Information Security Lab
Week-10

PRERIT SHARMA
21103121

QUESTION 1:

```
from PIL import Image

def display_colors(image_path):
    img = Image.open(image_path)
    pixels = img.load()

    width, height = img.size

    for i in range(width):
        for j in range(height):
            r, g, b = pixels[i, j]
            print(f"Pixel ({i}, {j}): RGB = ({r}, {g}, {b})")

# Example Usage
image_path = "path/to/your/image.jpg"
display_colors(image_path)
```

QUESTION 2:

```
from PIL import Image

def set_lowest_two_bits(image_path, color_name):
    img = Image.open(image_path)
    pixels = img.load()

    r, g, b = [int(x) for x in color_name.split(',')]

    width, height = img.size

    for i in range(width):
        for j in range(height):
            old_r, old_g, old_b = pixels[i, j]
            new_r = (old_r & ~3) | ((r >> 6) & 3)
            new_g = (old_g & ~3) | ((g >> 6) & 3)
            new_b = (old_b & ~3) | ((b >> 6) & 3)
```

```
            pixels[i, j] = (new_r, new_g, new_b)

    img.save("output_image.png")

# Example Usage
image_path = "path/to/your/image.jpg"
color_name = "100,150,200"
set_lowest_two_bits(image_path, color_name)
```

QUESTION 3:

```
from PIL import Image

def embed_secret_image(cover_path, secret_path):
    cover_img = Image.open(cover_path)
    secret_img = Image.open(secret_path)

    cover_pixels = cover_img.load()
    secret_pixels = secret_img.load()

    width, height = cover_img.size

    for i in range(width):
        for j in range(height):
            cover_r, cover_g, cover_b = cover_pixels[i, j]
            secret_r, secret_g, secret_b = secret_pixels[i % secret_img.width, j % secret_img.height]

            new_r = (cover_r & ~3) | (secret_r >> 6)
            new_g = (cover_g & ~3) | (secret_g >> 6)
            new_b = (cover_b & ~3) | (secret_b >> 6)

            cover_pixels[i, j] = (new_r, new_g, new_b)

    cover_img.save("output_cover_image.png")

def extract_secret_image(stego_path, secret_size):
    stego_img = Image.open(stego_path)
    stego_pixels = stego_img.load()

    width, height = stego_img.size
    secret_img = Image.new("RGB", (secret_size, secret_size))
    secret_pixels = secret_img.load()

    for i in range(secret_size):
        for j in range(secret_size):
            stego_r, stego_g, stego_b = stego_pixels[i, j]
            secret_r = (stego_r & 3) << 6
            secret_g = (stego_g & 3) << 6
            secret_b = (stego_b & 3) << 6
```

```
            secret_pixels[i, j] = (secret_r, secret_g, secret_b)

    secret_img.save("extracted_secret_image.png")

# Example Usage
cover_path = "path/to/cover/image.jpg"
secret_path = "path/to/secret/image.jpg"
embed_secret_image(cover_path, secret_path)
extract_secret_image("output_cover_image.png", secret_path)
```

QUESTION 4:

```
def CanbeHidden(cover_path, secret_path):
    cover_img = Image.open(cover_path)
    secret_img = Image.open(secret_path)

    cover_size = cover_img.size[0] * cover_img.size[1]
    secret_size = secret_img.size[0] * secret_img.size[1]

    return secret_size < cover_size

# Example Usage
cover_path = "path/to/cover/image.jpg"
secret_path = "path/to/secret/image.jpg"
result = CanbeHidden(cover_path, secret_path)
print("Can be hidden:", result)
```

QUESTION 5:

```
def encode_message(message):
    encoded_message = []
    for char in message:
        if char.isalpha():
            encoded_message.append(ord(char.upper()) - ord('A') + 1)
        elif char.isspace():
            encoded_message.append(27)
    encoded_message.append(0)
    return encoded_message

def hide_message(image_path, message):
    img = Image.open(image_path)
    pixels = img.load()

    width, height = img.size
    message_values = encode_message(message)
    message_index = 0
```

```
    for i in range(width):
        for j in range(height):
            r, g, b = pixels[i, j]

            if message_index < len(message_values):
                two_bits = message_values[message_index]
                r = (r & ~3) | ((two_bits >> 4) & 3)
                g = (g & ~3) | ((two_bits >> 2) & 3)
                b = (b & ~3) | (two_bits & 3)
                message_index += 1

            pixels[i, j] = (r, g, b)

    img.save("stego_image.png")

def retrieve_message(stego_path):
    stego_img = Image.open(stego_path)
    stego_pixels = stego_img.load()

    width, height = stego_img.size
    message_values = []

    for i in range(width):
        for j in range(height):
            r, g, b = stego_pixels[i, j]
            two_bits = ((r & 3) << 4) | ((g & 3) << 2) | (b & 3)
            message_values.append(two_bits)

    decoded_message = ""
    index = 0

    while index < len(message_values):
        value = message_values[index]
        if value == 0:
            break
        elif value == 27:
            decoded_message += " "
        else:
            decoded_message += chr(value + ord('A') - 1)
        index += 1

    return decoded_message

# Example Usage
image_path = "path/to/your/image.jpg"
message = "HELLO WORLD"
hide_message(image_path, message)
retrieved_message = retrieve_message("stego_image.png")
print("Retrieved Message:", retrieved_message)
```