Rep.Fac.Sci.Engrg. Saga Univ. 33-1(2004),

Reports of the Faculty of Science and Engineering, Saga University, Vol.33, No.1, 2004

# 大学教育とコンピュータ II

前橋敏之＊

# Computer in University Education

By

**Toshiyuki MAEBASHI**

**Abstract:** This is a sequel to my paper [0]. We continue to think about computer education in senior high schools and universities. Our primary concern is in how to use the javascript in university programming education because its use is most economical.

Keyword: Computer Education, Javascript, Excel, Euclid Algorithm, Inverses in mod $n$

キーワード: 情報科、理工系大学におけるプログラミング教育、ジャヴァスクリプト、ホームページ プログラミング

実験の結果から理論を推定する。その理論が大前提となり特殊な問題が小前提となって、科学の結論が演繹される。こういった方法論が長く続いてきたが、前世紀も後半になって、新しいやり方が現れた。

科学者は自然現象に幾つかのパラメータを付けモデル化する。そのパラメータがある範囲を動く間は、連続的にその自然現象は変化する。しかし、ある閾値に達すると突如激しく様相が変わる。僅かなパラメータ（例えば初期条件）の変化が、生じる現象に著しい差異を起こさせたのである。それが、（決定論的）カオスである。方程式に従って、コンピュータ画面上に点を打っていき、現れたパタンを分析する。こういった仕方で、「自然のフラクタル幾何学」が登場する。フラクタル、カオスは新しい科学の分野となった（[1][2][3]参照。[3]は歴史を扱ったものである）。コンピュータが科学の不可欠な道具となったのである。検索エンジンのついたデータベース、経済活動になくてはならぬ暗号（署名、認証）もコンピュータによって作成される([4]は手ごろな入門書。この論文のプログラムの中にそこの定理がつかわれている。[5]には公開鍵暗号のベーシックプログラムとＲＳＡの原論文の解説が載っている。[6]は読み物。一般の学生に勧める)。この他、ニューメキシコのサンタフェを中心に複雑系の理論が盛んになったこともあったが、多くを望み過ぎたこともあって、今日ではあまり振るわないようである([7][8]はこの方面の代表的な著書、[9]は短くて便利。簡単なプログラムが[10]にある。 [11]は最盛期のサンタフェ研究所を書いた読み物)。大学におけるコンピュータ教育も、

平成 16 年 5 月 1 日受理

＊ 理工学部数理科学科

＠佐賀大学理工学部

この多種な側面に照応するものでなければなるまい。その基礎は、プログラミング教育であろうが、その体制はいまだ整ってはいない。それは、高等学校、大学の教育にとって余りにも不幸なことである。書店に並んでいる高校数学の参考書には、コンピュータの項目はない。学校で教えないからだ。

平成 11 年 3 月学習指導要領が改定されて、情報 A、B、C　6 単位が高等学校で教えられることになった。その中には、公開鍵暗号の話もあるが、まことに総花的でこれでは教育効果は上がるまいと思われる。公開鍵暗号の仕組みは学生には分かりづらいようである。二三ページでは無理だろう。プログラミング教育について言えば、僅かではあるが数 B と情報 B、C にはプログラムが現れる。数 B では、「統計とコンピュータ」、「数値計算とコンピュータ」のなかで、簡単なプログラムを教えることになっている。また、情報 B の「プログラミング言語」では、条件分岐式、ループ、サブルーチンなどが、情報 C「情報機器の利用」では、HTML（ハイパーテクストマークアップ言語）を使ってホームページを作り情報を発信することが取り扱われている。情報 B で使用する言語については、現場の考えで決める趣旨であるが、要するに教科書ごとに使用する言語が異なってもよいということのようである。私の考えでは、高等学校でプログラミングを多少とも教えるならば、言語は統一すべきであると思う。そうしなければ、入試に出題も出来ないし、入試に出さなければ教えられることもなくなる。そのためには、どんな教室でも使用出来るものでなければならない。そういう言語の一つとして、ここではジャヴァスクリプトを挙げておく。HTML では繰り返したり、分岐したりすることは出来ないが、ジャヴァスクリプトを使えば、そういう事も可能である。その上、情報 B の「プログラミング言語」も含めて、ホームページの作成という形で、統一する事が出来る。新課程ではそこいらの整合性が欠けている感じがする。必須という点では、2 単位つまり A,B,C のうち 1 つをえらべばよいのだが、3 科目とることも出来る。全体の調和に一層の配慮が払われるべきであったかも知れない。実は、数 B もベーシックなどはやめにして、ここでもジャヴァスクリプトを使うことにしたら、高校での情報教育全体に幾分かの一貫性が出てくるのではないかと思っている。その視点で、数 B で扱うプログラムもジャヴァスクリプトで書いて、論文後半に載せておいた。ジャヴァスクリプトはネットスケープ社が無料で提供するプログラミング言語で、一定の方針に従って年々改良が加えられている。その点が、ただ古いだけの高校ベーシックと違っている。ジャヴァや C++とは、構築の仕方がまったく異なるといわれているが、形式的には似ている。だからそういう言語を後で学ぶ点でも役に立つと思う。以上のことを考慮した上でジャヴァスクリプトを提唱した次第である。

