東京大学玉原国際セミナーハウス
2007 年 9 月 15 日－ 17 日

群馬県教育委員会高校教育課
東京大学大学院数理科学研究科

平成１９年度高校生玉原数学セミナー
「素数」についての演習

演習のほとんどの時間は、Windows 上で十進ＢＡＳＩＣというフリーソフトを使って素数の振る舞いを探る。

十進ＢＡＳＩＣは、高校の教科書で取り上げられるＢＡＳＩＣとほとんど同じものだが、行番号が省略できる、自由につけられるという特徴がある。また、グラフィックスの機能が比較的易しく使える。十進ＢＡＳＩＣは、
http://hp.vector.co.jp/authors/VA008683/ からダウンロードできる。

1. エラトステネス (Eratosthenes) の篩（ふるい）

インターネット・エクスプローラーで次のページにアクセスする。

http://www.faust.fr.bw.schule.de/mhb/eratclass.htm

１から４００までの表がある。ある数字をマウスで左クリックするとその数字の２倍以上の倍数が消えていく。このことを実際に確かめよ。リロードすることにより何度も繰り返すことができる。（ドイツの小学校の先生が作ったものです。）
合成数がすべてふるい落とされると素数が残る。

問題 1.1. 素数を残すために、この表においてクリックする数字の数を最も少なくするにはどうすればよいか。

このプログラムは、ＪＡＶＡという言語で書かれている。これと同じようなものを十進ＢＡＳＩＣで作ったものがパッケージにある。
`ERATOS_2500.BAS`, `prime_sieve_2500.BAS`

2. 十進ＢＡＳＩＣの基本

2.1. パッケージのダウンロード.

インターネット・エクスプローラーで次のページにアクセスする。
http://tambara.ms.u-tokyo.ac.jp/TambaraBasicPrograms.zip
このファイルをディスクトップに保存する。この書庫ファイルを展開する。

2.2. 十進ＢＡＳＩＣを起動する.

ディスクトップ上のＢＡＳＩＣをクリックするか、.BAS のファイルをクリックする。

2.3. 割り算.

自然数 a, b に対し、

$$a \div b = q \text{ 余り } r$$

の計算は十進ＢＡＳＩＣ上では次のようになる。パッケージの WARIZAN.BAS をクリックすると、十進ＢＡＳＩＣのプログラムが表示される。

```
! WARIZAN.BAS                                   コメント: プログラムの名前
! 自然数 a,b を入力して割り算の結果を出力する    コメント:プログラムの説明
OPTION ARITHMETIC DECIMAL_HIGH          1000 桁の整数を扱えるようにする
INPUT PROMPT "自然数 a,b を入力して下さい：":a,b
  画面に  自然数 a,b を入力して下さい：  を出力し、変数 a, b の入力を求める
PRINT                                                  出力画面での改行
LET q=INT(a/b)                                  変数 q に a/b の整数部分を代入
LET r=MOD(a,b)                  変数 r に a − b ∗ INT(a/b) と同じものを代入
PRINT a;"÷";b;"=";q;"余り";r
                                 a, b, q, r は変数 a, b, q, r の値、"…"はその文字…
PRINT a;"=";b;"×";q;"＋";r                ; は改行しないで続けて出力する
END                                                    プログラムの終了
```

最後の行は、$a \div b = q$ 余り r を $a = b \times q + r$ と書いた。
▶ をクリックしてＢＡＳＩＣの実行する。１００桁くらいの割り算はすぐにできる。

