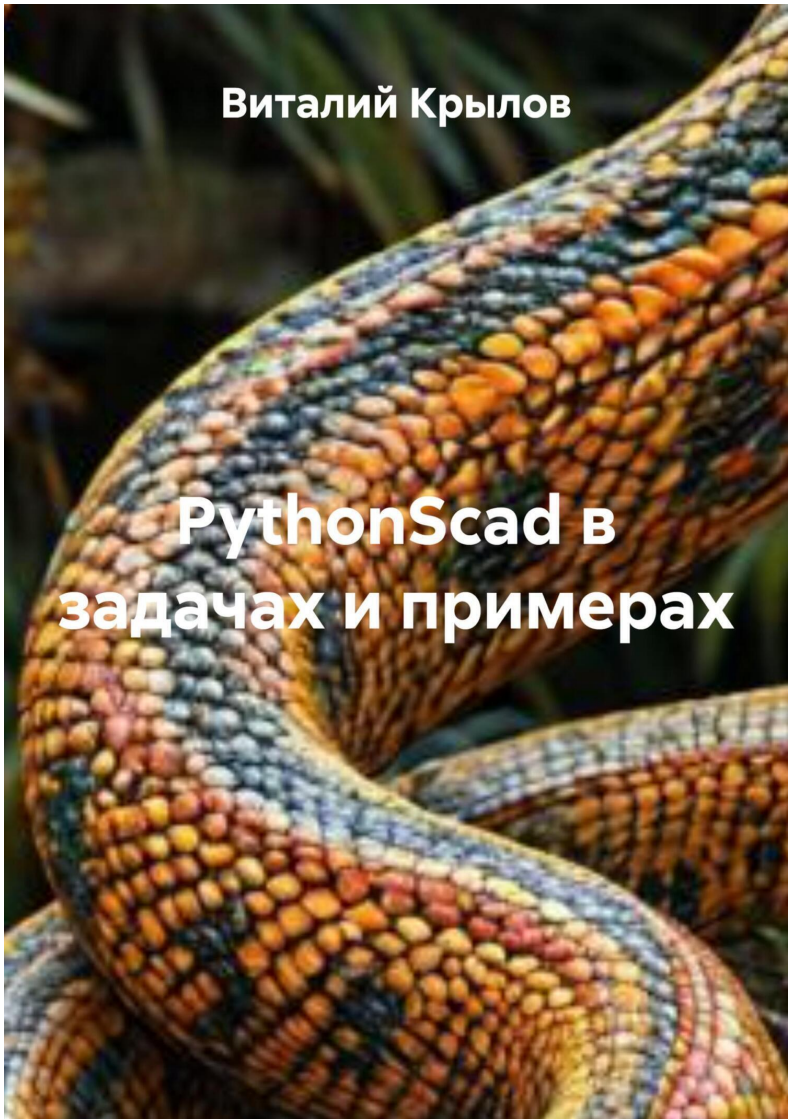
Виталий Крылов



PythonScad в задачах и примерах

Виталий Витальевич Крылов

PythonScad в задачах и примерах

http://www.litres.ru/pages/biblio_book/?art=72440416
SelfPub; 2026

Аннотация

Курс «Введение в PythonScad» обучает параметрическому 3D-моделированию с использованием языка PythonScad. Программа включает 18 модулей с теорией, примерами и тестами, позволяющими поэтапно освоить работу с примитивами, трансформациями, булевыми операциями, параметризацией и модулями. Особое внимание уделяется различиям синтаксиса OpenSCAD и PythonScad, что даёт возможность гибко применять оба инструмента. В курсе представлено множество практических заданий и детально разобранных примеров, способствующих формированию уверенных навыков проектирования. Завершается обучение выполнением финального проекта – созданием собственной 3D-модели. Курс рассчитан на школьников, студентов и начинающих разработчиков, интересующихся программированием и трёхмерным моделированием, и служит основой для дальнейшего углублённого изучения.

Виталий Крылов
PythonScad в задачах и примерах

Модуль 1: Введение в PythonScad

.1. Установка и настройка среды PythonSCAD

PythonSCAD – это интегрированная среда разработки (IDE), которая позволяет создавать и редактировать файлы OpenSCAD с помощью Python.

1. Установка OpenSCAD

1. Перейдите на сайт:https://openscad.org/ для загрузки OpenSCAD

2. В разделах выберите нужную версию (обычно это .exe файл).

3. Скачайте установочный файл и запустите его.

2. Установка Python 3.11

1. Перейдите на сайт:https://www.python.org/downloads/release/python-3110/ для загрузки Python 3.11

2. В разделах выберите нужную версию (обычно это .exe файл).

3. Скачайте установочный файл и запустите его.

3. Установка PythonSCAD

1. Перейдите на сайт:https://pythonscad.org/download.php

2. В разделах выберите нужную версию (обычно это .exe

файл).

3. Скачайте установочный файл и запустите его.
4. Следуйте инструкциям мастера установки:
5. Примите условия лицензионного соглашения.
6. Выберите папку для установки (по умолчанию).
7. Завершите установку.

4. Настройка PythonSCAD

После установки можно настроить программу под свои нужды.

Запуск программы:

На **Windows**: Запустите OpenSCAD через ярлык на рабочем столе или в меню «Пуск».

На **macOS**: Откройте из папки Applications.

На **Linux**: Откройте через меню приложений или выполните команду pythonscad в терминале.

Настройки программы:

Открытие настроек:

Перейдите в меню **"Edit" → "Preferences"** (или нажмите Ctrl+P).

Основные параметры:

General:

Выбор языка интерфейса.

Опция включения автоматического обновления программы.

Editor:

Настройка шрифтов и размера текста в редакторе.

Включение/выключение подсветки синтаксиса.

Настройки автодополнения.

3D View:

Настройка отображения объектов в 3D-просмотре.

Выбор темной или светлой темы интерфейса.

Render:

Настройка точности рендеринга (параметры для предварительного и финального рендеринга).

Горячие клавиши:

Вы можете настроить горячие клавиши для различных команд (например, рендеринг, предварительный просмотр и т.д.).

Настройка внешних редакторов (опционально):

В меню **Preferences** можно указать, какой внешний редактор будет использоваться для обработки дополнительных файлов, таких как скрипты Python или текстовые редакторы.

5. Дополнительные настройки

Установка библиотеки:

OpenSCAD поддерживает внешние библиотеки, которые можно использовать для упрощения работы с моделями.

Чтобы подключить библиотеку, нужно в коде использовать команду

use <library_path> или include <library_path>

Пример:

use <path_to_library>;

Обновления:

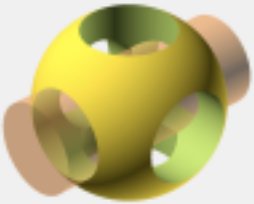
Программа автоматически проверяет наличие обновлений. Вы можете также вручную скачать и установить последнюю версию с официального сайта.

Заключение

Установка и настройка OpenSCAD проходят быстро и без проблем. Главное – не забывать периодически проверять наличие обновлений и настроить программу под личные предпочтения, чтобы сделать процесс моделирования максимально удобным.

.2. Интерфейс программы PythonSCAD

Welcome to Open(Python)SCAD - OpenSC



Open(Pyt

The Program

Создать | Python | Открыть | Помощь

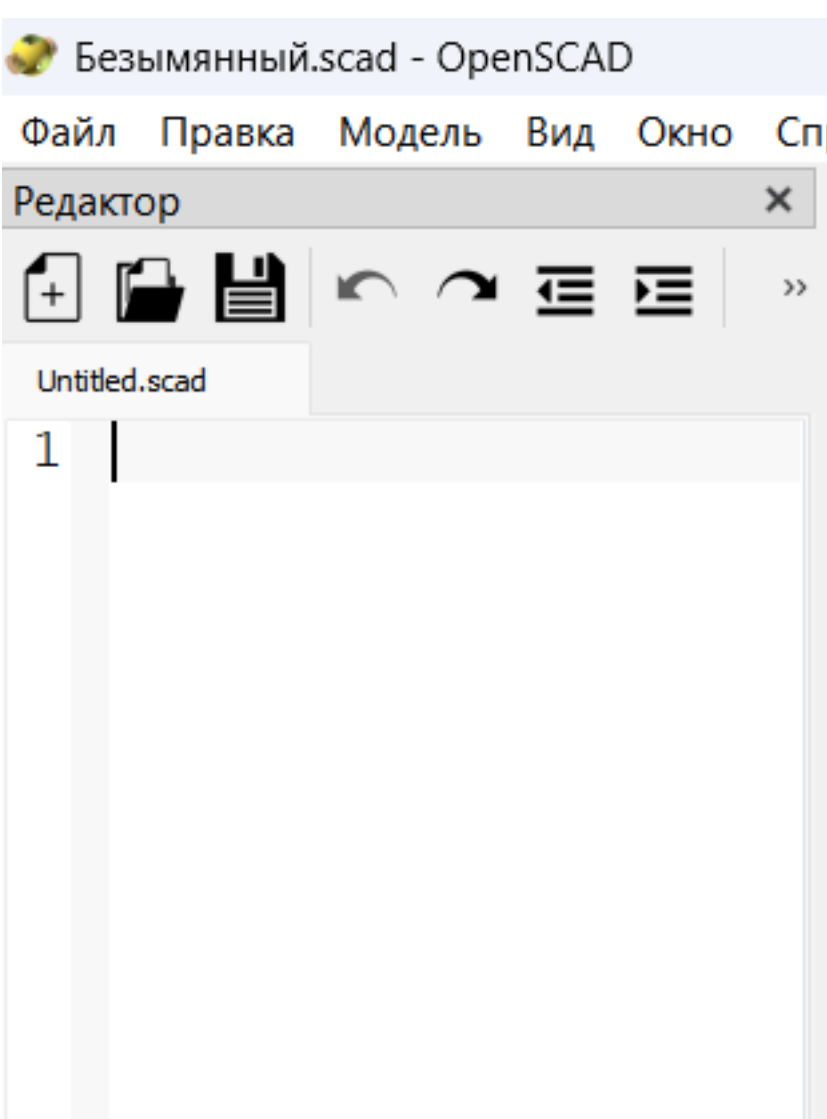
Недавние файлы

Безымянный.scad

Открыть недавний файл

Рис. 1. Окно запуска программы PythonScad.

Программа **PythonSCAD** имеет минималистичный интерфейс, ориентированный на текстовое программирование 3D-моделей. Он состоит из нескольких ключевых областей:


Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad
1

2. Окно программы PythonSCAD.

1. Главное окно

После запуска **PythonSCAD** отображает три основных панели:

Редактор кода (слева) – окно для написания кода модели.

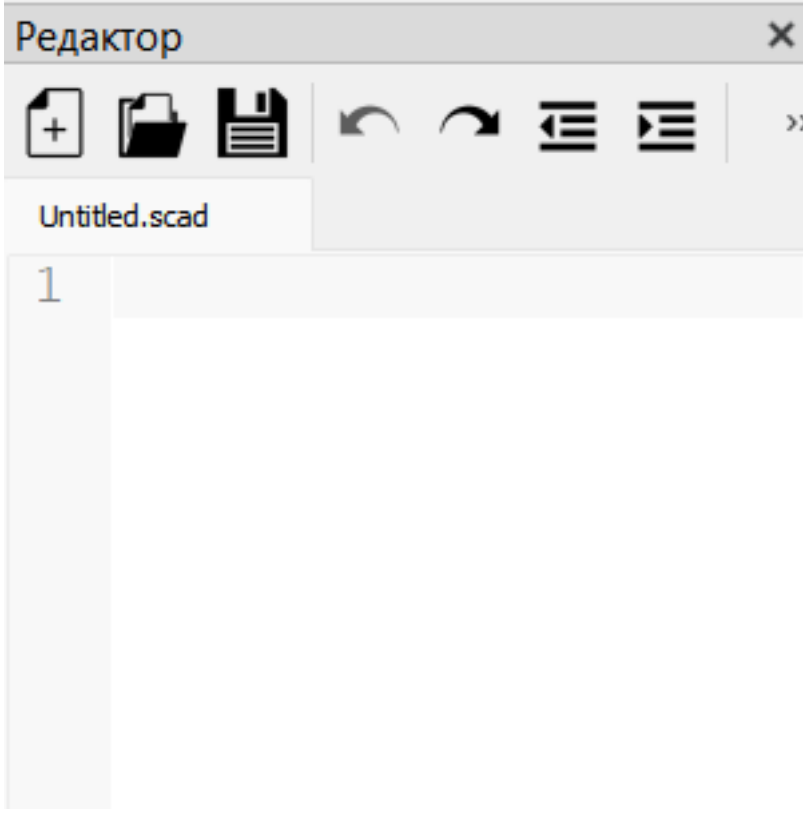


Рис. 3. Редактор кода программы PythonSCAD.

Окно рендеринга (справа) – отображает 3D-визуализацию модели.

60

Рис. 4. Окно рендеринга программы PythonSCAD.

Консоль вывода (внизу) – показывает сообщения об ошибках и предупреждениях.

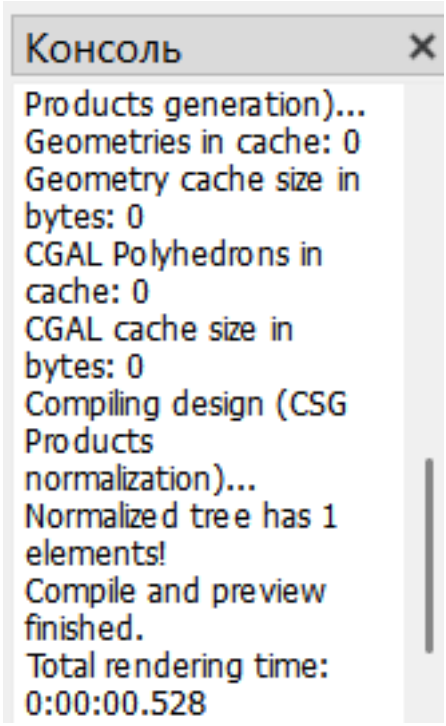


Рис. 5. Консоль программы PythonSCAD.

Также можно дополнительно отобразить другие панели PythonSCAD**:**

2. Панели и их функции

Редактор кода

Позволяет писать, редактировать и сохранять код **PythonSCAD**.

Поддерживает синтаксическую подсветку кода.

Возможны горячие клавиши для быстрого редактирования.

Окно рендеринга (3D-просмотр)

Визуализация 3D-модели на основе кода.

Поддерживает вращение (ПКМ), масштабирование (колесо мыши) и перемещение (Shift + ПКМ).

Верхняя панель содержит кнопки управления рендерингом.

Консоль сообщений

Отображает ошибки компиляции кода.

Показывает информацию о времени рендеринга и предупреждения.

3. Основные кнопки и меню

Меню "Файл"

New – создать новый проект.

Open – открыть существующий проект.

Save / Save As – сохранить код.

Export – экспорт 3D-модели в форматы STL, OFF, DXF.

Меню "Правка"

Undo / Redo – отмена и повтор изменений.

Find / Replace – поиск и замена текста в коде.

Правка Модель Вид Окно Справка

Отменить
Повторить

Вырезать
Копировать
Вставить

Добавить отступ
Убрать отступы
Закомментировать
Раскомментировать
Преобразовать табуляцию в пробелы
Установить/убрать закладку
Перейти к следующей закладке
Перейти к предыдущей закладке

Показать следующую вкладку

Рис. 6. Меню Правка программы PythonSCAD.

Меню "Вид"

Включает/выключает панели (Редактор, Консоль, 3D-просмотр)

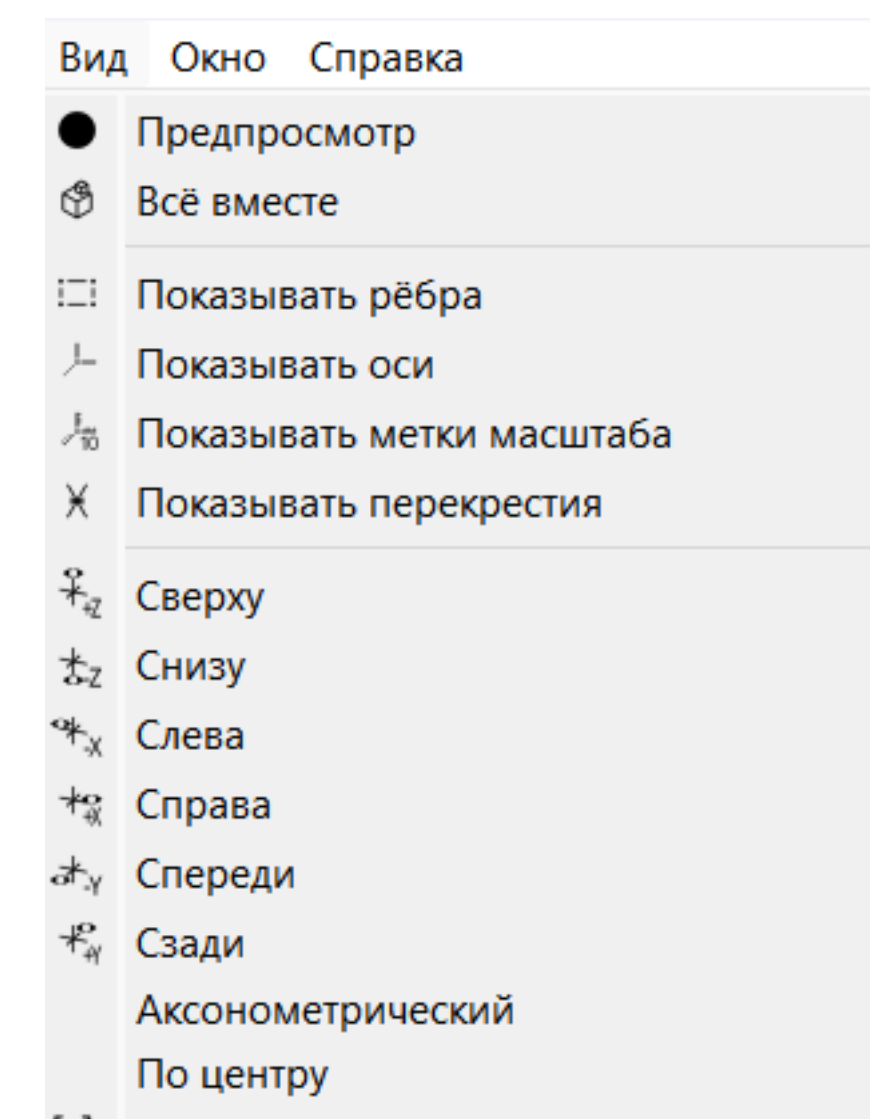

Вид
Окно
Справка
Предпросмотр
Всё вместе
Показывать рёбра
Показывать оси
Показывать метки масштаба
Показывать перекрестия
Сверху
Снизу
Слева
Справа
Спереди
Сзади
Аксонометрический
По центру

Рис. 7. Меню вид программы PythonScad.

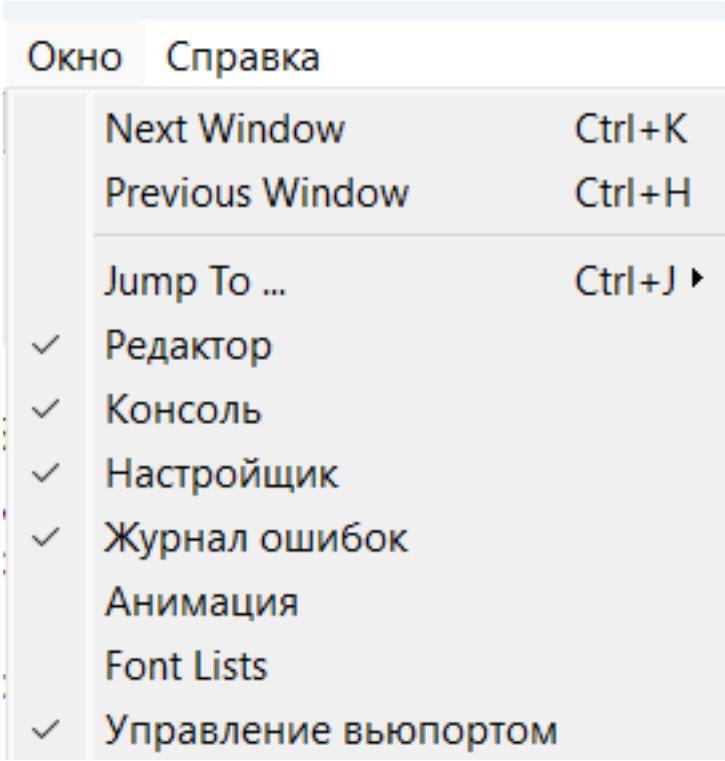


Рис. 8. Меню Окно программы PythonScad.

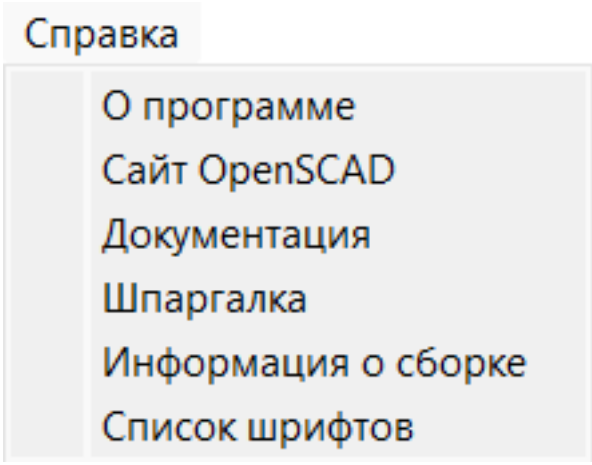


Рис. 9. Меню Справка программы PythonScad.

Кнопки управления рендерингом

[>] **Preview (F5) –** быстрый предварительный просмотр.

[img] **Render (F6) –** финальный рендеринг (более точный, но медленный).

[save] **Export STL –** экспорт модели в STL.

Модель Вид Окно Справка

- ✓ Автоматическое обновление и предпр
- Обновить и выполнить предпросмотр
- Предпросмотр
- Рендеринг
- 3D Печать
- Measure Distance
- Measure Angle
- ○ Find Handle
- Share Design
- Load shared Design

- Проверить корректность
- Показать AST...
- Display Python conversion
- Показать дерево CSG...
- Показать результаты CSG...

Рис. 10. Меню Модель программы PythonSCAD

4. Настройки (Preferences)

Изменение темы оформления.

Настройка шрифтов.

Установка точности рендеринга.

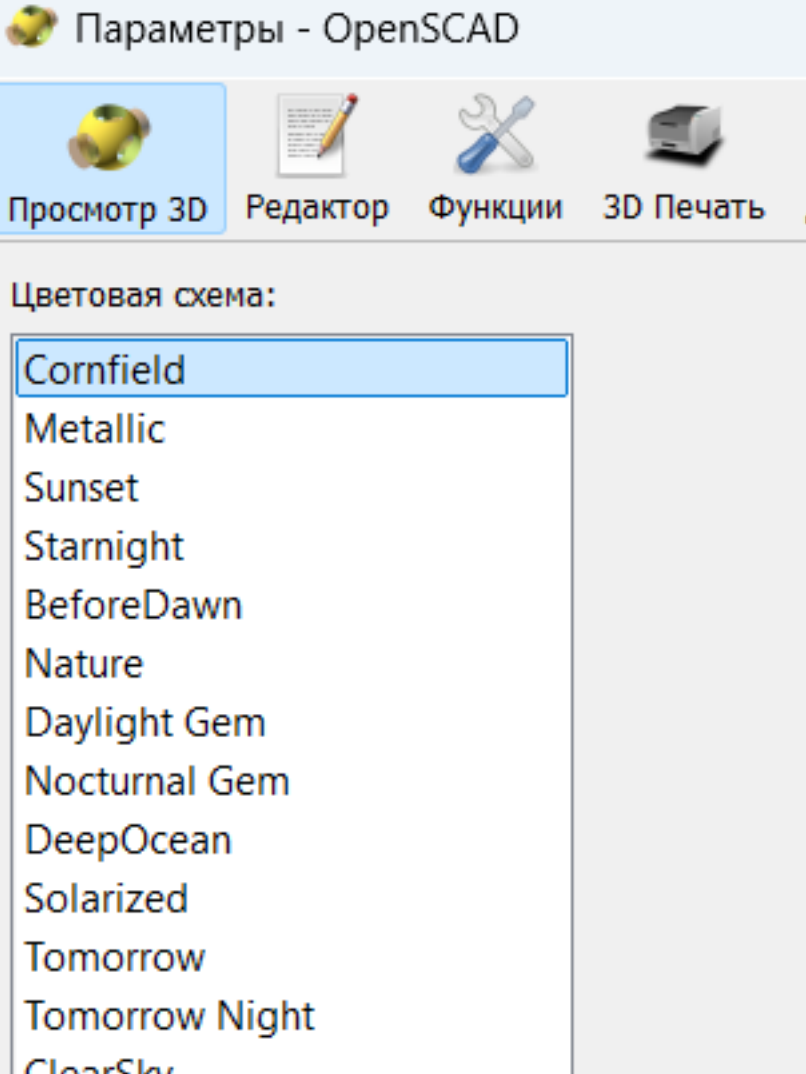

Параметры - OpenSCAD
Просмотр 3D
Редактор
Функции
3D Печать
Цветовая схема:
Cornfield
Metallic
Sunset
Starnight
BeforeDawn
Nature
Daylight Gem
Nocturnal Gem
DeepOcean
Solarized
Tomorrow
Tomorrow Night

Рис. 11. Меню правка программы PythonSCAD

Интерфейс **PythonSCAD** максимально ориентирован на разработчиков и математическое моделирование, без сложных графических элементов, как в других 3D-редакторах.

.3. Основные команды и операторы PythonSCAD(OpenSCAD)

PythonSCAD – это язык описания параметрических 3D-моделей. Он использует конструктивную геометрию (CSG) и операции над примитивами.

Определение 1:

Конструктивная сплошная геометрия (Constructive Solid Geometry, CSG) – это метод моделирования трехмерных объектов путем объединения простых геометрических примитивов (таких как кубы, сферы, цилиндры и др.) с использованием булевых операций (объединение, пересечение, разность). Этот метод широко используется в системах автоматизированного проектирования (CAD), таких как PythonSCAD, Blender и других.

Определение 2:

Системы автоматизированного проектирования (CAD) представляют собой программное обеспечение, предназначенное для автоматизации процесса проектирования различных объектов, будь то механические детали, архитектурные сооружения, электронные схемы или даже произведения искусства. CAD-системы помогают инженерам, архитек-

торам, дизайнерам и другим специалистам создавать точные чертежи, модели и схемы, а также проводить анализ и симуляцию поведения конструкций.

1. Листинг:Примитивные фигуры

cube(size, center=true/false); – создаёт куб.
cube([10, 20, 30]); // Куб 10×20×30
sphere(.....); – создаёт сферу.
sphere(10); // Радиус 10
cylinder(h, r1, r2); – создаёт цилиндр.
polyhedron(points, faces); – создаёт многогранник.

2. Листинг:Логические операции (CSG-моделирование)

union() {.....} – объединение фигур.
difference() {.....} – вычитание одной фигуры из другой.
intersection() {.....} – пересечение фигур.

```
difference() {

cube(20);

sphere(15);

}
```

3. Листинг:Трансформации

translate([x, y, z]) – сдвиг.
translate([10, 0, 0]) cube(5);
rotate([x, y, z]) – поворот.

rotate([0, 0, 45]) cube(10);
scale([x, y, z]) – масштабирование.
scale([2, 1, 1]) cube(10);
mirror([x, y, z]) – отражение.
mirror([1, 0, 0]) cube(10);
resize([x, y, z]) – изменение размеров.
resize([20, 30, 10]) sphere(10);

4. Листинг:Модули и переменные

module name() { … } – создание модуля (функции).

```
module my_shape() {
cube(10);
}
my_shape();
```

for (i=[start:step:end]) { … } – цикл.

```
for (i=[0:10:30]) {
translate([i, 0, 0]) sphere(5);
}
```

if (condition) { … } – условие.
function name(args) = expression; – функция.
function square(x) = x * x;

Выше основные команды PythonSCAD(OpenSCAD). Есть и другие возможности, такие как color(), hull(), minkowski(), projection(), но это уже более продвинутые темы.

1.4. Основные команды и операторы PythonSCAD(Python)

1. Примитивные фигуры

from openscad import*—обязательная команда для работы кода

cu = cube(x, y, z)—создаём переменную "cu" и закрепляем за ней куб "cube([z, y, z])"

cu = cube(5) //создаём куб размером 5

cy = cylinder(h, r1, r2)—создаём переменную "cy" и закрепляем за ней цилиндр "cylinder(h, r1, r2)"

cy = cylinder(10, 5, 5) //Цилиндр высотой 10 и размерами 5

show([cu,cy])—Команда для демонстрации модели

2. Логические операции (CSG-моделирование)

fusion = union([.....])—Соединение форм

fusion = union([cu,cy]) // соединили куб и цилиндр

diff = .difference(.....)—Вычитание форм

diff = cy.difference(cu) // вычли цилиндр из куба

inter = .intersection(.....)—Пересечение форм

inter = cu.intersection(sp)

3. Трансформации

rotated=.rotate([.....])—Поворот фигуры

rotated=cu.rotate([10,20,30]) // Повернули куб по осям X, Y, Z на 10, 20, 30 градусов соответственно

= .translate([.....])—Перемещение фигуры

cu = cu.translate([1, 2, 3]) // Переместили куб по осям X,

Y, Z на 1, 2, 3 соответственно

4. Модули и переменные

_ = .color(".....")—Покраска фигуры

cu_red = cu.color("red") // покрасили куб в красный

['name']="....."—Задаём имя фигуре

cu['name']="супер_куб" // Задаем кубу "cu" имя "супер_куб"

show(.....)—показываем фигуру

show(cu) // показываем куб "cu"

Существует команда команда позволяющая полностью перенести синтаксис **OpenSCAD** в **PythonSCAD(Python)**

```
from openscad import *
obj = scad("""
.....
""")
obj.show()
```

1.5. Примеры написания кода и результат(OpenSCAD)

1. Примеры базовых примитивов

Параллелепипед

```
cube([10, 20, 30]);
```

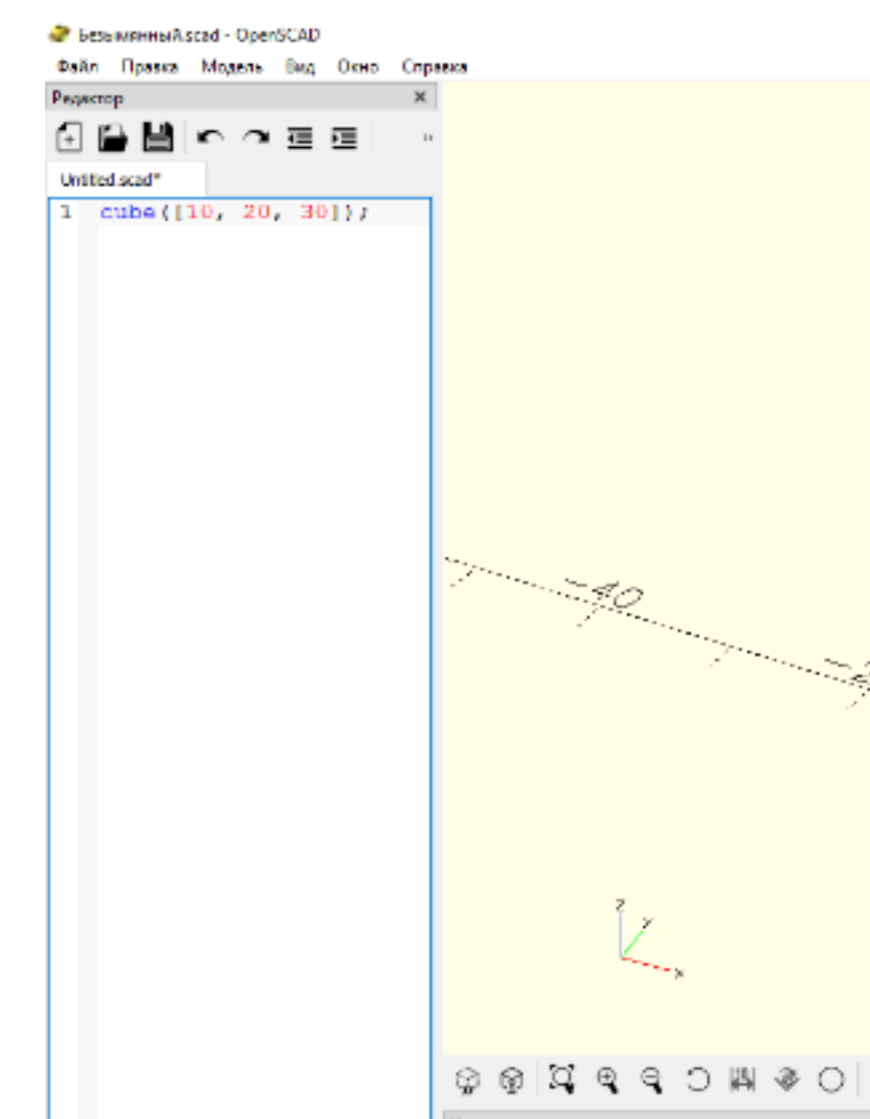

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.scad*
1 cube([10, 20, 30]);
40
Z
Y
X

Рис. 15. Пример параллелепипеда

Сфера
sphere(10);

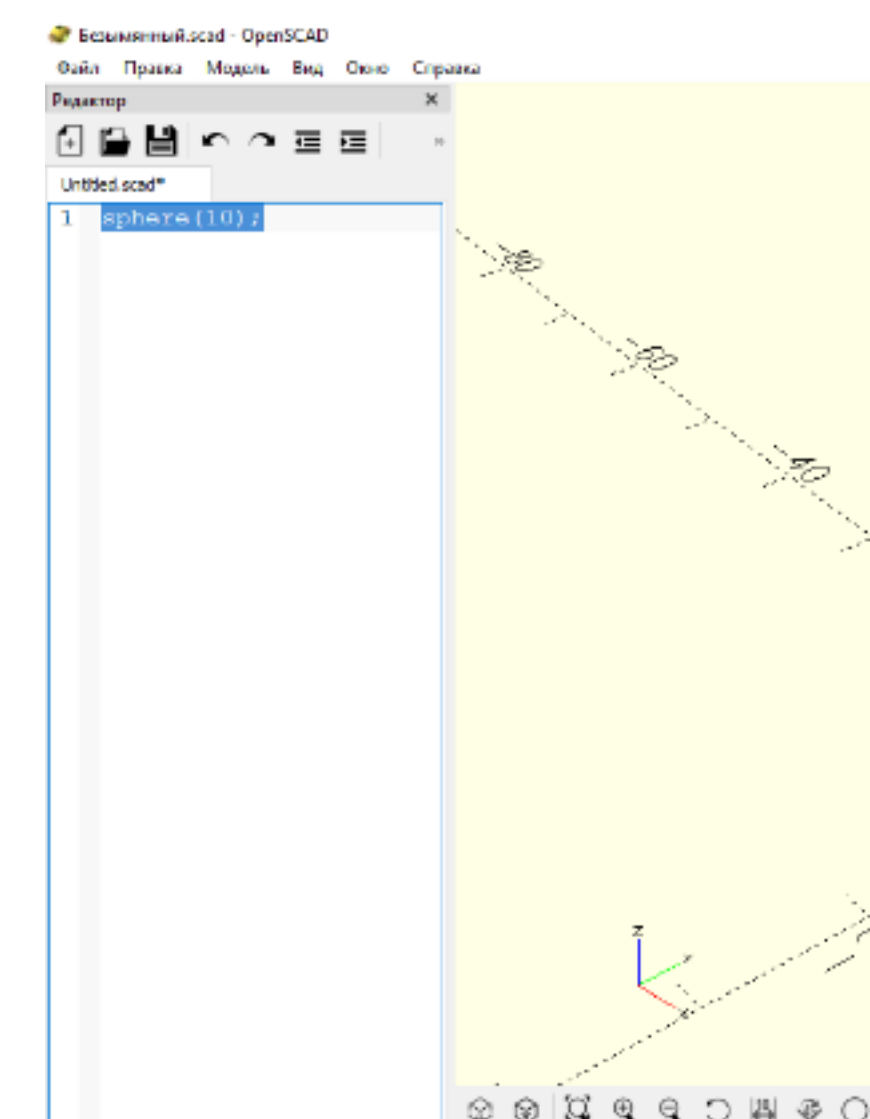

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.scad*
1 sphere(10);
Z

Рис. 12. Пример сферы

Цилиндр

cylinder(10, 5, 5);

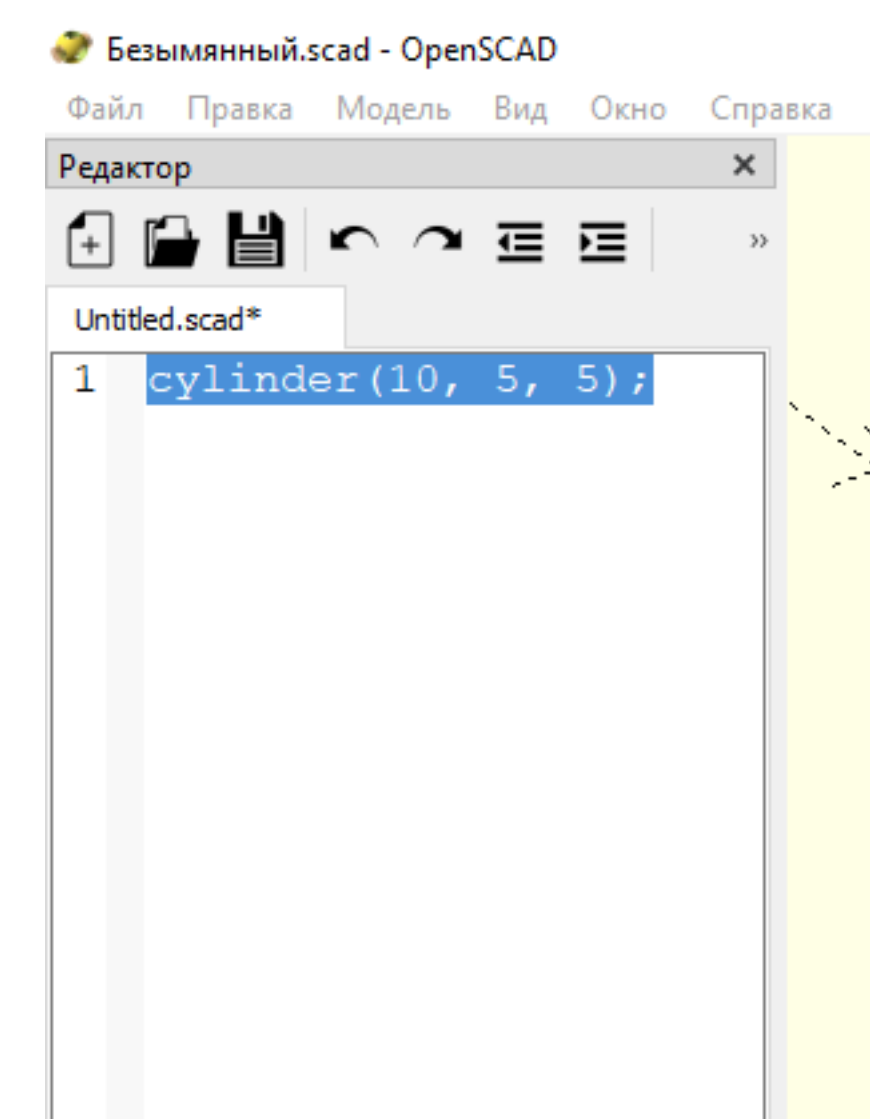

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.scad*
1 cylinder(10, 5, 5);

Рис. 16. Пример цилиндра

2. Логические операции
Соединение фигур

```
union() {
sphere(10);
cube([10, 20, 30]);
}
```

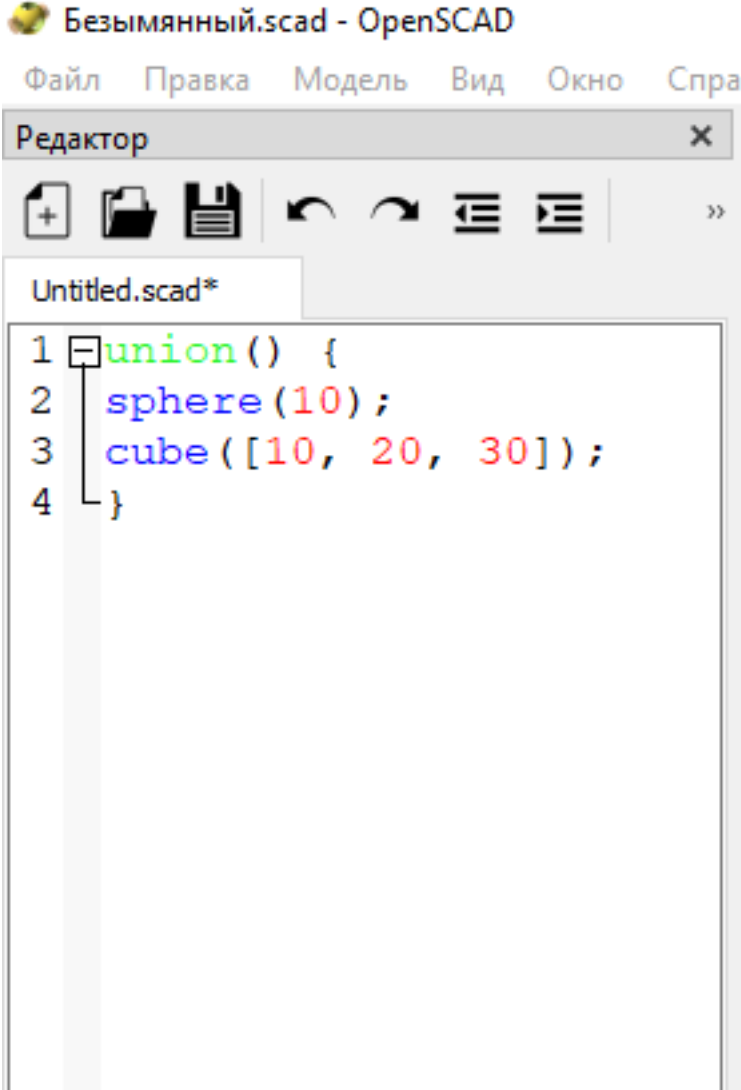

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.scad*
1 union() {
2 sphere(10);
3 cube([10, 20, 30]);
4 }

Рис. 17. Пример соединения сферы и параллелепипеда

Вычитание фигур

```
difference() {
sphere(10);
cube([10, 20, 30]);
}
```

Безымянный.scad - OpenSCAD

Файл Правка Модель Вид Окно Справка

Редактор

Untitled.scad*

```
1 difference() {
2   sphere(10);
3   cube([10, 20, 30]);
4 }
```

Рис. 18. Урезанная сфера

Пересечение фигур

```
intersection() {
sphere(10);
cube([10, 20, 30]);
}
```

Безымянный.scad - OpenSCAD

Файл Правка Модель Вид Окно Справка

Редактор

Untitled.scad*

```
1 intersection() {
2   sphere(10);
3   cube([10, 20, 30]);
4 }
```

Рис. 18. Пример урезанной сферы

Примеры написания кода и результат(Python)

1. Примеры базовых примитивов
Куб

```
from openscad import *
cu = cube(5)
show(cu)
```

Файл Правка Модель Вид Окно Справка

Редактор

Untitled.py*

```
1  from openscad import *
2  cu = cube(5)
3  show(cu)
```

Рис. 19. Пример куба

Цилиндр

```
from openscad import *
cy = cylinder(10, 5, 5)
show(cy)
```

Файл Правка Модель Вид Окно Справка

Редактор

Untitled.py*

```
1  from openscad import *
2  cy = cylinder(10, 5, 5)
3  show(cy)
```

Рис. 20. Пример цилиндра

Сфера

```
from openscad import *
sp = sphere(5)
show(sp)
```

Файл Правка Модель Вид Окно Справка

Редактор

Untitled.py*

```
1  from openscad import *
2  sp = sphere(5)
3  show(sp)
```

Рис. 21. Пример сферы

2. Логические операции
Соединение фигур

```
from openscad import *
sp = sphere(5)
cu = cube(6)
fusion = union([sp,cu])
show(fusion)
```

Файл Правка Модель Вид Окно Справка

Редактор

Untitled.py*

```
1  from openscad import *
2  sp = sphere(5)
3  cu = cube(6)
4  fusion = union([sp,cu])
5  show(fusion)
```

Рис. 22. Пример соединения куба и сферы

Вычитание фигур

```
from openscad import *
sp = sphere(5)
cu = cube(6)
diff = cu.difference(sp)
show(diff)
```

Файл Правка Модель Вид Окно Справка

Редактор

Untitled.py*

```
1  from openscad import *
2  sp = sphere(5)
3  cu = cube(6)
4  diff = cu.difference(
       sp)
5  show(diff)
```

Рис. 23. Пример вычитания сферы и куба

Пересечение фигур

```
from openscad import *
sp = sphere(5)
cu = cube(6)
inter = cu.intersection(sp)
show(inter)
```

Файл Правка Модель Вид Окно Справка

Редактор

Untitled.py*

```
1  from openscad import *
2  sp = sphere(5)
3  cu = cube(6)
4  inter = cu.
       intersection(sp)
5  show(inter)
```

Рис. 24. Пример части сферы

1.6. Тест по основам PythonSCAD

Задача 1: Создание куба

Напишите код, который создает куб с длиной ребра 10 мм.

А) cube(10);

Б) sphere(10);

В) cylinder(10);

Г) cube(5);

Задача 2: Создание сферы

Напишите код, который создает сферу радиусом 5 мм.

А) sphere(5);

Б) cube(5);

В) cylinder(5);

Г) sphere(10);

Задача 3: Создание цилиндра

Напишите код, который создает цилиндр высотой 20 мм и радиусом 3 мм.

А) cylinder(h=20, r=3);

Б) cylinder(h=10, r=3);

В) cylinder(h=20, r=5);

Г) sphere(20, 3);

Задача 4: Перемещение объекта

Напишите код, который перемещает куб с длиной ребра 10 мм на 5 мм вдоль оси X.

А) translate([5, 0, 0]) cube(10);

Б) rotate([5, 0, 0]) cube(10);

В) scale([5, 0, 0]) cube(10);

Г) mirror([5, 0, 0]) cube(10);

Задача 5: Поворот объекта

Напишите код, который поворачивает куб с длиной ребра 10 мм на 45 градусов вокруг оси Z.

А) rotate([0, 0, 45]) cube(10);

Б) rotate([0, 45, 0]) cube(10);

В) translate([0, 45, 0]) cube(10);

В) scale([0, 45, 0]) cube(10);

Задача 6: Создание массива

Создайте массив, содержащий координаты трех точек: (0,0,0), (10,0,0) и (0,10,0).

А) points = [[0,0,0], [10,0,0], [0,10,0]];

Б) points = [0,0,0], [10,0,0], [0,10,0];

В) points = (0,0,0), (10,0,0), (0,10,0);

Г) points = {0,0,0}, {10,0,0}, {0,10,0};

Задача 7: Использование цикла for

Используя цикл for, создайте ряд из 5 кубов с длиной ребра 10 мм, расположенных на расстоянии 20 мм друг от друга вдоль оси X.

А) for (i = [0:4]) { translate([i * 20, 0, 0]) cube(10); }

Б) for (i = [0:5]) { translate([i * 20, 0, 0]) cube(10); }

В) for (i = [0:4]) { translate([i * 10, 0, 0]) cube(10); }

Г) for (i = [0:4]) { rotate([i * 20, 0, 0]) cube(10); }

Задача 8: Использование цикла foreach

Используя цикл foreach, создайте три сферы радиусом 5 мм, расположенные в точках, заданных массивом из вопроса 6.

А) foreach(point = points) { translate(point) sphere(5); }

Б) foreach(point = points) { rotate(point) sphere(5); }

В) foreach(point = points) { scale(point) sphere(5); }

Г) foreach(point = points) { mirror(point) sphere(5); }

Задача 9: Создание пересечения

Создайте пересечение двух цилиндров радиусом 5 мм и высотой 20 мм, один из которых повернут на 90 градусов вокруг оси X.

А) intersection() { cylinder(h=20, r=5); rotate([90, 0, 0]) cylinder(h=20, r=5); }

Б) union() { cylinder(h=20, r=5); rotate([90, 0, 0]) cylinder(h=20, r=5); }

В) difference() { cylinder(h=20, r=5); rotate([90, 0, 0]) cylinder(h=20, r=5); }

Г) intersection() { cylinder(h=20, r=5); rotate([0, 90, 0]) cylinder(h=20, r=5); }

Задача 10: Создание объединения

Создайте объединение двух кубов с длиной ребра 10 мм, один из которых смещен на 5 мм вдоль оси X.

А) union() { cube(10); translate([5, 0, 0]) cube(10); }

Б) intersection() { cube(10); translate([5, 0, 0]) cube(10); }

В) difference() { cube(10); translate([5, 0, 0]) cube(10); }

Г) union() { cube(10); rotate([5, 0, 0]) cube(10); }

Задача 11: OpenSCAD

Что такое OpenSCAD?

А) Программное обеспечение для создания 3D-моделей с помощью кодирования

Б) Графический редактор для рисования 2D-изображений

В) Программа для анимации 3D-моделей

Г) Фоторедактор с поддержкой 3D-графики

Д) Онлайн-платформа для создания игр

Задача 12: полупрозрачный объект

Как сделать объект полупрозрачным?

А) % перед объектом

Б) transparency()

В) alpha(0.5)

Г) transparent()

Д) opacity(50%)

Задача 13: предпросмотр 3D-модели

Как запустить предпросмотр 3D-модели?

А) Нажать F5

Б) Нажать Shift + P

В) Нажать Ctrl + Z

Г) Нажать Ctrl + R

Д) Нажать Alt + P

Задача 14: экспорт 3D-моделей

Какой формат используется для экспорта 3D-моделей?

А) .stl
Б) .png
В) .svg
Г) .obj
Д) .jpeg

Задача 15: отразить объект относительно оси

Как отразить объект относительно оси?
А) mirror()
Б) invert()
В) flip()
Г) reverse()
Д) reflect()

Задача 16:Сохранение модели

Как сохранить модель в формате STL?
А) файл>экспорт
Б) справка>экспорт
В) вид>экспорт
Г) окно>экспорт

1.7. Тест по основам PythonSCAD

Задача 1:Начало кода

Что нужно писать в самом начале кода?
А) from openscad import *
Б) union()
В) difference()
Г) From OpenSCAD Import()

Задача 2:Создание куба

Какой командой создаётся куб?

А) cu = cube()

Б) c = cube()

В) cy = cylinder()

Г) cube()

Задача 3:Переменные

Можно поставить любое название для переменной?

А) Да

Б) Нет

Задача 4:Создание цилиндра

Какой командой создаётся цилиндр?

А) v = cylinder(6, 10, 5)

Б) cy = cylinder()

В) sp = sphere()

Г) cylinder()

Задача 5:Покраска фигур

Как покрасить фигуру в красный цвет?

А) cu_red = cu.color("red")

Б) cu_red = cu.color("green")

В) color("red")

Г) color(3, 8, 0)

Задача 6:Соединение фигур

Как соединить фигуры?

А) fusion = union([])

Б) diff = .difference()

В) difference()
Г) нельзя соединить

Задача 7:Вычитание фигур

как вычесть фигуры?
А) diff = .difference()
Б) difference()
В) Union()
Г) c = diff()

Задача 8:Наименование фигур

Как задать фигуре имя "Гринч"?
А) ['name']="Гринч"
Б) name = Гринч
В) name = "Гринч"
Г) Figurename = Grinch

Задача 9:Поворот Фигур

Как повернуть Фигуру?
А) rotated=.rotate([])
Б) tr = .translate([])
В) rotate()
Г) translate()

Задача 10:Перемещение Фигур

Как переместить Фигуру?
А) tr = .translate([])
Б) t = translate()
В) rotated=.rotate([])
Г) r = rotate

Задача 11:Основы

Какие две основы имеет PythonSCAD?

