Exercices d'applications des mathématiques - Corrigé de la série n° 6

Cours 3AMOS01

1. Nouveau code ASCII.

La commende `toascii('ceciestunessai')-97` donne

```
octave:13> toascii('ceciestunessai')-97
ans =

    2    4    2    8    4   18   19   20   13    4   18   18    0    8
```

2. Problème.

(1) Il suffit d'utiliser la commande `char(toascii('bonjour')+3)`.

(2) Il suffit d'utiliser la commande `char(mod(toascii('xyz')-97+1,26)+97)`.

3. Code César.

Le premier script ci-dessous permet de crypter un message avec une substitution monoalphabétique de type César et le deuxième de décrypter le message.

```
m="votremessageclair";
#M est le message code avec une substitution de type Cesar
M=char(mod(toascii(m)-97+3,26)+97)
```

```
#M est le message code avec une substitution de type Cesar
m=char(mod(toascii(M)-97-3,26)+97)
```

4. Code César.

En exécutant avec Octave le script suivant

```
l=length(M);
for n=1:26
  stat(n)=length(findstr(M,char(n+96)))/l*100;
end
bar([1:1:26],stat)
set(gca,'xtick',[1:1:26])
set(gca,'xticklabel',{'a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s'
    ,'t','u','v','w','x','y','z'})
```

on obtient le graphique de la figure 1. On constate que la lettre qui apparaît le plus souvent est le n. Il est donc probable que le n corresponde au e et que le texte ait été crypté avec un décalage César de 9 lettres. Or en décryptant le message avec la commande `m=char(mod(toascii(M)-97-9,26)+97)`, le texte reste illisible. La deuxième lettre la plus fréquente en français étant le a, il est possible que le n corresponde au a et que le texte ait été crypté avec un décalage César de 13 lettres. En décryptant le message avec la commande `m=char(mod(toascii(M)-97-13,26)+97)`, on obtient

```
m="antonvoylnarrivaitpasadormirilallumasonjazmarquaitminuitvingtilpoussaunprofondsoupirsassi
tdanssonlitsappuyantsursonpolochonilpritunromanillouvritillutmaisilnysaisissaitquunimbroglio
confusilbutaitatoutinstantsurunmotdontilignoraitlasignificationilabandonnasonromansursonliti
lallaasonlavaboilmouillaungantquilpassasursonfrontsursoncousonpoulsbattaittropfortilavaitcha
udilouvritsonvasistasscrutalanuitilfaisaitdouxunbruitindistinctmontaitdufaubourguncarillonpl
uslourdquunglasplussourdquuntocsinplusprofondquunbourdonnonloinsonnatroiscoupsducanalsaintma
rtinunclapotisplaintifsignaaitunchalandquipassaitsurlabattantduvasistasunanimalauthoraxindig
oalaiguillonsafranniuncafardniuncharanconmaisplutotunartisonsavancaittrainantunbrindalfailsa
pprochavoulantlaplatirduncoupvifmaislanimalpritsonvoldisparaissantdanslanuitavantquilaitpula
ssaillirladisparitiondegeorgesperec";
```