以下指導要領にしたがって、簡単なホームページ、ユークリッド互除法、面積計算について高校で扱い得るプログラムを挙げてみたが、同時に大学理工系課程におけるプログラミング教育ということをも念頭においてコードを作製した。実際、大学でこれらを教えるには一年 4 単位の講義が必要だろう。ここで使った言語は、ジャヴァスクリプト、エクセル VBA（マクロ）、ヴィジュアル　ベーシックであるが、ジャヴァスクリプトは、インタネットエクスプローラかネットスケープ　ナビゲータが入っていれば、オフラインで、つまりインターネットに繋がっていなくても、使用可能である。エクセルが入っているパソコンも多いが、そのときはマクロの形でベーシックが使える。癖の強いベーシックだが、初等的なところでは通常のベーシックと変わりは無く十分に役に立つ。バージョン 6 や．Net などのヴィジュアル　ベーシックを使える教室は少ないだろうが、諸賢の御参考に供すべくそれも載せてみた。ただし、バージョンは 6.0 である。コードは、大学の教室で教えるもの

として作製されている。大学でのプログラムの第一段階としては、ジャヴァスクリプトをとるか、エクセル画面でやるか、他の高級言語を使うかは別として、初歩としては、これで（教員養成以外の理工系の学科でも）十分だと思う。ここでは、まず、ジャヴァスクリプトによるコードを紹介し、次に同じ内容のものをエクセルマクロで書き、最後にヴィジュアルベーシックという順序になっている。高等学校の教壇では、それを適宜改変する必要があろうし、それも容易であろうが、念のため、若干のコメントを添えた。ジャヴァスクリプトのコードは、ホームページとして出力するさいのソースコードである。それは、HTML 言語の中に書かれている。最も簡単な HTML コードは

```
<html>
<head>．．．．．．．．</head>
<body> ．．．．．．．．．．．．．．．．．．
        </body>
</html>
```

の　．．．．．．．．．．．．．　の箇所を適宜埋めれば出来上がる。

<html>コードは、ネットスケープのコンポウザ　ページ（Composer Page）を使って書くのがもっとも便利だが、その<html>ソース ー タブをクリックすると、上のような枠が既に書かれてある。ジャヴァスクリプトは、エクセルＶＢＡのようにはロバスト（堅固）ではない。あまり無理はさせられない（たとえば、入れ子のループ）。ここにあげたコードが精一杯かもしれぬが、初歩の講義には便利であるし、十分でもある。その上、年々良くなっていくことは予想出来る。一般論としてコンピュータのソフトウェアというものは、考えられているほどロバストではない。その最大の理由は、ユーザーの入力予想というものが、あらゆる事を尽くして考えられているわけではないというところにある(**Validation** の問題)。コンピュータ　ヴィールスもそこを狙う。学生にも、コードを書かせる前に、論理と数理を完全に理解し、コンピュータに過剰な負担をかけさせないように努めさせるべきだ。

この論文で扱った問題は、次の３つである。

（1）　最大公約数をもとめる。任意に与えられた正整数 $m,n$ の最大公約数を、$m,n$ をキーボードから入力させて、結果を画面に表示させるものである。
（高等学校用のプログラムでは、$m,n$ の代わりに一定の数を最初の数より二番目の数が大きいように入れるようにして、関数 max,min を使うことを避ける。）

（２）　ユークリッド互除法の応用として逆元を求める問題を扱う。正整数ｍの剰余環　$\boldsymbol{Z}_m$　のなかで、ｍと互いに素な元の逆元を求めるのだが、素でないときは逆元がないとのメッセージが出る。この場合もｍ、逆元を求める元はユーザが入力する。RSA 公開鍵暗号で必要なコードである。

（3）　面積を計算する。中点公式、台形公式、シムプソンの公式を使って積分を求め、どれが一番精度が高いか比較する。この三つの近似の間には、昔から知られた関係がある。円周率の話になっているのは、関数$\sqrt{}(1-x^2)$の場合を例として扱ったからである。

ホームページのソースコードは、インターネット

エクスプローラならば、例えば、メモ帖に貼り付けて、適当な名前に拡張子 htm ないし html を付けて保存する。そのさい、保存した箇所(例えば、マイドキュメント等)を覚えておく。そして、インターネットのブラウザ(例えば、インターネット エクスプローラ )を立ち上げる。メニューバーの「ファイル」をクリック、現れたサブメニューから「開く」をクリックするとダイアログボックスが現れる。そこで、参照というボックスをクリックし、先ほど保存したコードを探し、それを開く。それだけのことである。ネットスケープならば、もっと簡単である。ファイルメニューの新規作成(new)から、コンポウザページ(Composer Page)を選んでクリックする。現れた画面下方の<html>ソース ー タブをクリックすれば、先に述べたようなソースの枠が現れる。その間をコードで埋めて「保存」すると(ネットスケープでは、その際拡張子をつける必要はない)、もう既にホームページの画面になる。Browse ボタンを押しても同じである。

また、エクセルの場合は、メニューバーから「ツール」ー「マクロ」を選び、さらに現れたプルダウンメニューから visual basic editor を選んでクリックする。上に General と書かれた白紙の標準モジュールが現れるから、そこに下の Euclid というコードを写して、マクロ Euclid を実行する。

ヴィジュアルベーシックでは、最初に現れた画面のダイアログボックスで新規作成の中の標準 EXE をダブルクリックすると、フォームが中央に現れて、デザインモードの画面になる。フォームに目的に応じてさまざまなコントロールと呼ばれるオブジェクトを植える。コントロールをダブルクリックするとそのコントロールのイヴェントプロシージャが、例えば

```
Private Sub Command1_Click()

End Sub
```

のように現れるから、その中にコードを書く。この例では、コマンドボタンをクリックすると、そのコードが実行されるという意味である。（ツールボックスが出ていないときは、マウスでメニューバーの上をロールオーバーして、ツールボックスというチップスが出たボタンをクリックするとよい）。このようにして、必要なコントロールを植え、以下のコードを打ち込んでいただく。下記では、コマンドボタンに名前をつけてあるから、例えば Command1_Click()は、右クリックしてプロパティを出し、最上部のオブジェクト名を変更し、更に “オブジェクト名_Click() ” と変えなければならない。そして実行する。数値はロング（long）であるから、ー21 億 5 千万から 21 億 5 千万まで入力出来る。

