Geoffrey Browning
Assignment 1 report

Purpose

To explore image steganography, encompassing payload concealment and recovery.

Requirements

Task	Status
Program 1: Determining Payload Size	Fully implemented
Program 2: Encrypting and Encoding payload	Fully implemented
Implementation of LSB encoding technique	Fully implemented
Optional encryption of payload	Fully implemented
Program 3: Decoding and Decrypting payload	Fully implemented
Accurate decoding of LSB technique	Fully implemented

Platforms

geoProgram1, geoProgram2, geoProgram3 have all been tested on:

- Windows 11
- Fedora 36

Language

Python3

Design

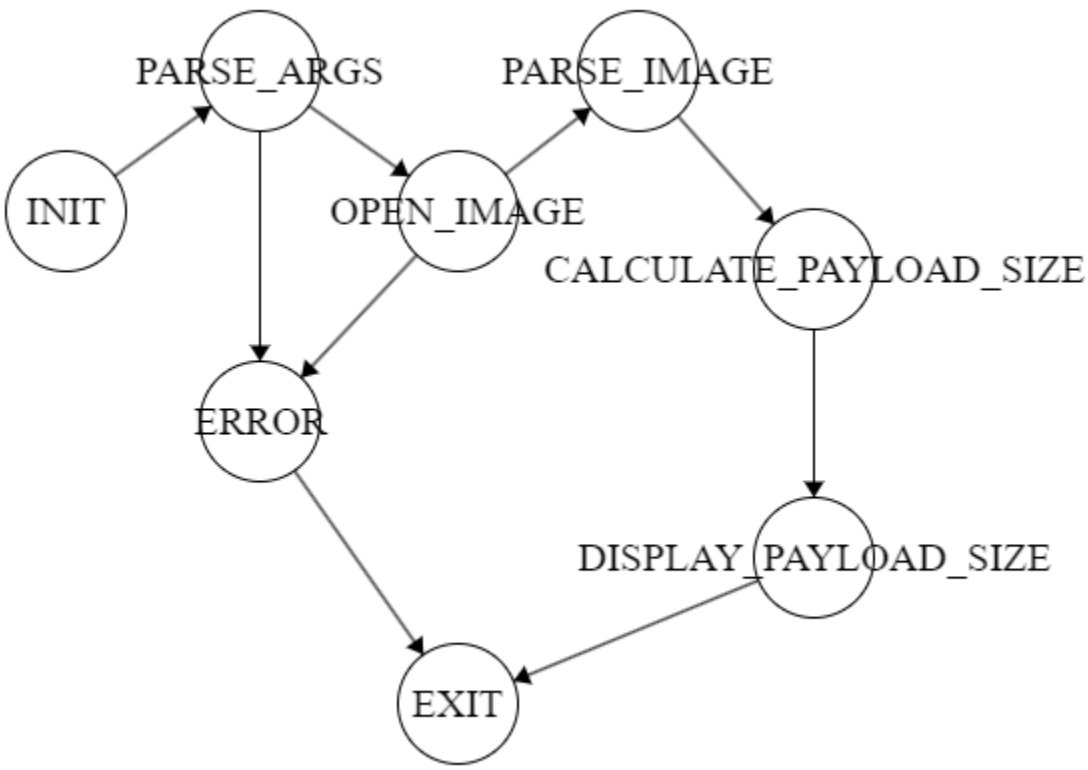
geoProgram1.py

Finite State Machine

State Table

From State	To State	Action
INIT	PARSE_ARGS	parse_args
PARSE_ARGS	OPEN_IMAGE	open
PARSE_ARGS	ERROR	argument_error
OPEN_IMAGE	PARSE_IMAGE	parse_image
OPEN_IMAGE	ERROR	file_not_found
PARSE_IMAGE	CALCULATE_PAYLOAD_SIZE	calculate_payload_size
CALCULATE_PAYLOAD_SIZE	DISPLAY_PAYLOAD_SIZE	display_payload_size
DISPLAY_PAYLOAD_SIZE	EXIT	Stop program
ERROR	EXIT	Stop program

State Transition Diagram



Pseudocode

```
INITIALIZE filename
INITIALIZE bitsPerPixel
parse command line arguments for fileName, bitsPerPixel
READ fileName as png coverImage
DETERMINE coverImage height and width
CALCULATE maxPayloadSize = height * width * bitsPerPixel
DISPLAY maxPayloadSize
```