3. BASIC でつくるエラトステネスのふるい

エラトステネスのふるい４００までの素数のリストを作る。１から４００までの変数（ＢＡＳＩＣでは配列と呼ぶ）を用意し、まずすべて０とする。２から順に２０までの自然数に対して、その２倍以上の倍数についてその数の配列変数を１とする。終わったら、０のまま残っている変数の添え字を取り出す（１をのぞいて）。

```
! ERATOS400.BAS                                 コメント：プログラムの名前
! ふるいによって４００までの素数のリストを作る          コメント：説明
DIM s(400)                  s は１から４００までの添え字をもつ配列（変数）
DIM p(100)
MAT s=ZER                                  s(1) から s(400) を 0 とおく。
MAT p=ZER                                          MAT は行列 MATRIX
FOR i=2 TO 20                                 変数 i について 2 から 20 まで
```

```
   FOR  j=2*i TO 400 STEP i      変数 j を 2 * i から 400 まで i ずつ増やして
      LET s(j)=1                                  変数 s(j) に 1 を代入
   NEXT j                                                        次の j
NEXT i                                                           次の i

LET  k=1                                                  変数 k に 1 を代入
FOR i=2 TO 400                                  変数 i について 2 から 400 まで
   IF s(i)=0 THEN                                  変数 s(i) の値が 0 ならば
       LET  p(k)=i                                   変数 p(k) = i を代入
       LET  k=k+1                    変数 k に k + 1 を代入（k の値を増やす）
   END IF                                  3 行上の条件でおこなうのはここまで
NEXT i                                                           次の i

PRINT "400 までの素数の個数は"; k-1;"個。"              素数の個数を書き出す
FOR kk=1 TO k-1                    変数 kk を使って k − 1 個の素数を書き出す
    PRINT p(kk);                                        続けて書き出す
NEXT kk                                                      次の kk
END                                                   プログラムの終了
```

この結果を最初のＪＡＶＡプログラムの結果と比べてみよ。

問題 3.1. プログラム ERATOS400.BAS を改造して１０００００００までの素数の個数を数え、最後の１０００個を書き出すプログラム ERATOS1000000.BAS にせよ。

注意：配列変数の添え字の大きさは、我々のＰＣでは１０００００００程度が限度である。

プログラムの最初に

```
  LET  t0=TIME
```

を書き、出力の最初を

```
PRINT "1000000 までの素数の個数は"; k-1;"個。";TIME-t0;"秒"
```

とすると計算の時間を計ることができる。

問題 3.2. 問題 *3.1* のプログラム ERATOS1000000.BAS を修正しての計算の時間を図ってみよ。また、プログラムの前半を次のように変えてかかる時間を比べてみよ。

```
FOR  j=4 TO 1000000 STEP 2
    LET s(j)=1
NEXT j
FOR i=3 TO 1000  step 2
  FOR  j=2*i TO 1000000 STEP i
    LET s(j)=1
  NEXT j
NEXT i
```

問題 3.3. 第 *1* 節、問題 *1.1* で見たように、2 から順に倍数を消していき残ったものの倍数を消すのが、手順が最も少なく 2 からの素数の表を作る方法であった。これをプログラム PRIME_SIEVE.BAS に書いてみよ。

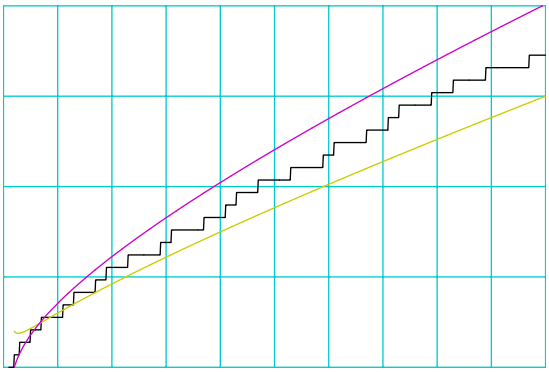
問題 3.4. 「エラトステネスのふるい」によって r を 1 から 9 として、１０の $r+4$ 乗までの最後の 1 万個の整数の中の素数の個数とリストを出力するプログラム ERATOS9999.BAS を作って実行してみよう。(問題 *3.3* のプログラム PRIME_SIEVE.BAS は素数表の添え字が大きくなりすぎるので、問題 *3.2* のプログラム ERATOS1000000.BAS から修正することをかんがえる。また、後のために出力されたテキストファイルをセーブしておく。) この個数を $10000 \times \dfrac{1}{(r+4)\log 10}$ と比較してみよう。ただし、$\log 10 = 2.302585\cdots$.

