Ćwiczenie 6

Zaobserwuj działanie następujących przekształceń punktowych:

1. Mnożenie obrazu przez stałą
2. Transformacja logarytmiczna
3. Zmiana dynamiki skali szarości (kontrastu)
4. Korekcja gamma

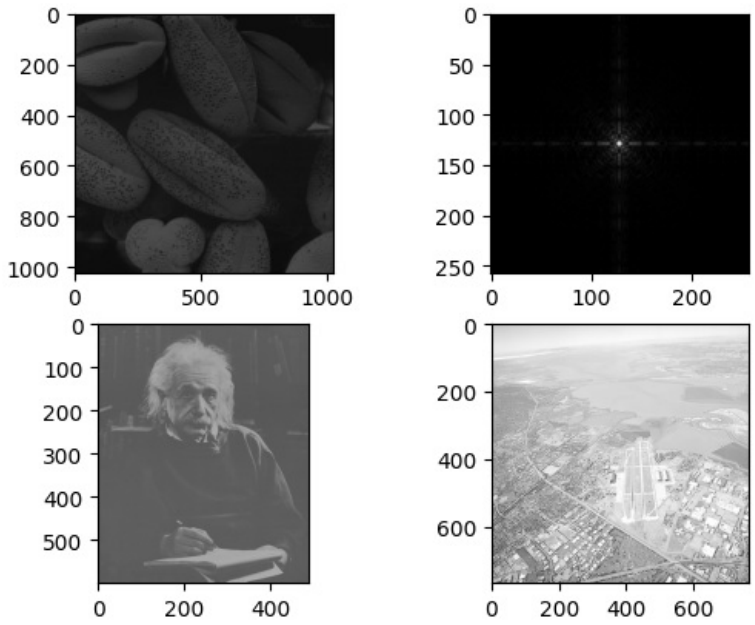
In [8]:
```
import matplotlib.pyplot as plt
import cv2
import numpy as np
```

Załadowanie wyznaczonych plików:

In [9]:
```
# Załadowanie pliku .tiff
img_a = cv2.imread("src/pollen-dark.tif")
img_b = cv2.imread("src/spectrum.tif")
img_c  = cv2.imread("src/einstein-low-contrast.tif")
img_d = cv2.imread("src/aerial_view.tif")
```

Obrazy bazowe:

In [10]:
```
#Wyświetlenie załadowanego obrazu
plt.figure()
plt.subplot(2,2,1)
plt.imshow(img_a, cmap='gray')

plt.subplot(2,2,2)
plt.imshow(img_b, cmap='gray')

plt.subplot(2,2,3)
plt.imshow(img_c, cmap='gray')

plt.subplot(2,2,4)
plt.imshow(img_d, cmap='gray')
plt.show()
```



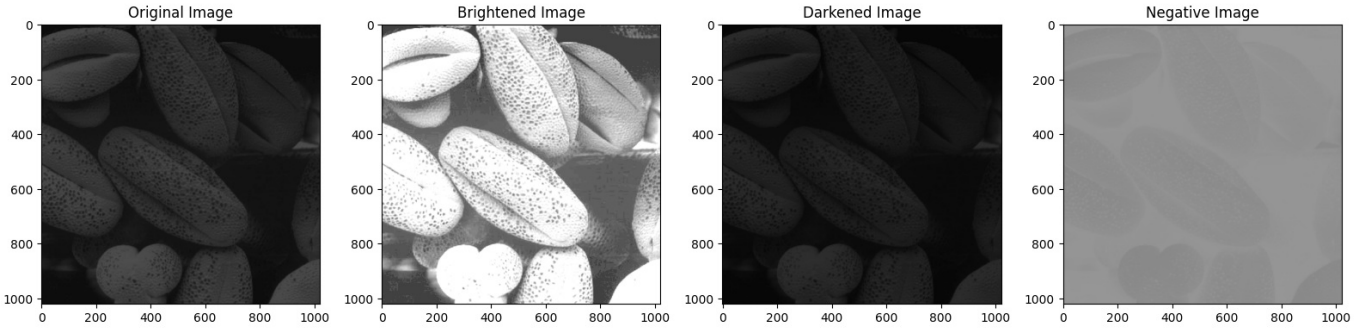
Funkcje przekształcające

Przyjmujemy że 'r', to nasz przetwarzany obraz
c jest stałą podaną przez użytkownika

