МИНОБРНАУКИ РОССИИ  
ФГБОУ ВПО «ИжГТУ имени М.Т. Калашникова»

Кафедра «Программное обеспечение»

Отчет  
по лабораторной работе № 2  
по дисциплине  
«Логическое и функциональное программирование»

|  |  |
| --- | --- |
| Выполнил:  студент гр. Б05-191-1  Принял:  д.т.н., профессор | Мусин Р.И.  Ложкин А.Г. |

Ижевск 2017

Вариант 6

Постановка задачи

Написать программу на языке AutoLISP, проектирующую тело. Элементарные тела, в данном случае это 9-ти гранная пирамида, цилиндр и шаровой пояс, должны быть расположены вдоль оси X и состоять из полигонов. При проектировании тела шаровой пояс вычитается из цилиндра. Вышеуказанные фигуры имеют следующие параметры: пирамида (pyramid\_height – высота и pyramid\_radius – радиус основания), цилиндр (cylinder\_height – высота и cylinder\_radius – радиус основания), шаровой пояс (sphere\_radius – радиус основания и sphere\_height - высота).

Исходный код

(defun RotatePoint (point rotation\_angle / x y cos1 sin1 x1 y1)

(setq x (car point))

(setq y (cadr point))

(setq cos1 (cos rotation\_angle))

(setq sin1 (sin rotation\_angle))

(setq x1 (- (\* x cos1) (\* y sin1)))

(setq y1 (+ (\* x sin1) (\* y cos1)))

(list x1 y1)

)

(defun BuildPoints(point\_count / rotation\_angle points current\_point)

(setq rotation\_angle (/ (\* 2 PI) point\_count))

(setq points '())

(setq current\_point '(1 0))

(repeat point\_count

(setq points (cons current\_point points))

(setq current\_point (RotatePoint current\_point rotation\_angle))

)

points

)

(defun LeftEach (points idx / points2 index)

(setq points2 '())

(setq index 0)

(foreach point points

(if (zerop (rem index connect\_each)) (setq points2 (cons point points2)))

(setq index (+ index 1))

)

points2

)

(defun ScalePoint (point radius) (list (\* (car point) radius) (\* (cadr point) radius)))

(defun PreparePoint (point radius height) (cons height (ScalePoint point radius)))

(defun Connect (points height1 height2 radius1 radius2 / iteration point1 point2)

(foreach point points

(setq point1 (PreparePoint point radius1 height1))

(setq point2 (PreparePoint point radius2 height2))

(command "\_line" point1 point2 "")

)

(princ)

)

(defun DrawCircle (points radius height)

(setq points (cons (last points) points))

(setq prepare (lambda (point) (PreparePoint point radius height)))

(setq points (mapcar 'prepare points))

(command "\_line" (foreach pt points (command pt)))

(princ)

)

(defun DrawPyramid (points edge\_count height radius / connect\_each)

(setq connect\_each (/ (length points) edge\_count))

(setq points (LeftEach points connect\_each))

(Connect points 0 height 0 radius)

(setq layer\_count 5)

(setq height\_step (/ height (- layer\_count 1)))

(setq radius\_step (/ radius (- layer\_count 1)))

(setq height 0)

(setq radius 0)

(repeat layer\_count

(DrawCircle points radius height)

(setq height (+ height height\_step))

(setq radius (+ radius radius\_step))

)

points

)

(defun DrawCylinder (points radius height start\_height)

(Connect points start\_height (+ start\_height height) radius radius)

(setq layer\_count 5)

(setq height\_step (/ height (- layer\_count 1)))

(repeat layer\_count

(DrawCircle points radius start\_height)

(setq start\_height (+ start\_height height\_step))

)

(princ)

)

(defun DrawSphere (points radius height start\_height / layer\_count height\_step GetRadius last\_height next\_height last\_radius next\_radius)

(setq layer\_count 5)

(setq height\_step (/ height (- layer\_count 1)))

(setq GetRadius (lambda (h) (sqrt (- (\* radius radius) (\* h h)))))

(DrawCircle points radius start\_height)

(setq last\_height start\_height)

