アルゴリズム論特講 （塩田）

2006 年 6 月 15 日の課題

課題

(1) 以下のファイルをダウンロードせよ：

- 関数定義部 crypto060608.py（前回と同じ）
- ElGamal 暗号の復号化雛形プログラム hina060615.py
- ElGamal 暗号の暗号鍵のデータファイル ElGamalKey.dat
- ElGamal 暗号の暗号文のデータファイル cyphertext.dat

(2) hina060615.py の復号化関数 ElGamalDecrypt を完成し、実行して暗号文を解読せよ。

提出期限：未定 （ 512 号室ポストまで ）

参考 Web には次のようなプログラム・データもアップしてあります：

- ElGamal 暗号の鍵生成プログラム ElGamalKey.py
- ElGamal 暗号の暗号化プログラム ElGamalEncrypt.py
- C 言語による実装例 rep09.c
- C 言語版の関数定義部 crypto060608.h
- rep09.c 用のデータファイル ElGamal04.dat
- 離散対数のデモプログラム discretelog1.py （大きな素数用）
- 離散対数のデモプログラム discretelog2.py （大きくない素数用）

雛形

```
#!/bin/env python
#
# hina060615.py
# ElGamal 暗号復号化

from sys import *
from math import *
from random import *
from time import *
from string import *
from crypto060608 import *
# crypto060608.py もホームページから download せよ
```

```
##### 公開鍵 p と秘密鍵 s を用いて暗号文 [u,v] を復号化する関数 #####
def ElGamalDecrypt(u,v,p,s):
    return ##### ここを埋めよ #####

##### 鍵の読み込み #####
keyfile = open('ElGamalKey.dat','r')
p = long(keyfile.readline())    # 1 行ずつ読み込み long 整数へ変換
g = long(keyfile.readline())
s = long(keyfile.readline())
t = long(keyfile.readline())
keyfile.close()
print '法     p =',p,'(',bitlen(p),'bits )'
print
print '原始根 g =',g
print
print '秘密鍵 s =',s
print
print '公開鍵 t =',t
print

##### ブロック長の設定 : 8 * blocksize < ( p のビット長 ) #####
blocksize = 32

##### 復号処理 #####
z = ''                      # 復号文用の文字列
cypherfile = open('cyphertext.dat','r')    # 暗号文ファイルの open
while 1:
    u0 = cypherfile.readline()    # 文字列の読み込み
    if not u0:            # 読み込めなくなったら break
        break
    u = long(u0)            # long 整数へ変換
    v = long(cypherfile.readline())    # もう 1 行読み込み long 整数へ変換
    y = ElGamalDecrypt(u,v,p,s)    # 復号化
    w = []
    for i in range(0,blocksize):
        w.append(chr(y % 256))    # 256-進数の下の桁から w に格納し
        y /= 256
    w.reverse()            # 最後に順番を逆転
    z = z + join(w,'')        # 文字リストを文字列に結合
cypherfile.close()
print z                     # 出力
```

C 言語による実装例

```
/* rep09.c
gcc rep09.c -o rep09 -lm
*/

#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include"crypto060608.h"
```

```
/* 暗号文 (u,v) を、公開鍵 p と秘密鍵 s を用いて復号する */
int ElGamaldecrypt(int u, int v, int p, int s)
{
    return modmul(powermod(u,p-1-s,p),v,p);
}

main()
{
    int p,g,t,s,u[128],v[128];
    int n,i,y;
    FILE *f;

    f=fopen("ElGamal04.dat","r");    // データファイルのオープン
    fscanf(f,"%d",&p);         // 法 p の読み込み
    fscanf(f,"%d",&g);         // 原始根 g の読み込み
    fscanf(f,"%d",&t);         // 公開鍵 t の読み込み
    fscanf(f,"%d",&s);         // 秘密鍵 s の読み込み
    printf("法      p = %d\n",p);
    printf("原始根 g = %d\n",g);
    printf("公開鍵 t = %d\n",t);
    printf("秘密鍵 s = %d\n",s);
    printf("\n");

    n=0;
    while(!feof(f)){
        fscanf(f,"%d %d",u+n,v+n); // 一文字分の暗号文 (u[n],v[n]) の読み込み
        n++;
    }
    fclose(f);

    printf("暗号文 :\n");
    for(i=0;i<n;i++){
        printf("%6d %6d\n",u[i],v[i]);
    }
    printf("\n");

    printf("復号文 : \n");
    for(i=0;i<n;i++){
        // i 文字目の暗号文 (u[i],v[i]) を復号文 y に復号
        y=ElGamaldecrypt(u[i],v[i],p,s);
        printf("%c",(char)y);    //復号文 y を文字型にキャストして出力
    }
    printf("\n");
}
```

離散対数の例

```
素数 p          = 37  ( 6 bits )
法 p の原始根 g = 2

log(     1) =      0
log(     2) =      1
log(     3) =     26
log(     4) =      2
log(     5) =     23
log(     6) =     27
log(     7) =     32
log(     8) =      3
log(     9) =     16
log(    10) =     24
log(    11) =     30
log(    12) =     28
log(    13) =     11
log(    14) =     33
log(    15) =     13
log(    16) =      4

-----

素数 p          = 926467  ( 20 bits )
法 p の原始根 g = 2

log(     1) =      0
log(     2) =      1
log(     3) = 718029
log(     4) =      2
log(     5) = 428643
log(     6) = 718030
log(     7) = 643752
log(     8) =      3
log(     9) = 509592
log(    10) = 428644
log(    11) = 525651
log(    12) = 718031
log(    13) = 796006
log(    14) = 643753
log(    15) = 220206
log(    16) =      4

-----

素数 p          = 256203664337605982780227503772 9  ( 102 bits )
法 p の原始根 g = 19

log( 893854420799811390028031918203) = 133011296887868561071571242607 5
log(168264168046944982831913050996 4) = 178568626695265602235561039315 5
log(196341530440929328500153276383 4) = 239488600101932860761885845462 4
log( 818352909423973965102540970161) =  858776025956353521616807395881
log( 597401162740021298332719326132) = 208201192341352502907012425732 3
```

log(11775904010764264686703499677 50) = 69847270286118847845124861 1480
log(38113488976388061757097271 0940) = 32576172066785276426238167 9539
log(17011323975616983626911397609 77) = 15532385086402236508118965 0818

素数 p = 56590536795922808330876395114204103583295015315982203379601759
126600466727197280216950542557282774103904177086374822182400042611004226895197 64
177920001 (501 bits)
法 p の原始根 g = 13

log(41635029140621071098430286373903643177284106842831778114526247237621097774 87
52495722441280365335455688942754202880429017269266219789370659214263447585 8)

= 27542861176920075775024787630773201980762432060732029513162342431182455259548 7
97464271134477861903859856115919241134047135826087107536751713311146897 21

log(55190898426081895729898596380917334201792029313737215534360373159024476267 49
70808267237614603222314538833997217477626746680096583426588708079716209563 5)

= 11705474336018699770617052460242814763832752409189245784306827139108942957318 3
41272372031193402172230261247205610155678999614503020125713109512854272 9