プログラム教育の前に教員が理解するための教材の一例

An Example of Material for Teachers in Charge of Programing Education

宇　多　賢治郎[1]
UDA Kenjiro

要約：本稿は、小学校におけるプログラム教育のカリキュラム化に対応できるよう、まず教員がプログラムの作成作業を理解するための説明をまとめたものである。説明には、小学校算数で用いる多数の図形を作成するプログラムを、例として用いた。また、作成方法の概要を理解してもらうため、まず図を紙の上に手で描く方法を確認した。次に、その作業内容を基に、人に代わりに作業をしてもらうための指示をまとめ、その内容をプログラムに「通訳」するという作業の流れを示した。また、プログラムを作成する意義が伝わるよう、まず手作業や汎用のソフトでは作成が難しい図形を描いてもらうプログラムを説明した。最後に、このプログラムに少し加筆するだけで、大量の図形を一度に描いてもらえるようにできることを示した。

キーワード：Excel VBA、図形描画、ペントミノ、タングラム、フリーフォーム

Ⅰ　はじめに

本稿では、小学校におけるプログラム教育がカリキュラム化され、教育課程に位置づけられたことを踏まえ、教員の理解に役立つよう小学校算数で扱う図形描画を例に、プログラムを組むまでの過程を説明する[2]。

筆者は、自身の研究のため作成したプログラムを、宇多（2019）、宇多（2011）、宇多（2003）などの論文で解説した経験を持つ。その経験に基づき、本稿ではまず図形を手描きする方法を説明し、次にそれを代わりにしてもらうための指示内容にした後、プログラムに「通訳する」という手順を示していく。

また、教員がプログラムを使用する頻度は多くないため、それなら手やマウスを使って描いてしまえばよいと、考えるであろう。そこで、今回はまずパソコン上で、マウスを使って描くのが難しい図形を例にする。小学校算数の授業では、図形の様々な性質を説明するため、条件のゆるい基本図形を描くことを、学年をわたって行うことになる。このような図形は、マウスでは描きづらいため、プログラムを組んで、代わりに描いてもらうのである。次に、このプログラムに最低限の加筆をし、複数の図形を一度に描いてくれるように発展させる方法を示す。

なお、本稿で説明するプログラムを組み込んだExcelファイルは、以下のWebページで公開しているので、活用いただきたい。

http://www.ccn.yamanashi.ac.jp/~kuda/

---

[1] 生活社会教育講座

[2] 本稿の執筆の際、本学部清水宏幸准教授には、数学教育を専門とされる立場から貴重な意見をいただくなど、執筆の際は大変お世話になった。ここに記し、感謝申しあげる。

Ⅱ　プログラム以前の作業内容の確認

１　作業内容と工程の確認

小学校３年生以降の算数では三角定規、分度器、コンパスを使い、方眼紙の上に図形を描くということを頻繁に行う。その目的は、図形の概念の理解、求積、合同、拡大図・縮図、線対称・点対称など多岐にわたる。授業中に、これらの図形を教壇で示そうとすれば、その都度、模造紙や黒板の上で、巨大な三角定規やコンパスを使って描くことになる。

この作業を、代わりにパソコンにしてもらう場合、作業の効率性や快適さは、用いるソフトウェアによって大きく異なるものになる。つまり、ソフトが描ける図形の多様さ、描きやすさ、またサイズ変更、移動、反転、回転などの編集のしやすさなどに依存することになる。例えば、Microsoft Officeのような職場で支給される可能性が高い汎用のソフトを使って、小学校の授業で扱う多様な図形を描くのは簡単ではない。小学校の算数の場合、図形の特徴や求積、合同、拡大図・縮図、線対称・点対称などを教えるため、条件のゆるい多様な基本図形を必要とするからである。

これらのことを踏まえ、まず作業内容を把握し、それを作業別に分け、行う手順を定めるといった、準備段階から説明する。

まず、今回行う作業の前提条件や目的を確認する。

・職場で多くの教員が使っていると思われる、Microsoft Officeを活用する
・Microsoft Officeに組み込まれたVisual Basic for Application（以下、VBA）を利用する
・プログラムを組む前提となる考え方から説明することで、汎用性を確保する
・プログラムの基本文は、わざわざ覚える必要がないことを示す
・描きづらい図形を描く作業を、大量にこなしてもらえることを示す

具体的には、VBAを使ってExcelファイルの座標データを引用し、それを描かせるプログラムを作成する。なお、描いた図形は授業でも使えるよう、PowerPointやWordで利用できるようにする。

これらを踏まえ、今回は作業を三段階に分けて行う。

作業１．方眼紙に手で描き、作業内容と手順を確認する
作業２．代わりに描いてもらうための指示を考え、まとめる
作業３．Excelに頼むためのプログラムに「通訳」し、描いてもらえるようにする

つまり、まず何をするかを「紙の上に描く」ことで確認し、次に「相手」に合わせて説明するのである。この場合の「相手」はExcelであるから、Excelが理解できるようにプログラムの形式で説明することになる。

２　作業１、２　方眼紙に描いて、作業内容を確認する　（相対的な指示）

まず、Excelのソフトでは書きづらい、左右対称ではない台形を描く方法を説明する。そのため、まず「作業１」では方眼紙を用意し、定規などを使って描く台形を確認する。

