Arcana imperii

Els criptògrafs de Cartago van interceptar els missatges enviats a Tarraco i van poder desxifrar-ne molts, ja que no tots feien servir un sistema de xifrat prou segur. A més, han ideat una versió millorada del clàssic xifrat Cèsar, el qual incorpora una paraula a mode de contrasenya.

Hem interceptat uns missatges que han enviat a uns erudits que contenen uns textos antics, amb algun tipus de profecia que pot ser important. Per sort, els nostres espies ens han donat una informació que ens ajudarà a poder desxifrar els missatges: coneixem una paraula que apareix al text en clar.

Sistema de xifrat

Primer, cal que coneguem el nou sistema de xifrat. S'utilitza una paraula com a contrasenya, on cada lletra representa un número segons la seva posició a l'alfabet llatí, sent el 0 per la A, i 22 per la Z. Començant per la primera lletra del text, aquesta es xifra amb un xifrat Cèsar, utilitzant com a desplaçament el número de la posició de la primera lletra de la clau. Per la segona lletra del text, s'utilitza la segona lletra de la clau, etc. Un cop s'han utilitzat totes les lletres de la contrasenya, es torna a començ̧a amb la primera.

El sistema només xifra lletres de l'abecedari llatí, la resta de símbols es copien tal qual al text xifrat i no fan que passem a la següent lletra de la clau. A més, manté si les lletres són minúscules o majúscules. Per exemple, donat el text "Ab, initio.", xifrat amb la clau CAT, el xifrat resultant seria "Cb, cpinmo.". Per a desxifrar, es segueix el mateix procès, sol que a l'hora de cridar la funció Cèsar, serà amb el mode de desxifrat (número negatiu).

Per a més informació, podeu veure un sistema semblant, anomenat xifrat de Vigenère.

Requeriments

- Heu d'implementar una funció per xifrar/desxifrar un text amb el sistema explicat (Vigenère adaptat, utilitzant l'abecedari llatí), la qual tingui com a paràmetres el text a xifrar/desxifrar i la clau a utilitzar (un string). Podeu fer una funció que pugui fer les dues operacions (amb un altre paràmetre per diferenciar, per exemple), o una funció per a cadascuna.
  Consell: podeu aprofitar el codi que teniu a `cesar.py`, que trobareu al Sakai a la pràctica 1, que xifra/desxifra un missatge segons una clau

(número), depenent si és positiu/negatiu. Aquesta serà una funció auxiliar per al programa principal.

- Recordeu que l'abecedari és el llatí, el qual conté les següents 23 lletres: "*abcdefghilmnopqrstuvxyz*".

- El programa principal ha de ser en `python`, a qui se li passa per paràmetre el nom del fitxer amb el text xifrat i una paraula coneguda del text en clar. Opcionalment, la longitud mínima de la clau i la longitud màxima de la clau. El vostre programa ha de trobar totes les possibles combinacions.

  Exemple: si l'abecedari és només AB, el mínim és 2 i el màxim és 3, llavors les claus de longitud 2 són AA, AB, BA, BB, i les de 3 serien AAA, AAB, ABA, ABB, BAA, BAB, BBA, BBB.

- Per trobar la clau correcta, la funció cridarà a la funció de desxifrat amb el nou sistema, amb la clau generada, i comprovarà si la paraula coneguda apareix en el text desxifrat: si és així, el programa acabarà i retornarà la *clau* trobada. Opcionalment, podeu guardar en un fitxer el text desxifrat. Si la funció acaba sense trobar la clau correcta, voldrà dir que la seva longitud no està entre el mínim i el màxim, i retornarà None.

- Heu de fer un informe que contingui, sobre l'algorisme més important: pseudo-codi, cost teòric i experimental. Tingueu en compte, per als costos, com varien segons la longitud del text i la longitud de la clau. A més, heu de posar a l'informe la clau trobada per a cadascun dels texts que hi ha a Sakai amb la pràctica.

Arguments i parametres

L'execució de l'aplicació haurà de seguir la següent sintaxi:

```
$ python keyfinder.py fitxer_codificat.txt paraula_coneguda [min_len max_len]
```

On la `paraula_coneguda` es un string (sense espais), i `min_len`, `max_len` són les longituds mínima i màxima de les claus a provar.

Avaluació

En l'informe de la pràctica, els algorismes principals, les taules i les gràfiques han d'estar comentats.

Els programes han de compilar sense errors ni warnings, i si el text ha estat xifrat utilitzant una clau entre les longituds donades, i la paraula

coneguda apareix en el text en clar, ha de trobar sempre una clau (es possible si feu proves, que tingueu falsos positius, no passa res).

La baremació de la pràctica (sobre 10) serà la següent:

Costos 3 punts, anàlisi de costos teòrics (recursiu i iteratiu) i empírics dels algorismes;

Disseny 3 punts, recursiu i iteratiu (es tindrà en compte el cost de l'algorisme);

Implementació 4 punts

- 1 punt, iteratiu en C
- 1 punt, recursiu en python
- 1 punt, bones pràctiques de programació
- 1 punt, utilització dels recursos del llenguatge de programació.
- 1 punt extra, recursiu en haskell.

Només per als *alumnes de pla antic*, hi ha un treball extra sobre la pràctica:

Especificació 1 punt, especificació formal de l'encriptador;

Especificació 1 punt, especificació formal del desencriptador;

Enviament

L'assignació és per parelles, i representa un 20% de la nota final. Presenteu la pràctica al Campus Virtual de la UdL amb dos fitxers: un pdf per a l'informe, i un arxiu comprimit, `tgz` o `zip` (no s'admeten altres formats de compressió), per al codi font.

Els programes que s'hagin de compilar ho han de fer amb:

```
$ make
```

Els programes es testejen fent:

```
$ make test
```

No s'avaluarà cap pràctica que no tingui tots els fitxers per a poder fer la compilació amb `make` i les proves de test amb `make test`.

La pràctica podria ser verificada individualment el primer dia de laboratori després de l'entrega.