「数式処理ソフト」の制作を指導して

北陸ポリテクカレッジ
（北陸職業能力開発大学校）
相川　政和

１．はじめに

当校へ着任してからの過去４年間の総合制作を振り返ってみますと，学生の力を引き出して良い作品を制作させることができたと思うこともあれば，うまく指導できず，たいした作品も作らせられずに終わってしまったこともありました。そんな中，平成15年度卒業生の作品「数式処理ソフト鱒の助」はうまくいったと思う制作の１つです。

今回原稿執筆の機会を与えていただきましたので，この時の制作を振り返ってみると同時に，当校の学生が情熱を傾けて制作した作品を紹介させていただこうと思います。

２．１年次のプログラミング教育

まず，総合制作に入る前の状況を纏めてみたいと思います。当校情報技術科で１年次に習うプログラミング言語というとＣのみです。プログラミングに直接関係する教科目としては「ソフトウェア制作実習Ｉ」（４単位），「アルゴリズム基礎」（４単位），「データ構造・アルゴリズム」（４単位），「データ構造・アルゴリズム実習」（４単位），計算機命令実習（２単位）などがあります。

入学前に多少プログラミング経験を持つ学生も存在するものの，ほとんどの学生にとってプログラミングは初めてか，もしくはわずかな経験しかありません。そんな学生達も，これらの科目を通して，プログラミング教育をみっちり受けた結果，２年生に進級する頃には，ある程度のプログラミング力が培われています。

といっても１年間で学んだことだけで何かを作ろうとしてもかなり制約されてしまい，総合制作に相応しい作品はなかなか作れません。従って，２年生になれば直ちに制作が進められるということはなく，開発テーマに応じて，数ヵ月間の学習期間を設定し，より多くのことを学んでから制作に入ります。

３．目標

さて，「数式処理ソフト鱒の助」の当初目標は「数式のグラフ化や微分方程式の解曲線表示などの機能を持つアプリケーションソフト」を制作することでした。例えば，ｙ=sin（ｘ+ｚ）*cos（ｘ−ｚ）のような数式を対話的に入力し，画面上にその数式を三次元的な曲面で表示するということです。その他の機能についても時間の許す限り充実させ，Mathematicaのような商用数式処理ソフトに負けないアプリケーションを目指そうと目標を定めました。

世の中にはすでに高機能な商用数式処理ソフトがいくつか存在し，数式処理ソフトの制作に新規性はありませんが，それを制作するうえで十分な難易度とエッセンスがあります。そのため，制作を通じて学生の技術力向上が期待できます。また，多くの学術研究機関では数式処理ソフトが導入され，教育や研究に役だてられています。私が昨年まで担当していた数値解析／数値計算実習では専らＣを使い実習を行っていますが，数式処理ソフトがあれば，この授業に役だてることができるとも考えました。

４．学生のプロフィール

数式処理ソフトの制作に携わったのは普通高校出身の新浜君と商業高校出身の田中君の２名でした。１年次の成績は２人とも特別優秀ということはありませんでしたが，ガッツには恵まれていました。特に新浜君はスイミングスクールのコーチをしているほどのスポーツマンで，体力も根気もずば抜けているところがありました。田中君も責任感が強く，最後まで自分の責任を全うしてくれました。余談ではありますが，部活動で厳しく鍛えられてきた学生はやる気になったときに，大変力を発揮するというのがここ数年で感じた私の印象です。

５．開発言語及び技術要素の習得

私のグループでは例年JavaまたはC++を主力開発言語として使っています。従って，初めに，これら言語の勉強とこれら言語を使いこなすうえで必須のオブジェクト指向について学習する期間を２～３月設けています。

その後，テーマごとに必要となる技術要素を学習するよう指導します。数式処理ソフトの制作では入力された数式を解釈する言語処理について習得する必要がありました。そこで，四則演算と括弧の使える数式アナライザを制作させつつ，字句解析や構文解析のやり方を指導しました。

６．インクリメンタルな制作手法

技術要素の習得過程で工夫したのは，最終目的の仕様を満たすソフトウェアを一気に作るのではなく，小さな目標を定めて，段階的に進むインクリメンタルあるいはスパイラル的な制作手法を取り入れた点です。例えば数式アナライザの制作では，最初に字句解析までのプログラムを作り，次に加算プログラムへ作り変え，さらに減算をサポート，乗除算をサポート，括弧をサポート，変数をサポート，……というように少しずつ機能を膨らませていきます。このような一見遠回りにみえる制作手法ですが，学生の理解向上に役だつだけでなく，ゴールへ無理なく到達できるようになります。

