東北女子大学・東北女子短期大学　紀要　No.53：67～74　2014

プログラミング入門用言語としての ActionScript の利用

友　田　志　郎*

Actionscript as a programming language for beginners.

Shirou TOMODA*

Key words : コンピュータプログラミング　Computer Programming
プログラミング言語　Programming Language
フラッシュ　Flash
アクションスクリプト　Actionscript

1．はじめに

ソフトウェア開発能力の裾野を広げる為、世界的にプログラミング教育の導入の動きが強まりつつある。わが国に於いても、新学習指導要領の中学校技術家庭科で「プログラムによる計測・制御」が必修化されるなどの動きが見られる。また、MIT メディアラボが開発し公開している「Scratch」(1) のようなブロックを組み上げる感覚で視覚的にプログラミングできるシステムを初等教育に取り入れる試みも盛んになってきている。こうしたシステムが Web ベースで動作可能となり、ネット環境があれば導入が容易であることなども追い風になっていると言えるであろう。

本論考ではアニメーション作成システムである FLASH 上で動作するプログラミング言語である Actionscript を取り上げ、プログラミング教育の入門用言語としての有用性について論じる。

2．プログラミングを学ぶことの重要性

プログラミングを学ぶことで養われる能力の一つとして、論理的な思考力が挙げられる。より具体的に言えば、段取りをつける能力と呼んでも良い。

プログラムとは詰まるところコンピュータに対しての「指示書」である。コンピュータは指示された事だけを実行し、指示されていないことは一切行わないという前提で動作する機械なので、この「指示書」にはこれこれの処理を行った後にこれこれの処理を行い、この場合にはこれこれの処理を行い、この条件を満たす間はこれこれの処理を繰り返し... と言った具合に、起こりうるあらゆる状況を想定して、事細かに段取りを記述しておく必要がある。

こうした記述は「アルゴリズム」と呼ばれるもので、コンピュータプログラミングの一つの大きな柱になっている。現実社会においても、「こうすればこうなる」「この場合にはこうする」「こうする為には、前もってこうしておく」といったアルゴリズム的思考は様々な局面で必要となる。コンピュータプログラミングを学ぶことはこうした思考法を養う為の良い訓練となると考えられる。

プログラミングによって養われる能力として、もう一つ、対象をモデル化する能力が挙げられる。

コンピュータプログラミングの名著の一つとして N.Wirth の「アルゴリズム＋データ構造＝プログラム」(2) という書籍が挙げられるように、データ構造はアルゴリズムと並ぶプログラミングの両輪とも呼べるものであり、実際のプログラミングに於いても最初にデータ構造の検討から始めることが一般的である。データ構造という言葉は、狭義には配列やハッシュ、リストやツリーなどのグラフ構造などの意味で使われる場合もあるが、広い意味ではデータ設計全般、つまり扱う対象を如何にしてコンピュータ上で表現するかという事を意味する。

＊東北女子大学

プログラムが対象とする現実世界の中の事物は数多くの属性を持った複雑なものである。それに対してコンピュータ自体は突き詰めて言えばメモリー中の０か１かのビット列、あるいはバイト列のデータに対しての演算機能しか持たない。従って、それら複雑な事物をコンピュータプログラムで扱う場合、処理対象の持つ数多くの属性の中からどの属性が必要なものか選び出す作業、更にそれらの属性をコンピュータで効率良く処理できる形で如何に表現するか検討する作業が必要となる。これがデータ設計作業である。対象の重要な属性を取り出して再構築するという点で、一般的な意味でのモデル化作業そのものと言える。

人間が物事を認識し理解する場合、対象の属性のすべてをそのまま扱う事は難しく、実際には対象の属性の一部を抽出して、それらを自分の持つ知識と経験に当てはめた上で自分の中で再構築する作業を行う。その際に重要な事は、物事の何が本質的に重要な点であるかを的確に見極め、それらを適切に再構築する能力である。コンピュータプログラミングに於けるモデル化・データ設計作業は、こうした能力を養成する訓練として極めて有用である。