А) OpenSCAD

Б) Python

В) Blender

Г) C++

Д) Компас 3-D

Е) 3-D MAX

Задача 12:Разница

Чем отличается операция difference от intersection?

А) difference вычитает один объект из другого, а intersection оставляет только ту часть объекта которая перекрывается

Б) ни одна из команд не относиться к PythonSCAD

В) intersection вычитает один объект из другого, а difference оставляет только ту часть объекта которая перекрывается

Г) Обе команды делают одно и тоже

Задача 13:Версия

Какая версия Python нужна для PythonSCAD?

А) 3.11

Б) 4.11

В) 3.15

Г) 6.10

Задача 14:Примитивы

Какие из фигур входят в базовые примитивы?

А) Cube

Б) Sphere

В) Cylinder

Г) Polyhedron

Задача 15:Python

Что такое Python?

А) Python – мультипарадигмальный высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью.

Б) Python – слайсер 3D-моделей с открытым исходным кодом для 3D-принтеров. Программа позволяет нарезать на слои и печатать сложные модели из различных материалов.

В) Python – слайсер 3D-моделей с открытым исходным кодом для 3D-принтеров. Программа позволяет нарезать на слои и печатать сложные модели из различных материалов.

Г) Python – программа для захвата видео с экрана устройств с операционной системой Windows. Позволяет создавать видеоуроки, делиться процессом прохождения игры, сохранять на ПК вебинары или ТВ-передачи.

Задача 16:F5

Для чего применяется на F5 по умолчанию в PythonSCAD?

А) Рендер

Б) булевые операции

В) прозрачный режим

Г) Центрирование

Модуль 2: Геометрические примитивы

2.1. Геометрический примитив "Куб" (cube) в PythonSCAD(OpenSCAD)

Примитив cube в **PythonSCAD(OpenSCAD)** используется для создания прямоугольного параллелепипеда (или куба, если все его размеры равны). Он является одним из основных примитивов для построения 3D-объектов и часто используется в CSG-моделировании.

Синтаксис

:

cube(size, center=true/false);

size – размеры куба. Это может быть:

Число – если указано одно число, то создается куб с длиной каждой стороны, равной этому числу.

Массив из трех чисел – задает размеры по осям X, Y и Z (ширина, высота и глубина).

center=true/false – параметр, который определяет, будет ли куб центрирован в начале координат. Если установлено true, то центр куба будет совпадать с началом координат, если false, то одна из сторон куба будет располагаться в начале координат.

Пример: Куб с одной стороной

cube(10); // Куб с длиной каждой стороны 10

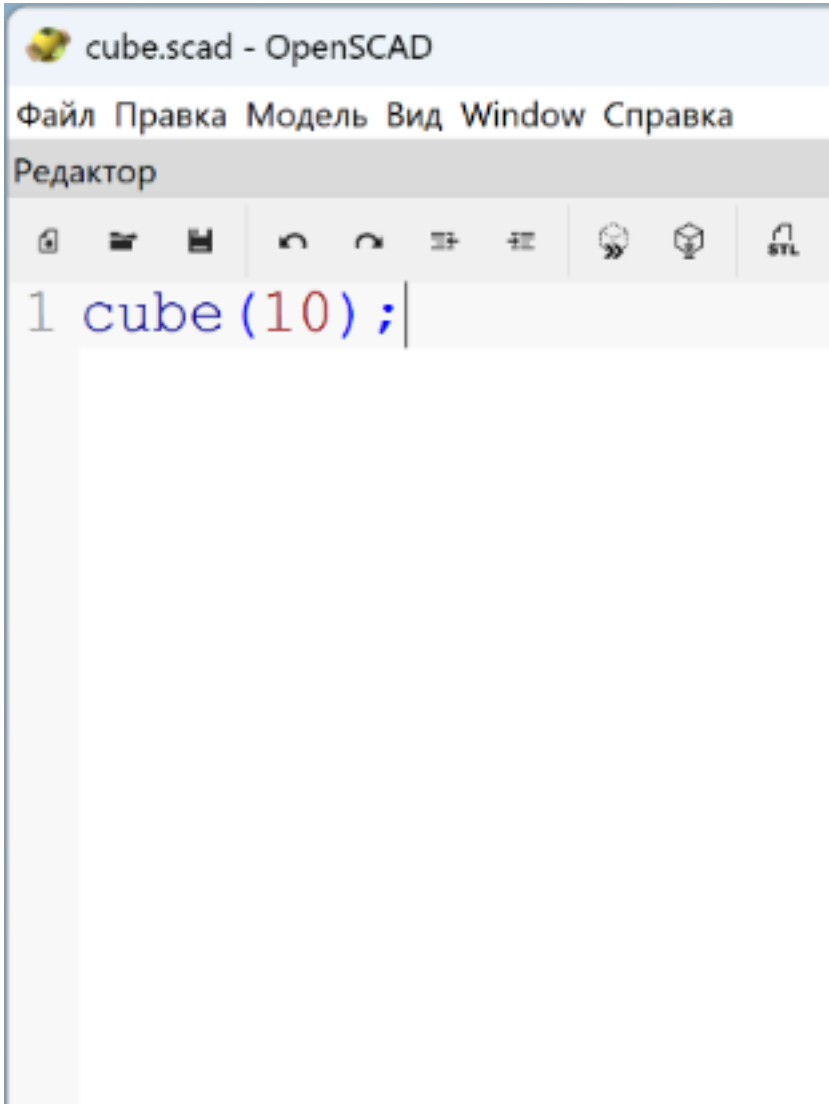

cube.scad - OpenSCAD
Файл Правка Модель Вид Window Справка
Редактор
STL
1 cube(10);

Рис. 1. Куб

Пример: Куб с разными размерами

cube([10, 20, 30]); // Прямоугольный параллелепипед размером 10x20x30

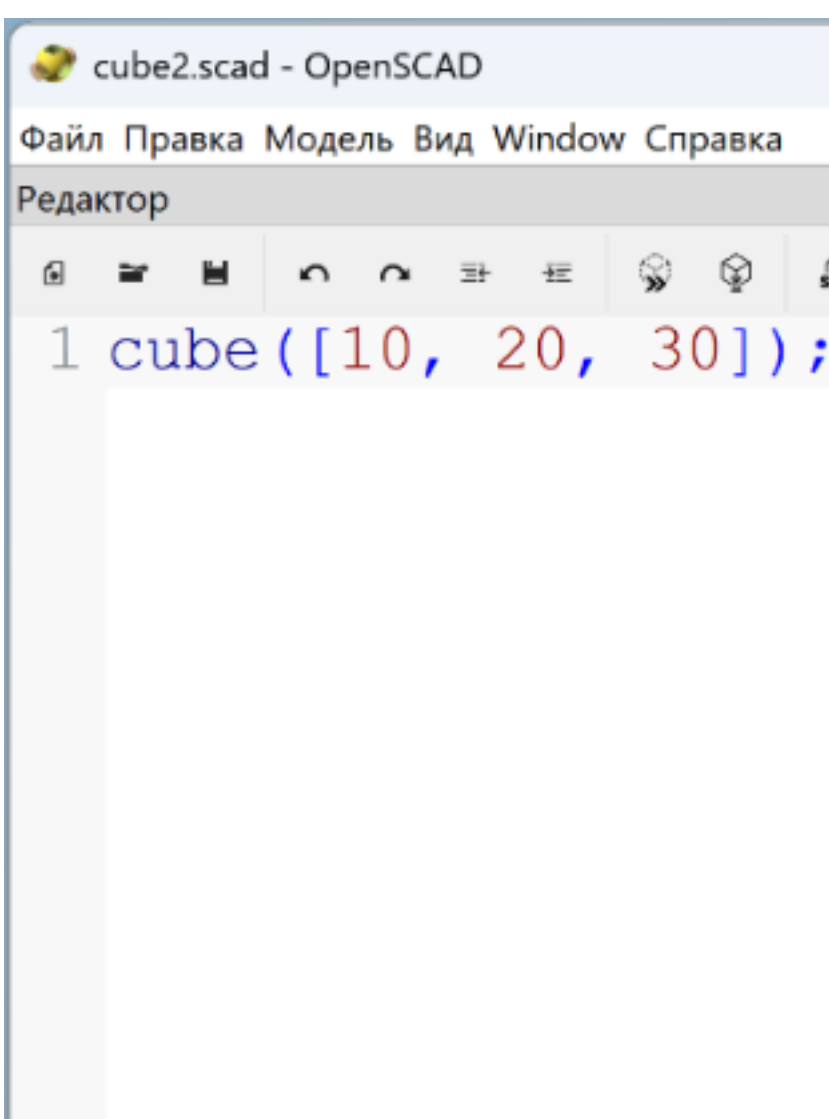

cube2.scad - OpenSCAD
Файл Правка Модель Вид Window Справка
Редактор
1 cube([10, 20, 30]);

Рис. 2. Цилиндр

Пример: Куб с центровкой

cube(10, center=true); // Куб с размером 10, центрированный в начале координат

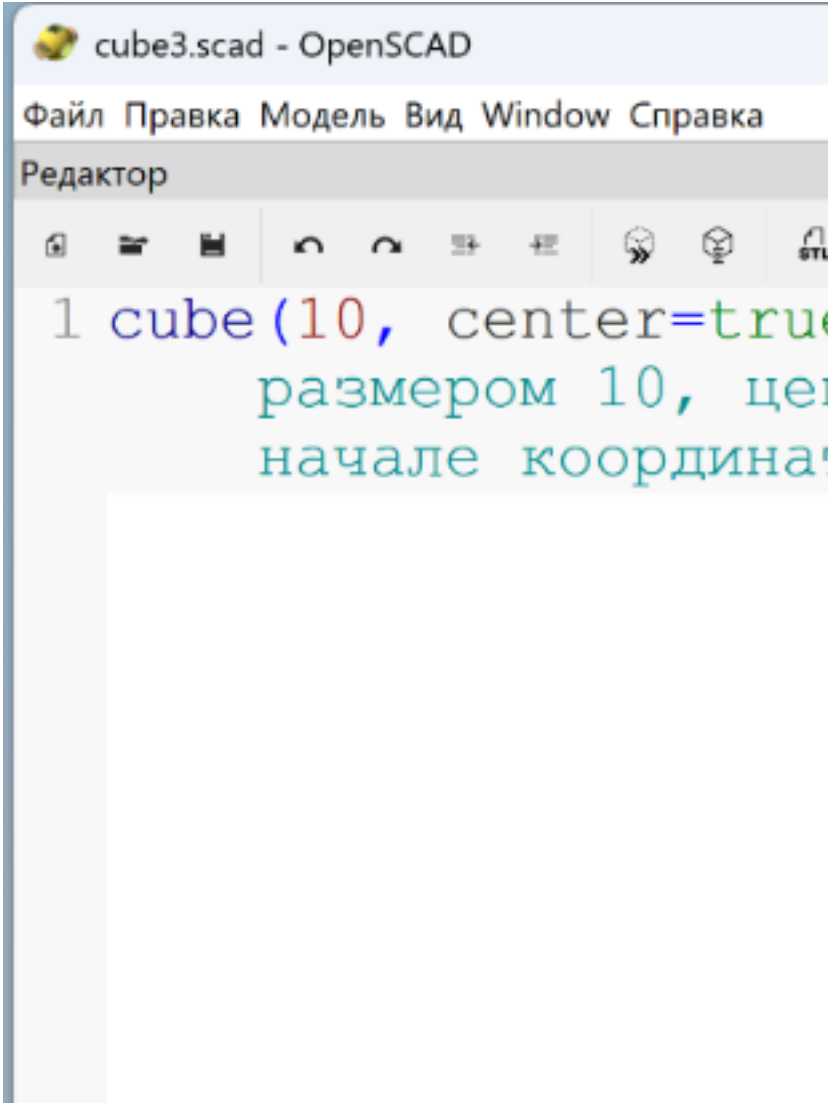

cube3.scad - OpenSCAD
Файл Правка Модель Вид Window Справка
Редактор
1 cube(10, center=tru
размером 10, це
начале координа

Рис. 3. Куб центрированый

В этом примере куб будет иметь размер 10×10×10, и его центр будет располагаться в начале координат.

Пример 4: Куб без центровки

cube(10, center=false); // Куб с размером 10, одна из сторон будет в начале координат

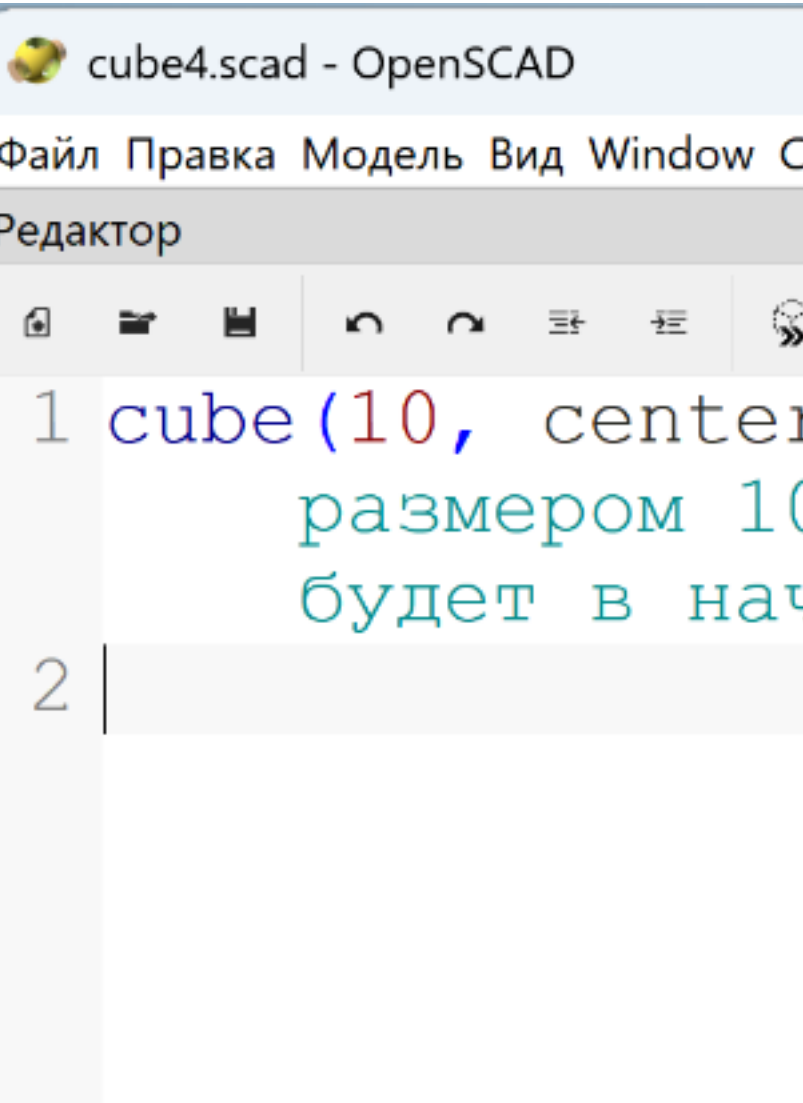

cube4.scad - OpenSCAD
Файл Правка Модель Вид Window C
Редактор
1 cube(10, center
размером 10
будет в нач
2

Рис. 4. Куб не центрированый

В этом случае куб будет иметь размер 10×10×10, и его одна из сторон будет расположена в начале координат.

Дополнительные параметры

Вы также можете использовать $fn для увеличения точности рендеринга, например, при использовании кубов с округлыми углами.

Пример 5: Куб с округлыми углами

```
$fn = 100; // Увеличиваем количество граней для округления
cube([10, 20, 30], center=true);
```

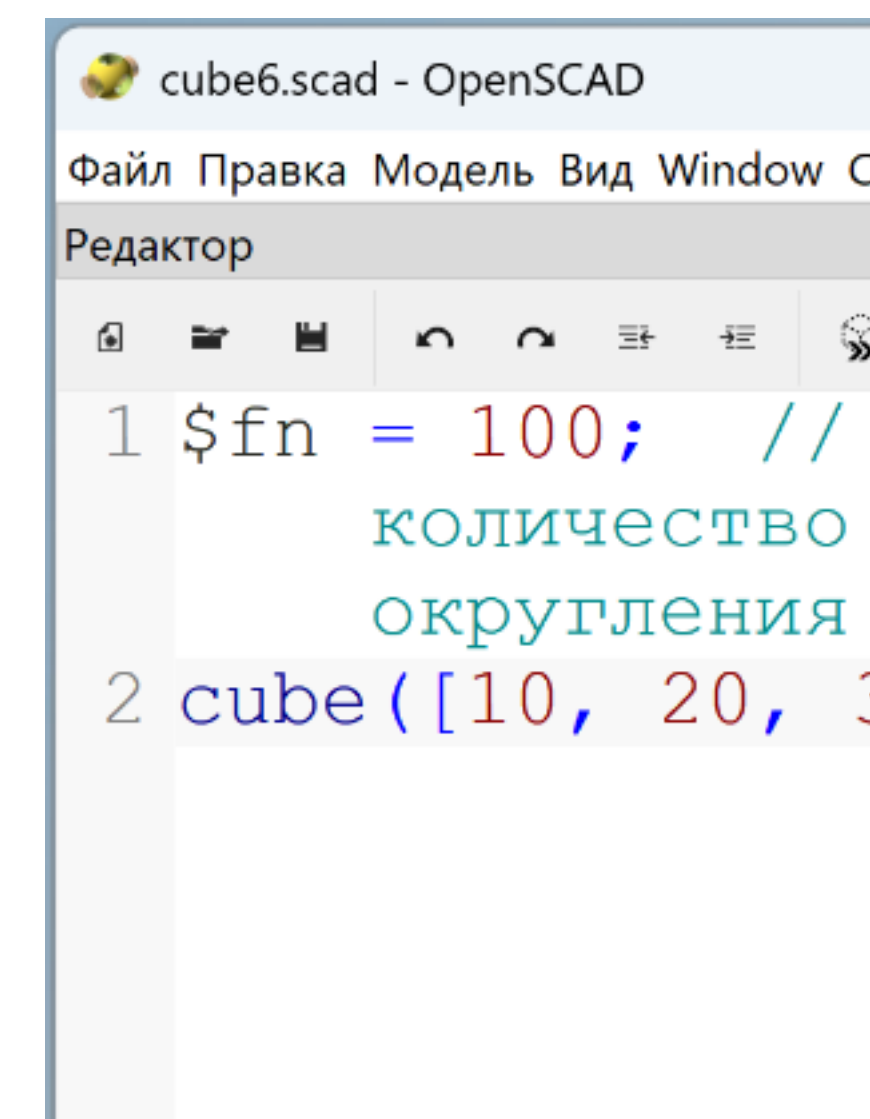

cube6.scad - OpenSCAD
Файл Правка Модель Вид Window C
Редактор
1 $fn = 100; //
количество
округления
2 cube([10, 20,

Рис. 5. Куб с округлыми углами

Примитив cube в PythonSCAD(OpenSCAD) позволяет создавать базовые геометрические формы, которые можно использовать в различных конструктивных операциях, таких как объединение, вычитание и пересечение.

2.1. Геометрический примитив "Куб" (cube) в PythonSCAD(Python)

Примитив cube в PythonSCAD(OpenSCAD) используется для создания прямоугольного параллелепипеда (или куба, если все его размеры равны). Он является одним из основных примитивов для построения 3D-объектов и часто используется в CSG-моделировании.

Синтаксис:

```
cu = cube(size)
```

size – размеры куба. Это может быть:

Число – если указано одно число, то создается куб с длиной каждой стороны, равной этому числу.

Массив из трех чисел – задает размеры по осям X, Y и Z (ширина, высота и глубина).

Пример: Куб с одной стороной

```
cu = cube(5)
show(cu) // Куб с длиной каждой стороны 5
```

Файл Правка Модель Вид Окно Справка

Редактор

Untitled.py*

```
1 from openscad import *
2 cu = cube(5)
3 show(cu)
```

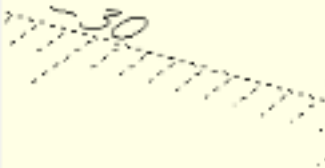


Рис. 6. Куб с одной стороной

Пример: Куб с разными размерами

```
cu = cube([10, 20, 30])
show(cu) // Прямоугольный параллелепипед размером 10x20x30
```

Untitled.py - OpenSCAD

Файл Правка Модель Вид Окно Сп

Редактор





Untitled.py*

```
1 from openscad import *
2 cu = cube([10, 20,
      30])
3 show(cu)
```

Рис. 7. Куб с разными размерами

Примитив cube в PythonSCAD(Python) позволяет создавать базовые геометрические формы, которые можно использовать в различных конструктивных операциях, таких как объединение, вычитание и пересечение.

Пример: Куб с центровкой

```
cu = cube([10, 10, 10], center=True)
show(cu) // Куб с размером 10, центрированный в начале координат
```

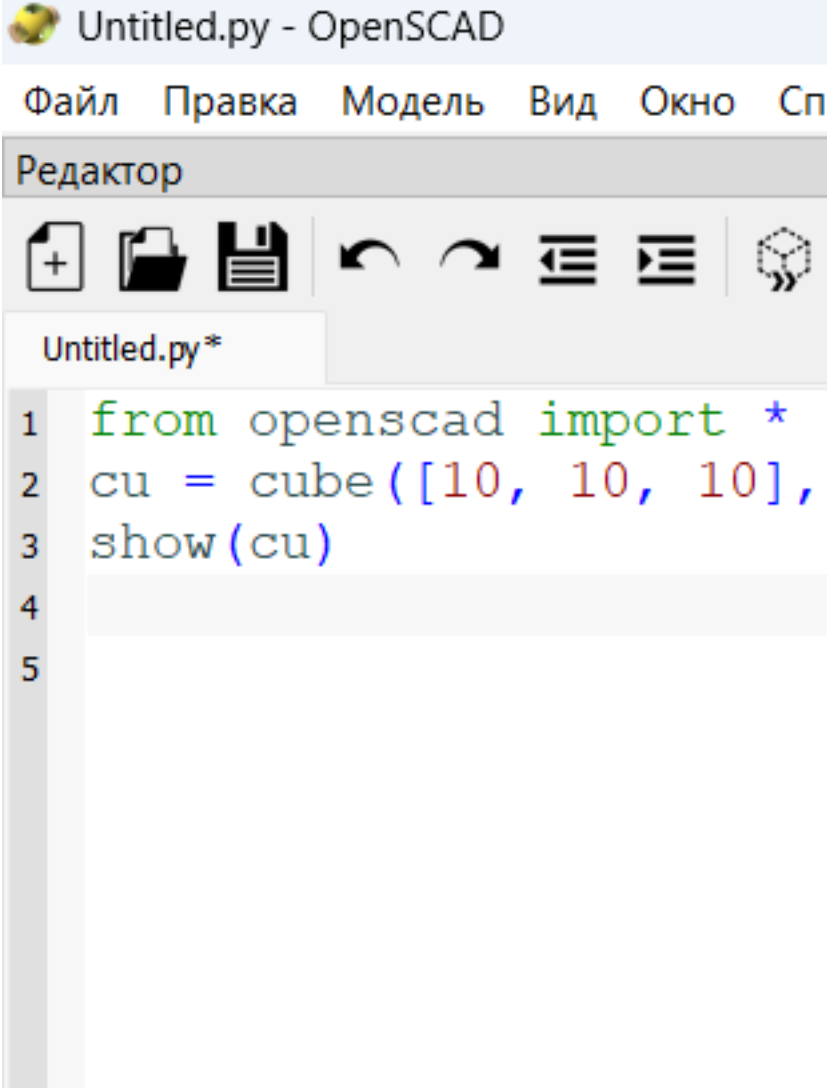

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cu = cube([10, 10, 10],
3 show(cu)
4
5

Рис.8. Куб с центровкой

Пример : Куб без центровки

```
cu = cube([10, 10, 10], center=False)
show(cu) // Куб с размером 10, одна из сторон будет в начале координат
```

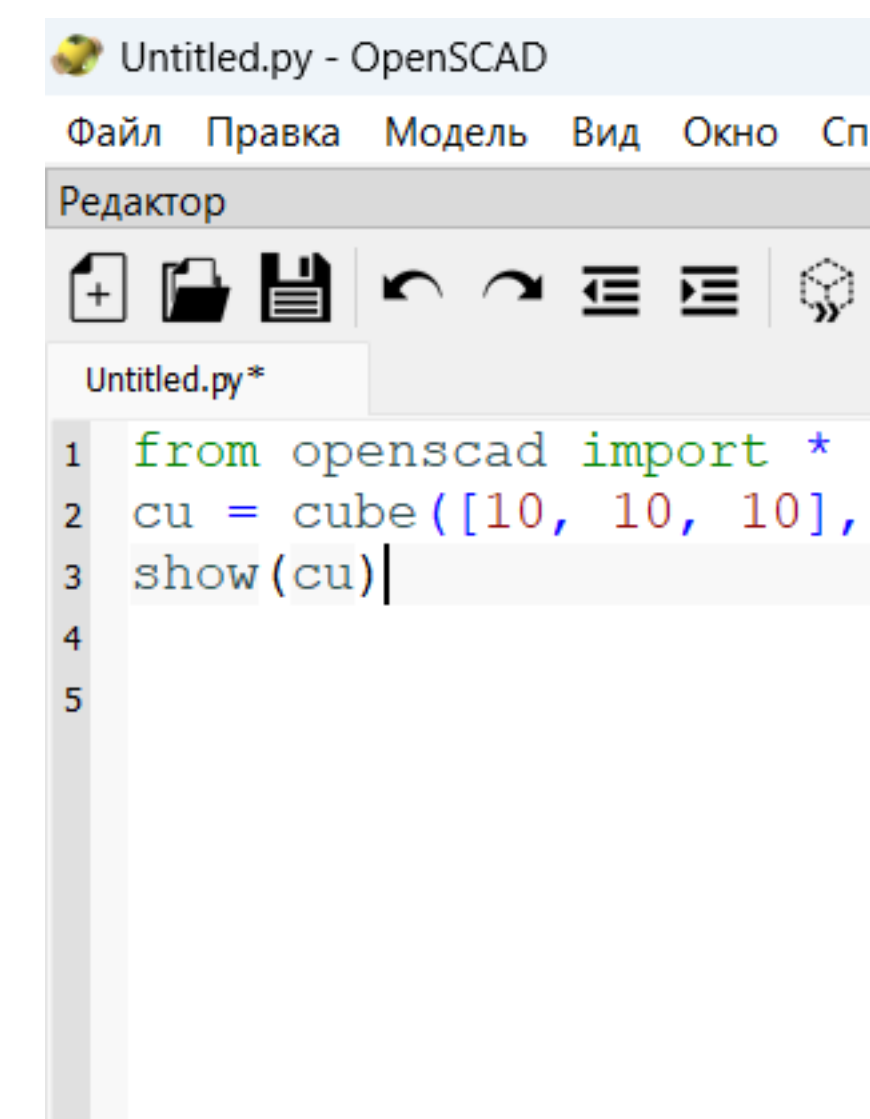

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cu = cube([10, 10, 10],
3 show(cu)
4
5

Рис. 9. Куб без центровки

Дополнительные параметры

Вы также можете использовать $fn для увеличения точности рендеринга, например, при использовании кубов с округлыми углами.

Пример: Куб с округлыми углами

```
c=cube(10);

mask=cube([30,1,30],center=True)

demo = [
c.fillet(2,fn=5)
]
show(demo)
```

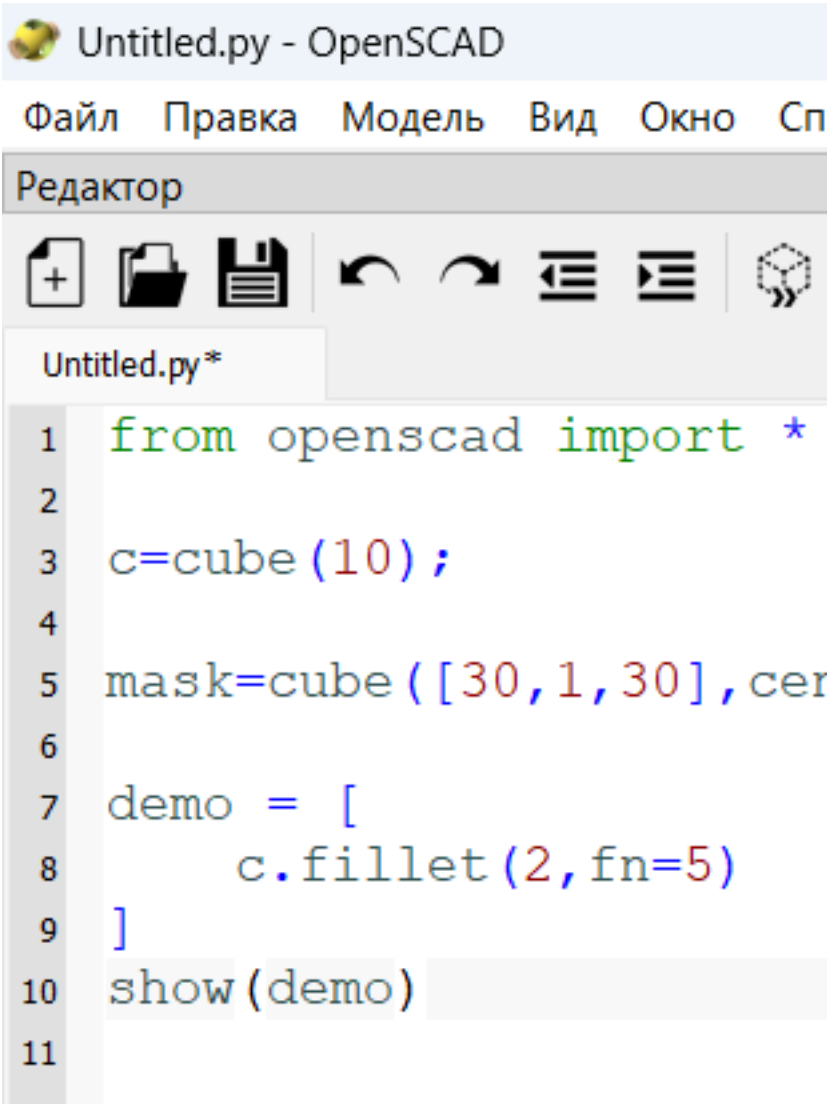

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2
3 c=cube(10);
4
5 mask=cube([30,1,30],cer
6
7 demo = [
8 c.fillet(2,fn=5)
9]
10 show(demo)
11

Рис. 10. Куб с округлыми углами

Примитив cube в PythonSCAD(Python) позволяет создавать базовые геометрические формы, которые можно использовать в различных конструктивных операциях, таких как объединение, вычитание и пересечение.

2.2. Геометрический примитив "Сфера" (sphere) в PythonSCAD(OpenSCAD)

Примитив sphere используется для создания 3D-сферы в PythonSCAD(OpenSCAD). Это один из основных примитивов, который может быть полезен при создании объектов с круглыми формами.

Синтаксис:

```
sphere(r, $fn=100);
```

r – радиус сферы. Это значение определяет размер сферы.

$fn – параметр, который управляет количеством грани, используемых для аппроксимации сферы. Чем выше значение, тем более гладкой будет сфера. Это опциональный параметр, по умолчанию установлен в 100.

Пример: Сфера с радиусом 10

```
sphere(10); // Сфера с радиусом 10
```

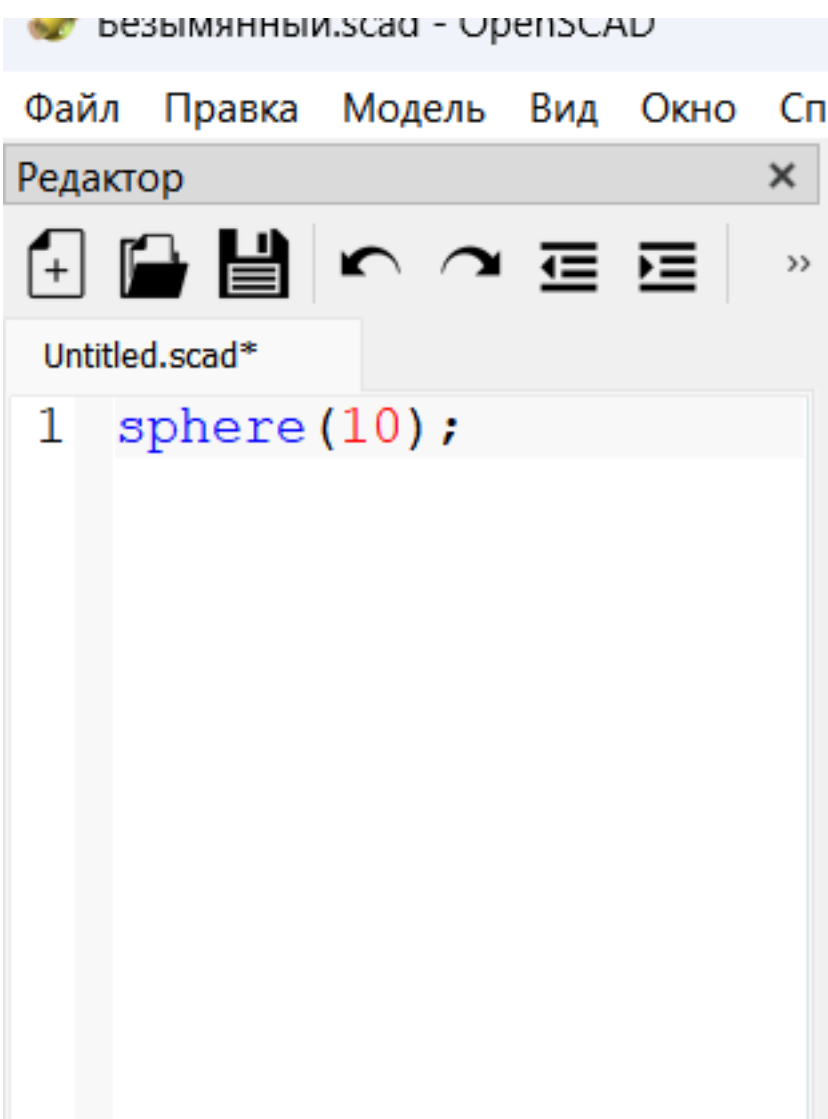

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 sphere(10);

Рис. 11. Сфера с радиусом 10

Пример: Сфера с настройкой качества

sphere(10, $fn=50); // Сфера с радиусом 10 и меньшим качеством (50 граней)


Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 sphere(10, $fn=50);

Рис. 12. Сфера с настройкой качества

В этом примере сфера будет менее гладкой, так как мы задали количество граней равным 50 вместо стандартных 100.

Пример: Сфера с высоким качеством

sphere(10, $fn=200); // Сфера с радиусом 10 и высоким качеством (200 граней)


Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 sphere(10, $fn=200);

Рис. 13. Сфера с высоким качеством

В этом случае сфера будет гораздо более гладкой, так как используется больше граней.

Пример: Полусфера

```
difference() {
sphere(10);
translate([-10, -10, 0]) cube([20, 20, 20]);
}
```

Безымянный.scad - OpenSCAD

Файл Правка Модель Вид Окно Сп

Редактор

Untitled.scad*

```
1 difference() {
2     sphere(10);
3     translate([-10
      , -10, 0]) cube
      ([20, 20, 20]);
4 }
```

Рис. 14. Полусфера

Этот пример вычитает куб из сферы, чтобы создать полусферу. Куб перекрывает нижнюю половину сферы, оставляя только верхнюю.

Пример: Сфера с центровкой

sphere(10, center=true); // Сфера с радиусом 10, центрированная в начале координат

Безымянный.scad - OpenSCAD

Файл Правка Модель Вид Окно Сп

Редактор

Untitled.scad*

```
1  sphere(10, center=
       true);  //
       Сфера с
       радиусом 10,
       центрированная
       в начале
       координат
```

Рис. 15. Сфера с центровкой

Если параметр center=true, то сфера будет центрирована в начале координат, а не располагаться так, чтобы её нижняя часть лежала на оси Z=0.

С помощью примитива sphere можно создавать разнообразные круглые объекты, такие как шарики, частицы и элементы дизайна для сложных моделей в PythonSCAD(OpenSCAD).

2.2. Геометрический примитив "Сфера" (sphere) в PythonSCAD(Python)

Синтаксис:

```
sp = sphere(r)
```

r – радиус сферы. Это значение определяет размер сферы.

Пример: Сфера с радиусом 10

```
sp = sphere(10)
show(sp)
```

Untitled.py - OpenSCAD

Файл Правка Модель Вид Окно Сп

Редактор

Untitled.py*

```
1 from openscad import *
2 sp = sphere(10)
3 show(sp)
```

Рис. 16. Сфера с радиусом 10

Пример: Сфера с настройкой качества

```
sp = sphere(10)
fil = sp.fillet(1)
show(fil)
```

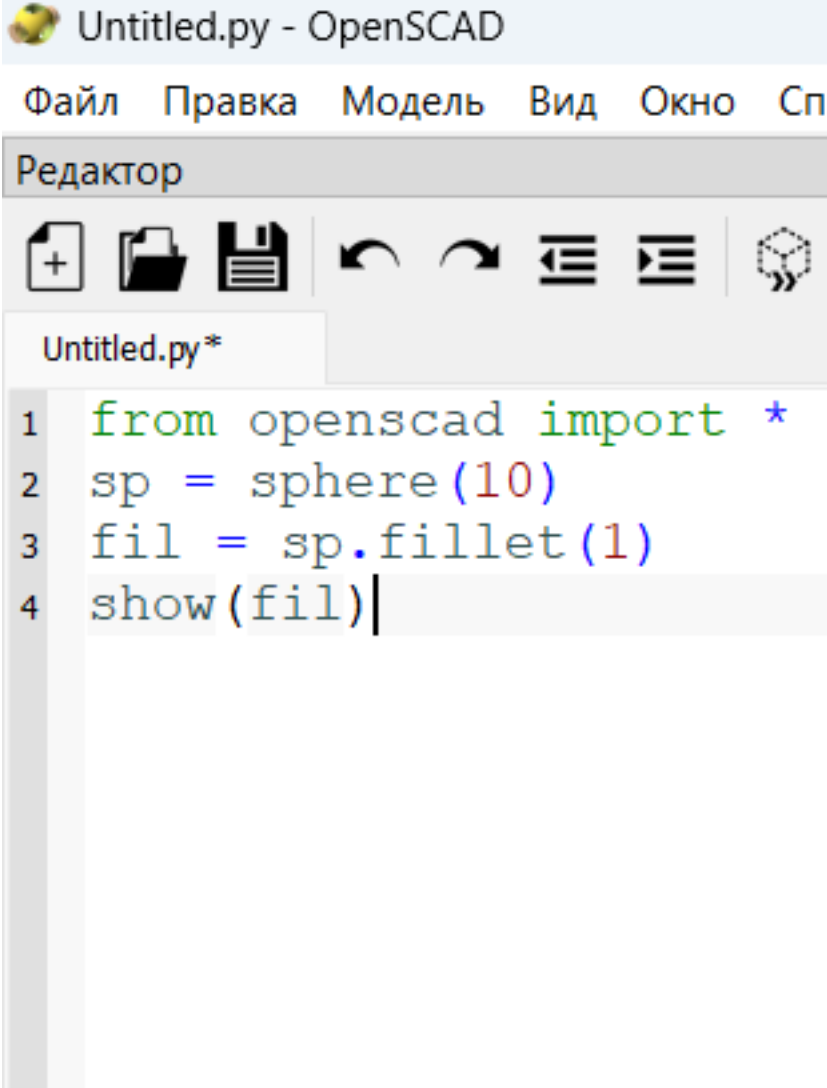

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 sp = sphere(10)
3 fil = sp.fillet(1)
4 show(fil)

Рис. 17. Сфера с настройкой качества

В этом примере сфера будет менее гладкой, так как мы задали количество граней равным 50 вместо стандартных 100.

Пример: Сфера с высоким качеством

```
sp = sphere(10)
fil = sp.fillet(100)
show(fil)
```

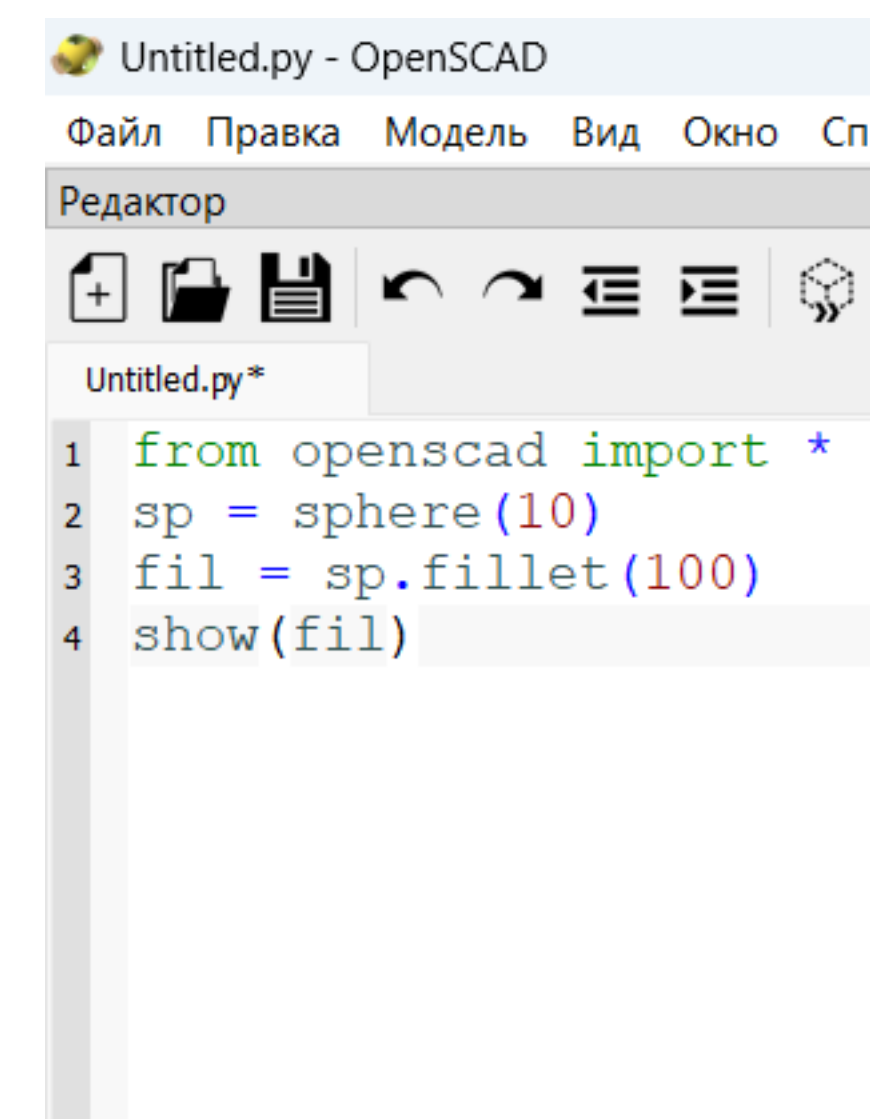

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 sp = sphere(10)
3 fil = sp.fillet(100)
4 show(fil)

Рис. 18. Сфера с высоким качеством

В этом случае сфера будет гораздо более гладкой, так как используется больше граней.

Пример: Полусфера

```
sp = sphere(10)
cu = cube(20)
cu = cu.translate([-10, -10, 0])
diff = sp.difference(cu)
cu = cu.translate([20, 20, 10])
show(diff)
```

Untitled.py - OpenSCAD

Файл Правка Модель Вид Окно Сп

Редактор






Untitled.py*

```
1 from openscad import *
2 sp = sphere(10)
3 cu = cube(20)
4 cu = cu.translate([-
      10, -10, 0])
5 diff = sp.difference
      (cu)
6 cu = cu.translate([
      20, 20, 10])
7 show(diff)
```

Рис. 19. Полусфера

С помощью примитива sphere можно создавать разнообразные круглые объекты, такие как шарики, частицы и элементы дизайна для сложных моделей в PythonSCAD(Python).

2.3. Геометрический примитив "Цилиндр" (cylinder) в PythonSCAD(OpenSCAD)

Примитив cylinder в PythonSCAD(OpenSCAD) используется для создания цилиндров и конусов. Этот примитив позволяет задавать высоту, радиусы основания и количество сегментов для сглаживания.

Синтаксис

cylinder(h, r, center);

cylinder(h, r1, r2, center);

h – высота цилиндра.

r – радиус цилиндра (если верхний и нижний радиус одинаковы).

r1 – радиус нижнего основания (используется для конусов).

r2 – радиус верхнего основания (используется для конусов).

center=true/false – если true, центрирует цилиндр относительно оси Z (по умолчанию false).

$fn – число граней, увеличивает сглаженность цилиндра.

Пример:Обычный цилиндр

```
cylinder(h=20, r=10);
```


Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 cylinder(h=20, r=10);

Рис. 20. Обычный цилиндр
Создает цилиндр высотой 20 и радиусом 10.
Основание находится в Z=0, верх цилиндра – в Z=20.
Пример:Центрированный цилиндр
cylinder(h=20, r=10, center=true);

Безымянный.scad - OpenSCAD

Файл Правка Модель Вид Окно Сп

Редактор





Untitled.scad*

```
1  cylinder(h=20, r=10
       , center=true);
```

Рис. 21. Центрированный цилиндр

Центрирован по оси Z, середина цилиндра располагается в Z=0 (нижняя граница в Z=-10, верхняя в Z=10).

Пример:Конус (Вершина сверху)

cylinder(h=30, r1=5, r2=15);

Безымянный.scad - OpenSCAD

Файл Правка Модель Вид Окно Сп

Редактор







Untitled.scad*

```
1  cylinder(h=30, r1=5
       , r2=15);
```

Рис. 22. Конус (цилиндр с разными радиусами)
Нижний радиус r1=5, верхний радиус r2=15, высота 30. Получается усеченный конус.

Пример:Конус (вершина внизу)

```
cylinder(h=30, r1=15, r2=5);
```

Безымянный.scad - OpenSCAD

Файл Правка Модель Вид Окно Сп

Редактор







Untitled.scad*

```
1  cylinder(h=30, r1=
       15, r2=5);
```

Рис. 23. Конус (вершина внизу)

Нижний радиус 15, верхний 5, получается перевернутый усеченный конус.

Пример:Сглаженный цилиндр ($fn)

```
cylinder(h=30, r=10, $fn=100);
```

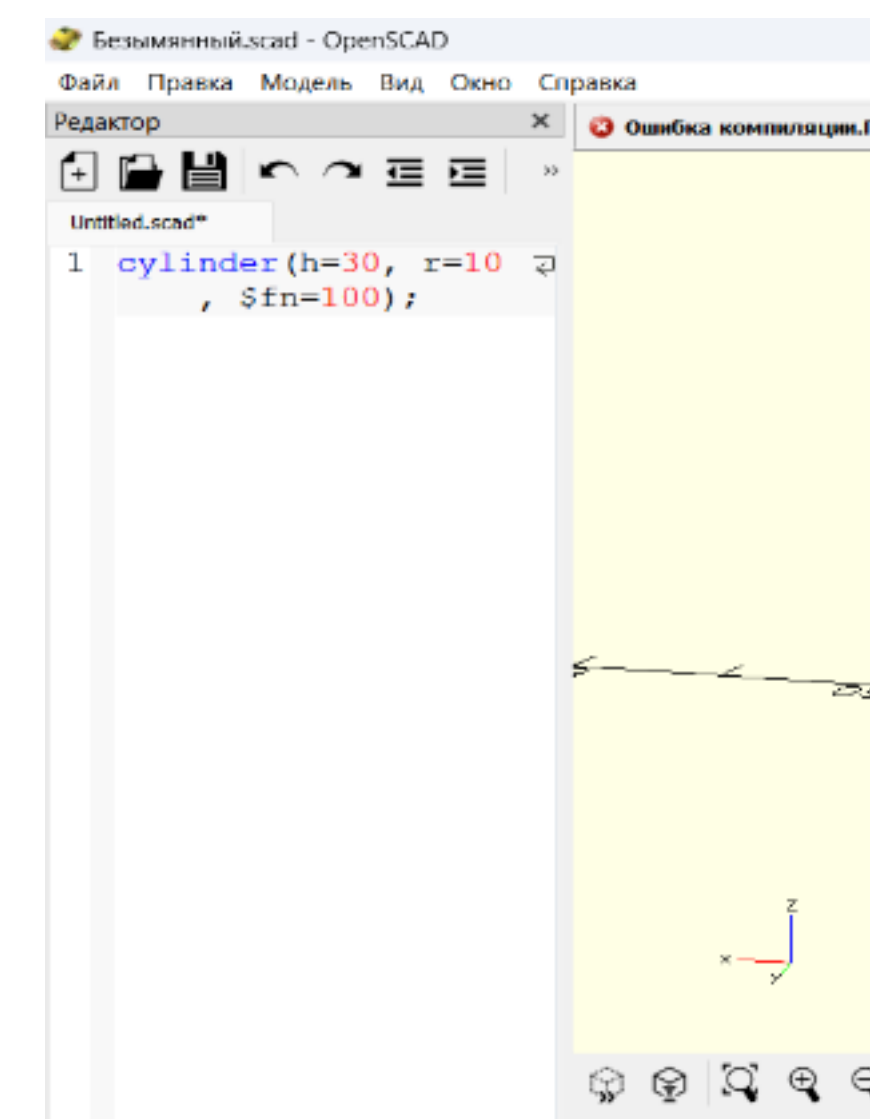

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.scad*
1 cylinder(h=30, r=10
, $fn=100);
Ошибка компиляции.
z
x
y

Рис. 24. Сглаженный цилиндр ($fn)

$fn=100 увеличивает количество граней, делая цилиндр более гладким.

Пример:Многогранный цилиндр (с низким $fn)

cylinder(h=30, r=10, $fn=6);

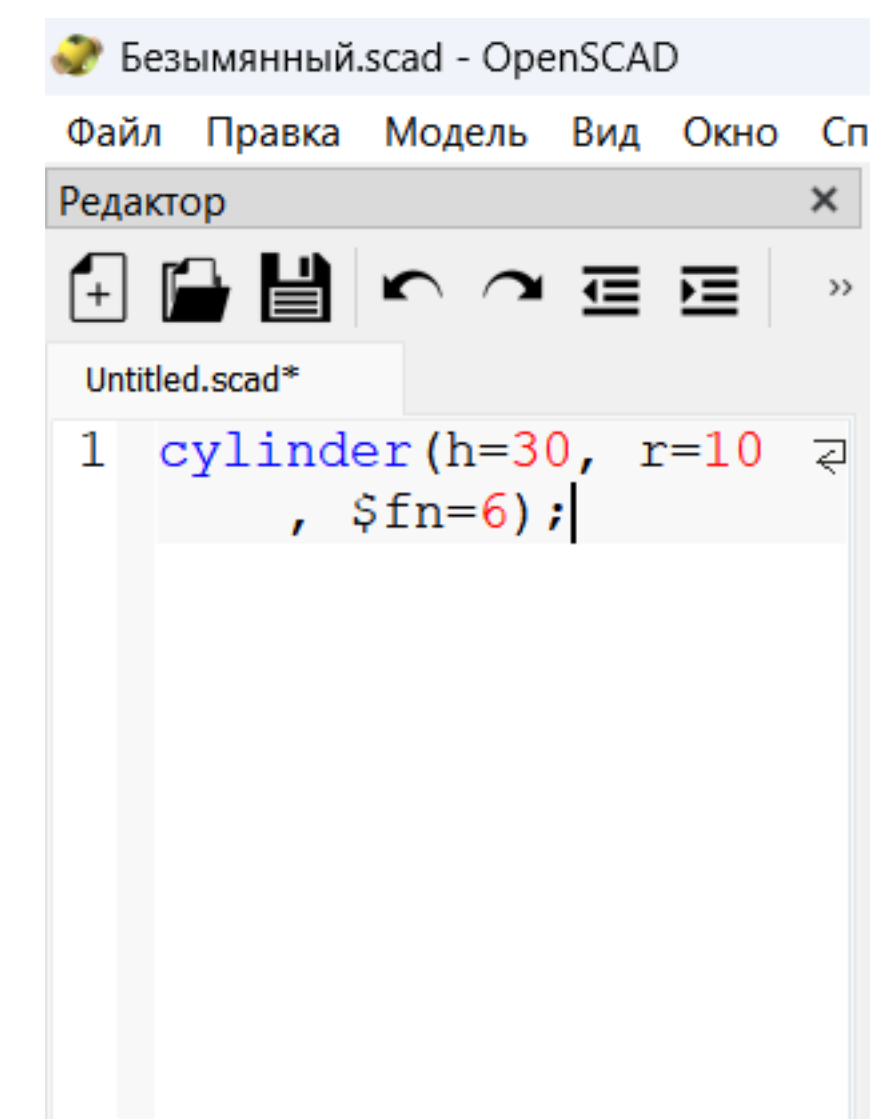

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 cylinder(h=30, r=10
, $fn=6);

Рис. 25. Многогранный цилиндр (с низким $fn)

Создает цилиндр с 6-угольным основанием (шестиугольная призма).