７．設計の見本を示す

また，技術要素の習得段階では学生に自由に設計させず，私が設計を行い，クラスのインタフェース仕様を提示しました。そして，学生には仕様通りに実装させるようにしました。というのも，自由に作らせた場合，外部的には要求通りのものができたとしても，内部の設計は滅茶滅茶で，下手をすれば，１つのクラスの１つのメソッド内ですべてを行ってしまうこともあります。でき上がったコードを眺めても，ネストが深く容易には把握できません。しばらく経てば作った本人でさえも悩ましく，機能追加を行うのは至難の業といえます。

かの山本五十六が「やってみせ，言って聞かせて，させてみて，ほめてやらねば人は動かじ」と言っているように，まずはやってみせることが必要だと思います。抽象的なオブジェクト指向設計論を唱えてみても具体的な設計事例を見せないと，その効果も使い方も経験の浅い学生には浸透しないというのが私の経験です。

また，UMLで記述したクラス図やインタフェース仕様書などを学生に提示することで，学生にドキュメントの見方・作り方を学ばせる効果もあります。以後学生には同様なドキュメントを作りながら制作を進めるよう指示します。

８．指導力不足

ここで，私の指導力のなさについて触れておきたいと思います。私の場合，技術要素の習得期間までは学生に密着して指導し，制作に入ってからは様子を見守っているだけというのが例年のスタイルです。というのも言語の習得段階では異なるテーマでも同じ言語を使う場合には，まとめて指導できる一方，各テーマ毎の制作が始まると，正直手に負えなくなるためです。本来ならば学生の設計やコードにまで目を光らせて，悪い点を逐一指摘しなければ高い教育効果は得られない考えていますが，そのような余力がなく，今後改善を図っていくべき点と自認しております。

平成15年度においては，制作テーマを３つに絞り，

私自身の指導力の分散を防いだことにより，なんとかある程度の指導を行うことができました。しかし，依然指導が手薄であることは否めません。周囲を見渡してみても，放任状態になっている光景をしばしば目にします。

９．分担

学習期間が長引いたため，制作に取り掛かったのは確か９月頃からだったと思います。初めは試作品として開始しましたが，そのまま本作品へ発展させていきました。

高い生産性を実現するために，当然ではありますが，各自の担当箇所を決め，平行して作業を進めました。新浜君が数式の解析部を，田中君がGUI部を担当しました。グラフ表示部は，前年度の総合制作の成果であるNOAH3D APIを拡張して利用することにしました。この部分だけは私が担当しました。開発はJavaで行い，下位層のAPIとしてはNOAH3DとJava標準のAPIのみを利用しました。NOAH3D自体Java標準APIのみで作られていますから，この数式処理ソフトはPure-Javaです。

制作の主要部分は学生に任せつつも，部分的に指導員が入り込み，良いものを作りたいという熱意をみせることで，学生も熱心に取り組んでくれます。先述のように，きめの細かい指導はできませんでしたが，私自身がこの制作に情熱を傾けていることを随所で伝えることで，学生自身も熱くなってくれたように思います。

１０．鱒の助の紹介

ここからは，学生が熱意を傾けて制作した数式処理ソフト「鱒の助」を紹介させていただきます。

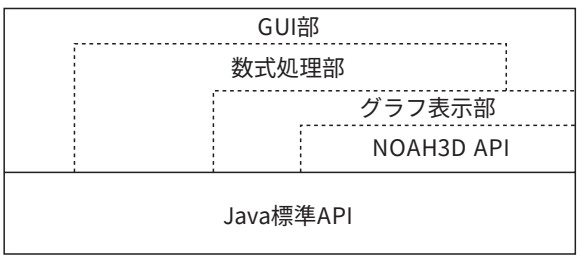


図１　モジュール構成

10.1　主な機能

鱒の助が提供する機能としては，算術式の解析と実行，変数，組込定数，組込関数，関数定義，総和，直積，微分，積分，微分方程式，回帰分析，相関係数，グラフ表示，グラフ操作，コマンド履歴，マクロ，プロファイル，入力支援，オプション設定，画像出力，編集（切取，複写，貼付），ログ出力，その他各種コマンドがあります。以下，いくつかの機能を紹介します。

