# Final Project

## version

**Miguel Angel Guerra Rangel | Alejandra Nohemí Tamez | Daniel Adrian Lozano Garza | Sebastian Gonzalez Medellin**

diciembre 06, 2020

# Contents

# Documentation of Project

Here you will find information of our project modules.

## Descritpion

*The code only works with parameters*.

The project has different modules with which you can to do different things, here we explain them:

### *1. Encode Messages*

*Our code include two encoding forms, transposition and Cesar*.

*The first one works taking your message and key (string), it use both to make a matrix, and change the columns for rows*.

*The second one works taking your message and key (number), it use the key to move of place each word*.

#### *Example(s)*

cmd: Application.py -opc 1 -lang 1 -msg “This is my message in english” -t_cifr 1 -rot 13

cmd: Application.py -opc 1 -lang 2 -msg “Este es mi mensaje en español” -t_cifr 2 -key “keyword”

### *2. Unencode Messages*

*Our code include two unencoding forms, transposition and Cesar*.

*The first one works taking your message and key (string), it use both to make a matrix, and change the columns for rows*.

*The second one works taking your message and key (number), it use the key to move of place each word*.

#### *Example(s)*

cmd: Application.py -opc 2 -lang 1 -msg “guvf vf zl zrffntr va ratyvfu” -t_cifr 1 -rot 13

cmd: Application.py -opc 2 -lang 2 -msg “e s lsmaetijse ep m aeeeñsnno” -t_cifr 2 -key “keyword”

### *3. Hack Messages*

*Our code include two hacking forms, transposition and Cesar*.

*The first one works taking your message, and try with differents keys, when finally create the message, verify if it is in english or spanish to know if unencode the message satisfactorily*.

*The second one works taking your message, and try with each number from 1 to 25, we take the number like probably key, when finally create the message, verify if it is in english or spanish to know if unencode the message satisfactorily*.

#### *Example(s)*

cmd: Application.py -opc 3 -lang 1 -msg “guvf vf zl zrffntr va ratyvfu” -t_cifr 1

cmd: Application.py -opc 3 -lang 2 -msg “e s lsmaetijse ep m aeeeñsnno” -t_cifr 2

### *4. Investigate Organizations*

*If you want to use this option, you have to create an api key in github*:

*Our code need two data, your api key and the domain or organization you want to investigate. The code generate a txt file with the found information and also print it in the terminal window*.

### *Example(s)*

cmd: Application.py -opc 4 -apikey ‘31mn93abbx811o05q119lDp1mms931ml5c31jjj7’ -domain “www.telmex.com”

cmd: Application.py -opc 4 -apikey ‘31mn93abbx811o05q119lDp1mms931ml5c31jjj7’ -domain “telmex.com”

cmd: Application.py -opc 4 -apikey ‘31mn93abbx811o05q119lDp1mms931ml5c31jjj7’ -domain “telmex”

## *5. Send Emails*

The code take the next data:

1. Your email account
2. Your password
3. The email of your friend
4. The subject
5. The message
6. The name of your picture and its directory (optional)

*Our code only accept Gmail or Outlook account*(*for the user*).

### *Example(s)*

cmd: Application.py -opc 5 -t_email 1 -email “example@gmail.com” -passw “yourpassword” -to “example2@account.com” -subj “Subject:” -msg “Your message” -pic “Picture.jpg”,”C:UsersUserNameDocuments”

cmd: Application.py -opc 5 -t_email 2 -email “example@outlook.com” -passw “yourpassword” -to “example2@account.com” -subj “Subject:” -msg “Your message” -pic “Picture.jpeg”,”C:UsersUserNameDocuments”

cmd: Application.py -opc 5 -t_email 1 -email “example@gmail.com” -passw “yourpassword” -to “example2@account.com” -subj “Subject:” -msg “Your message” -pic “Picture.jpg”,”C:UsersUserNameDocuments”

cmd: Application.py -opc 5 -t_email 2 -email “example@outlook.com” -passw “yourpassword” -to “example2@account.com” -subj “Subject:” -msg “Your message”

## *6. Verify Open Ports*

For this option is necesary you add the ip, and port. The code print in the terminal window the result.

### *Example(s)*

cmd: Application.py -opc 6 -ip “192.168.1.12” -port 8080

## *7. Get Your DNS*

*This option was programmed in PowerShell, You do not need to open this file, you can run the cmd: Application programmed in python, you would only have to add the parameter ‘-opc 7’*

*The code generate two files txt. In both save the same information, but it use different commands in consequence the information is organized of different form*

### *Example(s)*

cmd: Application.py -opc 7

# Modules

## *Python*

### *Message*

1. spanisDictionary()
2. englishDictionary()
3. setLanguage(language)
4. getLanguage()
5. setMessage(message)
6. getMessage()
7. messageWords(sentece)
8. verifyLanguage(sentence)

#### *spanishDictionary()*

We save a list with words in spanish.

