slide)

controlpoints(0) = Array(0,0,0) ' x = 0, y = 0, z = 0

controlpoints(1) = Array(randomBetween(-10,10),randomBetween(-10,10),15) ' x = random, y = random, z = 15

Rhino.Print ("Curve ID: " + Rhino.AddCurve(controlpoints)) ' this draws the curve and prints its I.D.

Rhino.Print ("Sphere ID: " + Rhino.AddSphere (controlpoints(1), 1)) ' this draws a sphere and its I.D.

Call Rhino.enableRedraw(True) ' nevermind this, it refreshes the screen

Rhino.ZoomExtents ' and this adjusts the zoom level

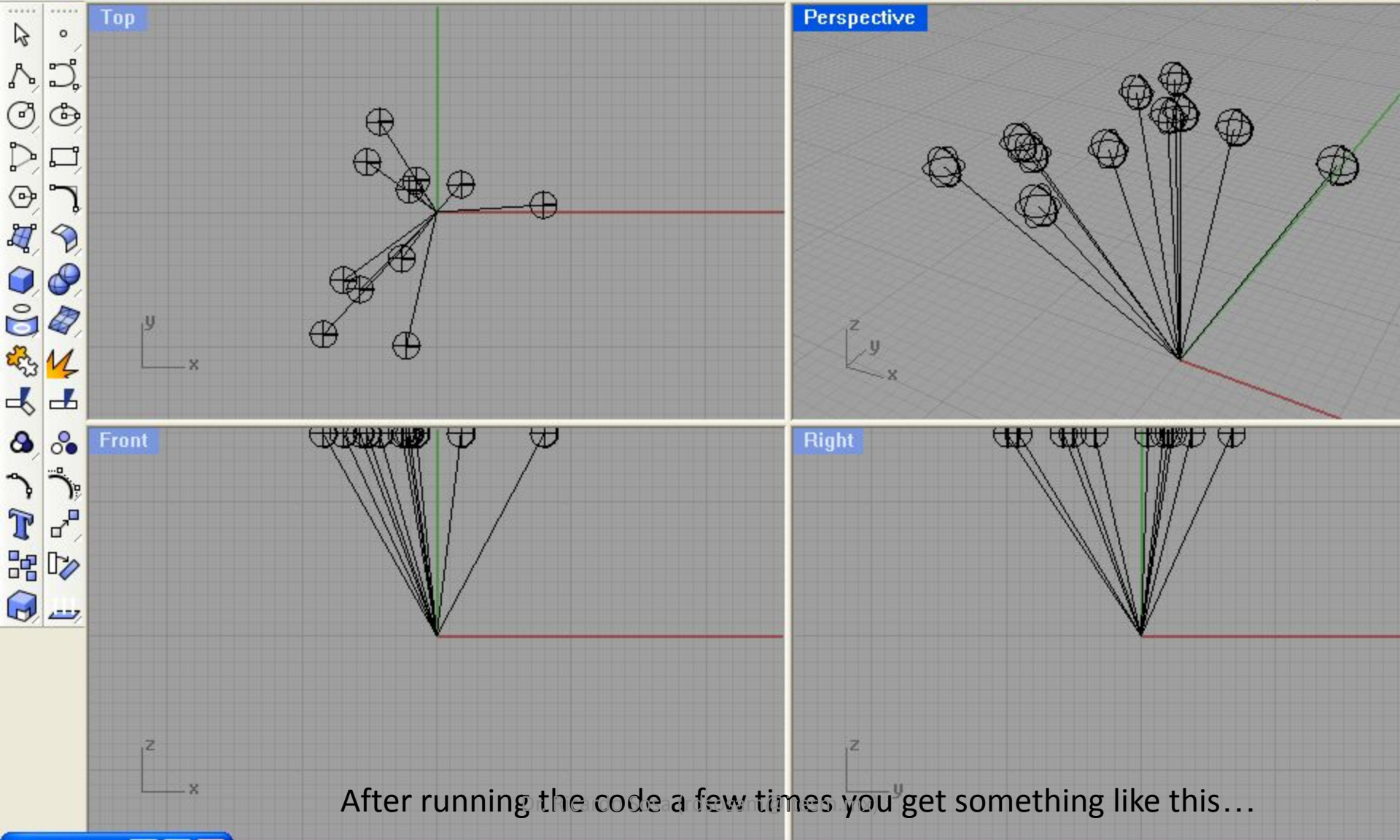
End Sub ' this is the end of the "DrawCurve" subroutine

Function randomBetween(min,max) ' this is the code to generate random numbers between limits

randomBetween = Rnd*(max-min)+min ' returns a random number between the limits specified

End Function ' end of the randomness function

Curve ID: cc912f73-45ce-4653-8bbd-2a27da96b3dc
Sphere ID: 1b2662d5-589c-4b12-9368-63372bf78c97
Command:



After running the code a few times you get something like this...

Now some recursion...

' Copy and paste this code in your RhinoScript Editor (Tools □ RhinoScript □ Edit...)

' This is a basic script to draw a curve with fixed coordinates (Not very useful, but a good starting point)

Option Explicit ' nevermind this, just make sure that your scripts always start with it

DrawCurve ' this tells the program what subroutine to run

Sub DrawCurve ' this is the code to run when "DrawCurve" is called above

Call Rhino.enableRedraw(False) ' nevermind this, it speeds up the execution of the code

Dim controlpoints(1) ' controlpoints is an array of 3-D points (see next slide)

Dim i

For i=0 To 100

controlpoints(0) = Array(0,0,0) ' x = 0, y = 0, z = 0

controlpoints(1) = Array(randomBetween(-10,10),randomBetween(-10,10),15) ' x = random, y = random, z = 15

Rhino.AddCurve controlpoints ' this draws the curve

Rhino.AddSphere controlpoints(1), 1 ' this draws a sphere

Next

Call Rhino.EnableRedraw(True) ' nevermind this, it refreshes the screen

Rhino.ZoomExtents ' and this adjusts the zoom level

End Sub ' this is the end of the "DrawCurve" subroutine

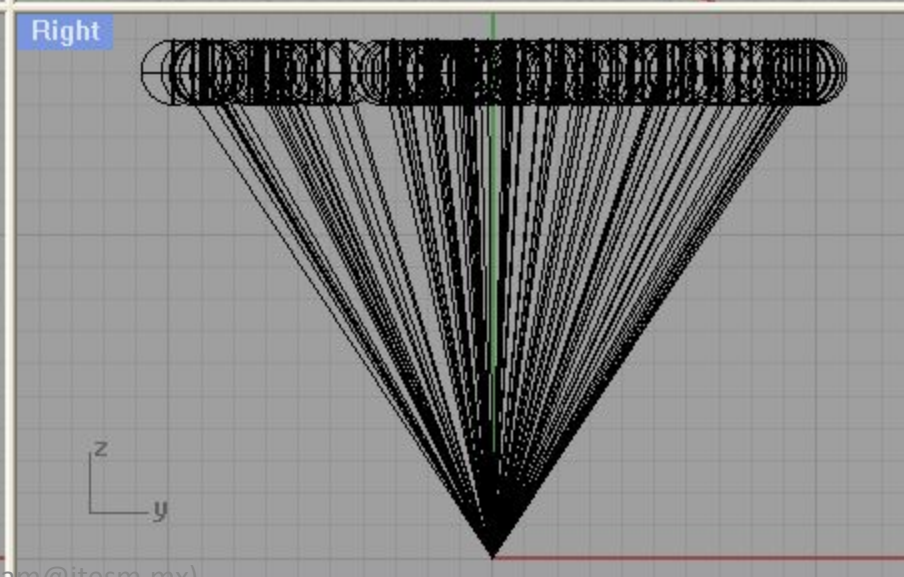
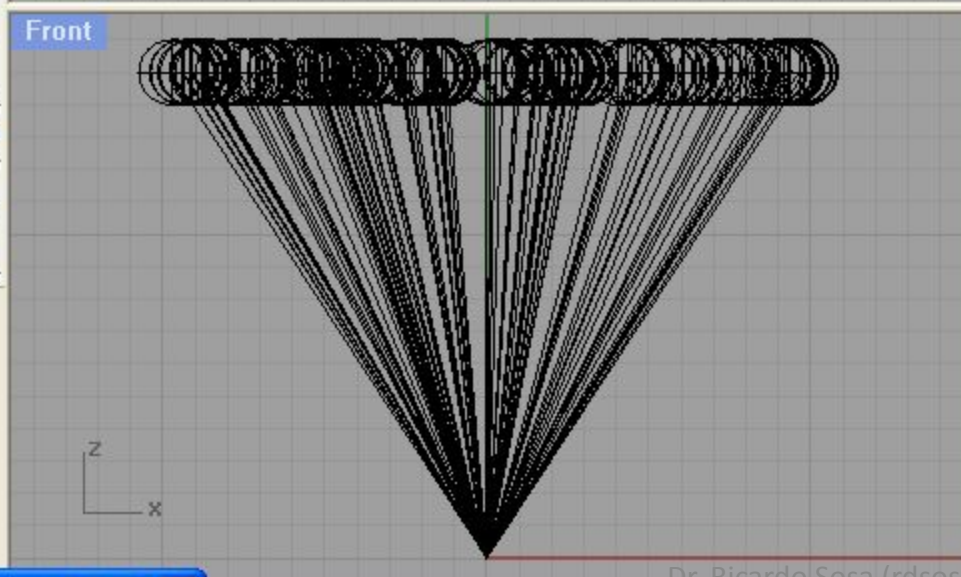
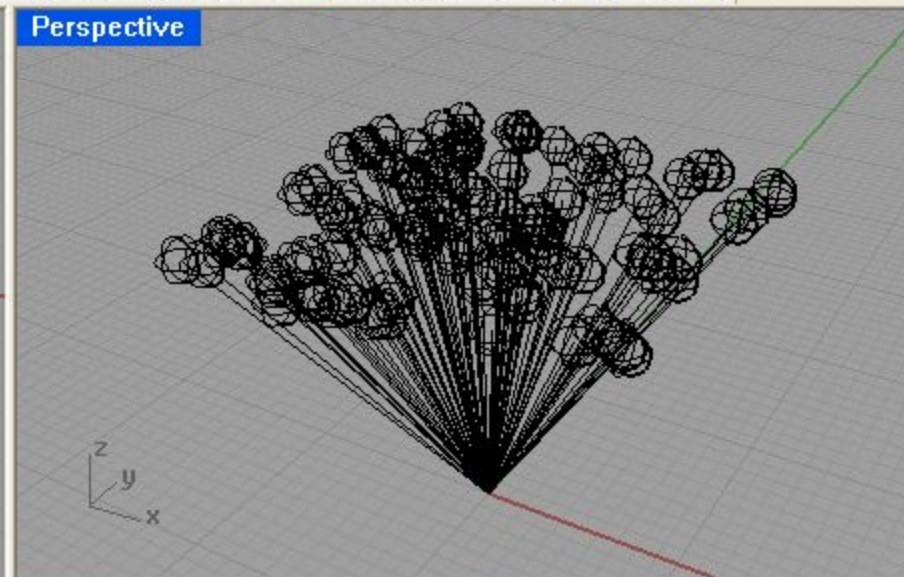
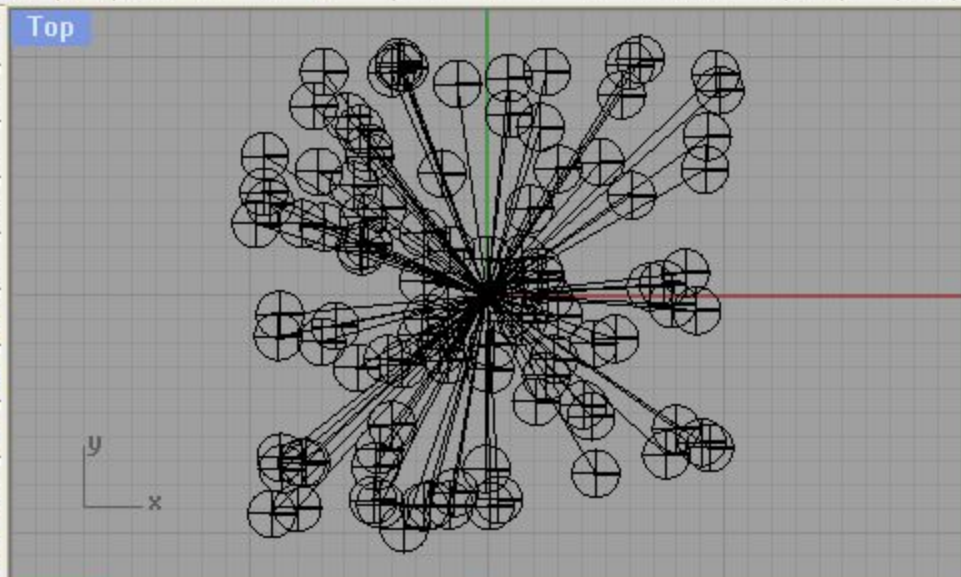
Function randomBetween(min,max) ' this is the code to generate random numbers between limits

