**4.6 圓錐**

var Conversions = Core.Conversions;

var Point2D = Core.Point2D;

var Debug = Core.Debug;

var Path2D = Core.Path2D;

var Point3D = Core.Point3D;

var Matrix2D = Core.Matrix2D;

var Matrix3D = Core.Matrix3D;

var Mesh3D = Core.Mesh3D;

var Plugin = Core.Plugin;

var Tess = Core.Tess;

var Sketch2D = Core.Sketch2D;

var Solid = Core.Solid;

var Vector2D = Core.Vector2D;

var Vector3D = Core.Vector3D;

// Template Code:

params = [

{ "id": "numberOfPoints", "displayName": "# of cone sides", "type": "int", "rangeMin": 10, "rangeMax":330, "default": 20 },

{ "id": "radius\_a", "displayName": "Radius a of ecllipse", "type": "length", "rangeMin": 2, "rangeMax": 150, "default": 10 },

{ "id": "radius\_b", "displayName": "Radius b of ecllipse", "type": "length", "rangeMin": 2, "rangeMax": 150, "default": 10 },

{ "id": "top\_height", "displayName": "height of cone top", "type": "length", "rangeMin":2, "rangeMax": 150, "default": 10 },

{ "id": "bottom\_height", "displayName": "height of cone bottom", "type": "length", "rangeMin": -150, "rangeMax": 0, "default": 0 },

{ "id": "x\_pos",

"displayName": "x position",

"type": "float",

"rangeMin": -200,

"rangeMax": 200,

"default": 0

},

{ "id": "y\_pos",

"displayName": "y position",

"type": "float",

"rangeMin": -200,

"rangeMax": 200,

"default": 0

},

{ "id": "z\_pos",

"displayName": "z position",

"type": "float",

"rangeMin": -200,

"rangeMax": 200,

"default": 0

}

];

function ran(maxNum,minNum)

{

var re = Math.random() \* (maxNum - minNum ) + minNum;

return re;

}

function process(params)

{

var numPoints = params['numberOfPoints'];

var ndivs = Math.max(10, numPoints);

var h\_top = params["top\_height"];

var h\_bottom = params["bottom\_height"];

var ra = params["radius\_a"] ;

var rb = params["radius\_b"] ;

ra=Math.max(2, ra) ;

rb=Math.max(2, rb) ;

var x\_pos = params.x\_pos;

var y\_pos = params.y\_pos;

var z\_pos = params.z\_pos;

var down=[];

var star = [];

var mesh = new Mesh3D();

var bot = [x\_pos,y\_pos,z\_pos+h\_bottom];

var top = [x\_pos,y\_pos,z\_pos+h\_top];

var net=[];

net[0]=[];

for (var i = 0; i <= ndivs; i++) {

net[0][i]=[];

var a = i/ndivs \* Math.PI\*2;

var x = Math.cos(a) \* ra;

var y = Math.sin(a) \* rb;

net[0][i].push(x\_pos+x, y\_pos+y, z\_pos);

}

for (var i = 0; i < ndivs; i++) {

mesh.triangle(bot, net[0][i], net[0][i+1]);

mesh.triangle(top, net[0][i], net[0][i+1]);

}

return Solid.make(mesh);

}