We toke the words of a file in txt, we saved that like ‘dictEsp.txt’.

#### *englishDictionary()*

We save a list with words in english.

We toke the words of a file in txt, we saved that like ‘dictEng.txt’.

#### *setLanguage(language)*

We recieve an input data of int type, and we use that to know in what language will be wrote, it can be in English or Spanish.

Finally we save a boolean in self.language.

#### *getLanguage()*

We return the boolean.

#### *setMessage(message)*

We recieve the message wrote by the user, and we check if that is a valid message.

#### *getMessage()*

We return the message.

#### *messageWords(sentece)*

We recieve a message and we save each word of the sentence in a list.

Finally, we return a list with the words.

#### *verifyLanguage(sentence)*

We recieve a sentence and we verify if this match with a language.

### *Cifrado Cesar*

1. verifyKey()
2. codifyMessage()
3. codifyMessage()
4. decodeMessage()
5. hackMessage()

### *verifyMode()*

The module recieve an input data of type int, and we take it to verify which is it, this is necesary because the module include the next modes:

1. Encoding
2. Unencoding
3. Hacking

### *verifyKey()*

The module recieve a key of int type, and we verify that to compire with ‘verifyMode()’function because, only the mode 1 and 2 need a key.

### *codifyMessage()*

This is the number one mode, we encode the message with the Cesar encoding.

### *decodeMessage()*

This is the number two mode, we unencode the message with the Cesar encoding.

### *hackMessage()*

This is the number three mode, we hack the message trying with numbers from 1 to 25, each number represents a word of the alphabet.

## *Transposition*

1. setKey(key)
2. getKey()
3. encodeMessage()
4. unencodeMessage()
5. hackMessage()

### *setKey(key)*

We recieve a key of string type, and we take the length of the key and we save that in a variable.

### *getKey()*

We return the length of the key

### *encodeMessage()*

We make the transposition to encode the message, taking the key, and the message.

### *unencodeMessage()*

We make the transposition to unencode the message, taking the key, and the message.

### *hackMessage()*

We make the transposition tu hack the message, but in this case we do not use the message.

### *SendEmail*

1. setOpc(opc)
2. getOpc()
3. setEmailAccount(email_account)
4. getEmailAccount()
5. setPassword(password)
6. getPassword()
7. setTo(to)
8. getTo()
9. setPicture(picture, directory)
10. getPicture()
11. sendEmail(subject, msg)

### *setOpc(opc)*

We use this function to verify if the option is valid (it has to be between 1 and 2).

### *getOpc()*

If the a valid option we return it.

### *setEmailAccount(email_account)*

The user add an email, but only accept two domains to send emails ‘Gmail’ and ‘Outlook’, so, we have to be sure the user add a valid email.

If that is a valid email, we save the email in a variable.

### *getEmailAccount()*

We return the emails.

### *setPassword(password)*

The domain specify some characteres in their passwords, so, we check the password added for the user.

### *getPassword()*

If the password is valid, we return that.

### *setTo(to)*

We check the email account where we will send the email.

### *getTo()*

We return the email

### *setPicture(image, directory)*

In this module, we take the name of the picture, and the directory where the picture is located, with this data, the save it in a list

### *getPicture()*

We return the list

### *sendEmail(subject, msg)*

The user add the subject and the message here, and taking the rest of the data, we send the email.

## *Hunter*

1. setAPIKey(apikey)
2. getAPIKey()
3. setDomain(domain)
4. getDomain()
5. search()
6. showInfo(results)
7. saveInfo(results)

### *setAPIKey(apikey)*

We check if the api key is valid.

### *getAPIKey()*

We return the api key

### *setDomain(domin)*

The domain can just have some specific characteres, so, we check it.

### *getDomain()*

We return the domain

### *search()*

We search the information of the domain.

### *showInfo(results)*

We take the information we got, and we print that in the terminal

### *saveInfo(results)*

We take the information we got, and we save that in the terminal.

## *Socket*

1. setIP(ip)
2. getIP()
3. checkPortSocket(port)

### *setIP(ip)*

We verify is the ip is valid.

### *getIP()*

We return the ip.

### *checkPortSocket(port)*

We check the open ports.

## *PowerShell*

### *DNS*

This code verify the dns of the user.