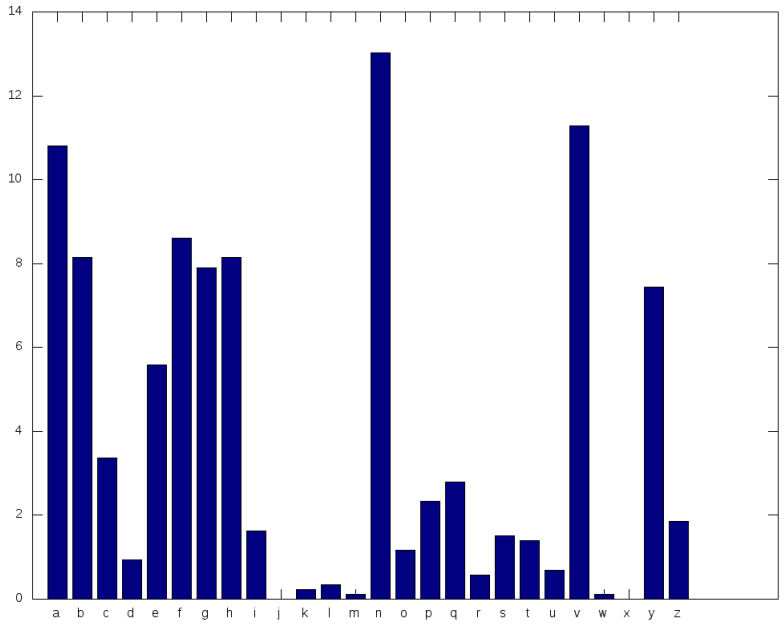

FIGURE 1. Exercice 4

c'est-à-dire le texte

"Anton Voyl n'arrivait pas à dormir. Il alluma. Son Jaz marquait minuit vingt. Il poussa un profond soupir, s'assit dans son lit, s'appuyant sur son polochon. Il prit un roman, il l'ouvrit, il lut; mais il n'y saisissait qu'un imbroglio confus, il butait à tout instant sur un mot dont il ignorait la signification.

Il abandonna son roman sur son lit. Il alla à son lavabo; il mouilla un gant qu'il passa sur son front, sur son cou.

Son pouls battait trop fort. Il avait chaud. Il ouvrit son vasistas, scruta la nuit. Il faisait doux. Un bruit indistinct montait du faubourg. Un carillon, plus lourd qu'un glas, plus sourd qu'un tocsin, plus profond qu'un bourdon, non loin, sonna trois coups. Du canal Saint-Martin, un clapotis plaintif signalait un chaland qui passait.

Sur l'abattant du vasistas, un animal au thorax indigo, à l'aiguillon safran, ni un cafard, ni un charançon, mais plutôt un artison, s'avançait, traînant un brin d'alfa. Il s'approcha, voulant l'aplatir d'un coup vif, mais l'animal prit son vol, disparaissant dans la nuit avant qu'il ait pu l'assaillir."

extrait du livre "La disparition" de Georges Perec.

5. Substitutions monoalphabétiques.

Il y a $26! \approx 4 \cdot 10^{26}$ substitutions monoalphabétiques distinctes sur un alphabet de 26 symboles et $256! \approx 8.6 \cdot 10^{506}$ pour un alphabet de 256 symboles. Par la formule de Stirling (1730), on trouve

$$n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n = \sqrt{2\pi n} \cdot 10^{n(\log(n)-\log(e))}$$

6. Substitutions monoalphabétiques.

(1) Code de chiffrage:

```
1 #alp=abcdefghijklmnopqrstuvwxyz
2 cle="acegikmoqsuwybdfhjlnprtvxz";
3 #
```

```
function M=g(m, cle)
  l=length(m);
  for n=1:l
    M(n)=cle(toascii(m(n))-96);
  end
end
#
g("bonjourcommentallezvous",cle)
```

(2) Code de déchiffrage:

```
#alp=abcdefghijklmnopqrstuvwxyz
cle="acegikmoqsuwybdfhjlnprtvxz";
#
function m=g(M, cle)
  l=length(M);
  for n=1:l
    m(n)=char(findstr(cle,M(n))+96);
  end
end
#
g("cdbsdpjedyyibnawwizrdpl",cle)
```

8. Substitutions monoalphabétiques.

En exécutant le script suivant

```
#alp=abcdefghijklmnopqrstuvwxyz
cle="gdqszjv t  hbcirnemloy ak ";
#
function m=g(M, cle)
  l=length(M);
  for n=1:l
    m(n)=char(findstr(cle,M(n))+96);
  end
end
#
g(M, cle)
```

on obtient le texte

"Aussitôt un courant se produisit, qui alla de la lame du flacon à celle du tube, et, ces deux lames ayant été reliées par un fil métallique, la lame du tube devint le pôle positif et celle du flacon le pôle négatif de l'appareil"

extrait du livre "L'île mystérieuse" de Jules Verne. (Pour une analyse complète permettant de deviner la clé, voir CRM n° 2, p. 6-7.)

