Crypto1

Compléments Informatique

Principe du cryptage affine

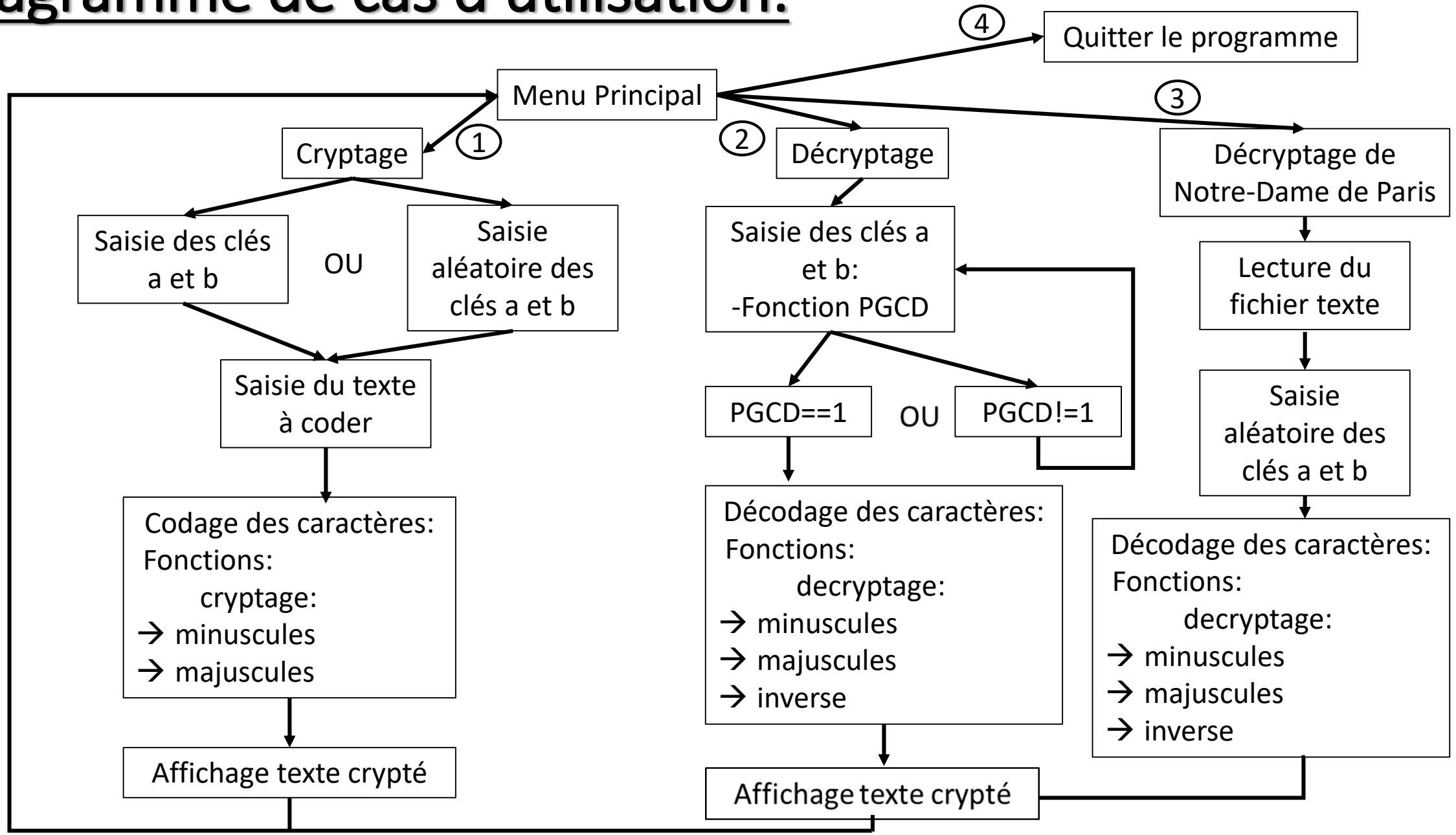
- Le principe du cryptage par chiffre affine est d'utiliser une fonction affine (ax+b) dont les coefficients sont déterminés par un couple de clés (a;b) et x correspondant au rang de la lettre dans l'alphabet (de 0 à 25). La congruence modulo 26 de la valeur obtenue par cette fonction donne le rang dans l'alphabet de la lettre cryptée.