Заключение

Примитив cylinder в PythonSCAD(OpenSCAD) используется для создания цилиндров, конусов и многогранных объектов. Он гибок в настройках, что делает его полезным для моделирования деталей в 3D.

2.3. Геометрический примитив "Цилиндр" (cylinder) в PythonSCAD(Python)

Примитив cylinder в PythonSCAD(OpenSCAD) используется для создания цилиндров и конусов. Этот примитив позволяет задавать высоту, радиусы основания и количество сегментов для сглаживания.

Синтаксис

cy = cylinder(h, r1, r2)

h – высота цилиндра.

r – радиус цилиндра (если верхний и нижний радиус одинаковы).

r1 – радиус нижнего основания (используется для конусов).

r2 – радиус верхнего основания (используется для конусов).

Примеры использования

Пример:Обычный цилиндр

```
cy = cylinder(20, 10)
show(cy)
```

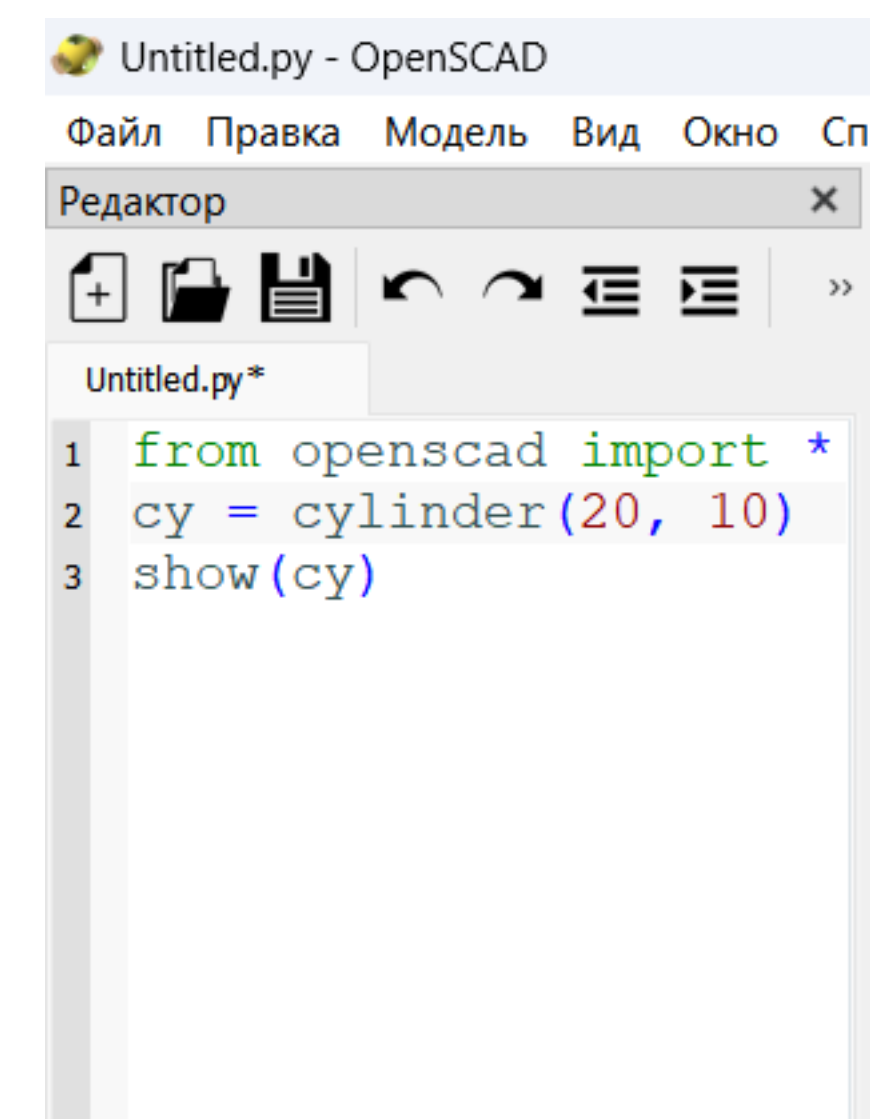

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cy = cylinder(20, 10)
3 show(cy)

Рис. 26. цилиндр

Пример:Конус (вершина сверху)

```
cy = cylinder(h=30, r1=5, r2=15)
show(cy)
```

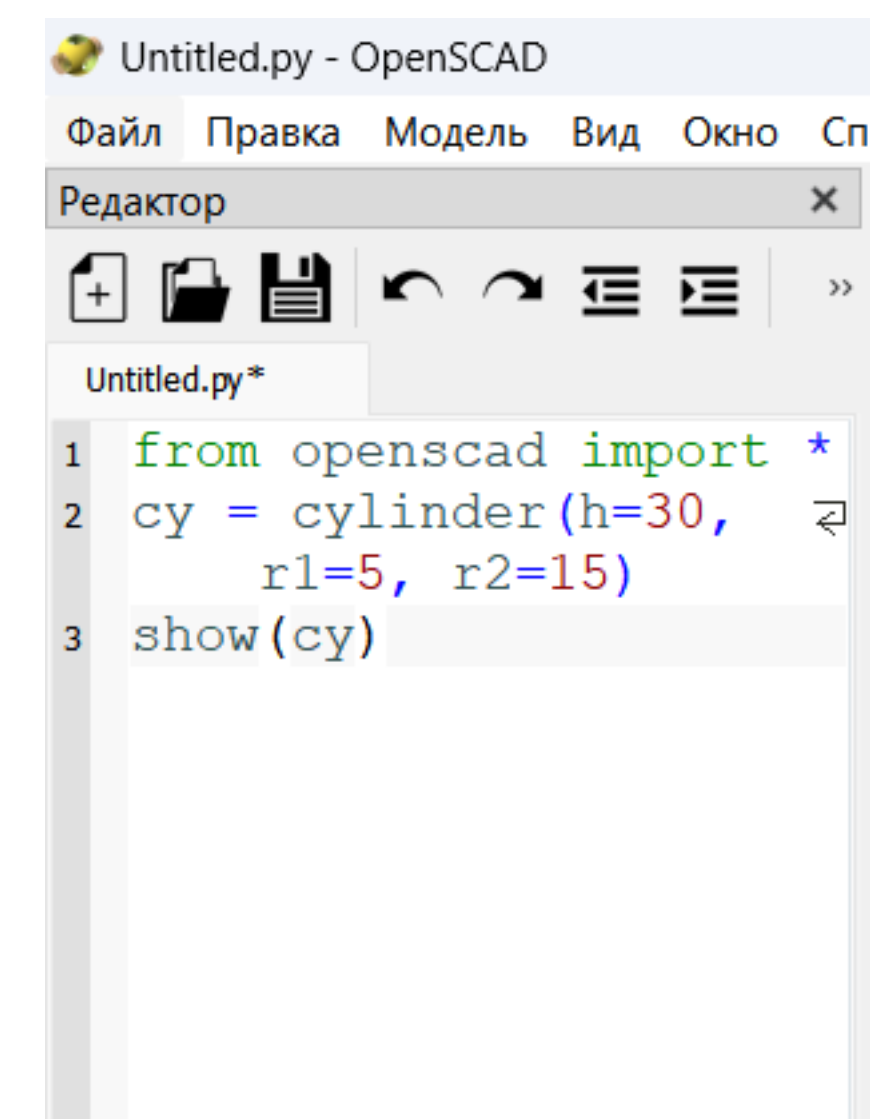

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cy = cylinder(h=30,
r1=5, r2=15)
3 show(cy)

Рис. 27. Конус (вершина сверху)

Пример:Конус (вершина внизу)

```
cy = cylinder(h=30, r1=15, r2=5)
show(cy)
```

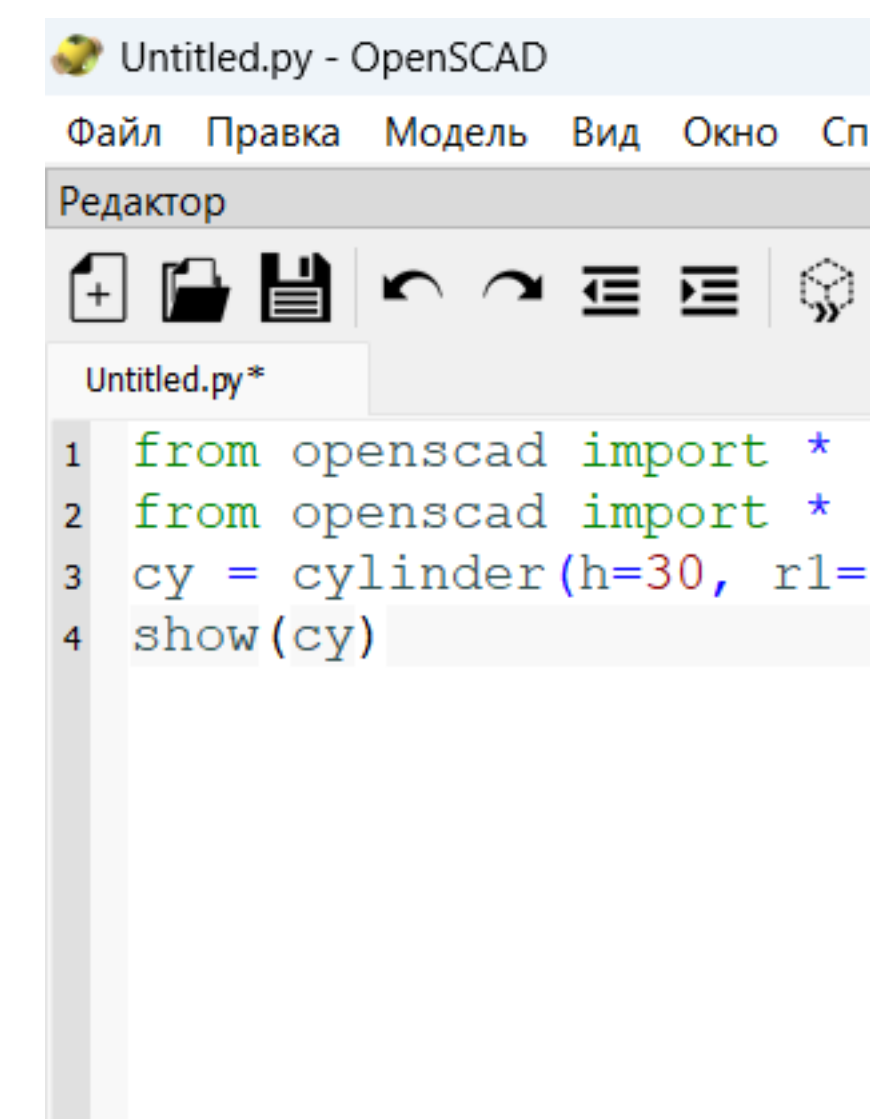

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 from openscad import *
3 cy = cylinder(h=30, r1=
4 show(cy)

Рис. 28. Конус (вершина внизу)

Заключение

Примитив cylinder в

PythonSCAD(Python)

используется для создания цилиндров, конусов и многогранных объектов. Он гибок в настройках, что делает его полезным для моделирования деталей в 3D.

2.4. Полигональные фигуры в OpenSCAD: *polygon* и *polyhedron*

В **OpenSCAD** есть два ключевых примитива для работы с многоугольниками и объемными полигонами:

polygon – для создания 2D-многоугольников.

polyhedron – для построения 3D-полигональных объектов.

polygon – 2D-многоугольники

Этот примитив позволяет создавать фигуры, задавая вершины и соединяя их по определенному порядку.

Синтаксис:

polygon(points, paths);

points – массив точек [x, y], которые определяют вершины.

paths (опционально) – порядок соединения точек (по умолчанию соединяются все по порядку).

Примеры использования

Пример: Треугольник

```
polygon(points=[[0,0], [10,0], [5,10]]);
```

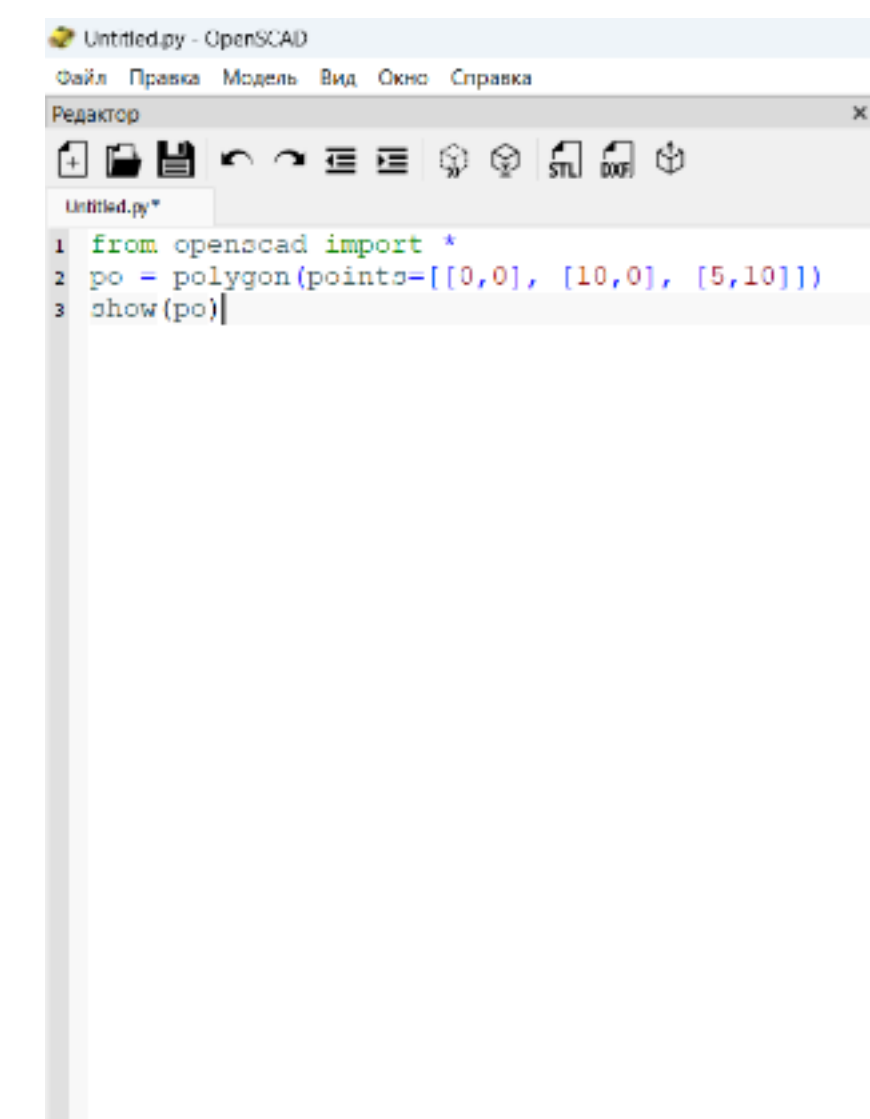

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
STL
DXF
Untitled.py*
1 from openscad import *
2 po = polygon(points=[[0,0], [10,0], [5,10]])
3 show(po)

Рис. 29. Треугольник

Определяет треугольник с вершинами (0,0), (10,0), (5,10).

Пример: Пятиугольник с заданными гранями

polygon(points=[[0,0], [10,0], [12,5], [5,10], [-2,5]], paths=[[0,1,2,3,4]]);

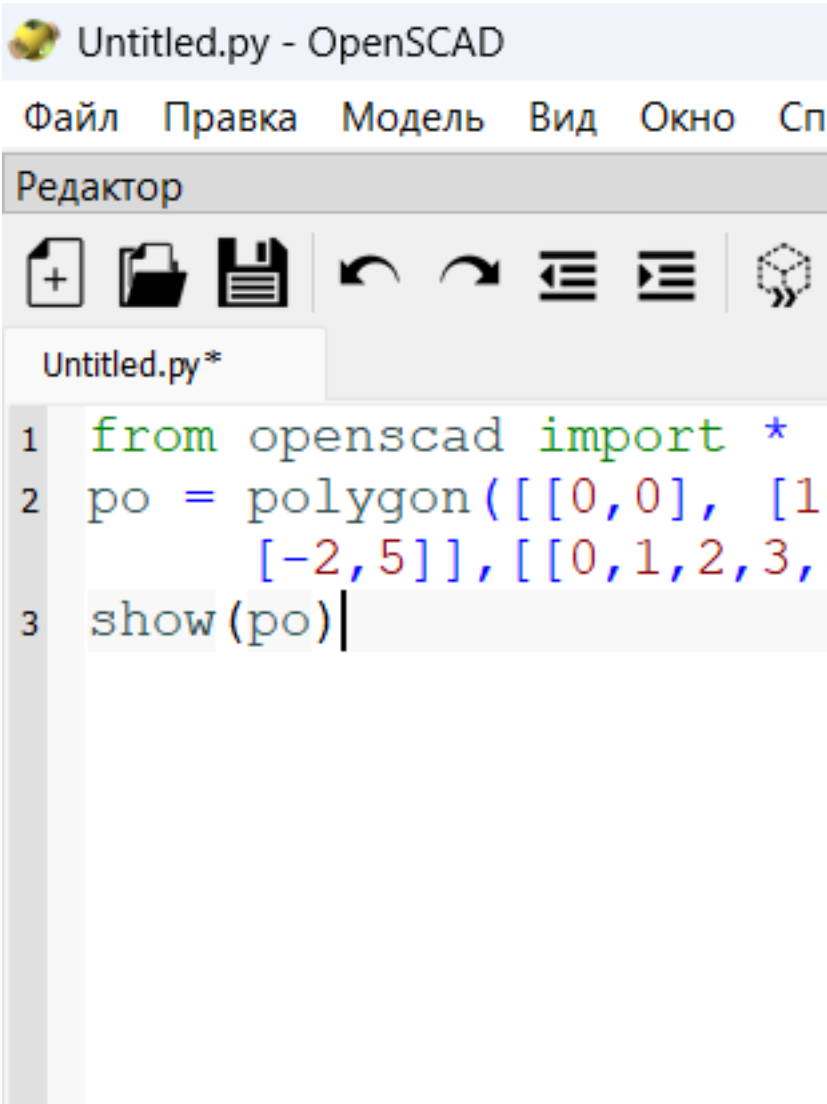

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 po = polygon([[0,0], [1
[-2,5]],[[0,1,2,3,
3 show(po)

Рис. 30. Пятиугольник с заданными гранями

Явно задает соединение точек для построения пятиугольника.

Пример: Вырез в фигуре (вложенные пути)

```
polygon(
points=[[0,0], [10,0], [10,10], [0,10], [3,3], [7,3], [7,7], [3,7]],
paths=[[0,1,2,3,0], [4,5,6,7,4]]
);
```

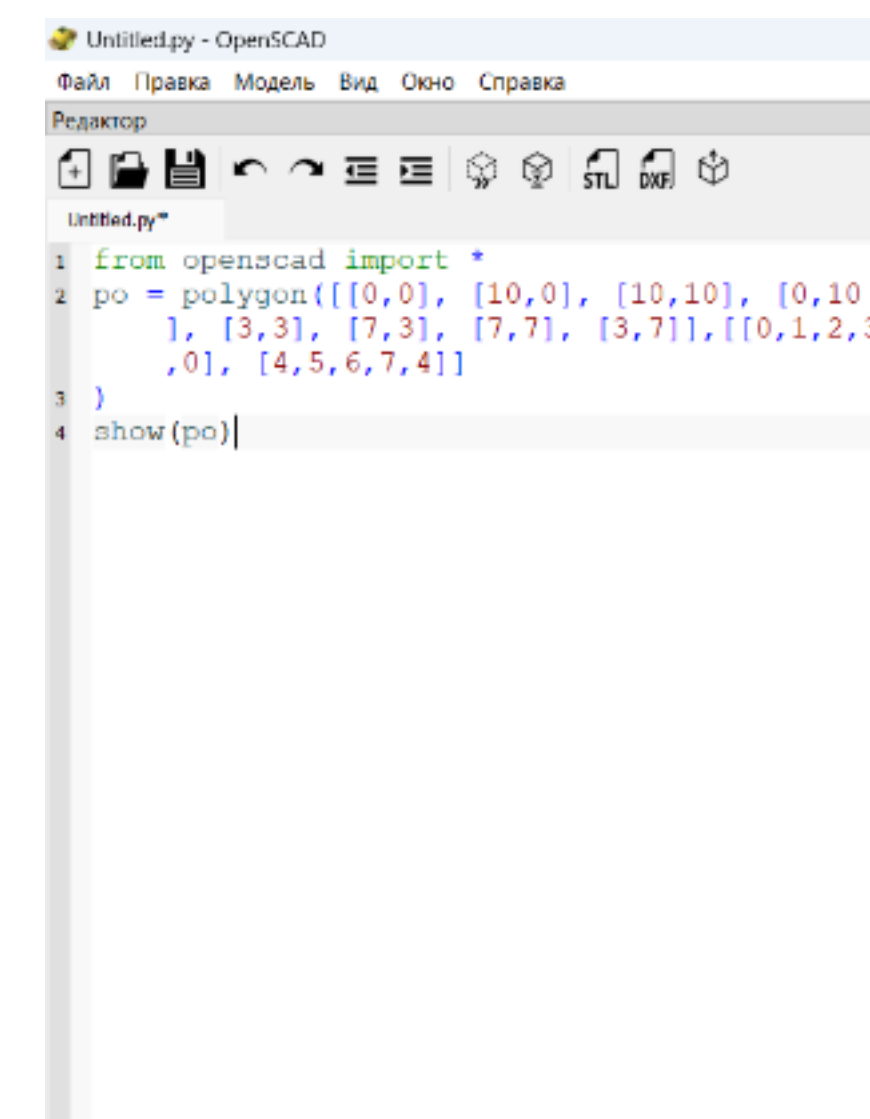

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
STL
DXF
Untitled.py*
1 from openscad import *
2 po = polygon([[0,0], [10,0], [10,10], [0,10
], [3,3], [7,3], [7,7], [3,7]],[[0,1,2,3
,0], [4,5,6,7,4]]
3)
4 show(po)

Рис. 31. Вырез в фигуре (вложенные пути)

Создает квадрат с вырезанным внутренним квадратом.

Экструзия 2D-фигур в 3D

Любую 2D-фигуру можно превратить в 3D с помощью linear_extrude:

linear_extrude(height=5) polygon(points=[[0,0], [10,0], [5,10]]);

Превращает треугольник в призму высотой 5.

2.

polyhedron

– 3D-полигональные объекты

Этот примитив позволяет создавать сложные объемные фигуры, соединяя грани на основе набора точек.

Синтаксис

:

polyhedron(points, faces, convexity);

points – массив координат вершин [x, y, z].

faces – массив граней, каждая грань указывается списком индексов точек.

convexity (опционально, по умолчанию 10) – используется для отображения сложных фигур (необязательный параметр).

Примеры использования

Пример: Тетраэдр (пирамида)

polyhedron(

points=[[0,0,0], [10,0,0], [5,10,0], [5,5,10]],

```
faces=[[0,1,2], [0,1,3], [1,2,3], [2,0,3]]
);
```

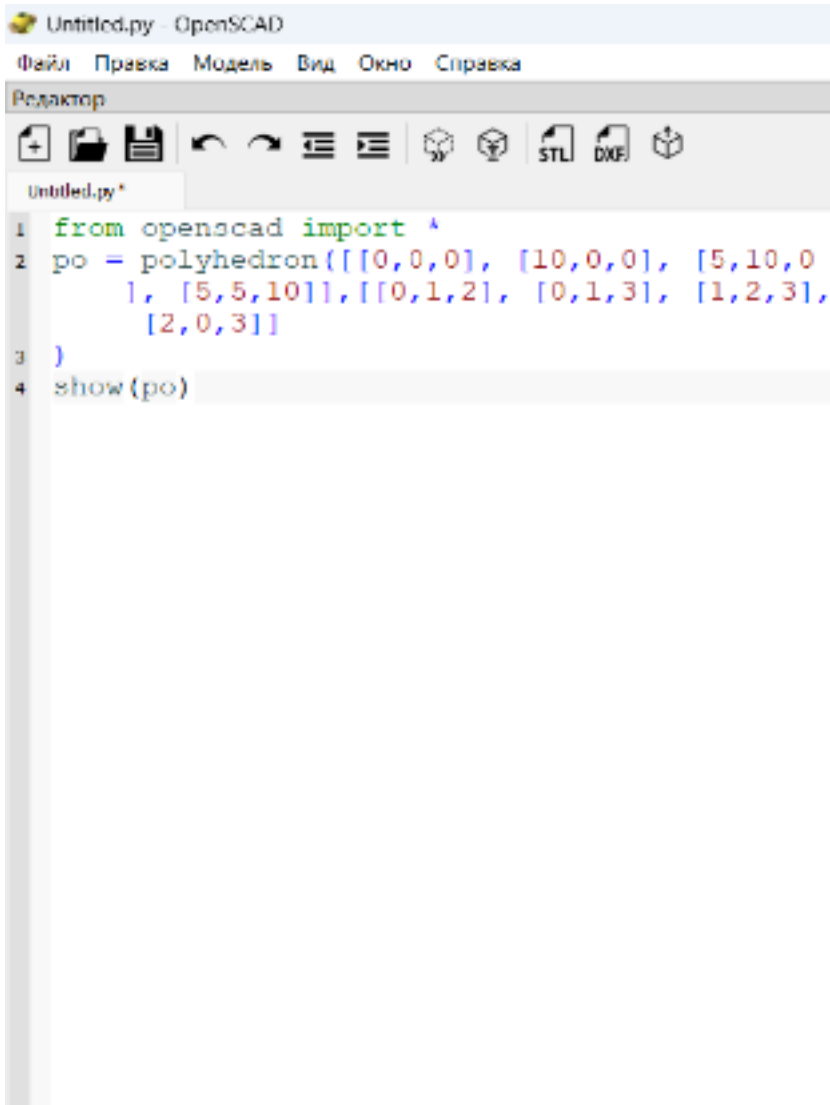

Untitled.py – OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
STL
DXF
Untitled.py*
1 from openscad import *
2 po = polyhedron([[0,0,0], [10,0,0], [5,10,0
], [5,5,10]],[[0,1,2], [0,1,3], [1,2,3],
[2,0,3]]
3)
4 show(po)

Рис. 32. Тетраэдр (пирамида)

Определяет пирамиду с четырьмя треугольными гранями.

Пример: Куб с polyhedron (альтернатива cube)

```
polyhedron(
points=[[0,0,0], [10,0,0], [10,10,0], [0,10,0], [0,0,10], [10,0,10], [10,10,10], [0,10,10]],
faces=[[0,1,2,3], [4,5,6,7], [0,1,5,4], [1,2,6,5], [2,3,7,6], [3,0,4,7]]
);
```

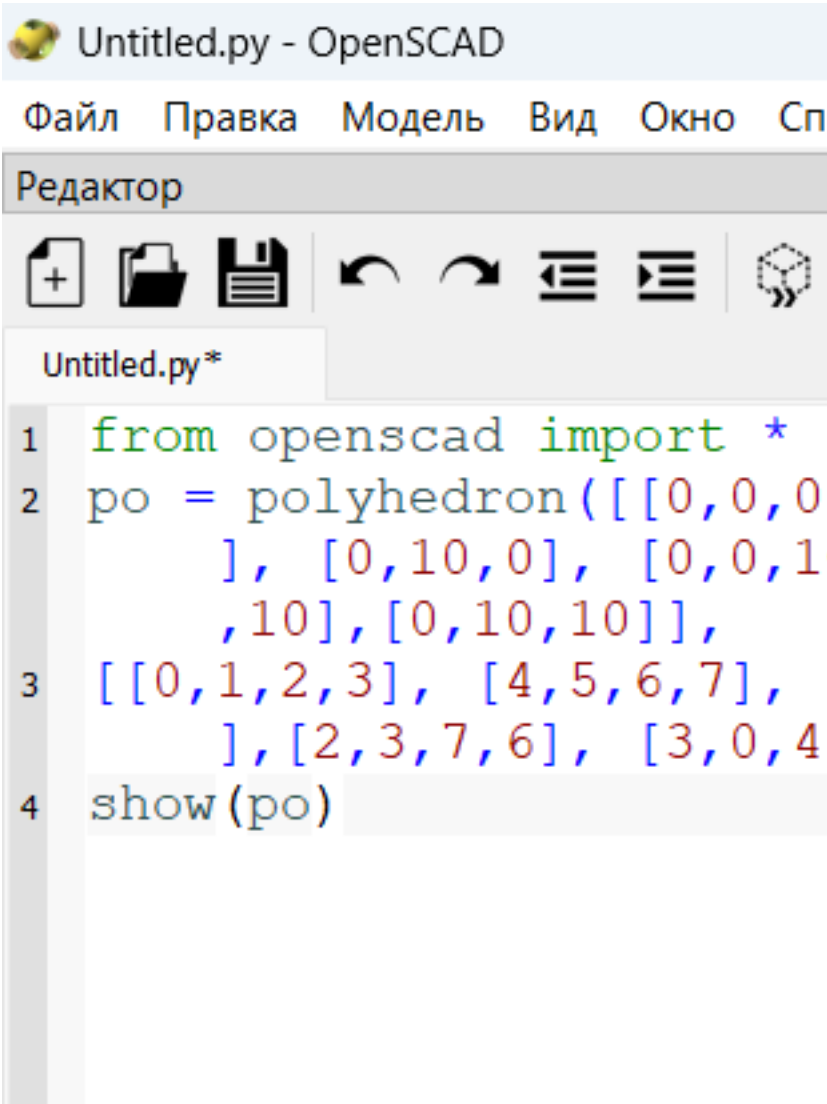

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 po = polyhedron([[0,0,0
], [0,10,0], [0,0,1
,10],[0,10,10]],
3 [[0,1,2,3], [4,5,6,7],
],[2,3,7,6], [3,0,4
4 show(po)

Рис. 33. Куб с polyhedron (альтернатива cube)

Определяет куб (как cube(10), но с явным определением граней).

Пример: Открытая пирамида

```
polyhedron(
points=[[0,0,0], [10,0,0], [10,10,0], [0,10,0], [5,5,10]],
faces=[[0,1,2,3], [0,1,4], [1,2,4], [2,3,4], [3,0,4]]);
```

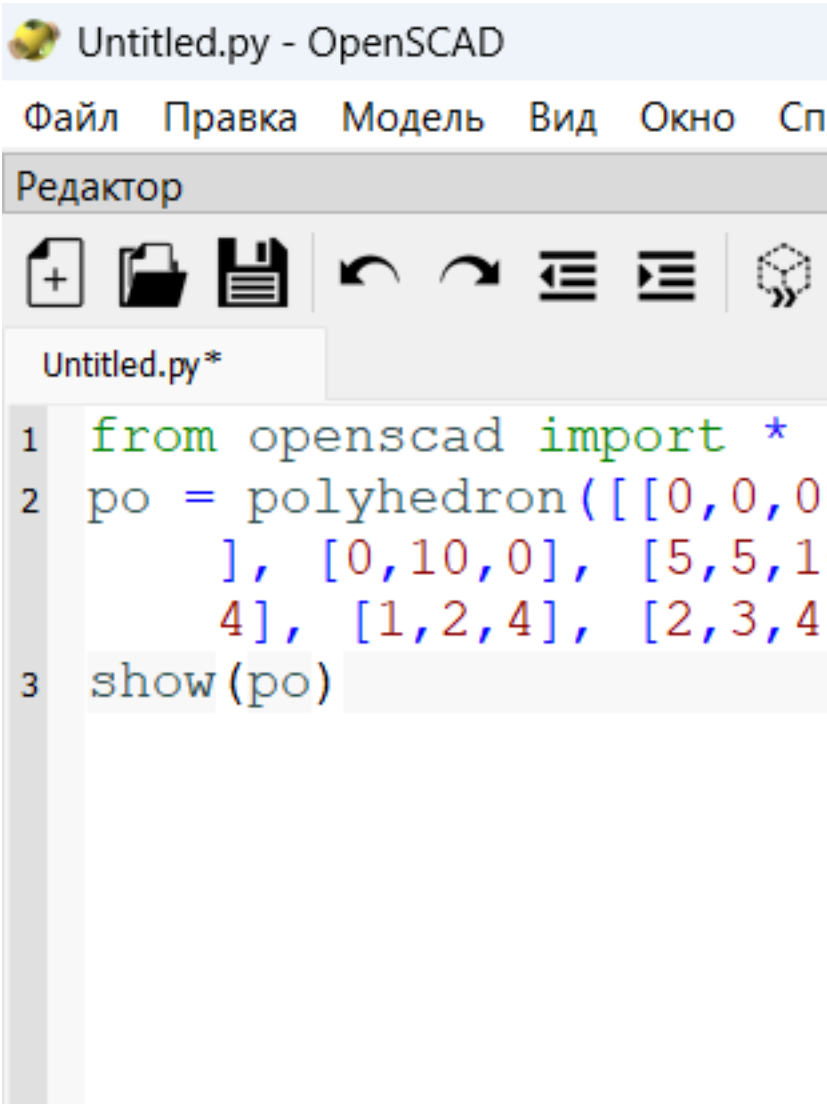

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 po = polyhedron([[0,0,0
], [0,10,0], [5,5,1
4], [1,2,4], [2,3,4
3 show(po)

Рис. 34. Открытая пирамида

Создает пирамиду с квадратным основанием.

Заключение

Примитивы polygon и polyhedron позволяют создавать любые сложные 2D и 3D фигуры. Они полезны, когда стандартные примитивы (cube, sphere, cylinder) недостаточны для моделирования нужных форм.

Полигональные фигуры в OpenSCAD:
polygon
и
polyhedron

В **OpenSCAD** есть два ключевых примитива для работы с многоугольниками и объемными полигонами:

polygon – для создания 2D-многоугольников.

polyhedron – для построения 3D-полигональных объектов.

1.
polygon
– 2D-многоугольники

Этот примитив позволяет создавать фигуры, задавая вершины и соединяя их по определенному порядку.

Синтаксис
:

po = polygon(points, paths);

points – массив точек [x, y], которые определяют верши-

ны.

paths (опционально) – порядок соединения точек (по умолчанию соединяются все по порядку).

Примеры использования

Пример: Треугольник

```
po = polygon(points=[[0,0], [10,0], [5,10]])
show(po)
```

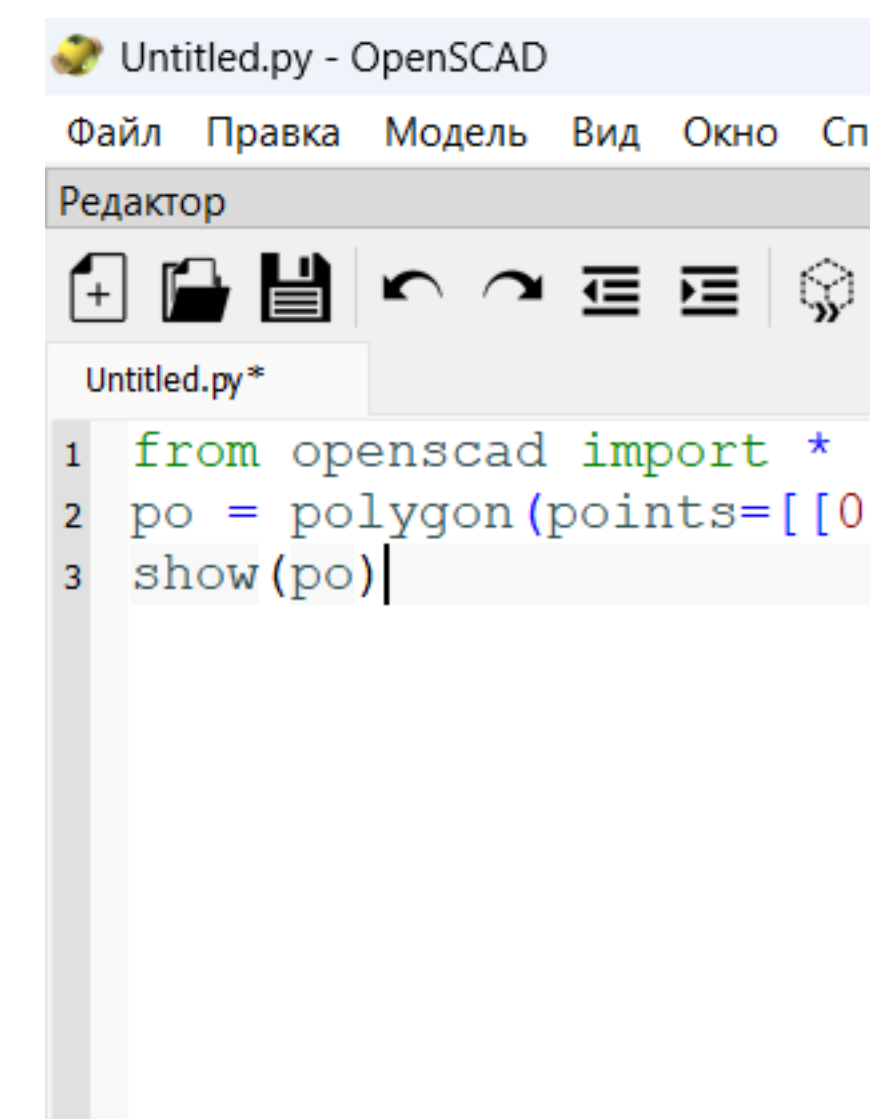

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 po = polygon(points=[[0
3 show(po)

Рис. 35. Треугольник

Определяет треугольник с вершинами (0,0), (10,0), (5,10).

Пример: Пятиугольник с заданными гранями

```
po = polygon([[0,0], [10,0], [12,5], [5,10], [-2,5]], [[0,1,2,3,4]])
show(po)
```

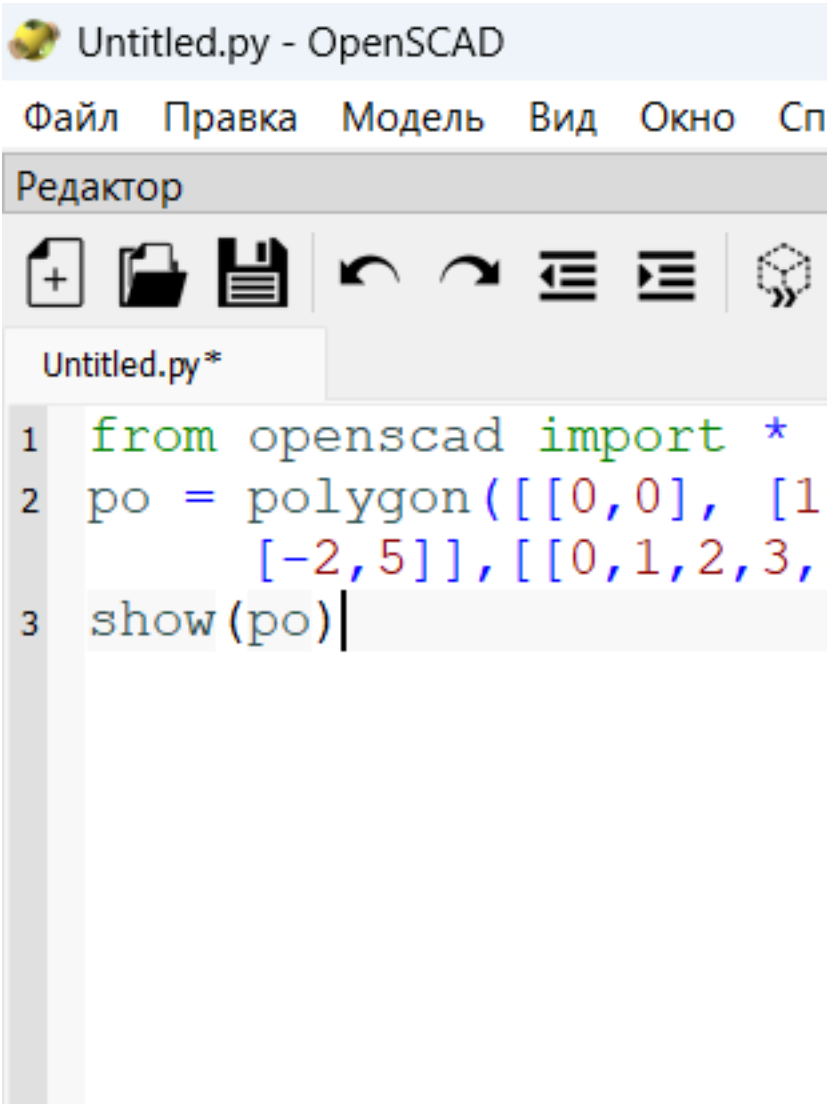

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 po = polygon([[0,0], [1
[-2,5]],[[0,1,2,3,
3 show(po)

Рис. 36. Пятиугольник с заданными гранями

Явно задает соединение точек для построения пятиугольника.

Пример: Вырез в фигуре (вложенные пути)

```
po = polygon([[0,0], [10,0], [10,10], [0,10], [3,3], [7,3], [7,7], [3,7]],[[0,1,2,3,0], [4,5,6,7,4]]
)
show(po)
```

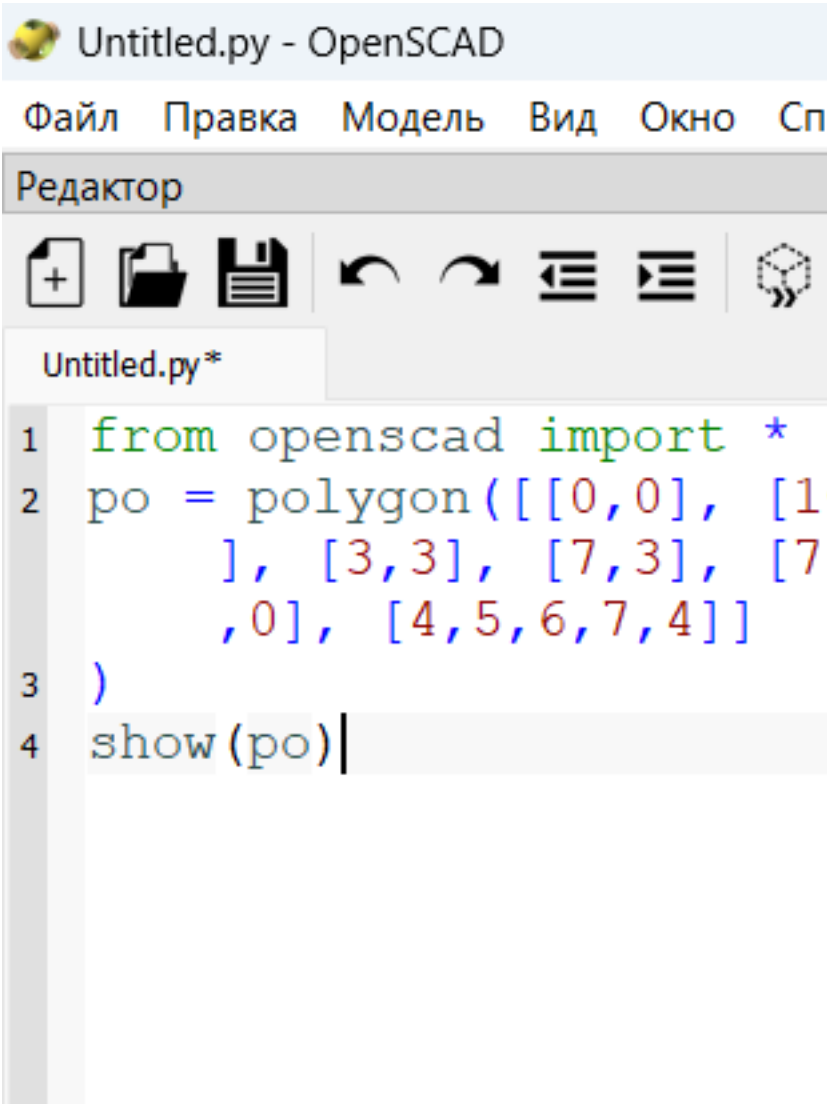

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 po = polygon([[0,0], [1
], [3,3], [7,3], [7
,0], [4,5,6,7,4]]
3)
4 show(po)

Рис. 37. Вырез в фигуре (вложенные пути)

Создает квадрат с вырезанным внутренним квадратом.

Экструзия 2D-фигур в 3D

Любую 2D-фигуру можно превратить в 3D с помощью linear_extrude:

```
po = polygon(points=[[0,0], [10,0], [5,10]])
l = po.linear_extrude(5)
show(l)
```

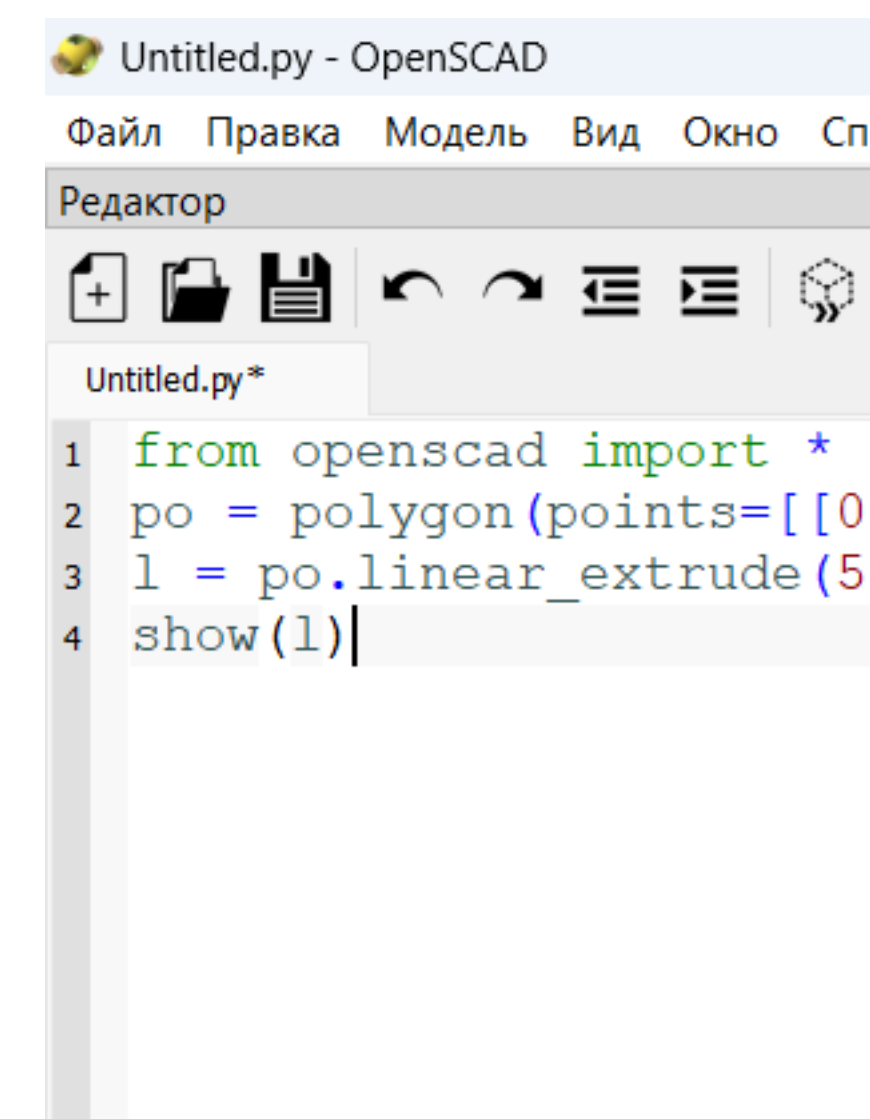

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 po = polygon(points=[[0
3 l = po.linear_extrude(5
4 show(l)

Рис. 38. Пример

Превращает треугольник в призму высотой 5.

2. polyhedron – 3D-полигональные объекты

Этот примитив позволяет создавать сложные объемные фигуры, соединяя грани на основе набора точек.

Синтаксис

:

polyhedron(points, faces, convexity);

points – массив координат вершин [x, y, z].

faces – массив граней, каждая грань указывается списком индексов точек.

convexity (опционально, по умолчанию 10) – используется для отображения сложных фигур (необязательный параметр).

Примеры использования

Пример: Тетраэдр (пирамида)

```
po = polyhedron([[0,0,0], [10,0,0], [5,10,0], [5,5,10]], [[0,1,2], [0,1,3], [1,2,3], [2,0,3]]
)
show(po)
```

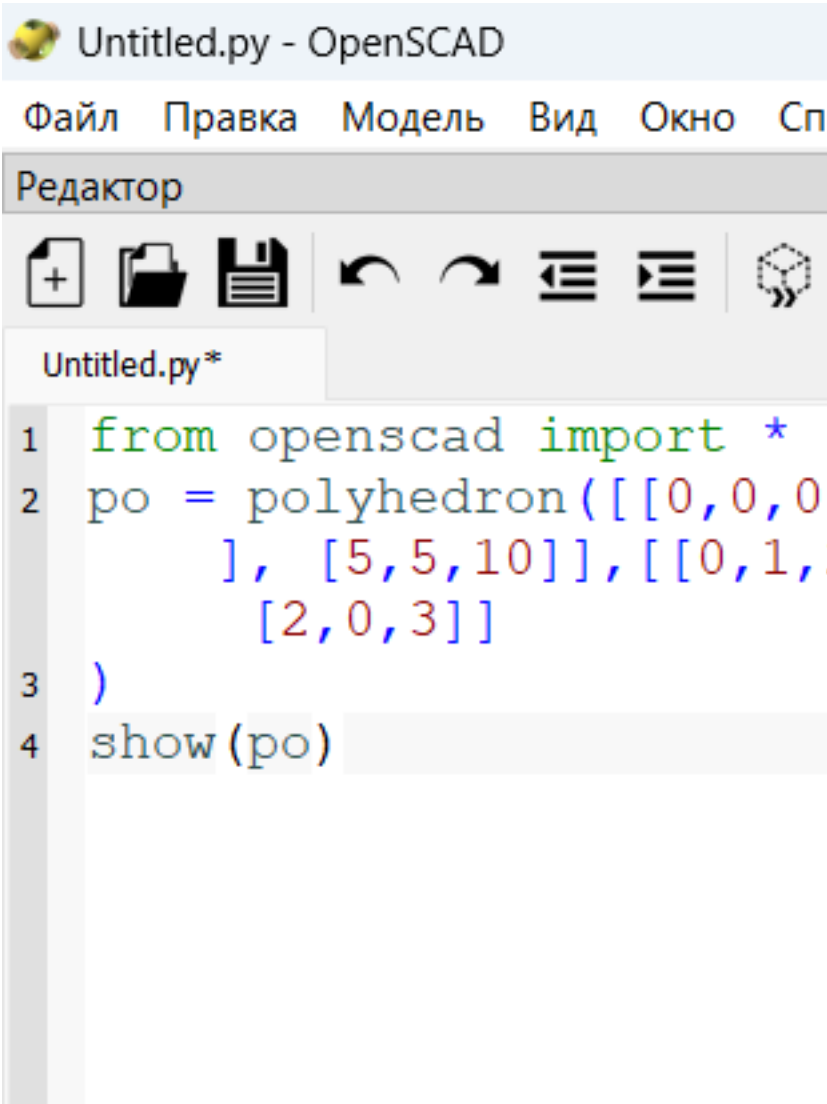

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 po = polyhedron([[0,0,0
], [5,5,10]],[[0,1,
[2,0,3]]
3)
4 show(po)

Рис. 39. Тетраэдр (пирамида)

Определяет пирамиду с четырьмя треугольными гранями.