randomBetween = Rnd*(max-min)+min ' returns a random number between the limits specified

End Function ' end of the randomness function

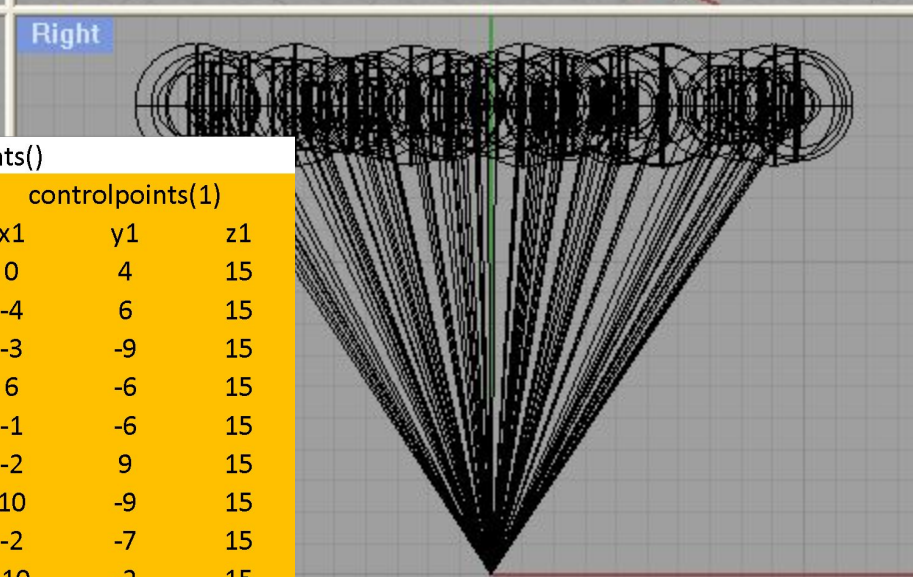
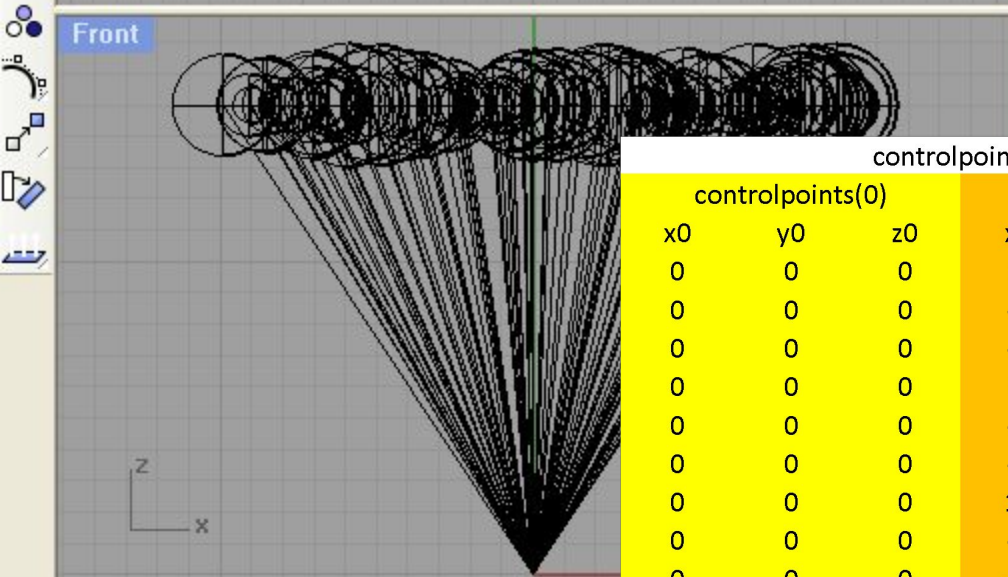
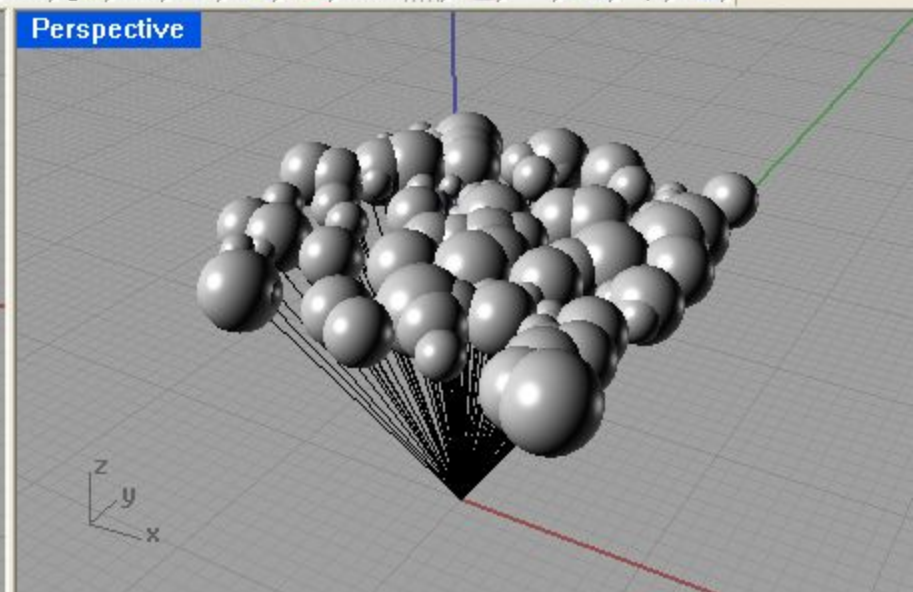
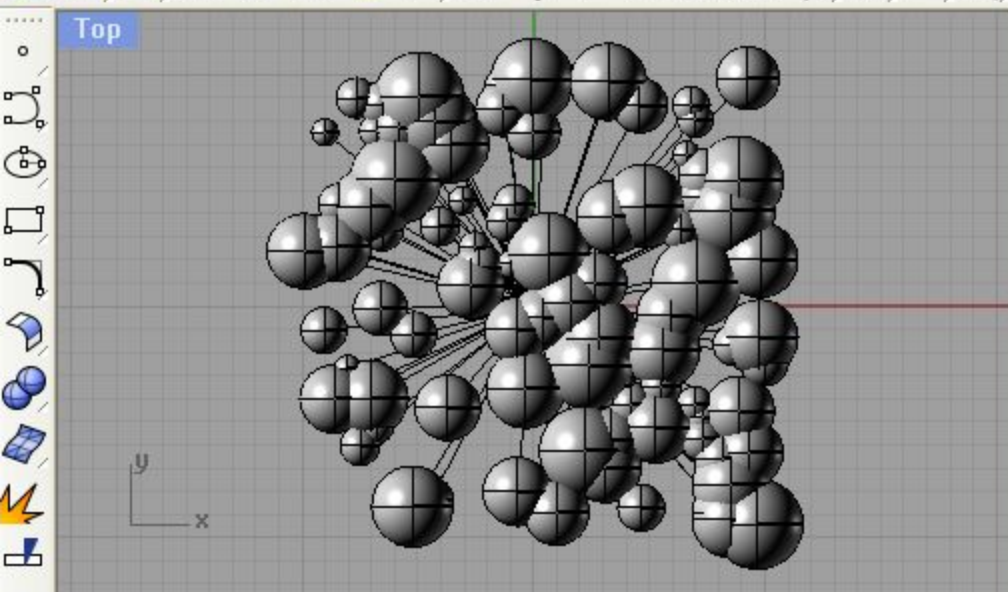
g a window to zoom (All Dynamic Extents Factor In Out Selected Target 1To1): _All
ose option (Extents Selected 1To1): _Extents

Command:



Command: _Shade

Shade settings (DisplayMode=Shaded DrawCurves=Yes DrawWires=No DrawGrid=Yes DrawAxes=Yes):

