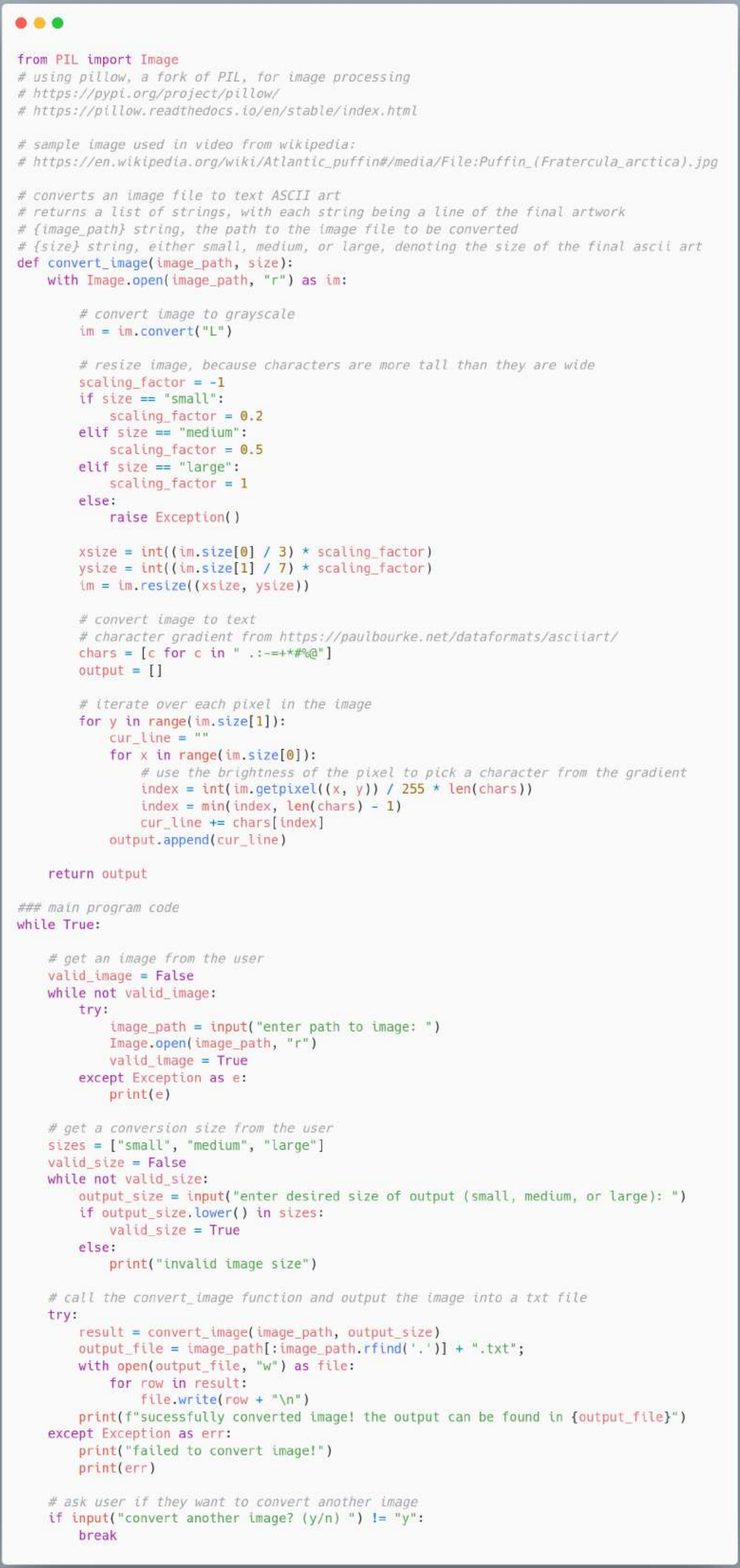

```python
from PIL import Image
# using pillow, a fork of PIL, for image processing
# https://pypi.org/project/pillow/
# https://pillow.readthedocs.io/en/stable/index.html

# sample image used in video from wikipedia:
# https://en.wikipedia.org/wiki/Atlantic_puffin#/media/File:Puffin_(Fratercula_arctica).jpg

# converts an image file to text ASCII art
# returns a list of strings, with each string being a line of the final artwork
# {image_path} string, the path to the image file to be converted
# {size} string, either small, medium, or large, denoting the size of the final ascii art
def convert_image(image_path, size):
    with Image.open(image_path, "r") as im:

        # convert image to grayscale
        im = im.convert("L")

        # resize image, because characters are more tall than they are wide
        scaling_factor = -1
        if size == "small":
            scaling_factor = 0.2
        elif size == "medium":
            scaling_factor = 0.5
        elif size == "large":
            scaling_factor = 1
        else:
            raise Exception()

        xsize = int((im.size[0] / 3) * scaling_factor)
        ysize = int((im.size[1] / 7) * scaling_factor)
        im = im.resize((xsize, ysize))

        # convert image to text
        # character gradient from https://paulbourke.net/dataformats/asciiart/
        chars = [c for c in " .:-=+*#%@"]
        output = []

        # iterate over each pixel in the image
        for y in range(im.size[1]):
            cur_line = ""
            for x in range(im.size[0]):
                # use the brightness of the pixel to pick a character from the gradient
                index = int(im.getpixel((x, y)) / 255 * len(chars))
                index = min(index, len(chars) - 1)
                cur_line += chars[index]
            output.append(cur_line)

    return output

### main program code
while True:

    # get an image from the user
    valid_image = False
    while not valid_image:
        try:
            image_path = input("enter path to image: ")
            Image.open(image_path, "r")
            valid_image = True
        except Exception as e:
            print(e)

    # get a conversion size from the user
    sizes = ["small", "medium", "large"]
    valid_size = False
    while not valid_size:
        output_size = input("enter desired size of output (small, medium, or large): ")
        if output_size.lower() in sizes:
            valid_size = True
        else:
            print("invalid image size")

    # call the convert_image function and output the image into a txt file
    try:
        result = convert_image(image_path, output_size)
        output_file = image_path[:image_path.rfind('.')] + ".txt";
        with open(output_file, "w") as file:
            for row in result:
                file.write(row + "\n")
        print(f"sucessfully converted image! the output can be found in {output_file}")
    except Exception as err:
        print("failed to convert image!")
        print(err)

    # ask user if they want to convert another image
    if input("convert another image? (y/n) ") != "y":
        break
```