  Ex : cryptage de F (x=5) par le couple (7;8): 3*7+8=29 et 29 mod 26 = 3.
  Donc la lettre F cryptée par le couple (7;8) done la lettre D

- Néanmoins la clé a doit être première avec 26 sinon plusieurs lettres peuvent avoir un cryptage similaire.

  Ex: avec le couple (4;1) A et N ont le même cryptage (B)

Diagramme de cas d'utilisation:



Description Programme principal

```
79
80  p=0
81  while(p!=4) :
82      p=int(input("que voulez vous faire (1:cryptage, 2:décryptage,3:crypter Notre-Dame de Paris(clés aléatoires),4:quitter le programme) : "
83      c=0
84      if p==1 :
85          fichier1=open('texte_codé.txt','w')
86          while(c!=1 and c!=2):
87              c=int(input("clés aléatoires (saisir 1) ou clés choisies(saisir 2) : "))
88          if c==1 :
89              a=random.randint(1,25)
90              b=random.randint(0,25)
91          if c==2 :
92              a=int(input('donner la clé a : '))
93              b=int(input ('donner la clé b : '))
94          texte=input('texte a coder : ')
95          texte_code=''
96          for k in texte:
97              texte_code=texte_code+cryptage(k,a,b)
98          print("le texte : \n",texte,"\n\ncodé avec le couple(%s,%d) donne \n\n:"%(a,b),texte_code)
99          fichier1.write(texte_code)                  #ecrit le texte codé dans un fichier texte
100         fichier1.close()
101     if p==2 :
102         a=int(input("donnez la clé a utilisée : "))
103         b=int(input("donnez la clé b utilisée : "))
104         texte=input("donnez le texte a décoder : ")
105         decodage=''
106         if PGCD(a,26)==1 :
107             for k in texte :
108                 lettre_de=decryptage(k,a,b)
109                 decodage=decodage+lettre_de
110             print(decodage)
111         else :
112             print("impossible a decrypter puisque a n'est pas premier avec 26")
113     if p==3 :
114         fichier=open(r"C:\Users\titou\Documents\UCO\Info\complement_info\projet_crypto\Notre_Dame_de_Paris.txt",errors="ignore")
115         f=fichier.read()
116         fichierP=open("Notre_Dame_de_Paris_Crypte.txt","w")
117         a=random.randint(1,25)
118         b=random.randint(0,25)
119         code=''
120         for k in f :
121             lettre=cryptage(k,a,b)
122             code=code+lettre
123         fichierP.write(code)
124         fichierP.close()
125         fichier.close()
126
```

Au démarrage, le programme demande à l'utilisateur ce qu'il veut faire: crypter ou décrypter un texte et l'utilisateur peut également quitter le programme.
En entrant 1, on demande à l'utilisateur d'entrer les clés a et b ou s'il le souhaite de choisir aléatoirement des clés, ensuite, il entre le texte à crypter. Le programme va crypter avec la fonction cryptage caractère par caractère pour ajouter au texte final. Finalement, le texte crypté sera affiché ainsi que les clés utilisées.
En entrant 2, même début, le programme utilise la fonction decryptage pour décrypter caractère par caractère et l'ajouter au texte codé final.
En entrant 3, le programme décrypte le fichier texte contenant Notre-Dame de Paris avec des clés aléatoires en utilisant la fonction cryptage, code recevra caractère par caractère le cryptage pour former un nouveau texte

