

# Práctica de cifrado

Algoritmo de sustitución polialfabética. Cifrado Vigenére.

Profesor: Carlos Fernández Lozano

Fecha propuesta: marzo-2020

Última revisión: 25 Marzo 2020

## Objetivos:

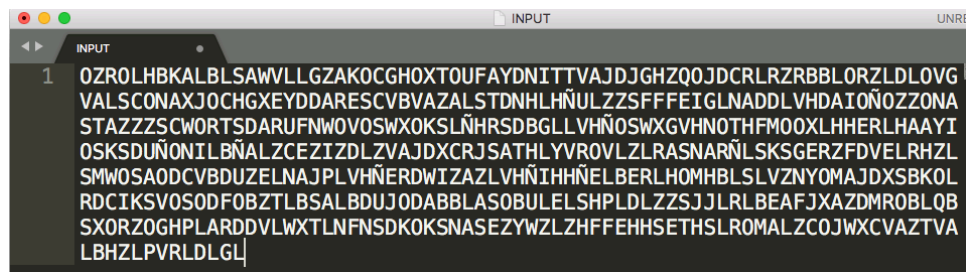
1. Utilizar de manera práctica una herramienta que permita cifrar, descifrar y criptoanalizar el cifrado de Vigenére. Se trata de un algoritmo de sustitución polialfabética que utiliza todos los caracteres del diccionario (independientemente de el diccionario que se decida utilizar).
2. Existen diferentes maneras de criptoanalizar el algoritmo y tratar de romper su clave, una de ellas es el método de Kasiski (el que implementa la herramienta software que usaréis después a modo de ejemplo).
3. Una vez se han realizado diferentes acciones de cifrado, descifrado y criptoanálisis con la herramienta software que se propone, ya se está en disposición de realizar vuestra propia implementación en Python.
4. Existen multitud de repositorios en GitHub con implementaciones del cifrado y descifrado de Vigenére en Python (en general de cualquier algoritmo de sustitución polialfabética). Se pide identificar aquel que os parezca más prometedor para construir vuestra propia implementación de un posible método de criptoanálisis al algoritmo de Vigenére.
5. Se desarrollará una COMPETICIÓN y se otorgará parte de la nota en función del tiempo que tarde cada implementación en resolver tres pruebas. El juego de pruebas será privado del profesor ;)

## Condiciones de implementación del algoritmo de criptoanálisis:

1. Debe poder ejecutarse por línea de comandos

**Uso: vigenere.py [-i, [INPUT]] [-d, [DICTIONARY]] [--hash, [HASH]]**

- a. INPUT: fichero con texto de entrada
- b. DICTIONARY: fichero con el diccionario a utilizar
- c. HASH: fichero que contenga un hash para comparar
- d. Cualquiera de los ficheros anteriores debe contener cadenas de texto, sin espacios, cabeceras, etc. Para que se pueda parsear sin problema. Ejemplo de fichero INPUT:

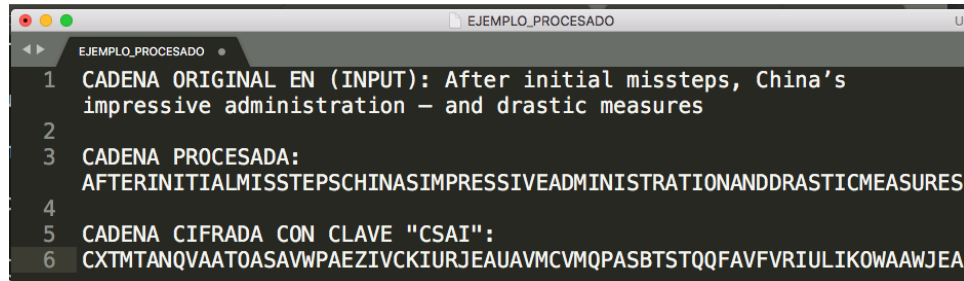


```
1 OZROLHBKALBLSAWVLLGZAKOCGH0XTOUFAYDNITTVAJDJGHZQJDCRLRZRBBLORZLDLOVG
VALSCONAXJOCHGXKEYDDARESCVBVAZALSTDNHLHÑULZZSFFFEIGLNADDLVHDAI0ÑOZZONA
STAZZZSCWORTSDARUFNW0V0SWXOKSLÑHRSDBGLLVHÑ0SWXGVHNOTHFM00XLHHERLHAAYI
OSKSDUÑONILBÑALZCEZIZDLZVAJDXCRJSATHLYVROVLZLRASNARÑLSKSGERZFDVELRHZL
SMWOSA0DCVBDUZELNAJPLVHÑERDWIZAZLVHÑIHHÑELBERLHOMHBLSLVZNYOMAJDXSBKOL
RDCIKSV0SODFOBZTLBSALBDUJODABBLASOBULELSHPLDLZZSJ1LRLEAFJXAZDMROBLQB
SX0RZ0GHPARDDVLWXTLNFNSDKOKSNASEZYWZLZHFFEHHSETHSLROMALZC0JWXCVAZTVA
LBHZLPVRLDLGL
```

2. A partir de un fichero con texto cifrado de entrada (INPUT) y de un fichero de texto con el diccionario (DICTIONARY) que se debe utilizar, vuestra implementación debe criptoanalizar el

texto y proponer una clave. Para saber si vuestro criptoanálisis es correcto deberéis calcular el hash del texto descifrado y compararlo con el que obtendréis del fichero (HASH).

3. El HASH se calculará con la librería hashlib.py (<https://docs.python.org/3.8/library/hashlib.html>) y un SHA256
4. Para reducir las posibilidades el texto solamente podrá estar en castellano, inglés y francés. En caso de que el idioma use símbolos como por ejemplo el inglés y sus ' y – se debe procesar la cadena de INPUT y eliminarlos, de esta manera el HASH será coincidente siempre. Una buena idea sería procesar la cadena de INPUT buscando ese tipo de símbolos y eliminarlos, igual que los espacios en blanco, comas, puntos, paréntesis, etc. Por ejemplo



```
EJEMPLO_PROCESADO
1 CADENA ORIGINAL EN (INPUT): After initial missteps, China's
  impressive administration – and drastic measures
2
3 CADENA PROCESADA:
  AFTERINITIALMISSTEPSCHINASIMPRESSIVEADMINISTRATIONANDDRASTICMEASURES
4
5 CADENA CIFRADA CON CLAVE "CSAI":
6 CXTMTANQVAATOASAVWPAEZIVCKIURJEAUAVMCMQPASBTSTQQFAVFVRIULIKOWAAWJEA
```

5. Por supuesto se puede configurar la herramienta con muchas más opciones, pero las que os propongo son las mínimas y obligatorias
6. La salida debe ser la palabra clave encontrada

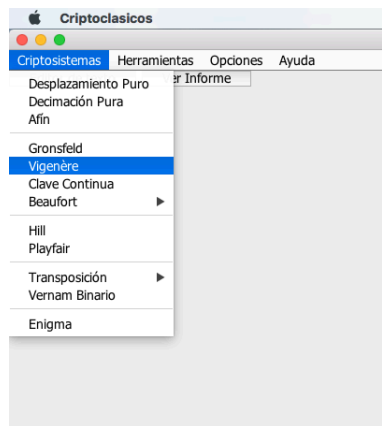
### ***Ejercicio práctico para comprender Vigenére.***