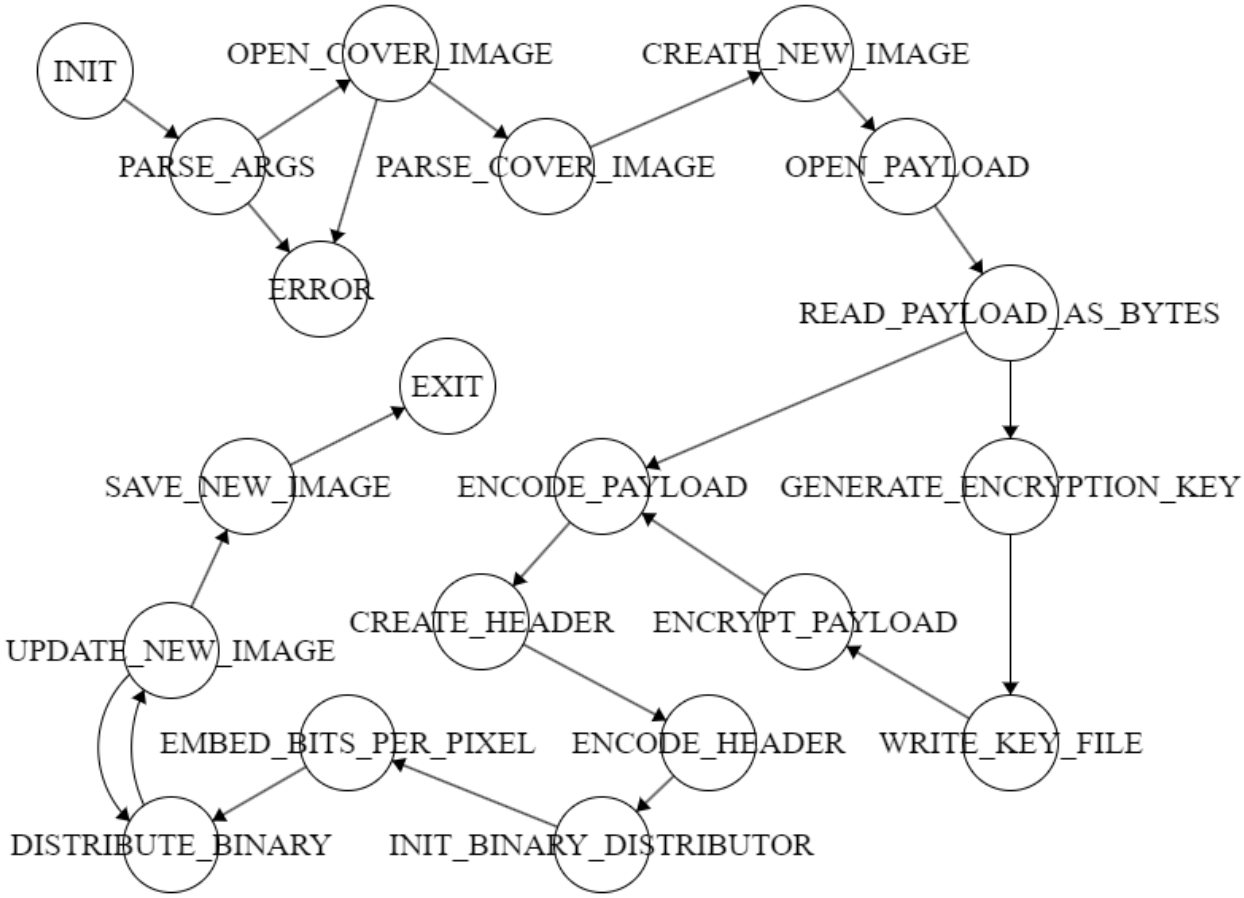
geoProgram2.py

Finite State Machine

State Table

From State	To State	Action
INIT	PARSE_ARGS	Parse_args
PARSE_ARGS	OPEN_COVER_IMAGE	Open_cover_image
PARSE_ARGS	ERROR	Argument_error
OPEN_COVER_IMAGE	PARSE_COVER_IMAGE	Parse_cover_image
OPEN_COVER_IMAGE	ERROR	File_not_found_error
PARSE_COVER_IMAGE	CREATE_NEW_IMAGE	Create_new_image
CREATE_NEW_IMAGE	OPEN_PAYLOAD	Open_payload
OPEN_PAYLOAD	READ_PAYLOAD_AS_BYTES	Read_payload
READ_PAYLOAD_AS_BYTES	ENCODE_PAYLOAD	Encode_payload
READ_PAYLOAD_AS_BYTES	GENERATE_ENCRYPTION_KEY	Generate_key
GENERATE_ENCRYPTION_KEY	WRITE_KEY_FILE	Write_key_file
WRITE_KEY_FILE	ENCRYPT_PAYLOAD	Encrypt_payload
ENCRYPT_PAYLOAD	ENCODE_PAYLOAD	Encode_payload
ENCODE_PAYLOAD	CREATE_HEADER	Create_header
CREATE_HEADER	ENCODE_HEADER	Encode_header
ENCODE_HEADER	INIT_BINARY_DISTRIBUTOR	Init_binary_distributor
INIT_BINARY_DISTRIBUTOR	EMBED_BITS_PER_PIXEL	Embed_bits_per_pixel
EMBED_BITS_PER_PIXEL	DISTRIBUTE_BINARY	Distribute_binary
DISTRIBUTE_BINARY	UPDATE_NEW_IMAGE	Update_new_image
UPDATE_NEW_IMAGE	SAVE_NEW_IMAGE	Save_new_image
SAVE_NEW_IMAGE	EXIT	Stop program

State Transition Diagram



Pseudocode

```
INITIALIZE fileName
INITIALIZE bitsPerPixel
INITIALIZE payloadFileName
INITIALIZE encrypted
parse command line arguments for fileName, bitsPerPixel, payloadFileName, encrypted
INITIALIZE mod to bitsPerPixel % 3
INITIALIZE div to ceil of bitsPerPixel / 3
READ fileName as png coverImage
DETERMINE coverImage height and width
INITIALIZE newImage with coverImage height and width
IF payloadFileName contains ".png" THEN
        READ payloadFileName as png payload
        PARSE payload for payloadWidth and payloadHeight
        CONVERT payload to bytes
ELSE
        READ payloadFileName bytes as payload
END IF
IF encrypted THEN
        generate encryption key
        WRITE encryption key to file
        ENCRYPT payload
END IF
ENCODE payload to base 2
INITIALIZE PAD_LENGTH to 50
IF length of payloadFileName > 15 THEN
        SET payloadFileName to default name
END IF
INITIALIZE header to length of payload + payloadWidth + payloadHeight + payloadFileName + encrypted
PAD header to 50 char length
ENCODE header to base 2
INITIALIZE binaryDistributor with encoded Header + encoded Payload
FOR each value x in RANGE(0, coverImageWidth)
        FOR each value y in RANGE(0, coverImageHeight)
                red, green, blue = coverImage[x, y]
                IF x and y == 0 THEN
                        convert bitsPerPixel to 5bit base2
                        EMBED first 2 bits of bitsPerPixel in red
                        EMBED second 2 bits of bitsPerPixel in green
                        EMBED last bit of bitsPerPixel in blue
                END IF
                IF binaryDistributor is empty THEN
                        newImage[x, y] = coverImage[x, y]
```

```
                CONTINUE
        END IF
        EMBED div bits into red
        IF mod == 1 THEN
                IF div > 1 THEN
                        EMBED div-1 bits into green
                        EMBED div-1 bits into blue
                END IF
        ELSE IF mod == 2 THEN
                EMBED div bits into green
                IF div > 1 THEN
                        EMBED div-1 bits into blue
                END IF
        ELSE
                EMBED div bits into green
                EMBED div bits into blue
        END IF
        newImage[x, y] = (red, green, blue)
SAVE newImage as “EmbeddedImage.png”
```