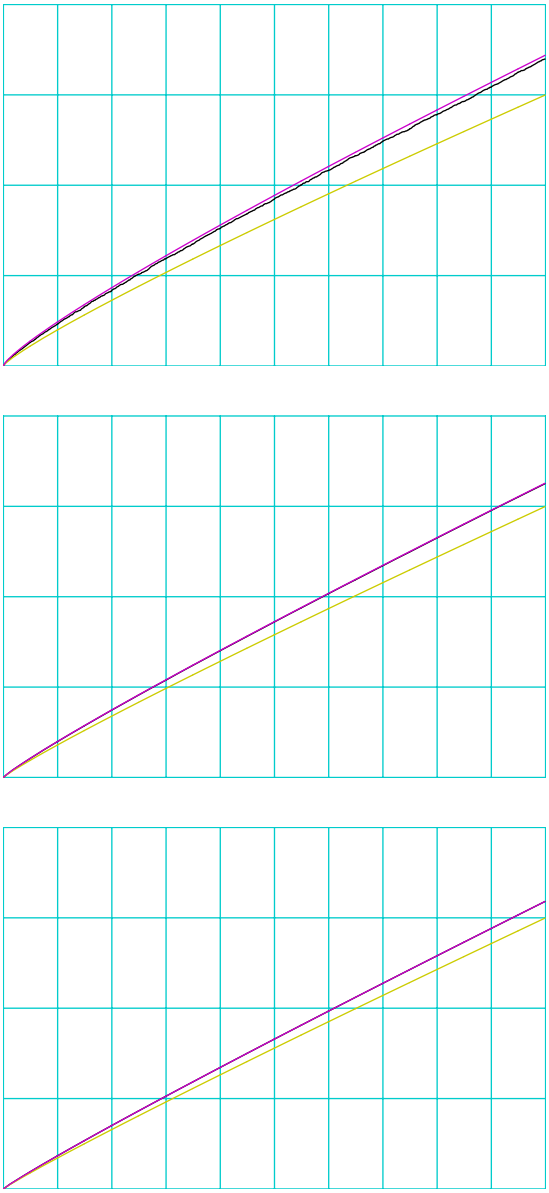
4. 素数の分布

問題 3.4 において、素数の個数を $10^{(r+4)} - 9999$ から $10^{(r+4)}$ までの素数の個数は、$10000 \times \dfrac{1}{(r+4)\log 10}$ に近いことが観察された。ルジャンドル、ガウスが予想した素数定理は次のもので、現在では複素関数論を使って証明されている。

定理 4.1. 自然数 n に対して $\pi(n)$ を n より小さい素数の個数とする。$\lim_{n\to\infty} \dfrac{\pi(n)}{\dfrac{n}{\log n}} = 1$

より詳しい分布の予想があり、それは、分布の密度がほぼ $\dfrac{1}{\log x}$, すなわち、m から n まで $(0 < m < n)$ の素数の個数はほぼ $\displaystyle\int_m^n \frac{\mathrm{d}x}{\log x}$ となるというものである。問題 3.4 においては、積分の代わりに、この場合値が大きく違わない $\dfrac{n-m}{\log n}$ を使っている。詳しい分布の予想とゼータ関数の零点の実部が $\dfrac{1}{2}$ というリーマン予想は同値である。以下は、0 から 100, 0 から 10000, 0 から 1000000, 0 から 100000000 における $p(n)$, $\dfrac{n}{\log n}$, $\displaystyle\int_2^n \frac{\mathrm{d}x}{\log x}$ のグラフを描いたものである。問題 3.4 からもわかったように、$p(n)$ と $\displaystyle\int_2^n \frac{\mathrm{d}x}{\log x}$ は非常に近いグラフとなっている。