1. Descarga la herramienta software, desarrollada en Java Criptoclásicos v2.1.

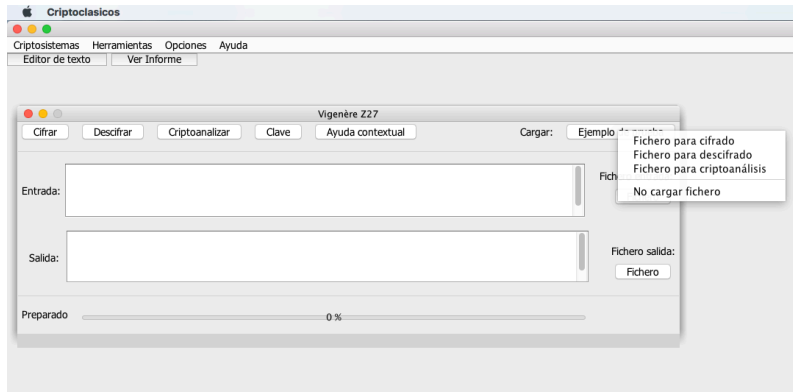
<http://www.criptored.upm.es/descarga/Criptoclasticosv2.1.jar.zip>

SHA256: B1CF2F610C3E04302BA5B63A3958389283818C4EB0BAEB413392651454D87F9C (actualizado 28/02/19)

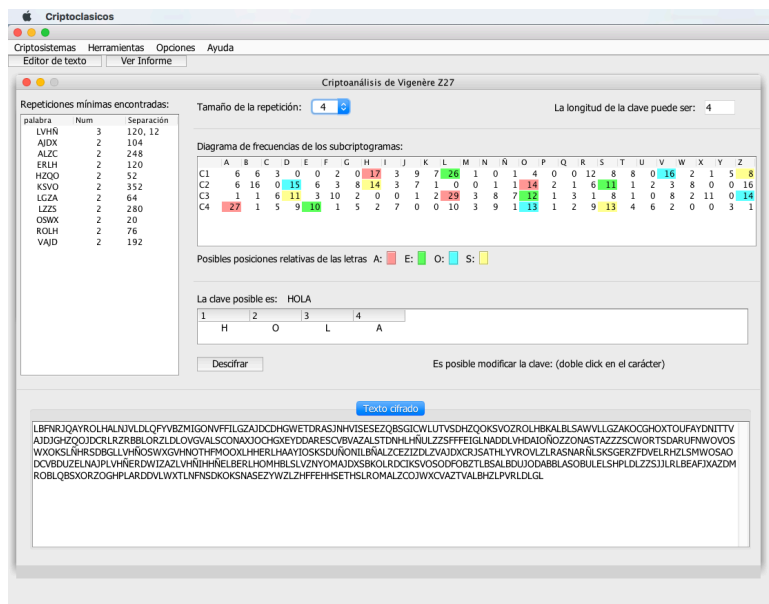
2. Al abrir accede al menú de cifrado Vigenére.



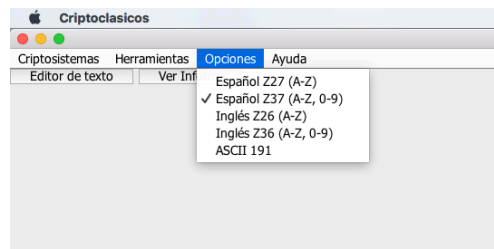
3. Realiza diferentes acciones de cifrado, descifrado y criptoanálisis. La herramienta dispone de ficheros precargados para cada acción.