余裕があれば、その他に、擬似素数を生成するプログラム（例えば、Miller-Robin 素数判定法[4]をプログラムとして書かせる等）、そして生成された擬似素数を使って公開鍵暗号を作るプログラム（拙著、「0 と 1 との世紀」、近代文芸社、には、ヴィジュアルベーシックで書かれたプログラムがある。ご参考のほどを乞う）などを加える、といったところがよいのではないかと考える。高校、情報Cにも公開鍵暗号が紹介されているが、その仕組みが判らなければ、使うのも不安であろう。少なくとも教壇にたつ教師としては、理解しておくべきことの一つであろう。この論文の前編に述べた成績簿であるが、学生数が少なければ、データメニューから集計（subtotal）を選んで、必要ならば、データ ー フィルタ ー フィルタオプションを使うのも便利である。しかし、ただ便利だからといって、組み込まれた機能のみに頼るのは良くない。ソルヴァやシナリオなどのアドウィンを教えとしても、その背後にどのようなプログラムが隠れており、どのような理論があるのか、学生に考え

させ、自身のプログラムをつくる努力をさせることが肝要である。そうでなければ、確りとした応用も出来まいし、大学の教育とも言えまい。

最近の大学は、雑用が多くて、時間を教育の準備に割くゆとりも少なくなった。こんなプログラムでも多忙な諸氏には、存外に役に立つのではないかと思う。知られた情報を取捨選択して、一つに纏める。個々の情報価値は低くとも、集められることによって幾ばくかの価値が生じる。この論文をそういったものとお受け取り頂いて結構かと思う。しかし、ここで主張された事柄は重要である。情報三流国に堕したくないなら、高等学校も大学も、それを監督する官庁も今のままでは駄目である。

**Euclid_algorithm**

**(1.1)**　ジャヴァスクリプト

```
<html><head><title>最大公約数</title></head>
<body bgcolor = "beige">
<script language = "javascript">
<H1 align = center><font color = "brown">最大公約数</font></H1>
/*スクリプト言語は&lt;script&gt;タグとそのエンドタグである&lt;/script&gt;の間に書き込まれる。*/
a = prompt("正整数を入れなさい。");
b = prompt("正整数を入れなさい。");
/*入力した文字は数字であっても文字型として変数 a,b に入れられる。整数型に直すためには次のように型を
変換する必要がある。*/
c = parseInt(a);
d = parseInt(b);
m = Math.min(c,d);
n = Math.max(c,d);
m0 = m;
n0 = n;
/*ここでは、組み込み関数 Math.min 等を使ったが、min,max を直接定義するやり方については、後述のヴィ
ジュアルベーシックのコードを参照。高等学校用には for_loop のほうが良いだろう。*/
do{
r = n % m;
n = m;
m = r;}while( m > 0)
msg = "<BR><BR><BR><P align = “ center ”><B><font size = 6 color = “ darkbrown ”>"
```

```
msg = msg + m0 + " と " + n0 + " との最大公約数は " + n + " です。"
msg = msg + "</font></B></P>"
document.write(msg);
/* ホームページへの出力文字は、その仕方（中央揃え等）を含めて、msg のなかに書かれている。余り長いと良くないようで数段に分けて書いてある。*/
</script>
</body>
</html>
```

実行すると、公約数を求める 2 つの正整数の入力が求められる。63 と 467 を入力したときの実行画面は次のようなものだろう。読者は、現れる二つのダイアログボックスに任意の自然数を入れることができる。もっと大きな数、例えば、123454321 と 12345654321 とで試されよ。この場合結果は 1 、つまり互いに素になる。意外と素になるケースは多い。

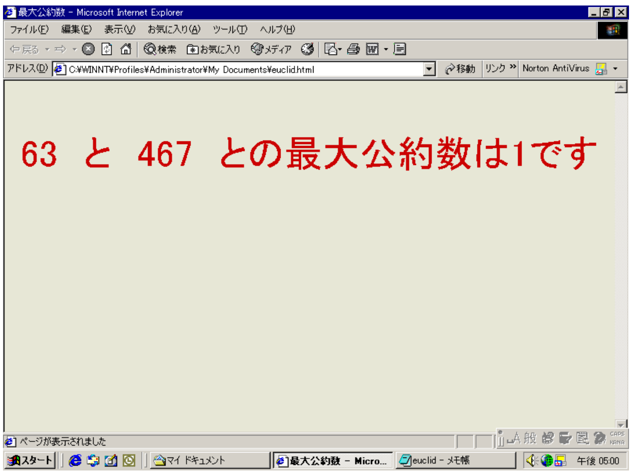


**Fig. 1** The display shows an illustration by the browser.

**(1.2)　エクセルマクロ**

```
Public m As Long
Public n As Long
Public m0, n0
Rem メモリを有効に使うためそれぞれの変数はスコープをもっている。つまり寿命である。下
Rem の a,b,c,d などはプロシージャ レベルのスコープでプロシージャ（この場合は、euclid と
Rem いうサブルーチン）が終われば消えてしまう。m,n,m0,n0 の場合はこれでは困る。関数
Rem  min,max でも使うからである。それは、サブルーチンの外にある。それで、Public と文
Rem 頭に定義をする。これは、プロジェクトが終わるまで寿命があるという意味である。

Sub ExcelMacro()
Dim a As String
Dim b As String
Dim c As Long
Dim d As Long
Worksheets(2).Activate
Rem ワークシートは通常３枚用意されている。その２枚目に出力。
a = InputBox("m (> 0) の値を入れよ。")
b = InputBox("n (> 0) の値を入れよ。")
e = CLng(a)
d = CLng(b)
m = min(e, d)
n = max(e, d)
m0 = m
n0 = n
For i = 1 To m0
r = n Mod m
If r = 0 Then Exit For
n = m
m = r
Next i
Range("A1") = "gcd(" & m0 & "," & n0 & ") "

Range("B1") = " = " & m
Rem　Range( “A1 ”) は Worksheets.Range( “A1 ”).Value の略で A1 セルに右辺が出力される。
```

```
End Sub
Rem 高等学校では、前にも注意したように常に m < n として、m = 35, n = 125 などと具体的
Rem に数字を予め決めて入力する。
Function min(x, y)

If x < y Then
min = x
Else
min = y
End If

End Function
Function max(x, y)

If x < y Then
max = y
Else
max = x
End If

End Function
```