1. T(r) = c*r
2. T(r) = c*log(1+r)
3. T(r) = 1/(1+(m/r)^e)
   m oraz *e* są ustalonymi wartościami całkowitymi
4. T(r) = c*(r^gamma); gdzie c>0 oraz gamma>0

In [11]:
```
def TransformByConst(img, c=1.0, inv_factor=0.0):
```

```
    """
    Transform the image using a constant c.
    """
    max_val = np.iinfo(img.dtype).max
    transformed_img = c * img.astype(np.float32)
    inverted = max_val - transformed_img
    result = (1 - inv_factor) * transformed_img + inv_factor * inverted
    return np.clip(result, 0, max_val).astype(img.dtype)

#pollen-dark.tif
processing = img_a
plt.figure(figsize=(16, 4))
plt.subplot(1,4,1)
plt.title("Original Image")
plt.imshow(processing, cmap='gray')
# rozjaśnienie
plt.subplot(1,4,2)
plt.title("Brightened Image")
img_an = TransformByConst(processing, c = 5.0)
plt.imshow(img_an, cmap='gray')
# ściemienienie
plt.subplot(1,4,3)
plt.title("Darkened Image")
img_ab = TransformByConst(processing, c = 0.6)
plt.imshow(img_ab, cmap='gray')
# negatyw
plt.subplot(1,4,4)
plt.title("Negative Image")
img_ac = TransformByConst(processing, c = 1.0, inv_factor = 0.6)
plt.imshow(img_ac, cmap='gray')
plt.tight_layout()
plt.show()
```

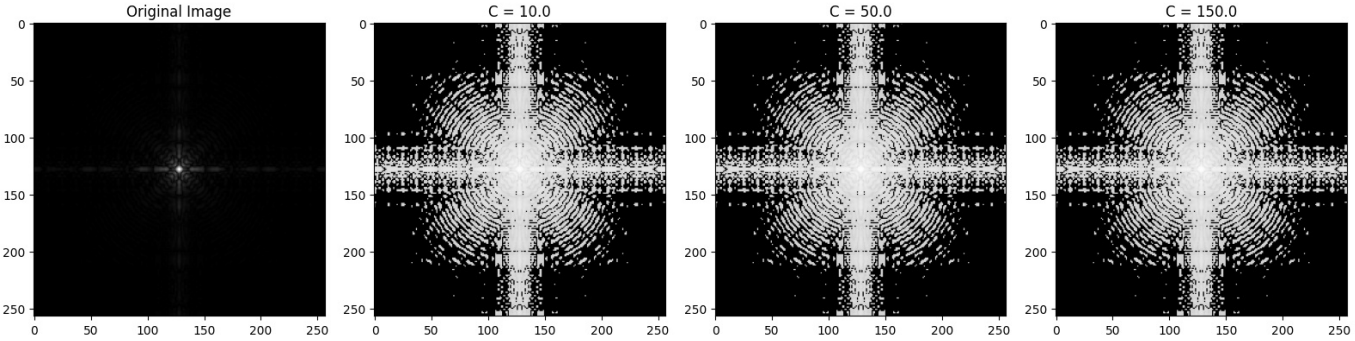


1. Przekształcenie przez stałą

Przekształcenie przy pomocy mnożenia przez stałą pozwala na zwiększenie (wartości > 1) lub zmniejszenie (wartości w zakresie (0,1)) jasności obrazu.
Mnożenie przez liczby ujemne daje obraz negatywowy gdize szarości odwrócone są liniowo.

