```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.patches import Rectangle,Polygon, Arc

def create_binary_image(r=50, image_size=300):
    # Create a blank binary image with white background
    img = np.full((image_size, image_size), 1.0, dtype=np.float32)  #
Background white

    center = (image_size // 2, image_size // 2)  # center as a tuple (x, y)

    # Draw the central square (light gray)
    square_top_left = (center[0] - r, center[1] - r)
    square_bottom_right = (center[0] + r, center[1] + r)
    img[square_top_left[1]:square_bottom_right[1],
square_top_left[0]:square_bottom_right[0]] = 0.8  #gray

    # Draw the top and bottom semicircles
    y, x = np.ogrid[:image_size, :image_size]
    top_circle = (x - center[0]) ** 2 + (y - (center[1] - r)) ** 2 <= r ** 2
    bottom_circle = (x - center[0]) ** 2 + (y - (center[1] + r))**2 <=r ** 2
    img[top_circle] = 0.9  #light gray
    img[bottom_circle] = 0.9

    # Draw the left and right triangles
    for i in range(r):
 # Left triangle
        img[center[1] - r + i:center[1] + r - i, center[0] - r - i] = 0.9
 # Right triangle
        img[center[1] - r + i:center[1] + r - i, center[0] + r + i] = 0.9

    # Draw the inner square (white)
    inner_r = r // 2
    inner_square_top_left = (center[0] - inner_r, center[1] - inner_r)
    inner_square_bottom_right = (center[0] + inner_r, center[1] + inner_r)
    img[inner_square_top_left[1]:inner_square_bottom_right[1],
inner_square_top_left[0]:inner_square_bottom_right[0]] = 1.0  # White

    return img

# Generate and display the binary image with outlines and angles
r = 50
image_size = 300
binary_image = create_binary_image(r=r, image_size=image_size)

fig, ax = plt.subplots()
ax.imshow(binary_image, cmap='gray')
ax.axis('off')
```

```
# Draw outlines
center = (image_size // 2, image_size // 2)  # center as a tuple (x, y)
square_top_left = (center[0] - r, center[1] - r)
square_bottom_right = (center[0] + r, center[1] + r)
inner_r = r // 2
inner_square_top_left = (center[0] - inner_r, center[1] - inner_r)
inner_square_bottom_right = (center[0] + inner_r, center[1] + inner_r)

# Outer square outline
rect_outer = Rectangle((square_top_left[0], square_top_left[1]), 2 * r, 2 * r,
edgecolor='black', facecolor='0.6', linewidth=1)
ax.add_patch(rect_outer)

# Inner square outline
rect_inner = Rectangle((inner_square_top_left[0], inner_square_top_left[1]), 2
* inner_r, 2 * inner_r, edgecolor='black', facecolor='white', linewidth=1)
ax.add_patch(rect_inner)

# Add angles
# 90° angles at the corners of the outer square
ax.text(square_top_left[1]-3, square_top_left[0] + 13, '45°', color='black',
fontsize=8, ha='center', va='center')
ax.text(square_bottom_right[1]+8, square_top_left[0] + 13, '45°',
color='black', fontsize=8, ha='center', va='center')

# 45° angles at the tips of the triangles
ax.text(center[0] - r - 33, center[1] - r+48, '90°', color='black',
fontsize=8, ha='center', va='center')
ax.text(center[0] + r + 33, center[1] - r+48, '90°', color='black',
fontsize=8, ha='center', va='center')

# Add top and bottom semicircles with outline for the top semicircle
arc_top = Arc((center[0], center[1] + r), width=2*r, height=2*r, angle=0,
theta1=0, theta2=180, color='black', linewidth=1, edgecolor='black')
ax.add_patch(arc_top)

arc_bottom = Arc((center[0], center[1] - r), width=2*r, height=2*r, angle=0,
theta1=180, theta2=360, color='black', linewidth=1, edgecolor='black')
ax.add_patch(arc_bottom)

# Add outlines for the triangles
# Left triangle
triangle_left = Polygon([(center[0] - r, center[1] - r),
                         (center[0] - r, center[1] + r),
                         (center[0]-2*r, center[1])],
                        closed=True, edgecolor='black', facecolor='none',
linewidth=1)
```

```
ax.add_patch(triangle_left)

# Right triangle
triangle_right = Polygon([(center[0] + r, center[1] - r),
                          (center[0] + r, center[1] + r),
                          (center[1]+2*r, center[0])],
                         closed=True, edgecolor='black', facecolor='none',
linewidth=1)
ax.add_patch(triangle_right)

plt.show()
```