図１は、これからプログラムに描かせる台形である。

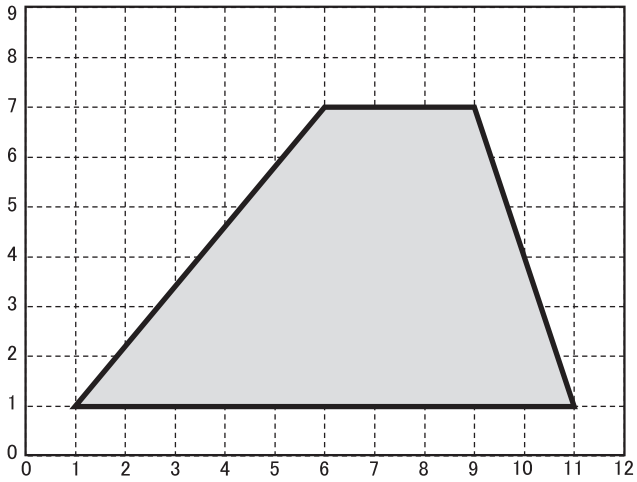

図１　方眼紙を使って描く図

ExcelなどOfficeの描画機能には、このような左右対称ではない台形は用意されていない。線だけで描く方法もあるが、図１のように台形の内側に色をつけるには、一つの「図形」として描く必要がある。

そのため、今回は 「フリーフォーム」という図形を用いる。この「フリーフォーム」では、マウスで図形の角の位置を、マウスを使って順に指定するという、細かい操作が必要になる。そこで座標を指定し、プログラムに代わりに描いてもらうのである。

このプログラムに代わりに描いてもらうための指示の内容は、授業で児童に説明するものに似ている。そのことを確認するため、この台形の定義を習う、小学校３年生の児童に対して指示するための説明をまとめてみる。

指示０：始点の座標は、方眼紙の「縦１、横１」である
指示１：そこから「上に６、右に５」の座標まで、直線を引く
指示２：そこから「右に３」の座標まで、直線を引く
指示３：そこから「下に６、右に２」の座標まで、直線を引く
指示４：そこから「左に10」の座標まで、直線を引く

この指示を方眼紙上にまとめたものが、図２である。

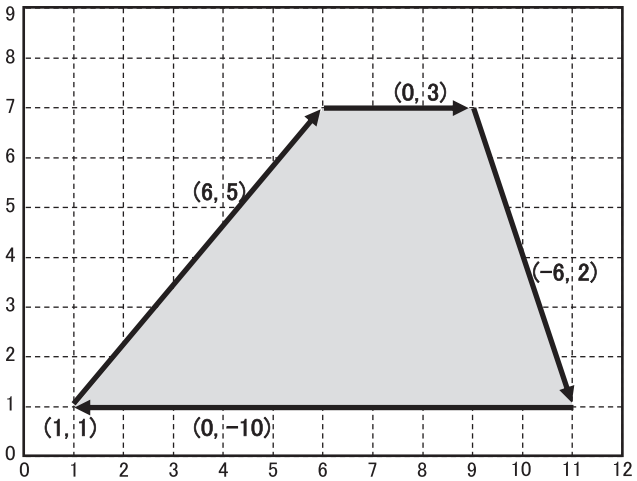

図２　順番に直線を描くように指示した場合（相対的な指示）

この指示の場合、始点の位置を除き、「相対的」なものになっている。つまり、前の作業を前提としている。そのため、情報が誤って伝われば、以降の作業全てに支障をきたすことになる。また、始点と終点がずれるため、図形が閉じなくなってしまうことになる。

3　作業１、２の指示を言い換えてみる　（絶対的な指示）

これに対し、次のように指示する方法もある。

指示０：始点の座標は、方眼紙の「縦１、横１」である
指示１：そこから座標「縦７、横６」まで、直線を引く
指示２：そこから座標「縦７、横９」まで、直線を引く
指示３：そこから座標「縦１、横11」まで、直線を引く
指示４：そこから始点、つまり座標「縦１、横１」まで、直線を引く

この指示を方眼紙上にまとめたものが、図３である。

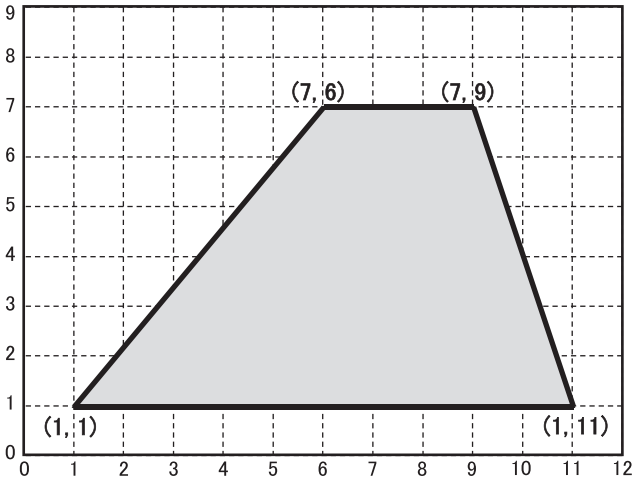

図３　座標で指示した場合（絶対的な指示）

この場合、指示が全て「絶対的」なものになっているため、指示が間違って伝わった箇所があるとしても、その点の位置がおかしいだけで済む。

4　作業３前半　プログラムで指示するための準備

次に、これまでの内容をExcelに「指示する」ために、プログラムに変換する方法を説明する。描画の指示方法を「相対」、「絶対」の二種類で説明してきたのは、プログラムが片方の指示方法に対応していない場合は、その指示をもう一方の方法に変換するためである。そこで、まず２章２節で説明した「相対的な指示」を、２章３節で説明した「絶対的な指示」に変換する方法を説明する。