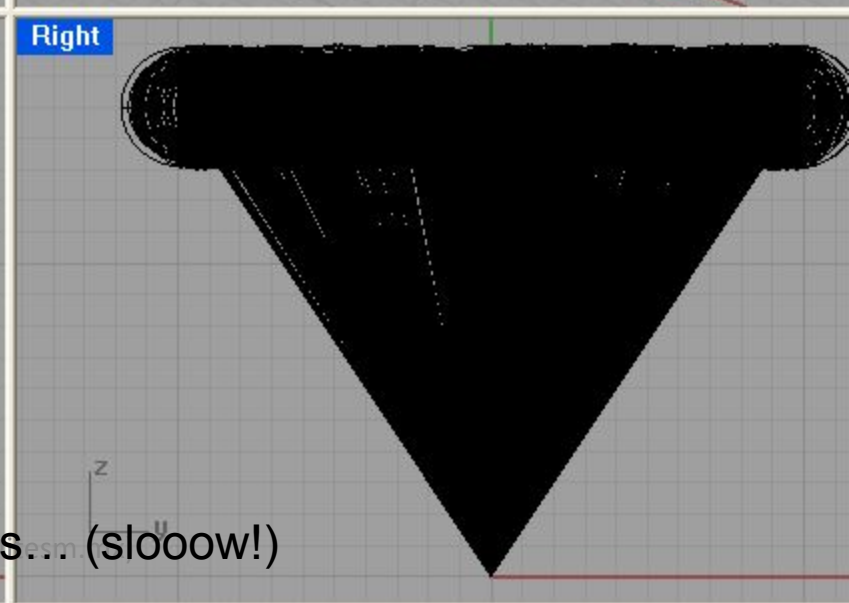
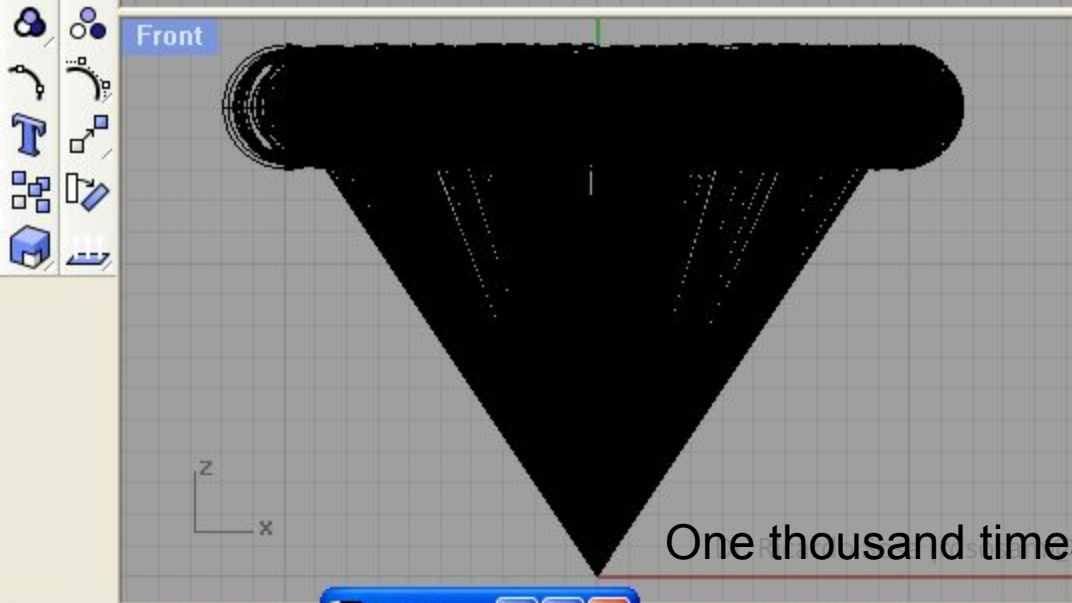
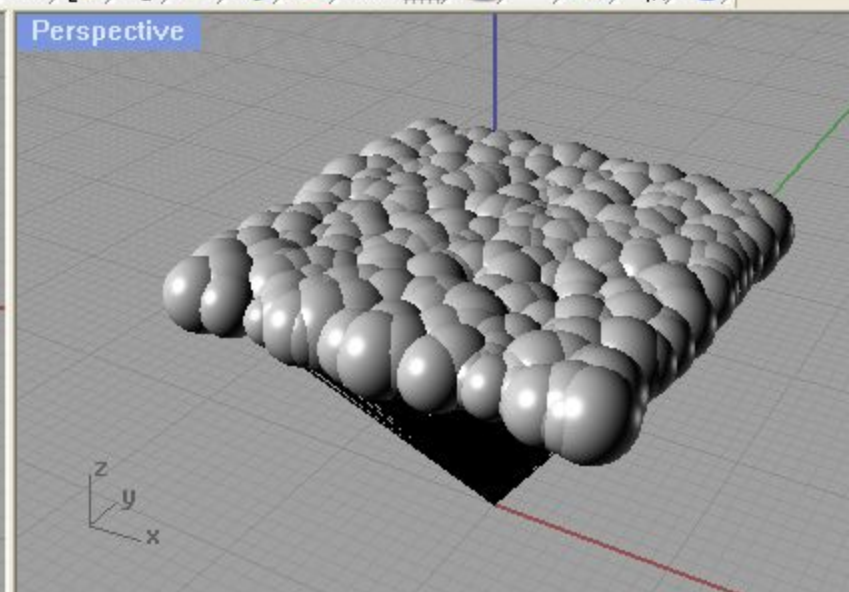
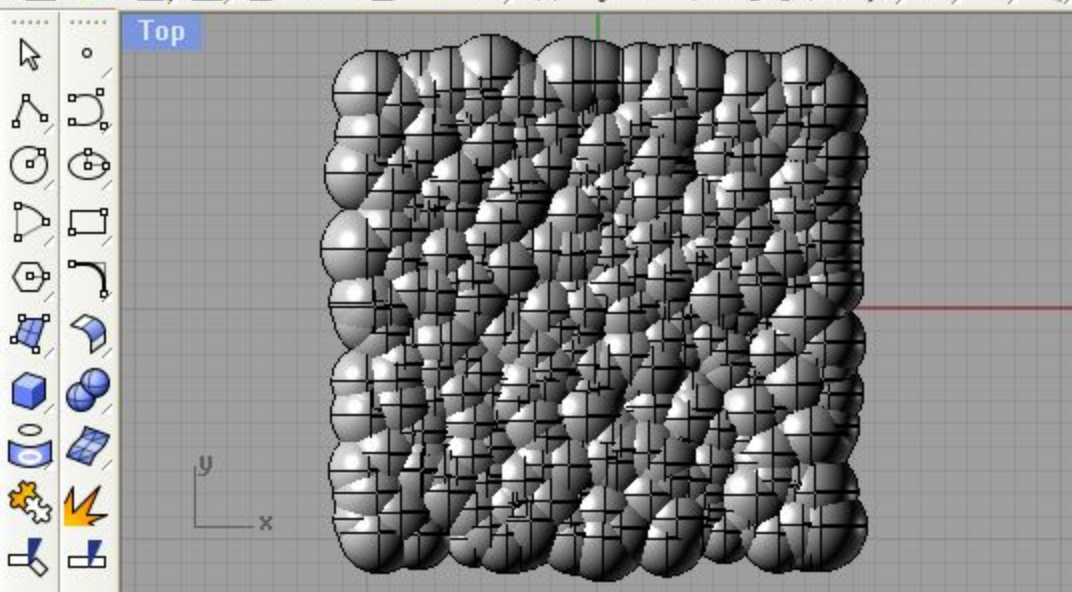


controlpoints()					
controlpoints(0)			controlpoints(1)		
x0	y0	z0	x1	y1	z1
0	0	0	0	4	15
0	0	0	-4	6	15
0	0	0	-3	-9	15
0	0	0	6	-6	15
0	0	0	-1	-6	15
0	0	0	-2	9	15
0	0	0	10	-9	15
0	0	0	-2	-7	15
0	0	0	-10	-2	15
0	0	0	-4	9	15
0	0	0	-5	-3	15

Dr. Ricardo Sosa (rdsosam@itesm.mx)

Creating meshes... Press Esc to cancel

Choose Shade settings (DisplayMode=Shaded DrawCurves=Yes DrawWires=No DrawGrid=Yes DrawAxes=Yes):



One thousand times... (sloooow!)

More interesting curves...

' Copy and paste this code in your RhinoScript Editor (Tools □ RhinoScript □ Edit...)

' This is a basic script to draw a curve with fixed coordinates (Not very useful, but a good starting point)

Option Explicit ' nevermind this, just make sure that your scripts always start with it

DrawCurve ' this tells the program what subroutine to run

Sub DrawCurve ' this is the code to run when "DrawCurve" is called above

Call Rhino.enableRedraw(False) ' nevermind this, it speeds up the execution of the code

Dim **controlpoints(2)**, i ' controlpoints is an array of 3-D points (see next slide)

For i=0 To 50

controlpoints(0) = Array(0,0,0) ' x = 0, y = 0, z = 0

controlpoints(1) = Array(randomBetween(-5,5),randomBetween(-5,5),0)

controlpoints(2) = Array(randomBetween(-10,10),randomBetween(-10,10),15)

Rhino.AddCurve controlpoints, **2** ' this draws the curve **of two degrees now**

Rhino.AddSphere controlpoints(1), **0.25** ' this draws a **small** sphere at second point