これらの、論理的思考力や、対象をモデル化して把握する能力は、現実社会で発生する様々な問題解決にも必要な物であり、コンピュータプログラミングを学ぶことは広く問題解決能力を養う為の有効なトレーニング方法の一つと言えるだろう。

３．入門段階向きのプログラミング言語

1950年代のFORTRANやCOBOLの登場以来、コンピュータプログラムの作成は、プログラミング言語を用いて行うことが一般的である。かつて、大型計算機に端末を接続してタイムシェアリングシステムで皆が計算機を利用していた時代には、FORTRANの他、PASCALが教育用プログラミングとして大学などで良く使われていた。また、個人ユーザー向けに登場したパーソナルコンピュータ上ではBASICインタープリタが標準環境として使われていた事もあり、MS-BASIC（及びコンピュータメーカによる改造版）が入門用言語として位置づけられていた時期も存在した。その後、ワークステーションの普及とパーソナルコンピュータの性能向上に歩調を合わせるように、C言語が主流となった。更にWindowsなどのGUI環境が一般化し、オブジェクト指向プログラミングが重要な概念として登場し、インターネットが爆発的に普及するに至った現在では、C言語以外にもJAVAやVisualBasic、様々なスクリプト言語（JavaScipt やPython）など、様々なプログラミング言語が入門用として使用されるようになった。

C言語は1970年にUNIXオペレーティングシステム上の標準的なプログラミング言語としてD.Richieらによって開発され、コンピュータの機械語に近い処理から複雑で高度な処理まで幅広く対応できる事などから、様々なオペレーティングシステム上で使用できるようになった。プログラミング教育にも良く使用されるていて、東北女子大学に於いても、家政学科の情報教諭課程で筆者が担当している「プログラミング２・３」の授業ではC言語を最初に解説し、アルゴリズム、配列、関数（サブルーチン）、ポインタなどについての演習を行っている。

４．FLASHとActionscript

FLASHはWeb文書上のアニメーションなど、動きのあるコンテンツの作成の為のシステムとして、1990年代に登場し、2000年代にかけて 普及したソフトウェアで、現在はAdobe社が開発・販売を行っている。東北女子大学ではコンピュータ実習室のコンピュータシステムにFLASH開発システム（Flash CS5）が導入されており、FLASHを使用したアニメーション動画作成なども授業の演習題材に取り入れている。

ActionscriptはFLASHの為のプログラミング言語である。最新のバージョンはActionscript3.0であり、本格的なオブジェクト指向プログラミング言語として設計されている[(3)]。

東北女子大学で筆者が担当している授業では、

家政学科情報教諭課程の「コンピュータ３」で授業回数の半分ほどを Actionscript を用いたプログラミングの演習に当てている。また、児童学科の「コンピュータ概論」の授業でも回数は少ないがプログラミング演習の題材として用いている。

FLASH で扱うオブジェクト

FLASH では画面（ステージ）上で様々なオブジェクトを動かす事で、アニメーションを作成する事ができる。オブジェクトとしては、画像やテキスト、線画などを取り扱うことができる。また、テキストや線画については、諸属性を取り除いたシェイプに変換する事によって、形や色の変化を伴うアニメーションも作成可能である。

シンボルとインスタンス

FLASH でオブジェクトを動かす場合、そのオブジェクトの定義を定めた「シンボル」を作成し、そのシンボルを画面（ステージ）上で実体化させた「インスタンス」を操作することでアニメーションを作成する。シンボルにはムービークリップシンボル、グラフィックシンボル、ボタンシンボルが存在するが、これらの中で、ムービークリップシンボルはアニメーションをシンボルとするもので、シンボル自体を動きのあるものとして作成する事ができるなどの利点がある。複数のムービークリップシンボルをパーツとして組み合わせることで、上位のシンボルを作成する事も可能で、こうしたシンボルの階層化によって複雑な動きをするアニメーションを作成できる。

タイムライン上でのアニメーション作成

アニメーションの原理は、人間の視覚の残像現象を利用し、少しずつ変化した絵を高速で切り替える事で「動く絵」として知覚させるものである。それぞれの絵はフレームと呼ばれ、FLASH CS５ではデフォルトで 24 フレーム／秒のフレームレートで切り替える（フレームレートは自由に設定可能である）。