(1.4.1) ヴィジュアルベーシック 1

```
Rem コマンドボタンを一つ植えて、コマンド 1 のキャプションを「実行」、名前を ArrayFor に変える。
Rem 実行ボタンをクリックすればダイアログボックスが現れて、二つの数を入力するよう求めてくる。
Public x As Long
Public y As Long
Public m0 As Long
Public n0 As Long
Private Sub Form_Load()

BackColor = vbCyan
WindowState = vbMaximized
```

```
End Sub

Private Sub ArrayFor_Click()

Dim a As String
Dim b As String
Dim c As Long
Dim d As Long
Rem ジャヴァスクリプトの prompt に対応するベーシックのコマンドは InputBox である。
Rem 右辺の括弧の中は、ダイアログボックスの中に表示される文言である。
a = InputBox("m (> 0) の値を入れよ。")
b = InputBox("n (> 0) の値を入れよ。")
Rem 文字型の変数 a,b はロングの変数 x,y に CLng を用いて変換される。
x = CLng(a)
y = CLng(b)
m0 = min(x, y)
n0 = max(x, y)
Call calculate1
Rem この Call は無くても良い。
End Sub

Private Sub calculate1()
Rem 配列は０から始まり 999 万 9999 まで。つまり最大な Size は 1000 万。
Dim m(10000) As Long
Rem 付記に述べるように、配列の大きさは、 [5 log10(m)] で十分である。Long データタイプは
Rem  2147483648 までの整数を表すわけだから、[5 log10(2147483648)] = 46 。ここで[ ]はガウスの記号
Rem である。したがって配列の大きさは、実際は 46 で十分である。
m(0) = m
m(1) = n
For i = 2 To x
m(i) = m(i－2) Mod m(i－1)
If m(i) = 0 Then Exit For
Next i
MsgBox (m0 & “　, ”　n0 & " の最大公約数は　" & m(i－2) & " です。")
```

```
End Sub
Function min(x, y)

If x < y Then
min = x
Else
min = y
End If

End Function
Function max(x, y)

If x < y Then
max = y
Else
max = x
End If

End Function
```

(1.4.2) ヴィジュアルベーシック 2

```
Private Sub ForLoop_Click()

Dim a As String
Dim b As String
Dim c As Long
Dim d As Long
a = InputBox("m (> 0) の値を入れよ。")
b = InputBox("n (> 0) の値を入れよ。")
x = CLng(a)
y = CLng(b)
n = max(x, y)
m = min(x, y)

Call calculate2
```

```
End Sub
Sub calculate2()

For i = 1 To m
r = n Mod m
If r = 0 Then Exit For
n = m
m = r
Next i
MsgBox (m0 & “　, ”　n0 & " の最大公約数は " & m & " です。")

End Sub
Function min(x, y)

If x < y Then
min = x
Else
min = y
End If

End Function
Function max(x, y)

If x < y Then
max = n
Else
max = x
End If

End Function
```

(1.4.3)　ヴィジュアルベーシック 3

```
Private Sub DoLoopv_Click()

Dim a As String
Dim b As String
Dim c As Long
Dim d As Long
a = InputBox("m (> 0) の値を入れよ")
b = InputBox("n (> 0) の値を入れよ")
x = CLng(a)
y = CLng(b)
m = min(x, y)
n = max(x, y)
m0 = m
n0 = n
End Sub
Sub calculate3()

Do
r = n Mod m
n = m
m = r
Loop Until m = 0
MsgBox (m0 & “　, ”　n0 & " の最大公約数は　" & m & " です。")
End Sub
Rem このように do－loop を使うのが最も良いとおもう。数学との相違について学生に考えさ
Rem せると良い。
Function min(x, y)

If x < y Then
min = x
Else
min = y
End If
```

```
End Function
Function max(x, y)

If x < y Then
max = y
Else
max = x
End If

End Function
```