5. 割り算による素数の判定

問題 5.1. 割り算によって入力した自然数 n が素数かどうか判定し、合成数ならば 1, n と異なる約数を１つ与えるプログラム PRIME.BAS を作れ。

問題 5.2. 入力した自然数の前の素数、後の素数を表示し、プログラムの実行にかかった時間を出力するプログラム PrevNextPRIME.BAS を作れ。ただし入力した数が素数のときはそれ自身を前の素数とする。このプログラムを修正して、１０の５乗から２０乗に対し、それを実行し、プリントするプログラムつくれ。

問題 5.3. 割り算により r を１から９として、１０の $r+4$ 乗までの最後の１万個の整数の中の素数の個数とリストを出力するプログラム WARIZAN9999.BAS を作って実行してみよ。問題 *3.4* のプログラム ERATOS9999.BAS と時間の比較をしてみよう。

問題 5.4. 素数であることを、問題 *5.1* の方法で判定するのに１年かかるのは何桁のときか。

後で見るように、合成数であることを判定する方法には優れた方法フェルマーテストがある。

6. 合同式の計算

6.1. 合同式における和、差、積.

自然数 a, b に対し、MOD(a,b) という a を b で割った余りを与える関数があるので、合同式における和、差、積の十進ＢＡＳＩＣ上での計算は非常に簡単である。すなわち、

```
!CONGRUENT.BAS
INPUT PROMPT "整数 a1,a2, 正整数 b を入力":a1,a2,b
PRINT a1;"＋";a2;"≡";MOD(a1+a2,b);"mod" b
PRINT a1;"－";a2;"≡";MOD(a1-a2,b);"mod" b
PRINT a1;"×";a2;"≡";MOD(a1*a2,b);"mod" b
END
```

合同式の計算する場合は、桁があふれるような現象がほぼ起きないし、計算時間が桁が大きくなることにより大きくなりすぎることもない。コンピュータでの計算に好適である。

6.2. 合同式における商.

合同式で割り算を行うことは必ずしもできないが、a, b, y が与えられたとき、$ax \equiv y \mod b$ を満たす x を計算できる場合がある。

- a, b の最大公約数を d とする。
- 後で確かめるように、$au + bv = d$ を満たす整数 u, v が存在する。
- このとき、y が d の倍数 $y = kd$ ならば、$x = ku$ が $ax \equiv y \mod b$ を満たす。
- ２つの解 x_1, x_2 に対し、$a(x_1 - x_2) \equiv 0 \mod b$ が成立する。
- $a = a_0 d$, $b = b_0 d$ のとき、$x_1 - x_2$ は b_0 の倍数となる。

- 特に、a, b が互いに素であるときには $d = 1$ で $au + bv = 1$ を満たす整数 u, v について、$x = yu$ が $ax \equiv y \mod b$ を満たす。
- ２つの解 x_1, x_2 に対し、$x_1 - x_2$ は b の倍数となる。従って $0 \leqq x < b$ となる解は一意である。

6.3. ユークリッドの互除法.

自然数 a, b の最大公約数を求めるためにはユークリッドの互除法が用いられる。

ユークリッドの互除法の原理は簡単である。

- 自然数 $a > b > 0$ に対して、$a \div b = q$ 余り r のとき、$a = b \times q + r$ であるが、$b > r > 0$ であり、$\gcd(a, b) = \gcd(b, r)$ である。
- 従って、数字の小さい $b > r > 0$ に対する $\gcd(b, r)$ を同じ手続きで求めていけば、最後には、余りが 0 となる。そのときの q が最大公約数である。
- 実際、

$$\begin{aligned} a &= b \times q_1 + r_1 \\ b &= r_1 \times q_2 + r_2 \\ r_1 &= r_2 \times q_3 + r_3 \\ &\vdots \\ r_{n-1} &= r_n \times q_n \end{aligned}$$

となり r_n が最大公約数である。

n 桁までの 2 つの自然数の最大公約数は、多くともほぼ $5n$ 回互いに割り算すると得られる。これは、実際にはフィボナッチ数列の隣り合う項が最も時間のかかるものであることからわかる。

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, 17711, 28657, 46368, 75025, 121393, 196418, 317811, 514229, 832040, ...