時間軸に沿ってフレームを並べたものをタイムラインと呼ぶ。インスタンスを画面上に配置したり属性値を変えたりする為には、キーフレームをタイムライン上に設定し、キーフレーム時点でのインスタンスの属性値（座標、縮尺、回転角、斜変形角）を設定する。キーフレームはインスタンスを画面上で動かす際の基準フレームとなるものである。

例えば、キーフレーム A を設定して座標 a にインスタンスを配置し、24 フレーム先にもう一つキーフレーム B を設定してインスタンスの場所を座標 b に変えた場合、アニメーションとして再生すると、キーフレーム A の再生時点でインスタンスが座標 a の位置に出現し、キーフレーム B の再生時点でその場所が座標 b に突然変化する事になる。これでは滑らかな動きにはならないので、インスタンスを座標 a から座標 b までスムーズに動かす為には、キーフレーム A とキーフレーム B の間にトウィーン（クラシックトウィーン）を作成する。トウィーンとはキーフレーム間のインスタンス属性値をコンピュータが計算しながら滑らかに変化させるものである。これにより、滑らかな動きのアニメーションを簡単に作成する事ができる。

実際にタイムラインを用いて FLASH アニメーションを作成してみると、この作業がプログラミングに通ずるものであることに気づく。ただし、アルゴリズムとしては、条件処理や繰り返しループの無い逐次実行のみと言うことになる。

５．プログラミング言語としての Actionscript

Actionscript はオブジェクト指向プログラミング言語になっており、タイムラインで作成した FLASH シンボルをクラスとして取り扱う事もできる。マウスやキーボードなどをイベントとして扱う事もでき、インタラクティブな動画やアプリケーションの作成もできる。作成したプログラムは、FlashPlayer 上で実行される。

オブジェクト指向

オブジェクト指向プログラミングは 1980 年代

に提唱されたパラダイムで、それを体現した最初のプログラミング言語（或いはコンピュータシステム）としてはSmalltalkが挙げられる。その後、C言語を拡張したC++言語が登場し、1990年代にはJAVAなど多くのオブジェクト指向言語が登場するなど、プログラミングの主流を占める考え方となった。

オブジェクト指向プログラミングでは、プログラムが処理する対象を幾つかの属性が集まった「オブジェクト」として捉え、更にオブジェクトに対しての操作・処理もメソッドとしてオブジェクト自身の属性に含む。

そのオブジェクトがどのようなものか定義したものをクラスと呼び、クラスを実体化したものをインスタンスと呼ぶ。クラスは、より包括的・抽象的な上位クラス（或いは親クラス）、個別的・具体的な下位クラス（子クラス・派生クラス）といった階層性を持っており、上位クラスのもつ属性はそこから派生した下位クラスに継承される。

実行制御文

アルゴリズムを記述する為の条件分岐やループなどの実行制御文は、C言語の構文・文法をほぼ踏襲している。また、複数の文を一つにまとめるためのブロックを｛｝を用いて表現する点もC言語と同様である。

JAVAなど、比較的後発のプログラミング言語の場合、実行制御文をC言語と同様にしている言語は珍しくない。Actionscriptもこの例に習ったものと言える。手続き型のプログラミング言語の場合、アルゴリズム記述の為に必要な要素には本質的な違いが無い。既存の言語でプログラミング経験のあるプログラマーが新たな言語を習得する場合、既知の実行制御文がそのまま使える方が新しい言語に馴染みやすく、メリットは大きいと言える。

変数等の宣言・データ型

変数宣言文や関数宣言の構文はPASCALのそれに近い。ただ、PASCALと違い、サブルーチンと関数で異なった構文にはなっていず、返値のデータ型によって区別される。

データ型はint（整数型）、Number（実数型）、Boolean（論理型）、String（文字列型）の基本スカラー型が用意されている。String型が存在する等、データ型についてもPASCALを踏襲していると言える。