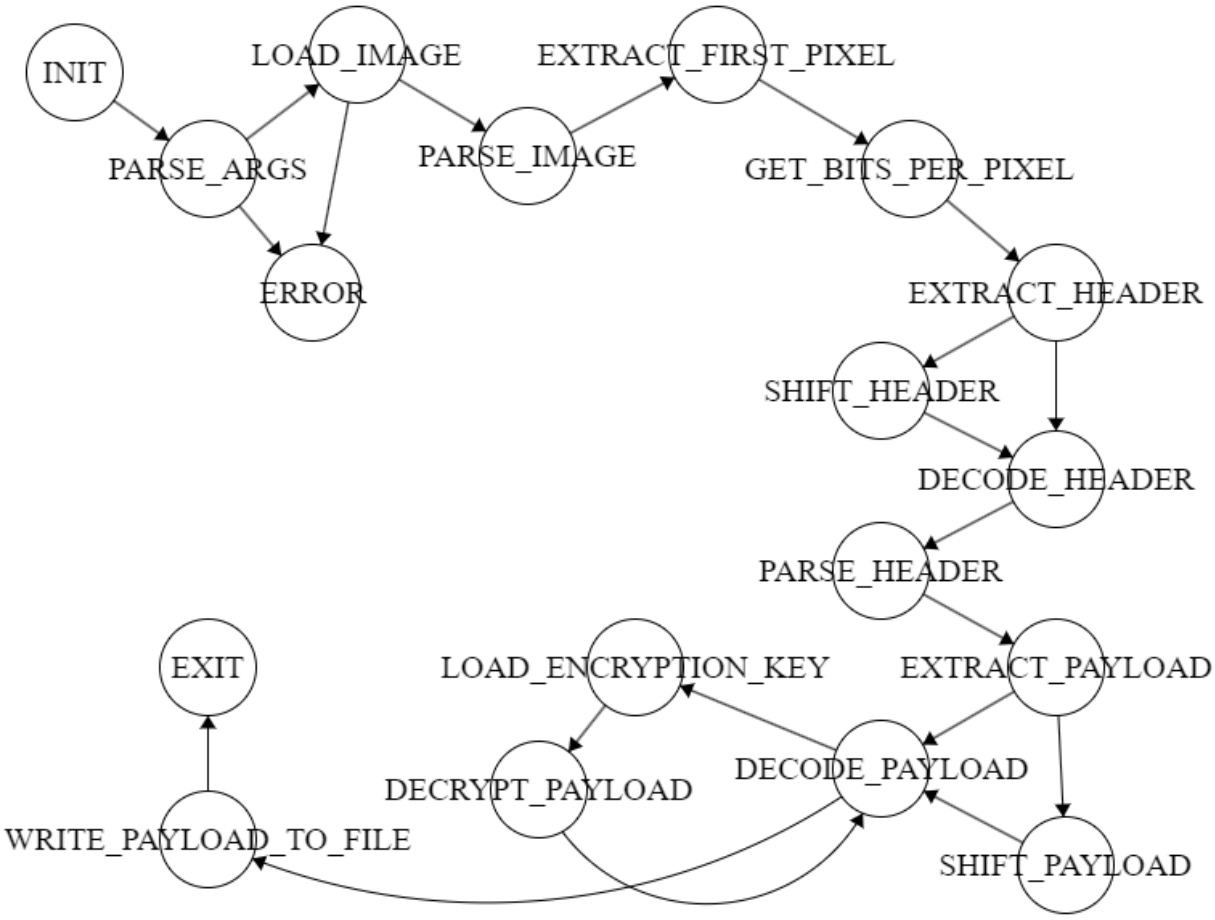
geoProgram3.py

Finite State Machine

State Table

INIT	PARSE_ARGS	Parse_args
PARSE_ARGS	LOAD_IMAGE	Load_image
PARSE_ARGS	ERROR	Argument_error
LOAD_IMAGE	ERROR	File_not_found
LOAD_IMAGE	PARSE_IMAGE	Parse_image
PARSE_IMAGE	EXTRACT_FIRST_PIXEL	Extract_first_pixel
EXTRACT_FIRST_PIXEL	GET_BITS_PER_PIXEL	Get_bits_per_pixel
GET_BITS_PER_PIXEL	EXTRACT_HEADER	Extract_header
EXTRACT_HEADER	SHIFT_HEADER	Shift_header
EXTRACT_HEADER	DECODE_HEADER	Decode_header
SHIFT_HEADER	DECODE_HEADER	Decode_header
DECODE_HEADER	PARSE_HEADER	Parse_header
PARSE_HEADER	EXTRACT_PAYLOAD	Extract_payload
EXTRACT_PAYLOAD	DECODE_PAYLOAD	Decode_payload
EXTRACT_PAYLOAD	SHIFT_PAYLOAD	Shift_payload
SHIFT_PAYLOAD	DECODE_PAYLOAD	Decode_payload
DECODE_PAYLOAD	LOAD_ENCRYPTION_KEY	Load_encryption_key
DECODE_PAYLOAD	WRITE_PAYLOAD_TO_FILE	Write_payload_to_file
LOAD_ENCRYPTION_KEY	DECRYPT_PAYLOAD	Decrypt_payload
DECRYPT_PAYLOAD	DECODE_PAYLOAD	Decode_payload
WRITE_PAYLOAD_TO_FILE	EXIT	Exit program