$a_{n+1} = a_n + a_{n-1}$ で定まる数列をフィボナッチ数列と呼ぶ。

- $\{a_n\}$ に対して $x^2 - x - 1 = 0$ の解 $u = \dfrac{1 - \sqrt{5}}{2}$, $v = \dfrac{1 + \sqrt{5}}{2}$ をとる。
- $a_n = ku^n + \ell v^n$ は $a_{n+1} = a_n + a_{n-1}$ を満たす。
- $a_0 = k + \ell$, $a_1 = ku + \ell v$ を解いて、k, ℓ を定めれば、a_n が初期値 a_0, a_1 をとる解となる。
- この a_n の絶対値は n が十分大きければ、v^n と同じ桁数となる。$\log_{10} \dfrac{1 + \sqrt{5}}{2} = 0.20898764$ であって、a_n はほぼ $\dfrac{n}{5}$ 桁の十進数である。

さて、ユークリッドの互除法のプログラムは以下のようになる。

```
! EUCLID1.BAS
! ユークリッドの互除法により
! GCD(a,b) を求める
OPTION ARITHMETIC DECIMAL_HIGH
INPUT PROMPT  "a>b>0 となる整数":a,b
PRINT
10 LET  q=INT(a/b)
LET  r=MOD(a,b)
IF r=0 THEN GOTO 30 ELSE GOTO 20          r = 0 ならば、行番号 30 へ、
                                          そうでなければ行番号 20 へ
20 PRINT a;"=";b;"×";q;"+";r
LET a=b
LET b=r
GOTO 10                                   行番号 10 へ
30 PRINT a;"＝";b;"×";q
END
```

このユークリッドの互除法は、a, b に対し、$au + bv = \gcd(a, b)$ となる u, v の計算に使える。その理由は、

- $a = b \times q_1 + r_1$, $b = r_1 \times q_2 + r_2$ のとき、$a - b \times q_1 = r_1$ で、
- $b - (a - b \times q_1) \times q_2 = r_2$, すなわち $a(-1) + b(1 + q_1 q_2) = r_2$,
- $r_1 = r_2 \times q_3 + r_3$ のとき、r_1, r_2 に a, b の 1 次式を代入して r_3 も a, b の 1 次式.
- 同様に続けて、r_n が a, b の 1 次式となる。

```
! EUCLID2.BAS
! ユークリッドの互除法により
! ax+by=GCD(a,b) となる整数 x,y と GCD(a,b) を求める
OPTION ARITHMETIC DECIMAL_HIGH
INPUT PROMPT  "a>b>0 となる整数":a0,b0
PRINT
LET  s1=0
LET  t1=1
LET  q=INT(a0/b0)
LET  r=MOD(a0,b0)
PRINT a0;"=";b0;"×";q;"+";r
LET  s2=1
LET  t2=-q
PRINT "(";s2;") ×";a0;"＋ (";t2;") ×";b0;"＝";r
LET a=b0
LET b=r

10 LET  q=INT(a/b)
   LET  r=MOD(a,b)
   IF r =0 THEN GOTO 30 ELSE GOTO 20
20 PRINT a;"＝";b;"×";q;"＋";r
   LET  s3=s1-q*s2
   LET  t3=t1-q*t2
   PRINT "(";s3;") ×";a0;"＋ (";t3;") ×";b0;"＝";r
   LET a=b
   LET b=r
   LET s1=s2
   LET t1=t2
   LET s2=s3
   LET t2=t3
   GOTO 10
30 PRINT a;"＝";b;"×";q
END
```