剰余環の中で逆元を求める

(2.1)　ジャヴァスクリプト

剰余環のなかで逆元を求める問題をコンピュータで解く。これは公開鍵暗号をプログラムするとき必要不可欠なプログラムであるが、抽象代数学としても興味のある話題である。

```
<html><head>
<script language = "javascript">
var m;
var n;
var m0;
var n0;
var msg1,msg2,msg3,msg4;
a = prompt("逆元を求める元は");
b = prompt("剰余環の元の数は");
m = parseInt(a);
n = parseInt(b);
m0 = m;
n0 = n;
msg1 = n0 + " の剰余環の中で " + m0 + " の逆元を求める。";

document.write("<BR><BR><P align = ‘ center’ ><font color = ‘ blue’  size = ‘ 8’ >" +
msg1 + "</font></P>");
</script></head>
<body bgcolor = "beige"><script language = "javascript">
```

```
if (0 < m && m < n){
/*ampasand 二つ並べたものは論理積を表す。つまり “かつ ”ということ。 */
s = 0;
p = 1;
   do {
   r = n % m;
   q = Math.floor(n/m);
/*Math はジャヴァスクリプトの組み込みオブジェクトである。丸めるときは Math.round を使う。*/
   temp = p;
/* p の値は次式で変わってしまうから現在の値を変数 temp に保存。*/
   p = s － q*p;
   s = temp;
   n = m;
   m = r;
      if (m == 1 && p<0){
         p = p + n0;
      }
   } while (1<m);
   if (m == 1){
   msg2 = m0 + " の mod " + n0 + " での逆元は " + p + " です。";
     msg2 = "<P align = ‘ center‘ ><font color = ‘ blue‘  size = ‘ 6‘ >" + msg2 + "</font></P>";
   document.write(msg2);}
   else {
   msg3 = m0 + " と " + n0 + " との最大公約数は " + n + " です。<BR>";
   msg3 = "<P align = ‘ center‘ ><font color = ‘ blue‘ >" + msg3 + m0 + " の逆元はありません。</font></P>" ;
   document.write(msg3);
   }
}
else {
msg4 = "<P align = center><font color = blue>0 &lt; m &lt; n ではありません。<BR>";
msg4 = msg4 + "もう一度、初めからやり直して下さい。</font></P>";
document.write(msg4);}
</script></body></html>
```

(2.2) ヴィジュアルベーシック

```
Rem  0 < m < n とする。 m の法 n での逆元を求める。逆元をもたないときは、
Rem その旨メッセージボックスによって伝える。
Public m As Long
Public n As Long
Public m0 As Long
Public n0 As Long

Private Sub DoLoop_Click()
Rem DoLoop はコマンドボタンにつけられた名前。
Dim a As String
Dim b As String
Dim c As Long
Dim d As Long
a = InputBox("m (> 0) の値を入れよ。")
b = InputBox("n (> m) の値を入れよ。")
m = Clng(a)
n = Clng(b)
If m <= 0 Or m > n Then
msg = "0 < m < n ではありません。"
msg = msg + "もう一度、初めからやり直して下さい。"
MsgBox msg
Exit Sub
End If

m = CLng(a)
n = CLng(b)
m0 = m
n0 = n
calculate

End Sub
Sub calculate()
```

```
s = 0
p = 1
Do
r = n Mod m
q = Int(n / m)
temp = p
p = s － q * p
s = temp
n = m
m = r
If m = 1 Then

  If p < 0 Then
  p = p + n0
  End If

MsgBox m0 & " の mod " & n0 & " での逆元は " & p & "です。"

End If

Loop Until m = 0

If n > 1 Then

msg = m0 & " , " & n0 & " の最大公約数は " & n & " です。"
msg = msg & m0 & " は逆元をもちません。"
MsgBox msg

End If
End Sub
```