まず２章２節で説明した、「相対的な指示」を一覧にまとめたものが、表１である。

表１　相対的な指示

	数値		記号（変数）	
	縦	横	縦	横
指示０：始点（方眼紙の左下）	1	1	y(0)	x(0)
指示１：１本目の移動	6	5	y(1)	x(1)
指示２：２本目の移動	0	3	y(2)	x(2)
指示３：３本目の移動	-6	2	y(3)	x(3)
指示４：４本目の移動	0	-10	y(4)	x(4)

表１の指示では、方向を縦横の別に、プラスとマイナスで示すように変更してある。今回は算数のグラフ（第一象限）や今回の方眼紙に基づき、横をｘとし、右方向をプラス、左方向をマイナスとしている。また、縦はｙとし、上をプラス方向、下をマイナス方向としている。また表の右側には指示の内容を英字で、何番目の指示かをカッコ内の数字で示している。

これらの指示を、２章３節で説明した「絶対的な指示」に変更する必要がある。
そこで、表１と比較できるよう、２章３節の「絶対的な指示」を一覧にまとめたものが、表２である。

表２　絶対的な指示

	数値		記号（変数）	
	縦	横	縦	横
指示０：始点（方眼紙の左下）	1	1	ya(0)	xa(0)
指示１：１本目の終わり	7	6	ya(1)	xa(1)
指示２：２本目の終わり	7	9	ya(2)	xa(2)
指示３：３本目の終わり	1	11	ya(3)	xa(3)
指示４：４本目の終わり	1	1	ya(4)	xa(4)

表２の指示を示す変数にはxa、yaのように、「a」を付けてある。これは絶対（absolute）的な値に変換したことを示している。

表１と表２を比較すると、「表２の一つ上の値に、表１の同じ位置にある値を足す」と、表２の値になることが分かる。

Ⅲ　プログラムの組み方

１　Excelでプログラムを動かすための準備

Excelファイルにプログラムを組み込むには、ファイルを通常のExcel形式（拡張子　.xlsx ）ではなく、「マクロ有効ブック」（拡張子　.xlsm ）にする必要がある。

そのため、図４のように、ファイルを保存する際に「ファイルの種類」で、「Excelマクロ有効ブック（*.xlsm）」を選択する必要がある。そのためには、Excel操作画面左上にある「ファイル」を押すと左側に現れる「名前を付けて保存」を選択し、「保存」の設定画面を表示する必要がある。

また、通常のExcelの設定では、プログラムを動かせないようになっているため、事前に設定を変更しておく必要がある。

手順１：Excelの操作画面上部、左側にある「ファイル」の上で「選択」する[3]
手順２：現れたメニューの下中央にある「オプション」を、「選択」する
手順３：現れたメニューにある「セキュリティ センター」を、「選択」する
手順４：現れた項目一覧から「セキュリティ センターの設定」を、「選択」する
手順５：「マクロの設定」を、「選択」する
手順６：現れた項目から「警告を表示してすべてのマクロを無効にする」を、「選択」する
設定前に「マクロ有効ブック」（拡張子 .xlsm）開いている場合は、いったん閉じ、再度開く必要がある。

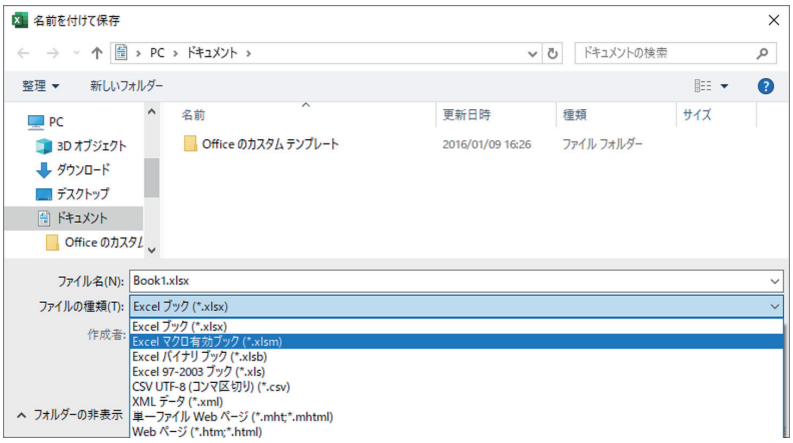

図４　「名前を付けて保存」の設定画面

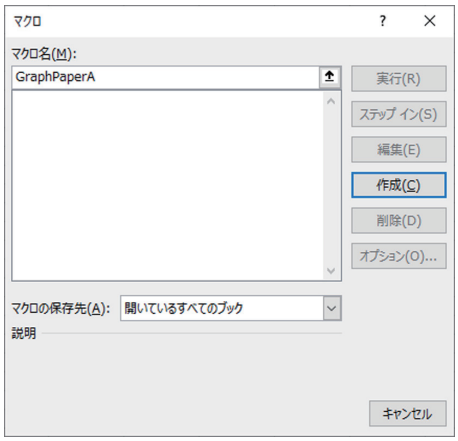

図５　「マクロ」の設定画面

次に、プログラムを入力する画面を表示してから、入力したプログラムを動かすまでを説明する。以下は、本節で説明するプログラム「GraphPaperA」を、一から作成する場合を例に説明する。