Пример: Куб с polyhedron (альтернатива cube)

```
po = polyhedron([[0,0,0], [10,0,0], [10,10,0], [0,10,0], [0,0,10], [10,0,10], [10,10,10],[0,10,10]],
[[0,1,2,3], [4,5,6,7], [0,1,5,4], [1,2,6,5],[2,3,7,6], [3,0,4,7]])
show(po)
```

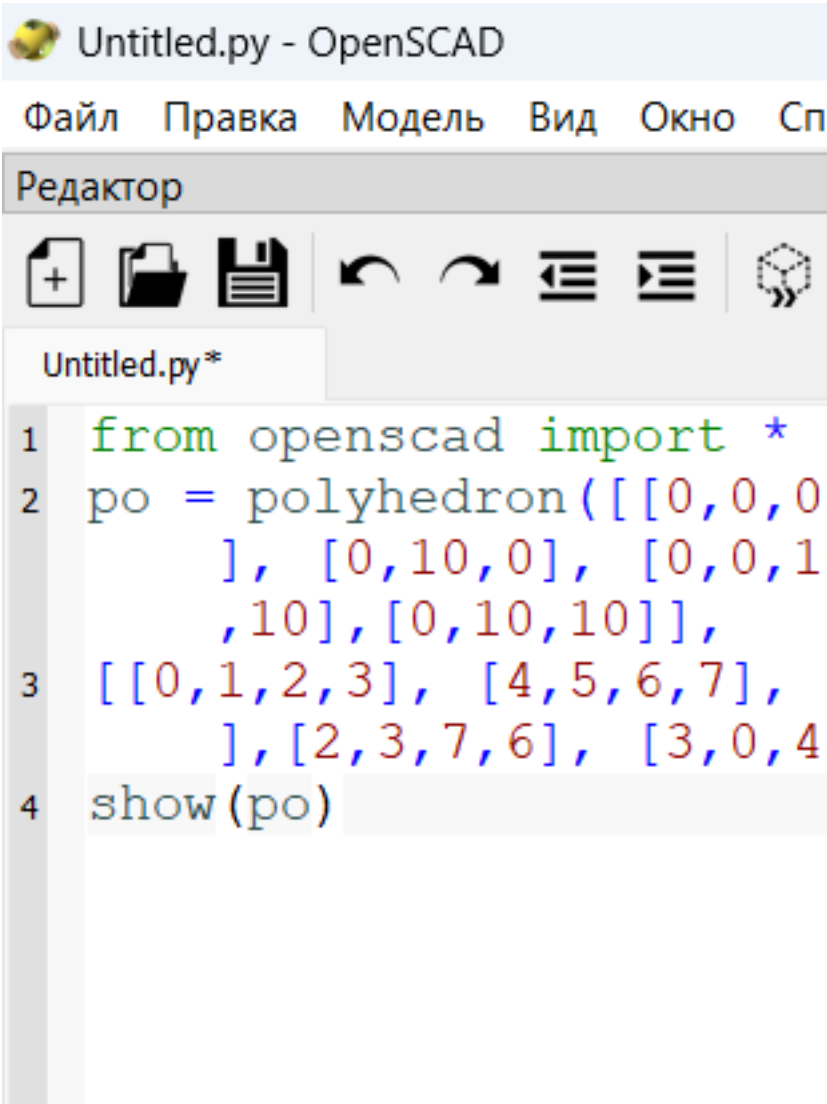

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 po = polyhedron([[0,0,0
], [0,10,0], [0,0,1
,10],[0,10,10]],
3 [[0,1,2,3], [4,5,6,7],
],[2,3,7,6], [3,0,4
4 show(po)

Рис. 40. Куб с polyhedron (альтернатива cube)

Определяет куб (как cube(10), но с явным определением граней).

Пример: Открытая пирамида

```
po = polyhedron([[0,0,0], [10,0,0], [10,10,0], [0,10,0], [5,5,10]],[[0,1,2,3], [0,1,4], [1,2,4], [2,3,4], [3,0,4]])
show(po)
```

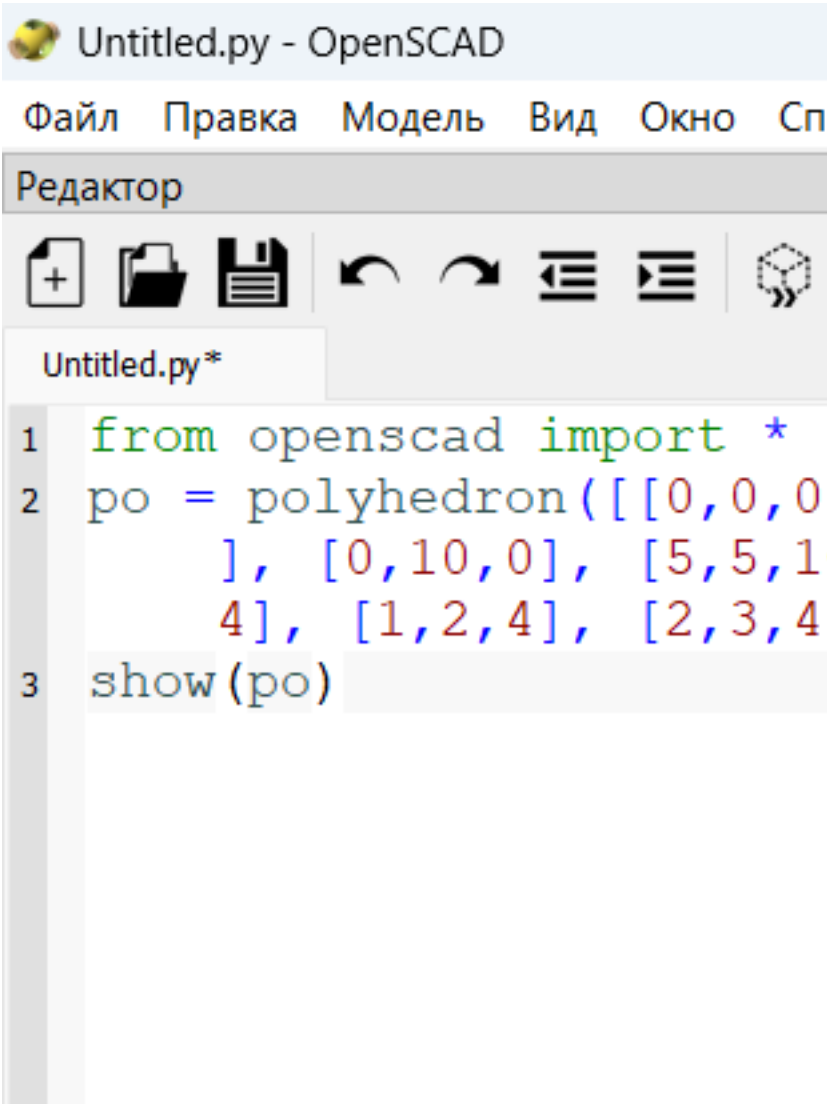

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 po = polyhedron([[0,0,0
], [0,10,0], [5,5,1
4], [1,2,4], [2,3,4
3 show(po)

Рис. 41. Открытая пирамида

Создает пирамиду с квадратным основанием.

Заключение

Примитивы polygon и polyhedron позволяют создавать любые сложные 2D и 3D фигуры. Они полезны, когда стандартные примитивы (cube, sphere, cylinder) недостаточны для моделирования нужных форм.

Задача 1: Создание Сферы

Какой оператор используется для создания сферы в PythonSCAD(OpenSCAD)?

1) sphere()

2) circle()

3) ellipse()

4) ball()

5) round()

Задача 2:Радиус Сферы

Какой параметр отвечает за радиус сферы?

1) r

2) d

3) size

4) radius

5) length

Задача 3: Создание куба

Какая команда создаёт куб в PythonSCAD(OpenSCAD)?

1) cube()

2) rectangle()
3) square()
4) block()
5) box()

Задача 4: Размеры Куба

Как задать размеры куба 10 x 20 x 30?
1) cube([10,20,30])
2) cube(10,20,30)
3) cube(size=[10,20,30])
4) cube({10,20,30})
5) cube<10,20,30>

Задача 5: Создание Цилиндра

Как создать цилиндр высотой 50 и радиусом 10?
1) cylinder(h=50, r=10)
2) cylinder(10,50)
3) cylinder(radius=10, height=50)
4) cylinder(h=10, r=50)
5) cylinder(d=10, h=50)

Задача 6: Создание Цилиндра

Какой параметр позволяет сделать цилиндр усечённым (разные радиусы сверху и снизу)?
1) r1, r2
2) d1, d2
3) scale
4) radius_top, radius_bottom
5) trunc

Задача 7: Создание Конуса
Какая команда создаёт конус?
1) cylinder(r1=10, r2=0, h=20)
2) pyramid()
3) cylinder(10,0,20)
4) trapezoid()
5) cone()
Задача 8: Создание 2D-прямогульника
Какой примитив создаёт 2D-прямоугольник?
1) square()
2) rectangle()
3) box2D()
4) shape()
5) plane()
Задача 9: Создание 2D-прямогульника
Как создать прямоугольник 15x25?
1) square([15,25])
2) square(15,25)
3) rect([15,25])
4) square({15,25})
5) square<15,25>
Задача 10: Создание многогранника
Как создать трёхмерный многогранник?
1) polyhedron(points, faces)
2) polyhedron()
3) polygon()

4) mesh()
5) shape3D()

Задача 11: Создание 2D-круга

Какой оператор создаёт 2D-круг?
1) circle()
2) sphere()
3) round()
4) ellipse()
5) disc()

Задача 12: Создание 2D-круга

Как задать диаметр у 2D-круга?
1) circle(d=20)
2) circle(radius=20)
3) circle(r=20)
4) circle(diameter=20)
5) circle(20)

Задача 13: Создание 2D-элипса

Какая команда создаёт 2D-эллипс?
1) scale([1,2]) circle(10)
2) ellipse_shape()
3) oval()
4) ellipse(10,20)
5) circle([10,20])

Задача 14: Вычитание фигур

Какой оператор позволяет вычитать одну фигуру из другой?

1) difference()
2) subtract()
3) union()
4) intersect()
5) add()

Задача 15: Пересечение фигур

Какой оператор создаёт пересечение двух фигур?
1) intersection()
2) merge()
3) blend()
4) unite()
5) overlap()

Задача 16: объединение фигур

Какой оператор объединяет несколько объектов?
1) union()
2) merge()
3) sum()
4) group()
5) combine()

Задача 1:Создание куба

Какая команда создаёт куб в PythonSCAD(OpenSCAD)?
1) cu = cube(size)
2) c = cube(x, y, z)
3) c = coil(size)
4) sp = sphere(radius)

Задача 2:Радиус Сферы

Какой параметр отвечает за радиус сферы?

1) (r)

2) (radius)

3) (size)

4) (tall)

Задача 3:Создание Сферы

Какой оператор используется для создания сферы в PythonSCAD(OpenSCAD)?

1) cu = cube(Size)

2) cu = sphere(r)

3) c = cylinder(h, r)

4) sp = sphere(r)

Задача 4: Размеры Куба

Как задать размеры куба 10 x 20 x 30?

1) cu = cube(10, 20, 30)

2) cube(10, 20, 30)

3) cu.cube(10, 20, 30)

4) cube = cu(10, 20, 30)

Задача 5: Создание Цилиндра

Как создать цилиндр высотой 50 и радиусом 10?

1) cy = cylinder(50, 10, 10)

2) cy = cylinder(50, 10)

3) cylinder(10, 50)

4) cy = cylinder(10, 50)

Задача 6: Создание Цилиндра

Какой параметр позволяет сделать цилиндр усечённым

(разные радиусы сверху и снизу)?

1) r1 r2

2) h

3) L1 L2

4) W

Задача 7: Создание Конуса

Какая команда создаёт конус?

1) cy = cylinder(r1=10, r2=0, h=20)

2) sp = sphere(r1=10, r2=0, h=20)

3) c = conus(r1=10, r2=0, h=20)

4) tr = trapezoid(r1=10, r2=0, h=20)

Задача 8: Создание 2D-прямогульника

Какой примитив создаёт 2D-прямоугольник?

1) sq = square()

2) b = box2D()

3) p = plane()

4) c = coil()

Задача 9: Создание 2D-прямогульника

Как создать прямоугольник 15x25?

1) sq = square([15,25])

2) square([15,25])

3) sq = ([15,25])

4) sq = square[15,25]

Задача 10: Создание многогранника

Как создать трёхмерный многогранник?

1) poly = polyhedron(point, path)

2) poly = polyhedron

3) poly = (point, path)

4) polyhedron(point, path)

Задача 11: Создание 2D-круга

Какой оператор создаёт 2D-круг?

1) c = circle()

2) c = circle

3) circle()

4) c = ()

Задача 12: Создание 2D-круга

Как задать диаметр у 2D-круга?

1) c = circle(20)

2) circle(20)

3) c = (20)

4) c = circle

Задача 13: Создание 2D-элипса

Какая команда создаёт 2D-эллипс?

1) c = circle(10) cc = c.scale([1,2])

2) 0 = oval()

3) cc = c.scale([1,2])

4) c = circle(10)

Задача 14: Вычитание фигур

Какой оператор позволяет вычитать одну фигуру из другой?

1) diff = difference()

2) fusion = union()

3) union()
4) difference()

Задача 15: Пересечение фигур

Какой оператор создаёт пересечение двух фигур?
1) inter = .intersection()
2) m = merge()
3) b = blend()
4) u = union()

Задача 16: объединение фигур

Какой оператор объединяет несколько объектов?
1) fusion = union()
2) diff = .difference()
3) inter = .intersection()
4) b = blend()

Модуль 3: Операции над объектами

Объединение (
union
) в
PythonSCAD
(
OpenSCAD
)

В **PythonSCAD(OpenSCAD)** оператор **union()** используется для объединения двух или более геометрических объектов в один. Это часть булевых операций, которые позволя-

ют комбинировать, вычитать и пересекать 3D-объекты.

Синтаксис

```
union() {
объект1;
объект2;
…
}
```

Все объекты внутри union() объединяются в единое целое.

Оператор union() можно не писать явно, так как OpenSCAD по умолчанию соединяет объекты, если они расположены внутри одного блока {}.

Примеры использования

Листинг 1:Простое объединение куба и цилиндра

```
union() {
cube([10, 10, 10]);
translate([5,5,10])
cylinder(10, 5);
}
```

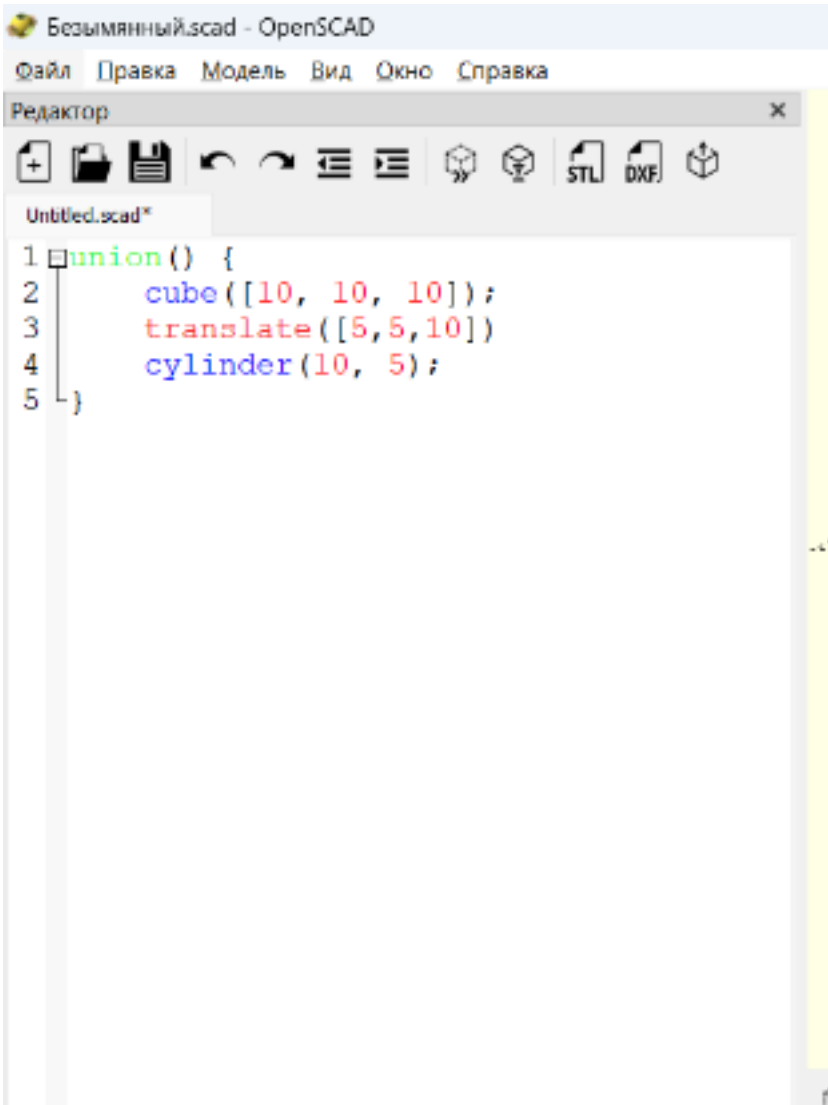

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
STL
DXF
Untitled.scad*
1 union() {
2 cube([10, 10, 10]);
3 translate([5,5,10])
4 cylinder(10, 5);
5 }

Рис.1. Простое объединение куба и цилиндра

Куб 10×10×10 и цилиндр высотой 10 и радиусом 5 объединяются в единое целое.

Цилиндр сдвинут в точку [5,5,0], чтобы частично пересекаться с кубом.

Листинг 2:Объединение без union() (неявное объединение)

```
cube([10,10,10]);
translate([5,5,10])
cylinder(10, 5);
```


Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 cube([10,10,10]);
2 translate([5,5,10])
3 cylinder(10, 5);

Рис.2. Объединение без union() (неявное объединение)

OpenSCAD по умолчанию объединяет пересекающиеся объекты, даже без union().

Листинг 3:Объединение нескольких объектов

```
union() {
cube([10, 10, 10]);
translate([5,5,20])
cylinder(10, 5);
translate([5,5,15])
sphere(5);
}
```

Безымянный.scad - OpenSCAD

Файл Правка Модель Вид Окно Сп

Редактор







Untitled.scad*

```
1 union() {
2     cube([10, 10, 10]);
3     translate([5,5,20])
4     cylinder(10, 5);
5     translate([5,5,15])
6     sphere(5);
7 }
```

Рис.3. Объединение нескольких объектов

Две сферы и куб объединяются в один сложный объект.

Листинг 4:Объединение с difference()

Можно комбинировать union() с difference() для создания сложных форм:

```
difference() {
union() {
cube([20,20,10]);
translate([5,5,0]) cylinder(h=10, r=10);
}
translate([10,10,-1]) cylinder(h=12, r=5);
}
```

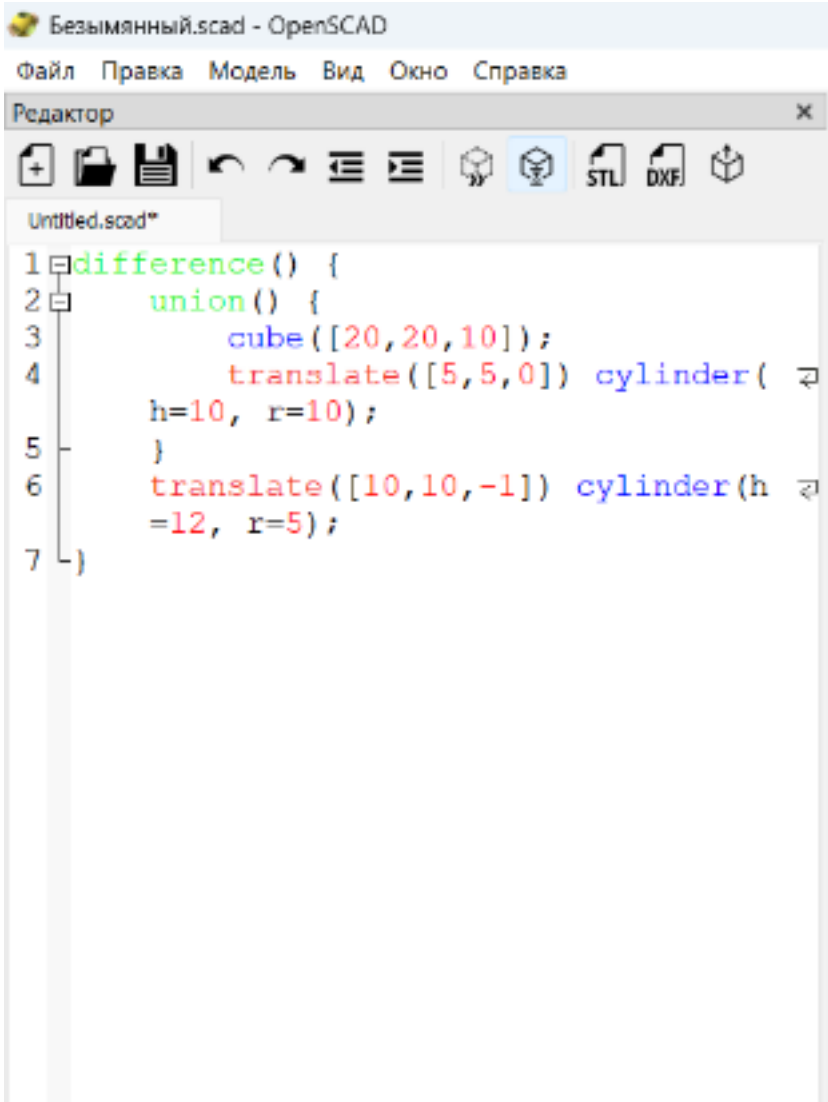

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
STL
DXF
Untitled.scad*
1 difference() {
2 union() {
3 cube([20,20,10]);
4 translate([5,5,0]) cylinder(
h=10, r=10);
5 }
6 translate([10,10,-1]) cylinder(h
=12, r=5);
7 }

Рис.4. Объединение с difference()

Объединяем куб и цилиндр.

Вычитаем из них второй цилиндр.

Заключение

union() помогает создавать сложные объекты путем объединения простых примитивов. Его можно не писать, если просто размещать объекты внутри {}. Работает в комбинации с difference() и intersection().

Объединение (union) в PythonSCAD(Python)

В PythonSCAD(Python) оператор union() используется для объединения двух или более геометрических объектов в один. Это часть булевых операций, которые позволяют комбинировать, вычитать и пересекать 3D-объекты.

Синтаксис

```
from openscad import *

x = обьект1

y = обьект2

fusion = union([x,y])

show(fusion)
```

Все объекты внутри union() объединяются в единое целое.

Оператор union() можно не писать явно, так как

OpenSCAD по умолчанию соединяет объекты, если они расположены внутри одного блока {}.

Примеры использования

Листинг 1:Простое объединение куба и цилиндра

```
from openscad import *

cu = cube(10)

cy = cylinder(10, 5)

fusion = union([cu,cy])

show(fusion)
```

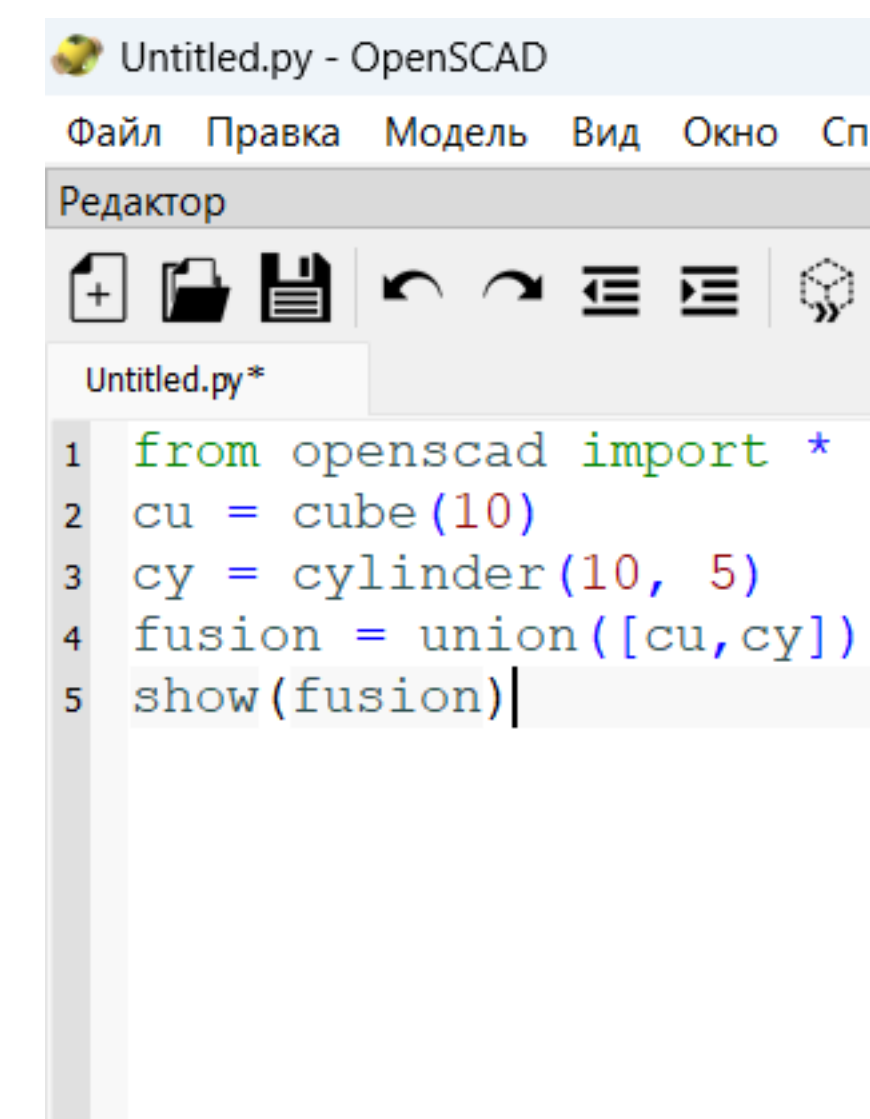

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cu = cube(10)
3 cy = cylinder(10, 5)
4 fusion = union([cu,cy])
5 show(fusion)

Рис. 5. Простое объединение куба и цилиндра

Куб 10×10×10 и цилиндр высотой 10 и радиусом 5 объединяются в единое целое.

Листинг 2:Объединение без union() (неявное объединение)

```
from openscad import *

cu = cube(10)

cy = cylinder(12, 5)

cy = cy.translate([5, 5, 0])

show([cu, cy])
```

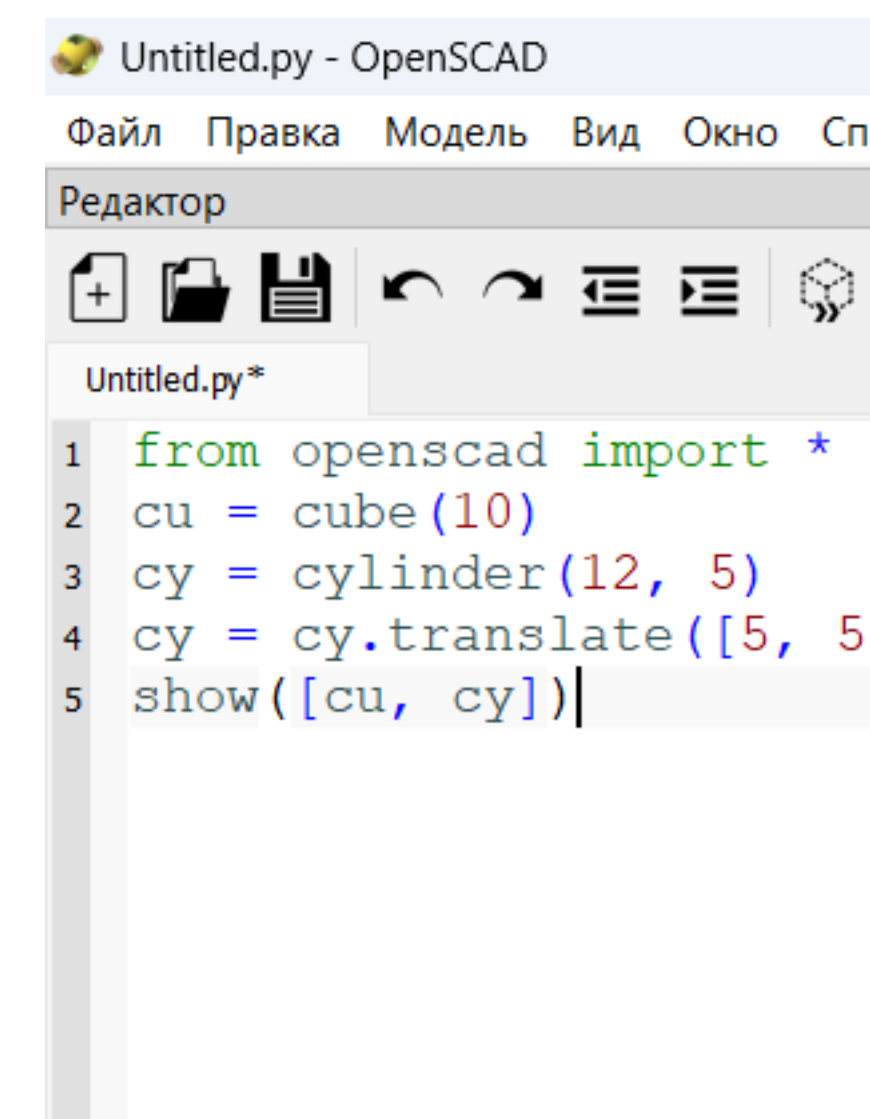

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cu = cube(10)
3 cy = cylinder(12, 5)
4 cy = cy.translate([5, 5
5 show([cu, cy])

Рис. 6. Объединение без union() (неявное объединение)

OpenSCAD по умолчанию объединяет пересекающиеся объекты, даже без union().

Листинг 3:Объединение нескольких объектов

```
from openscad import *

sp = sphere(5)

cu = cube(10)

cy = cylinder(20)

fusion = union([cu,cy,sp])

show(fusion)
```

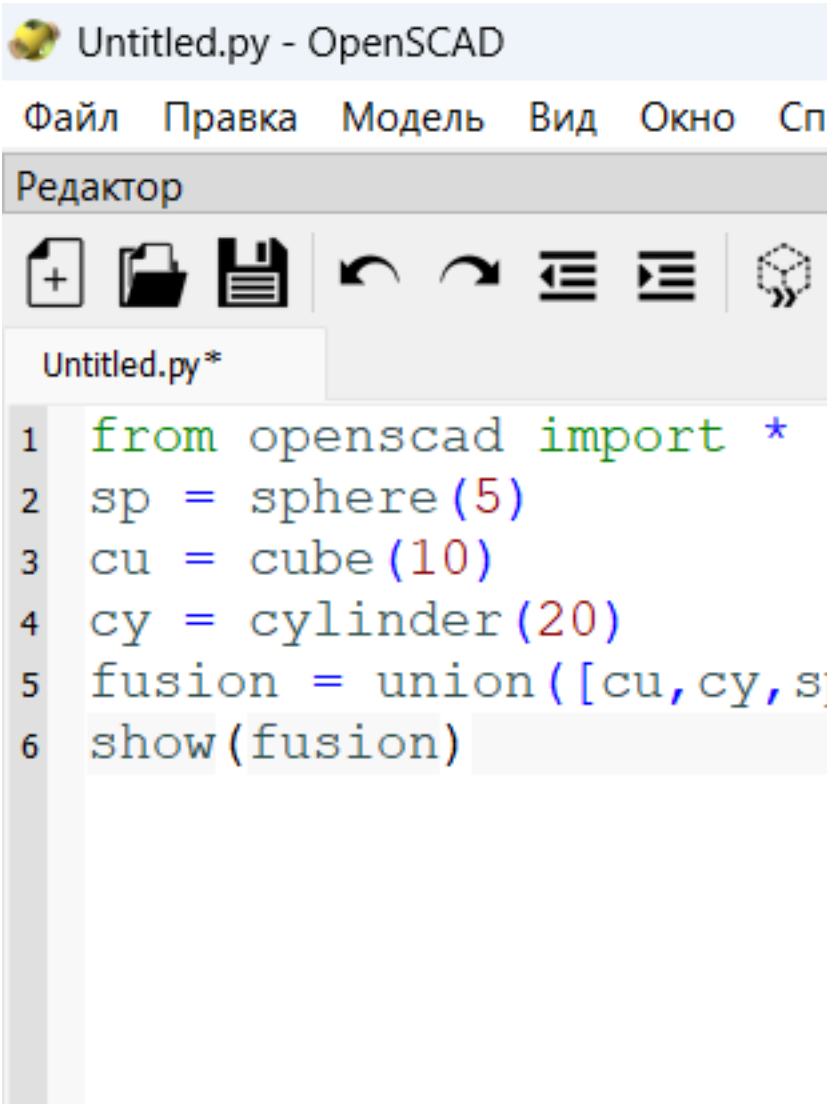

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 sp = sphere(5)
3 cu = cube(10)
4 cy = cylinder(20)
5 fusion = union([cu,cy,s
6 show(fusion)

Рис. 7. Объединение нескольких объектов

Цилиндр, сфера и куб объединяются в один сложный объект.

Листинг 4: Объединение с difference()

```
from openscad import *

cu = cube([20, 20, 10])

cy = cylinder(10, 10)

cy2 = cylinder(12, 5)

cu = cu.translate([5,5,0])

cy = cy.translate([10,10,0])

cy2 = cy2.translate([10,10,-1])

fusion = union([cu, cy])

diff = fusion.difference(cy2)
```

show(diff)

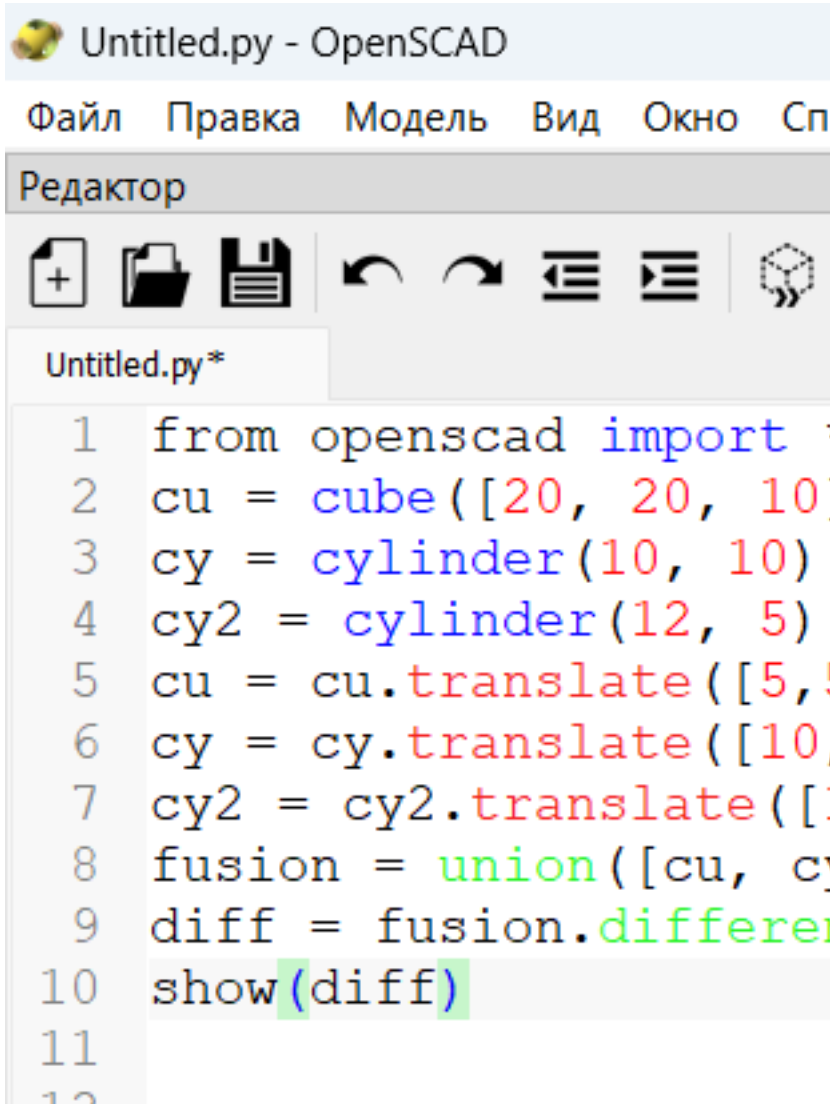

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import
2 cu = cube([20, 20, 10
3 cy = cylinder(10, 10)
4 cy2 = cylinder(12, 5)
5 cu = cu.translate([5,
6 cy = cy.translate([10
7 cy2 = cy2.translate([
8 fusion = union([cu, c
9 diff = fusion.differe
10 show(diff)
11

Рис. 8. Объединение с difference()

Заключение

union() помогает создавать сложные объекты путем объединения простых примитивов. Его можно не писать, если просто размещать объекты внутри {}. Работает в комбинации с difference() и intersection().

Вычитание (difference) в PythonSCAD(OpenSCAD)

Оператор difference() в OpenSCAD используется для вычитания одного или нескольких объектов из базовой формы. Это одна из булевых операций (Boolean operations), которая позволяет создавать сложные 3D-модели путем удаления частей из объемных фигур.

Синтаксис

```
difference() {
объект_основание;
объект_для_вычитания1;
объект_для_вычитания2;
…
}
```

Первый объект в блоке difference() является основным. Все последующие объекты вычитаются из него.

Примеры использования

Листинг 1:Отверстие в кубе

```
difference() {
cube([20, 20, 10]); // Основа: куб 20×20×10
translate([10,10,-1]) // Сдвигаем цилиндр вниз, чтобы было сквозное отверстие
cylinder(h=12, r=5); // Вычитаем цилиндр высотой 12 и радиусом 5
}
```

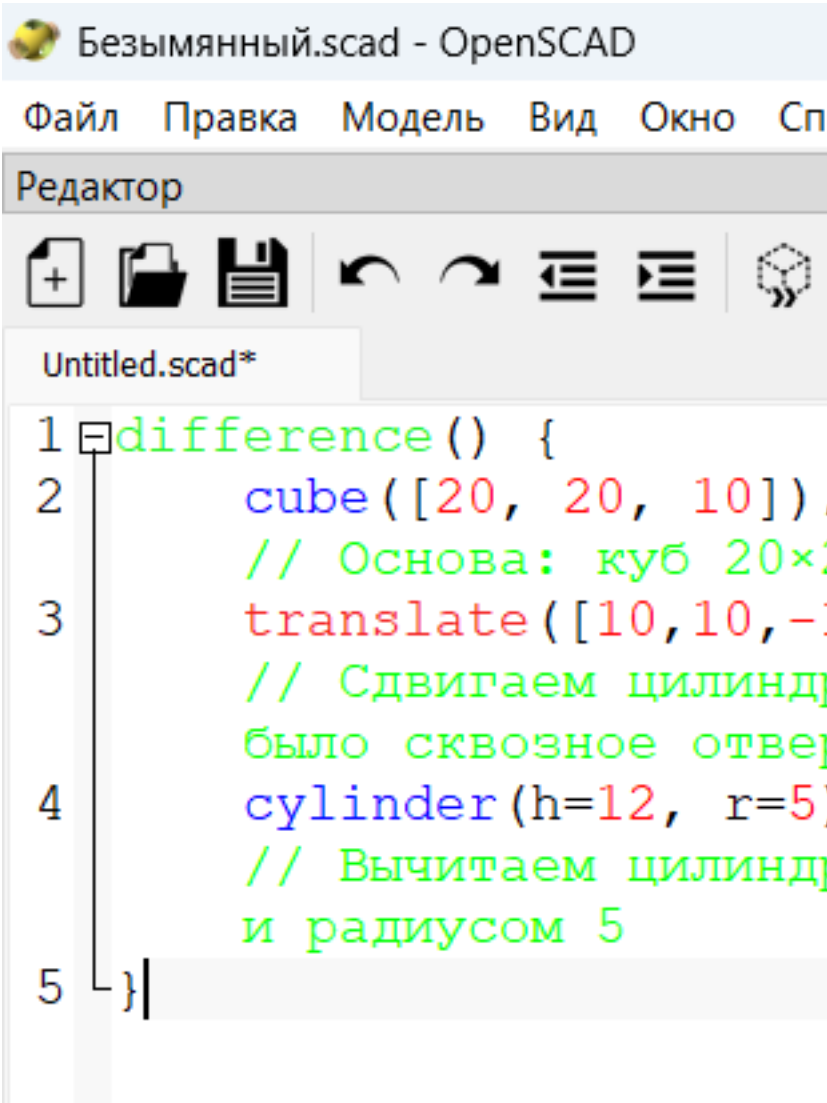

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 difference() {
2 cube([20, 20, 10]);
// Основа: куб 20×
3 translate([10,10,-
// Сдвигаем цилинд
было сквозное отве
4 cylinder(h=12, r=5
// Вычитаем цилинд
и радиусом 5
5 }

Рис. 9. Отверстие в кубе

Из куба вырезается цилиндр, создавая сквозное отверстие.

Листинг 2:Вычитание нескольких объектов

```
difference() {
cube([30, 30, 10]); // Основной куб
translate([10,10,-1]) cylinder(h=12, r=5); // Первое отверстие
translate([20,20,-1]) cylinder(h=12, r=5); // Второе отверстие
}
```

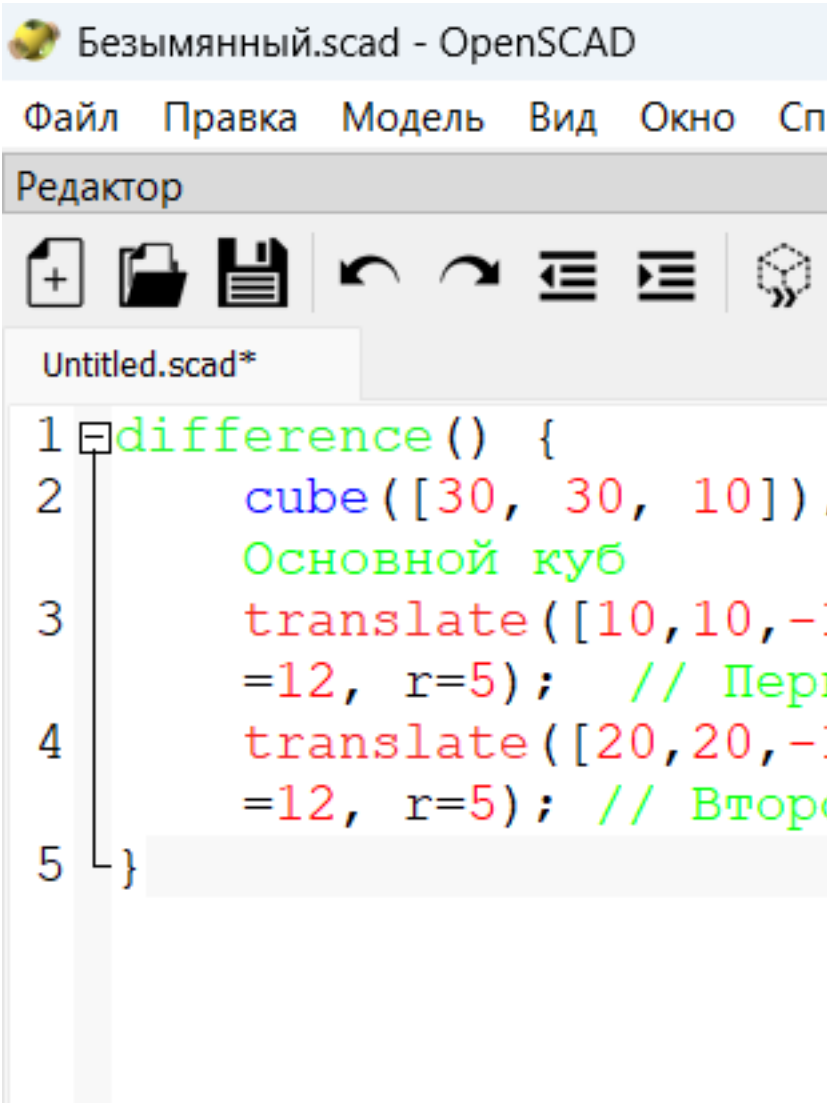

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 difference() {
2 cube([30, 30, 10]);
Основной куб
3 translate([10,10,-1
=12, r=5); // Пер
4 translate([20,20,-1
=12, r=5); // Втор
5 }

Рис. 10. Вычитание нескольких объектов

Куб 30×30×10 с двумя вырезанными отверстиями.

Листинг 3:Создание рамки путем вычитания внутреннего куба

```
difference() {
cube([30, 30, 10], center=true); // Внешний куб (большой)
cube([20, 20, 12], center=true); // Внутренний куб (меньший) вырезается
}
```

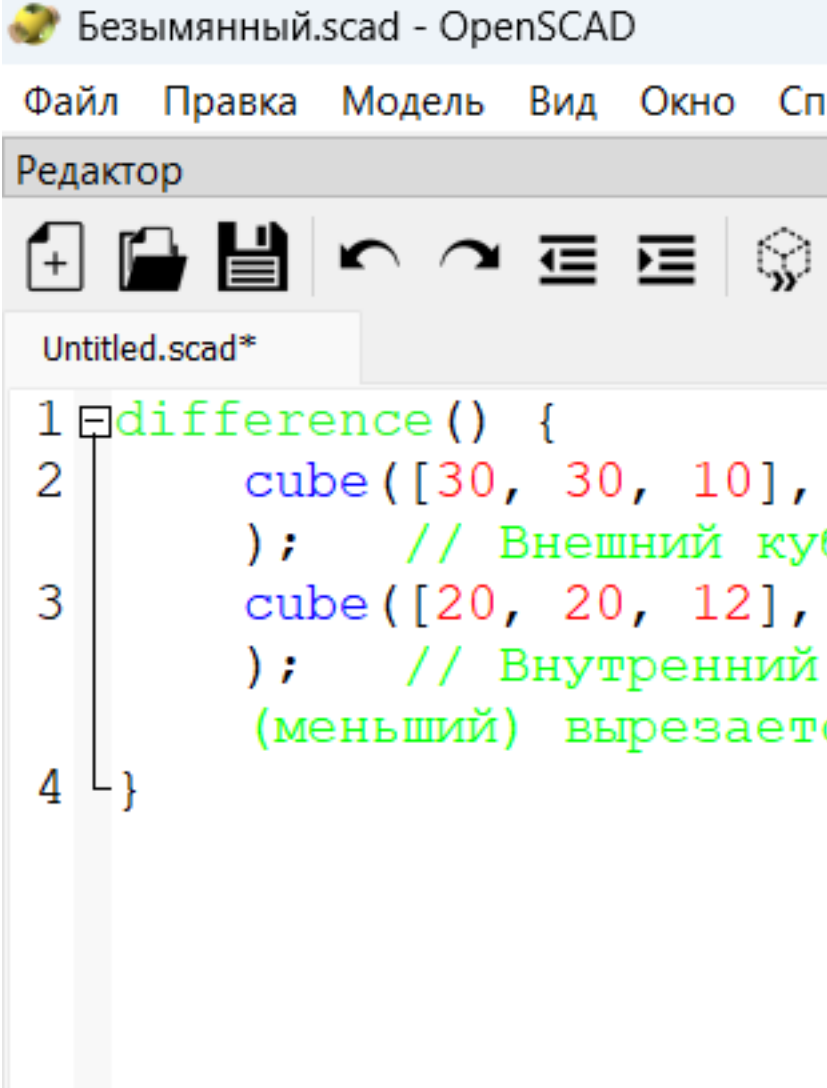

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 difference() {
2 cube([30, 30, 10],
); // Внешний ку
3 cube([20, 20, 12],
); // Внутренний
(меньший) вырезает
4 }

Рис. 11. Создание рамки путем вычитания внутреннего куба

Создает рамку из куба, вычитая из него меньший куб.

Заключение

difference() позволяет вычитать объекты, создавая сложные формы. Важно помнить, что первый объект остается, а остальные удаляются. Работает отлично в сочетании с union() и intersection().

Вычитание (difference) в PythonSCAD(Python)

Оператор difference() в OpenSCAD используется для вычитания одного или нескольких объектов из базовой формы. Это одна из булевых операций (Boolean operations), которая позволяет создавать сложные 3D-модели путем удаления частей из объемных фигур.

Синтаксис

```
from openscad import *
x = обьект1
y = обьект2
diff = x.difference(y)
show(diff)
```

Первый объект в блоке difference() является основным. Все последующие объекты вычитаются из него.

Примеры использования

Листинг 1:Отверстие в кубе

```
from openscad import *
cu = cube([3, 3, 2]) // создаём куб размерами 3x3x2
cy = cylinder(4, 1) // создаём цилиндр для создания отверстия
cy = cy.translate([1.5 , 1.5 , -1]) // перемещаем цилиндр для создания отверстия
diff = cu.difference(cy) // создаем отверстия путем вырезание ранее сделанного цилиндра
show(diff)
```

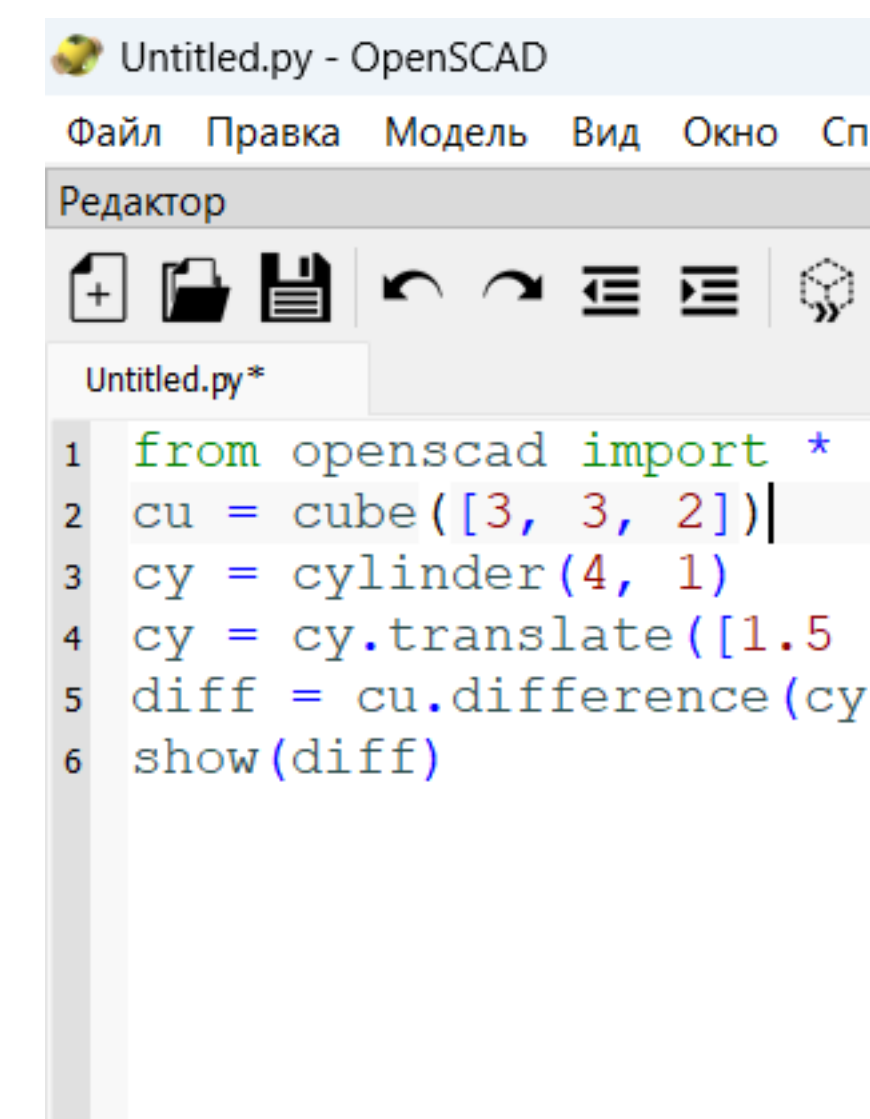

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cu = cube([3, 3, 2])
3 cy = cylinder(4, 1)
4 cy = cy.translate([1.5
5 diff = cu.difference(cy
6 show(diff)

Рис. 12. Отверстие в кубе

Из куба вырезается цилиндр, создавая сквозное отверстие.