State Transition Diagram



Pseudocode

```
INITIALIZE fileWithData
parse command line arguments for fileWithData
READ fileWIthData as png imageWithPayload
DETERMINE imageWithPayload Width and Height
INITIALIZE END_OF_HEADER to (50*8)
INITIALIZE headerIndex, payloadIndex, endOfPayload, fileWidth, fileHeight to 0
INITIALIZE bitsPerPixel to 5
INITIALIZE mod to bitsPerPixel % 3
INITIALIZE div to ceil bitsPerPixel / 3
INITIALIZE header, payload, filetype, fileName to empty string
INITIALIZE encrypted to False
FOR value x in range(0, imageWithPayload width)
    FOR value y in range(0, imageWithPayload height)
        red, green, blue = imageWithPayload[x, y]
        IF x and y == 0 THEN
            EXTRACT last 2 bits of red
            EXTRACT last 2 bits of green
            EXTRACT last bit of blue
            bitsPerPixel = integer of (red + green + blue)
            mod = bitsPerPixel % 3
            div = ceil(bitsPerPixel / 3)
        ELSE IF headerIndex < END_OF_HEADER THEN
            header = header + red[-div to end]
            IF mod == 1 THEN
                IF div > 1 THEN
                    header = header + green[-(div-1) to end] + blue[-(div-1) to end]
                END IF
            ELSE IF mod == 2 THEN
                header = header + green[-div to end]
                IF div > 1 THEN
                    header = header + blue[-(div-1) to end]
                END IF
            ELSE
                header = header + green[-div to end] + blue[-div to end]
            END IF
            headerIndex += bitsPerPixel
            IF headerIndex >= END_OF_HEADER THEN
                INITIALIZE shift to headerIndex – END_OF_HEADER
                IF shift > 0 THEN
                    payload = payload + header[-shift to end]
                    payloadIndex += shift
                    header = header[start to -shift]
                END IF
```

```
                    DECODE header
                    PARSE header for endOfPayload, fileWidth, fileHeight, filename,
                        encrypted
            END IF
        ELSE IF payloadIndex < endOfPayload THEN
            payload = payload + red[-div to end]
            IF mod == 1 THEN
                IF div > 1 THEN
                    payload = payload + green[-(div-1) to end] + blue[-(div-1) to end]
                END IF
            ELSE IF mod == 2 THEN
                payload = payload + green[-div to end]
                IF div > 1 THEN
                    payload = payload + blue[-(div-1) to end]
                END IF
            ELSE
                payload = payload + green[-div to end] + blue[-div to end]
            END IF
            payloadIndex += bitsPerPixel
            IF payloadIndex >= endOfPayoad THEN
                shift = payloadIndex – endOfPayload
                IF shift > 0 THEN
                    payload = payload[start to -shift]
        ELSE
            break loop
        END IF
DECODE payload
IF encrypted THEN
    READ encryptionKey.key as KEY
    DECRYPT payload with KEY
ELSE
    CONVERT payload to BYTES
IF fileType == '.png' THEN
    WRITE payload as image
ELSE
    WRITE payload as file with fileName
```

Testing

Test Results

Command	Description	Result	Example
geoProgram1.py -f coverImage50x50.png -b 6	Calculate Maximum Payload size for coverImage50x50 using 6 bits per pixel	Maximum payload size 15000 bits	Example 1
geoProgram1.py	Call program 1 without arguments	Exit with Option Error	Example 2
geoProgram1.py -f fakefile.png -b 3	Call program 1 with nonexistent file name	Exit with missing file error	Example 3
geoProgram2.py -f coverImage200x300.png -b 1 -p payloadImage25x25.png	Embed payloadImage25x25.png into coverImage200x300 using 1 bit per pixel and no encryption	EmbeddedImage.png Created	Example 4, Example 5
geoProgram3.py -f EmbeddedImage.png	Extract 25x25 embedded image from previous command	Payload.png created	Example 5, Example 6, Example 7, Example 8
geoProgram2.py -f coverImage200x300.png -b 24 -p superLargeTextFile.txt	Embed large text file into coverImage200x300.png using 24bits per pixel	EmbeddedImage.png Created	Example 9
geoProgram3.py -f EmbeddedImage.png	Extract largeTextFile.txt from coverImage200x300.png	Payload.txt created	Example 10
geoProgram2.py -f coverImage1200x400.png -b 18 -p superLargeTextFile.txt	Embed largeTextFile into coverImage 1200x400.png using 18bits per pixel	EmbeddedImage.png Created	Example 11
geoProgram3.py -f EmbeddedImage.png	Extract largeTextFile.txt from coverImage1200x400.png using 18bits per pixel	Payload.txt created	Example 12
geoProgram2.py -f coverImage200x300.png -b 2 -p superSmallPDFFile.pdf	Embed superSmallPDFFile.pdf into coverImage200x300.png using 2 bits per pixel	EmbeddedImage.png Created	Example 13
geoProgram3.py -f EmbeddedImage.png	Extract pdf file from coverImage using 2bits per pixel	Payload.pdf created	Example 14
geoProgram2.py -f coverImage200x300.png -b 3 -p superLargePDFFile.pdf	Embed superLargePDFFile.pdf into coverImage200x300.png	Exit with error payload too large for cover image	Example 15

geoProgram2.py -f coverImage200x300.png -b 3 -p textToEncode1.txt -n	Embed textToEncode1.txt into coverImage 200x300 using 22 bits per pixel	EmbeddedImage.png Created	Example 16
geoProgram3.py -f EmbeddedImage.png	Extract and decrypt textToEncode1.txt	Payload.txt created	Example 17

Example 1

```
D:\Documents\Networking\Steganography>python geoProgram1.py -f coverImage50x50.png -b 6
Running Program 1.
Filename: coverImage50x50.png
Bits Per Pixel: 6
The Maximum Payload Size for coverImage50x50.png is: 15000 bits
```

Example 2

```
D:\Documents\Networking\Steganography>python geoProgram1.py
Running Program 1.
Filename: None
Bits Per Pixel: None
No fileName provided
```

