```python
"""
This project has been developed by Sonal Sarin with help from some online tutorials.
References to the online tutorials will be listed below.
This project takes an image and a text and embeds the text inside the image and saves the
embedding in a lossless file. The project uses the bottom
right 11 pixels to encrypt the text length and the rest of the pixels to encrypt the
text. The embedding is saved in an "testImage.png" file.

References:
http://ijact.org/volume3issue4/IJ0340004.pdf
http://www.aaronmiller.in/thesis/
http://www.lia.deis.unibo.it/Courses/RetiDiCalcolatori/Progetti98/Fortini/lsb.html
https://pillow.readthedocs.io/en/4.1.x/reference/Image.html
https://docs.python.org/2/library/re.html
https://docs.python.org/2/library/functions.html
Code for the embedding and extracting is inspired by:
https://github.com/rnikoopour/TextInImage

"""




import re
from PIL import Image


#This function will convert the text to binary
def ConvertTextToBinary(text):
    string = '' #Will hold the converted text to binary
    for char in text: #traverse through the text
        letterInt = ord(char) #convert each character in the string to integer
        letterBinary = format(letterInt, 'b') #Convert the character to binary
        letterBinary = ('0' * (8-len(letterBinary))) + letterBinary
        string += letterBinary

    return string #returns the converted text that is in binary


#This function will convert the text length to binary
def ConvertTextLengthToBinary(text):
    #print("Text Length: %s" % len(text))
    eightBit = format(8*len(text), 'b')
    t2Bit = ('0' * (32 - len(eightBit))) + eightBit
    #print("Text Length is: %s" % t2Bit)
    return t2Bit


#This function will embed the text in the image
def EmbedInImage(imageData, TextToEmbed, start):
    TextToEmbed = list(TextToEmbed) #put the text in a list
    ImageData2 = []

    while TextToEmbed: #traverse through the text list
        red, green, blue = imageData[start] #assign rgb the corresponding pixel (start)
        start +=1
        red2 = list(format(red, 'b')) #Format the pixel of the red color to binary
        red2[-1] = TextToEmbed[0]
        red2 = ''.join(red2)
        red = int(red2, 2)
        TextToEmbed.pop(0) #remove it
        if not TextToEmbed:
            ImageData2.append((red, green, blue)) #append the values
            break
        green2 = list(format(green, 'b')) #Format the pixel of the green color to binary
        green2[-1] = TextToEmbed[0]
        green2 = ''.join(green2)
        green = int(green2, 2)
        TextToEmbed.pop(0) #remove it
        if not TextToEmbed:
            ImageData2.append((red, green, blue)) #append the values
            break
        blue2 = list(format(blue, 'b')) #Format the pixel of the blue color to binary
        blue2[-1] = TextToEmbed[0]
        blue2 = ''.join(blue2)
        blue = int(blue2, 2)
        TextToEmbed.pop(0)
        ImageData2.append((red, green, blue)) #append it

    return (ImageData2, start)


#In this function, we will embed the text and length. The function will accept the
Binarytext, Binarylength, and the imagedata and embed it.
def EmbedTextAndLength(TextToBinary, TextLengthToBinary, ImageData):

    ImageData2 = []
    NewImageData, textLengthEnds = EmbedInImage(ImageData, TextLengthToBinary, 0)
    ImageData2 += NewImageData
    NewImageData, textEnds = EmbedInImage(ImageData, TextToBinary, 11)
    ImageData2 += NewImageData
    ImageData2 += imageData[textEnds:]
    return ImageData2


#In this function we will get the lsb of each rgb value
def getlastbit(col):
    bit = list(format(col, 'b')) #Format the value to binary
    return bit[-1] #retrive the last item in the bit (lsb)


#In this function we will be extracting the text length. The function will take in the
imageData of the image

def extractTextLength(ImageData):
    bit = ''
    start = 0
    quit = start + 32
    while start <= quit:
        red, green, blue = ImageData[start]
        start+=1
        bit += getlastbit(red)
        bit += getlastbit(green)
        bit += getlastbit(blue)

           #Save the binary values into a variable to be converted to integer
    Binarylength = bit[:32]
    length = int(Binarylength, 2)
    return length







#In this function we will extract the text. The function will take in the imagedata as
well as the length of the text
def extractText(imageData, length):
    bit = '' #Here we are defining the bit variable to hold the value of the lsb bit

    quit = 11 + length #We are going to be traversing through the while loop until the
end of text starting from the 11th pixel
    start = 11
    while start <= quit: #loop
        red, green, blue = imageData[start] #Set the red, green, and blue values to the
corresponding pixel value of the traversal
       # green = imageData[start]
       # blue = imageData[start]
        start += 1 #increment start so the following iteration will move to the next
pixel
        bit += getlastbit(red) #Get the last bit of the red, green, and blue values
        bit += getlastbit(green)
        bit += getlastbit(blue)


    BinaryText = bit[:length] #After the while loop is complete we want to assign
BinaryText the values of bit
    characters = re.findall("........", BinaryText)


    #In this for loop, we will be getting each character from the binary values and
convert those values to characters
    text = ''
    for char in characters:
        letter = int(char, 2)
        text += chr(letter)

    return text





#This is the start of the program. It will ask the user to encrypt or decrypt the file.
Input = input("Would you like to encrypt (e) or decrypt (d) the file? ")


if Input == 'e':
    image_path = input("Enter the image path of the image you would like to encrypt: ")
    print("You wished the encrypt this image: %s" % image_path)

    choice = input("Would you like to encrypt a file (f) or a string (s)? ")
    text = ''

    #In this if else statement, the user is able to encrypt a file or a string
    if choice == "f":

        text = input("Enter the file you would like to encrypt: ")
        print("You wished to encrypt this file: %s" % text)

        file = open(text, "r")
        text = file.read()

    else:
        text = input("Enter the string you would like to encrypt: ")
        print("You wished to encrypt this string: %s" % text)



    #Now we want to open the image
    image = Image.open(image_path)
    imageData = list(image.getdata())
    #print (len(imageData))



    #Here we will check whether the text entered is greater than the image data
    #If it is, then we want to gracefully exit the program otherwise, we will continue
execution
    if len(text) < len(imageData):

        #Now we will convert the text length to binary
        textLengthToBinary = ConvertTextLengthToBinary(text)

        #Now we will convert the text to binary
        textToBinary = ConvertTextToBinary(text)

        #Now we will embed the text and text length in the image
        EmbeddedImage = EmbedTextAndLength(textToBinary, textLengthToBinary, imageData)
        image.putdata(EmbeddedImage)

        #Now we will save the embedded data into a png file called "EmbeddedImage.png"
        image.save("testImage.png", 'PNG')

        print("Embedding completed in the \"testImage.png\" file")
        print("To extract the text, run the program again and choose 'd' to decrypt the
file and enter")
        print("\"testImage.png\" to extract the data")



else:

    #This else condition will execute if the user wishes to extract the data from the
image

    image_path = input("Enter the image path of the image you would like to decrypt: ")
    #print(image_path)
```

```
#Open the image and get it's data
image = Image.open(image_path)
imageData = list(image.getdata())


#Now we will extract the length from the image
length = extractTextLength(imageData)
#print("Length is: %s" % length)

#Now we will extract the text
extractedText = extractText(imageData, length)
print("The extracted text is: %s" % extractedText)


print("Extraction Complete! Yay!")
```