まず、356 と代入し、続いて 645 と入れると次のように逆元 491 がメッセジボックスの中に現れる。

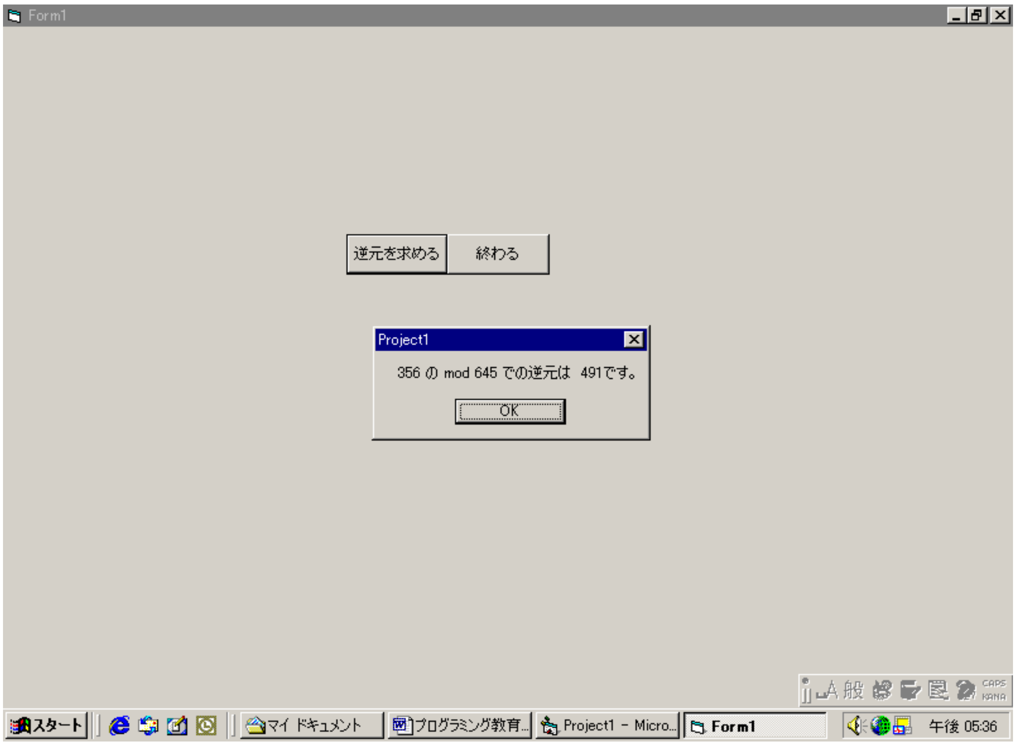


**Fig. 2** The display shows this illustration, which includes a dialog box.

また 12，150 なる２つの数をこの順序でいれると画面は次のようになって､逆元が存在しない旨が伝えられる。

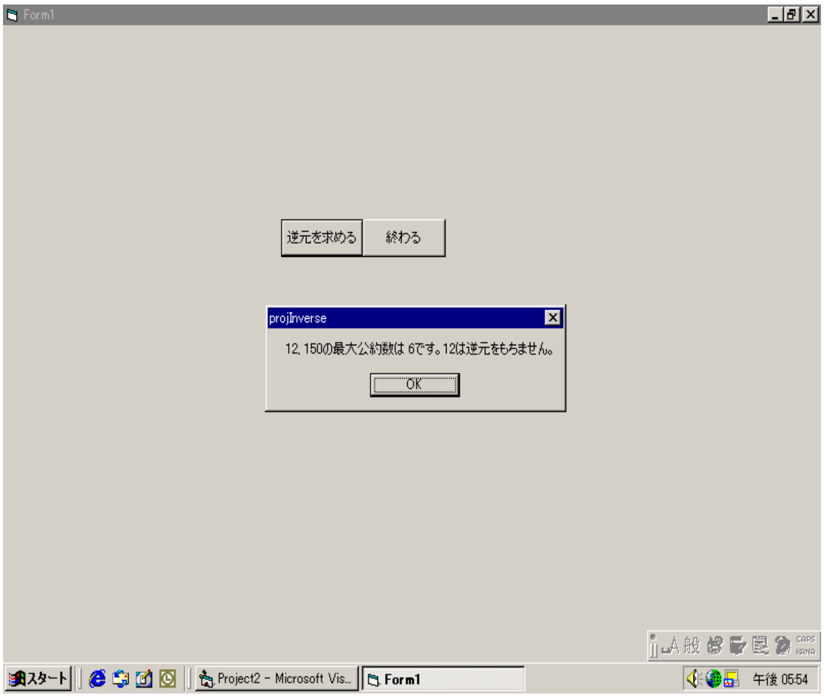


**Fig. 3** Another illustration.

面積の計算

(3.1) ジャヴァスクリト(台形公式)

```
<html><head><script>
var x;
var sum;
function func(x){
y = Math.sqrt(1−x^2);
return y;
}
</script>
</head>
<body>
<script>
a = −1;
b = 1;
```

```
h = (b － a)/n;
sum= (func(x) + func(b))/*h/2;
n = 100000;
x = a
for(i = 0; i < n; i++){
x = x + h
sum = sum + h*func(x);}
document.write(sum);
</script>
</body></html>
```

(3.2) エクセル

```
function func(x)
func = sqr(1 －  x^2)
Rem このところをいろいろな関数で変えてみる。
end function
Sub integral()
a = －1
b = 1
Rem  [a,b]は積分区間
n = 10000000
h = (b － a) / n
Sum = (func(a) + func(b))*h/2
y = a
For i = 1 To (n– 1)
y = y + h
Sum = Sum + func(y)*h
Next i
Range("A1") = "円周率＝ "
Range("B1") = 2 * Sum
End Sub
```

(3.3) ヴィジュアルベーシック（３つの方法による数値積分）([13]を参照した)

```
Rem ツールボックスからテクストボックス４個とコマンドボックス２個をフォーム上に植え
Rem る。各テクストボックスに３方法による結果が書き込まれる。
Dim x As Double
Function func(x)
  func = Sqr(1 － x ^ 2)
End Function
Rem  func をいろいろな関数に変えてみる。

Private Sub Command1_Click()
a = －1
b = 1
h = b － a
e = 0.000001
t = (func(a) + func(b)) / 2
Do
m = 0
Rem 常に初期化しなければならぬところが、中点法の悪いところである。中点は次の分割
Rem の中点ではない。
For x = a + h / 2 To b Step h
m = m + func(x) * h
Next x
s = (t + 2 * m) / 3
t = (t + m) / 2
h = h / 2
Loop While Abs(t － m) / Abs(s) > e
Text1.Text = "中点方式 " & 2 * m
Text2.Text = "台形方式 " & 2 * t
Text3.Text = "シンプソン方式 " & 2 * s
Text4.Text = "実際の円周率 " & 3.14159265358979
Rem ３つの法式のうちどれが精度が高いか比較する。
End Sub

Private Sub Command2_Click()
End
```

```
End Sub

Private Sub Form_Load()
WindowState = vbMaximized
Text1.Left = ScaleWidth * 2 / 3
Text2.Left = ScaleWidth * 2 / 3
Text3.Left = ScaleWidth * 2 / 3
Text4.Left = ScaleWidth * 2 / 3
Text1.Top = ScaleHeight * 2 / 3
Text2.Top = ScaleHeight * 2 / 3 + 500
Text3.Top = ScaleHeight * 2 / 3 + 1000
Text4.Top = ScaleHeight * 2 / 3 + 1500
Text1.Height = 400
Text2.Height = 400
Text3.Height = 400
Text4.Height = 400
Text1.Width = ScaleWidth * 2 / 3
Text2.Width = ScaleWidth * 2 / 3
Text3.Width = ScaleWidth * 2 / 3
Text4.Width = ScaleWidth * 2 / 3
Command1.Top = Text4.Top + 1500
Command2.Top = Text4.Top + 1500
Command1.Left = ScaleWidth * 2 / 3 + 3000
Command1.Width = 1500
Command2.Left = ScaleWidth * 2 / 3 + 4500
Command2.Width = 1000
Command1.Height = 400
Command2.Height = 400
Command1.Caption = "円周率の計算"
Command2.Caption = "終わり"
Text1.Text = "中点方式"
Text2.Text = "台形方式"
Text3.Text = "シンプソン方式"
Text4.Text = "実際の円周率"
End Sub
```

