Original.py

```python
import numpy as np
import matplotlib.pyplot as plt
```

import library

```python
# Fungsi untuk menghasilkan pola sederhana (garis diagonal)
def sample_image(x, y):
    if abs(x - y) <= 2: # Garis diagonal tebal dalam 2 piksel
        return 1.0  # Warna putih untuk garis
    else:
        return 0.0  # Warna hitam untuk area lainnya
```

buat fungsi sample_image dengan parameter x,y
gunain percabangan dengan fungsi absolute dari python jika x-y <= 2 maka return 1 (warna putih)
kalau tidak return 0 (warna hitam)

```python
# Image properties
width = 20
height = 20

# Membuat gambar dengan resolusi yang diinginkan
image = np.zeros((height, width))
```

buat variable gambar dengan lebar dan tinggi 20
buat variable image yang pake fungsi dari zeros dari numpy yang gunanya untuk buat array 2D sesuai sama nilai dari variable height sama width

```python
# Melakukan sampel gambar
for y in range(height):
    for x in range(width):
        image[y, x] = sample_image(x, y)
```

Untuk y masih di range tinggi dan x di range lebar maka nilai dari fungsi sample_image (x,y) akan di simpan ke matrix image y,x (baris, kolom)

```python
# Display gambar tanpa supersampling
plt.imshow(image, cmap='gray')
plt.title('Gambar Original')
plt.axis('off')
plt.show()
```

Menampilkan image dengan fungsi imshow, dengan colormap grayscale
dengan judul window “Gambar Original”
dan axis(sumbu) off

Supersampling.py

```python
import numpy as np
import matplotlib.pyplot as plt
```

import library

```python
# Fungsi untuk menghasilkan pola sederhana (garis diagonal)
def sample_image(x, y):
    if abs(x - y) <= 2: # Garis diagonal tebal dalam 2 piksel
        return 1.0  # Warna putih untuk garis
    else:
        return 0.0  # Warna hitam untuk area lainnya
```

buat fungsi sample_image dengan parameter x,y
gunain percabangan dengan fungsi absolute dari python jika x-y <= 2 maka return 1 (warna putih) kalau tidak return 0 (warna hitam)

```python
# Fungsi Supersampling
def supersample_image(width, height, super_factor):
    super_width = width * super_factor
    super_height = height * super_factor
```

buat fungsi supersample_image dengan 3 parameter
buat 2 variabel super width dan super height dimana lebar dan tinggi original akan di kali dengan super_factor

```python
# Membuat gambar dengan resolusi tinggi
higher_res_img = np.zeros((super_height, super_width))
```

variable higher_res_img untuk membuat matrix 2D menggunakan fungsi numpy zero dengan ukuran sesuai dengan variable super height dan width.

```python
# Melakukan sampel gambar dengan resolusi tinggi
for y in range(super_height):
    for x in range(super_width):
        orig_x = x / super_factor
        orig_y = y / super_factor

        higher_res_img[y, x] = sample_image(orig_x, orig_y)
```

Untuk y di range super_height dan x di range super_width untuk iterasi dari resolusi tinggi yang telah ditingkatkan (dikali dengan faktor supersampling).
dan mengonversi posisi piksel dalam gambar dengan resolusi tinggi yaitu orig_x dan orig_y untuk kembali ke posisi aslinya dalam gambar asli yang memiliki resolusi lebih rendah.
Menugaskan nilai dari orig_x dan orig_y ke fungsi sample_image dan menyimpan nilai nya ke matrix higher_res_img.

```python
# Mengurangi resolusi gambar (rata-rata piksel)
downsampled_img = higher_res_img.reshape(height, super_factor, width, super_factor).mean(axis=(1, 3))

return downsampled_img
```

variable downsampled_img untuk me reshape Matriks gambar dengan resolusi tinggi diubah bentuknya menjadi blok-blok super_factor x super_factor.

fungsi mean untuk menghitung rata-rata dari setiap blok super_factor x super_factor untuk mengurangi resolusi pada sumbu 1 dan sumbu 3 (sumbu tinggi (blok) dan lebar (blok))
return untuk mengembalikan nilai.

```
# Image properties
width = 20
height = 20
super_factor = 5  # Supersampling factor
```

variable width dan height dengan nilai 20 dan super_factor dengan nilai 5.

```
# Menghasilkan gambar Supersampling
supersampled_image = supersample_image(width, height, super_factor)
```

variable supersampled_image untuk fungsi supersample_image dengan parameter width height dan super_factor.

```
# Display gambar tanpa supersampling
plt.imshow(image, cmap='gray')
plt.title('Gambar Original')
plt.axis('off')
plt.show()
```

Menampilkan image dengan fungsi imshow, dengan colormap grayscale
dengan judul window “Gambar Original”
dan axis(sumbu) off

Konsep supersampling : proses supersampling melibatkan pembuatan gambar dengan resolusi yang lebih tinggi (misalnya, menggandakan jumlah piksel dengan faktor supersampling) untuk mendapatkan detail yang lebih halus atau lebih akurat. Setelah gambar dengan resolusi tinggi tersebut dibuat, langkah selanjutnya adalah mengembalikan hasil dari resolusi tinggi ke resolusi yang lebih rendah yang sesuai dengan gambar asli.

Areasampling.py

Semuanya sama cuman di tambah

```
# Area sampling: Menambahkan offset acak
orig_x += np.random.uniform(-0.5, 0.5)
orig_y += np.random.uniform(-0.5, 0.5)
```

untuk mengubah nilai dari orig_x dan orig_y untuk setiap perulangan mengunakan fungsi random. Uniform dari numpy yaitu untuk menghasilkan bilangan acak dari distribusi seragam antara -0.5 hingga 0.5. jadi setiap nilai di antara rentang tersebut memiliki probabilitas yang sama untuk dihasilkan.
maka nilai dari orig_x dan orig_y akan bergeser secara acak di range -0.5 sampai 0.5

Konsep :

menambahkan nilai acak seperti ini adalah untuk mengubah posisi sampel dengan jumlah yang sangat kecil secara acak. Dalam konteks supersampling atau pengambilan sampel gambar, ini bertujuan untuk memperkenalkan sedikit variasi kecil dalam lokasi sampel yang diambil, menghasilkan hasil yang lebih natural dan mengurangi pola terstruktur atau efek aliasing yang mungkin terjadi jika sampel-sampel itu terlalu teratur atau terstruktur. Ini cuman terjadi di area garis (atau pixel dengan nilai 1) area background tidak terpengaruh (nilai 0)

Pixelphasing.py

Sama cuman di tambah

```
# Pixel phasing: Menambahkan noise random
pixel_value += np.random.uniform(-0.25, 0.25)

higher_res_img[y, x] = pixel_value
```

Variable pixel_value yang akan bertambah seiring berjalannya perulangan untuk menggeser nilai piksel secara acak antara -0.25 hingga 0.25.

nilai piksel tersebut ditempatkan kembali ke gambar dengan resolusi tinggi pada posisi **(y, x)**.

konsep :
Tujuan dari teknik pixel phasing ini adalah untuk memberikan variasi kecil atau noise ke nilai piksel dalam gambar. Noise yang ditambahkan ini dapat memberikan hasil yang lebih alami, mengurangi pola terstruktur, atau menyamarkan artefak yang mungkin muncul akibat dari pola-pola yang terlalu teratur dalam gambar. Dalam konteks pengolahan gambar, teknik ini dapat membantu mengurangi efek aliasing atau membuat gambar terlihat lebih alami dengan variasi yang lebih halus dalam nilai pikselnya. Ini berlaku ke segala pixel yang ada di gambar.