Description fonction de reconnaissance des caractères

```
10  def minuscules(k,a,b) :
11      """cette fonction crypte les lettres en minuscules en utilisant leurs codes ASCII """
12      num=a*(ord(k)-97)+b
13      num=num%26
14      lettre=chr(num+97)
15      return(lettre)
16
17  def majuscules (k,a,b) :
18      """cette fonction crypte les lettres en majuscules en utilisant leurs codes ASCII"""
19      num=a*(ord(k)-65)+b
20      num=num%26
21      lettre=chr(num+65)
22      return(lettre)
23
```

Pour chaque lettre, on multiplie la valeur donnée par la fonction ord entre 0 et 25 par la clé a, on ajoute à ce produit la clé b. On congrue cette somme par 26, et enfin on ajoute 97 si la lettre est une minuscule ou 65 si c'est une majuscule pour appliquer la fonction chr et obtenir la lettre crypté

Description fonction de cryptage

```
50  def cryptage(lettre,cle_a,cle_b) :
51      """fonction cryptant les termes en fonction de leurs codes ASCII"""
52      if (ord(lettre)>=97 and ord(lettre)<=122) :
53          return(minuscules(lettre,cle_a,cle_b))
54      elif(ord(lettre)>=65 and ord(lettre)<=90) :
55          return(majuscules(lettre,cle_a,cle_b))
56      else :
57          return(lettre)
58
```

On détermine la nature du caractère grâce à la fonction ord:
-lettre minuscule (ord(lettre)∈[97,122])
-lettre majuscule (ord(LETTRE)∈[65,90])
-caractère spécial (,é!èù…)
Pour ensuite appliquer la fonction majuscules ou minuscules en fonction de la lettre ou laisser le caractère tel quel pour les caractères spéciaux

Description fonction Calcul du PGCD

```
25  def PGCD(a,b) :
26      """cette fonction calcule le PGCD de 2 nombres par l'algorithme d'Euclide
27         quel que soit l'ordre d'entrée des termes
28      """
29      c=max(a,b)
30      d=min(a,b)
31      """prise du max et du min pour que le calcul puisse se faire
32         peu  importe l'ordre de saisie des termes"""
33      while(c>=d):
34          r=c%d
35          if(r==0):
36              return(d)
37          else:
38              c=d
39              d=r
40
```

Cette fonction calcule le PGCD de 2 entiers a et b par l'algorithme d'Euclide. On prend d'abord le plus grand élément entre a et b puis le minimum pour effectuer la congruence entre le plus grand et le plus petit élément

Description fonction Inverse

```
34 def inverse(a,b):
35
36     k=0
37     while(((a*k)%b)!=1):
38         k=k+1
39     return(k)
40
```

Renvoie l'inverse d'un nombre a au modulo b.
a doit nécessairement être premier avec b.

Description fonction de décryptage

```
60  def decryptage(lettre,cle_a,cle_b):
61      """ permet le décryptage des termes en fonction du couple de clés
62          (a,b) utilisées """
63      if (ord(lettre)>=97 and ord(lettre)<=122) :
64          num=ord(lettre)-97
65          num=(num-cle_b)%26
66          num2=(num*inverse(cle_a,26))%26
67          return(chr(num2+97))
68      elif(ord(lettre)>=65 and ord(lettre)<=90):
69          num=ord(lettre)-65
70          num=(num-cle_b)%26
71          num2=(num*inverse(cle_a,26))%26
72          return(chr(num2+65))
73      else :
74          return(lettre)
75
```

Pour décrypter une minuscule, après avoir entré les clés, on soustrait une valeur entre 0 et 25, obtenue après avoir effectué ord(lettre), par 97. Ensuite, on soustrait le résultat par la clé b et on congrue par 26.
On multiplie le résultat par l'inverse de la clé a modulo 26 que l'on congrue à nouveau modulo 26.
Enfin, on ajoute à ce dernier résultat 97 pour que la fonction chr retrouve la lettre minuscule codée.
Même processus pour une lettre majuscule en remplaçant 97 par 65