手順１：Excel操作画面上の「表示」を選択すると現れる、「マクロ」を「選択」する
これにより、図５の「マクロ」の設定画面が現れる。
手順２：「マクロ」の設定画面の上部にある、「マクロ名」の欄に「GraphPaperA」と入力する
手順３：「マクロ」の設定画面の右側にある「作成」を「選択」する
これにより、画面左上に「Microsoft Visual Basic for Application」と表示されている、プログラムを入力する画面が現れる。
手順４：現れた画面に、プログラムを入力する
なお、ファイルを閉じた後に、またプログラムを編集する際は、「マクロ名」に「GraphPaperA」が表示される。これを「選択」し、右側の「編集」を押すと、プログラムを作成するための画面が現れる。
ここにプログラムを入力するが、その説明はここでは省略する。
手順５：プログラムができあがったら、図５右上の「実行」を「選択」する
プログラム入力画面の　「Sub/ユーザーフォームの実行（F5）」を「選択」するか、キーボードの「F5」キーを押す方法もある。
プログラムが実行され、台形が描かれる。

[3] 図形やアイコン（ボタン）などに、マウスのカーソル（矢印）を重ね、マウスの左ボタンを押すことを、「選択」する、という表現を使って省略する。

2　プログラムで描く方法の確認

次に、これまでの説明を、プログラムの形式で表現する方法を説明する。

図６は、台形などの多角形を描くためのプログラム「GraphPaperA」を組み込んだExcelファイル「GraphPaperA.xlsx」のシート「data」である[4]。

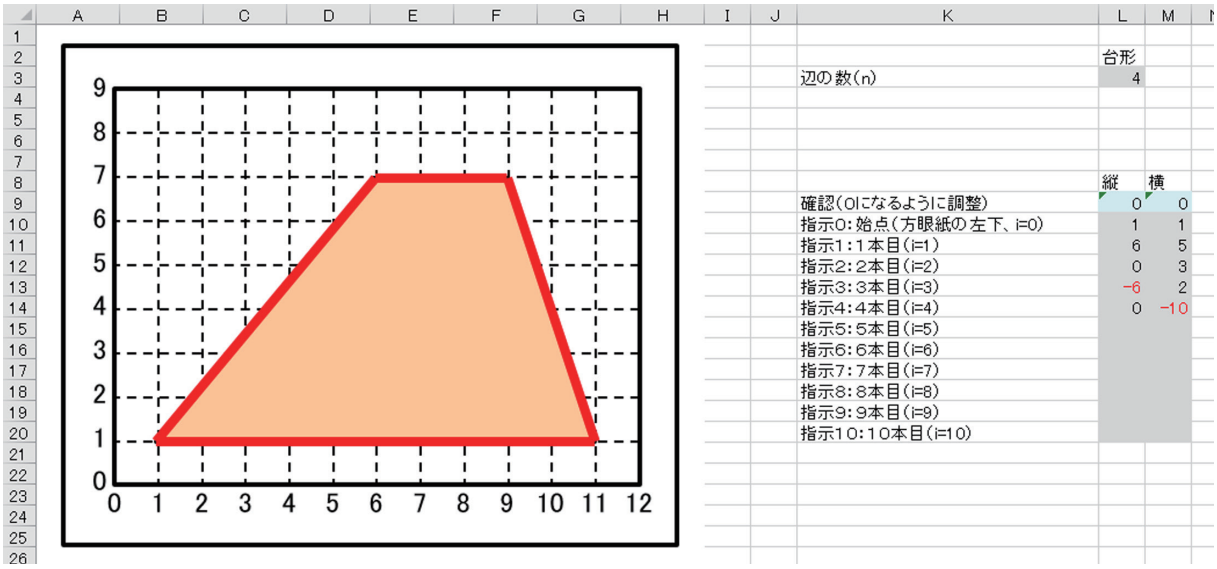

図６　GraphPaperA の作業画面（台形を描かせた状態）

図６のように、Excelのシート上に表１のデータを入力し、それをプログラムに読み込ませ、同じシートに図形を描かせる。

図６左の方眼紙を模したものは、こちらで用意したものである。この方眼紙は１マス１cmで作成したものである[5]。

ここでは、二種類の台形の描き方を説明する。

描き方１：図形を線に分解できるように、線４本の別に描く
描き方２：図形内に色が塗れるように、「フリーフォーム」で描く

単に図形を描くだけなら、「フリーフォーム」だけで済む。しかし、授業では完成形をただ見せるだけではなく、作業工程を示すこともある。また、一組の向かい合う二本の線が平行という台形の性質を示す必要もある。それらの説明がしやすくなるよう、線別に描く方法も説明する。