9. Substitutions polyalphabétiques.

(1) Le script suivant permet de crypter des messages avec le procédé de substitution polyalphabétique périodique de Johannes Trithemius.

```
clear
#
m="messageclairacrypteravecleprocededetrithemiusvigenere";
l=length(m);
#
for n=1:l
  d=mod(n-1,25)+1;
  M(n)=char(mod(toascii(m(n))-97+d,26)+97);
end
#
M
```

(2) Le script suivant permet de décrypter les messages cryptés avec le procédé de substitution polyalphabétique périodique de Johannes Trithemius.

```
clear
#
M="ngvwfmlkuktdnqgoglxlvrbakfrushkkmmoedvhwudanmqedcmfth";
l=length(M);
#
for n=1:l
   d=mod(n-1,25)+1;
   m(n)=char(mod(toascii(M(n))-97-d,26)+97);
end
#
m
```

10. Substitutions polyalphabétiques.

(1) Le script suivant permet de crypter des messages avec le procédé de substitution polyalphabétique périodique de Giovanni Batista Belaso en utilisant la clé contenue dans la variable `c`.

```
clear
#
c="clepastreslongue";
lc=length(c);
#
m="messageclairacrypteravecleprocededegiovannibatistabelaso";
l=length(m);
#
for n=1:l
   d=toascii(c(mod(n-1,lc)+1))-97;
   M(n)=char(mod(toascii(m(n))-97+d,26)+97);
end
#
M
```

(2) Le script suivant permet de décrypter les messages cryptés avec le procédé de substitution polyalphabétique périodique avec clé de Giovanni Batista Belaso.

```
clear
#
c="clepastreslongue";
lc=length(c);
#
M="opwhayxtpstfnilcreiganxtpwafbiyhgoivigorrftpnzcwvlftlslf";
l=length(M);
#
for n=1:l
   d=toascii(c(mod(n-1,lc)+1))-97;
   m(n)=char(mod(toascii(M(n))-97-d,26)+97);
end
#
m
```

11. Substitutions polyalphabétiques.

(1) Le script suivant permet de crypter des messages avec le procédé de substitution polyalphabétique autoclave périodique de Blaise de Vigenère.

```
clear
#
c="clepastreslongue";
lc=length(c);
#
m="messageclairacrypteravecleprocededeautoclavedevigenere";
l=length(m);
```

```
#
for n=1:l
  if (n<=lc)
    d=toascii(c(n))-97;
    else
      d=toascii(m(n-lc))-97;
  end
  M(n)=char(mod(toascii(m(n))-97+d,26)+97);
end
#
M
```

(2) Le script suivant permet de décrypter les messages cryptés avec le procédé de substitution polyalphabétique périodique autoclave de Blaise de Vigenère.

```
clear
#
c="clepastreslongue";
lc=length(c);
#
M="opwhayxtpstfnilcbxwjabiewexioevbtwiruosewekvrgzlkhrelx";
l=length(M);
#
for n=1:l
  if (n<=lc)
    d=toascii(c(n))-97;
    else
      d=toascii(m(n-lc))-97;
  end
  m(n)=char(mod(toascii(M(n))-97-d,26)+97);
end
#
m
```

12. Cryptage de Blaise de Vigenère.

En exécutant avec Octave le script suivant

```
c="honoredebalzac";
lc=length(c);
#
l=length(M);
#
for n=1:l
  if (n<=lc)
    d=toascii(c(n))-97;
    else
      d=toascii(m(n-lc))-97;
  end
  m(n)=char(mod(toascii(M(n))-97-d,26)+97);
end
#
m
```