基本スカラー型以外の変数はすべて、しかるべき「クラス」のインスタンスである。配列ですら「配列クラス」のインスタンスとして実現されている。配列の要素についてのデータ型宣言も無く、任意のデータ型の要素をセットできる。C言語やPASCALなどのデータ型宣言の厳密なプログラミング言語とは、このあたりの感覚はかなり異なっている。

6．入門用プログラミング言語としてのActionscriptの利点

プログラミング言語としてのActionscriptは極めて本格的なものであり、言語仕様の面でプログラミング教育の導入言語として利用することには特に不足はないと言える。更に、Actionscriptを導入言語とする事には、以下のような利点がある。

(1) 視覚的でわかりやすい出力結果を得られる

C言語の場合、プログラムの出力の為に用意されている標準ライブラリ関数はprintf()など、基本的にはテキストを表示する機能しか持たない。処理結果のメッセージやデータが端末画面に文字として表示されるだけである。これではプログラミングを学ぼうとする初心者にとっては少々ものたりないであろう。それではと言うことでグラフィカルな出力を得ようとした場合、UNIX上であればX11や様々なGUIライブラリ、Windows上であればMFCなどのクラスライブラリを使用する必要があるが、これらを使用する為のコーディングの量や説明事項の多さを考慮すると、入門段階でそれを教えるのは困難である。

一方、Actionscriptの場合は元来FLASHの為

のプログラミング言語なので、FLASH で作成したヴィジュアルなオブジェクトを扱うプログラムが容易に作成できる。単に FLASH シンボルのインスタンスを画面上に配置するだけであっても、テキスト文字が端末画面に表示されるだけのものより単純に「楽しい」ものである。また、ペンプロッタタイプの線画グラフィックの為の描画メソッドも用意されており、画面上にプログラムで図形等を描かせるプログラムを題材として扱う事もできる。線画グラフィックは、座標についての概念、あるいは多少高度なものであっても三角関数が理解できていれば十分対応することができ、どのようなプログラムを書けばどのような出力結果になるのかというレスポンスが分かり易く、美しい出力結果が得られるような題材も設定しやすいなど、プログラミング入門、特にアルゴリズム的な考え方を学ぶには良いアプローチだと言える。

1980 年代に各メーカーのパーソナルコンピュータ上で提供されていた BASIC 言語では、線画や塗りつぶしなどのグラフィック命令がサポートされていた。当時の BASIC 言語は実行速度が遅い上に、関数やサブルーチンの独立性が無く、複雑なデータ構造も構築できないなど、大規模なプログラムを作成するには不向きなシステムだったが、グラフィック命令が手軽に利用できる点は「アルゴリズム」の演習手段としては有効だったのではないだろうか。

(2) 動きのあるプログラムを容易に作れる

画面上で何らかのオブジェクトをアニメーションとして動かす場合、単純に考えれば、オブジェクトを画面上から消去し、その後、座標を少し変化させて再描画する という処理を繰り返すことで実現できる。単純な背景でオブジェクト同士の重なりを考えなくとも良い場合には、これは比較的簡単で、実際に初期のコンピュータゲームなどではこうした方法が使われていた。

しかし、オブジェクト同士の重なりを考えた場合には事情が異なる。前面に配置されたオブジェクトによって背後のオブジェクトが隠されるため、個々のオブジェクトの消去／再描画ではなく、各オブジェクトの重なりを考慮した上で、隠されずに見えている部分のみを描画する形で画面全体を再構築できるような、ハードウェア／ソフトウェアによる描画エンジンが必要となる。

Actionscript の場合、実行環境自体がそうした描画エンジンを備えているため、動きを実現する上で、オブジェクトの消去／再描画を意識する必要がない。画面上のインスタンスが基本プロパティとして持つ xy の座標値を変えてやれば、画面上での表示位置も新しい座標に変わる。その際に、他のオブジェクトとの関係（背面にあるオブジェクトを隠す、前面にあるオブジェクトに隠される）について考慮する必要はない。このように、画面に表示されているオブジェクトの属性値の変更結果がそのまま視覚的に反映されるという点で、プログラミングの入門時の題材としては大変わかりやすいと言える。