なお、この方法では、描画に必要なデータを方眼紙の右側に配置している。図６の右側に配置したセルL10、M10には始点、縦11行以降には表１の相対的な位置を示している。また、縦９行目の「確認」で図形が閉じているか、つまり始点と終点が同じであるかを確認できるようにしてある。今回は、線の動きを示す11行目以降の値を全て足して結果が「０」になれば、図形が閉じていることが分かるようにしてある。

なお、図６には示していないが、配布しているプログラムファイルでは、横L列、M列の下の方に「相対的な指示」を「絶対的な指示」を変換する式を、また図が用意した方眼紙に収まるかを確認できるよう、図形の最端の座標を示すようにしてある。これらは、プログラムが設計通りに作っても、

[4] これらのプログラムは、理解しやすさを優先して作成した。そのため、プログラムとしては効率悪く、より簡潔にすることができる、という指摘が可能なものになっている。
[5] 縦９cm、横12cmの表を、PowerPointのスライド内に収めるなら、１マス２cmで描くとよい。

想定していた通りに動いてくれることはまずなく、結局は手探りで調整しなければならなくなることによる。この調整作業を円滑に行えるよう、点検作業を簡単にできるように、問題の箇所を見つけやすいようにしておくのである。

3　図形描画に必要な指示のプログラム化

このシートを用い、次に台形を描くための指示を、プログラムに書き換える方法を説明する。

まず、本稿で示すプログラムは、暗記する必要がないことを確認しておく。例えば、筆者はこのようなプログラムを、パソコンや技術書のない所で白紙の上に書くことはできない。今まで作成した、入手したプログラムを再利用する、参考文献にあるような技術書などで確認する、またExcelの操作を記録させ、加工して利用するなどの方法を組み合わせて、何とか作成しているのである。

専門家の中には、プログラムを紙の上にいきなり書ける人もいるようであるが、筆者はプログラムを生業にしているわけではないので、これで済ませている。つまり、使用頻度の少ないことを記憶しておくことは困難なため、記録に頼っているのである。このような場合は、プログラムの技能はさほど重要ではない。それよりは、２章で説明したような、行うことをまとめておく論理性と、それを実行するために必要なExcelの操作方法の方が重要である。

そこで次に、プログラムをExcelに書いてもらう方法を説明する。今回は方眼紙を載せたファイルを用意した。このファイル上で、以下の手順で、マウスを使って台形を直線４本と「フリーフォーム」で描く作業を行う。

手順１：Webページで公開している「GraphPaper.xlsm」を入手し、開く
　　画面上部に「セキュリティの警告」と書かれた黄色い帯が現れる[6]。
手順２：「コンテンツの有効化」を「選択」する
手順３：Excelの操作画面上部にある、「表示」を「選択」する
手順４：現れたメニューの右側、「マクロ」の下にある、「∨」を「選択」する
手順５：現れたメニューの、「マクロの記録」を「選択」する
手順６：方眼紙の上に台形を直線四本で描き、次に同じ台形を「フリーフォーム」で描く
手順７：手順３～４を行い、現れた「記録終了」を「選択」する
手順８：Excel操作画面上の「表示」を選択すると現れる、「マクロ」を「選択」する

これで、マウスで行った台形を描く作業から、プログラムで指示する場合の文面を入手できる。以下は、それを取り出したものである。

まず、直線四本を合わせて、台形を描かせるプログラムは、次のようになる。

```
ActiveSheet.Shapes.AddConnector(msoConnectorStraight, 80, 270, 220, 100).Select
ActiveSheet.Shapes.AddConnector(msoConnectorStraight, 220, 100, 305, 100).Select
ActiveSheet.Shapes.AddConnector(msoConnectorStraight, 305, 100, 360, 270).Select
ActiveSheet.Shapes.AddConnector(msoConnectorStraight, 360, 270, 80, 270).Select
```

[6] それ以外の形式で、警告文が現れる場合もある。

また、台形を「フリーフォーム」で描かせるプログラムは、次のようになる。

```
With ActiveSheet.Shapes.BuildFreeform(msoEditingAuto, 80, 270)
  .AddNodes msoSegmentLine, msoEditingAuto, 220, 100
  .AddNodes msoSegmentLine, msoEditingAuto, 305, 100
  .AddNodes msoSegmentLine, msoEditingAuto, 360, 270
  .AddNodes msoSegmentLine, msoEditingAuto, 80, 270
  .ConvertToShape.Select
End With
```

これらのプログラムを見ると、ある程度の英単語が分かっていれば、何が書いてあるかは「想像できる」ものであることが確認できる。ただ、文法が独特であり、これを「作文」できるようになるには、相当慣れが必要であろうことも確認できる[7]。

また、各行を比較すると、直線を描く指示はほぼ同じ文面であることが分かる[8]。そこで、異なっている数値を比較すると、これらは「絶対的な指示」でされていることが分かる。ただし、この場合の「絶対的な指示」は表２の値ではなく、Excelのシート上の座標である。そのため、表１の相対的な位置の値を使って指示をする場合は、まず「絶対的な指示」に変換し、次にExcelシートの座標に変換する必要があることが確認できる。また、座標は直線の始点の横、縦、終点の横、縦の順に並んでおり、直線の終点が次の直線の始点であることも分かる。