In [15]:
```
def LogarithmicTransform(img, c):
    """
    Apply logarithmic transformation to the image.
    """
    img_float = img.astype(np.float32)  + 1e-9 # Avoid log(0)

    log_img = c * np.log(img_float)  # Apply log only to positive values

    log_img -= log_img.min()
    log_img /= log_img.max()

    return log_img

#spectrum.tif
processing = img_b
plt.figure(figsize=(16, 4))
plt.subplot(1,4,1)
plt.title("Original Image")
plt.imshow(processing, cmap='gray')

plt.subplot(1,4,2)
img_ba = LogarithmicTransform(processing, c = 10.0)
plt.title("C = 10.0")
plt.imshow(img_ba, cmap='gray')

plt.subplot(1,4,3)
img_bb = LogarithmicTransform(processing, c = 50.0)
plt.title("C = 50.0")
plt.imshow(img_bb, cmap='gray')
```

```
plt.subplot(1,4,4)
img_bc = LogarithmicTransform(processing, c = 150.0)
plt.title("C = 150.0")
plt.imshow(img_bc, cmap='gray')
plt.tight_layout()
plt.show()
```

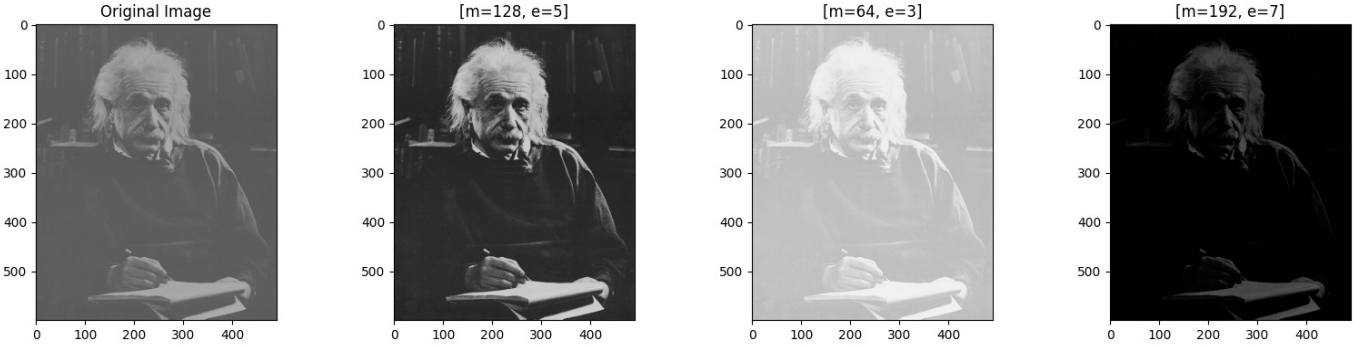


2. Przekształcenie logarytmiczne

Przekształcenie logarytmiczne ma za zadanie wzmocnić ciemne obszary poprzez wzór - $T(r) = c · log (1 + r)$, gdzie $c = const$
Przez wzgląd na działanie logarytmiczne $c > 0$

W wyniku badań, wartości $c=[10, 50, 150]$ nie przyniosły widocznych zmian

In [13]:
```
def GammaCorrection(img, m=1, e=1):
    """
    Apply contrast transformation to the image. Using formula:
    T(r) = 1/(1 + (m/img)^e)

    :param img: Input image
    :param m: Contrast factor
    :param e: Exponent factor (default is 1)
    :return: Transformed image
    """
    img_float = img.astype(np.float32) + 1e-9  # Avoid division by zero
    transformed_img = 1 / (1 + (m / img_float) ** e)
    return np.clip(transformed_img * 255, 0, 255).astype(np.uint8)

# einstein-low-contrast.tif
processing = img_c
plt.figure(figsize=(16, 4))
plt.subplot(1,4,1)
plt.title("Original Image")
plt.imshow(processing, cmap='gray')

plt.subplot(1,4,2)
img_ca = GammaCorrection(processing, m = 128, e = 5)
plt.title("[m=128, e=5]")
plt.imshow(img_ca, cmap='gray')

plt.subplot(1,4,3)
img_cb = GammaCorrection(processing, m = 64, e = 3)
plt.title("[m=64, e=3]")
plt.imshow(img_cb, cmap='gray')

plt.subplot(1,4,4)
img_cc = GammaCorrection(processing, m = 192, e = 7)
plt.title("[m=192, e=7]")
plt.imshow(img_cc, cmap='gray')
plt.tight_layout()
plt.show()
```



