```
#!/usr/bin/python
import numpy as np
import pylab


def gaussian(x, y, Lx, Ly):
    return sum([
        sum([
            np.exp(-((x + (n * Lx))**2 + (y + (m * Ly))**2) / (2 * (25**2)))
            for n in range(-1, 1)
        ])
        for m in range(-1, 1)
    ])

def part_a():
    img_data = np.loadtxt('blur.txt')
    pylab.imshow(img_data)
    pylab.set_cmap('Greys_r')
    pylab.show()


def part_b():
    img_data = np.loadtxt('blur.txt')
    shape = img_data.shape
    blur = [[gaussian(x, y, shape[0], shape[1])
             for x in range(shape[0])]
            for y in range(shape[1])]
    pylab.imshow(blur)
    pylab.set_cmap('Greys_r')
    pylab.show()


def part_c():
    img_data = np.loadtxt('blur.txt')
    shape = img_data.shape
    blur = [[gaussian(x, y, shape[0], shape[1])
             for x in range(shape[0])]
            for y in range(shape[1])]
    fft_img = np.fft.rfft2(img_data)
    fft_blur = np.fft.rfft2(blur)
    fft_unblured = fft_img
    for i, row in enumerate(fft_unblured):
        for j, elem in enumerate(row):
            if fft_blur[i][j] > 1e-3:
                fft_unblured[i][j] /= (fft_blur[i][j])
    unblured = np.fft.irfft2(fft_unblured)
    pylab.imshow(unblured)
    pylab.set_cmap('Greys_r')
    pylab.show()


def main():
    part_a()
    part_b()
    part_c()


if __name__ == "__main__":
    main()
```