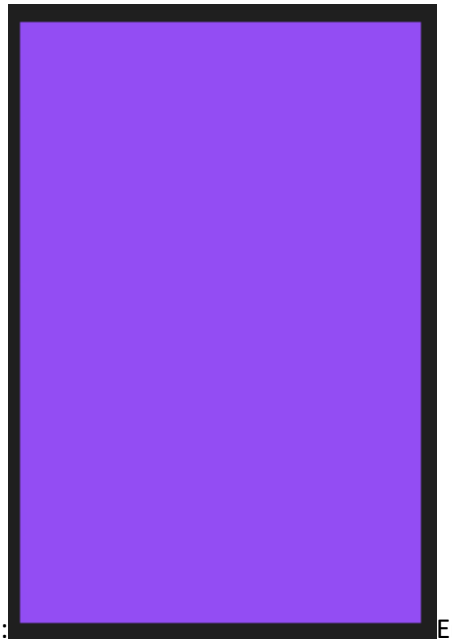
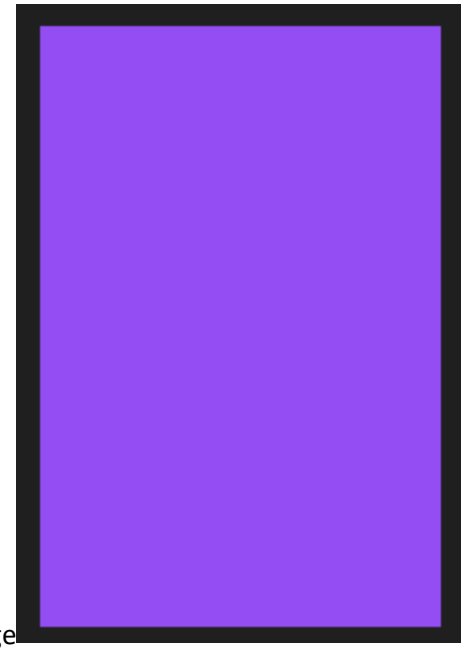
Example 3

```
D:\Documents\Networking\Steganography>python geoProgram1.py -f fakefile.png -b 3
Running Program 1.
Filename: fakefile.png
Bits Per Pixel: 3
Error opening image: fakefile.png
```

Example 4

```
D:\Documents\Networking\Steganography>python geoProgram2.py -f coverImage200x300.png -b 1 -p payloadImage25x25.png
Handling command line arguments...
Accessing cover image...
Constructing payload...
Encoding payload in base 2...
Creating payload header...
Payload file name longer than 15 characters. Swapping name with 'payload'
Required padding for header: 26
Padded header: 736$25$25$payload.png$n$111101000011011000110101 0$
Length of str header: 50
Encoded header: 0011011100110011001101100010010000110010001101010010010000110010001101010010010001110000011000010111100101101
1100011011110110000101100100001011100111000001101110011001110010010001101110001001000011000100110001001100010011000100110000
0011000100110000001100000011000000110000001100010011000100110000001100010011000100110000001100000011000000110001001100010011
0000001100010011000000110001001100000100100
Creating Binary Distributor...
Encoding bits per pixel into first pixel...
Bits per pixel: 1
Encoded bits per pixel: 00001
Red: 147 Green: 77 Blue: 243
Bits per pixel embedded in RGB
Red: 10010000 Green: 01001100 Blue: 11110011
Embedding payload into cover image...
Before Embedding: Red 10010011 Green 01001101 Blue 11110011
Encoding Next Chars: 0
After Embedding: Red 10010010 Green 01001101 Blue 11110011
```

Example 5

Cover Image:  Embedded Image 

Example 6

```
D:\Documents\Networking\Steganography>python geoProgram3.py -f EmbeddedImage.png
Handling command line arguments...
Extracting image data...
Extracting bits per pixel information...
Red: 10010000 Green: 01001100 Blue: 11110011
Bits per pixel: 1
Extracting header information...
Header index: 1 Bits per pixel: 1 End of header: 400
Header index: 2 Bits per pixel: 1 End of header: 400
Header index: 3 Bits per pixel: 1 End of header: 400
Header index: 4 Bits per pixel: 1 End of header: 400
Header index: 5 Bits per pixel: 1 End of header: 400
Header index: 6 Bits per pixel: 1 End of header: 400
Header index: 7 Bits per pixel: 1 End of header: 400
Header index: 8 Bits per pixel: 1 End of header: 400
Header index: 9 Bits per pixel: 1 End of header: 400
```

Example 7

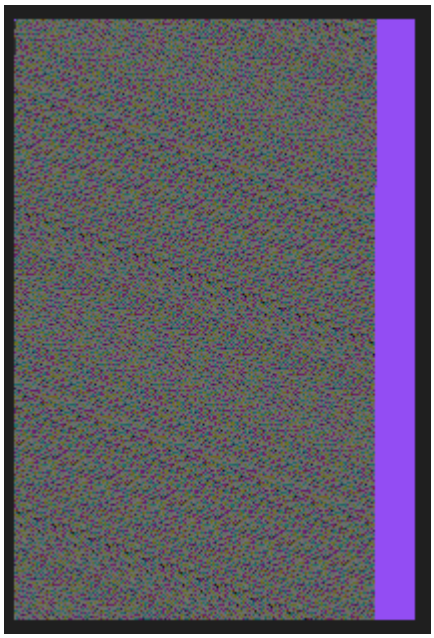
```
Header index: 400 Bits per pixel: 1 End of header: 400
Done extracting header.
Header in bytes: [55, 51, 54, 36, 50, 53, 36, 50, 53, 36, 112, 97, 121, 108, 111, 97, 100, 46, 112, 110, 103, 36, 110, 36, 4
9, 49, 49, 49, 48, 49, 48, 48, 48, 48, 49, 49, 48, 49, 49, 48, 48, 48, 49, 49, 48, 49, 48, 49, 48, 36]
Decoded header: 736$25$25$payload.png$n$111101000011011000110101 0$
Split header: ['736', '25', '25', 'payload.png', 'n', '111101000011011000110101 0', '']
Extracting payload information...
Payload index: 1 Bits per pixel: 1 End of payload: 736
Payload index: 2 Bits per pixel: 1 End of payload: 736
Payload index: 3 Bits per pixel: 1 End of payload: 736
Payload index: 4 Bits per pixel: 1 End of payload: 736
Payload index: 5 Bits per pixel: 1 End of payload: 736
Payload index: 6 Bits per pixel: 1 End of payload: 736
```