on trouve

```
m="quandvousentrezdansunemaisondejeulaloicommenceparvousdepoui
   llerdevotrechapeauestceuneparaboleevangeliqueetprovidentiellen
   estcepasplutotunemanieredeconclureuncontratinfernalavecvousene
   xigeantjenesaisquelgageseraitcepourvousobligeragarderunmaintie
   nrespectueuxdevantceuxquivontgagnervotreargentestcelapolicetap
   iedanstouslesegoutssociauxquitientasavoirlenomdevotrechapelier
   oulevotreetsivouslavezinscritsurlacoiffeestceenfinpourprendrel
   amesuredevotrecraneetdresserunestatistiqueinstructivesurlacapa
   citecerebraledesjoueurssurcepointladministrationgardeunsilence complet";
```

c'est-à-dire le texte

"Quand vous entrez dans une maison de jeu, la loi commence par vous dépouiller de votre chapeau. Est-ce une parabole évangélique et providentielle? N'est-ce pas plutôt une manière de conclure un contrat infernal avec vous en exigeant je ne sais quel gage? Serait-ce pour vous obliger à garder un maintien respectueux devant ceux qui vont gagner votre argent? Est-ce la police, tapie dans tous les égouts sociaux, qui tient à savoir le nom de votre chapelier ou le vôtre, et si vous l'avez inscrit sur la coiffe? Est-ce, enfin, pour prendre la mesure de votre crâne et dresser une statistique instructive sur la capacité cérébrale des joueurs? Sur ce point, l'administration garde un silence complet."

extrait du livre "La peau de chagrin" de Balzac.

13. Crible d'Érathosthène.

Le script donné ci-dessous affiche un tableau de `n` nombres. Les nombres différents de `0` sont premiers.

```
n=10000;
tableau=[1:1:n];
tableau(1)=0;
for l=1:n
  if (tableau(l)>0)
    k=2;
    while (k*l<=n)
      tableau(k*l)=0;
      k=k+1;
    end
  end
end
tableau(n-100:n)
```

14. Nombres premiers.

Le script donné ci-dessous détermine si le nombre `p` donné par l'utilisateur est premier.

```
p=input("Tapez un nombre entier: ");
k=2;
while (mod(p,k)>0) & (k<p)
  k=k+1;
end
if (k==p)
  printf("Ce nombre est premier.\n")
  else
    printf("Ce nombre n'est pas premier.\n")
end
```

Soit r un diviseur de p. Alors $p = r \cdot s$, où r et s sont des entiers. Si $r > \sqrt{p}$, alors

$$s = \frac{p}{r} < \frac{p}{\sqrt{p}} = \sqrt{p}$$

En conclusion, si p admet un diviseur plus grand que $\sqrt{p}$, alors p admet un diviseur plus petit que $\sqrt{p}$. Par conséquent, il suffit de tester tous les nombres plus petit que $\sqrt{p}$ comme dans le script suivant:

```
p=input("Tapez un nombre entier: ");
k=2;
while (mod(p,k)>0) & (k<sqrt(p))
  k=k+1;
end
if (k>sqrt(p))
  printf("Ce nombre est premier.\n")
  else
    printf("Ce nombre n'est pas premier.\n")
end
```

15. Puissances et modulo.

(1) $23^{55} \bmod 23 = 0$.

(2) La commande `mod(23^55,23)` donne **ans** = −1.0043e+59. Le résultat n'est pas le même qu'au point 1 ci-dessus.

(3) Le script donné ci-dessous calcule exactement $k^e \bmod n$.

```
clear
function x=pm(k,e,n)
  s=1;
  km=mod(k,n);
  for l=1:e
    s=mod(s*km,n);
  end
  x=s;
end
//
pm(23,55,23)
```

(4) On trouve `s = 0`

16. Puissances: algorithme naïf.

(1) $12^{123456} \bmod(3) = 12^{1234567} \bmod(3) = 12^{12345678} \bmod(3) = 12^{123456789} \bmod(3) = 0$

(2) Pour calculer $12^{123456789} \bmod(3)$, l'algorithme naïf effectue 123456789 passages dans la boucle.

(3) Non, le temps de calcul serait supérieur à l'âge de l'Univers !