(repeat layer\_count

(setq next\_height (- last\_height height\_step))

(setq last\_radius (GetRadius (- start\_height last\_height)))

(setq next\_radius (GetRadius (- start\_height next\_height)))

(DrawCircle points next\_radius next\_height)

(Connect points next\_height last\_height next\_radius last\_radius)

(setq last\_height next\_height)

)

(Connect points last\_height last\_height 0 next\_radius)

(princ)

)

(defun start()

(setq points (BuildPoints 90))

(setq pyramid\_edge\_count 9)

(setq pyramid\_height (getdist "Choose pyramid height"))

(setq pyramid\_radius (getdist "Choose pyramid radius"))

(setq pyramid\_points (DrawPyramid points pyramid\_edge\_count pyramid\_height pyramid\_radius))

(setq cylinder\_height (getdist "Choose cylinder height"))

(setq cylinder\_radius (getdist "Choose cylinder radius"))

(setq cylinder\_side\_size (/ (length points) pyramid\_edge\_count))

(setq cylinder\_edge\_count (\* pyramid\_edge\_count cylinder\_side\_size))

(setq cylinder\_start\_height pyramid\_height)

(Connect pyramid\_points pyramid\_height cylinder\_start\_height pyramid\_radius cylinder\_radius)

(DrawCylinder points cylinder\_radius cylinder\_height cylinder\_start\_height)

(setq sphere\_height (getdist "Choose sphere height"))

(setq sphere\_radius (getdist "Choose sphere radius"))

(setq sphere\_edge\_count cylinder\_edge\_count)

(setq sphere\_start\_height (+ pyramid\_height cylinder\_height))

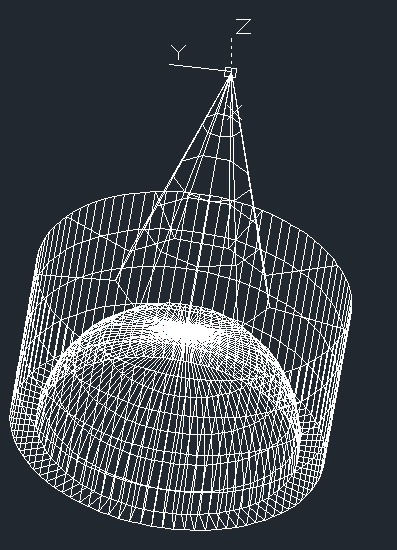
(Connect points sphere\_start\_height sphere\_start\_height cylinder\_radius sphere\_radius)

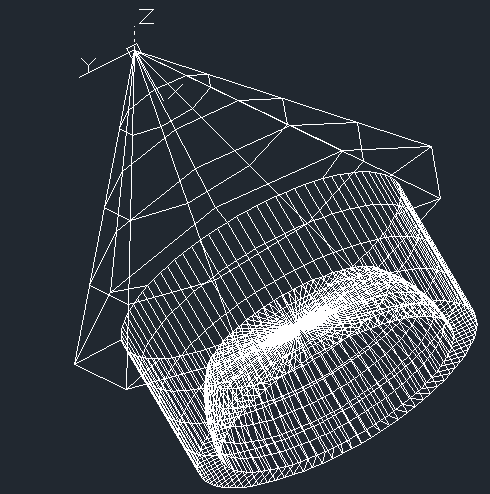
(DrawSphere points sphere\_radius sphere\_height sphere\_start\_height)

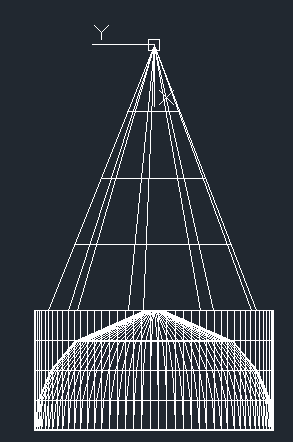
)

(start)

Пример работы программы







Вывод

В ходе выполнения лабораторной работы были получены навыки работы в среде AutoCad с функциональным языком Autolisp, рассмотрены основные особенности разработки программ на Lisp'e. Была разработана программа, удовлетворяющая всем поставленным требованиям.