プログラムを実行してみよ。

7. フェルマーの小定理

フェルマーの小定理は次のものである。

定理 7.1. a が素数 p の倍数ではないとき、$a^{p-1} \equiv 1 \mod p$ である。

これが成立する様子を十進ＢＡＳＩＣで確かめてみよう。

そこで問題は $a^{p-1} \mod p$ の計算である。

- １０００桁までは計算できる十進ＢＡＳＩＣであるが、例えば 234^{90916} は１００万桁はあるのでそのままでは計算できない。
- しかし、 mod 90917 で計算する限りは、５桁の積を再び５桁の数字で表すことになり、累乗を順に積をとる形で計算すると桁数の問題はおこらない。
- しかし、90916 回計算するのは時間をとることが予想される。これをうまく解決するのが、２進展開である。
- 90916 は２進数では 10110001100100100 すなわち、

$$\begin{aligned} &90916 = 2^{16} + 2^{14} + 2^{13} + 2^9 + 2^8 + 2^5 + 2^2 \\ &= ((((((2^2+1)\times 2+1)\times 2^4+1)\times 2+1)\times 2^3+1)\times 2^3+1)\times 2^2 \end{aligned}$$

である。

- $$\begin{aligned} a^{90916} = (((((((((((((a^2)^2 \times a)^2 \times a)^2)^2)^2)^2 \\ \times a)^2 \times a)^2)^2)^2 \times a)^2)^2)^2 \times a)^2)^2 \end{aligned}$$

である。

- 結局２２回の２乗または $\times a$ の操作を mod 90917 で行えばよい。
- この２２は２進法の表示の桁数と１の個数の和から２を引いたものである。

7.1. ２進展開.

自然数 n を２進展開するには

- $n \div 2 = q_1$ 余り r_1 として、n の２進展開の第１位は r_1,
- 次に再び $q_1 \div 2 = q_2$ 余り r_2 として、n の２進展開の第２位は r_2、
- 以下同様に、$q_{m-1} \div 2 = q_m = 1$ となるまで続けると、
- $q_m r_{m-1} r_{m-2} \cdots r_2 r_1$ として得られる。

与えられた１０進法の自然数を２進数に直して表示するプログラムを書いてみよう。

```
! BIN_EXPANSION.BAS
! 入力された自然数を２進表示する
OPTION ARITHMETIC DECIMAL_HIGH
INPUT PROMPT "十進数":n
LET  nn$=""                          nn$は文字列を表す。長さ 0 の文字列を代入

50 LET  nn$=STR$(MOD(n,2)) & nn$      n を２で割った余りを表す文字列と
                                  それまでの nn$ を並べた文字列を nn$ に代入
   LET  n=(n- MOD(n,2))/2
   IF n>0 THEN GOTO 50                          n > 0 ならば 50 行目に戻る。
   PRINT nn$                                          文字列 nn$ を出力
END
```

色々な十進整数を２進数に書いてみよう。

7.2. **累乗の剰余類.**

$x^n \mod a$ を計算するプログラムを書いてみよう。このプログラムでは、外部関数 TONTHMODP を定義して用いる形をとっている。

```
! TO_THE_NTH_MOD_A.BAS
! x, n, a に対し x^n mod a を与える。
OPTION ARITHMETIC DECIMAL_HIGH
DECLARE EXTERNAL FUNCTION TONTHMODP          関数 TONTHMODP を使う
INPUT PROMPT "x^n mod a の x, n, a":x,n,a
LET  nn$=""
50 LET  nn$=STR$(MOD(n,2)) & nn$
   LET  n=(n- MOD(n,2))/2
   IF n>0 THEN GOTO 50
   END IF
   PRINT nn$
   PRINT TONTHMODP(x,nn$,a)
END                                          プログラムの終了

EXTERNAL FUNCTION TONTHMODP(x,n$,p)                関数の定義
   OPTION ARITHMETIC DECIMAL_HIGH
     LET  y=x
     FOR i=2 TO LEN(n$)
     LET  y=Mod(y*y,p)
     IF n$(i:i)=STR$(1) THEN GOTO 100 ELSE GOTO 130 文字列の i 番目が
       1（という文字）ならば行番号１００へ、そうでなければ行番号１３０へ
100  LET  y=MOD(x*y,p)
     GOTO 130
130  NEXT i
     LET TONTHMODP=y  ここで得られた y の値を TONTHMODP の値とする。
END FUNCTION                                   関数の定義終わり
```