10.2　メインウィンドウ

図２が鱒の助のメインウィンドウです。画面中央が入力領域です。利用者はプロンプト（^_^）の右側に数式やコマンドを入力します。矢印キー↑↓を押せば，過去に入力したコマンドが表示されます（コマンド履歴機能）。

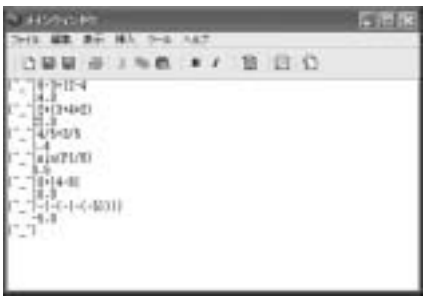


図２　メインウィンドウ

10.3　算術演算とサポート演算子

鱒の助は表１の演算子をサポートしています。この表の上に位置する演算子ほど優先順位が高いことを意味します。算術式の入力例を以下に示します。

```
(^_^) -3*|(8-12)*2+1|
          -21.0
(^_^) (5-3)*(2+4)
          12.0
```

表１ サポート演算子

	演算子	説明	結合
1	( ) ||	括弧、絶対値	→
2	! !! #	階乗	←
3	^	累乗	←
4	* / %	乗除余	→
5	+ -	加減	→
6	=	代入	←

(^_^) 5!
120.0

10.4 変数／組込定数

計算の途中経過を変数に保存することができます。また，円周率や重力加速度など有用だと思われる物理定数が組み込まれています。

(^_^) var=3*4−7
5.0
(^_^) var+4
9.0
(^_^) PI
3.14159265359
(^_^) g
9.81

10.5 組込関数

三角関数，対数関数，統計関数等数多くの組込関数を提供しています。以下に一部の関数の使用例を示します。lnは自然対数，logは常用対数です。factorは素因数分解です。

(^_^) sin(PI/6)
0.5
(^_^) sqrt(2)
1.414213562373
(^_^) y=exp(3)
20.08553692319
(^_^) E*E*E
20.08553692319
(^_^) ln(y)
3.0
(^_^) log(10000)
4.0
(^_^) ceil(3.2)
4.0
(^_^) floor(3.2)
3.0
(^_^) factor(2730)
2 3 5 7 13
5.0

10.6 関数定義

既存の関数や数式を組み合わせて新しい関数を定義することができます。例えば，ｙ=2x＋１を定義する場合，def ｙ(ｘ)＝２＊ｘ＋１ と入力します。関数定義後，ｙ(3)と入力すれば関数値が計算され結果が表示されます。この例では２＊３＋１の結果７が表示されることになります。変数は複数指定できます。

(^_^) def f(x)=x*(x−2)−1
(^_^) f(1)
−2.0
(^_^) def g(x)=|f(x)|+sin(x)
(^_^) g(2)
1.909297426826
(^_^) def h(x,y)=|x−y|
(^_^) h(3,12)
9.0

10.7 数値計算

方程式や微分方程式の数値解，微分係数値，定積分値等を計算することもできます。その例を示します。differで微分係数を，integralで定積分値を求めることができます。

(^_^) def f(x)=x^2
(^_^) differ(f(0))
0.0
(^_^) differ(f(1))
2.0
(^_^) integral(0, 1, f)
0.333333333333