Листинг 2:Вычитание нескольких объектов

```
from openscad import *
cu = cube([3, 7, 2]) // создаём куб
cy = cylinder(4, 1) // создаём цилиндр
cy = cy.translate([1.5 , 1.5 , -1]) // перемещаем первый цилиндр
cy2 = cylinder(4, 1) // создаём второй цилиндр
cy2 = cy2.translate([1.5 , 5 , -1]) // перемещаем второй цилиндр
diff = cu.difference([cy, cy2]) // вырезаем два цилиндра в кубе для создания фигуры
show(diff)
```

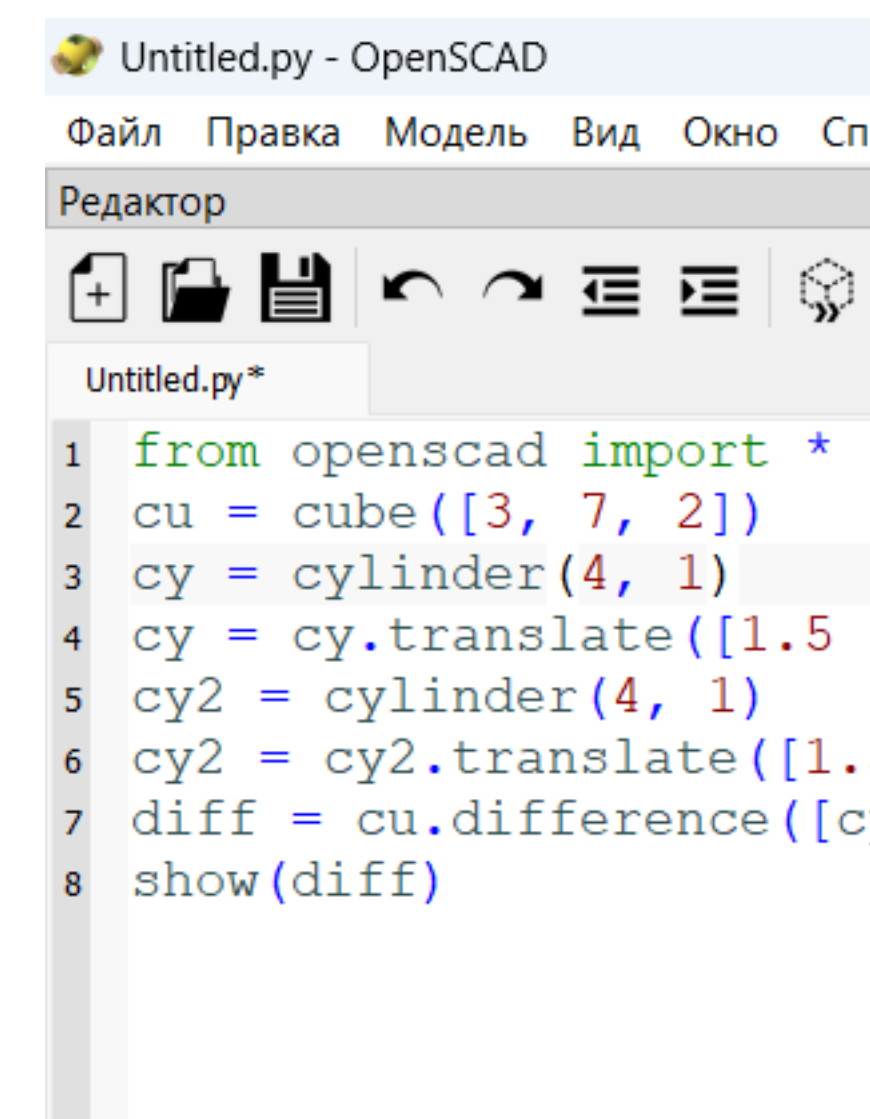

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cu = cube([3, 7, 2])
3 cy = cylinder(4, 1)
4 cy = cy.translate([1.5
5 cy2 = cylinder(4, 1)
6 cy2 = cy2.translate([1.
7 diff = cu.difference([c
8 show(diff)

Рис. 13. Вычитание нескольких объектов

Листинг 3:Создание рамки путем вычитания внутреннего куба

```
from openscad import *
x = cube([3, 3, 1]) // Внешний куб (большой)
y = cube([2, 2, 1.2]) // Внутренний куб (меньший) вырезается
y = y.translate([0.5 , 0.5 , 0.1]) // перемещение второго куба
diff = x.difference(y) // булевая операция
show(diff)
```

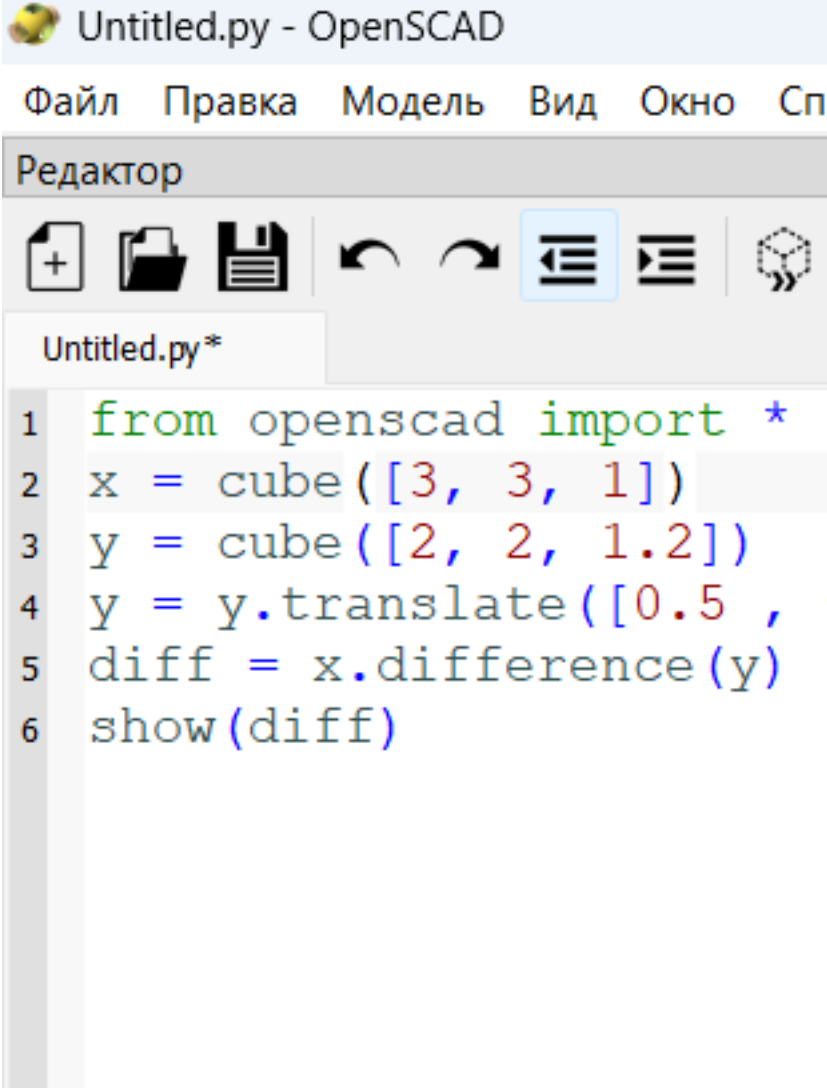

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 x = cube([3, 3, 1])
3 y = cube([2, 2, 1.2])
4 y = y.translate([0.5 ,
5 diff = x.difference(y)
6 show(diff)

Рис. 14. Создание рамки путем вычитания внутреннего куба

Создает рамку из куба, вычитая из него меньший куб.

Заключение

difference() позволяет вычитать объекты, создавая сложные формы. Важно помнить, что первый объект остается, а остальные удаляются.

Пересечение (intersection) в PyhtonSCAD(OpenSCAD)

Оператор intersection() используется для нахождения общей (пересекающейся) части между двумя или более объектами. Это одна из булевых операций (Boolean operations), которая позволяет выделять общую область у нескольких фигур.

Синтаксис

```
intersection() {
объект1;
объект2;
…
}
```

Пересекающиеся части объектов остаются.

Все области, которые не пересекаются, удаляются.

Примеры использования

Листинг 1:Пересечение куба и сферы

```
intersection() {
cube([20, 20, 20], center=true); // Куб 20x20x20
sphere(r=15); // Сфера радиусом 15
}
```

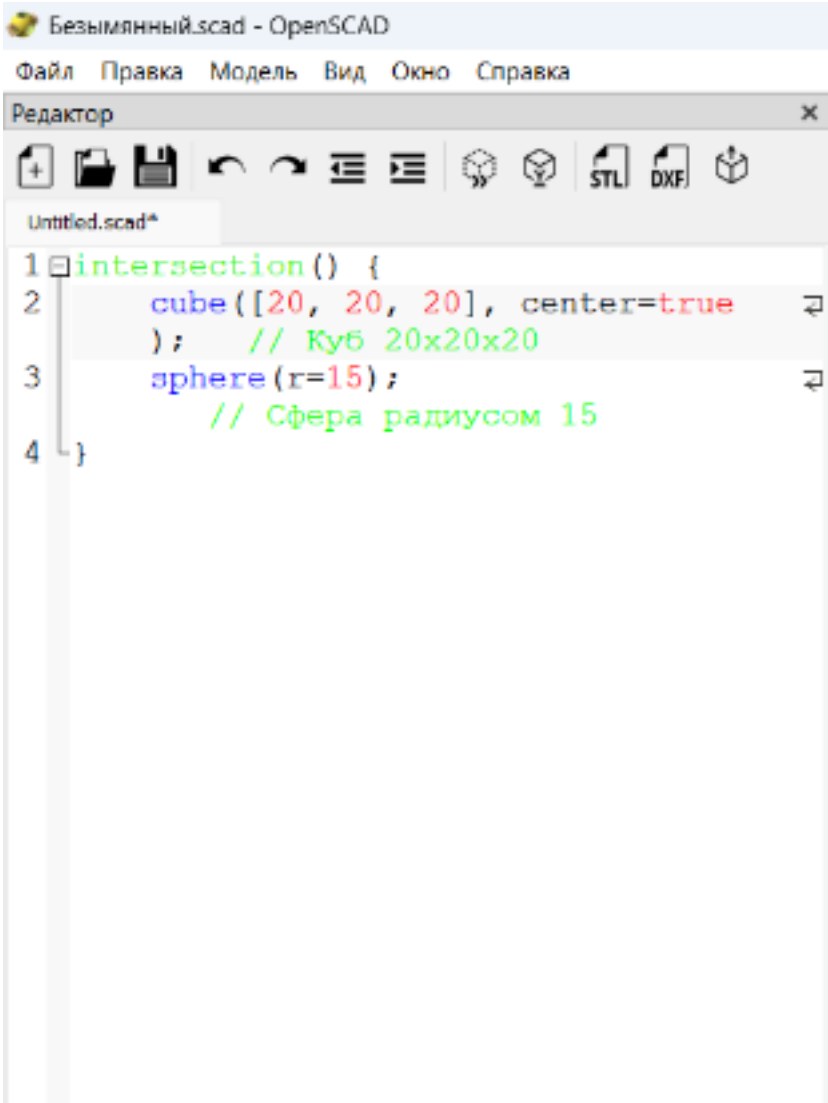

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
STL
DXF
Untitled.scad*
1 intersection() {
2 cube([20, 20, 20], center=true
); // Куб 20x20x20
3 sphere(r=15);
// Сфера радиусом 15
4 }

Рис. 15. Пересечение куба и сферы

Остается только область, где пересекаются куб и сфера.

Листинг 2:Пересечение цилиндра и куба

```
intersection() {
cylinder(h=30, r=10); // Вертикальный цилиндр
cube([20, 20, 15], center=true); // Куб (центрированный)
}
```

Безымянный.scad - OpenSCAD

Файл Правка Модель Вид Окно Сп

Редактор

Untitled.scad*

```
1 intersection() {
2     cylinder(h=30, r=1
      Вертикальный цилин
3     cube([20, 20, 15],
      ); // Куб (центрир
4 }
```

Рис. 16. Пересечение цилиндра и куба

Остается часть цилиндра, которая внутри куба.

Листинг 3:Пересечение нескольких объектов

```
intersection() {
cube([30,30,10]); // Основание – куб
translate([10,10,0]) sphere(15); // Пересекающаяся сфера
translate([15,0,0]) cylinder(h=10, r=10); // Пересекающийся цилиндр
}
```

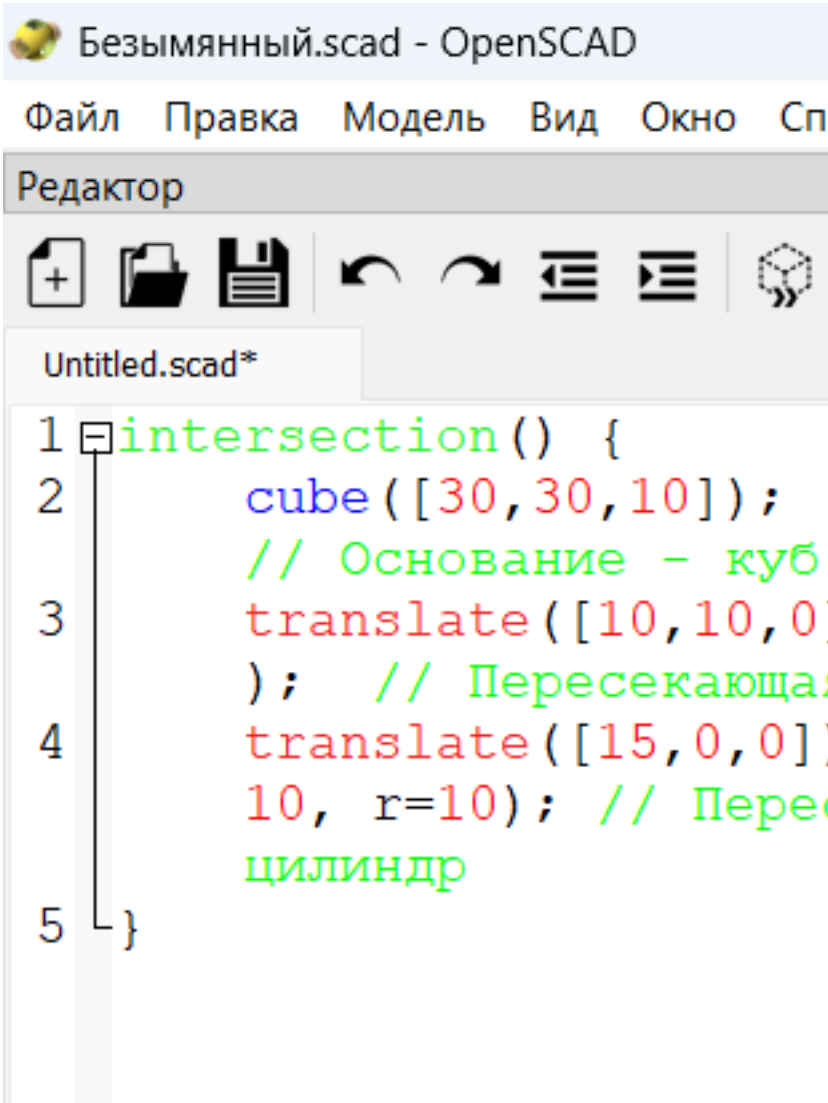

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 intersection() {
2 cube([30,30,10]);
// Основание - куб
3 translate([10,10,0
); // Пересекающа
4 translate([15,0,0]
10, r=10); // Пере
цилиндр
5 }

Рис. 17. Пересечение нескольких объектов

Только общая часть всех трех объектов останется.

Листинг 4:Пересечение с difference()

Можно использовать intersection() совместно с difference(), чтобы вырезать общую часть из другой фигуры:

```
intersection() {
difference() {
cube([30,30,10]);
translate([5,5,-1]) cylinder(h=12, r=10);
}
translate([10,10,0]) sphere(15);
}
```

Безымянный.scad - OpenSCAD

Файл Правка Модель Вид Окно Сп

Редактор

Untitled.scad*

```
1 intersection() {
2     difference() {
3         cube([30,30,10
4         translate([5,5,
      (h=12, r=10);
5     }
6     translate([10,10,0
7 }
```

Рис. 18. Пересечение с difference()

Сначала из куба вырезается цилиндр.

Потом остается только пересечение с сферой.

Заключение

intersection() оставляет только общие области всех входящих объектов. Можно использовать для создания сложных деталей. Работает в комбинации с union() и difference().

Пересечение (intersection) в PyhtonSCAD(Python)

Оператор intersection() используется для нахождения общей (пересекающейся) части между двумя или более объектами. Это одна из булевых операций (Boolean operations), которая позволяет выделять общую область у нескольких фигур.

Синтаксис

```
from openscad import *
x = обьект1
y = обьект2
inter = x.intersection(y)
show(inter)
```

Пересекающиеся части объектов остаются.

Все области, которые не пересекаются, удаляются.

Примеры использования

Листинг 1:Пересечение куба и сферы

```
from openscad import *
cu = cube(20)
sp = sphere(15)
inter = cu.intersection(sp)
show(inter)
```

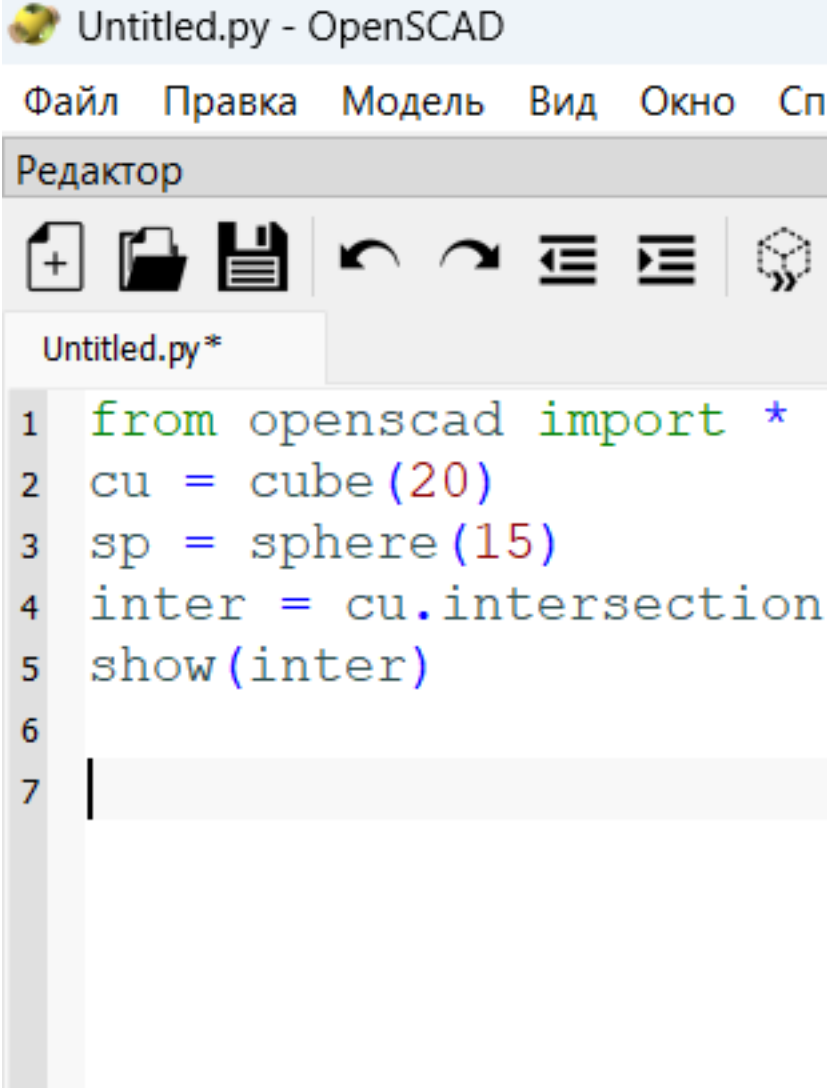

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cu = cube(20)
3 sp = sphere(15)
4 inter = cu.intersection
5 show(inter)
6
7

Рис. 19. Пересечение куба и сферы

Остается только область, где пересекаются куб и сфера.

Листинг 2:Пересечение цилиндра и куба

```
from openscad import *
cu = cube([20, 20, 15])
cy = cylinder(30, 10)
inter = cu.intersection(cy)
show(inter)
```

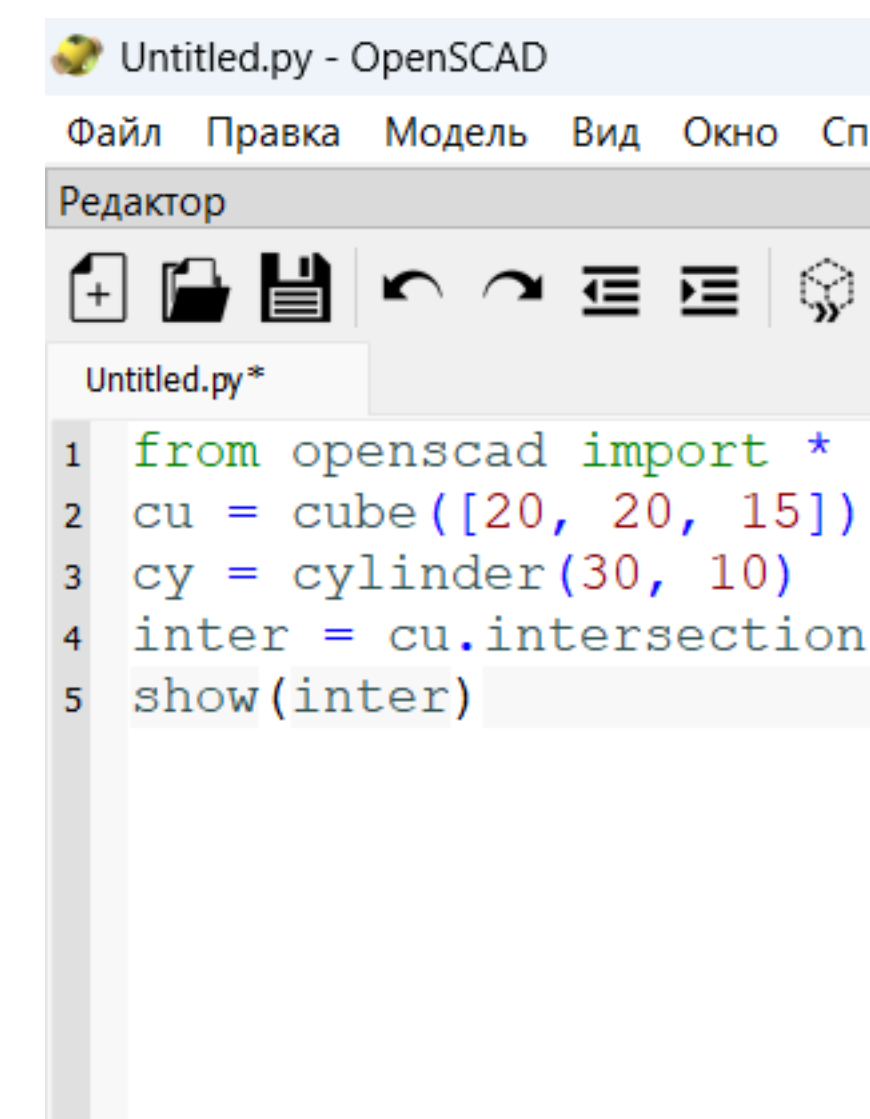

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cu = cube([20, 20, 15])
3 cy = cylinder(30, 10)
4 inter = cu.intersection
5 show(inter)

Рис. 20. Пересечение цилиндра и куба

Остается часть цилиндра, которая внутри куба.

Листинг 3:Пересечение нескольких объектов

```
from openscad import *
cu = cube([30,30,10])
sp = sphere(15)
cu = cu.translate([10,10,0])
cy = cylinder(10, 10)
cy = cy.translate([15,0,0])
inter = sp.intersection([cy, cu])
show(inter)
```

Untitled.py - OpenSCAD

Файл Правка Модель Вид Окно Сп

Редактор

Untitled.py*

```
1  from openscad import *
2  cu = cube([30,30,10])
3  sp = sphere(15)
4  cu = cu.translate([10,1
5  cy = cylinder(10, 10)
6  cy = cy.translate([15,0
7  inter = sp.intersection
8  show(inter)
9
```

Рис. 21. Пересечение нескольких объектов

Только общая часть всех трех объектов останется.

Листинг 4:Пересечение с difference()

Можно использовать intersection() совместно с difference(), чтобы вырезать общую часть из другой фигуры:

```
from openscad import *
cu = cube([30,30,10])
sp = sphere(15)
cy = cylinder(12, 10)
cy = cy.translate([15,0,-1])
sp = sp.translate([10,10,0])
diff = cu.difference(cy)
inter = diff.intersection(sp)
show(inter)
```

Untitled.py - OpenSCAD

Файл Правка Модель Вид Окно Сп

Редактор

Untitled.py*

```
1  from openscad import *
2  cu = cube([30,30,10])
3  sp = sphere(15)
4  cy = cylinder(12, 10)
5  cy = cy.translate([15,0
6  sp = sp.translate([10,1
7  diff = cu.difference(cy
8  inter = diff.intersecti
9  show(inter)
10
11
```

Рис. 22. Пересечение с difference()

Сначала из куба вырезается цилиндр.

Потом остается только пересечение с сферой.

Заключение

intersection() оставляет только общие области всех входящих объектов. Можно использовать для создания сложных деталей. Работает в комбинации с union() и difference().

Задача 1:Создание примитивов

Какой оператор используется для создания сферы?

1) sphere()

2) circle()

3) ball()

4) round()

Задача 2:Сфера

Какой параметр отвечает за радиус сферы?

1) r

2) d

3) radius

4) length

Задача 3:Создание примитивов

Какая команда создаёт куб?

1) cube()

2) rectangle()

3) block()

4) square()

5) box()

Задача 4:Куб

Как задать размеры куба 10x20x30?

1) cube([10,20,30])

2) cube(10,20,30)

3) cube(size=[10,20,30])

4) cube({10,20,30})

5) cube<10,20,30>

Задача 5:Создание примитивов

Как создать цилиндр высотой 50 и радиусом 10?

1) cylinder(h=50, r=10)

2) cylinder(10,50)

3) cylinder(radius=10, height=50)

4) cylinder(h=10, r=50)

5) cylinder(d=10, h=50)

Задача 6:Цилиндр

Какой параметр позволяет сделать цилиндр усечённым (разные радиусы сверху и снизу)?

1) r1, r2

2) scale

3) d1, d2

4) radius_top, radius_bottom

5) trunc

Задача 7:Цилиндр

Какая команда создаёт конус?

1) cylinder(r1=10, r2=0, h=20)f

2) trapezoid()
3) cone()
4) pyramid()
5) cylinder(10,0,20)

Задача 8:Создание примитивов

Какой примитив создаёт 2D-прямоугольник?
1) square()
2) rectangle()
3) box2D()
4) shape()
5) plane()

Задача 9:Прямоугольник

Как создать прямоугольник 15x25?
1) square([15,25])
2) square(15,25)
3) rect([15,25])
4) square({15,25})
5) square<15,25>

Задача 10:Создание примитивов

Как создать трёхмерный многогранник?
1) polyhedron()
2) polygon()
3) polyhedron(points, faces)
4) mesh()
5) shape3D()

Задача 11:Создание примитивов

Какой оператор создаёт 2D-круг?

1) circle()

2) sphere()

3) round()

4) ellipse()

5) disc()

Задача 12: 2D-круг

Как задать диаметр у 2D-круга?

1) circle(d=20)

2) circle(radius=20)

3) circle(r=20)

4) circle(diameter=20)

5) circle(20)

Задача 14:Вычитание

Какой оператор позволяет вычитать одну фигуру из другой?

1) difference()

2) union()

3) intersect()

4) add()

5) subtract()

Задача 15:Пересечение

Какой оператор создаёт пересечение двух фигур?

1) intersection()

2) merge()

3) blend()

4) unite()
5) overlap()

Задача 16:Объединение

Какой оператор объединяет несколько объектов?
1) union()
2) merge()
3) sum()
4) group()
5) combine()

Задача 1:Создание примитивов

Какой оператор используется для создания сферы?
1) sp = sphere(r)
2) r = round()
3) x = cube()
4) l = coil()

Задача 2:Сфера

Какой параметр отвечает за радиус сферы?
1) r
2) x
3) v
4) w

Задача 3:Создание примитивов

Какая команда создаёт куб?
1) cu = cube()
2) sp = sphere()
3) cy = cylinder()

4) tr = trasnparent()

Задача 4:Куб

Как задать размеры куба 10x20x30?

1) cu = cube([10, 20, 30])

2) cu = cube(10, 20, 30)

3) sp = sphere([10, 20, 30])

4) cy = cylinder(10, 20, 30)

Задача 5:Создание примитивов

Как создать цилиндр высотой 50 и радиусом 10?

1) cy = cylinder(50, 10)

2) cy = cube(50, 10)

3) cy = sphere(50, 10)

4) r = round(5)

Задача 6:Цилиндр

Какой параметр позволяет сделать цилиндр усечённым (разные радиусы сверху и снизу)?

1) r1, r2

2) v1, v2

3) h

4) g

Задача 7:Цилиндр

Какая команда создаёт конус?

1) cy =cylinder(10, 5 ,0)

2) cy =cylinder(10, 0 ,5)

3) cy =cylinder(10, 5)

4) cy =cylinder(10, 5 ,5)

Задача 8:Операторы

Можно ли использовать несколько операторов в одном коде?

1) Да

2) Нет

Задача 9:Операторы

Какой код совмещает два оператора?

1) diff = fusion.difference(.....)

2) fusion = union([diff, x])

3) diff = x.difference(.....)

4) c = coil()

Задача 10:Операторы

Какой код соединяет куб и сферу?

1) fusion = union([cu, sp])

2) diff = sp.difference()cu

3) inter = sp.intersection(cu)

4) c = cu.coil(sp)

Задача 11:Операторы

Какой код соединяет цилиндр и сферу?

1) fusion = union([cy, sp])

2) inter = cy.intersection(cu, sp)

3) diff = cy.difference(cu)

4) c = cu.coil(sp)

Задача 12:Операторы

Какой код вырезает цилиндр и куб?

1) diff = cy.difference(cu)

2) c = coil()

3) fusion = union([cy, sp])

4) inter = cy.intersection(cu, sp)

Задача 13:Операторы

Какой код пересекает цилиндр, куб и сферу?

1) inter = cy.intersection(cu, sp)

2) fusion = union([cy, sp])

3) c = coil()

4) diff = cy.difference(cu)

Задача 14:Вычитание

Какой оператор позволяет вычитать одну фигуру из другой?

1) diff = .difference()

2) fusion = union([])

3) inter = .intersection()

4) c = coil()

Задача 15:Пересечение

Какой оператор создаёт пересечение двух фигур?

1) inter = intersection()

2) c = coil()

3) diff = difference()

4) fusion = union()

Задача 16:Объединение

Какой оператор объединяет несколько объектов?

1) fusion = union()

2) c = coil()

3) diff = difference()

4) inter = intersection()

Модуль 4: Трансформации объектов

Повороты

(rotate)

в

PythonSCAD(OpenSCAD)

Оператор rotate() используется для поворота объектов вокруг одной или нескольких осей (X, Y, Z). Он позволяет изменять ориентацию объектов в 3D-пространстве.

Синтаксис

rotate([угол_X, угол_Y, угол_Z]) объект;

угол_X – поворот вокруг оси X (горизонтальная ось).

угол_Y – поворот вокруг оси Y (вертикальная ось, направлена вглубь экрана).

угол_Z – поворот вокруг оси Z (перпендикулярна экрану).

Угол задается в градусах (°) и может быть положительным (по часовой стрелке) или отрицательным (против часовой стрелки).

Также можно использовать альтернативный синтаксис с одной осью:

rotate(угол, [x, y, z]) объект;

Здесь [x, y, z] – вектор направления оси вращения (1 – ось активна, 0 – нет).

Примеры использования

Листинг 1:Поворот вокруг оси Z

```
rotate([0, 0, 45]) cube([10, 20, 5]);
```

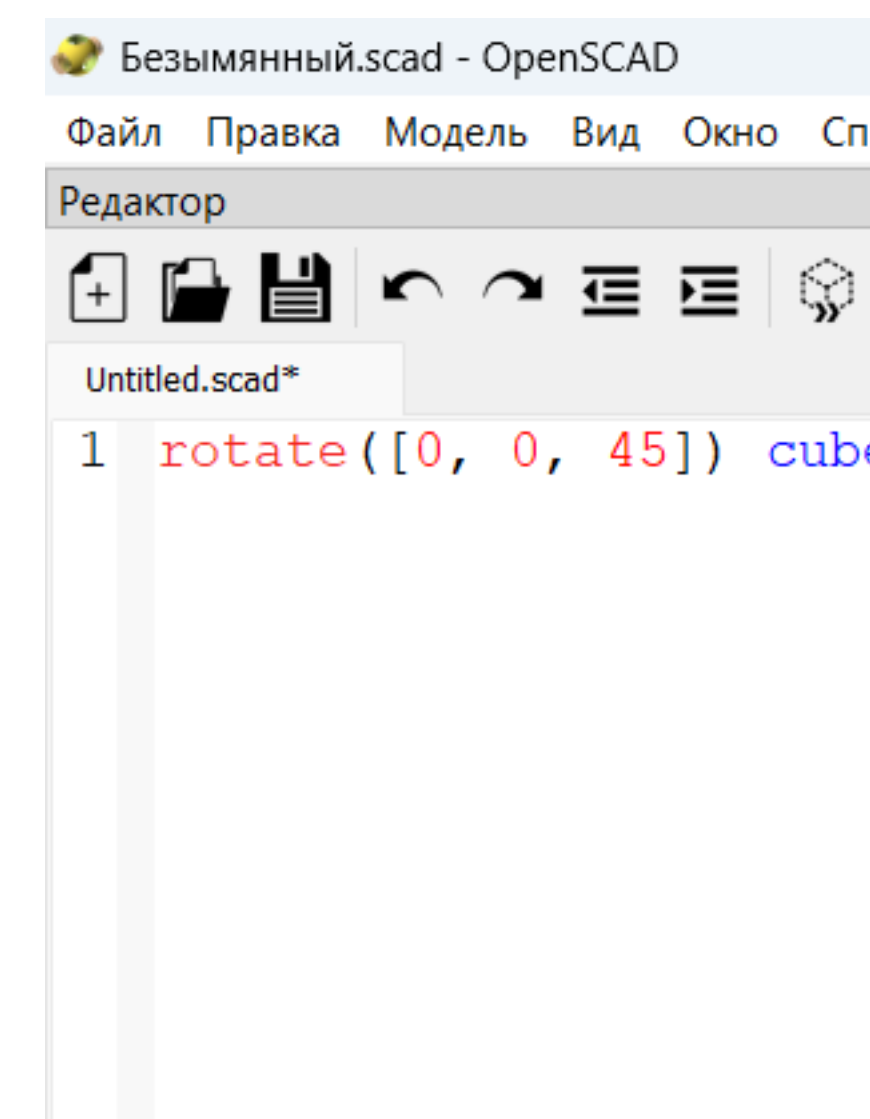

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 rotate([0, 0, 45]) cube

Рис. 1. Поворот вокруг оси Z
Куб повернется на 45° вокруг оси Z.
Листинг 2:Поворот вокруг нескольких осей
rotate([30, 45, 0]) cylinder(h=20, r=5);

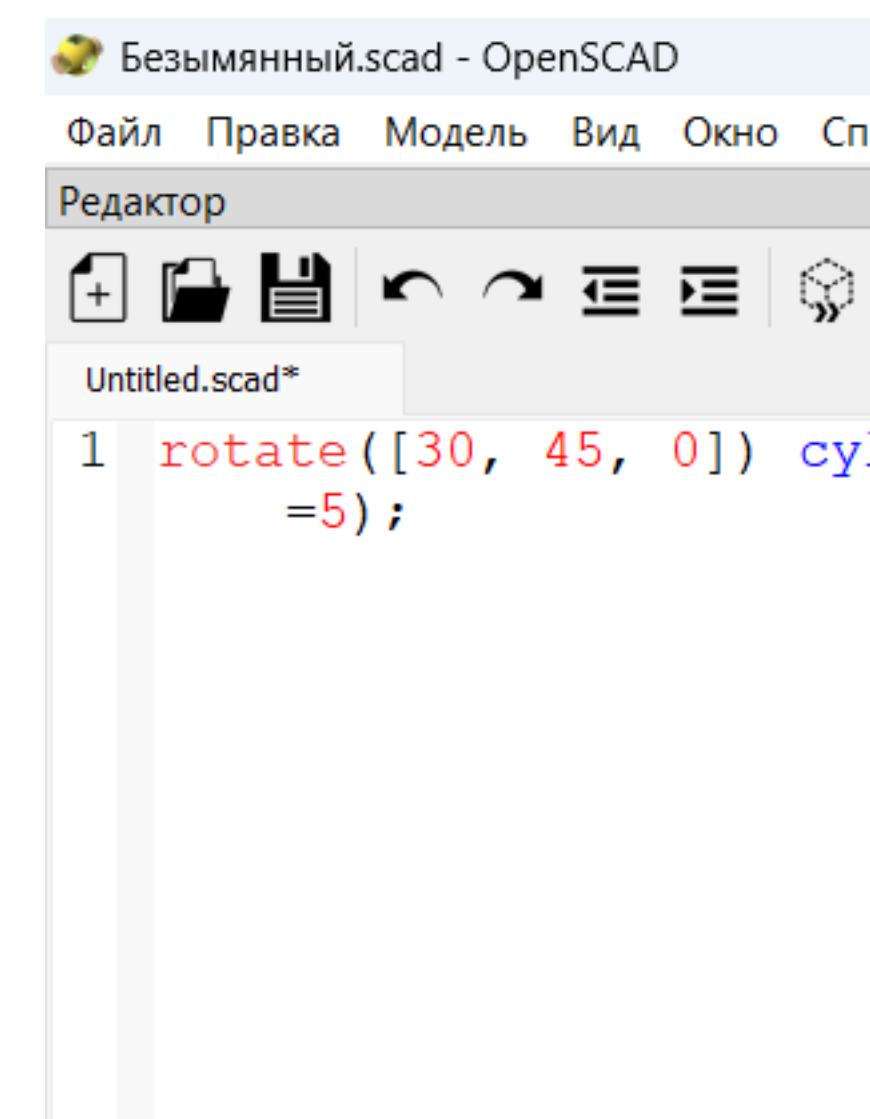

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 rotate([30, 45, 0]) cy
=5);

Рис. 2. Поворот вокруг нескольких осей
Цилиндр наклонится на 30° по X и на 45° по Y.

Листинг 3:Поворот вокруг произвольной оси

```
rotate(60, [1, 1, 0]) cube([10, 10, 10]);
```

Безымянный.scad - OpenSCAD

Файл Правка Модель Вид Окно Сп

Редактор






Untitled.scad*

```
1  rotate(60, [1, 1, 0]) c
       10]);
```

Рис. 3. Поворот вокруг произвольной оси

Куб повернется на 60° вокруг диагонали XY.

Листинг 4:Комбинация с translate()

Поворот выполняется относительно начала координат, поэтому перед вращением можно переместить объект:

```
translate([10, 0, 0])
rotate([0, 0, 45]) cube([10, 10, 10]);
```

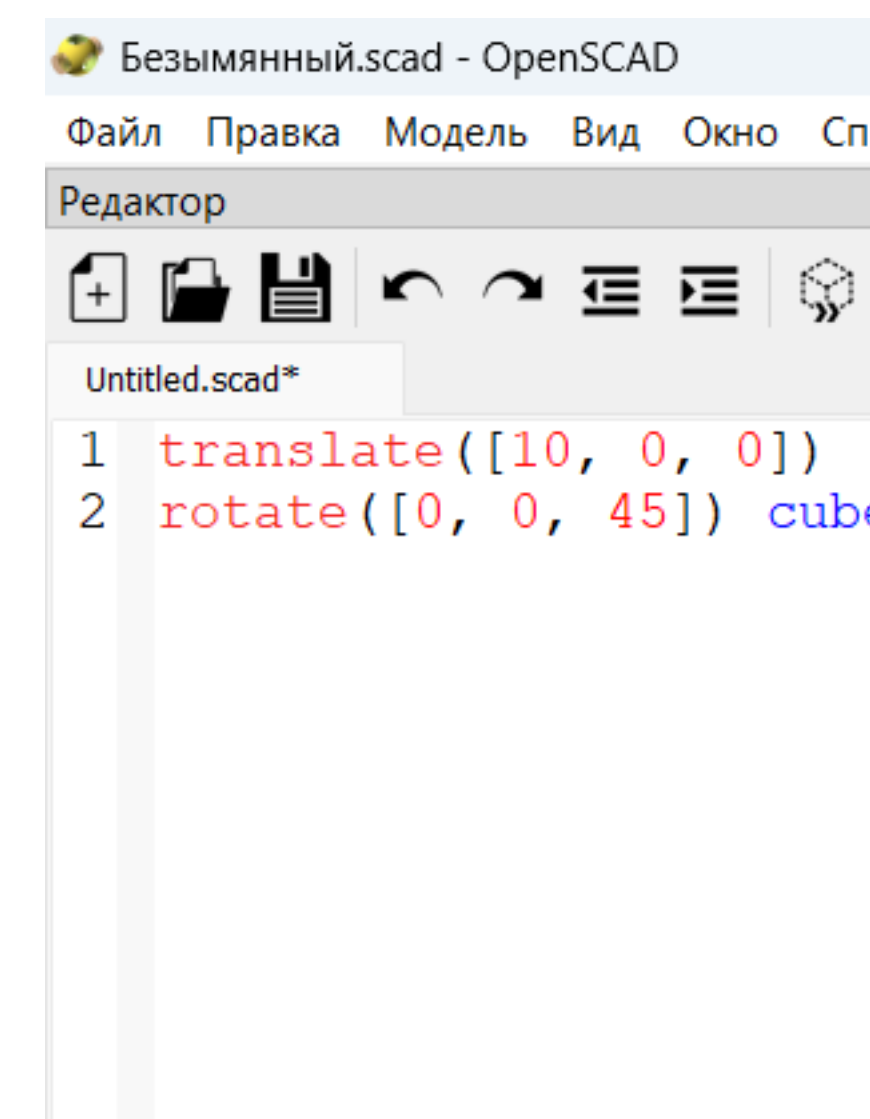

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 translate([10, 0, 0])
2 rotate([0, 0, 45]) cube

Рис. 4. Комбинация с translate()

Куб сначала сдвинется, а потом повернется.

Если порядок команд поменять (rotate() → translate()), то объект сначала повернется в исходной точке, а потом сместится.

Листинг 5: Вращение в цикле (for)

```
for (i = [0:30:360]) {
rotate([0, 0, i]) translate([20, 0, 0]) sphere(2);
}
```

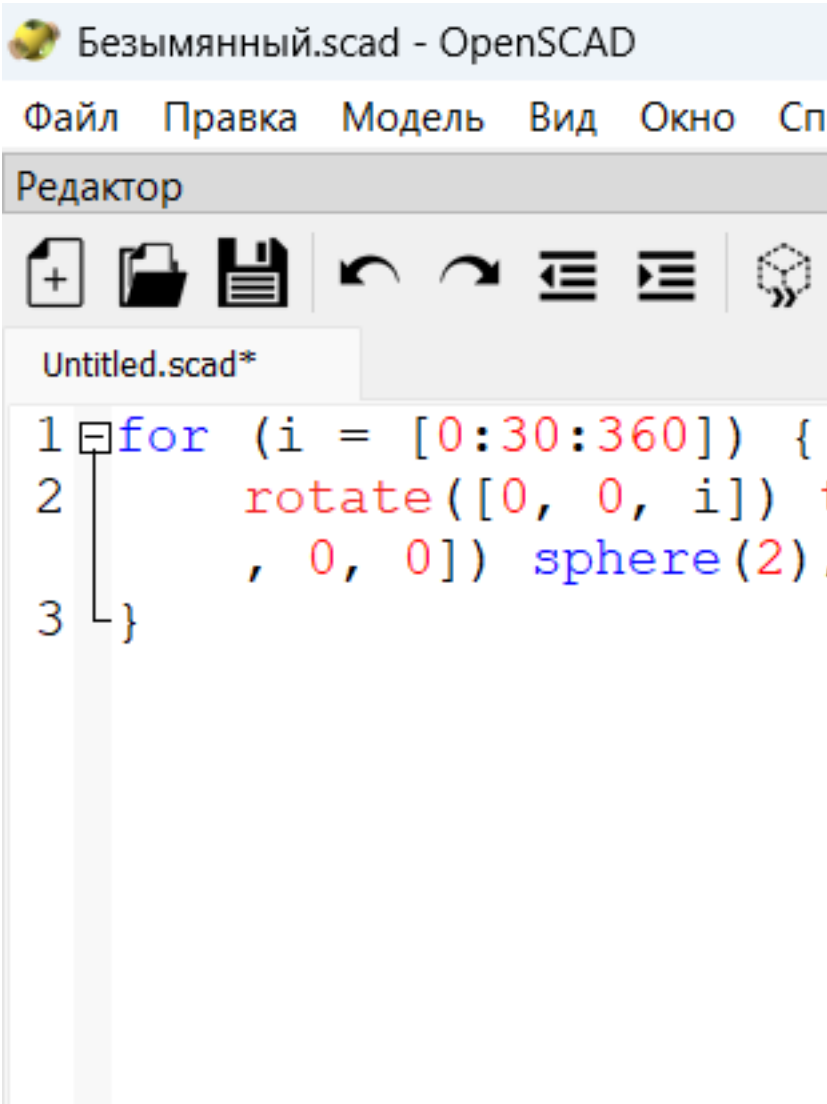

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 for (i = [0:30:360]) {
2 rotate([0, 0, i]) t
, 0, 0]) sphere(2)
3 }

Рис. 5. Вращение в цикле (for)

Создает кольцо сфер, расположенных каждые 30° вокруг центра.

Заключение

rotate([x, y, z]) поворачивает объект вокруг одной или нескольких осей. Важно учитывать порядок операций, особенно при использовании translate(). Можно использовать for, чтобы создавать повторяющиеся структуры.

Повороты (rotate) в PythonSCAD(Python)

Оператор rotate() используется для поворота объектов вокруг одной или нескольких осей (X, Y, Z). Он позволяет изменять ориентацию объектов в 3D-пространстве.

Синтаксис

rotated=объект.rotate([угол_X, угол_Y, угол_Z])

угол_X – поворот вокруг оси X (горизонтальная ось).

угол_Y – поворот вокруг оси Y (вертикальная ось, направлена вглубь экрана).

угол_Z – поворот вокруг оси Z (перпендикулярна экрану).

Угол задается в градусах (°) и может быть положительным (по часовой стрелке) или отрицательным (против часовой стрелки).

Здесь [x, y, z] – вектор направления оси вращения (1 – ось активна, 0 – нет).

Примеры использования

Листинг 1:Поворот вокруг оси Z

```
from openscad import *
cu = cube([10, 20, 5])
rotated = cu.rotate([0, 0, 45])
show(rotated)
```

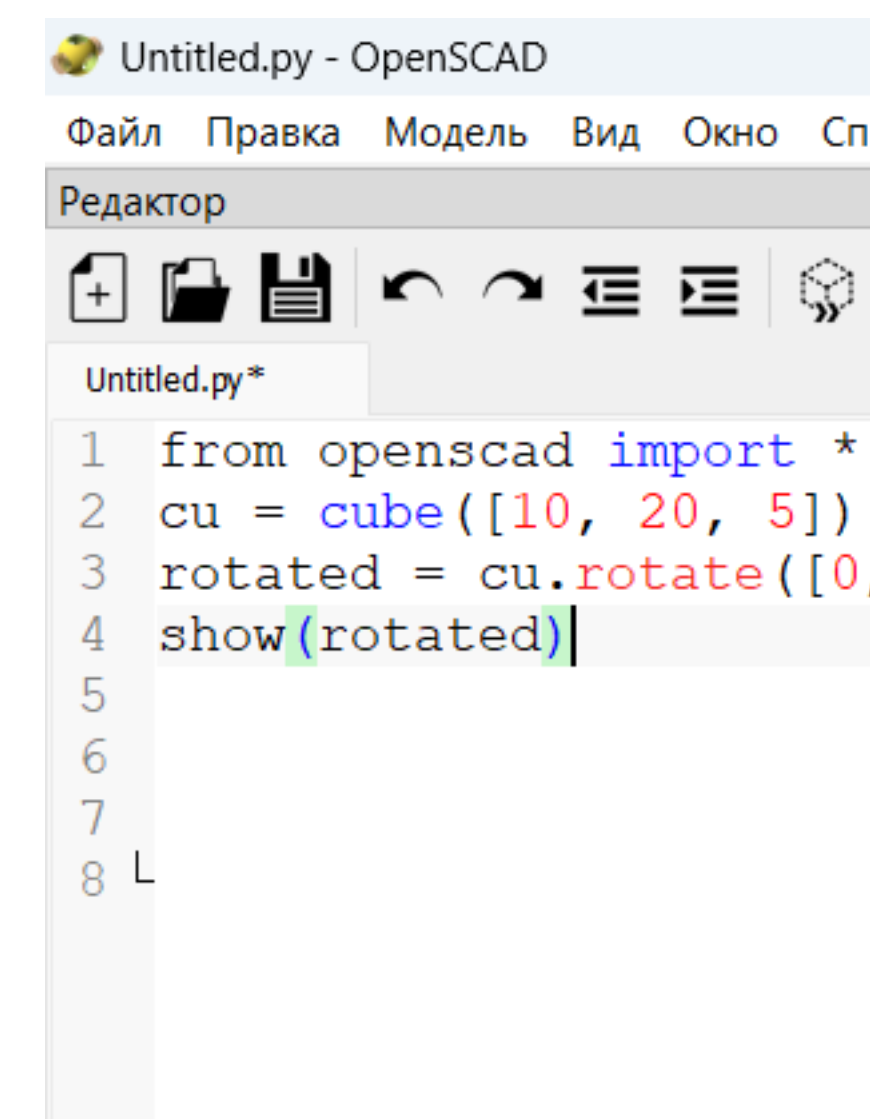

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cu = cube([10, 20, 5])
3 rotated = cu.rotate([0,
4 show(rotated)
5
6
7
8

Рис. 6. Поворот вокруг оси Z
Куб повернется на 45° вокруг оси Z.