フェルマーの小定理が成り立っている様子が観察できる。

問題 7.2. プログラム TO_THE_NTH_MOD_A.BAS を修正して、素数 p を適当にとり、$2^{p-1} \mod p, \ldots, 100^{p-1} \mod p$ を計算させよ。

7.3. **フェルマーテスト.**

フェルマーの小定理により素数 p に対しては $a^{p-1} \equiv 1 \mod p$ である。

この命題の対偶は、$a^{n-1} \not\equiv 1 \mod n$ ならば n は合成数であるというものである。

問題 7.3. プログラム TO_THE_NTH_MOD_A.BAS を修正して、与えられた n に対し、$2^{n-1} \mod n$, $3^{n-1} \mod n$, $5^{n-1} \mod n$, $7^{n-1} \mod n$, $11^{n-1} \mod n$ を計算させるプログラムを作れ。これがすべて 1 になるものを $2, 3, 5, 7, 11$ 擬似素数と呼ぶことにする。

問題 7.4. r を１から３０として、１０の $r+4$ 乗までの最後の１万個の整数の中の $2, 3, 5, 7, 11$ 擬似素数の個数とリストを出力するプログラムを作って実行してみよう。*4* ページの問題 *3.4* の結果と比較せよ。

8. ＲＳＡ暗号

8.1. 情報を数字で送る.

コンピュータの中ではすべてのデータが２進数の形をしている。従って、文章も図形も数字で表示される。

通常のアルファベット A, B, ... はアスキーコードという２進数で８桁、１０進数で０から１２７で表記される。アルファベットは大文字小文字で５２、数字、記号、句読点などで大体１２８個の記号をなしている。

文字を数字に直すプログラムは次のようになる。

```
!MOJI2SUJI.BAS
!文字を数字の列に直す
INPUT a$
DIM b(LEN(a$))
FOR k =1 TO LEN(a$)
   LET  b(k)=ORD(a$(k:k))
   PRINT b(k);",";
NEXT k
END
```

これで、半角の英文を直すと、ほぼ２桁の数の列になる。全角のものを直すと、４，５桁の数の列となる。この数字だけでも十分暗号に見えるが、これが、文字のコードだと気付く人がいればすぐに読まれる。

実際上のプログラムの END の前に

```
PRINT
DIM c$(LEN(a$))
FOR k =1 TO LEN(a$)
   LET c$(k)=CHR$(b(k))
   PRINT c$(k);
NEXT k
```

を書けば、もとの文字に戻る。

8.2. オイラーの定理の特別な場合.

定理 8.1. 素数 p, q の積 pq に対し、a が p, q の倍数でなければ、$a^{(p-1)(q-1)} \equiv 1 \mod pq$ である。

問題 8.2. 第 *7.2* 節の問題 *7.2* のように、上の命題を適当な p, q, $a = 2, \ldots, 100$ に対して検証せよ。

8.3. **公開するもの.**

- 玉原さんはＲＳＡ暗号を作るためには素数 p, q を用意する。$n = pq$ を計算しておく。
- $(p-1)(q-1)$ と互いに素な e をとり、d を $ed = 1 \mod (p-1)(q-1)$ の解とする。
- 玉原さんは、玉原さん宛ての通信には、n, e を使ってくださいと掲示する（p,q あるいは d は秘密である）。