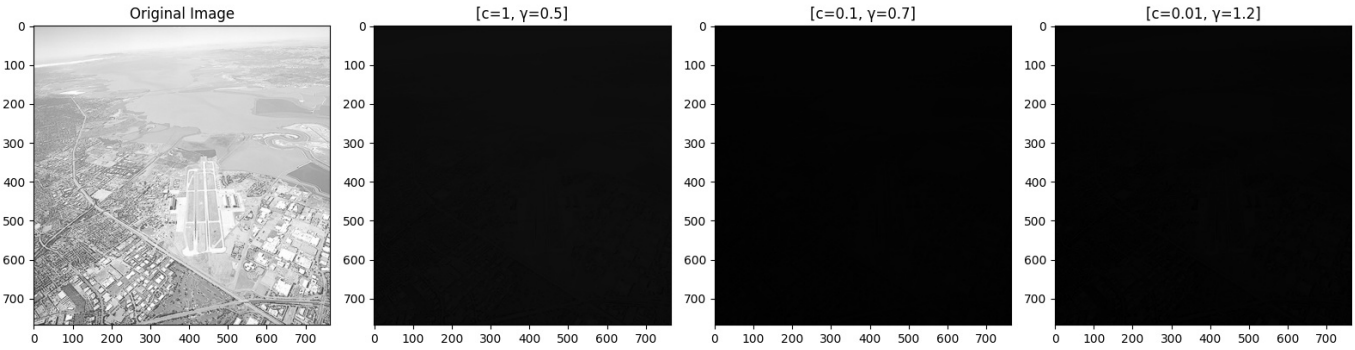
3. Przekształcenie Sigmoid (odwrotność funkcji potęgowej)

Przekształcenie to przy pomocy parametrów m oraz e zmieniają kontrast na podstawie wybranego poziomu szarości m odpowiada za poziom szarości wokół którego następuje szybka zmiana kontrastu, natomiast e definiuje stromość czyli kontrast

W celu przeprowadzenia badań wybrano wartości

- m=128, e=5.0
  Obraz z podwyższonym kontrastem wokół średniej szarości. Ciemne stają się cemniejsze, jasne jaśniejsze.
- m=64, e=3.0
  Próg przesunięty w stronę ciemnych tonów.
- m=192, e=7.0
  Próg przesunięty w stronę ajnsych tonów, bardzo ostry kontrast.

In [14]:
```
def GammaCorrection(img, c=1, γ=1):
    """
    Apply gamma correction to the image. Using formula:
    s = c * (r ** γ)
    where s is the output pixel value, r is the input pixel value, c is a constant, and γ is the exponent.

    :param img: Input image
    :param c: Constant factor (default is 1)
    :param γ: Exponent factor (default is 1)
    :return: Transformed image
    """
    img_float = img.astype(np.float32) + 1e-9  # Avoid division by zero
    transformed_img = c * (img_float ** γ)
    return np.clip(transformed_img, 0, 255).astype(np.uint8)

#aerial_view.tif
processing = img_d
plt.figure(figsize=(16, 4))
plt.subplot(1,4,1)
plt.title("Original Image")
plt.imshow(processing, cmap='gray')

plt.subplot(1,4,2)
img_da = GammaCorrection(processing, c=1, γ=0.5)
plt.title("[c=1, γ=0.5]")
plt.imshow(img_da, cmap='gray')

plt.subplot(1,4,3)
img_db = GammaCorrection(processing, c=0.1, γ=0.7)
plt.title("[c=0.1, γ=0.7]")
plt.imshow(img_db, cmap='gray')

plt.subplot(1,4,4)
img_dc = GammaCorrection(processing, c=0.01, γ=1.2)
plt.title("[c=0.01, γ=1.2]")
plt.imshow(img_dc, cmap='gray')
plt.tight_layout()
plt.show()
```



4. Konwersja gamma

Przekształcenie to przy pomocy parametrów c oraz $γ$ zmieniają kontrast lub jasność obrazu

W celu przeprowadzenia badań wybrano wartości

- c=1, γ=0.5
  Obraz został wyraźnie rozjaśniony. Wartość $γ < 0$ powoduje, że ciemne piksele stają się jaśniejsze. Cienie są podbite.
- c=0.1, γ=0.7
  Z powodu niskiego współczynika c obraz jest przyciemniony pomimo wartości $γ<0$, dzięki temu cienie są tylko

nieznacznie podbite.

- c=0.01, e=1.2
  Silnie przyciemniony obraz z przyciemnieniem jasnych obszarów