Листинг 2:Поворот вокруг нескольких осей

```
cy = cylinder(20, 5)
rotated = cy.rotate([30, 45, 0])
show(rotated)
```

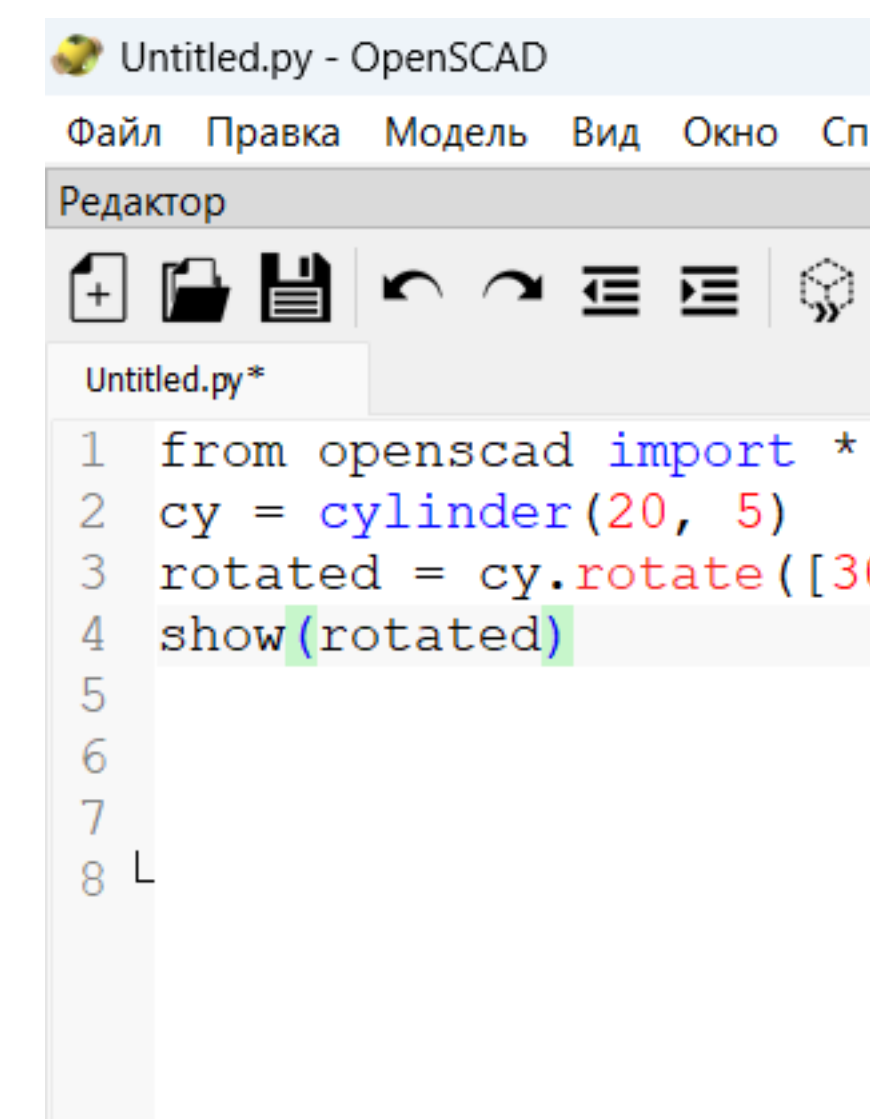

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cy = cylinder(20, 5)
3 rotated = cy.rotate([3
4 show(rotated)
5
6
7
8

Рис. 7. Поворот вокруг нескольких осей
Цилиндр наклонится на 30° по X и на 45° по Y.

Листинг 3:Поворот вокруг произвольной оси

```
cu = cube([10, 10, 10])
rotated = cu.rotate(60, [1, 1, 0])
show(rotated)
```

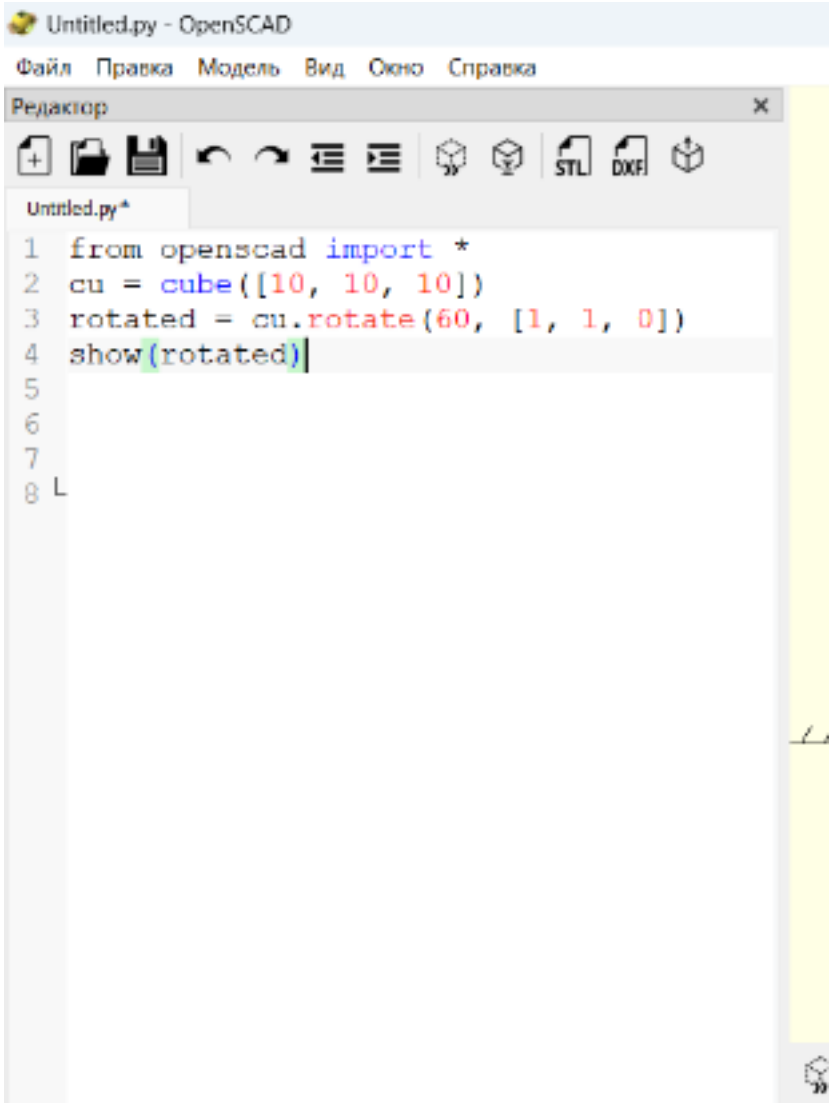

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
STL
DXF
Untitled.py*
1 from openscad import *
2 cu = cube([10, 10, 10])
3 rotated = cu.rotate(60, [1, 1, 0])
4 show(rotated)
5
6
7
8

Рис.8. Поворот вокруг произвольной оси
Куб повернется на 60° вокруг диагонали XY.

Листинг 5:Комбинация с translate

```
()
cu = cube([10, 10, 10])
cu = cu.translate([10, 0, 0])
rotated = cu.rotate([0, 0, 45])
show(rotated)
```

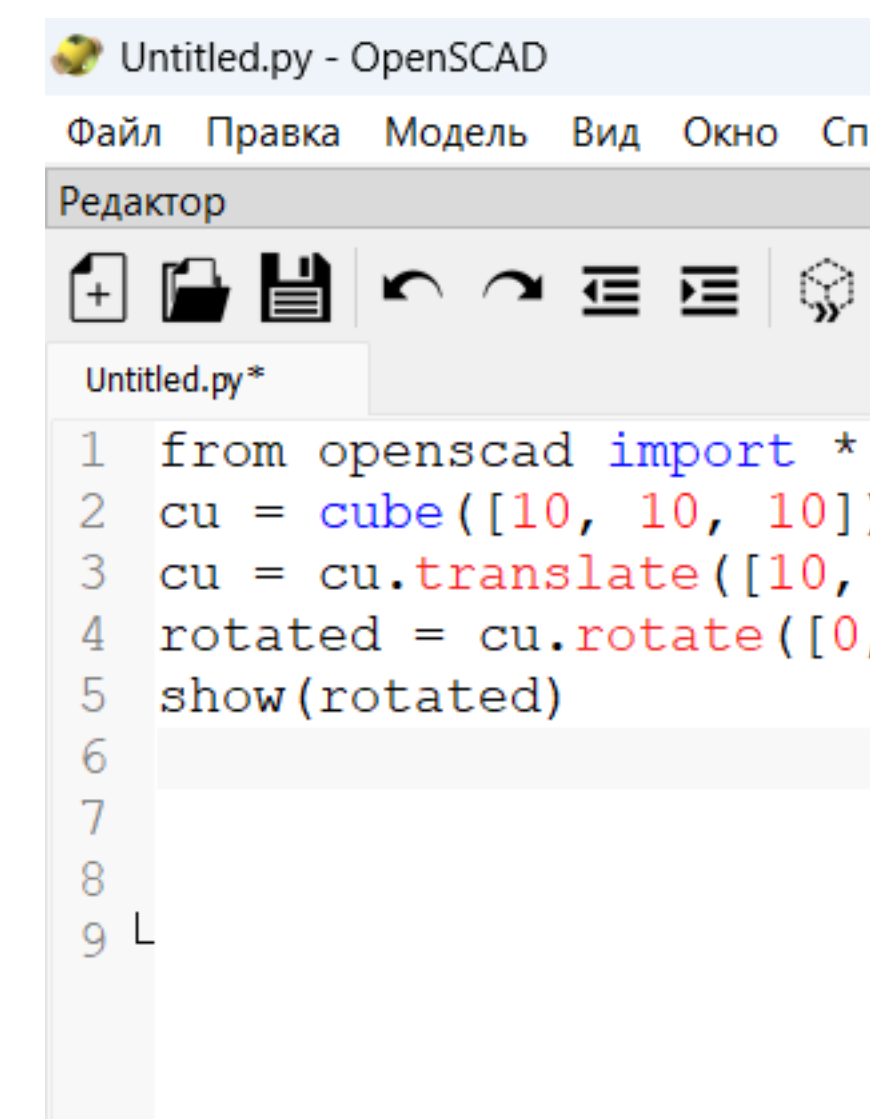

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cu = cube([10, 10, 10])
3 cu = cu.translate([10,
4 rotated = cu.rotate([0,
5 show(rotated)
6
7
8
9

Рис. 9. Комбинация с translate()

Куб сначала сдвинется, а потом повернется.

Если порядок команд поменять (rotate() → translate()), то объект сначала повернется в исходной точке, а потом сместится.

Заключение

rotate([x, y, z]) поворачивает объект вокруг одной или нескольких осей. Важно учитывать порядок операций, особенно при использовании translate(). Можно использовать for, чтобы создавать повторяющиеся структуры.

Смещения (translate) в PythonSCAD(OpenSCAD)

Оператор translate() используется для перемещения объектов в 3D-пространстве вдоль осей X, Y и Z.

Синтаксис

translate([X, Y, Z]) объект;

X – сдвиг вдоль оси X (вправо – положительные значения, влево – отрицательные).

Y – сдвиг вдоль оси Y (вперед – положительные значения, назад – отрицательные).

Z – сдвиг вдоль оси Z (вверх – положительные значения, вниз – отрицательные).

Примеры использования

Листинг 1:Простое смещение куба

```
translate([10, 20, 5]) cube([10, 10, 10]);
```

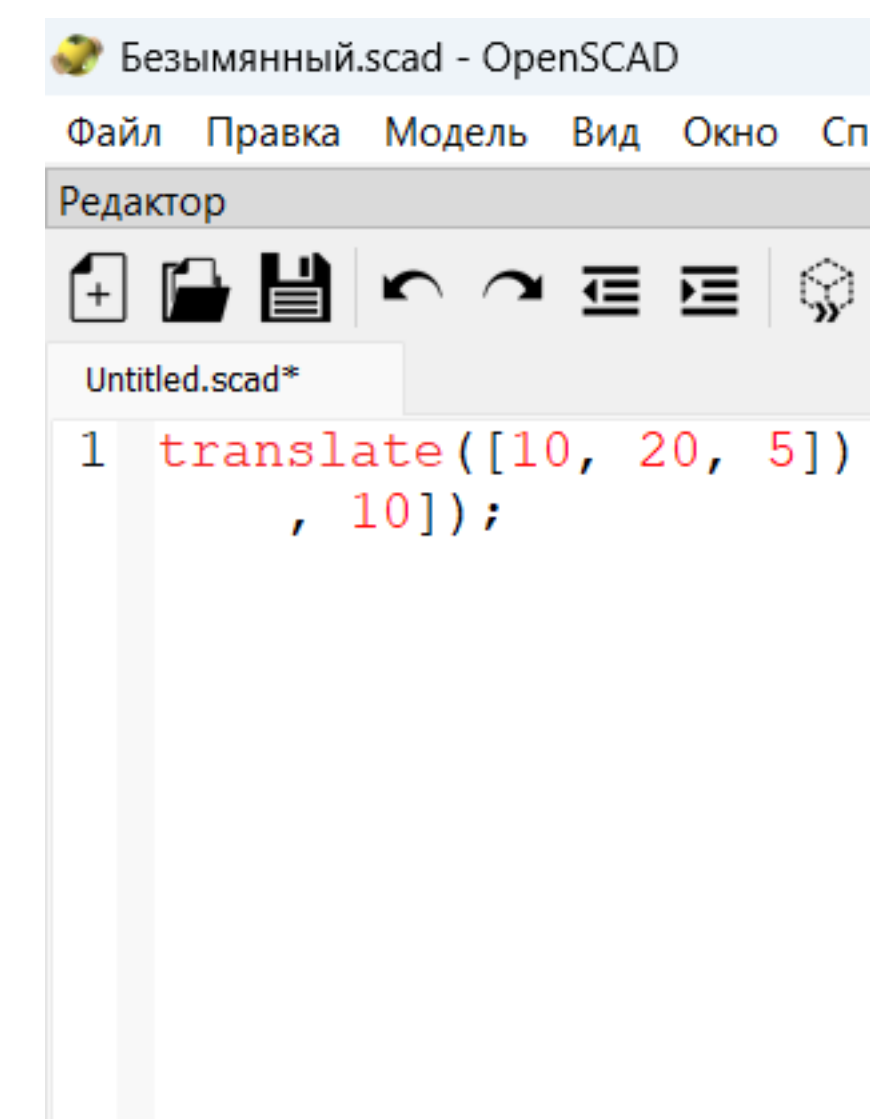

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 translate([10, 20, 5])
, 10]);

Рис. 10. Простое смещение куба

Куб размером 10×10×10 сдвигается на 10 единиц вправо, 20 вперед и 5 вверх.

Листинг 2:Смещение сферы вверх

```
translate([0, 0, 15]) sphere(10);
```

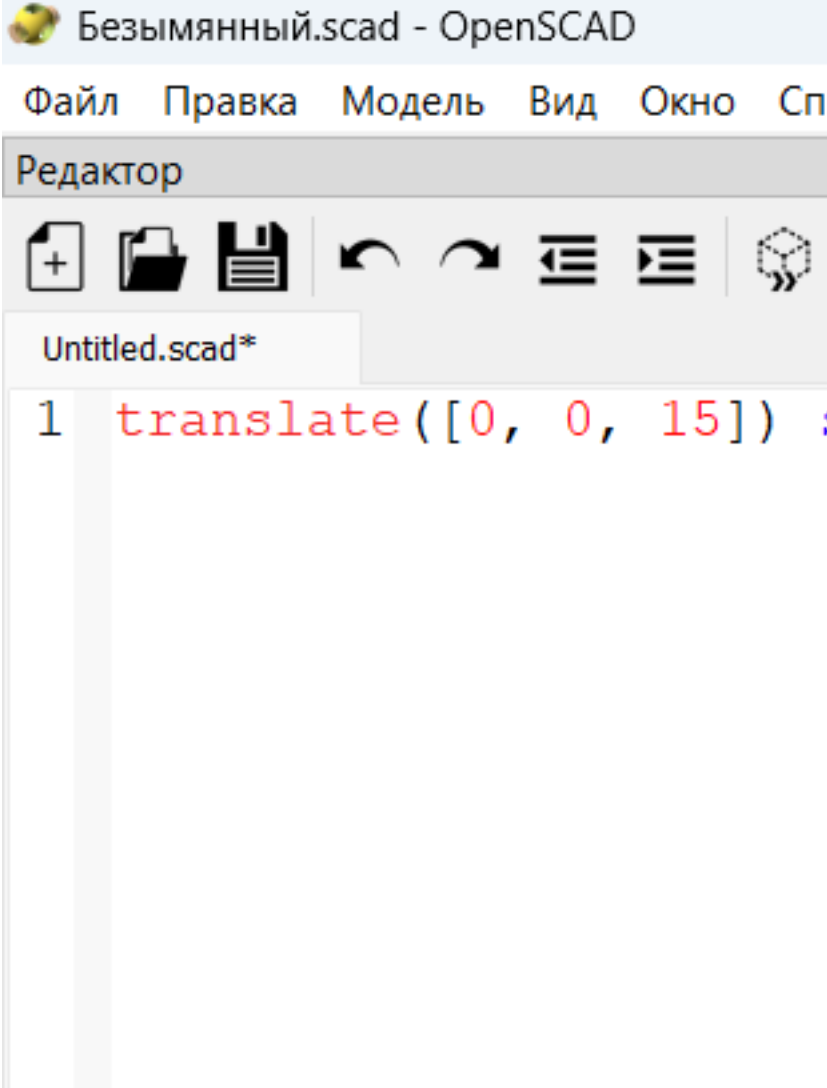

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 translate([0, 0, 15])

Рис. 11. Смещение сферы вверх
Сфера радиусом 10 поднимается на 15 единиц вверх.
Листинг 3: Комбинация translate() и rotate()

```
translate([20, 0, 0])
rotate([0, 0, 45]) cube([10, 10, 10]);
```


Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 translate([20, 0, 0])
2 rotate([0, 0, 45]) cube

Рис. 12. Комбинация translate() и rotate()

Куб сначала сдвигается, затем поворачивается.

Если поменять порядок команд (rotate() → translate()), результат будет другим!

Листинг 4:Размещение нескольких объектов в for -цикле

```
for (i = [0:20:60]) {
translate([i, 0, 0]) sphere(5);
}
```

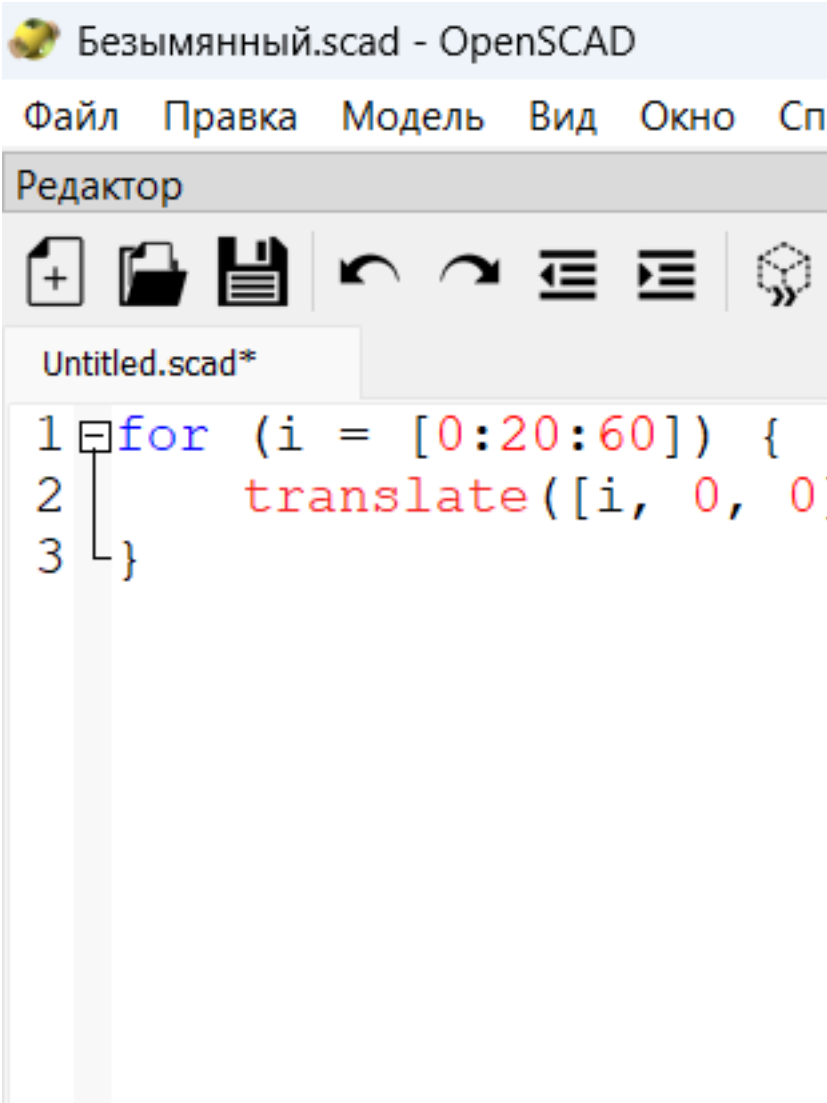

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 for (i = [0:20:60]) {
2 translate([i, 0, 0
3 }

Рис. 13. Размещение нескольких объектов в for-цикле

Создает ряд сфер, сдвинутых вдоль оси X на 0, 20 и 40 единиц.

Листинг 5: Вложенные Translate()

```
translate([10, 0, 0]) {
translate([0, 20, 0]) cube([10, 10, 10]);
}
```

Безымянный.scad - OpenSCAD

Файл Правка Модель Вид Окно Сп

Редактор







Untitled.scad*

```
1 translate([10, 0, 0])
2     translate([0, 20,
      10, 10]);
3 }
```

Рис. 14. Вложенные Translate()

Сначала объект смещается на 10 единиц по X,

Потом внутри блока – на 20 по Y.

Заключение

translate([X, Y, Z]) перемещает объект в 3D-пространстве. Важно учитывать порядок операций при сочетании с rotate(). Используется в циклах for для создания повторяющихся структур.

Смещения
(translate)
в
PythonSCAD(Python)

Оператор translate() используется для перемещения объектов в 3D-пространстве вдоль осей X, Y и Z.

Синтаксис

x = обьект.translate([x, y, z])

X – сдвиг вдоль оси X (вправо – положительные значения, влево – отрицательные).

Y – сдвиг вдоль оси Y (вперед – положительные значения, назад – отрицательные).

Z – сдвиг вдоль оси Z (вверх – положительные значения, вниз – отрицательные).

Примеры использования

Листинг 1:Простое смещение куба

```
from openscad import *
```

```
cu = cube([10, 10, 10])
tr = cu.translate([10, 20, 5])
show(tr)
```

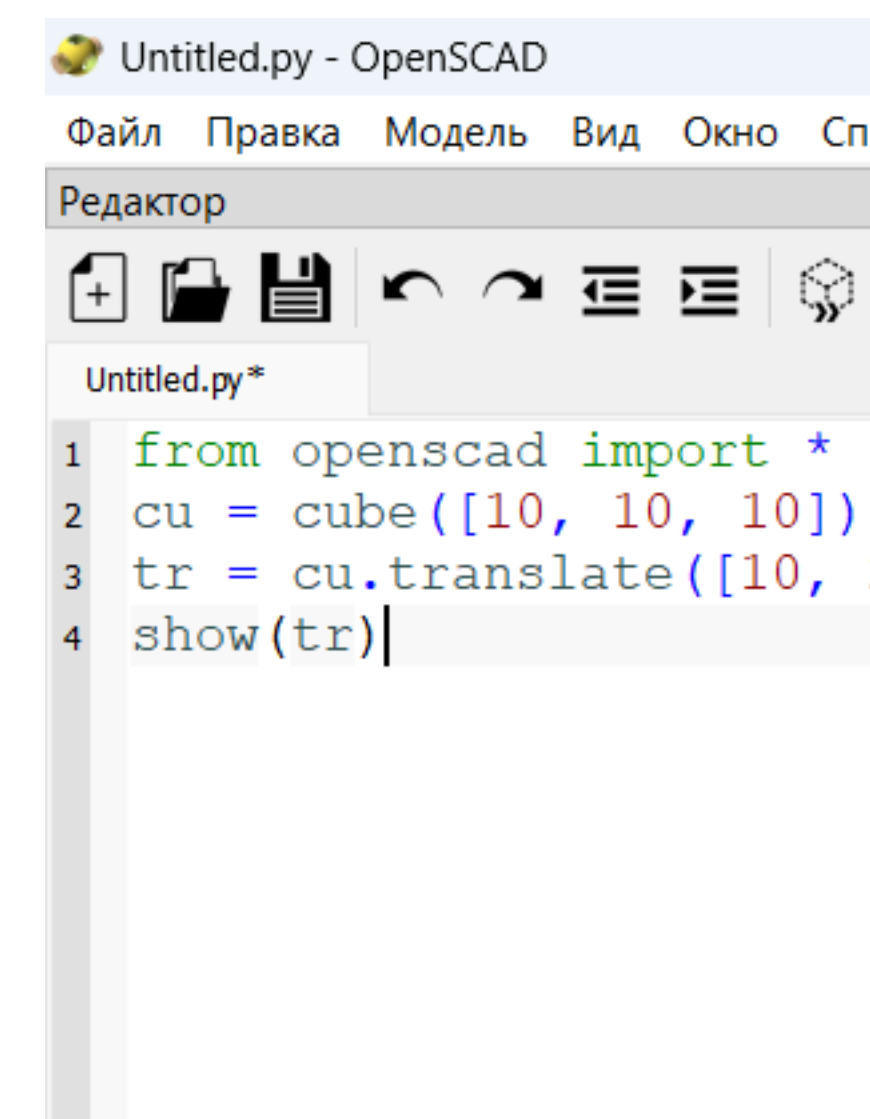

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cu = cube([10, 10, 10])
3 tr = cu.translate([10,
4 show(tr)

Рис. 15. Простое смещение куба

Куб размером 10×10×10 сдвигается на 10 единиц вправо, 20 вперед и 5 вверх.

Листинг 2:Смещение сферы вверх

```
from openscad import *
sp = sphere(10)
tr = sp.translate([0, 0, 15])
show(tr)
```

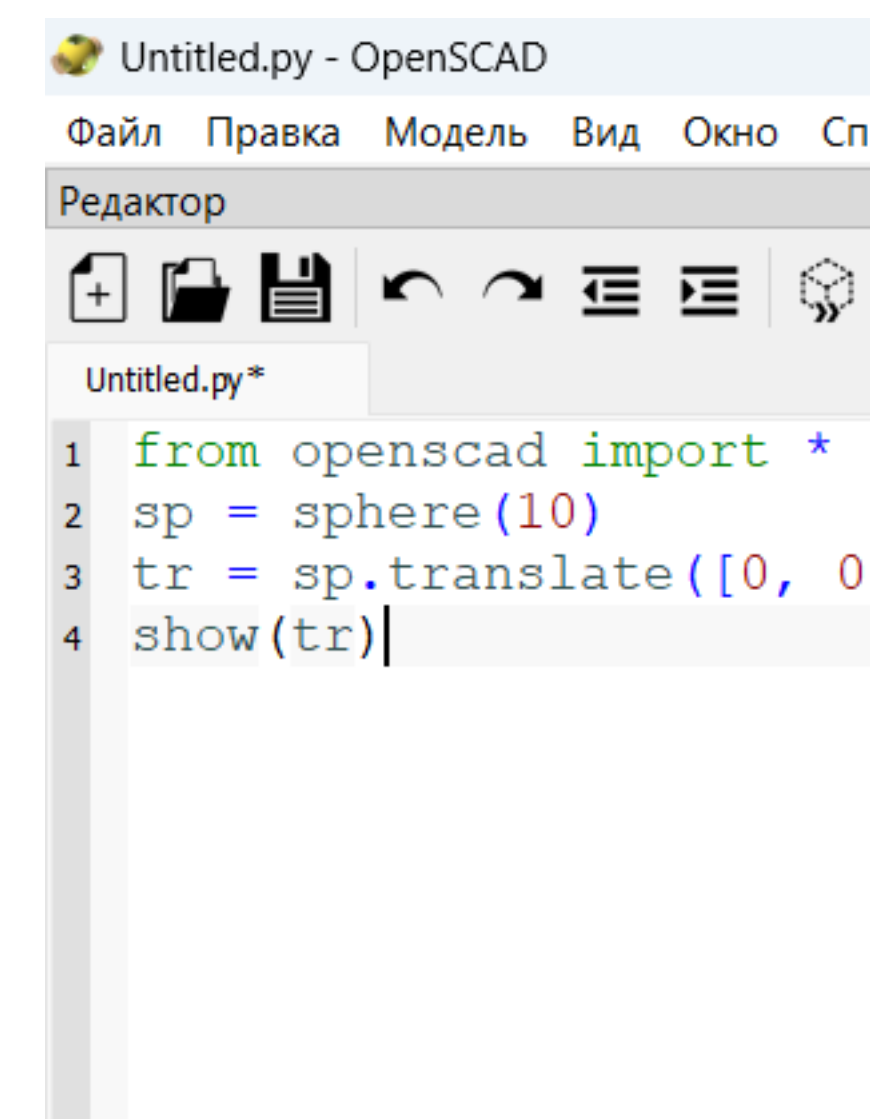

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 sp = sphere(10)
3 tr = sp.translate([0, 0
4 show(tr)

Рис. 16. Смещение сферы вверх

Сфера радиусом 10 поднимается на 15 единиц вверх.

Листинг 3: Комбинация translate() и rotate()

```
cu = cube([10, 10, 10])
tr = cu.translate([20, 0, 0])
rotated = tr.rotate([0, 0, 45])
show([rotated])
```

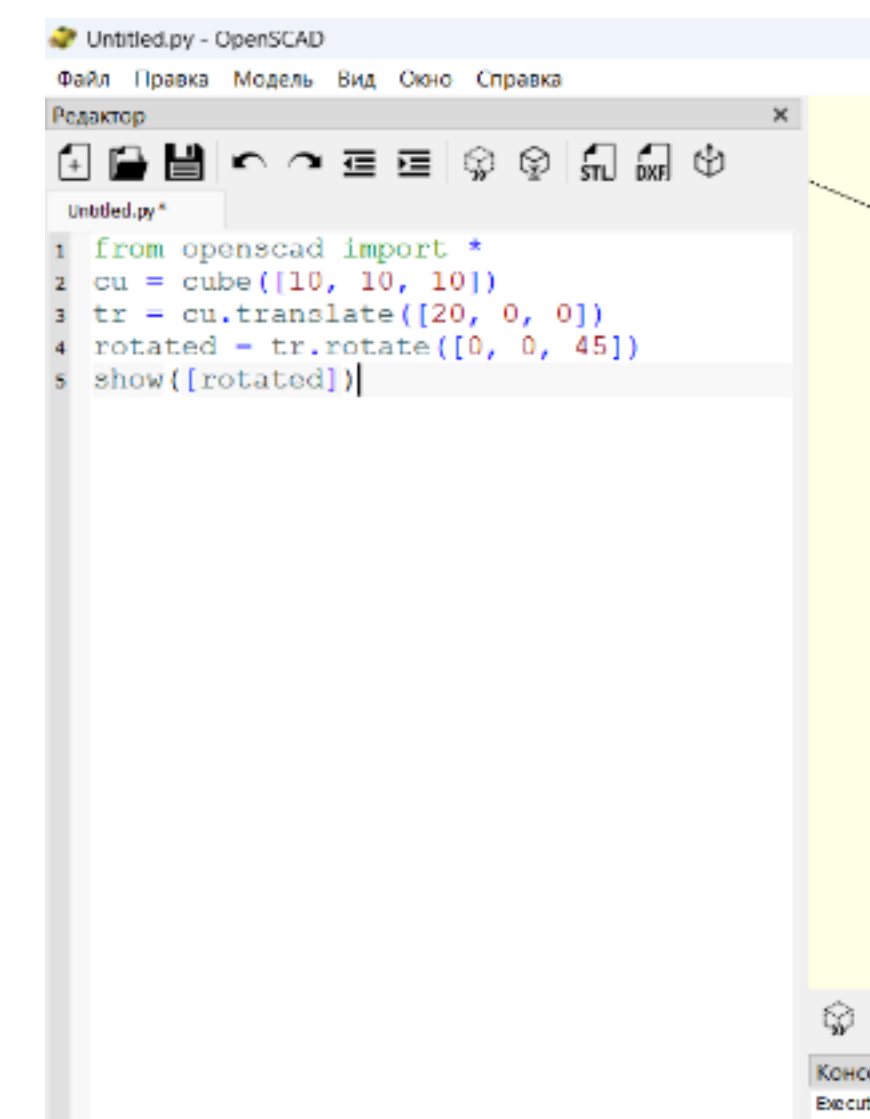

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
STL
DXF
Untitled.py *
1 from openscad import *
2 cu = cube([10, 10, 10])
3 tr = cu.translate([20, 0, 0])
4 rotated = tr.rotate([0, 0, 45])
5 show([rotated])

Рис. 17. Комбинация translate() и rotate()

Куб сначала сдвигается, затем поворачивается.

Если поменять порядок команд (rotate() → translate()), результат будет другим!

Листинг 4:Размещение нескольких объектов в for –
цикле

```
from openscad import *
sp = sphere(5)
result = sp
for i in range(1, 4):
result = result | sp.translate([15 * i, 0, 0])
show(result)
```

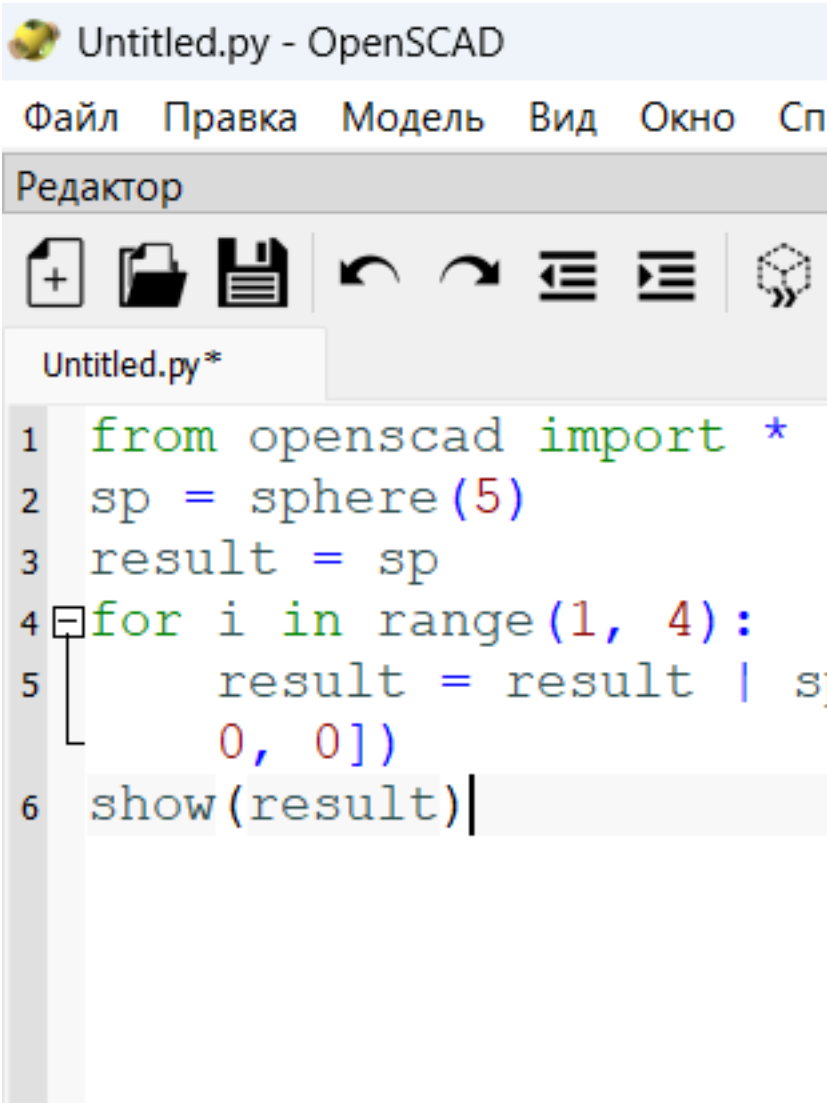

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 sp = sphere(5)
3 result = sp
4 for i in range(1, 4):
5 result = result | s
0, 0])
6 show(result)

Рис. 18. Размещение нескольких объектов в for-цикле

Листинг 5: Вложенные translate()

```
cu = cube([10, 10, 10])
tr = cu.translate([10, 10, 10])
tr2 = tr.translate([10, 0, 0])
show(tr2)
```

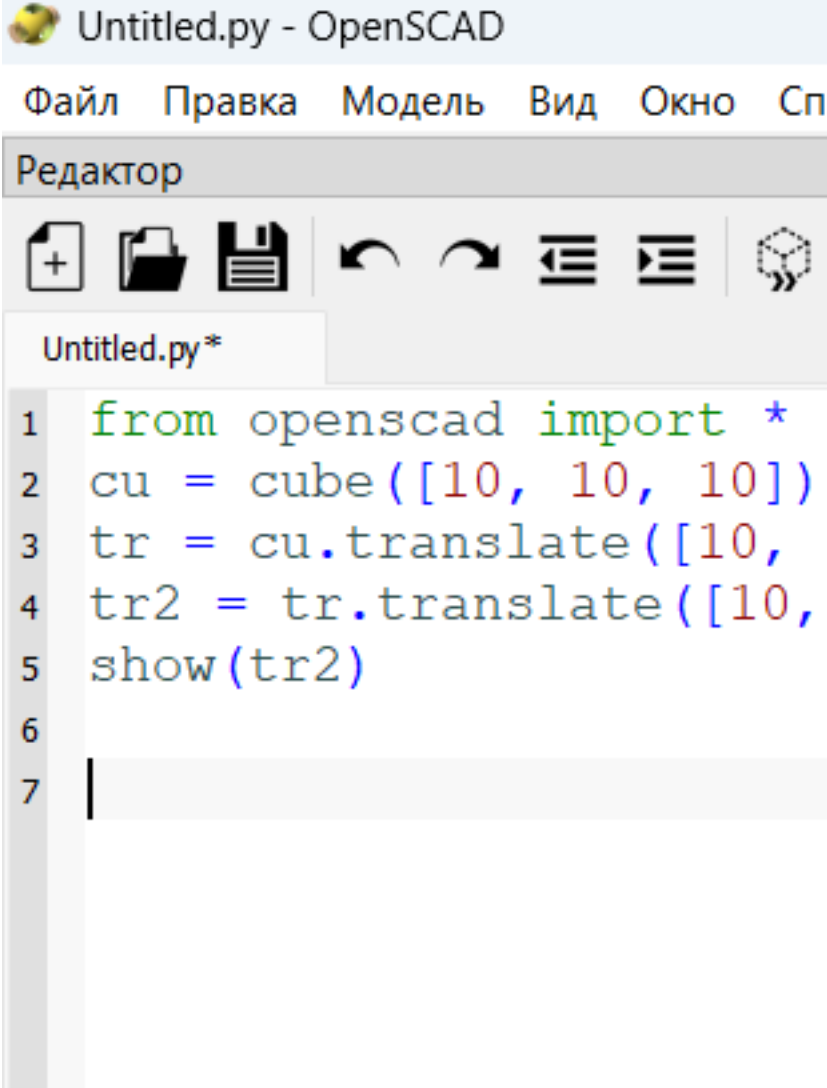

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cu = cube([10, 10, 10])
3 tr = cu.translate([10,
4 tr2 = tr.translate([10,
5 show(tr2)
6
7

Рис. 19. Вложенные translate()

Сначала объект смещается на 10 единиц по X,

Потом внутри блока – на 20 по Y.

Заключение

translate([X, Y, Z]) перемещает объект в 3D-пространстве. Важно учитывать порядок операций при сочетании с rotate(). Используется в циклах for для создания повторяющихся структур.

Масштабирование
(scale)
в
PythonSCAD(OpenSCAD)

Оператор scale() используется для изменения размеров объекта по осям X, Y и Z. Это позволяет увеличивать или уменьшать модель в нужных направлениях.

Синтаксис

scale([Sx, Sy, Sz]) объект;

Sx – масштабирование по оси X (ширина).

Sy – масштабирование по оси Y (глубина).

Sz – масштабирование по оси Z (высота).

Если Sx, Sy, Sz < 1 – объект уменьшается.

Если Sx, Sy, Sz > 1 – объект увеличивается.

Если Sx, Sy, Sz = 1 – объект остается без изменений.

Примеры использования

Листинг 1:Увеличение размера объекта в 2 раза

```
scale([2, 2, 2]) cube([10, 10, 10]);
```

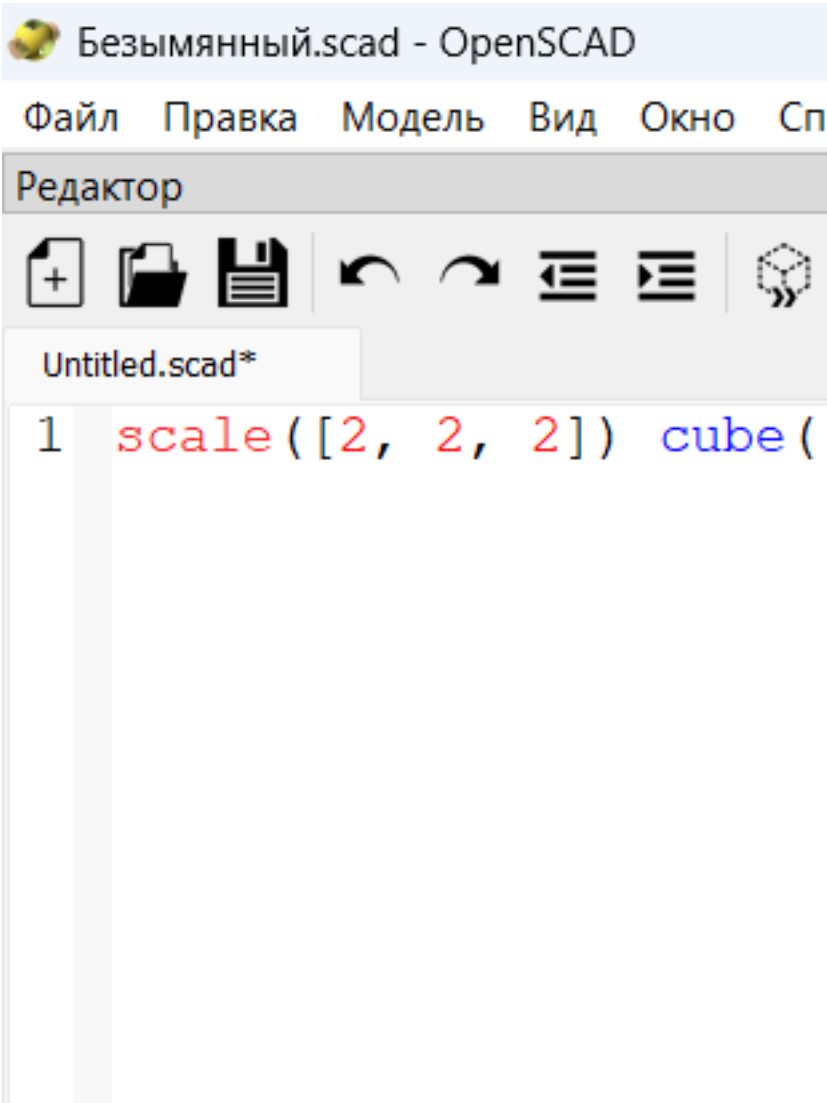

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 scale([2, 2, 2]) cube(

Рис. 20. Увеличение размера объекта в 2 раза
Исходный куб 10×10×10 увеличится до 20×20×20.
Листинг 2:Растяжение по одной оси
scale([2, 1, 1]) cylinder(h=10, r=5);

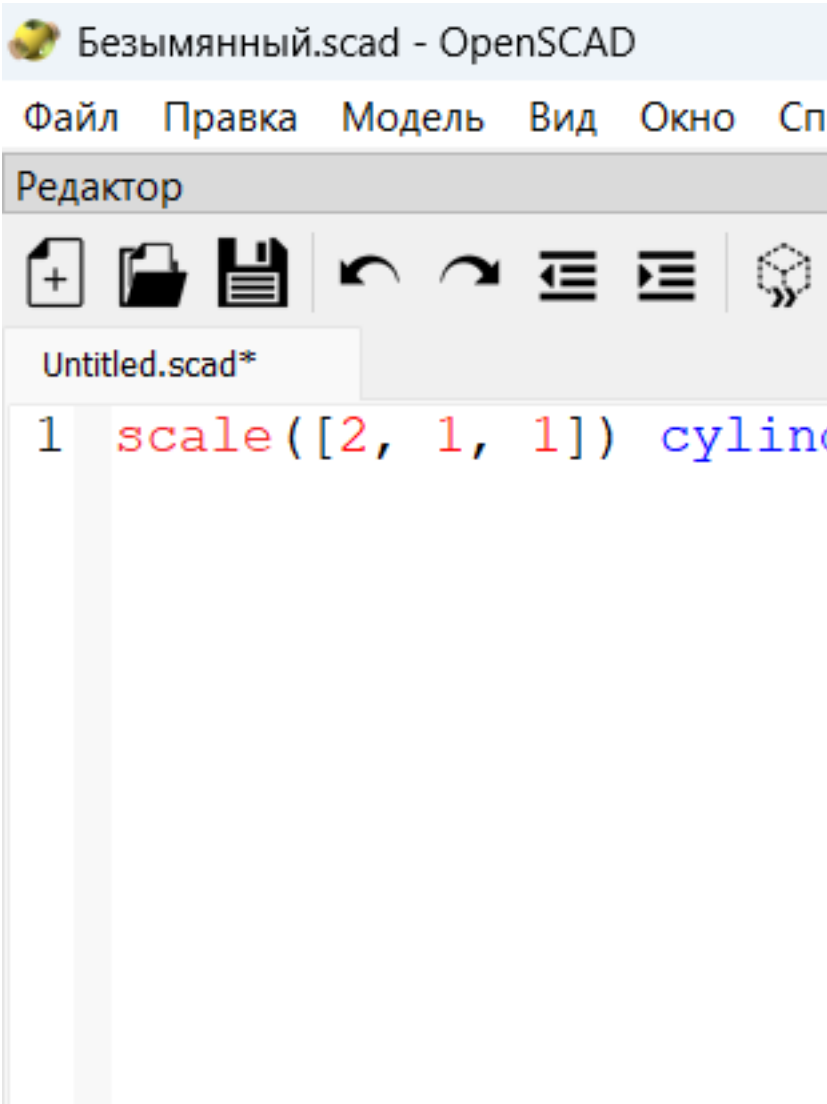

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 scale([2, 1, 1]) cylin

Рис. 21. Растяжение по одной оси

Цилиндр растянется в 2 раза по X, а другие оси останутся без изменений.

Листинг 3:Уменьшение по высоте

```
scale([1, 1, 0.5]) sphere(10);
```

Безымянный.scad - OpenSCAD

Файл Правка Модель Вид Окно Сп

Редактор







Untitled.scad*

```
1  scale([1, 1, 0.5]) sphe
```

Рис. 22. Уменьшение по высоте

Сфера сожмется в 2 раза по высоте, превращаясь в сплюснутую форму.

Листинг 4:Симметричное отражение (отрицательный масштаб)

```
scale([-1, 1, 1]) cube([10, 10, 10]);
```

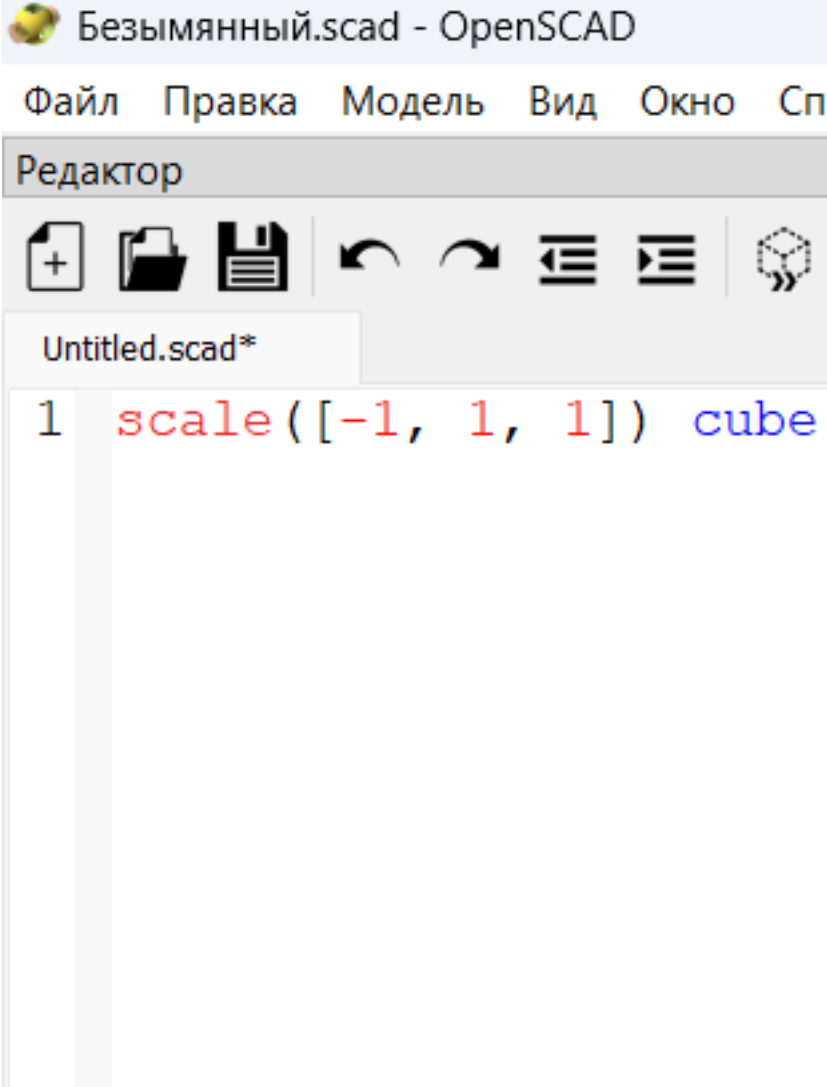

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.scad*
1 scale([-1, 1, 1]) cube

Рис. 23. Симметричное отражение (отрицательный масштаб)

Куб отразится относительно оси X (зеркальное отражение).

Если поставить -1 по другой оси, произойдет отражение по соответствующему направлению.

Листинг 5:Масштабирование в цикле

```
for (i = [1:0.5:2]) {
scale([i, i, i]) sphere(10);
}
```

Безымянный.scad - OpenSCAD

Файл Правка Модель Вид Окно Сп

Редактор

Untitled.scad*

```
1 for (i = [1:0.5:2]) {
2     scale([i, i, i]) sp
3 }
```

Рис. 24. Масштабирование в цикле

Создаст несколько сфер разного размера, увеличивающихся от 10 до 20 единиц.

Заключение

scale([X, Y, Z]) изменяет размер объекта по каждой оси отдельно. Можно увеличивать, уменьшать и отражать объекты. Используется в сочетании с translate() и rotate() для сложных конструкций.

Масштабирование (scale) в PythonSCAD(Python)

Оператор scale() используется для изменения размеров объекта по осям X, Y и Z. Это позволяет увеличивать или уменьшать модель в нужных направлениях.

Синтаксис

scale = Обьект * ([Sx, Sy, Sz])

Sx – масштабирование по оси X (ширина).

Sy – масштабирование по оси Y (глубина).

Sz – масштабирование по оси Z (высота).

Если Sx, Sy, Sz < 1 – объект уменьшается.

Если Sx, Sy, Sz > 1 – объект увеличивается.

Если Sx, Sy, Sz = 1 – объект остается без изменений.