Next

Call Rhino.enableRedraw(True) ' nevermind this, it refreshes the screen

Rhino.ZoomExtents ' and this adjusts the zoom level

End Sub ' this is the end of the "DrawCurve" subroutine

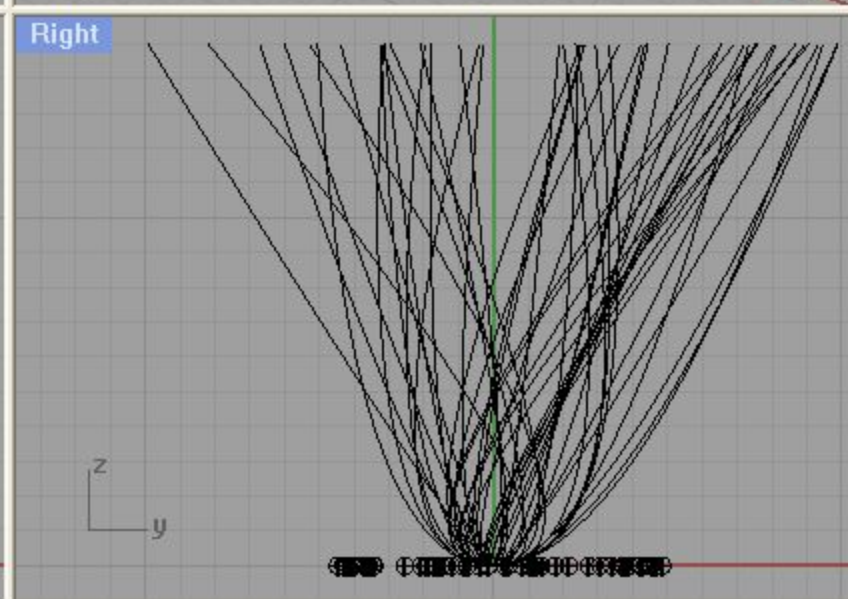
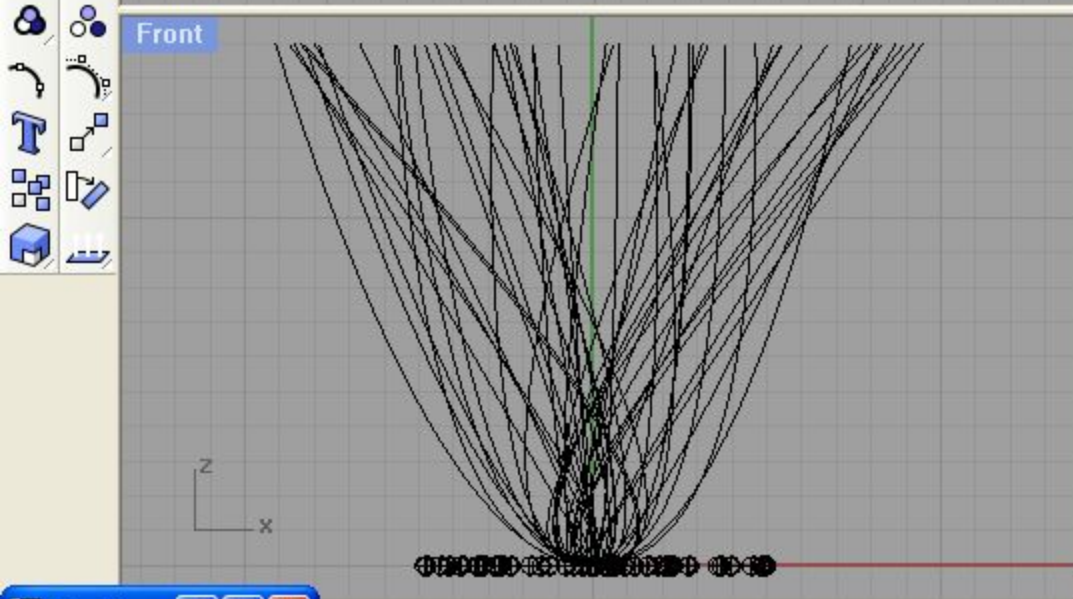
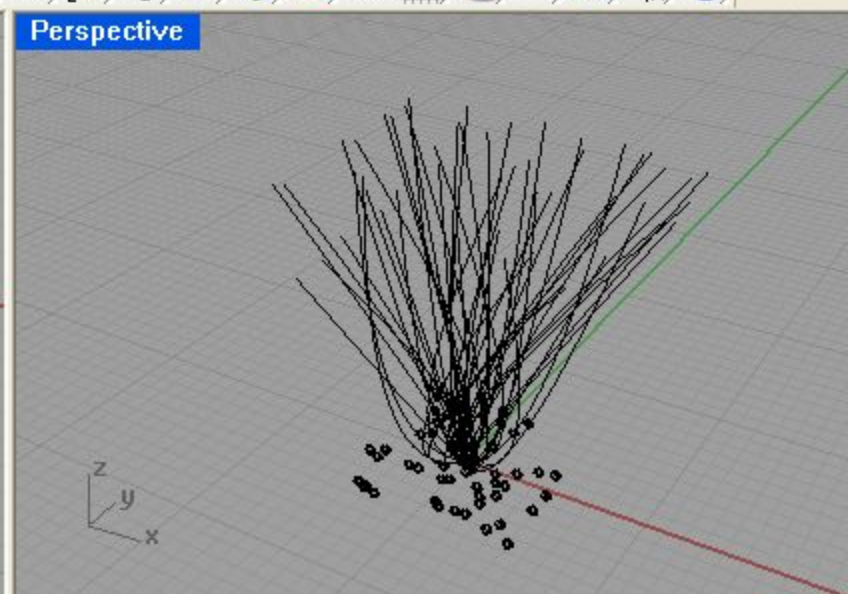
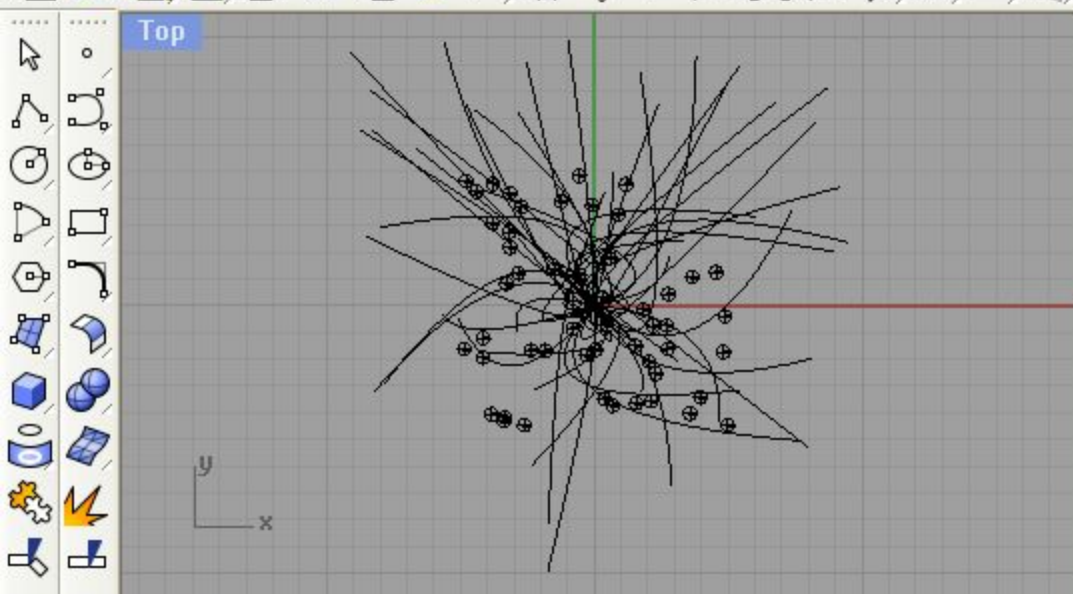
Function randomBetween(min,max) ' this is the code to generate random numbers between limits

randomBetween = Rnd*(max-min)+min ' returns a random number between the limits specified

End Function ' end of the randomness function

Choose option (Extents Selected 1To1): _Extents

Command:

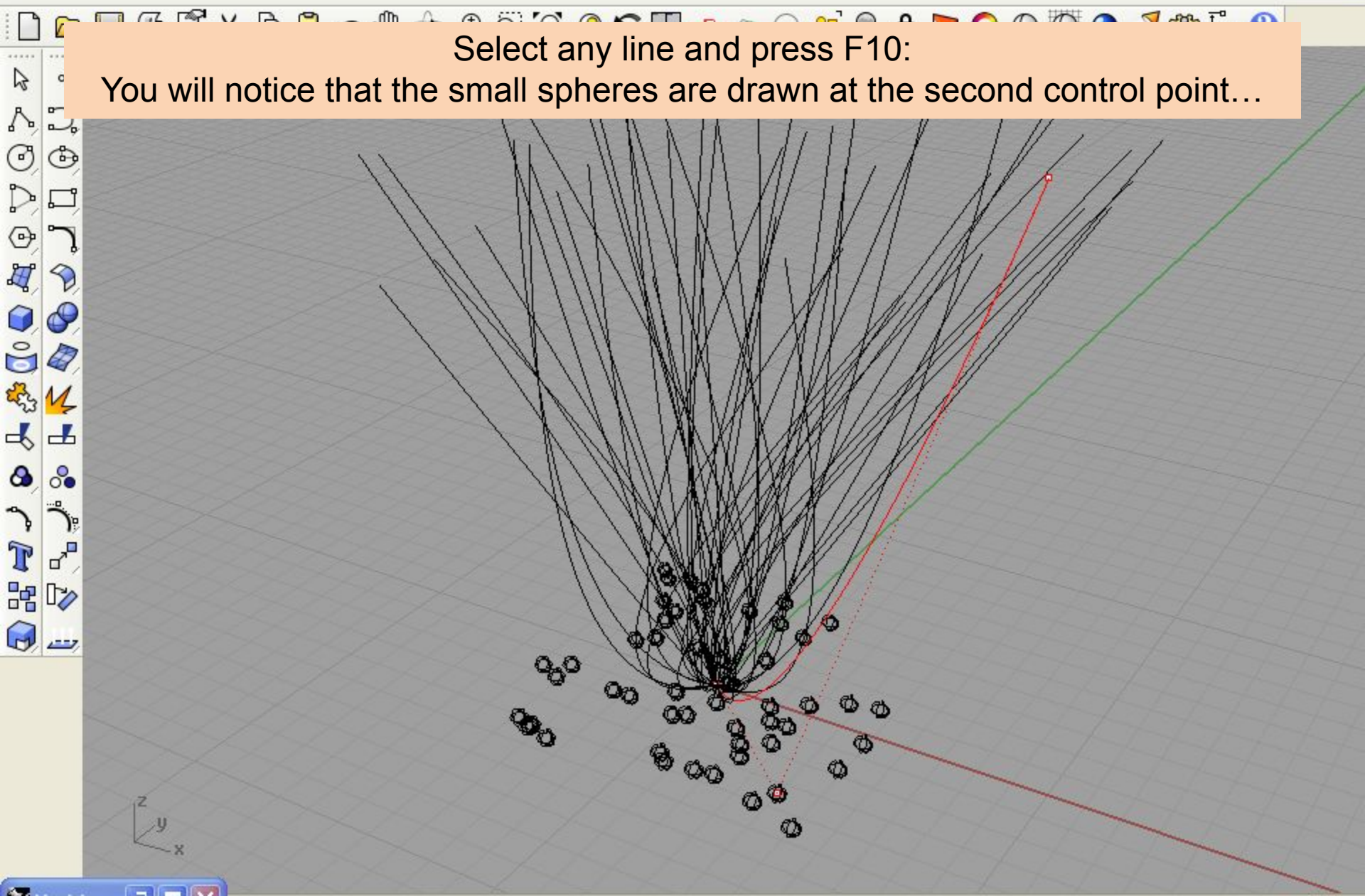


Command: _PointsOn

Command:

Select any line and press F10:

You will notice that the small spheres are drawn at the second control point...



- ' Copy and paste this code in your RhinoScript Editor (Tools □ RhinoScript □ Edit...)
- ' This is a basic script to draw a curve with fixed coordinates (Not very useful, but a good starting point)