integral (0, 1, f) は，０から１までの範囲で，関数ｆを定積分することを意味します。

10.8 関数のグラフ化{y=f(x)}

関数ｙ=ｆ(ｘ)をグラフ化するのも大変簡単です。関数を定義後，graphコマンドのパラメータで関数名を与えるだけです。

(^_^) def ｆ(ｘ)=２＊sin(12＊ｘ)＊sin(ｘ)
(^_^) graph ｆ

実行結果を図３に示します。グラフの表示倍率や

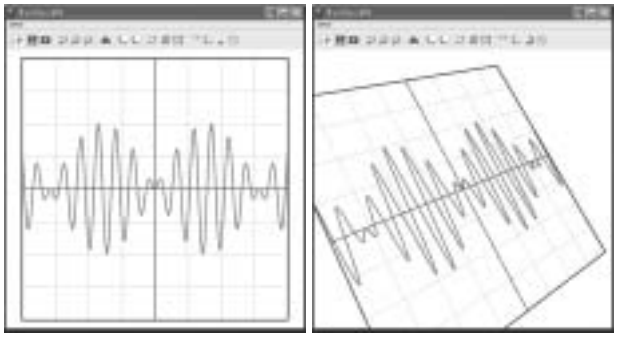


図３　グラフ出力例１

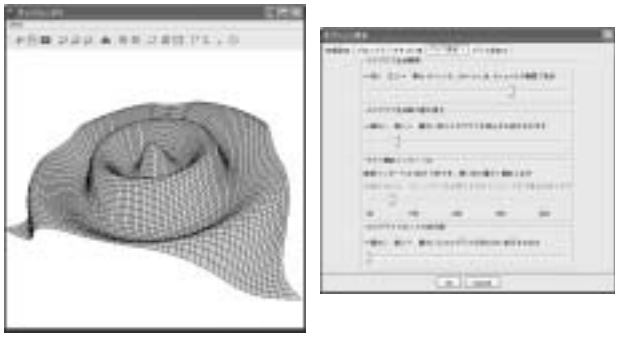


図４　グラフ出力例２とオプション設定

位置などはツールバーのコマンドを使い，簡単に変えられます。また，マウス操作やツールボタンにより，表示面を回転させることもできます。

10.9　二変数関数のグラフ化{y=f(z，x)}

二変数関数 y=f(z，x) を曲面で表現します。出力手順はy=f(x) の場合と同様です。

(^_^) def y(z，x)=cos(2*sqrt(x^2+z^2))
(^_^) graph y

図４に実行結果を示しますが，図形の生成範囲と生成時の面の細かさはオプション設定で変えることができます。

10.10　微分・積分

関数 y=f(x) をグラフ上で微分・積分する機能もサポートしています。y=x*(x－2) を微積分する例を以下に示します。

(^_^) def f(x)=x*(x–2)
(^_^) graph f
(^_^) graph color red
(^_^) def g(x)=differ(f(x))
(^_^) graph g

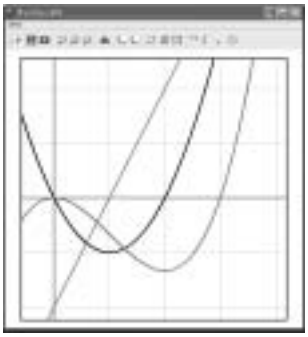


図５　グラフ出力例３（微分・積分）

(^_^) graph color green
(^_^) graph integral(f)

differ(f(x)) はf(x) の導関数を意味します。

def h(x)=integral(0，x，f)により，関数fの定積分関数hを定義した後，グラフ化することもできますが，この方法ではグラフ化に時間がかかりますので，graph integral(f)により，積分グラフの生成を可能にしました。ちなみに，この機能はx=0からの定積分値をグラフ化します。

10.11　微分方程式の解グラフ

微分方程式の解グラフを出力する機能もサポートしました。二階の常微分方程式まで扱うことができます。

10.12　パラメトリック曲線

y=f(x) 形式の関数だけでなく，媒介変数（パラメータ）で表される{x=x(t)，y=y(t)，a≦t≦b}の形式の曲線も描けます。この曲線を描くには，関数x(t) とy(t) を定義した後，graph x:y[a，b]と入力するだけです。a，bはパラメータtの範囲を指定します。以下にパラメトリック曲線

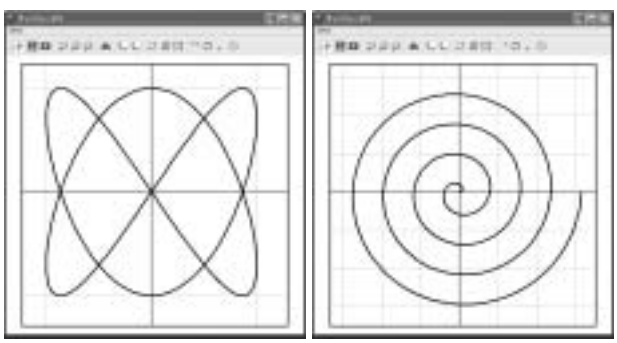


図６　グラフ出力例（リサージュと螺旋）

の描画例を示します。

```
(^_^) def x(t)=sin(2*t)
(^_^) def y(t)=sin(3*t)
(^_^) graph x:y[0, 2*PI]
(^_^) def g(u)=u*cos(u)/8
(^_^) def h(u)=u*sin(u)/8
(^_^) graph g:h[0, 8*PI]
```