Примеры использования

Листинг 1:Увеличение размера объекта в 2 раза

```
from openscad import *
cu = cube([10, 10, 10])
scale = cu * ([2, 2, 2])
```

show(scale)

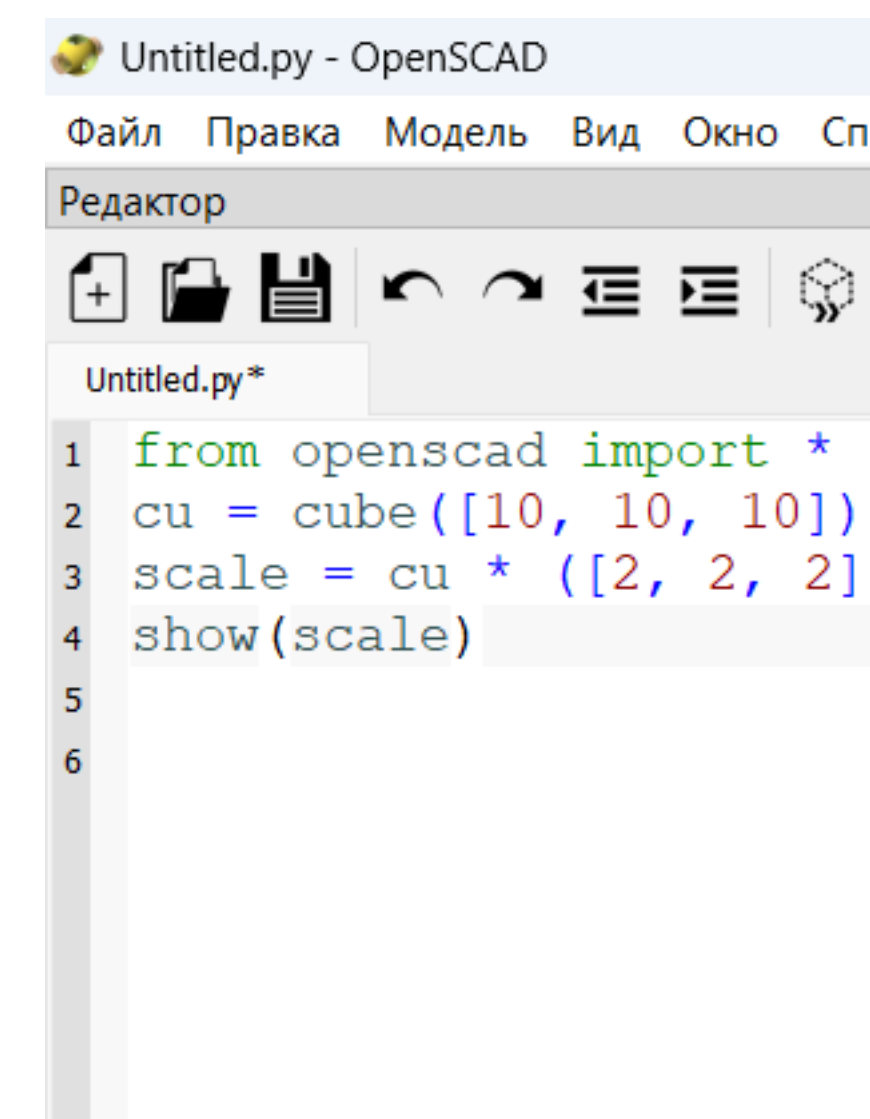

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cu = cube([10, 10, 10])
3 scale = cu * ([2, 2, 2]
4 show(scale)
5
6

Рис. 25. Увеличение размера объекта в 2 раза
Исходный куб 10×10×10 увеличится до 20×20×20.

Листинг 2:Растяжение по одной оси

```
from openscad import *
cy = cylinder(h=10, r=5)
scale = cy * ([2, 1, 1])
show(scale)
```

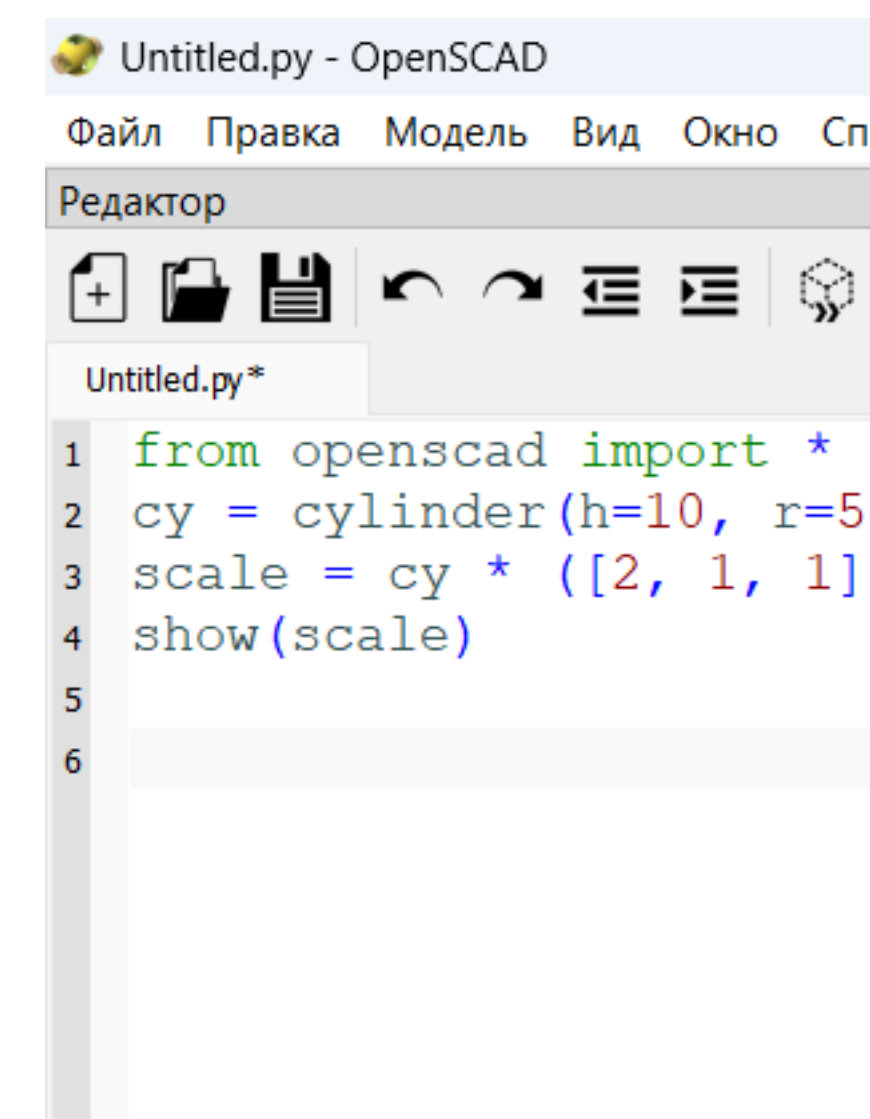

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cy = cylinder(h=10, r=5
3 scale = cy * ([2, 1, 1]
4 show(scale)
5
6

Рис. 26. Растяжение по одной оси

Цилиндр растянется в 2 раза по X, а другие оси останутся без изменений.

Листинг 3:Уменьшение по высоте

```
from openscad import *
sp = sphere(10)
scale = sp * ([1, 1, 0.5])
show(scale)
```

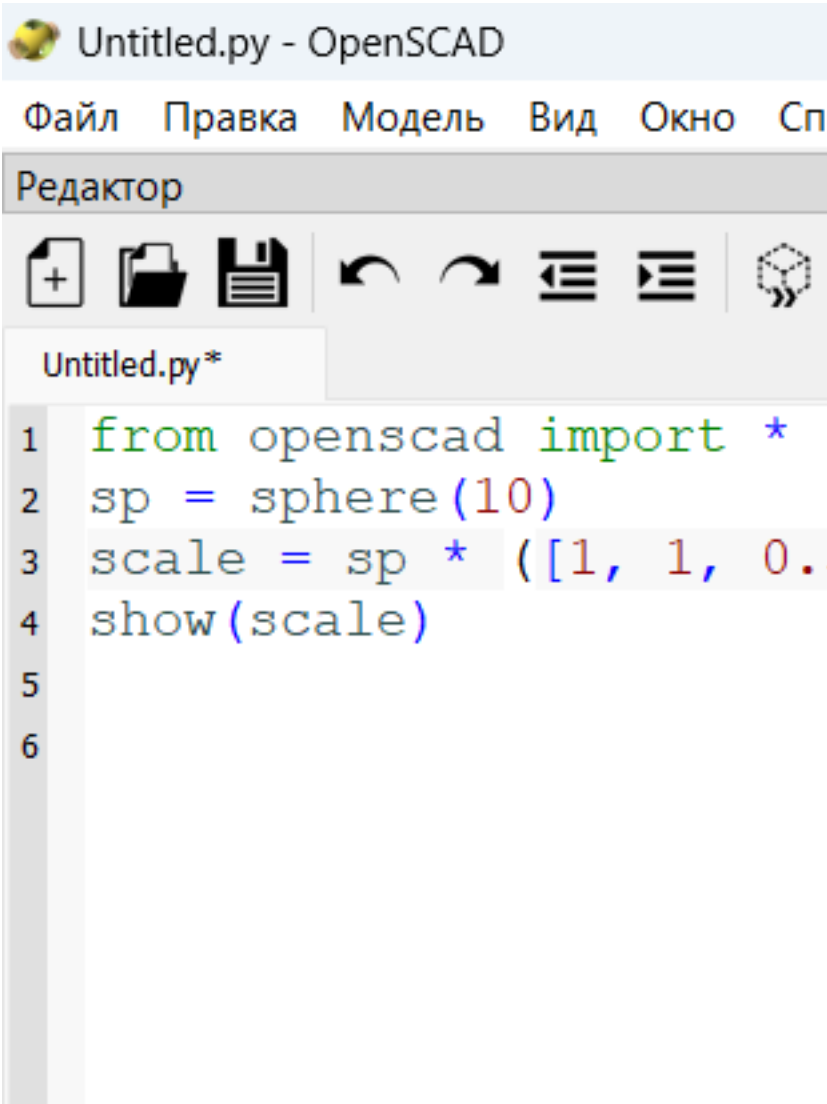

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 sp = sphere(10)
3 scale = sp * ([1, 1, 0.
4 show(scale)
5
6

Рис. 27. Уменьшение по высоте

Сфера сожмется в 2 раза по высоте, превращаясь в сплюснутую форму.

Листинг 4:Симметричное отражение (отрицательный масштаб)

```
cu = cube([10, 10, 10])
scale = cu * ([-1, 1, 1])
show(scale)
```

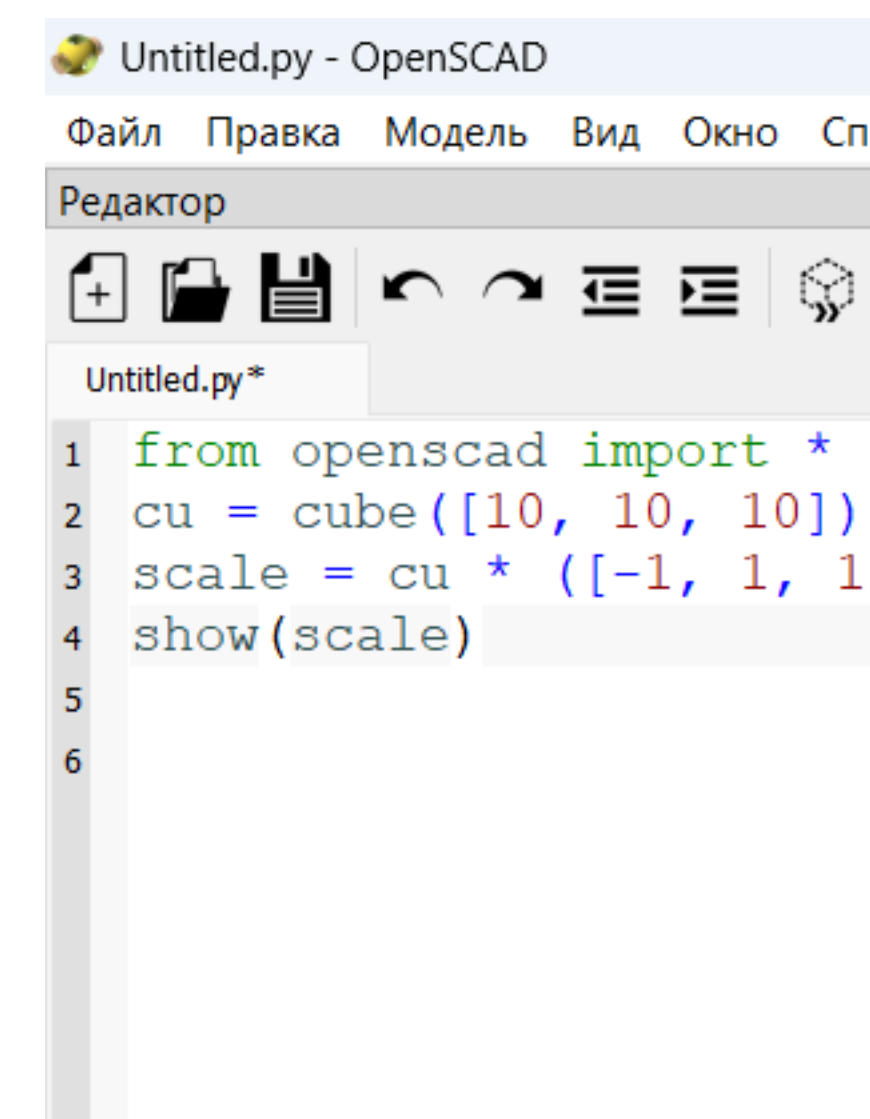

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Сп
Редактор
Untitled.py*
1 from openscad import *
2 cu = cube([10, 10, 10])
3 scale = cu * ([-1, 1, 1
4 show(scale)
5
6

Рис. 28. Симметричное отражение (отрицательный масштаб)

Куб отразится относительно оси X (зеркальное отражение).

Если поставить -1 по другой оси, произойдет отражение по соответствующему направлению.

Задача 1:Команды

Какая команда выполняет перемещение объекта в заданную точку?

1) translate()
2) position()
3) moveTo()
4) displace()
5) offset()

Задача 2:Передвижение кодом

Какой код сдвинет объект на 15 по X, -10 по Y и 5 по Z?

1) translate([15, -10, 5])
2) shift([15, -10, 5])
3) translate(15, -10, 5)
4) move([15, -10, 5])
5) position(15, -10, 5)

Задача 3:Масштабирование

Какая команда выполняет масштабирование объекта?

1) scale()
2) zoom()

3) expand()
4) stretch()
5) resize()

Задача 4:Масштабирование

Как уменьшить объект в 2 раза по всем осям?

1) scale([0.5, 0.5, 0.5])
2) scale(0.5, 0.5, 0.5)
3) scale([0.5])
4) resize([0.5, 0.5, 0.5])
5) shrink(0.5)

Задача 5:Смещение

Как задать поворот объекта на 90 градусов вокруг оси Y?

1) rotate([0,90,0])
2) rotate(a=90, v=[0,1,0])
3) rotateY(90)
4) turn([0,90,0])
5) spin(90, [0,1,0])

Задача 6:Отзеркаливание

Как выполнить зеркальное отражение объекта относительно плоскости XY?

1) mirror([0,0,1])
2) reflect([1,1,0])
3) mirror([0,1,0])
4) flip([0,1,0])
5) invert([0,0,1])

Задача 7:Задать размер объекта

Какая команда позволяет задать конкретные размеры объекта, а не масштабировать его?

1) resize()

2) stretch()

3) scale()

4) expand()

5) adjustSize()

Задача 8:Полупрозрачный объект

Как сделать объект полупрозрачным?

1) % cube([10,10,10])

2) transparency(0.5)

3) opacity(50)

4) fade(50)

5) alpha(0.5)

Задача 9:Невидимый объект

Как сделать объект невидимым, но оставшимся в коде?

1) *cube([10,10,10])

2) hide()

3) invisible()

4) disable()

5) remove()

Задача 10:Линейное Выдавливание

Какая команда создаёт линейное выдавливание 2D-фигуры?

1) linear_extrude()

2) push()

3) extrude()
4) stretch_extrude()
5) height_extrude()

Задача 11:Линейное Выдавливание

Как выдавить 2D-объект на 20 единиц вверх?
1) linear_extrude(height=20)
2) extrude(20)
3) push(20)
4) stretch(20)
5) lift(20)

Задача 12:Вращательное Выдавливание

Как сделать вращательное выдавливание 2D-объекта?
1) rotate_extrude()
2) twist_extrude()
3) spin_extrude()
4) circular_extrude()
5) revolve()

Задача 13:Оболочка

Какая команда позволяет создать оболочку объекта?
1) offset()
2) expand()
3) outline()
4) thicken()
5) hollow()

Задача 14:Сглаживание

Какая операция позволяет сгладить края объекта?

1) minkowski()
2) smooth()
3) round()
4) bevel()
5) soften()

Задача 15:Сглаживание

Как можно сделать скруглённые края у куба?

1) minkowski() { cube([10,10,10]) sphere(2) }
2) round_cube()
3) smooth_cube(10, 2)
4) soften_cube(10, 2)
5) bevel_cube([10,10,10], 2)

Задача 16:Многократное повторение

Какая команда выполняет многократное повторение объекта?

1) clone()
2) repeat()
3) for()
4) array()
5) loop()

Задача 1:Команды

Какая команда выполняет перемещение объекта в заданную точку?

1) rotated = .rotate()
2) c = coil
3) diff = .difference()

4) inter = .intersection()

Задача 2:Передвижение кодом

Какой код сдвинет объект на 15 по X, -10 по Y и 5 по Z?

1) tr = .translate([15, -10, 5])

2) translate([15, -10, 5])

3) c = coil([15, -10, 5])

4) rotated = .rotate([15, -10, 5])

Задача 3:Масштабирование

Какая команда выполняет масштабирование объекта?

1) Scale = * ()

2) tr = .translate()

3) c = coil()

4) translate([15, -10, 5])

Задача 4:Масштабирование

Как уменьшить объект в 2 раза по всем осям?

1) scale = * -2

2) Sc = scale * -2

3) scale = * 2

4) Sc = scale * 2

Задача 5:Смещение

Как задать поворот объекта на 90 градусов вокруг оси Y?

1) rotated = .rotate([0,90,0])

2) rotated =([0,90,0])

3) x = ([0,90,0])

4) ([0,90,0]) = x

Задача 6:Отзеркаливание

Как выполнить зеркальное отражение объекта относительно плоскости XY?

1) mr = .mirror([0,0,1])

2) mirror([0,0,1])

3) mr = ([0,0,1])

4) mr = mirror

Задача 7:Задать размер объекта

Какая команда позволяет задать конкретные размеры объекта, а не масштабировать его?

1) re = .resize([])

2) c = coil([])

3) re = ([])

4) resize([])

Задача 8:Смешенный код

Какой код смешивает вычитание и задание размеров объектов?

1) diff = re.difference(cy)

2) c = re.coil(cy)

3) fusion = union([re, cu])

4) inter = re.intercection(cu)

Задача 9:Смешенный код

Какой код смешивает пересечение и задание размеров объектов?

1) inter = re.intersection(cy)

2) c = cu.coil(re)

3) diff = re.difference(cy)

4) fusion = union([re, cu])

Задача 10:Линейное Выдавливание

Какая команда создаёт линейное выдавливание 2D-фигуры?

1) l=linear_extrude(square(),[[],[], [],[]]);

2) path_extrude()

3) linear_extrude

4) l = linear_extrude()

Задача 11:Линейное Выдавливание

Как выдавить 2D-объект на 20 единиц вверх?

1) p=path_extrude(square(1),[[0,0,0],[0,0,20]]);

2) rotate_ extrude(square(1),[[0,0,0],[0,0,20]]);

3) l=linear_extrude(square(1),[[0,0,0],[0,0,20]]);

4) e=extrude(square(1),[[0,0,0],[0,0,20]]);

Задача 12:Вращательное Выдавливание

Как сделать вращательное выдавливание 2D-объекта?

1) circle(3).right(10).rotate_extrude(v=[0,0,20],angle=600).sl

2) linear_extrude()

3) path_extrude()

4) extrude()

Задача 13:Оболочка

Какая команда позволяет создать оболочку объекта?

1) .offset()

2) .clone()

3) .array()

4) .loop()

Задача 14:Сглаживание

Какая операция позволяет сгладить края объекта?

1) .fillet

2) .edge

3) .core

4) .ore

Задача 15:Сглаживание

Как можно сделать скруглённые края у куба?

1) .fillet

2) .edge

3) .core

4) .ore

Задача 16:Многократное повторение

Какая команда выполняет многократное повторение объекта?

1) con = .clone()

2) f = for()

3) loo = loop()

4) ar = array()

Модуль 5: Основы параметризации

Переменные

Переменные в **PythonSCAD(OpenSCAD)** задаются с помощью оператора = и хранят значения различных типов: числа, строки, списки и другие. Важно помнить, что **PythonSCAD(OpenSCAD)** является декларативным языком, поэтому переменные неизменяемы после первого при-

сваивания.

Пример объявления переменных:

```
width = 10;
height = 20;
depth = 30;
cube([width, height, depth]);
```

Здесь переменные используются для задания размеров куба.

Особенности переменных

Листинг 1:Одноразовое присвоение

```
x = 5;
x = 10; // Это не изменяет x, значение остается 5
echo(x); // Выведет 5
```

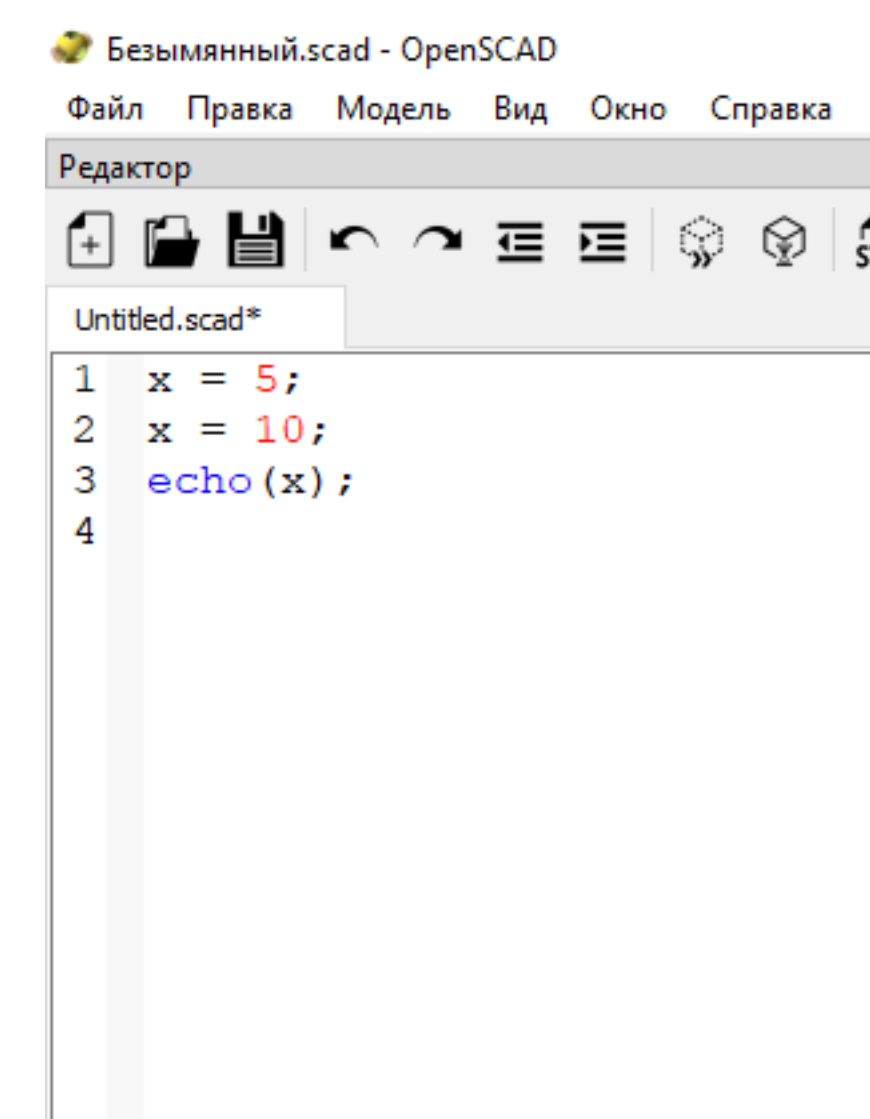

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.scad*
1 x = 5;
2 x = 10;
3 echo(x);
4

Рис. 1. Одноразовое присвоение

Листинг 2:Порядок выполнения

```
y = x + 5; // Используется текущее значение x (например, 5)
echo(y); // Выведет 10
```

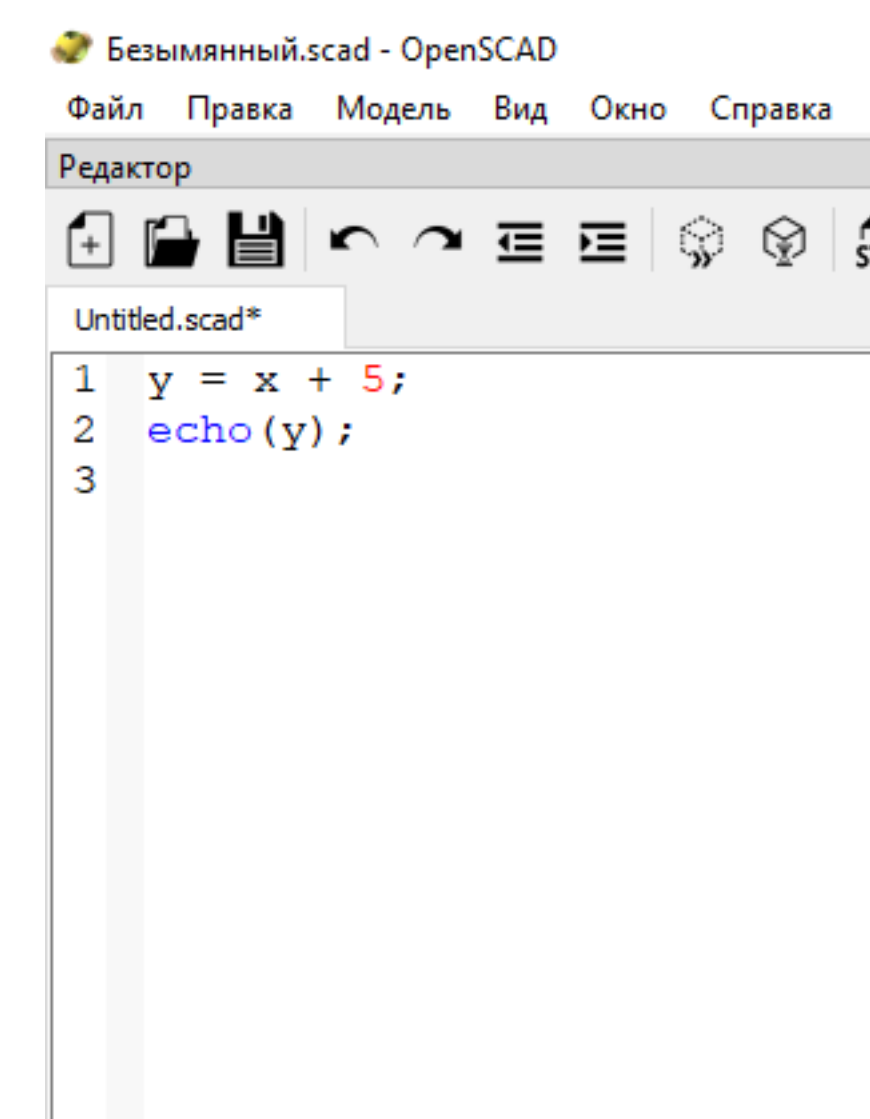

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.scad*
1 y = x + 5;
2 echo(y);
3

Рис. 2. Порядок выполнения

Константы

Хотя в **PythonSCAD(OpenSCAD)** отсутствует специальный механизм для объявления констант, часто используют соглашение: писать имена констант заглавными буквами (PI, GOLDEN_RATIO), чтобы отличать их от обычных переменных.

Пример констант:

```
PI = 3.141592;
GOLDEN_RATIO = 1.618;
RADIUS = 50;
circle(RADIUS);
```

Вычисляемые переменные

Переменные могут зависеть от других переменных, что делает возможным создание сложных моделей.

Листинг 1:Вычисляемые переменные

```
base_diameter = 40;
height = base_diameter / 2; // Высота зависит от диаметра
cylinder(d=base_diameter, h=height);
```

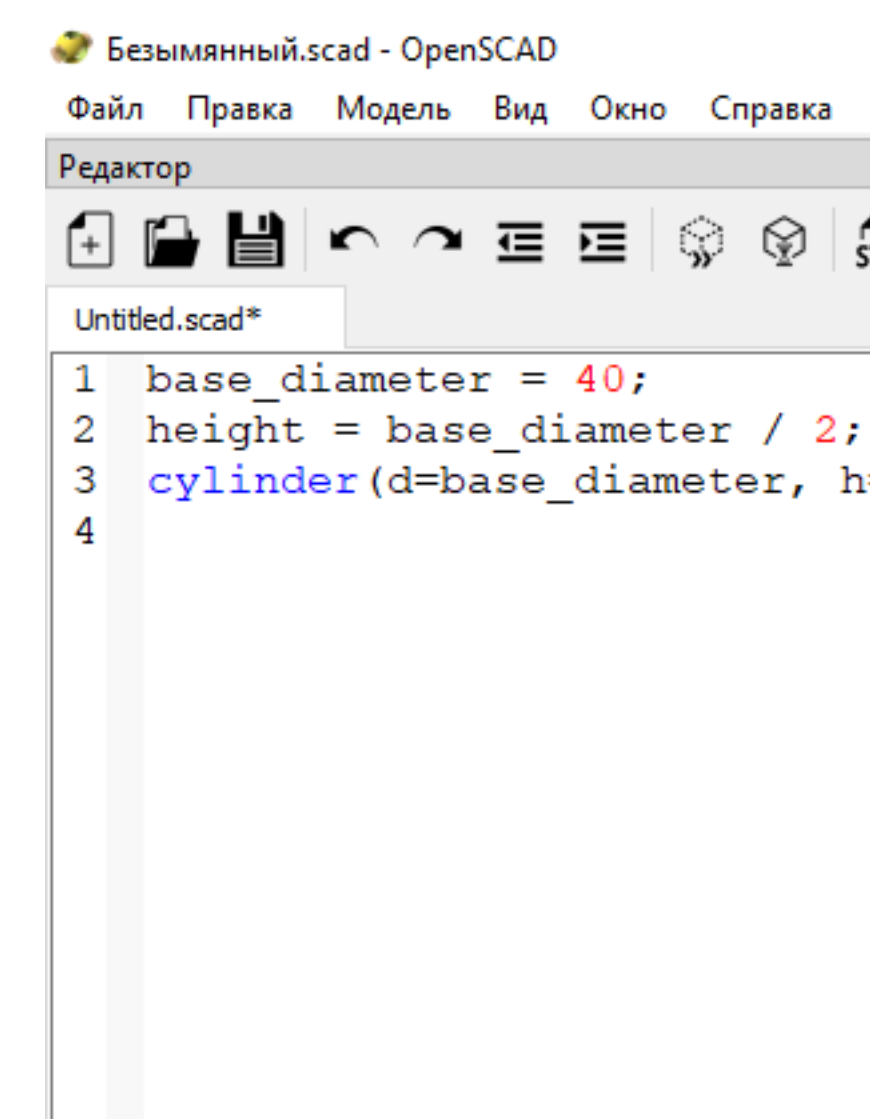

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.scad*
1 base_diameter = 40;
2 height = base_diameter / 2;
3 cylinder(d=base_diameter, h
4

Рис. 3. Вычисляемые переменные

Локальные переменные (let)

Для ограничения области видимости переменной используется конструкция let().

```
openscad
let(r = 10) {
circle(r);
}
```

Здесь переменная r доступна только внутри блока let().

Отладка с помощью echo

Функция echo() полезна для вывода значений переменных в процессе отладки.

```
a = 5;
b = a * 2;
echo("Значение b:", b); // Выведет: ECHO: "Значение b:", 10
```

Выводы:

1) Переменные в PythonSCAD(OpenSCAD)неизменяемы после присвоения.

2) Константы обозначаются прописными буквами.

3) Локальные переменные создаются с помощью let().

4) Функция echo() удобна для отладки.

Переменные в **PythonSCAD(Python)** задаются с помощью оператора и хранят значения различных типов:

числа, строки, списки и другие. Важно помнить, что **PythonSCAD(Python)** является декларативным языком, поэтому переменные неизменяемы после первого присваивания.

Пример объявления переменных:

```
width = 10;
height = 20;
depth = 30;
c = cube([width, height, depth]);
show(c)
```

Здесь переменные используются для задания размеров куба.

Особенности переменных

Листинг 1:Одноразовое присвоение

```
x = 5
x = 10 // Это не изменяет x, значение остается 5
cu = cube(x)
show(cu) // Выведет куб размером 5
```

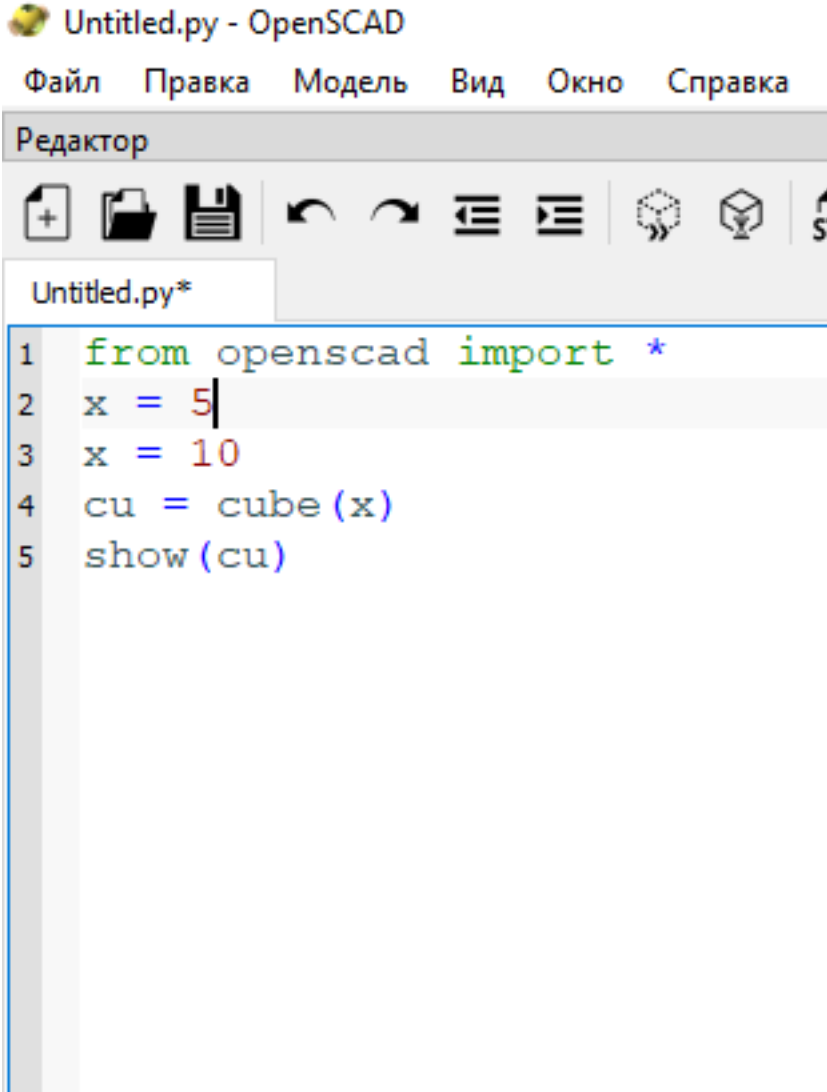

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.py*
1 from openscad import *
2 x = 5
3 x = 10
4 cu = cube(x)
5 show(cu)

Рис. 4. Одноразовое присвоение

Листинг 2:Порядок выполнения

```
x = 5
x = 10
y = x + 5 // Используется текущее значение x (например, 5)
cu = cube(y)
show(cu) // Выведет куб рамзером 10
```

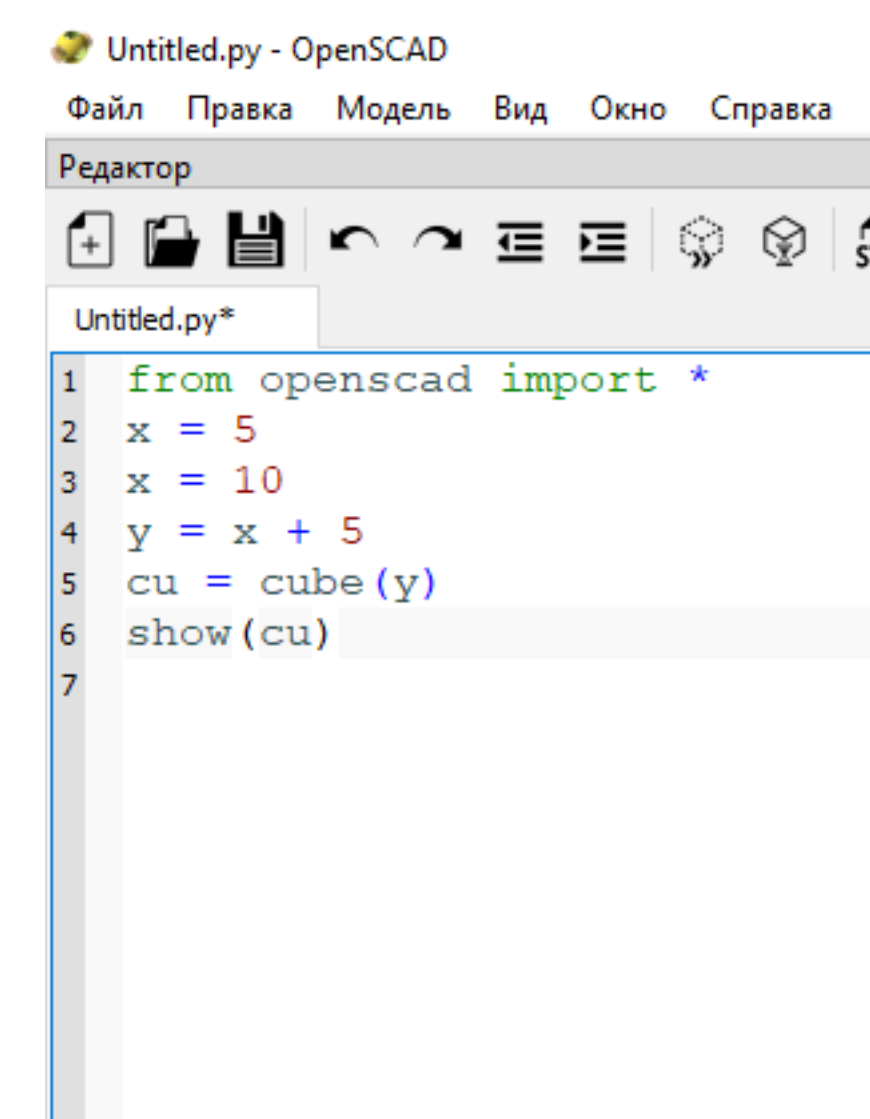

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.py*
1 from openscad import *
2 x = 5
3 x = 10
4 y = x + 5
5 cu = cube(y)
6 show(cu)
7

Рис. 5. Порядок выполнения

Константы

Хотя в PythonSCAD(Python) отсутствует специальный механизм для объявления констант, часто используют соглашение: писать имена констант заглавными буквами (PI, GOLDEN_RATIO), чтобы отличать их от обычных переменных.

```
PI = 3.141592;
GOLDEN_RATIO = 1.618;
RADIUS = 50;
circle(RADIUS);
```

Вычисляемые переменные

Переменные могут зависеть от других переменных, что делает возможным создание сложных моделей.

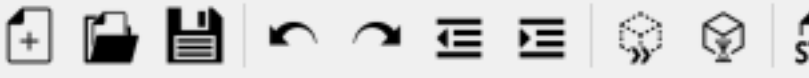
Листинг 3: Вычисляемые переменные

```
base_diameter = 40;
height = base_diameter / 2; // Высота зависит от диаметра
cy = cylinder(d=base_diameter, h=height);
show(cy)
```

Untitled.py - OpenSCAD

Файл Правка Модель Вид Окно Справка

Редактор



Untitled.py*

```
1  from openscad import *
2  base_diameter = 40;
3  height = base_diameter / 2;
4  cy = cylinder(d=base_diamete
       height);
5  show(cy)
6
```

Рис. 6. Вычисляемые переменные

Локальные переменные (let)

Для ограничения области видимости переменной используется конструкция example().

Листинг 4: Локальные переменные (let)

```
def example():
x = 10
y = 20
return x + y
result = cube(example())
show(result)
```

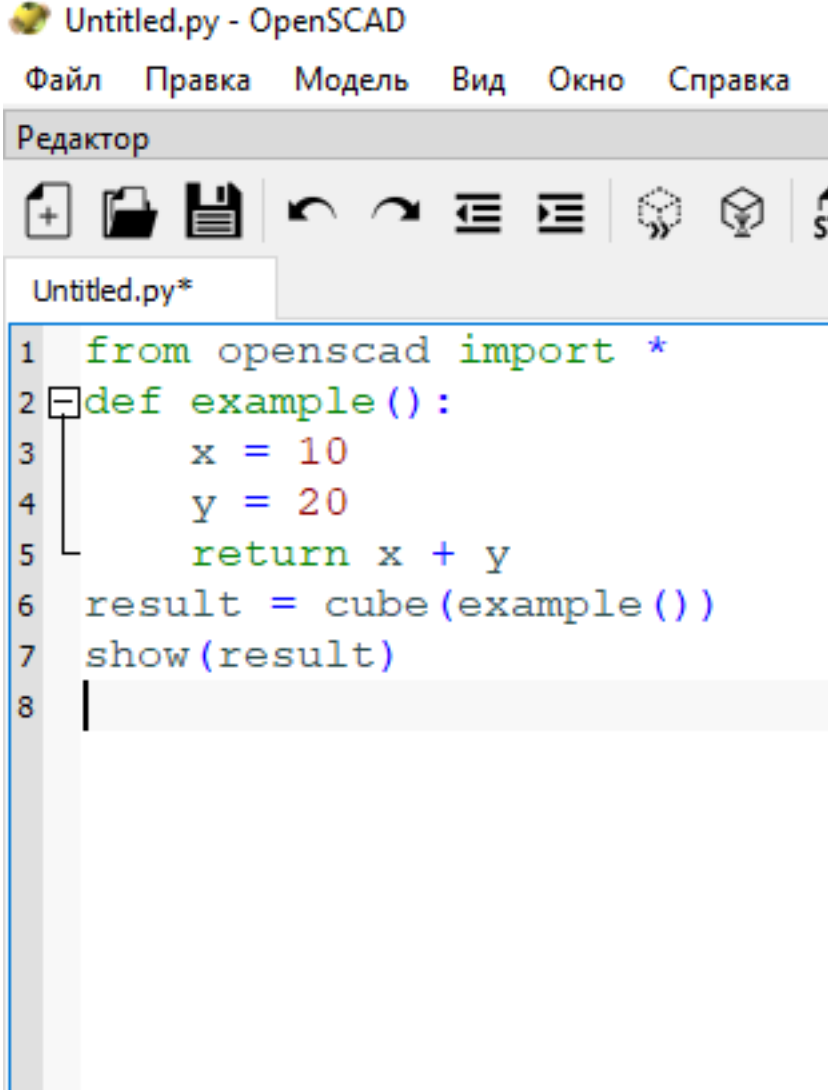

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.py*
1 from openscad import *
2 def example():
3 x = 10
4 y = 20
5 return x + y
6 result = cube(example())
7 show(result)
8

Рис. 7. Локальные переменные (let)

Здесь переменная r доступна только внутри блока example().

Функция show() обязательна для вывода значений переменных в процессе отладки.

```
from openscad import *
a = 5
b = a * 2
cu = cube(b)
show(cu) // Выведет куб который имеет параметры b
```

Выводы:

Переменные в PythonSCAD(Python)неизменяемы после присвоения. Константы обозначаются прописными буквами. Локальные переменные создаются с помощью let().

Функции и аргументы в PythonSCAD(OpenSCAD)

PythonSCAD(OpenSCAD) позволяет использовать функции для вычисления значений и работы с геометрией. Эти функции действуют подобно математическим функциям и не способны изменять сцену напрямую, поскольку **OpenSCAD** является декларативным языком.

Листинг 1:Определение функций

Функции определяются с использованием ключевого слова function, принимают аргументы и возвращают результат. Они полезны для упрощения и повторной реализации вы-

числений.

```
function square(x) = x * x;
echo(square(5)); // ECHO: 25
```

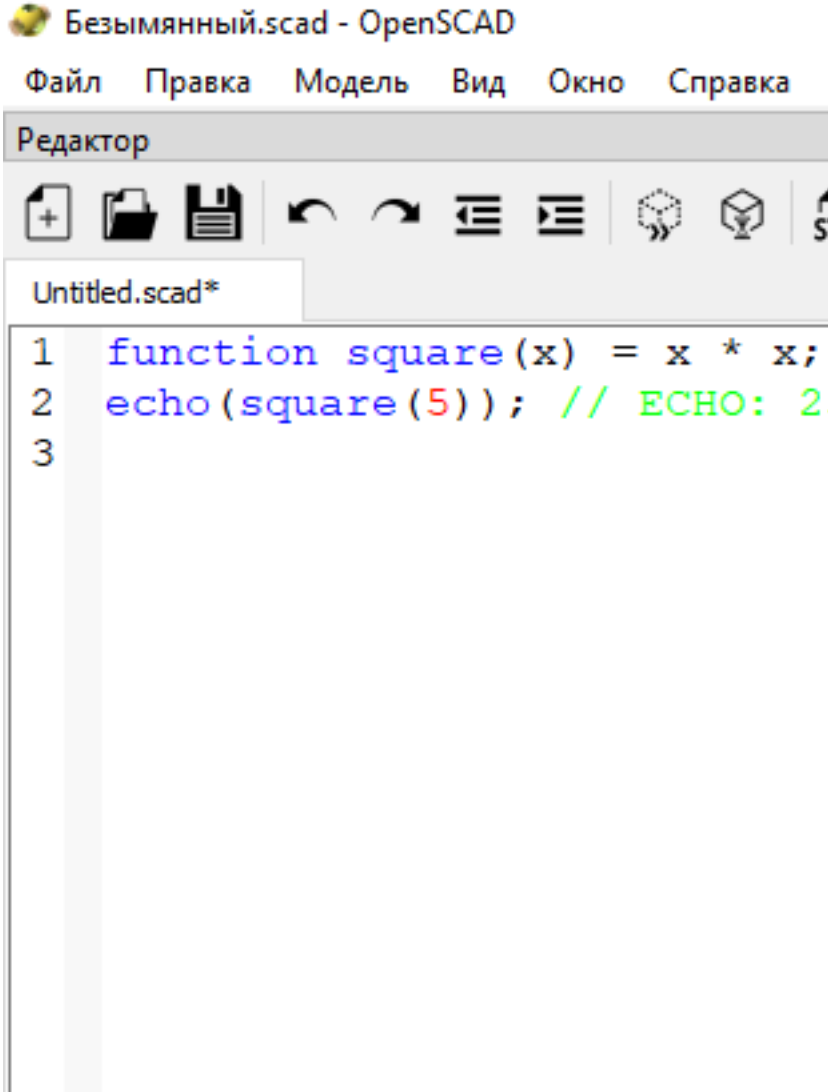

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.scad*
1 function square(x) = x * x;
2 echo(square(5)); // ECHO: 2
3

Рис. 8. Определение функций

Эта функция square(x) возводит переданное ей число в квадрат.

Листинг 2:Функции с несколькими аргументами

```
function sum(a, b) = a + b;
echo(sum(10, 20)); // ECHO: 30
```

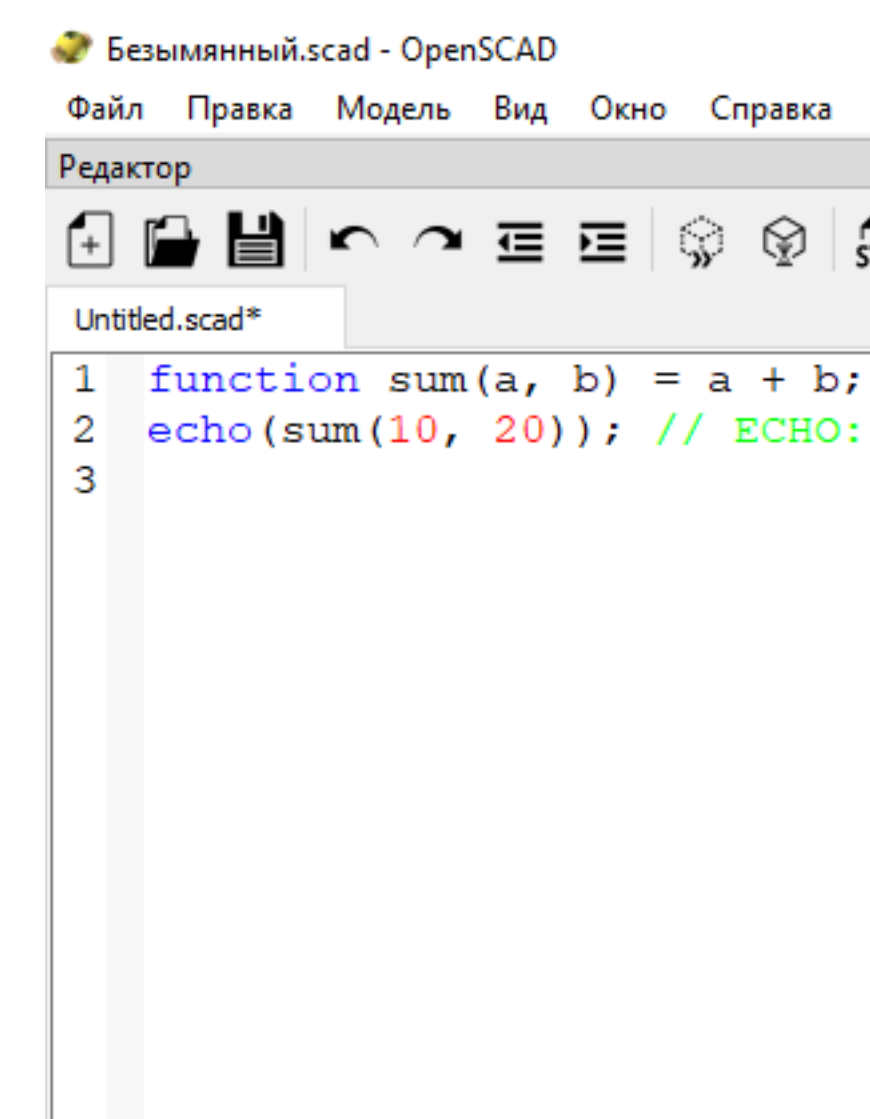

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.scad*
1 function sum(a, b) = a + b;
2 echo(sum(10, 20)); // ECHO:
3

Рис. 9. Функции с несколькими аргументами

Листинг 3:Условные функции (тернарный оператор)

```
function abs_value(x) = x >= 0 ? x : -x;
echo(abs_value(-7)); // ECHO: 7
```

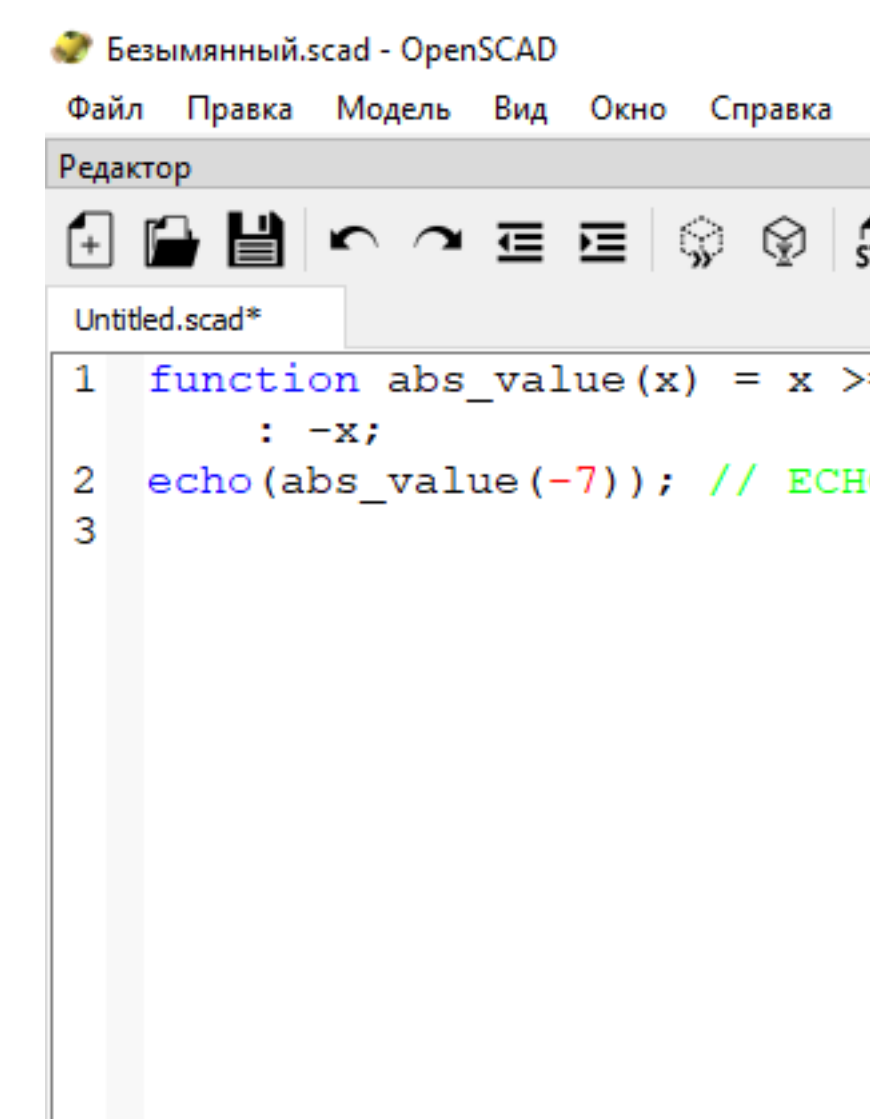

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.scad*
1 function abs_value(x) = x >
: -x;
2 echo(abs_value(-7)); // ECH
3

Рис. 10. Условные функции (тернарный оператор)

Функция abs_value(x) возвращает абсолютное значение переданного числа.