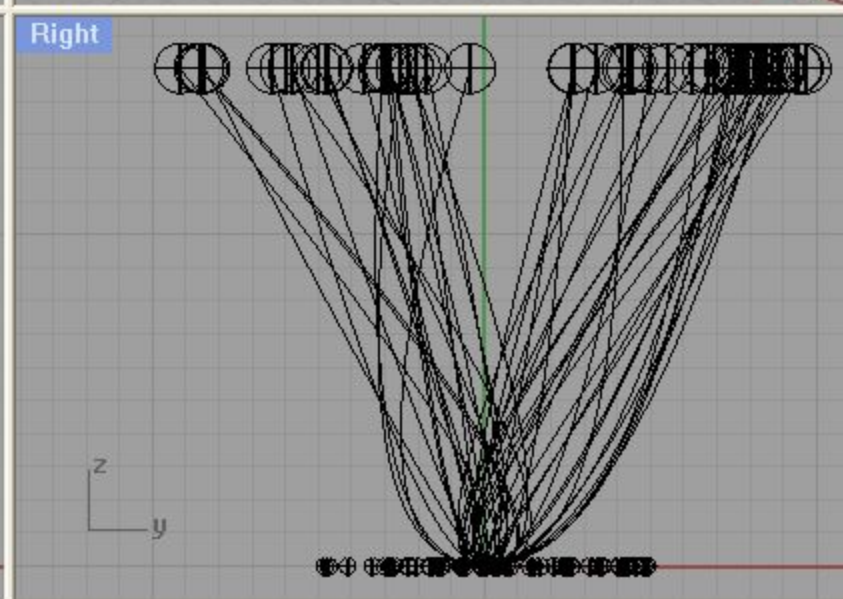
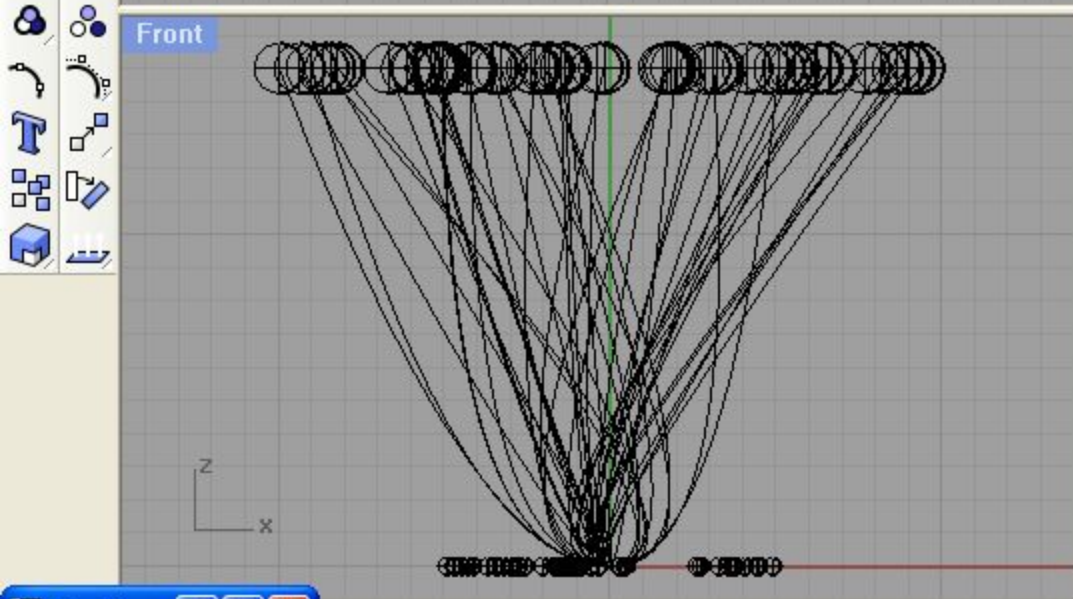
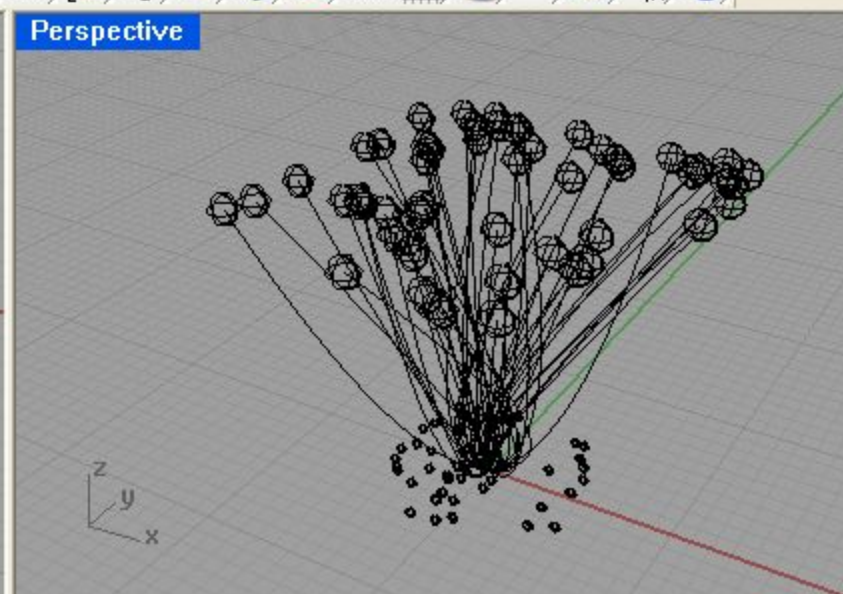
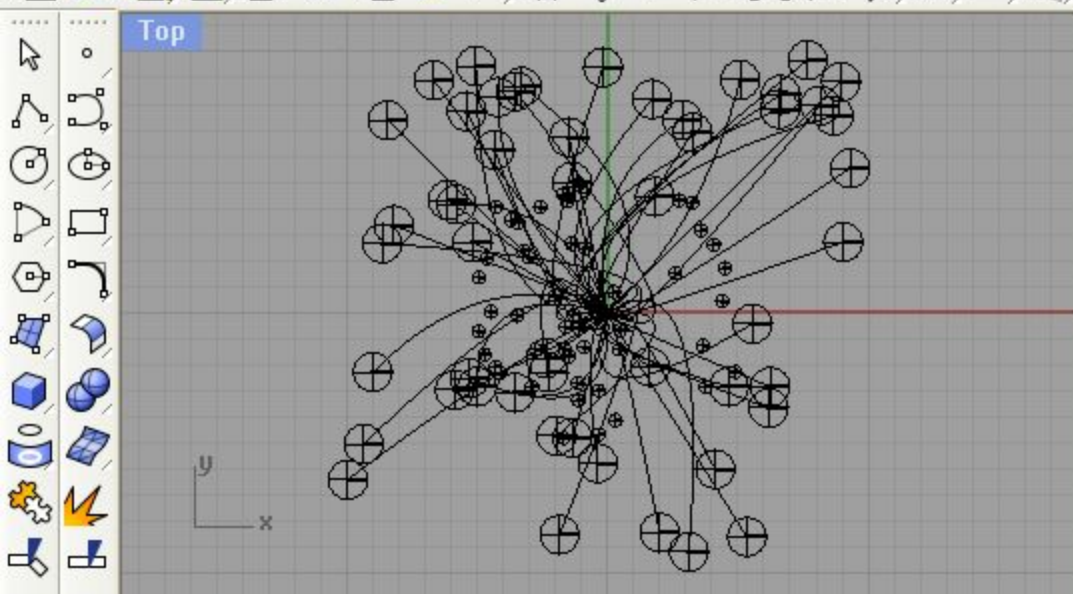
Option Explicit ' nevermind this, just make sure that your scripts always start with it
DrawCurve ' this tells the program what subroutine to run

```
Sub DrawCurve ' this is the code to run when "DrawCurve" is called above
    Call Rhino.enableRedraw(False) ' nevermind this, it speeds up the execution of the code
    Dim controlpoints(2), i ' controlpoints is an array of 3-D points (see next slide)
    For i=0 To 50
        controlpoints(0) = Array(0,0,0) ' x = 0, y = 0, z = 0
        controlpoints(1) = Array(randomBetween(-5,5),randomBetween(-5,5),0)
        controlpoints(2) = Array(randomBetween(-10,10),randomBetween(-10,10),15)
        Rhino.AddCurve controlpoints, 2 ' this draws the curve
        Rhino.AddSphere controlpoints(1), 0.25 ' this draws a small sphere at second point
        Rhino.AddSphere controlpoints(2), 0.75 ' this draws a big sphere at third point
    Next
    Call Rhino.enableRedraw(True) ' nevermind this, it refreshes the screen
    Rhino.ZoomExtents ' and this adjusts the zoom level
End Sub ' this is the end of the "DrawCurve" subroutine
```

```
Function randomBetween(min,max) ' this is the code to generate random numbers between limits
    randomBetween = Rnd*(max-min)+min ' returns a random number between the limits specified
End Function ' end of the randomness function
```


Choose option (Extents Selected 1To1): _Extents

Command:

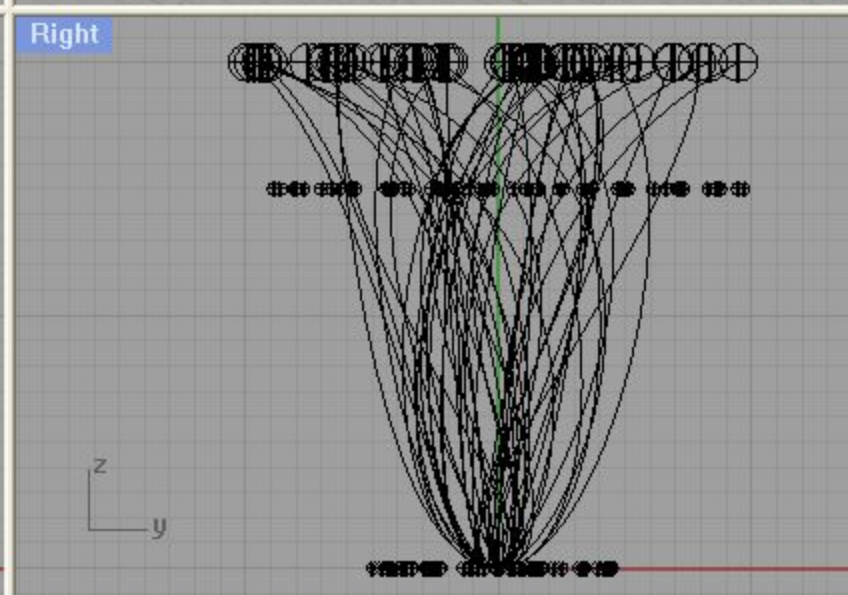
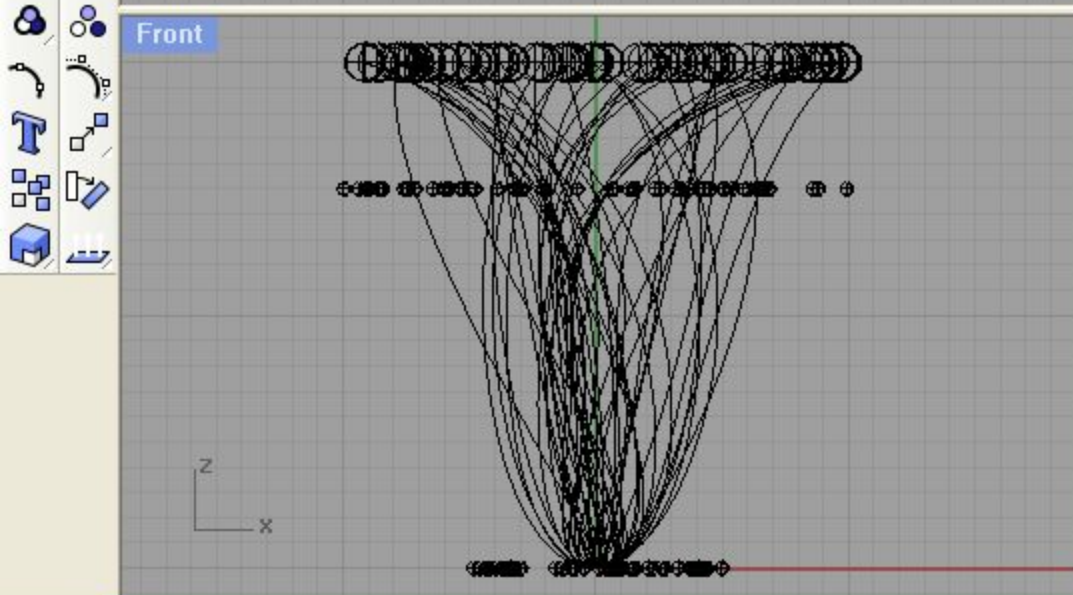
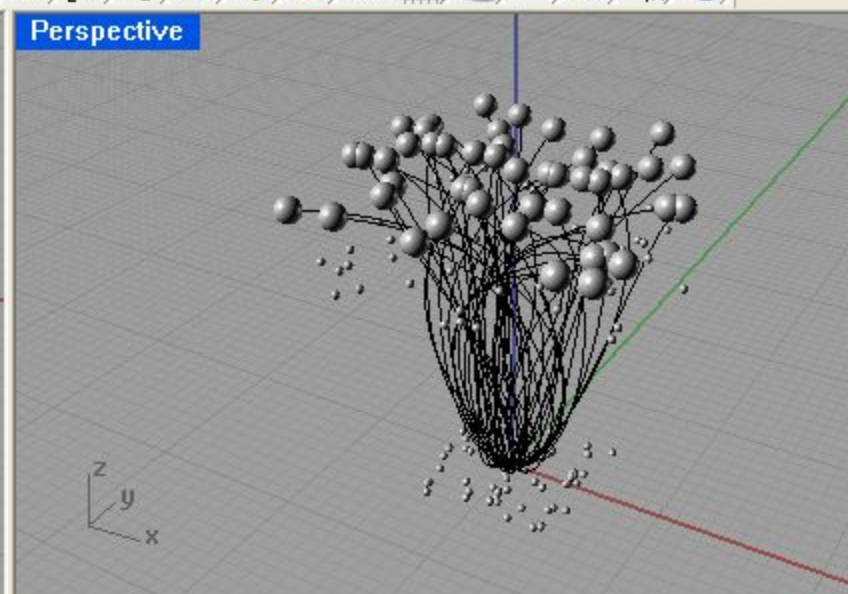
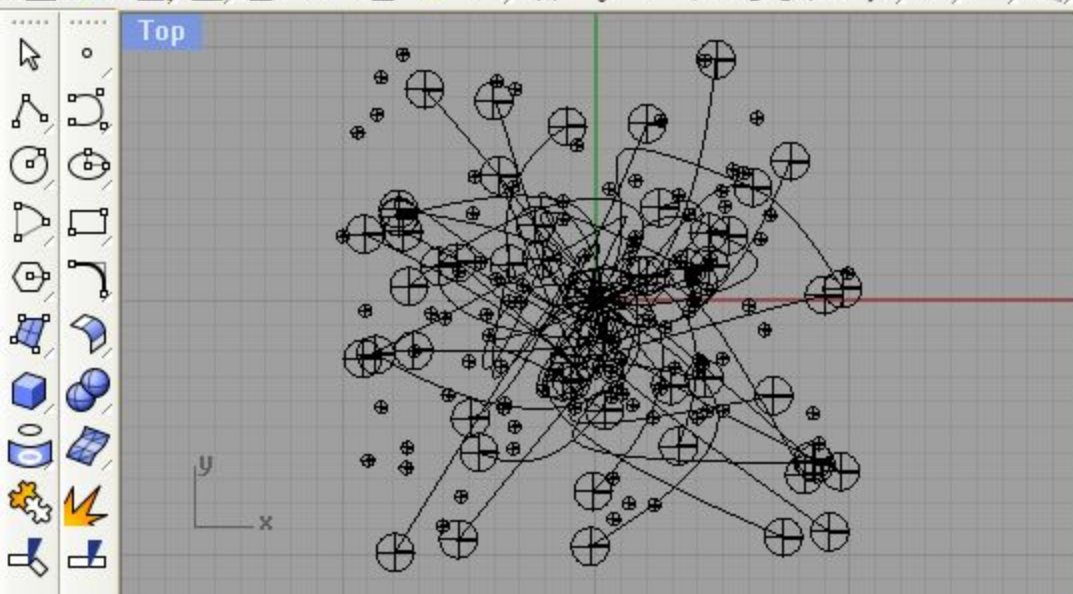


Time for a challenge...

How do you achieve the following?

Creating meshes... Press Esc to cancel

Choose Shade settings (DisplayMode=Shaded DrawCurves=Yes DrawWires=No DrawGrid=Yes DrawAxes=Yes):



- ' Copy and paste this code in your RhinoScript Editor (Tools □ RhinoScript □ Edit...)
- ' This is a basic script to draw a curve with fixed coordinates (Not very useful, but a good starting point)

Option Explicit ' nevermind this, just make sure that your scripts always start with it
DrawCurve ' this tells the program what subroutine to run

```
Sub DrawCurve ' this is the code to run when "DrawCurve" is called above
  Call Rhino.enableRedraw(False) ' nevermind this, it speeds up the execution of the code
  Dim controlpoints(3), i ' controlpoints is an array of 3-D points (see next slide)
  For i=0 To 50
    controlpoints(0) = Array(0,0,0) ' x = 0, y = 0, z = 0
    controlpoints(1) = Array(randomBetween(-5,5),randomBetween(-5,5),0)
    controlpoints(2) = Array(randomBetween(-10,10),randomBetween(-10,10),15)
    controlpoints(3) = Array(randomBetween(-10,10),randomBetween(-10,10),20)
    Rhino.AddCurve controlpoints, 3 ' this draws the curve
    Rhino.AddSphere controlpoints(1), 0.25 ' this draws a small sphere at second point
    Rhino.AddSphere controlpoints(2), 0.25 ' this draws a big sphere at third point
    Rhino.AddSphere controlpoints(3), 0.75 ' this draws a big sphere at third point
  Next
  Call Rhino.enableRedraw(True) ' nevermind this, it refreshes the screen
  Rhino.ZoomExtents ' and this adjusts the zoom level
End Sub ' this is the end of the "DrawCurve" subroutine
```

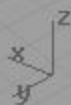
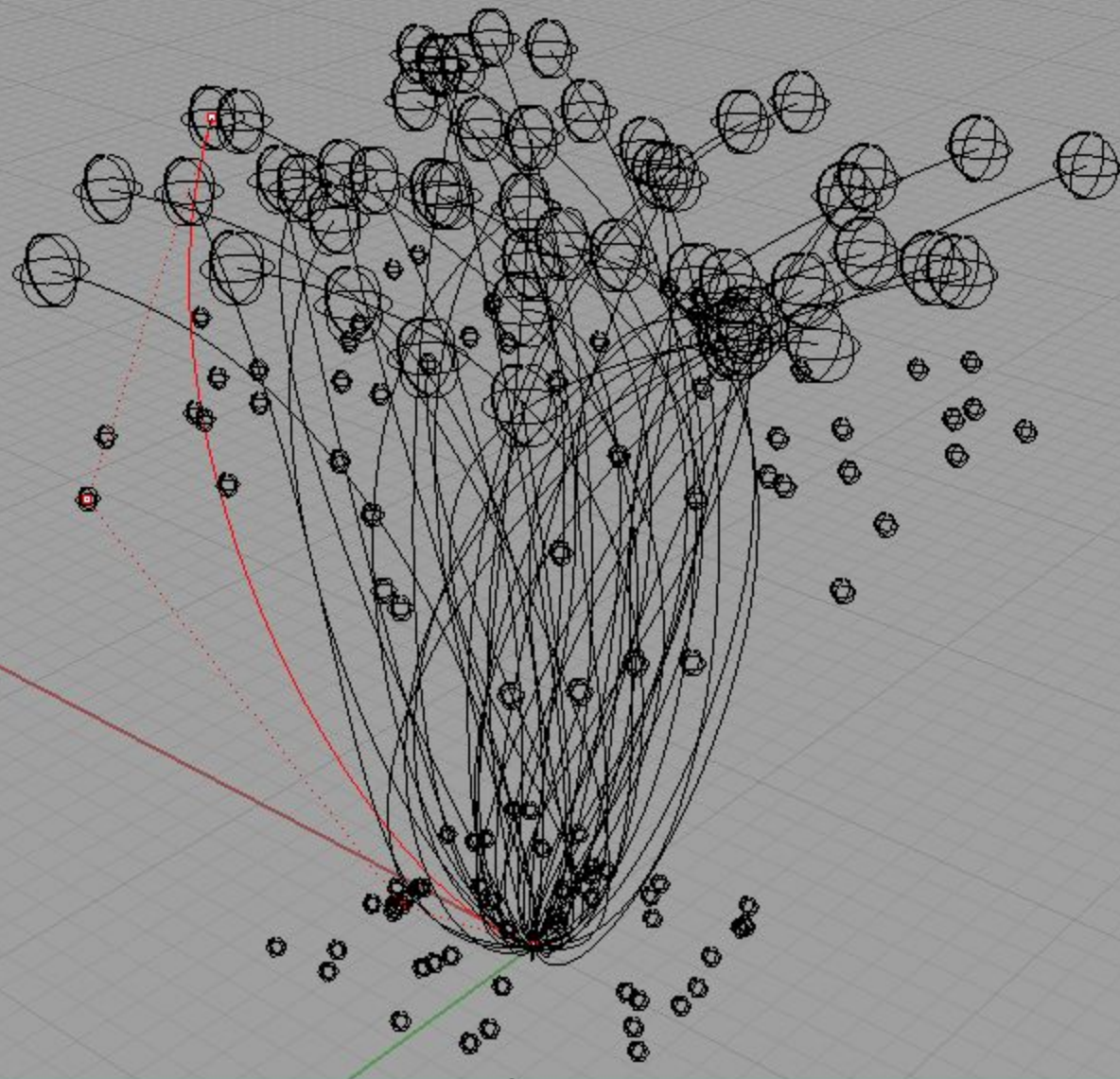
```
Function randomBetween(min,max) ' this is the code to generate random numbers between limits
  randomBetween = Rnd*(max-min)+min ' returns a random number between the limits specified
End Function ' end of the randomness function
```


Command: _PointsOn

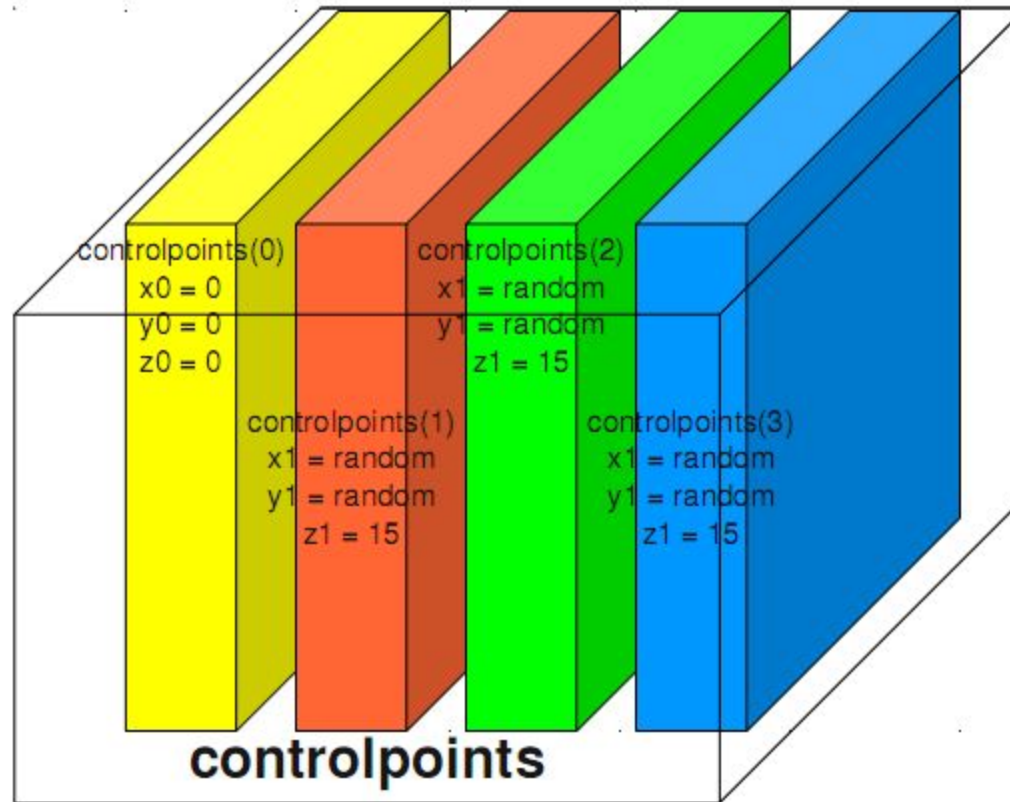
Command:



Perspective



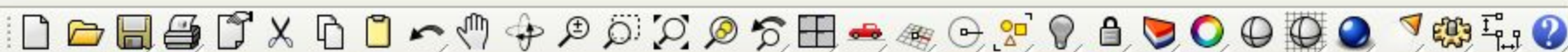
Add another set of coordinates...