4. Al cargar el fichero para el criptoanálisis pulsa sobre el botón de criptoanalizar, la salida será la siguiente:

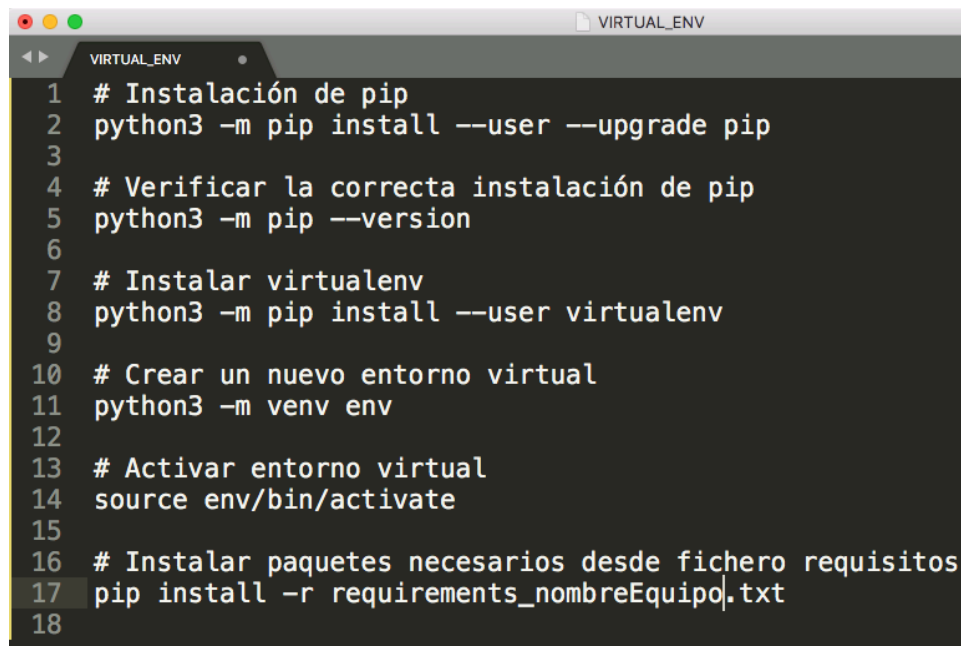


5. Repite las mismas acciones, con los mismos textos o ficheros de prueba pero cambia el diccionario:



### Entrega:

1. Realiza una memoria en la que describas (30% de la nota):
  - a. Nombre del equipo e integrantes. (Para evitar confusiones, mismos grupos que para la práctica de abastecido de servidor).
  - b. Implementación del algoritmo de sustitución polialfabeto original tomado como base y lugar del que se ha obtenido. Es necesario referenciar la fuente original o se considerará plagio. Comentar la aproximación seguida.
  - c. Breve explicación del algoritmo de criptoanálisis implementado
    - i. Comentad todas aquellas mejoras del original puestas en práctica
    - ii. Cualquier opción que se implemente para mejorar tiempos de respuesta es válida siempre que cumpla con los requisitos de llamada vistos en el punto 1 de las condiciones del enunciado
2. El código debe estar en Python 3.
  - a. Se puede utilizar cualquier librería del lenguaje disponible y las pruebas de la COMPETICIÓN se lanzarán sobre un equipo LINUX (Ubuntu o Red Hat).
3. Para reducir los posibles problemas es necesario entregar:
  - a. Script con las instrucciones necesarias para genera un entorno virtual de Python (<https://packaging.python.org/guides/installing-using-pip-and-virtual-environments/>) a modo de ejemplo:



```
1 # Instalación de pip
2 python3 -m pip install --user --upgrade pip
3
4 # Verificar la correcta instalación de pip
5 python3 -m pip --version
6
7 # Instalar virtualenv
8 python3 -m pip install --user virtualenv
9
10 # Crear un nuevo entorno virtual
11 python3 -m venv env
12
13 # Activar entorno virtual
14 source env/bin/activate
15
16 # Instalar paquetes necesarios desde fichero requisitos
17 pip install -r requirements_nombreEquipo.txt
18
```

- b. Fichero requirements\_nombreEquipo.txt (cada equipo deberá poner su nombre) con aquellas librerías y dependencias necesarias para que vuestra propuesta funcione.
  - c. Si hace falta alguna librería de sistema o compilador específico es necesario incluir la información en el documento
4. Código completo y funcional que cumpla con los requisitos del enunciado (50% de la nota)
  5. Clasificación final del CAMPEONATO (20% de la nota)