次に、「フリーフォーム」で台形を描かせる指示を比較すると、表２のように、指示０（始点）から順に、指示がされていることが分かる。

これらを踏まえ、今回は以下のようにプログラムを作成することにする。

・多角形（n角形）を描くように指示し、角が四つ（n=4）であることは別に指示する
・座標は、Excelシートにあるものを読み込ませる
・座標は、表１の相対的な値を読み込み、表２の絶対的な値に計算式で変換させる
・求めた絶対的な値を、Excelシートに配置するための座標に変換させる
・変換した座標に基づいて、図表を描かせる

４　指示の汎用化

次に、これらの指示をプログラムに組み込む方法を説明する。

以下は、配布している「GraphPaperA.xlsm」から抜粋する形で説明する[9]。

まず、座標を数値で指示していたものを、表１、表２の右側にある「記号」、つまり変数をプログラムに組み込む方法を説明する。そのため、`Dim`と`ReDim`を使って`x`、`y`、`xa`、`ya`といった記号の存在を「定義」する[10]。

今回は、まずA2で`Dim`を使って`x()`、`y()`、`xa()`、`ya()`と定義する。次にA3でExcelシート「data」

[7] 本稿では、説明をしやすくするため、抽出した指示の数値を四捨五入するなど調整を行なった。
[8] 行初めの「ActiveSheet」は、「表示しているシートの上に」という意味である。これにより、プログラムを起動した時に、一番前にあるシートの上に、直線や図形が描かれることになる。
[9] 本稿末尾のAppendixにもプログラムを掲載しているが、こちらはページ数上限の都合、「GraphPaperA」ではなく、これに加筆して多くの図形を一度に描けるようにした、４章で説明する「GraphPaperB」である。
[10] ただの数値（スカラー）なら定義を省略しなくても問題ないが、一つの記号が数値の組み合わせを意味するベクトルや行列として使う場合、あらかじめ定義しておく必要がある。

のセルL3、つまり左上から縦横の順で（3, 12）にある、数値を読み込んで「n=4」とし、最後にA4でReDimを使ってx(n)、y(n)、xa(n)、ya(n)と再定義する方法をとる。

```
´ A2　変数の定義　（Bの下線部分を省略）[11]
  Dim x(), y(), xa(), ya()
´ A3　辺の数の指定
  n = Worksheets("data").Cells(3, 12)
´ A4　変数の再定義
  ReDim x(n), y(n), xa(n), ya(n)
```

次に、定めた変数を式に組み込む。組み込む変数は、(i)には指示番号が入ることにより、x(i)、y(i)、xa(i)、ya(i)の四種だけで済ませることができるようになる。例えばxa(1)なら、「絶対的な指示」の１番目の横の値なので、表２の「６」、図６のセルL11を読み込むことになる。

このようにしてx(i)、y(i)に、Excelのシートにある値を読み込ませる。読み込ませる方法は上記のA3と同じであるが、四角形の角の数、つまり四回分まとめて、次のように指示する。

```
´ A6　辺データ（相対的な指示）の読み込み
  For i = 1 To n
    x(i) = Worksheets("data").Cells(i + 10, 13)
    y(i) = Worksheets("data").Cells(i + 10, 12)
  Next i
```

このように「For」と「Next」で挟むと、同じような指示を省略して示せるようになる。この場合は、「i=」以降が示すように、１から始めて数値が「n」になるまで、「For」と「Next」で挟んだ内容を繰り返すように指示することができる。今回は「n=4」なので、例えばx(i)の場合は、座標（11, 13）から（14, 13）、つまりセルL11からL14まで、値を読み込む作業を繰り返させることになる。

次に、表２で示した「相対的な指示」を、表３の「絶対的な指示」に変換する式を説明する。

２章４節でした説明では、表２の値を表３の値に変換する方法は、「表２の一つ上の値に、表１の同じ位置にある値を足す」であった。これを汎用の形式で示すと、次のようになる。

```
´ A7　辺データの指示を相対から絶対に変換　（Bの下線部分を省略）
  For i = 1 To n
    xa(i) = xa(i - 1) + x(i)
    ya(i) = ya(i - 1) + y(i)
  Next i
```

これに、始点の値を追加する。ここでは変数を、「i=0」とし、例えばx(0)にも数値を置くことができる性質を利用し、x(0)で始点を指示する。

しかし、表２のx(0)と表３のxa(0)の値が同じであること、x(-1)のデータがないことから、A7の式でxa(0)を計算することはできない。そこで、xa(0)に直接シートの値を読み込むため、次のよ

[11] プログラムで「´」を記入すると、同じ段落のそれ以降の部分は、プログラムとして機能しなくなる。これを利用して、作成のためのメモなどの補足説明を記している。

うに指示する。

```
' A5　始点の読み込み
  xa(0) = Worksheets("data").Cells(10, 13)
  ya(0) = Worksheets("data").Cells(10, 12)
```