Actionscript で実際にアニメーションを実現する場合はフレーム再生イベントや Tween クラスを使用する。これらについては後述する。

(3) インタラクティブなプログラムを容易に作成できる

プログラムに対してのユーザーの操作手段としては、マウスやキーボードなどが挙げられる。Actionscript ではこれらの機器操作やタイマー、フレーム再生などは全てイベントとして扱われ、そうしたイベントが発生した時に対応する動作をイベントハンドラとして記述するという形で、ユーザーのインタラクティブな操作に対応するプログラムを作成できる。

イベントの通知要求は画面上の各インスタンスについて設定することができ、イベントの種類もユーザーの操作以外にタイマーイベントやフレーム再生イベントなど数多く存在するが、イベント処理の手順自体は共通している。一つのやり方を理解すれば事足りるという点は、プログラミングに慣れていない段階では有利である。

```
// リスト１　線画グラフィック
var shape1:Shape;       // Shape クラスのインスタンス変数宣言
var i:int;
var px,py:Number;

shape1 = new Shape(); // Shape クラスのインスタンス作成
px = 40; py = 40;
shape1.x = 0; shape1.y = 0;    // 原点設定
addChild(shape1);       // shape1 を画面に登場させる
shape1.graphics.lineStyle(1,0xFFFF00);// 線の太さと色を設定
for (i=0; i<=27; i++) {         // 変数 i の値 0 から 27 まで増やしながら繰り返し
        shape1.graphics.moveTo(px, py+i*20);   // ペンを移動させる
        shape1.graphics.lineTo(px+i*20, py+540);// 直線を描画
}
```

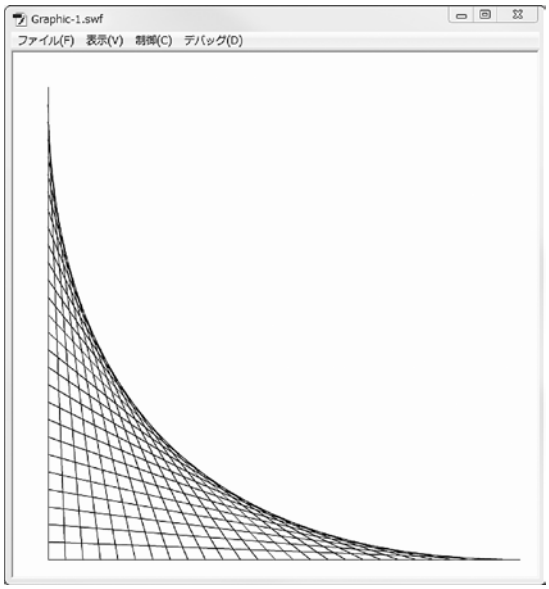

図２　リスト１の線画グラフィック出力

７．Actionscript を使用した演習題材

(1) 線画グラフィックを用いたプログラム

レーザープリンタが普及する以前、コンピュータにグラフを描画させる機器としてはペンプロッタが主に使用されていた。ペンプロッタの操作は基本的に、①ペンの種類を選択、②目的の座標までペンを上げたまま移動、③目的の座標までペンを下げて線を描きながら移動、の３つである。操作自体が単純な為、目的とする図形を描くにはどうすれば良いかという点が重要となる。

Actionscript では Shape クラスを用いることで、プロッタ式の描画プログラムを容易に記述できる。ペンの太さや色は .graphics.lineStyle ()、ペンアップでの移動は .graphics.moveTo ()、線を描きながらの移動は .graphics.lineTo () の各メソッドをインスタンスに対して使用する。実際に授業で用いた題材をリスト１に、その実行結果を図２に示す。

(2) イベント処理と「動き」のあるプログラム

画面上のインスタンスをアニメーションとして動かす方法の一つにフレーム再生イベントを使用する方法が上げられる。フレーム再生イベントは、新しいフレームが再生される度に発生するイベントで、FLASH ならではものだとも言える。フレーム再生イベントの発生する度にインスタンスの座標値を少しずつ変化させれば、アニメーションとして画面上で動く事になる。