Example 8

```
Payload index: 736 Bits per pixel: 1 End of payload: 736
Done extracting payload.
Shift: 0
Decoded payload: b'\x89PNG\r\n\x1a\n\x00\x00\x00\rIHDR\x00\x00\x00\x19\x00\x00\x00\x19\x08\x02\x00\x00\x00K\x8b\x124\x00\x00
\x00#IDATx\x9cc\xfctx\x19\x03\x95\x00\x13\xb5\x0c\x1a5k\xd4\xacQ\xb3F\xcd\x1a5k\xd4\xacad\x16\x009\x00\x02\x8d0\x02O\xf2\x00
\x00\x00\x00IEND\xaeB`\x82'
```

Payload:  Extracted Image: 

Example 9

EmbeddedImage.png: 

```
Before Embedding: Red 10010011 Green 01001101 Blue 11110011
Encoding Next Chars: 01101100
Encoding Next Chars: 01100101
Encoding Next Chars: 01100011
After Embedding: Red 01101100 Green 01100101 Blue 01100011
```

Example 10

```
Payload index: 1297448 Bits per pixel: 24 End of payload: 1297584
Payload index: 1297472 Bits per pixel: 24 End of payload: 1297584
Payload index: 1297496 Bits per pixel: 24 End of payload: 1297584
Payload index: 1297520 Bits per pixel: 24 End of payload: 1297584
Payload index: 1297544 Bits per pixel: 24 End of payload: 1297584
Payload index: 1297568 Bits per pixel: 24 End of payload: 1297584
Payload index: 1297592 Bits per pixel: 24 End of payload: 1297584
Done extracting payload.
Shift: 8
Shifting payload
Decoded payload: b'Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et do
lore magna aliqua. Vel turpis nunc eget lorem dolor sed viverra. Posuere urna nec tincidunt praesent semper feugiat nibh sed
. Sapien pellentesque habitant morbi tristique senectus et. Eros donec ac odio tempor orci dapibus ultrices. Amet nulla faci
lisi morbi tempus iaculis urna. Vitae semper quis lectus nulla at volutpat diam ut. Phasellus vestibulum lorem sed risus ult
ricies tristique nulla. Nisl pretium fusce id velit. Dui id ornare arcu odio ut sem. Ullamcorper morbi tincidunt ornare mass
```

Steganography > payload.txt

```
1  Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt
2  Magna sit amet purus gravida. Egestas sed tempus urna et pharetra pharetra massa. Posuere
3  Integer feugiat scelerisque varius morbi enim nunc faucibus. Netus et malesuada fames ac.
4  At in tellus integer feugiat scelerisque varius morbi. Sodales ut eu sem integer vitae ju
5  Ac turpis egestas maecenas pharetra convallis posuere morbi. Pulvinar pellentesque habita
6  Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt
7  Magna sit amet purus gravida. Egestas sed tempus urna et pharetra pharetra massa. Posuere
8  Integer feugiat scelerisque varius morbi enim nunc faucibus. Netus et malesuada fames ac.
9  At in tellus integer feugiat scelerisque varius morbi. Sodales ut eu sem integer vitae ju
10 Ac turpis egestas maecenas pharetra convallis posuere morbi. Pulvinar pellentesque habita
11 Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt
12 Magna sit amet purus gravida. Egestas sed tempus urna et pharetra pharetra massa. Posuere
13 Integer feugiat scelerisque varius morbi enim nunc faucibus. Netus et malesuada fames ac.
14 At in tellus integer feugiat scelerisque varius morbi. Sodales ut eu sem integer vitae ju
15 Ac turpis egestas maecenas pharetra convallis posuere morbi. Pulvinar pellentesque habita
16 Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt
17 Magna sit amet purus gravida. Egestas sed tempus urna et pharetra pharetra massa. Posuere
18 Integer feugiat scelerisque varius morbi enim nunc faucibus. Netus et malesuada fames ac.
19 At in tellus integer feugiat scelerisque varius morbi. Sodales ut eu sem integer vitae ju
20 Ac turpis egestas maecenas pharetra convallis posuere morbi. Pulvinar pellentesque habita
21 Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt
22 Magna sit amet purus gravida. Egestas sed tempus urna et pharetra pharetra massa. Posuere
23 Integer feugiat scelerisque varius morbi enim nunc faucibus. Netus et malesuada fames ac.
24 At in tellus integer feugiat scelerisque varius morbi. Sodales ut eu sem integer vitae ju
25 Ac turpis egestas maecenas pharetra convallis posuere morbi. Pulvinar pellentesque habita
26 Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt
27 Magna sit amet purus gravida. Egestas sed tempus urna et pharetra pharetra massa. Posuere
28 Integer feugiat scelerisque varius morbi enim nunc faucibus. Netus et malesuada fames ac.
29 At in tellus integer feugiat scelerisque varius morbi. Sodales ut eu sem integer vitae ju
30 Ac turpis egestas maecenas pharetra convallis posuere morbi. Pulvinar pellentesque habita
31 Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt
32 Magna sit amet purus gravida. Egestas sed tempus urna et pharetra pharetra massa. Posuere
33 Integer feugiat scelerisque varius morbi enim nunc faucibus. Netus et malesuada fames ac.
34 At in tellus integer feugiat scelerisque varius morbi. Sodales ut eu sem integer vitae ju
35 Ac turpis egestas maecenas pharetra convallis posuere morbi. Pulvinar pellentesque habita
36 Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt
37 Magna sit amet purus gravida. Egestas sed tempus urna et pharetra pharetra massa. Posuere
38 Integer feugiat scelerisque varius morbi enim nunc faucibus. Netus et malesuada fames ac.
39 At in tellus integer feugiat scelerisque varius morbi. Sodales ut eu sem integer vitae ju
40 Ac turpis egestas maecenas pharetra convallis posuere morbi. Pulvinar pellentesque habita
41 Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt
42 Magna sit amet purus gravida. Egestas sed tempus urna et pharetra pharetra massa. Posuere
43 Integer feugiat scelerisque varius morbi enim nunc faucibus. Netus et malesuada fames ac.
44 At in tellus integer feugiat scelerisque varius morbi. Sodales ut eu sem integer vitae ju
45 Ac turpis egestas maecenas pharetra convallis posuere morbi. Pulvinar pellentesque habita
46 Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt
```