次に、A4を使って変換した「絶対的な指示」を、Excelシートの座標に変換する方法を説明する。Excelでは、A9で示すように、シートの座標に合うよう、xa(i)とya(i)の値を変換する必要がある。

```
' A9　辺データを座標に変換　(Bの下線部分を省略)
  For i = 0 To n
    xa(i) = xa(i) * s + x0
    ya(i) = -ya(i) * s + y0
  Next i
```

今回は始点も含めるため、例えばx(0)からx(4)の五つの値を変換することになる。この場合の「=」等号は「等式」、つまり左右の値が同じという意味ではなく、右辺の計算で出された値を、左辺に代入、この場合は上書きするという意味になる。また「x0」、「y0」はA8で事前に定めたExcelのシート上の座標であり、これらは図６の方眼紙の横０、縦０にあたる。また「s」は方眼紙の座標に合わせて書かれている値を、Excelのシートの座標の単位に変換するための係数である[12]。また、方眼紙が右と上をプラスとしているのに対し、Excelのシート上では右と下がプラスの方向と定められている。そこで今回は、縦の値を正負逆にするため、式の頭にマイナスを付けている。

次に、このExcelのシート上に配置するために変換した値を使って、直線を引く。

この場合、直線を引かせる指示は、次のようになる。

```
' A11　辺の描画　(Bの下線部分を省略)
  For i = 1 To n
    ActiveSheet.Shapes.AddConnector(msoConnectorStraight, xa(i - 1), ya(i - 1), xa(i),
    ya(i)).Select
    Selection.ShapeRange.Line.Weight = 6
    Selection.ShapeRange.Line.ForeColor.SchemeColor = 2
  Next i
```

この直線の始点と終点の座標を示す記号にaが付いているように、「絶対の指示」で指定されている。例えば「i=1」、つまり１番目の指示では、表２の０番目（i-1）と１番目（i）の座標を線で結べ、という指示をしていることになる。

また、「.Select」の後には、その選択している図形の設定を変更する指示を行っている。この場合は、線の色は色番号２、太さは６ポイントに変更する指示をしている[13]。なお、後から通常のマウス

[12]「GraphPaperA」の「y0」は 298、「s」は 28.346 にしてある。これらは、何度か描いて、調整した値である。なお、Windows 版の Excel 上で調整したものであり、Mac 版では方眼紙上に正しく配置されないことがある。その場合は、台形を PowerPoint 上にコピーした際に、サイズを横 10cm、縦 6 cm に変更すればよい。

[13] 色番号の説明は、４章１節を参照。

を使った操作で変更できるため、指示が書かれていなくても問題はない。

次に、これを図形として描く方法を説明する。今回は、直線をつないだ図形なら、それぞれの角の座標を指定すれば、つなげて図形にしてくれる「フリーフォーム」を用いる。
この「フリーフォーム」だけで多角形を描くためのプログラムは、次のようになる。

```
´ A10　フリーフォームの描画　より抜粋（Bの下線部分を省略）
  With ActiveSheet.Shapes.BuildFreeform(msoEditingAuto, xa(0), ya(0))
    For i = 0 To n
      .AddNodes msoSegmentLine, msoEditingAuto, xa(i), ya(i)
    Next i
    .ConvertToShape.Select
    Selection.ShapeRange.Line.Weight = 3
    Selection.ShapeRange.Line.ForeColor.SchemeColor = 8
    Selection.ShapeRange.Fill.ForeColor.SchemeColor = 47
  End With
```

この場合は、図形の辺の太さを３ポイントに、線の色を色番号８、内側の色を色番号47に指定している。

なお、「GraphPaperA」では、A10の「フリーフォーム」の方を、A11の「辺の描画」よりも先に指示するようにしてある。これは、後から描いた方が上に配置される性質を利用して、図形の上に直線を配置するためである。

５　Excelに描かれた図をPowerPointに移す

次に、作成された台形を、PowerPointファイルに貼り付ける方法を説明する。

まず、Excel上にある「台形」を選択して「コピー」の操作をし、PowerPointの作成画面で「貼り付け」の操作を行う[14]。Excel上の台形の高さが６cmであるのに対し、それを貼り付けるPowerPointのスライドの縦は約19cmであるため、このままではスライドに対して小さいと感じられるであろう。そこで、PowerPoint上に貼り付けた台形に対し、次の操作を行う。

手順１：図形を、「選択」する
手順２：画面上部に現れる、「図形の書式」を、「選択」する
　　画面上部メニュー右端に「サイズ」という項目が現れる。
手順３：サイズ欄の「高さ」、「幅」に２倍の数値を入力する

今回の台形は縦６cm、横10cmなので、「高さ」に12cm、「幅」に20cmと記入すればよい。

次に、直線で描いた図形をPowerPointファイルに貼り付ける方法を説明する。まず、直線をまとめてコピーするには、次のようにする。

手順１：一本目の線の上で、「選択」する
手順２：二本目以降は、「Shift」キーまたは「Ctrl」キーをおさえたまま、「選択」する

[14] Excelなどの基本操作については、宇多（2017）を参照。

手順３：コピー（Ctrl+C）し、PowerPointに移動し、スライド（用紙）上に貼り付ける（Ctrl+V）
手順４：そのまま線の上にカーソルを重ね、マウスの右ボタンを押す
手順５：現れた一覧から、「グループ化」を「選択」すると、四本の線が一つにまとめる
手順６：画面上部のメニューに「図形」の「書式設定」が現れるので「選択」する
手順７：右端に現れた「サイズ」欄の数字を「高さ」を12cm、「幅」を20cmに変更する