リスト２はインスタンスが画面上を左右に動きながら、画面上でマウスがクリックされた場合には瞬時にその位置に移動するというプログラムになっている。リスト中の Zou クラスは、図１で示した象のシンボルで、そのインスタンスを boo という変数に格納している。画面のサイズは 550 × 400 ピクセルとし、象が左右の端まで移動したら、向きを反転させる。

インスタンスに対してイベントハンドラを設定するには、イベントハンドラとなる関数を作成した上で、.addEventListener () メソッドを用いてイベントの種類とそれに対するハンドラを指定する。画面上のマウスクリックに対して waapZou () 関数をハンドラとして呼び出し、象のインスタンスに対するフレーム再生イベントに対して moveZou () を呼び出す。イベントハンドラを設定した後は、プログラムはイベント待ちの状態で待機する。

```
// リスト２　　画面上を歩く象
var boo:Zou = new Zou();        //  Zou クラスのインスタンス作成
var dx:int;                     //  X 方向の移動量変数

// ステージにマウスクリックイベントのハンドラ関数 waapZou() を設定
stage.addEventListener(MouseEvent.CLICK, waapZou);
dx = -2;            // X方向の移動量設定
boo.scaleX = 0.25; boo.scaleY = 0.25; // インスタンスの縮尺
boo.x = 400; boo.y = 300;     // インスタンスの初期座標

// インスタンス boo にフレームイベントのハンドラとして moveZou() を設定
boo.addEventListener(Event.ENTER_FRAME, moveZou);
addChild(boo);        // インスタンス boo を画面に表示

//  フレーム再生イベントのハンドラ
function moveZou( eventObj:Event):void
{
        boo.x = boo.x + dx;    // X座標を dx だけ変化させる
        if ((boo.x > 500)||(boo.x < 50)) { // 画面の端まで移動した場合
                dx = -dx;                       // 移動方向を逆にする
                boo.scaleX = -boo.scaleX;       // 左右を反転
        }
}

// マウスクリックに対するイベントハンドラ
function waapZou(eventObj:MouseEvent):void
{
        boo.x = eventObj.stageX;        // マウスクリック位置を boo の
        boo.y = eventObj.stageY;        //  座標として設定する
        if (boo.x < 50)  boo.x = 50; // 画面の端でクリックした場合の処理
        if (boo.x > 500) boo.x = 500;
}
```

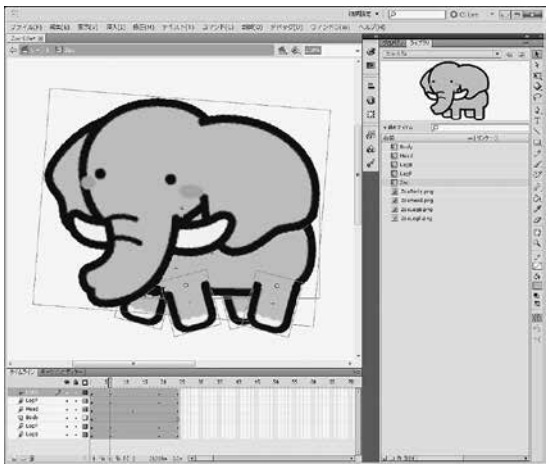

図１　タイムライン上での Flash シンボルの作成例

演習授業で題材としている象のシンボル。学生が親しみやすいようなフリー素材のイラストを使用している。四肢を動かして頭を振りながら「歩く」

８．入門用言語としての Actionscript の問題点

Tween クラスとその問題点

一般に、FLASH アニメーションをタイムライン上で作成する際にはトウィーンを用いる。Actionscript プログラムに於いても Tween クラスを用いることで、トウィーンを使うことができ、インスタンスの座標等の値を１フレーム毎に設定しなくても、「○○から××まで△△秒かけて変化させる」といった指定ができる。Tween クラスを用いてインスタンス boo を座標（zx, zy）の位置まで３秒で動かす為には、例えばリスト３のように記述する。