Rhino.Command "anycommand"

Shift + Rightclick any tool icon to see
its _Command

Command:



Top



Front

Perspective

Linked toolbar

Name:

☐ Float to top

Tooltips

Left

Right

Button text

☒ Show bitmap only
☐ Show text only
☐ Show bitmap and text

Left mouse button command

Right mouse button command

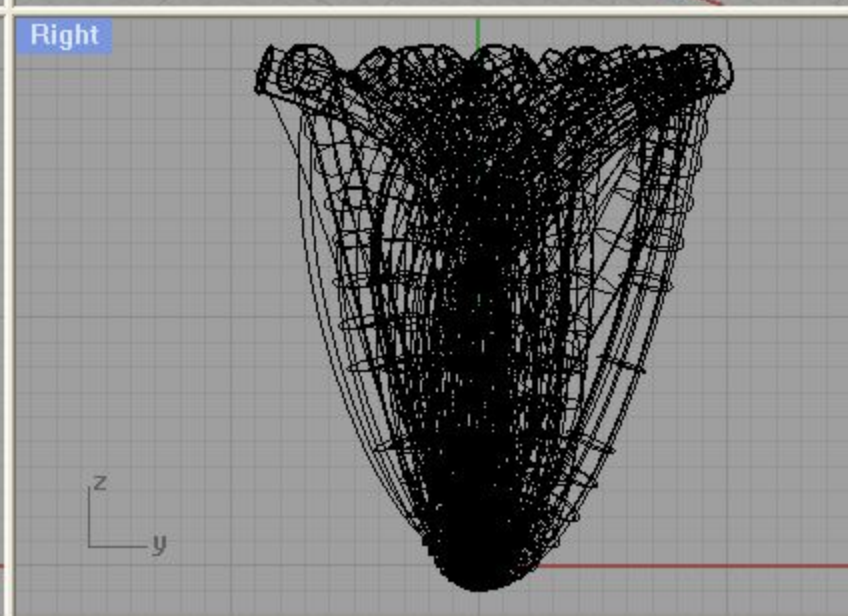
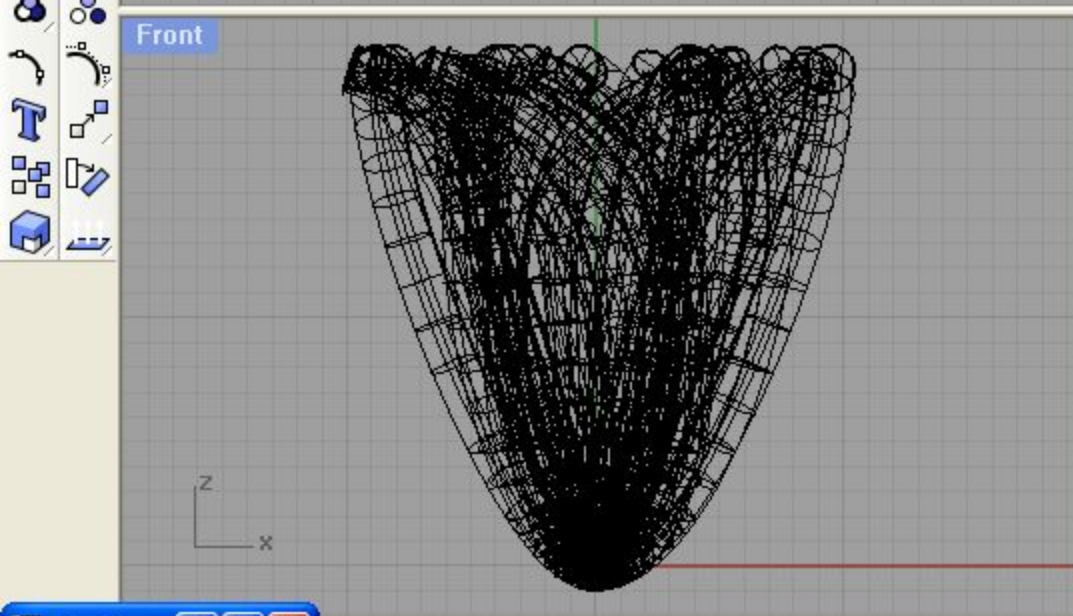
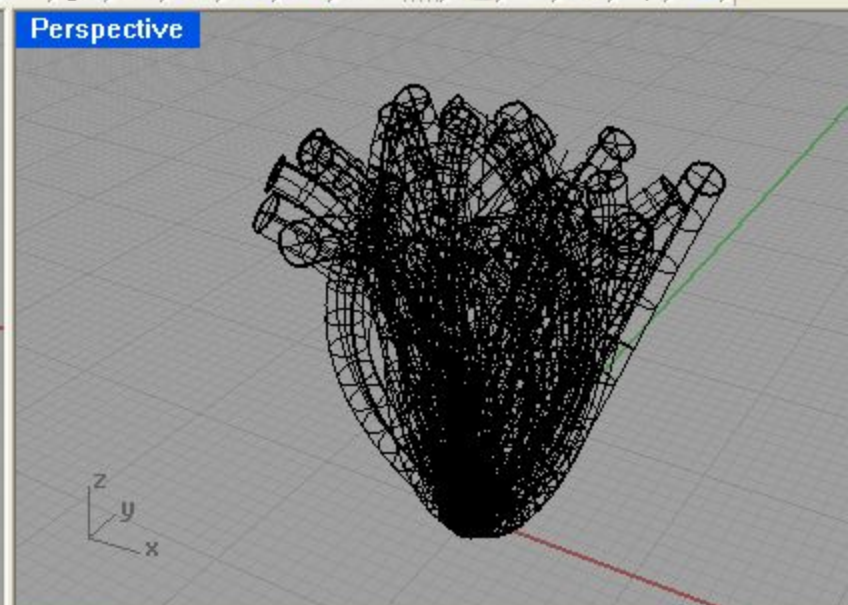
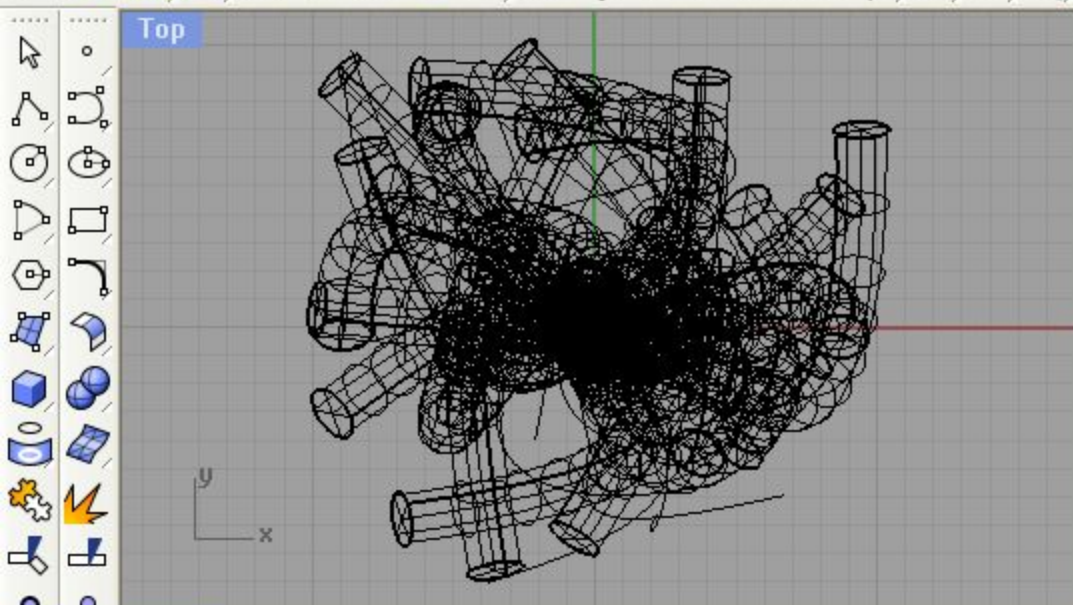
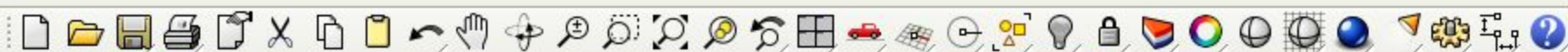
- ' Copy and paste this code in your RhinoScript Editor (Tools □ RhinoScript □ Edit...)
- ' This is a basic script to draw a curve with fixed coordinates (Not very useful, but a good starting point)

Option Explicit ' nevermind this, just make sure that your scripts always start with it
DrawCurve ' this tells the program what subroutine to run

