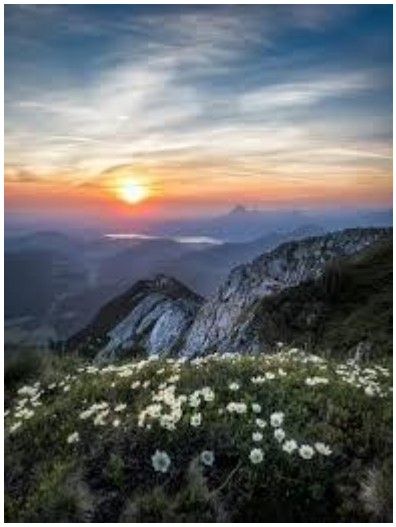
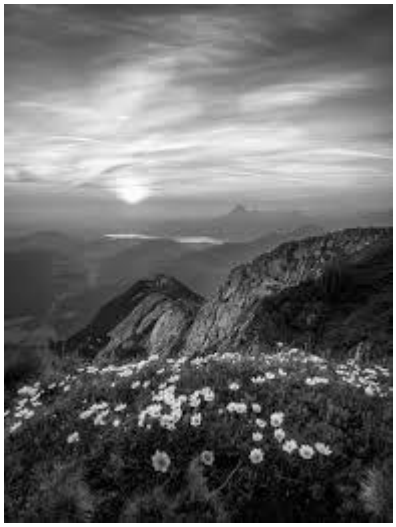
IMAGE PROCESSING

EXPERIMENT 05

```
from PIL import Image

# Load image
image = Image.open('input_image.jpg')

# Convert the image to grayscale
gray_image = image.convert('L')

# Display the grayscale image
gray_image.show()

# Save the grayscale image
gray_image.save('grayscale_image.jpg')
```


(input_image.jpg)


(greyscale_image.jpg)

```
from PIL import Image, ImageOps

# Load grayscale image
gray_image = Image.open('grayscale_image.jpg')

# Convert the grayscale image to a digital negative
negative_image = ImageOps.invert(gray_image)

# Display the digital negative image
negative_image.show()

# Save the digital negative image
negative_image.save('digital_negative.jpg')
```


(digital_negative.jpg)

```
from PIL import Image

# Open the grayscale image
gray_image = Image.open('grayscale_image.jpg')

# Define the threshold value
threshold_value = int(input("Enter threshold value:"))

# Apply thresholding to grayscale image
threshold_image = gray_image.point(lambda p: 255 if p > threshold_value
else 0)

# Display the thresholded image
threshold_image.show()

# Save the thresholded image
threshold_image.save('threshold_image.jpg')
```

```
Enter threshold value:125
```

(threshold_image.jpg)

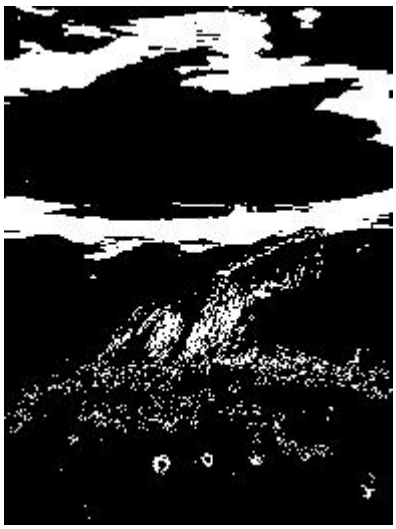
```
Enter threshold value:200
```

(threshold_image.jpg)

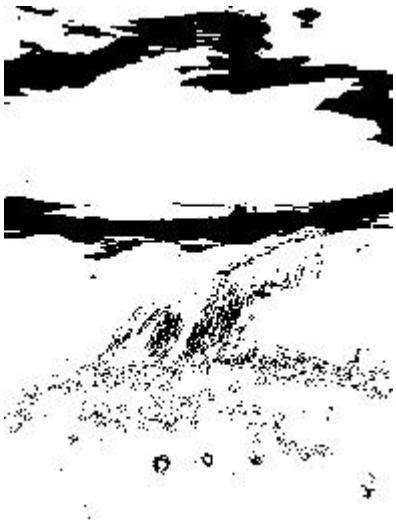
Intensity Slicing (4 parameters):

```python
from PIL import Image
import numpy as np

# Open the grayscale image
gray_image = Image.open('grayscale_image.jpg').convert('L')

# Convert the grayscale image to a numpy array for easier manipulation
image_array = np.array(gray_image)

# Define the intensity range for slicing
r1 = 100  # Lower bound of the range
r2 = 150  # Upper bound of the range

# Apply intensity slicing
```

```
sliced_image_array = np.where((image_array > r1) & (image_array < r2),
255, 0).astype(np.uint8)

# Convert the result back to a PIL image
sliced_image = Image.fromarray(sliced_image_array)

# Display the sliced image
sliced_image.show()

# Save the sliced image
sliced_image.save('sliced_image.jpg')
```



```
from PIL import Image

# Open the grayscale image
gray_image = Image.open('grayscale_image.jpg').convert('L')
# Define the intensity range (r1 and r2)
r1 = 100  # Lower bound of the range
r2 = 150  # Upper bound of the range

# Apply intensity slicing to the grayscale image
sliced_image = gray_image.point(lambda p: 0 if r1 < p < r2 else 1)

# To display the image properly, convert it back to a displayable
format (255 scale)
display_image = sliced_image.point(lambda p: p * 255)

# Display the intensity sliced image
display_image.show()

# Save the intensity sliced image
display_image.save('intensity_sliced_image.jpg')
```



```
from PIL import Image

# Open the grayscale image
gray_image = Image.open('grayscale_image.jpg')

# Convert image to numpy array for manipulation
import numpy as np
gray_array = np.array(gray_image)

# Define the intensity range and value to be set
r1, r2 = 100, 200  # Set the lower and upper bounds for the intensity
range
k = 150            # Set the intensity value for pixels within the
range

# Perform intensity slicing
sliced_array = np.where((gray_array > r1) & (gray_array < r2), k,
gray_array)

# Convert the numpy array back to an image
sliced_image = Image.fromarray(sliced_array.astype('uint8'))

# Display the intensity sliced image
sliced_image.show()

# Save the intensity sliced image
sliced_image.save('intensity_sliced_image3.jpg')
```

```
from PIL import Image
import numpy as np

# Open the grayscale image
gray_image = Image.open('grayscale_image.jpg').convert('L')

# Convert image to numpy array for pixel manipulation
image_array = np.array(gray_image)

# Define the intensity slicing parameters
r1 = 100  # Lower bound of the intensity range
r2 = 200  # Upper bound of the intensity range
k = 150    # Constant value for pixels outside the range

# Apply intensity slicing
sliced_image_array = np.where((image_array > r1) & (image_array < r2),
image_array, k)

# Convert the resulting numpy array back to an image
sliced_image = Image.fromarray(sliced_image_array.astype(np.uint8))

# Display the sliced image
sliced_image.show()

# Save the sliced image
sliced_image.save('sliced_image4.jpg')
```