Tween クラスを用いてインスタンスの動きを制御する場合、一つ問題となるのは作成した Tween インスタンスが基本的に同時並行的に動作する点である。リスト３の例では twZou_x と twZou_y のトウィーンは並行的に実行され、従ってインスタンスの x 座標と y 座標は同時に変化する。仮に x 座標を３秒かけて変化させた後で y 座標を変化するようにしたい場合、すなわち Tween を逐次的に実行したい場合は、先に実行する Tween の完了イベントに対するハンドラを設定し、その中で後に実行する Tween を記述する必要がある。これでは記述に手間がかかるだけでなく、処理の流れが掴みにくくなってしまう。

```
// リスト３　　Tween による座標変化の例
// インスタンス boo の座標を (zx, zy) に 3 秒で変化させる
var twZou_x, twZou_y:Tween;    // Tween クラスのインスタンス変数
twZou_x = new Tween(boo,"x",None.easeInOut,boo.x,zx,3,true);
twZou_y = new Tween(boo,"y",None.easeInOut,boo.y,zy,3,true);
```

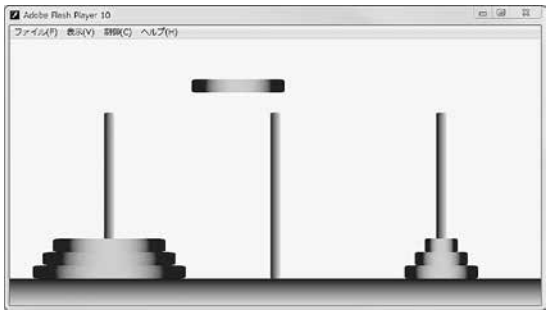

図３　動作中の「ハノイの塔」
１枚の円盤を「上にあげる」「横に移動する」「下に降ろす」動作を Tween クラスを用いて実装している。

こうした Tween の逐次処理や同期処理に対応するため、Actionscript を利用するユーザーが独自に開発・公開している Tweener や Between、Tween24 などの Tween 制御ライブラリがよく利用されている。しかし、これらはあくまで公式のライブラリではないという点には留意する必要がある。

「ハノイの塔」の演習題材

「ハノイの塔」の問題は、再帰的処理による問題解決の例としてしばしば取り上げられる。３本の柱の１本に大きさの違う n 枚の円盤が刺さっていて、

①小さな円盤の上には大きな円盤を乗せられない。

②円盤は１枚ずつしか動かせない。

という制約の下で、n 枚の円盤すべてを別の柱に移動させるという問題である。これは、「n 枚の円盤の移動」という問題を「n-1 枚の円盤の移動」という部分問題に置き換えて、関数を再帰的に呼び出すことで、極めてスマートに記述できる。

再帰的処理はプログラミングの重要な概念の一つであるし、Actionscript では関数を再帰的に呼び出すこともできるため、「ハノイの塔」の円盤の動きをアニメーションとして表現するプログラムを演習の題材として準備した際に、最もやっかいな問題となったのが Tween の制御の問題であった。結局のところ、円盤の動きをすべて配列に登録した後に、登録した通りに Tween で動かすという方法で、それらしいプログラムにしたが、あまり本質的な解決策とは言えないであろう。

９．まとめ

プログラミングは無から有を生み出す、極めて創造的な作業である。また、プログラミングが必要な職業に就かない者にとっても問題解決のトレーニングとして有用である事は前述の通りである。しかしながら、最低限の文法・構文やライブラリの使用法を把握しておく必要があり、関連知識を全く持たない者が最初に学ぼうとする時にはそれなりの障壁の高さがあると言わざるを得ない。

Actionscript は比較的少数の構文やライブラリ関数だけで、視覚的に判りやすく、ある程度まとまった動きをするプログラムを作成することが可能で、入門用のプログラミング言語としては有用である。東北女子大学のコンピュータ実習室には幸いなことに FLASH 開発システムが導入されていることもあり、今後もプログラミング教育の手段として活用して行きたい。

参考文献

１）Scratch – Imagine, Program, Share – MIT, http://scratch.mit.edu/

２）Niklaus Wirth,『アルゴリズム＋データ構造＝プログラム』訳：片山卓也 , (1975)

３）Adobe Flash Platform 用 ActionScript 3.0 リファレンスガイド, http://help.adobe.com/ja_JP/FlashPlatform/reference/actionscript/ 3 /