Proyecto 1

Cifrado de Hill

Información del curso

Criptografía y Seguridad - Facultad de Ciencias, UNAM.

- Profesor: Criptografía y Seguridad
- Ayudante: Gerardo Rubén López Hernández
- Laboratorio: José Canek García Aguilar

Descripción de la práctica

Implementar el algoritmo de Hill para cifrar y descifrar un mensaje.

En el archivo de especificación de la prática viene todo explicado `doc.pdf`.

Entorno

- **`OS`**: Ubuntu 18.04.2 LTS o macOS Mojave 10.14.2
- **`Python`**: Python 3.7.0
- **`pip3`**: pip 18.0
  - **`sympy`**: sympy==1.3
  - **`numpy`**: numpy==1.16.1

Ejecución del programa

Se creo un archivo `Makefile` para facilitar la preparación del entorno y ejecutar el programa.

Se requiere tener el binario `make` instalado. Si estás en Ubuntu basta con poner.

```
$ sudo apt install make
```

En caso de no estar en Ubuntu, instalar `make` con tu manejador de paquetes.

Antes de proceder a ejecutar el programa, se requiere tener instalado `pip3` (un manejador de paquetes de Python 3) y sus respectivas dependencias. Para esto, lo automaticé con un comando que instala `pip3` (en caso de no estar instalado) y las dependencias necesarias (ver `requirements.txt`).

Para instalar las dependencias necesarias (en Ubuntu) se procederá a ejecutar el siguiente comando con privelegios de administrador (ósea `sudo`).

```
$ sudo make prepare-env
```

Si desconfías de esto proceso (por el hecho de hacerlo como administrador) o tienes otra distribución de Linux, las dependencias necesarias con `pip3`son instaladas de la siguiente manera:

```
$ pip3 install -r requirements.txt
```

Una vez ya configurado nuestro entorno de trabajo procederemos a ejecutar la aplicación.

Para limpiar el proyecto se deberá ejectar el comando:

```
$ make clean
```

Para probar el cifrado con unos *test cases* sobrepuesto en el código, se debe hacer de la siguiente manera

```
$ make run
```

Comentarios

Ver las versiones usadas para no crear conflictos.

Los casos de prueba solo están sobrepuestos en el código.

Se utilizó `numpy` para hacer operaciones sobre matrices, específicamente para saber si una matriz tiene inversa y `sympy` para encontrar la inversa módulo n de una matriz.

Integrante(s)

- Ángel Iván Gladín García - *(angelgladin@ciencias.unam.mx)*