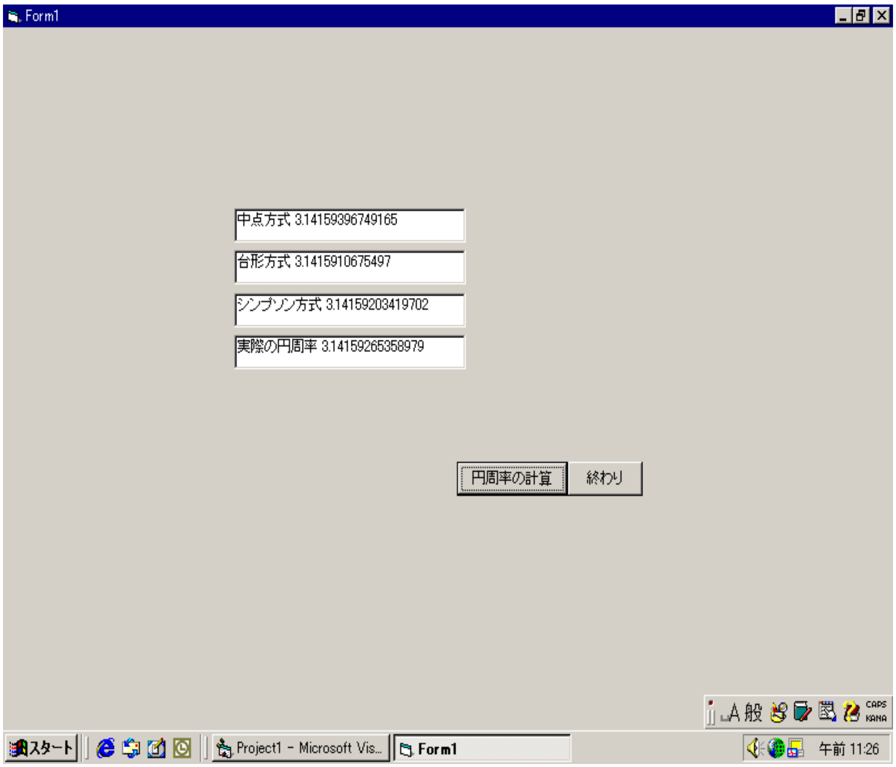


**Fig3** This illustration shows the result when the program is executed

## 付記 ユークリッドのアルゴリズムと配列の大きさ (Size)

正整数の列 $m_0, m_1, \ldots\ldots, m_k$ が、各 $i = 1, \ldots\ldots, k-1$ に対して、 $m_{(i-1)}$ を $m_i$ で割った余りが $m_{(i+1)}$ になっているとき、それを剰余列と呼ぼう。さらに、$m_{(k-1)}$ が $m_k$ で割り切れるならば、その剰余列は閉じているということにしよう。

正整数 $n > m$ が与えられたとき、それによって一意的に $q, r$ が定まるが、$r \neq 0$ ならば、さらに

$$m = q'r + r' \quad , 0 \leqq r' < r$$

として、剰余が0になるまで同じ演算を繰り返すことができる。そのさい

$n = m_0, m = m_1, q = q_1, r = m_2, q' = q_2$,

$r' = m_3, \ldots\ldots$

と置くと

$$m_{(i-1)} = q_i m_i + m_{(i+1)}$$

$$0 < m_{(i+1)} < m_i$$

$$(i = 1, 2, \ldots\ldots, k-1)$$

$$m_{(k-1)} = q_k m_k$$

が成り立つ。ここで $2 \leqq q_k$ に注意する。要するに 正整数 $m < n$ に閉じた剰余列

$$m_0, m_1, m_2, \ldots\ldots, m_k$$

が定まったことになる。このように $m,n$ から閉じた剰余列を作る手続きを、ユークリッドのアルゴリズムという。k は手続き中のステップの回数である。この同じ手続きを s 変数の多項式 $Q_s(x_1, \ldots ,x_s)$ に拡張しよう。そのためには

$$Q_{-1} = 0$$

$$Q_0 = 1$$

$$Q_1(q_k) = q_k$$

$Q_s\,(q_{(k\text{-}i+1)}, \ldots\ldots, q_k) =$
$q_{(k\text{-}i+1)}\, Q_{(i\text{-}1)}(q_{(k\text{-}i+2)}, \ldots\ldots, q_k) +$
$Q_{(i\text{-}2)}(q_{(k\text{-}i+3)}, \ldots\ldots, q_k)$

$$(i = 1,2, \ldots\ldots, k)$$

なる再帰式を用いればよい。そのとき、例えば

$$Q_s\,(q_{(k\text{-}1)}, q_k) = q_{(k\text{-}1)}\, q_k + 1$$

となる。以下、誤りの恐れがないときは、$Q_s\,(q_{(k\text{-}i+1)}, \ldots\ldots, q_k)$ の代わりに、単に $Q_s$ と書くことにする。m、$n$ の最大公約数 $gcd(m,n)$ を d で表し

$$\mathrm{m}_i = Q_{(k\text{-}i)}d \;\; (i = 0, \ldots\ldots, k)$$
$$n = qm + r \quad , 0 \leqq r < m$$

とおくと

$$\mathrm{m}_k = Q_0\,\mathrm{d} \;\; = d$$
$$\mathrm{m}_{(k\text{-}1)} = Q_1 d = q_k\,\mathrm{m}_k$$
$$\mathrm{m}_{(k\text{-}2)} = Q_2\,\mathrm{d} = q_{(k\text{-}1)}\, Q_1(q_k)d + d$$
$$= q_{(k\text{-}1)}\,\mathrm{m}_{(k\text{-}1)} + \mathrm{m}_k$$

が得られるが、一般に

$$\mathrm{m}_i = Q_{(k\text{-}i)}d = q_{(i+1)}\, Q_{(k\text{-}i\text{-}1)}\, d + Q_{(k\text{-}i\text{-}2)}$$
$$= q_{(i+1)}\,\mathrm{m}_{(i+1)} + \mathrm{m}_{(i+2)}$$