Листинг 4:Рекурсивные функции

OpenSCAD поддерживает рекурсию, хотя циклы отсутствуют.

Факториал

```
function factorial(n) = (n <= 1) ? 1 : n * factorial(n – 1);
echo(factorial(5)); // ECHO: 120
```

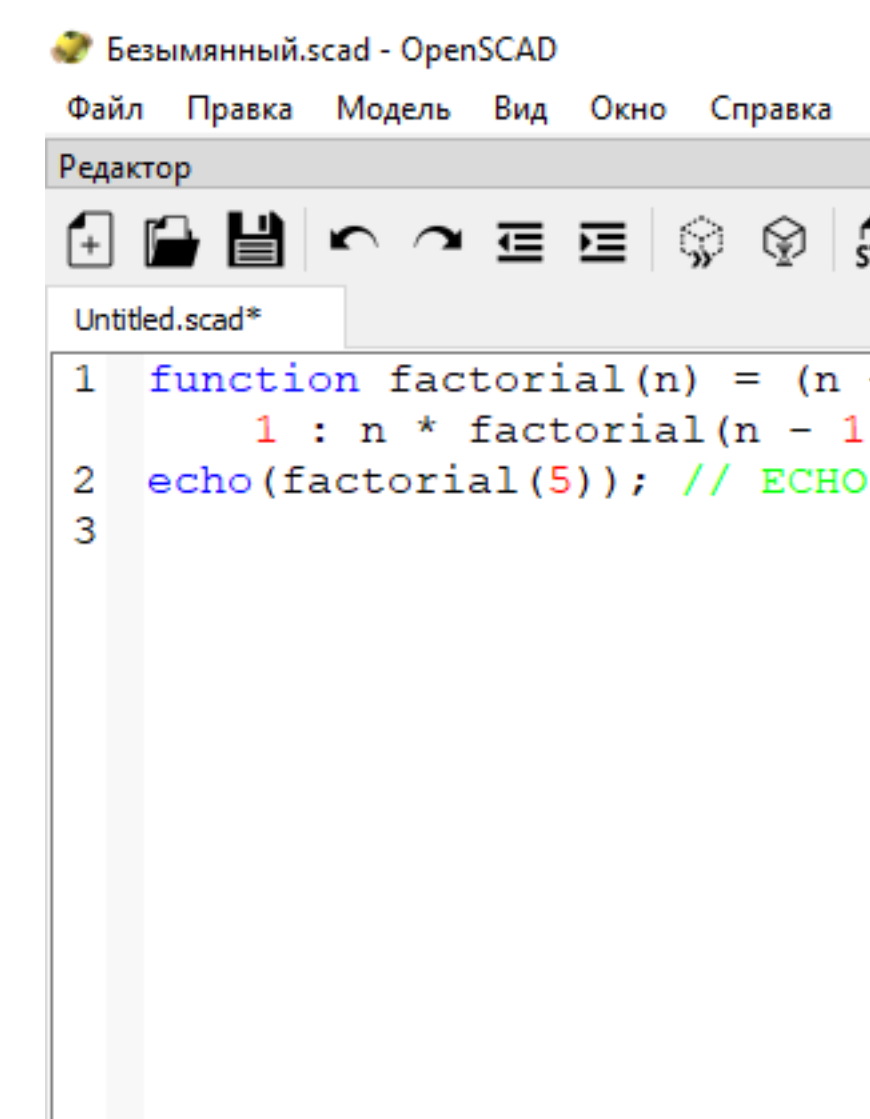

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.scad*
1 function factorial(n) = (n
1 : n * factorial(n - 1
2 echo(factorial(5)); // ECHO
3

Рис. 11. Рекурсивные функции

Листинг 5: Работа с массивами

```
function sum_list(lst) = (len(lst) == 0) ? 0 : lst[0] + sum_list([for (i = 1; i < len(lst); i = i + 1) lst[i]]);
echo(sum_list([1, 2, 3, 4, 5])); // ECHO: 15
```

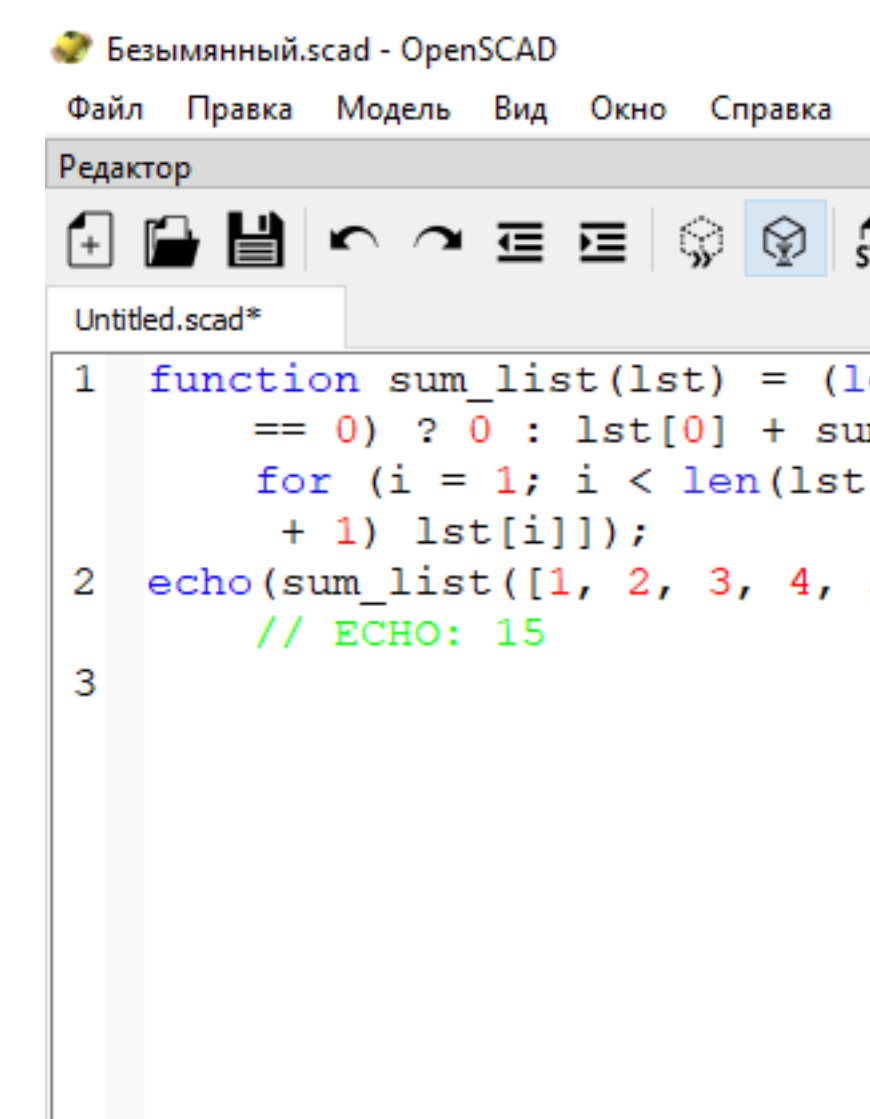

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.scad*
1 function sum_list(lst) = (l
== 0) ? 0 : lst[0] + su
for (i = 1; i < len(lst
+ 1) lst[i]]);
2 echo(sum_list([1, 2, 3, 4,
// ECHO: 15
3

Рис. 12. Работа с массивами

Листинг 6:Передача функций как аргумента

```
function operation(type, x) = (type == "square") ? x * x : (type
== "double") ? x * 2 : x;
echo(operation("square", 4)); // ECHO: 16
echo(operation("double", 4)); // ECHO: 8
```

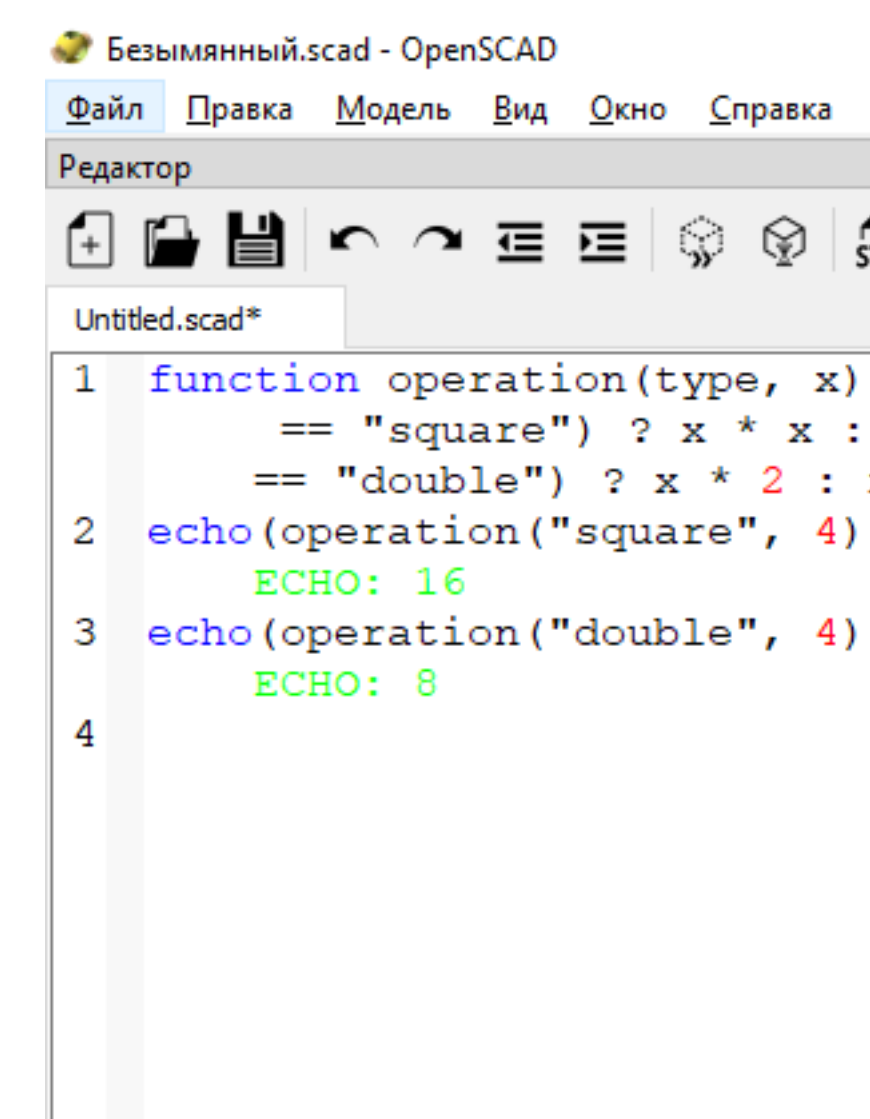

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.scad*
1 function operation(type, x)
== "square") ? x * x :
== "double") ? x * 2 :
2 echo(operation("square", 4)
ECHO: 16
3 echo(operation("double", 4)
ECHO: 8
4

Рис. 13. Передача функций как аргумента

Листинг 7:Использование функций в модулях

```
function diameter(radius) = radius * 2;
module custom_cylinder(r) {
cylinder(d = diameter(r), h = 10);

}
custom_cylinder(15);
```

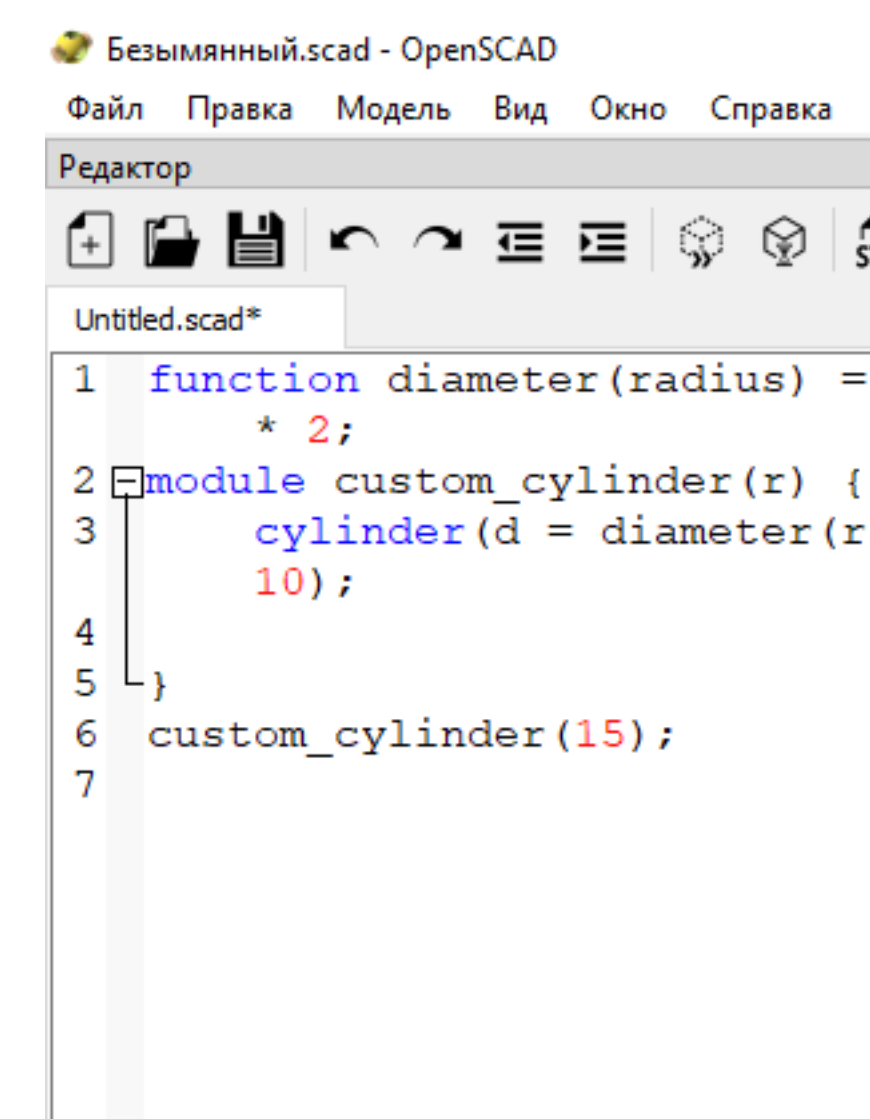

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.scad*
1 function diameter(radius) =
* 2;
2 module custom_cylinder(r) {
3 cylinder(d = diameter(r
10);
4
5 }
6 custom_cylinder(15);
7

Рис. 14. Использование функций в модулях

Функция diameter(r) вычисляет диаметр цилиндра, который затем передается в модуль custom_cylinder.

Заключение

Функции возвращают значения, но не влияют непосредственно на геометрию. Поддерживаются множественные аргументы, условные выражения, работа с массивами и рекурсия. Функции идеально подходят для параметрической настройки моделей.

Теперь давайте рассмотрим практический пример создания шестеренки с использованием функций.

Параметрическая шестеренка

Функция рассчитывает радиус зубца, а модуль создает трехмерную модель шестеренки.

Листинг 1:Рассчёт функции шестерни

```
// Функция для расчета радиуса зубцов шестеренки
function gear_radius(base_r, teeth, tooth_height) =
base_r + tooth_height * (teeth > 0 ? 1 : 0);
// Модуль для создания шестеренки
module gear(base_r, teeth, tooth_height) {

difference() {
// Основной диск шестеренки
circle(r = gear_radius(base_r, teeth, tooth_height), $fn =
100);
```

```
// Создание выемок для зубцов
for(i = [0:teeth-1]) {
angle = i * (360 / teeth);
translate([base_r * cos(angle), base_r * sin(angle)])
rotate(angle)
square([tooth_height * 2, base_r * 0.2], center = true);
}
}
}
// Вызов модуля шестеренки с заданными параметрами
gear(base_r = 20, teeth = 12, tooth_height = 3);
```

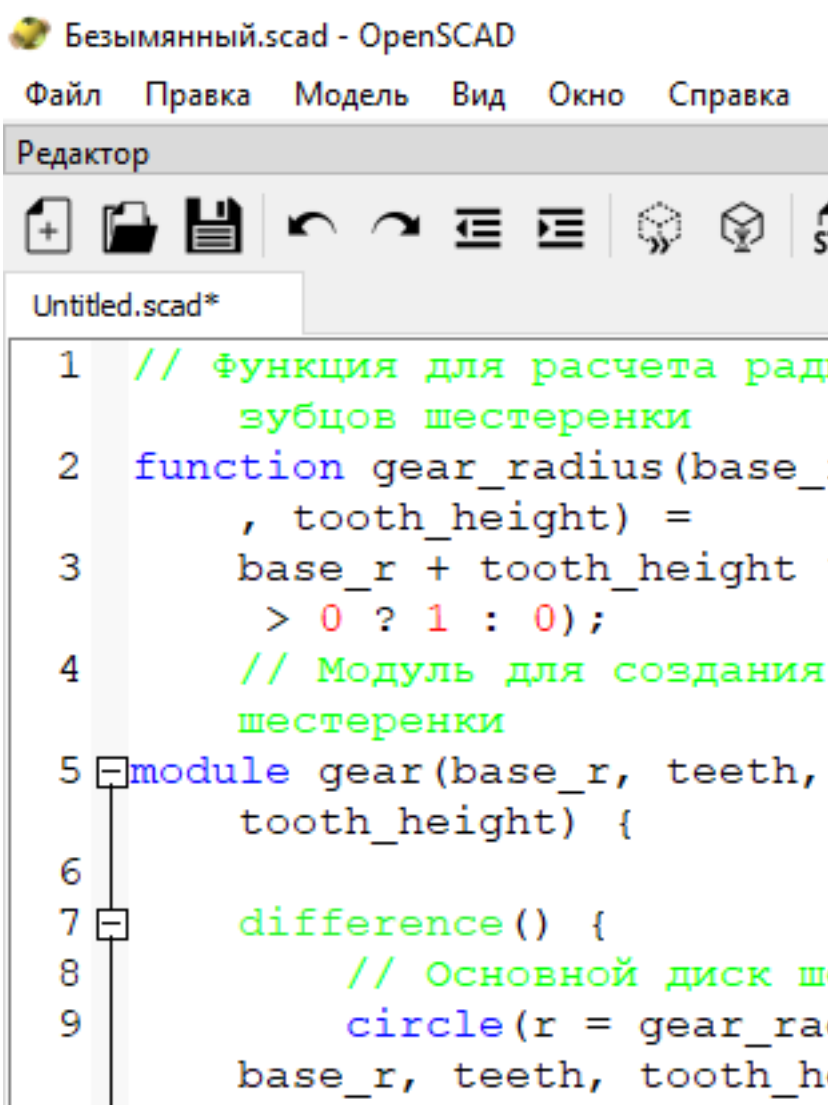

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.scad*
1 // Функция для расчета рад
зубцов шестеренки
2 function gear_radius(base_
, tooth_height) =
3 base_r + tooth_height
> 0 ? 1 : 0);
4 // Модуль для создания
шестеренки
5 module gear(base_r, teeth,
tooth_height) {
6
7 difference() {
8 // Основной диск ш
9 circle(r = gear_ra
base_r, teeth, tooth_h

Рис. 15. Рассчёт функции шестерни

Описание работы шестеренки

Функция gear_radius вычисляет максимальный радиус шестеренки, включая выступающие зубья.

Модуль gear:

– Создается основной диск шестеренки.

– С помощью цикла for создаются выемки для зубцов.

Шестеренку можно вызвать с параметрами: gear(20, 12, 3), где задаются радиус основания, количество зубцов и высота зуба.

Улучшение шестеренки

Добавим возможность задавать диаметр центрального отверстия.

Листинг 2:Возможность задавать диаметр центрального отверстия

```
module gear(base_r, teeth, tooth_height, hole_r = 5) {
difference() {
// Основной диск шестеренки
circle(r = gear_radius(base_r, teeth, tooth_height), $fn = 100);
// Вырезание зубцов

for(i = [0:teeth-1]) {
angle = i * (360 / teeth);
translate([base_r * cos(angle), base_r * sin(angle)])
rotate(angle)
```

```
square([tooth_height * 2, base_r * 0.2], center = true);
}
// Центральное отверстие
circle(r = hole_r, $fn = 50);
}
}
// Пример вызова шестеренки с отверстием
gear(base_r = 20, teeth = 16, tooth_height = 3, hole_r = 6);
```

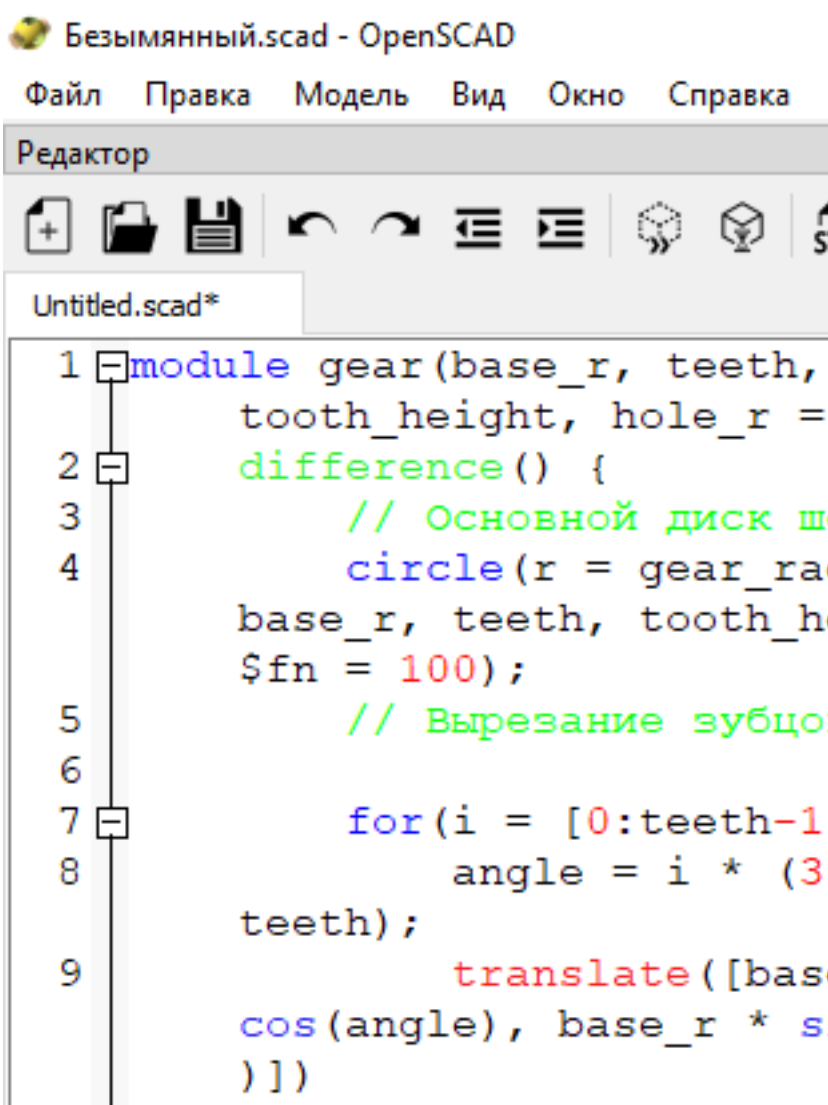

Безымянный.scad - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.scad*
1 module gear(base_r, teeth,
tooth_height, hole_r =
2 difference() {
3 // Основной диск ш
4 circle(r = gear_ra
base_r, teeth, tooth_h
$fn = 100);
5 // Вырезание зубцо
6
7 for(i = [0:teeth-1
8 angle = i * (3
teeth);
9 translate([bas
cos(angle), base_r * s
)])

Рис. 16. Возможность задавать диаметр центрального отверстия

Теперь можно регулировать такие параметры, как количество зубцов, высоту зубьев и размер центрального отверстия.

Приведенный пример демонстрирует, как функции позволяют вычислить необходимые параметры для создания сложных геометрических объектов. Модули же создают сами объекты на основе этих параметров.

Функции и аргументы в PythonSCAD(Python)

PythonSCAD(Python) позволяет использовать функции для вычисления значений и работы с геометрией. Эти функции действуют подобно математическим функциям и не способны изменять сцену напрямую, поскольку OpenSCAD является декларативным языком.

Листинг 1:Простая функция

```
Простая функция
def square(x):
return x * x
result = square(5)
print(result)
```

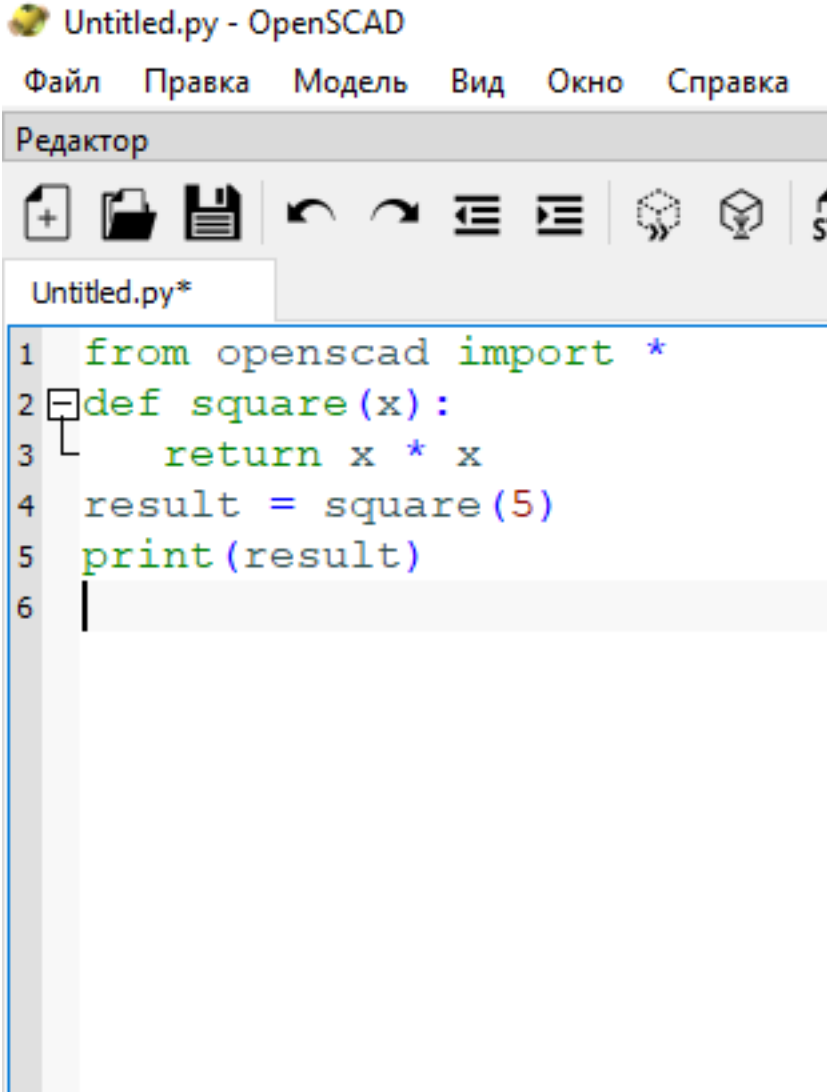

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.py*
1 from openscad import *
2 def square(x):
3 return x * x
4 result = square(5)
5 print(result)
6

Рис. 17. Простая функция

Эта функция square(x) возводит переданное ей число в квадрат.

Листинг 2:Функции с несколькими аргументами

```
def sum(a, b):
return a + b
result = sum(10, 20)
print(result)
```

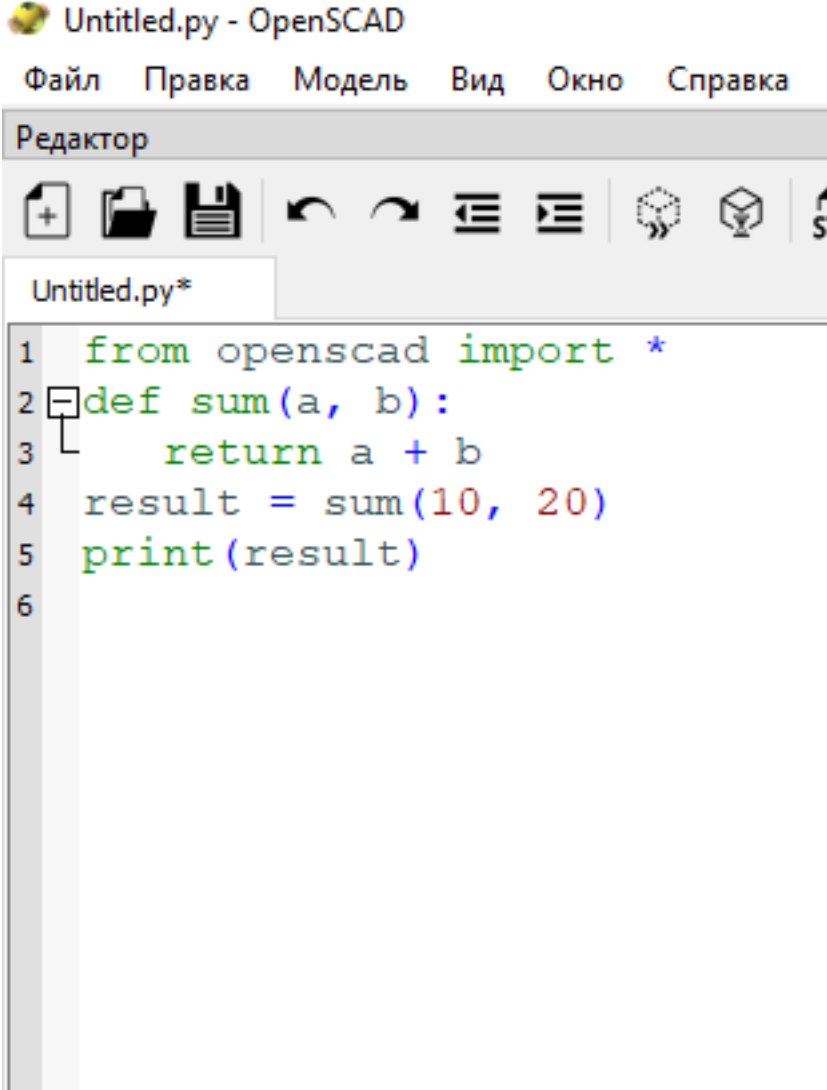

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.py*
1 from openscad import *
2 def sum(a, b):
3 return a + b
4 result = sum(10, 20)
5 print(result)
6

Рис. 18. Функции с несколькими аргументами

Листинг 3:Условные функции (тернарный оператор)

```
def absolute_value(x):
if x < 0:
return -x
return x
result = absolute_value(-7)
print(result)
```

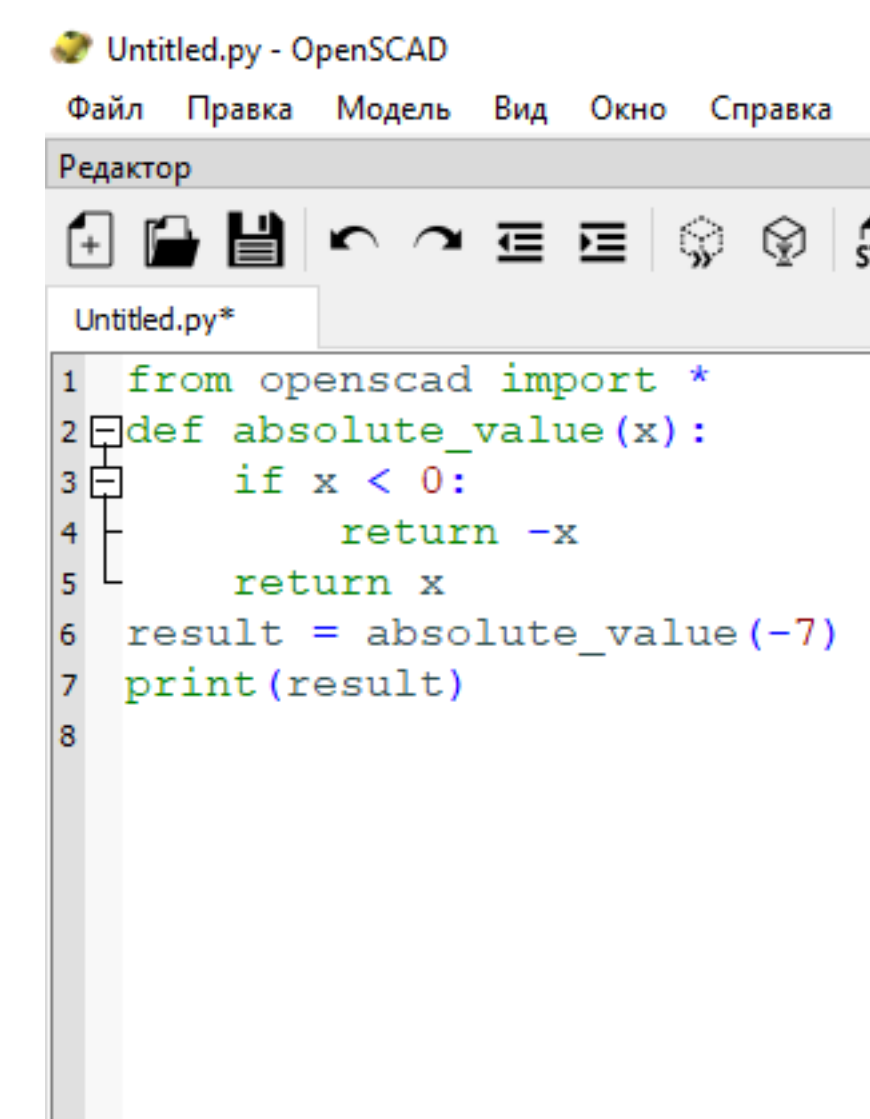

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.py*
1 from openscad import *
2 def absolute_value(x):
3 if x < 0:
4 return -x
5 return x
6 result = absolute_value(-7)
7 print(result)
8

Рис. 19. Условные функции (тернарный оператор)

Функция abs_value(x) возвращает абсолютное значение переданного числа.

Листинг 4:Рекурсивные функции

Факториал

```
def factorial(n):
if n < 0:
return "Факториал не определён для отрицательных чисел"
elif n == 0 or n == 1:
return 1
else:
result = 1
for i in range(2, n + 1):
result *= i
return result
result = factorial(5)
print(result)
```

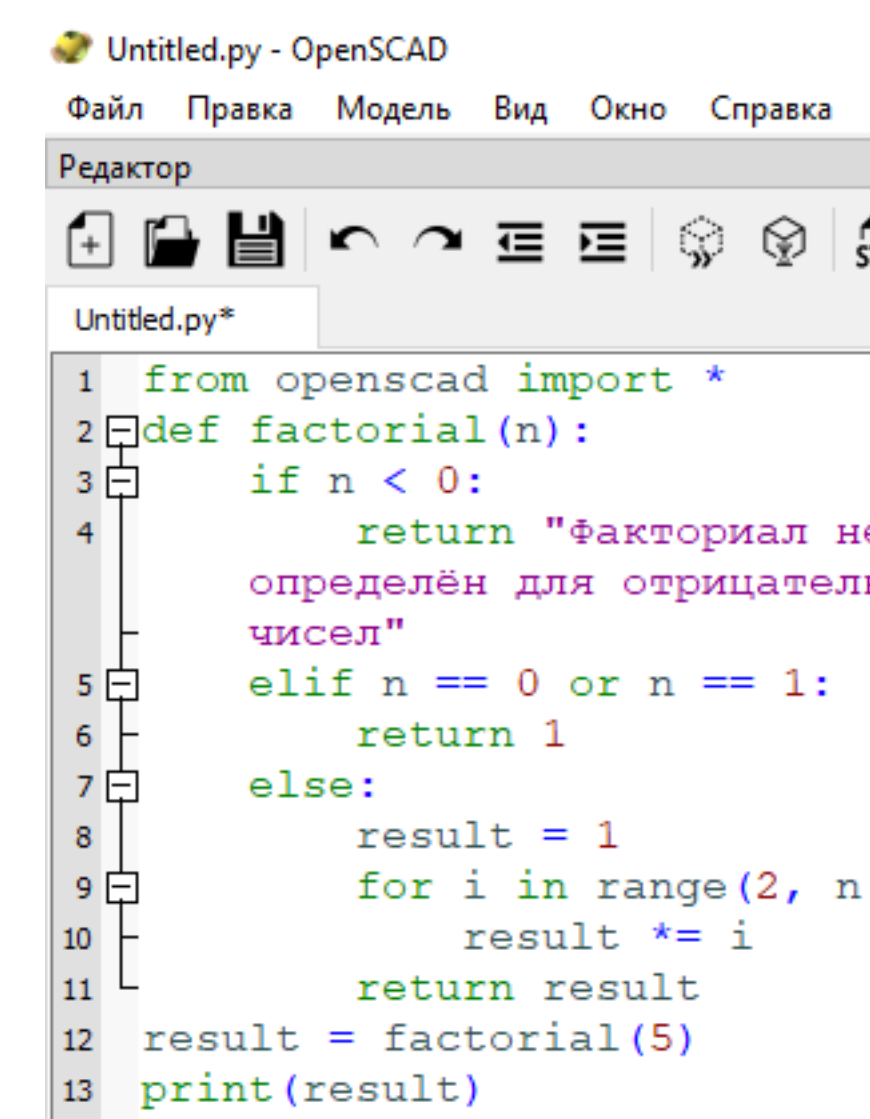

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.py*
1 from openscad import *
2 def factorial(n):
3 if n < 0:
4 return "Факториал не
определён для отрицатели
чисел"
5 elif n == 0 or n == 1:
6 return 1
7 else:
8 result = 1
9 for i in range(2, n
10 result *= i
11 return result
12 result = factorial(5)
13 print(result)

Рис. 20. Рекурсивные функции

Листинг 5:Работа с массивами

```
def sum_list(lst):
total = 0
for element in lst:
total += element
return total
result = sum_list([1, 2, 3, 4, 5])
print(result)
```

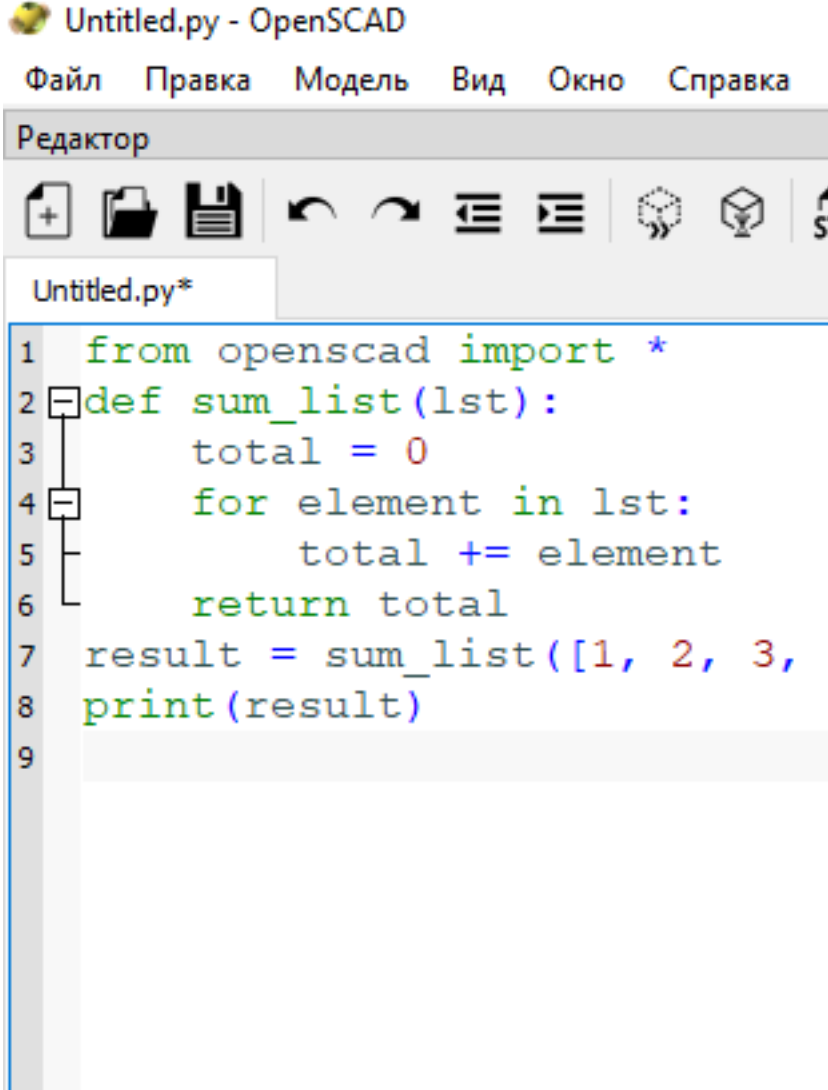

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.py*
1 from openscad import *
2 def sum_list(lst):
3 total = 0
4 for element in lst:
5 total += element
6 return total
7 result = sum_list([1, 2, 3,
8 print(result)
9

Рис. 21. Работа с массивами

Эта функция суммирует элементы массива.

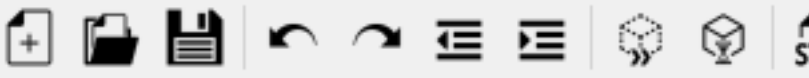
Листинг 6:Передача функций как аргумента

```
def operation(type, lst):
return [type(x) for x in lst]
def square(x):
return x * x
def double(x):
return x * 2
number = [4]
squared = operation(square, number)
doubled = operation(double, number)
print(squared)
print(doubled)
```

Untitled.py - OpenSCAD

Файл Правка Модель Вид Окно Справка

Редактор



Untitled.py*

```
1  from openscad import *
2  def operation(type, lst):
3      return [type(x) for x in
4  def square(x):
5      return x * x
6  def double(x):
7      return x * 2
8  number = [4]
9  squared = operation(square,
10 doubled = operation(double,
11 print(squared)
12 print(doubled)
13
```

Рис. 22. Передача функций как аргумента

Листинг 7:Использование функций в модулях

```
def cylinder_module(radius, height):
output = f"""
module cylinder_module(r, h) {{
// Создаем цилиндр
translate([0, 0, 0])
cylinder(r = r, h = h, center = true);
}}
cylinder_module({radius}, {height});
"""
return output
radius = 5
height = 10
cylinder_code = cylinder_module(radius, height)
print(cylinder_code)
```

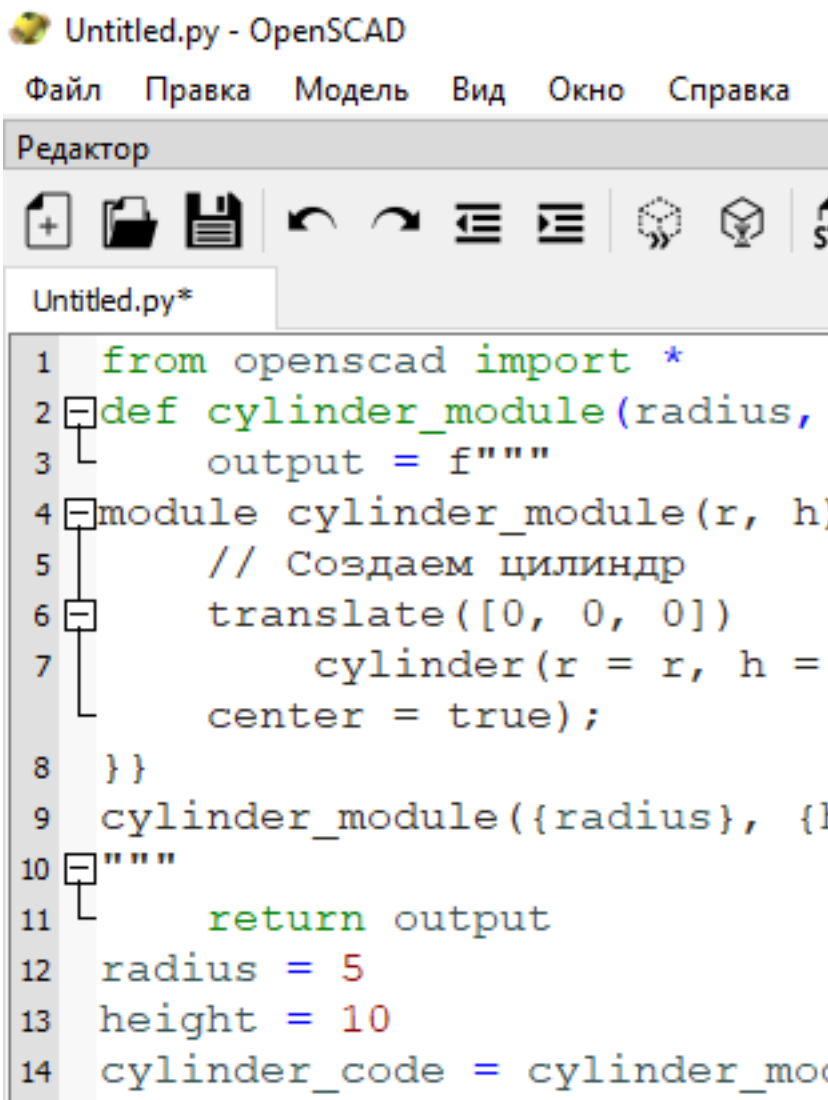

Untitled.py - OpenSCAD
Файл Правка Модель Вид Окно Справка
Редактор
Untitled.py*
1 from openscad import *
2 def cylinder_module(radius,
3 output = f"""
4 module cylinder_module(r, h)
5 // Создаем цилиндр
6 translate([0, 0, 0])
7 cylinder(r = r, h =
center = true);
8 }}
9 cylinder_module({radius}, {h
10 """
11 return output
12 radius = 5
13 height = 10
14 cylinder_code = cylinder_mo

Рис. 23. Использование функций в модулях

Заключение

Функции возвращают значения, но не влияют непосредственно на геометрию. Поддерживаются множественные аргументы, условные выражения, работа с массивами и рекурсия. Функции идеально подходят для параметрической настройки моделей.

Теперь давайте рассмотрим практический пример создания шестеренки с использованием функций.

Параметрическая шестеренка

Функция рассчитывает радиус зубца, а модуль создает трехмерную модель шестеренки.

Листинг 1:Функция рассчитывает радиус зубца

```
import math

def gear_radius(base_r, teeth, tooth_height):
"""Функция для расчета радиуса зубцов шестеренки."""
return base_r + tooth_height * (1 if teeth > 0 else 0)

def gear(base_r, teeth, tooth_height):
"""Создает шестеренку с заданными параметрами."""
output = "difference() {\n"
# Основной диск шестеренки
output += f" circle(r = {gear_radius(base_r, teeth, tooth_height)}, $fn = 100);\n"
```

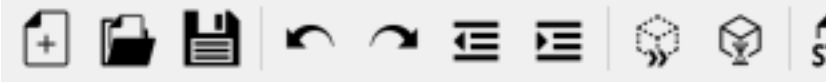
```
    # Создание выемок для зубцов
    for i in range(teeth):
    angle = i * (360 / teeth)
    x = base_r * math.cos(math.radians(angle))
    y = base_r * math.sin(math.radians(angle))
    output += f" translate([{x}, {y}])\n"
    output += f" rotate({angle})\n"
    output += f" square([{tooth_height * 2}, {base_r * 0.2}],
center = true);\n"
    output += "}\n"
    return output

    # Вызов функции с заданными параметрами
    gear_code = gear(base_r=20, teeth=12, tooth_height=3)
    print(gear_code)
```

Untitled.py - OpenSCAD

Файл Правка Модель Вид Окно Справка

Редактор



Untitled.py*

```
1  from openscad import *
2  import math
3
4  def gear_radius(base_r, teet
       tooth_height):
5      """Функция для расчета р
       зубцов шестеренки."""
6      return base_r + tooth_he
       (1 if teeth > 0 else 0)
7
8  def gear(base_r, teeth,
       tooth_height):
9      """Создает шестеренку с
       заданными параметрами."
10     output = "difference()
```

Рис. 24. Функция рассчитывает радиус зубца

Объяснение кода

1. gear_radius

– Эта функция считает радиус зубцов шестеренки.

2. gear

– Главный модуль, который создает шестеренку.

– Использует `difference()` для вычитания зубцов из основного круга.

3. Цикл for

– Генерирует зубцы шестеренки, вычисляя их расположение на основе угла.

4. translate и rotate

– Перемещают и поворачивают зубцы в нужные позиции.

5. Вывод

– Функция возвращает сгенерированный код, который можно использовать в PythonSCAD.

Улучшение шестеренки

Добавим возможность задавать диаметр центрального отверстия.

Листинг 1:Возможность задавать диаметр цен-

трального отверстия

```
    def cos(angle):
    import math
    return math.cos(math.radians(angle))
    def sin(angle):
    import math
    return math.sin(math.radians(angle))
    def radians(degrees):
    import math
    return degrees * (math.pi / 180)
    def circle(r, fn=100):
    return f'circle(r={r}, $fn={fn})'
    def square(size, center=False):
    return f'square(size={size}, center={center})'
    def translate(vector):
    return f'translate({vector})'
    def rotate(angle):
    return f'rotate(angle={angle})'
    def difference(*shapes):
    return 'difference() {' + ''.join(shapes) + '}'
    def gear_radius(base_r, teeth, tooth_height):
    return base_r + tooth_height
    def create_gear(base_r, teeth, tooth_height, hole_r=5):
    gear_shape = difference(
    circle(gear_radius(base_r, teeth, tooth_height), fn=100)
    )
```

```
    for i in range(teeth):
    angle = i * (360 / teeth)
    x = base_r * cos(angle)
    y = base_r * sin(angle)
    tooth = translate([x, y, 0])(
    rotate(angle)(
    square([tooth_height * 2, base_r * 0.2], center=True)
    )
    )
    gear_shape += tooth
    # Центральное отверстие
    central_hole = circle(hole_r, fn=50)
    gear_shape += central_hole
    return gear_shape
    # Пример вызова шестеренки с отверстием
    gear_model = create_gear(base_r=20, teeth=16,
tooth_height=3, hole_r=6)
```

Конец ознакомительного фрагмента.

Текст предоставлен ООО «Литрес».

Прочитайте эту книгу целиком, купив полную легальную версию на Литрес.

Безопасно оплатить книгу можно банковской картой Visa, MasterCard, Maestro, со счета мобильного телефона, с платежного терминала, в салоне МТС или Связной, через PayPal, WebMoney, Яндекс.Деньги, QIWI Кошелек, бонусными картами или другим удобным Вам способом.