Example 11

EmbeddedImage.png

Example 12

```
Payload index: 1297562 Bits per pixel: 18 End of payload: 1297584
Payload index: 1297580 Bits per pixel: 18 End of payload: 1297584
Payload index: 1297598 Bits per pixel: 18 End of payload: 1297584
Done extracting payload.
Shift: 14
Shifting payload
Decoded payload: b'Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et do
lore magna aliqua. Vel turpis nunc eget lorem dolor sed viverra. Posuere urna nec tincidunt praesent semper feugiat nibh sed
. Sapien pellentesque habitant morbi tristique senectus et. Eros donec ac odio tempor orci dapibus ultrices. Amet nulla faci
lisi morbi tempus iaculis urna. Vitae semper quis lectus nulla at volutpat diam ut. Phasellus vestibulum lorem sed risus ult
ricies tristique nulla. Nisl pretium fusce id velit. Dui id ornare arcu odio ut sem. Ullamcorper morbi tincidunt ornare mass
a eget egestas purus viverra. Sit amet est placerat in egestas erat. Orci ac auctor augue mauris augue neque gravida. Faucib
```

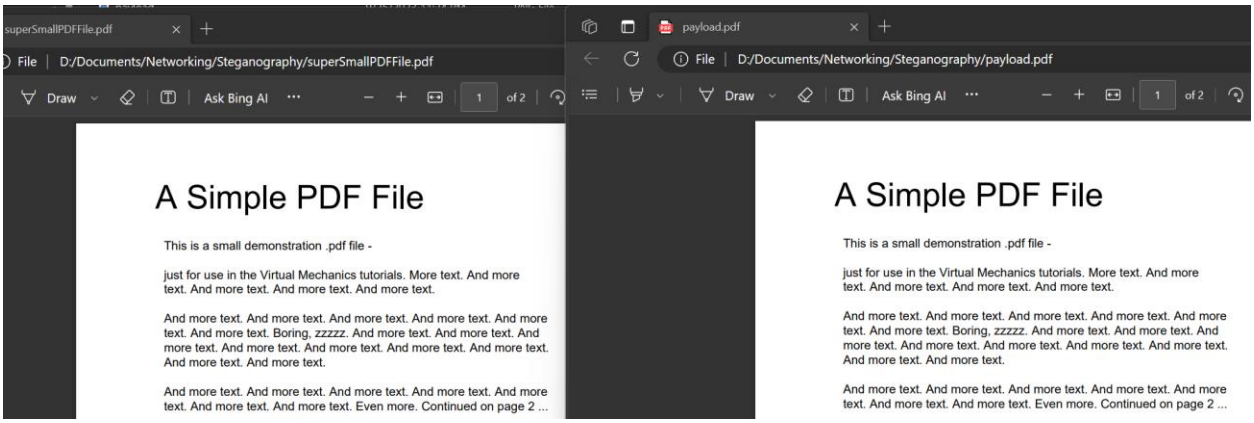
Steganography > payload.txt

```
1  Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt
2  Magna sit amet purus gravida. Egestas sed tempus urna et pharetra pharetra massa. Posuere
3  Integer feugiat scelerisque varius morbi enim nunc faucibus. Netus et malesuada fames ac.
4  At in tellus integer feugiat scelerisque varius morbi. Sodales ut eu sem integer vitae ju
5  Ac turpis egestas maecenas pharetra convallis posuere morbi. Pulvinar pellentesque habita
6  Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt
7  Magna sit amet purus gravida. Egestas sed tempus urna et pharetra pharetra massa. Posuere
8  Integer feugiat scelerisque varius morbi enim nunc faucibus. Netus et malesuada fames ac.
9  At in tellus integer feugiat scelerisque varius morbi. Sodales ut eu sem integer vitae ju
10 Ac turpis egestas maecenas pharetra convallis posuere morbi. Pulvinar pellentesque habita
11 Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt
12 Magna sit amet purus gravida. Egestas sed tempus urna et pharetra pharetra massa. Posuere
13 Integer feugiat scelerisque varius morbi enim nunc faucibus. Netus et malesuada fames ac.
14 At in tellus integer feugiat scelerisque varius morbi. Sodales ut eu sem integer vitae ju
15 Ac turpis egestas maecenas pharetra convallis posuere morbi. Pulvinar pellentesque habita
16 Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt
17 Magna sit amet purus gravida. Egestas sed tempus urna et pharetra pharetra massa. Posuere
18 Integer feugiat scelerisque varius morbi enim nunc faucibus. Netus et malesuada fames ac.
19 At in tellus integer feugiat scelerisque varius morbi. Sodales ut eu sem integer vitae ju
20 Ac turpis egestas maecenas pharetra convallis posuere morbi. Pulvinar pellentesque habita
21 Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt
22 Magna sit amet purus gravida. Egestas sed tempus urna et pharetra pharetra massa. Posuere
```