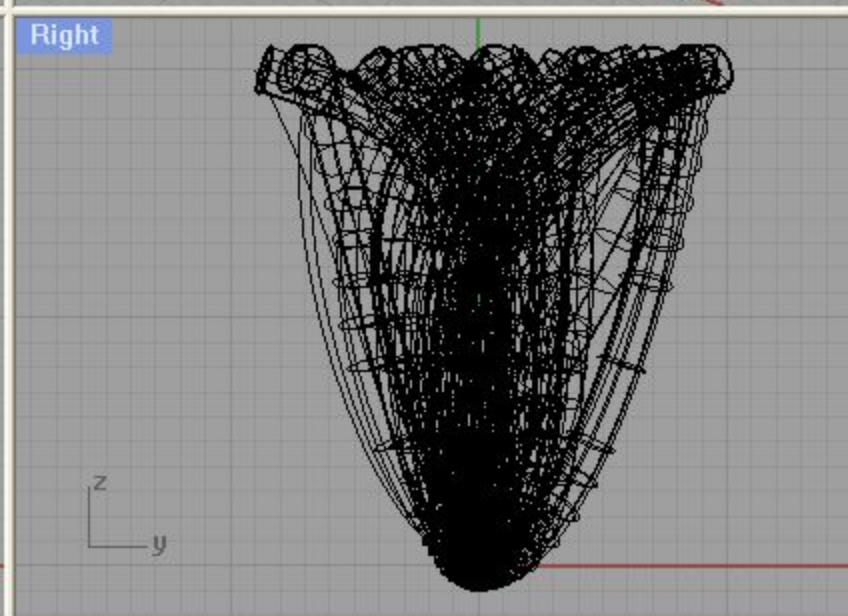
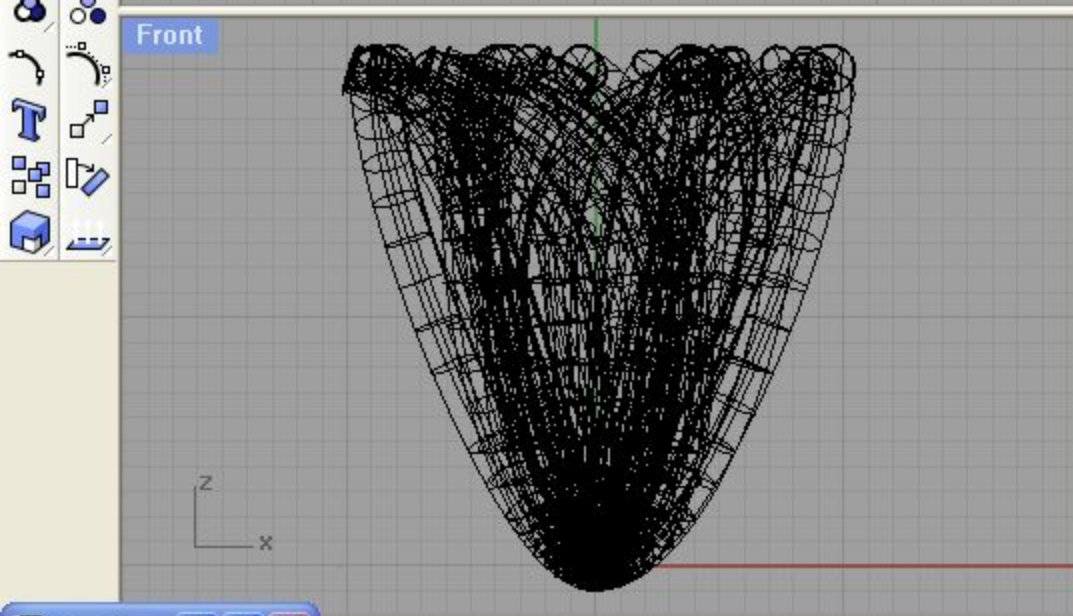
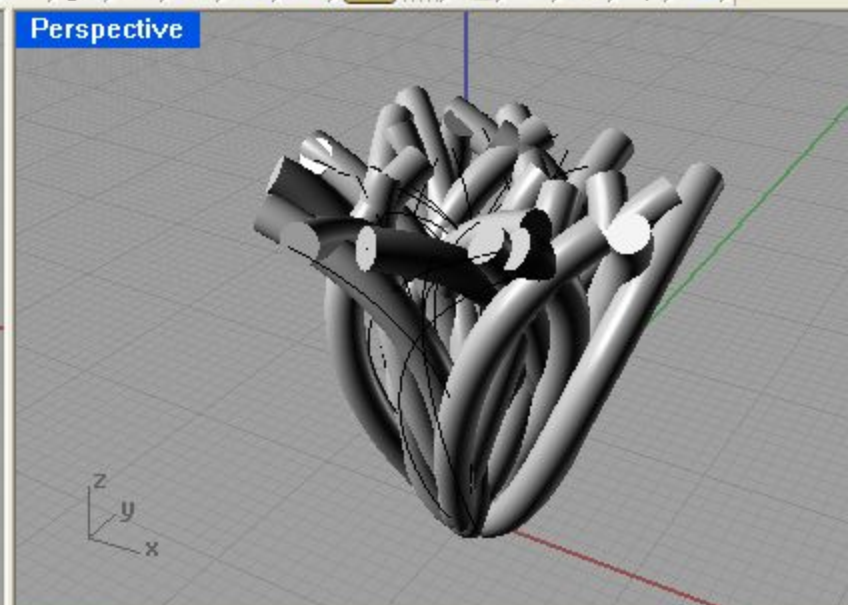
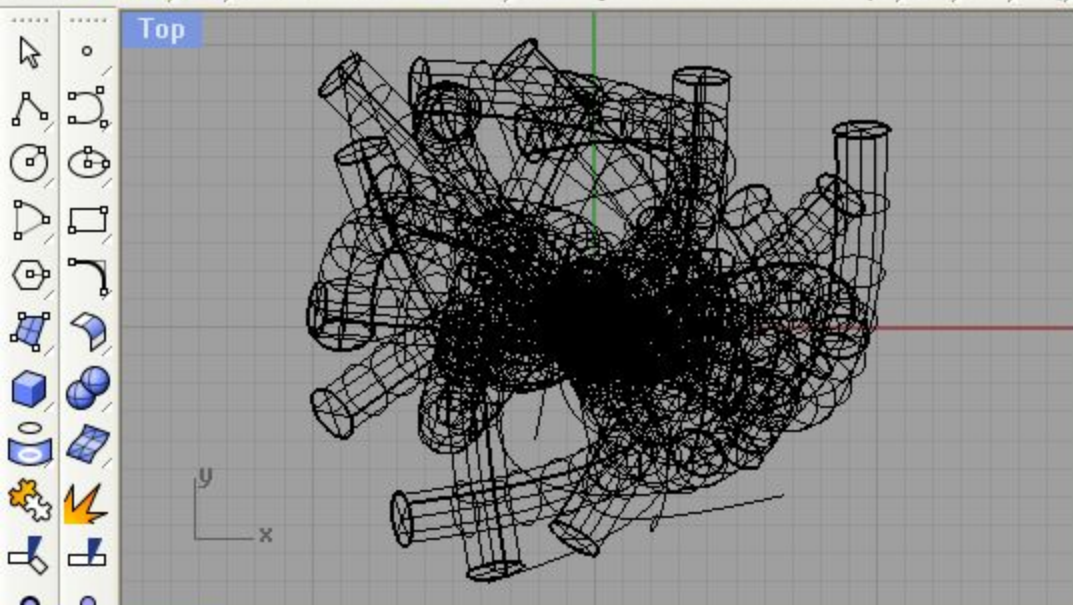
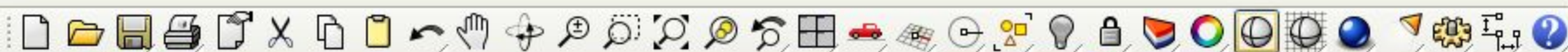
```
Sub DrawCurve ' this is the code to run when "DrawCurve" is called above
    Call Rhino.enableRedraw(False) ' nevermind this, it speeds up the execution of the code
    Dim controlpoints(3), i ' controlpoints is an array of 3-D points (see next slide)
    Dim strCmd, curveID
    For i=0 To 50
        controlpoints(0) = Array(0,0,0) ' x = 0, y = 0, z = 0
        controlpoints(1) = Array(randomBetween(-5,5),randomBetween(-5,5),0)
        controlpoints(2) = Array(randomBetween(-10,10),randomBetween(-10,10),15)
        controlpoints(3) = Array(randomBetween(-10,10),randomBetween(-10,10),20)
        curveID = Rhino.AddCurve(controlpoints, 3) ' this draws the curve
        Rhino.SelectObject(curveID)
        Rhino.Command "_Pipe " & 1.0 & " Enter " & 1.0 & " Enter"
    Next
    Call Rhino.enableRedraw(True) ' nevermind this, it refreshes the screen
    Rhino.ZoomExtents ' and this adjusts the zoom level
End Sub ' this is the end of the "DrawCurve" subroutine
```

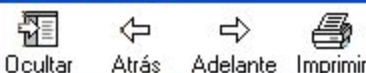
```
Function randomBetween(min,max) ' this is the code to generate random numbers between limits
    randomBetween = Rnd*(max-min)+min ' returns a random number between the limits specified
End Function ' end of the randomness function
```

Command:



Choose Shade settings (DisplayMode=Shaded DrawCurves=Yes DrawWires=No DrawGrid=Yes DrawAxes=Yes):





Contenido Índice Búsqueda

Escriba la palabra clave a buscar:

Command

- Command
- CommandHistory
- CompareGeometry
- ConvertCurveToPolyline
- CopyMaterial
- CopyObject
- CopyObjects
- CopyObjectsToLayer
- Cosh
- CreatePreviewImage
- CreateShortcut
- CreateSolid
- CullDuplicateNumbers
- CullDuplicatePoints
- CullDuplicateStrings
- CurrentDetail
- CurrentDimStyle
- CurrentHatchPattern
- CurrentLayer
- CurrentModelInfo
- CurrentPrinter
- CurrentView
- Curve Methods
- CurveArcLengthPoint
- CurveArea
- CurveAreaCentroid
- CurveArrows
- CurveBooleanDifference
- CurveBooleanIntersection
- CurveBooleanUnion
- CurveBrepIntersect
- CurveClosestObject
- CurveClosestPoint
- CurveContourPoints
- CurveCurvature
- CurveCurveIntersection
- CurveDegree
- CurveDeviation
- CurveDim
- CurveDirectionsMatch
- CurveDiscontinuity

Mostrar

Command

Runs a Rhino command script. All Rhino commands can be used in command scripts. The command can be a built-in Rhino command or a command that is provided by a 3rd party plug-in.

Write command scripts just as you would type the command sequence at the command line. A space between characters or a new line act like pressing <Enter> at the command line. For more information on writing command scripts, see "Scripting" in the Rhino help file.

Note, this method is designed to run one command and one command only. Do not combine multiple Rhino commands into a single call to this method. For example:

WRONG:

```
Rhino.Command "_Line _SelLast _Invert"
```

CORRECT:

```
Rhino.Command "_Line"
```

```
Rhino.Command "_SelLast"
```

```
Rhino.Command "_Invert"
```

Also, the exclamation point and space character (!) combination used by button macros and batch-driven scripts to cancel the previous command is not valid. For example:

WRONG:

```
Rhino.Command "! _Line _Pause _Pause"
```

CORRECT:

```
Rhino.Command "_Line _Pause _Pause"
```

After the command script has run, you can obtain the identifiers of most recently created or changed object by calling [LastCreatedObjects](#).

Syntax

```
Rhino.Command (strCommand [, blnEcho])
```

After you add a curve, select it with:

Rhino.SelectObject(curveID)

Then apply the command:

**Rhino.Command "_Pipe " & 1.0 & " Enter "
& 1.0 & " Enter"**

Due next class...

Do something interesting of your
own!