Shiv Nadar University, Chennai
School of Engineering
Department of Computer Science

CS1802 -- Programming in Python Lab Class: 2024-2028 B. Tech CSE (Cyber)

Date: 23/04/2025 Continuous Lab Evaluation – 12 (10 Marks)

Statement: The process of hiding text in image is steganography. The process followed to encode and decode the data is given below.

Encode Data into an Image:

1. Convert Data to Binary: Each character in the secret message is converted to its 8-bit binary representation using ASCII values.

2. Modify Pixel Values:

- Pixels are processed three at a time (9 RGB values in total).
- The first 8 values store binary data:
- If the **binary bit is 1**: make the value **odd**.
- If the **binary bit is 0:** make the value **even**.
- The 9th value ensures the message continues—it's even if there's more data to encode.

For example:

Message to hide: "Hii" (3 bytes – needs 9 pixel values).

- H – ASCII 72 = Binary 01001000
- I – ASCII 105 = Binary 01101001
- I – ASCII 105 = Binary 01101001
- Total binary data = 24 bits
- Modify pixel values accordingly:

Original Pixels:

[(27, 64, 164), (248, 244, 194), (174, 246, 250),
(149, 95, 232), (188, 156, 169), (71, 167, 127),
(132, 173, 97), (113, 69, 206), (255, 29, 213),
(53, 153, 220), (246, 225, 229), (142, 82, 175)]

Take the first 3 pixels:

(27, 64, 164), (248, 244, 194), (174, 246, 250)

Convert 'H' (ASCII 72) to binary: 01001000
Modify pixel values:
0: Keep Even, 1: Make Odd

Original Value	Binary Bit	Modified Value
27 (Odd)	0 → Make Even	**26**
64 (Even)	1 → Make Odd	**63**
164 (Even)	0 → Keep Even	**164**
248 (Even)	0 → Keep Even	**248**
244 (Even)	1 → Make Odd	**243**
194 (Even)	0 → Keep Even	**194**
174 (Even)	0 → Keep Even	**174**
246 (Even)	0 → Keep Even	**246**

Final Modified Image (After Encoding "Hii"):
[(26, 63, 164), (248, 243, 194), (174, 246, 250), (148, 95, 231),
(188, 155, 168), (70, 167, 126), (132, 173, 97), (112, 69, 206),
(254, 29, 213), (53, 153, 220), (246, 225, 229), (142, 82, 175)]

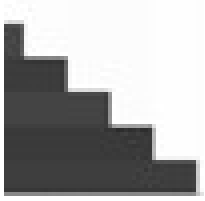
Decode the data:
To extract the hidden message:
1. Read the image three pixels at a time.
2. Extract the binary values using the same encoding rule:
    - Odd value – 1
    - Even value – 0
3. Stop when the last value is odd, indicating the message has ended.
4. Convert the extracted binary sequence back to text.

Task: Develop a python code to decode the encrypted image pixels and retrieve the message back.

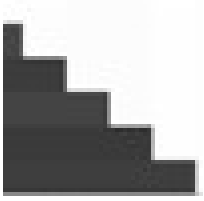
```
number = 10
binary_representation = bin(number)[2:]
print(f"The binary representation of {number} is: {binary_representation}")


binary_string = '01001000 01100101 01101100 01101100 01101111'
ascii_text = ''.join([chr(int(binary, 2)) for binary in binary_string.split(' ')])
print(ascii_text)

....
....
ascii_text = "Binary"
binary_string = ' '.join([f'{ord(char):08b}' for char in ascii_text])
print(binary_string)

....
```

Test Image:

Original Image



Encrypted Image:



Sample12.txt file has 50 x 50 Pixels values.

Decode the text back !