Example 13

```
Before Embedding: Red 10010011 Green 01001101 Blue 11110011
Encoding Next Chars: 0
Encoding Next Chars: 0
After Embedding: Red 10010010 Green 01001100 Blue 11110011
Before Embedding: Red 10010011 Green 01001101 Blue 11110011
Encoding Next Chars: 1
Encoding Next Chars: 1
After Embedding: Red 10010011 Green 01001101 Blue 11110011
```

Example 14



Example 15

```
D:\Documents\Networking\Steganography>python geoProgram2.py -f coverImage200x300.png -b 3 -p superLargeTextFile.txt
Handling command line arguments...
Accessing cover image...
Constructing payload...
Encoding payload in base 2...
Creating payload header...
Payload file name longer than 15 characters. Swapping name with 'payload'
Required padding for header: 24
Padded header: 1297584$0$0$payload.txt$n$011111001001000101001 01$
Length of str header: 50
Encoded header: 0011000100110010001110010011011100110101001110000011010000100100001100000010010000110000001001000111000001100
0010111100101101100011011110110000101100100001011100111010001111000011101000010010001101110001001000011000000110001001100011
0011000100110001001100010011000000110000001100010011000000110000001100010011000000110000001100010011000000110000010011
0000001100000011000100110000001100010010 0100
Creating Binary Distributor...
Bits to be sent: 1297984 Available bits: 180000
Selected payload is too large for selected cover image.Please increase the chosen bits per pixel or select a larger cover im
age
```

Example 16

```
D:\Documents\Networking\Steganography>python geoProgram2.py -f coverImage200x300.png -b 3 -p textToEncode1.txt -n
Handling command line arguments...
Accessing cover image...
Constructing payload...
Optional encryption selected. Encrypting payload...
Encryption key: b'UWvoIZF2ks66SKMXjjgyq8z1ttEANhiqJL2K8JgY1OU=' Successfully created...
The encrypted payload: b'gAAAAABlEolEfndRofT2BdkZhkAh5ZSz9aRo6cOwOQzzGLURGeeFdZIKYNWU3D6rEvX669gBO_rOXrD5GlQMZwR3ki21Bt0EKh2
fqJjpdX1QgBhsYGulHCxP_ylhp6B7SAmG11QiZM_qlZ9D2fqamKg-mMPEh1Z3plNn9RIS8jPsCB1uSfVczq8='
Encoding payload in base 2...
Creating payload header...
Payload file name longer than 15 characters. Swapping name with 'payload'
Required padding for header: 27
Padded header: 1472$0$0$payload.txt$y$1000011010111011000011100$
Length of str header: 50
Encoded header: 00110001001101000011011100110010001001000011000000100100001100000010010001110000011000010111100101101100011
0
1111011000010110010000101110011101000111100001110100001001000111100100100100001100010011000000110000001100000011000000110001
0011000100110000001100010011000000110001001100010011000100110001001100000011000100110001001100000011000000110000001100000011
0001001100010011000100110000001100000100100
Creating Binary Distributor...
Encoding bits per pixel into first pixel...
Bits per pixel: 3
Encoded bits per pixel: 00011
Red: 147 Green: 77 Blue: 243
Bits per pixel embedded in RGB
Red: 10010000 Green: 01001101 Blue: 11110011
Embedding payload into cover image...
```

Example 17

```
Payload index: 1469 Bits per pixel: 3 End of payload: 1472
Payload index: 1472 Bits per pixel: 3 End of payload: 1472
Done extracting payload.
Shift: 0
Decoded payload: b'gAAAAABlEolEfndRofT2BdkZhkAh5ZSz9aRo6cOwOQzzGLURGeeFdZIKYNWU3D6rEvX669gBO_rOXrD5GlQMZwR3ki21Bt0EKh2fqJjpd
X1QgBhsYGulHCxP_ylhp6B7SAmG11QiZM_qlZ9D2fqamKg-mMPEh1Z3plNn9RIS8jPsCB1uSfVczq8='
Encryption was used. Reading encryption key...
Initializing decrypter
Payload decrypted.
Decrypted payload: b'helloworld\r\nhelloworld\r\nhellomynameisgeodude\r\nlv99geodude\r\ngeodude'
```

User Guide

Installing

Pip install Pillow

Pip install cryptography

Running

Python geoProgram1.py -f <target coverImage.png> -b <bits per pixel>

Python geoProgram2.py -f <target coverImage.png> -b <bits per pixel> -p <target payload file> -n <Optional Encryption Flag>

Python geoProgram3.py -f <Target EmbeddedImage.png>

Findings

Steganography, often referred to as the art of hidden communication, is a fascinating practice that involves concealing secret information within seemingly innocuous carriers, such as images, audio files, or text, in a way that goes unnoticed by casual observers. Unlike cryptography, which focuses on encrypting messages to make them unreadable, steganography takes a different approach by ensuring that the very existence of a hidden message remains concealed. This covert technique has been employed throughout history for various purposes, from espionage and data protection to artistic expression and privacy preservation. The purpose of this assignment is to get familiar with the concept and implementation of steganography, as well as to implement our own least significant bit encoding algorithm.

The main point of struggle in coding the programs for this assignment came in managing the indexing of the binary string's associated to the header and payload, and accurately applying necessary padding or shifting to isolate the header and payload individually. I spent the large majority of my time fiddling with indexes to ensure that my algorithm for embedding and extracting data from the cover image worked within all required parameters.

When it came to optionally applying encryption, I found a Fernet encryption algorithm that accepts and encodes data in binary bytes which maintains the data format I was already using during embedding and extraction. This made it exceptionally easy to add encryption as an optional extra layer to my payload constructors.

For my LSB Encoding algorithm I would evenly distribute bits in priority of red->green->blue. Example using 4 bits per pixel: embed 2 bits in red 1 bit in green and 1 bit in blue. To decode my payload from program 3 I read the bits per pixel used from the first pixel at 0, 0 then use that data to adjust my algorithm to accurately extract the rest of the header and payload using the same bits per pixel that was used to encode them.