が $i = 0, 1, 2, \ldots\ldots, k-2$ に対して成り立つ。これはユークリッドの手続きを降順に書き直したものに他ならない。ゆえに

$$n = \mathrm{m}_0 = Q_k\,\mathrm{d}$$
$$m = \mathrm{m}_0 = Q_{(k\text{-}1)}\,\mathrm{d}$$

となる。

$$1 \leqq q_1, 1 \leqq q_2, \ldots\ldots, 1 \leqq q_{(k\text{-}1)},$$
$$2 \leqq q_k$$

であるから、常に

$$Q_i(1, 1, \ldots\ldots, 1, 2)$$
$$\leqq Q_i(q_{(k\text{-}i+1)}, \ldots\ldots, q_k)$$

となる。以下

$$G_i = Q_i(1, 1, \ldots\ldots, 1, 2)$$

とおくと　$G_{\text{-}1} = 0, G_0 = 1, G_1 = Q_1(2) = 2$

であり、$i = 2, \ldots\ldots, k$ にたいして

$$G_i = G_{(i\text{-}1)} + G_{(i\text{-}2)}$$

が成り立つから

$$G_2 = 3, G_3 = 5, G_4 = 8, \ldots\ldots$$

を得る。これとフィボナッチ数 $F_i$ の定義[12]とを比較して

$$G_i = F_{(i+2)}$$

であることが分かる。フィボナッチ数については、関係式

$$F_i = (1/\sqrt{5})((\tau_+)^i - (\tau_-)^i)$$

が知られている。ここで

$$\tau = \tau_+ = (1 + \sqrt{5})/2$$

は古来黄金比として知られているものである。また

$$\tau_- = (1 - \sqrt{5})/2$$

である。

$$(1/\sqrt{5})(\sqrt{5} - 1)/2 < 1/2,$$
$$(1/\sqrt{5})\{(\sqrt{5} - 1)/2\}^2 < 1/3$$

であることに注意する。したがって $G_i$ は $(\tau)^{i+2}/\sqrt{5}$ に最も近い整数である。

$$m = Q_{(k\text{-}1)}\,(q_2, \ldots\ldots, q_k)d >$$
$$Q_{(k\text{-}1)}\,(1, \ldots\ldots, 1, 2) = G_{(k\text{-}1)}$$
$$n = Q_k\,(q_1, \ldots\ldots, q_k)d >$$
$$Q_k\,(1, \ldots\ldots, 1, 2) = G_k$$

であるから

$$m > \tau^{k+1}/\sqrt{5} - 1/3,$$
$$n > \tau^{k+2}/\sqrt{5} - 1/3$$

あるいは

$$\ln(\sqrt{5}\,m + 1/3)/\ln\tau > k+1,$$
$$\ln(\sqrt{5}\,n + 1/3)/\ln\tau > k+2$$

平均値の定理より

$$\ln(x+a) = \ln x + a(1/y)$$
$$x < y < x+a$$

したがって

$$(\ln\sqrt{5}\,m + 1/3)/\ln\tau > k+1,$$
$$(\ln\sqrt{5}\,n + 1/3)/\ln\tau > k+2$$

$1/3 \ln\tau < 1/(3\times0.5) = 2/3$ だから

$(\ln\sqrt{5}\,m)/\ln\tau > k+1/3$ ,

$(\ln\sqrt{5}\,n)/\ln\tau > k+4/3$

を得る。また

$(\ln\sqrt{5}\,a)/\ln\tau = 4.785\ldots\ldots \log_{10} a$

だから、結局

$k+1/3 < 5\log_{10} m$ , $k+4/3 < 5\log_{10} n$

となる。この両不等式を同時に満たす $k$ のうち最大のものをとるのである。したがって

$$\boldsymbol{k}_1 = [5\log_{10}\boldsymbol{m}]\,,\ \boldsymbol{k}_2 = [5\log_{10}\boldsymbol{n}]-1$$

のうち小なるものをとればよいということになる。結局は、$\boldsymbol{k}_1$ でよい。大雑把に言って、配列の大きさとして 10 進数で表したときのｍの桁数に 1 を加えたものの 5 倍をとればよいということである。

## 参考文献


[0] 前橋敏之 「大学教育とコンピュータ」 佐賀大学理工学部集報 第 32 巻第 2 号 41 －52

[1] B.B.マンデルブロー 「フラクタル幾何学」 日経サイエンス

[2] E.N.ローレンツ 「カオスのエッセンス」 共立出版 1997 年

[3] 前橋敏之 「カオスの系譜」 近代文芸社 1996 年

[4] D.R.スチンソン 「暗号理論の基礎」 共立出版 1996 年

[5] 前橋敏之 「０と１との世紀」 近代文芸社 2000 年

[6] S.シン 「暗号解読」 新潮社 2001 年

[7] R.アクセルロッド 「つきあい方の科学」 HBJ 出版局 1987 年

[8] S.カウフマン 「自己組織化と進化の論理」 日本経済新聞社 1999 年

[9] M.A.ノワック R.M.メイ K.ジグムント 「“囚人のジレンマ”と生物の進化」 日経サイエンス 1995 年 8 月

[10] A.L.ロイド 「囚人のジレンマ ゲームを作ろう」 日経サイエンス 1995 年 8 月

[11] M.M.ワールドロップ 「複雑系」 新潮文庫 2000 年

[12] D.E. クヌース The art of computer Programming = 1,4 サイエンス社 1958,1966 年

[13] J.G.ケメニ T.E.カーツ 「ベーシック入門」 共立出版 1971 年