三次元パラメトリック曲線の描画例を示します。

```
(^_^) def r(t)=2.5-|0.1*t|
(^_^) def x(t)=r(t)*cos(t)
(^_^) def y(t)=0.2*t
(^_^) def z(t)=r(t)*sin(t)
(^_^) graph x:y:z[-8*PI, 8*PI]
```

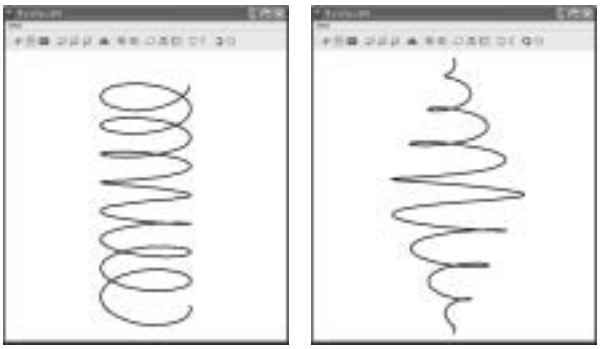


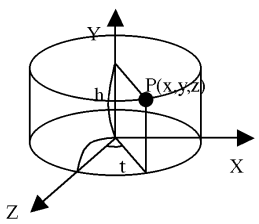
図7　三次元パラメトリック曲線

10.13　パラメトリック曲面

媒介変数を２つ使うと曲線ではなく，曲面になります。{x=x(u, v),y=y(u, v),z=z(u, v)}

円筒を描く場合を例にとって説明します。円筒上の点P(x, y, z)は半径r，中心角t，高さhを用いて次のように表されます。

$$\begin{cases} x=r\sin(t) \\ y=h \\ z=r\cos(t) \\ 0 \leqq t \leqq 2\pi \end{cases}$$



中心角tと高さhをパラメータとして式を与えます。半径rを一定にすれば円筒になり，中心角tにより半径を螺旋状に変化させればロール紙のようになります。以下に，実行例を示します。

```
(^_^) def r(t)=0.5*t/PI
```

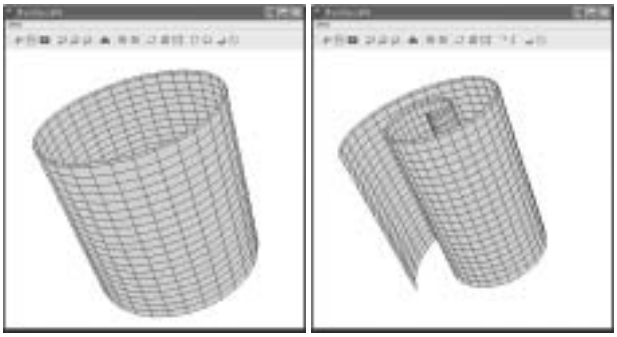


図8　パラメトリック曲面

```
(^_^) def x(t,h)=r(t)*sin(t)
(^_^) def y(t,h)=h
(^_^) def z(t,h)=r(t)*cos(t)
(^_^) graph x:y:z[0, 4*PI][-2, 2]
```

この機能を使えば，次のような複雑な形状も簡単に生成できます。

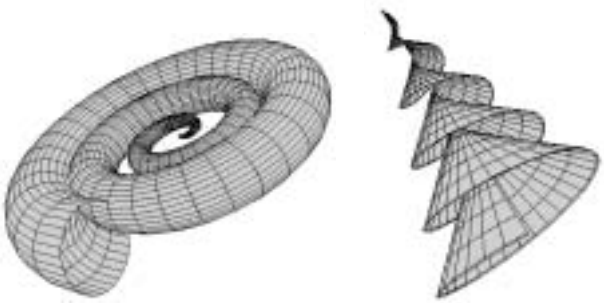


図9　図形生成例

11．おわりに

完成したシステムは，学生が卒業した３月以降にさらに３ヵ月間を費やしてドキュメントを整え，ベクターデザインのライブラリへ登録しました。こちらhttp://www.vector.co.jp/soft/other/java/se334311.htmlからダウンロードできます。数学教育等に役だててくださる方がいらっしゃれば，何よりです。大勢の目につく場所へ作品を置くというのは大変励みになったようで，２人とも資料の整備に最後まで力を振り絞ってくれました。大分時間はとられましたが，ここまでやって良かったと思っています。

最後に，夢中になって取り組んでくれた学生２人のお陰で，私自身もこの制作に熱中することができました。この場を借りて，新浜君と田中君の２人に心より感謝したいと思います。