Introduction

The code is a Python script designed to generate a 3D model of a portal frame structure using the Open Cascade Technology (OCCT) library. The structure includes columns, rafters, and purlins. The final model is saved in a STEP file format, which is a common CAD exchange format.

Libraries and Imports
The script uses the following libraries:
- Open Cascade Technology (OCCT): A powerful CAD kernel for 3D modeling and computation.
  - gp_Vec, gp_Trsf, gp_Ax1, gp_Pnt, gp_Dir: Geometric primitives and transformations.
  - BRepPrimAPI_MakeBox: To create box shapes.
  - BRepAlgoAPI_Fuse: To perform Boolean operations (e.g., fusing shapes).
  - BRepBuilderAPI_Transform: To apply geometric transformations to shapes.
  - STEPControl_Writer, STEPControl_AsIs: To export shapes to STEP files.
  - init_display: For displaying shapes using a simple GUI.
- math: For mathematical operations such as trigonometric functions.

Functions

1. create_i_section
This function creates an I-section (I-beam) shape.

- Parameters:
  - length: Length of the I-section.
  - width: Width of the I-section.
  - depth: Depth of the I-section.
  - flange_thickness: Thickness of the flange.
  - web_thickness: Thickness of the web.

- Logic:
  - The I-section is created by combining three boxes: bottom flange, top flange, and web.
  - The top flange is translated to the top of the web.
  - The web is positioned between the two flanges.

2. create_purlin_layout
This function creates a layout of purlins on the roof.

- Parameters:
  - num_purlins: Number of purlins.
  - purlin_width: Width of each purlin.
  - purlin_height: Height of each purlin.
  - purlin_depth: Depth of each purlin.
  - rafter_angle: Angle of the roof (affects the rise of the roof).

- Logic:
  - Purlins are distributed evenly along the roof, taking into account the rise due to the roof angle.
  - Each purlin is positioned using a translation transformation.

3. create_rafter
This function creates a rafter with a specified angle.

- Parameters:
  - length: Length of the rafter.
  - width: Width of the rafter.
  - purlin_depth: Depth of the purlin.
  - rafter_flange_thickness: Thickness of the rafter flange.
  - rafter_web_thickness: Thickness of the rafter web.
  - angle: Angle of the rafter.

- Logic:
  - The rafter is created as an I-section.
  - It is then rotated by the specified angle using a transformation.

4. create_portal_frame

The `create_portal_frame` function constructs a complete 3D model of a portal frame structure using the afore mentioned functions.

Parameters

The function takes several parameters to define the dimensions and characteristics of the portal frame:

- column_length: Length of the column.
- column_width: Width of the column.
- column_height: Height of the column.
- column_flange_thickness: Thickness of the column flange.
- column_web_thickness: Thickness of the column web.
- rafter_width: Width of the rafter.
- rafter_depth: Depth of the rafter.
- rafter_flange_thickness: Thickness of the rafter flange.
- rafter_web_thickness: Thickness of the rafter web.
- rafter_angle: Angle of the rafter from the horizontal.
- num_rafters: Number of rafters.
- purlin_width: Width of the purlin.
- purlin_height: Height of the purlin.
- purlin_depth: Depth of the purlin.

Steps

1. Create Columns

Columns are created using the `create_i_section` function, which constructs an I-section profile based on the given dimensions. This profile includes the flange and web thicknesses to represent the column's structure. The columns are positioned along the sides of the portal frame by translating them along the y-axis and x-axis to form the left and right sides of the frame. The column spacing is calculated based on the purlin depth and the number of columns per side.

2. Create Purlins

Purlins are horizontal beams that run along the length of the roof, providing support. These are created using the `create_purlin_layout` function. The function calculates the roof rise using the formula involving the width of the frame and the tangent of the rafter angle. The purlins are positioned to follow the slope of the roof, ensuring they are evenly spaced and correctly aligned with the rafters. Each purlin is

translated and combined into a single solid to represent the entire purlin layout.

3. Create Rafters

Rafters are diagonal beams that form the primary support structure of the roof. They are created using the `create_i_section` function and positioned on top of the columns. The rafter length is calculated to ensure it spans half the width of the frame, adjusted for the roof angle. Each rafter is translated to the correct position and rotated to match the specified angle. The right rafters are additionally aligned with the purlins by translating them along the x-axis and z-axis to match the roof's slope and rise.

4. Combine Components

The final step is to combine the columns, purlins, and rafters into a single solid representing the complete portal frame structure. This is done using Boolean fusion operations. The individual shapes of the columns, purlins, and rafters are merged to form a cohesive structure, ensuring that all components are properly connected and aligned.

5. save_to_step
This function saves the generated shape to a STEP file.

- Parameters:
  - shape: The shape to be saved.
  - filename: The name of the STEP file.

- Logic:
  - The shape is transferred to a STEP writer and written to the specified file.

Main Execution Block
- Parameters: Dimensions and properties of the portal frame components.
- Portal Frame Creation:
  - The portal frame is created using the `create_portal_frame` function.
- Display and Save:
  - The created portal frame is displayed using a simple GUI.
  - The portal frame is saved to a STEP file.

Units used:
- Length: Millimeters (mm).
- Angle: Degrees .

Calculations and Formulas
- Roof Rise:
 roof_rise = (8400 / 2) * math.tan(math.radians(rafter_angle))

  This calculates the vertical rise of the roof at the midpoint.

- Purlin Positioning:
  - Purlins are evenly spaced along the width of the roof.
  - The z-coordinate of each purlin is calculated based on its position relative to the roof rise.

- Rafter Length:

```
rafter_length = 8000 / 2 / math.cos(math.radians(rafter_angle))
```

This calculates the length of each rafter based on the roof angle.