- 群馬さんは玉原さん宛ての文章を、まず n より小さい自然数の列 a_1, a_2, ..., a_k に直し、それから

  $$b_1 \equiv (a_1)^e \mod n,\ b_2 \equiv (a_2)^e \mod n,\ \ldots,\ b_k \equiv (a_k)^e \mod n$$

  を計算し、この列 b_1, b_2, ..., b_k を玉原さんに送る。

- 玉原さんは、この列 b_1, b_2, ..., b_k に対し、

  $$c_1 \equiv (b_1)^d \mod n,\ c_2 \equiv (b_2)^d \mod n,\ \ldots,\ c_k \equiv (b_k)^d \mod n$$

  を計算する。
- このとき

  $$c_i \equiv (b_i)^d \equiv ((a_i)^e)^d \equiv (a_i)^{ed} \equiv a_i$$

  だから、c_1, c_2, ..., c_k は a_1, a_2, ..., a_k と同じもので、玉原さんは群馬さんの文章を復元することができる。

8.4. **プログラムでの実験.**

1．問題 5.2 のプログラムを使って、まず素数 p, q をとり、$n = pq$ を計算する。また、同じプログラムで、比較的小さい素数 e をとる。
2．EUCLID2.BAS を用いて、$ed+(p-1)(q-1)r = 1$ となる $d \mod (p-1)(q-1)$ を定める。
3．ANGOU.BAS を見つけた p, q, n, e, d で書き換えて、適当な文章 $a$$を書き込んで実行する。

```
!ANGOU.BAS
  !暗号化復号化の実験
  OPTION ARITHMETIC DECIMAL_HIGH
  DECLARE EXTERNAL FUNCTION BIN_EXP$
  DECLARE EXTERNAL FUNCTION TONTHMODP

  LET p=13
  LET q=17
  LET e=7
  LET d=55
  LET n=p*q
  PRINT "n=";n; "e=";e

  LET  a$="I love you."
  PRINT a$
  DIM b(LEN(a$))
  DIM c(LEN(a$))
```

```
DIM f(LEN(a$))

FOR k =1 TO LEN(a$)
   LET  b(k)=ORD(a$(k:k))
   PRINT b(k);",";
NEXT k
PRINT

LET e$=BIN_EXP$(e)
PRINT e$
FOR k=1 TO LEN(a$)
   LET c(k)=TONTHMODP(b(k),e$,n)
   PRINT c(k);",";
NEXT k
PRINT

LET d$=BIN_EXP$(d)
PRINT d$
FOR k=1 TO LEN(a$)
   LET f(k)=TONTHMODP(c(k),d$,n)
   PRINT f(k);",";
NEXT k
PRINT

LET  g$=""
FOR k=1 TO LEN(a$)
   LET  g$=g$ & CHR$(f(k))
NEXT k
PRINT g$
END

EXTERNAL FUNCTION BIN_EXP$(n)
OPTION ARITHMETIC DECIMAL_HIGH
LET  nn$=""
50 LET  nn$=STR$(MOD(n,2)) & nn$
   LET  n=(n- MOD(n,2))/2
   IF n>0 THEN GOTO 50
   LET BIN_EXP$=nn$
END FUNCTION

EXTERNAL FUNCTION TONTHMODP(x,n$,p)
   OPTION ARITHMETIC DECIMAL_HIGH
   LET  y=x
   FOR i=2 TO LEN(n$)
       LET  y=Mod(y*y,p)
       IF n$(i:i)=STR$(1) THEN GOTO 100 ELSE GOTO 130
100    LET  y=MOD(x*y,p)
       GOTO 130
```

```
  130 NEXT i
      LET TONTHMODP=y
  END FUNCTION
```

メールを使って n, e を送り、相手がそれを使って、暗号化した数字列から復号してみると良い。