わざわざグループ化をするのは、グループ化をしないまま、「選択」した複数の図形のサイズを変更すると、配置がおかしくなってしまうからである。

Ⅳ　プログラムの配置、実行、その後の作業

１　応用例　ペントミノの描画

３章で説明したプログラム「GraphPaperA」の台形を描く作業は、手作業でも行えるため、実用性はそれほど高くない。そこで次に、このプログラムに少し加筆するだけで、手作業で描くには困難な数の図形を、まとめて描いてくれるプログラム「GraphPaperB」にする方法を説明する。

今回は、図７のペントミノ、正方形を５つ組み合わせて作った図形を使ったパズルを、自動的に描いてくれるように、プログラムを改良する。

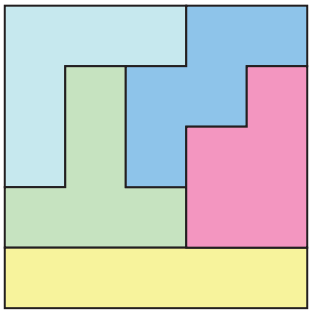

図７　ペントミノの例

そこで今回は、図７の縦５マス、横５マスの正方の枠の中に５つの図形を入れた、ペントミノを例にした。このプログラムは汎用性が高く、４章２節で示すように、縦横４マスに三角形や平行四辺形などを組み込んだタングラムや、12の図形からなる縦横８マスのペントミノを描くこともできる。

また、「GraphPaperA」、「GraphPaperB」の両プログラムを比較すると分かるように、基本は同じである。しかし、改良したプログラム「GraphPaperB」は、辺の数が異なる図形を複数作成し、それぞれの位置を指定することができるようになっている。また、図形ごとに色を指定できるようにしてある。なお、３章の「GraphPaperA」は、１cm単位の縦９cm、横12cmの方眼紙に配置していた。これに対し、「GraphPaperB」では、８cmの正方形の中に図形を配置するようになっており、８cm四方を分割する数は「枠のサイズ」で指定できるようにしてある。

次に、図形の性質と座標のデータは、描く図形が複数に増えたことから、表３のようになる。

表３　ペントミノのデータ（相対的な指示）

枠を分割する数（ｌ）	5									
図形の数（m）	5									
図形の辺の数（最大）（n）	10									
図形番号（j）	1		2		3		4		5	
辺の数（nm）	6		8		4		4		4	
色番号（I列参照、c）	41		42		43		44		45	
	縦	横	縦	横	縦	横	縦	横	縦	横
指示０：始点（図形によって異なる、i=0）	0	0	0	3	0	4	2	1	3	2
指示１：１本目（i=1）	0	3	0	1	0	1	0	2	0	2
指示２：２本目（i=2）	1	0	3	0	5	0	1	0	2	0
指示３：３本目（i=3）	0	−2	0	−1	0	−1	0	−1	0	−3
指示４：４本目（i=4）	2	0	−1	0	−5	0	1	0	−1	0
指示５：５本目（i=5）	0	−1	0	−2			0	−1	0	1
指示６：６本目（i=6）	−3	0	−1	0			1	0	−1	0
指示７：７本目（i=7）			0	2			0	−1		
指示８：８本目（i=8）			−1	0			−2	0		
指示９：９本目（i=9）							0	1		
指示１０：１０本目（i=10）							−1	0		

なお、今回は各図形を直線に分けて描いても意味をなさないため、「フリーフォーム」だけを描くようにしてある。

図８は、配布しているExcelファイル「GraphPaperB.xlsm」のシート「data」である。

図８　GraphPaperB の作業画面

図８は、右側を図形番号３の途中から省略しているが、実際は「図形の数（m）」に入力した数、この場合は五つの図形を描くようになっている。

また、「GraphPaperB」は多数の図形を描くため、「GraphPaperA」の x(i)、y(i)、xa(i)、ya(i) に「j」を加え、x(i,j)、y(i,j)、xa(i,j)、ya(i,j) にしてある。この「j」に図表番号を入れるのである。例えば、x(3,2) なら、図形番号２の「相対的な指示」の３番目の横の値となり、表３の「０」、図８のセルO13を読み込むことになる。

また、「For i = 1 To n」と「Next i」で挟むことにより、繰り返しさせるようにしていた指示を、さらに「For j = 1 To m」と「Next j」で挟むことで、さらに繰り返させるようにしている。これにより、辺を描くことを「n」回繰り返させて図形を描かせるという作業を、さらに図形の数だけ、つまり「m」回繰り返させることができるようになる。

また、仕様の変更に合わせ、いくつか変更を加えた。まず、直線の別に分けても意味がないため、「フリーフォーム」のみにした。次に、８cmの正方形に納めるように、単位である「s」の大きさを「枠を分割する数」、図８のセルL1で８cmを割った値とし、簡単に変更できるようにした。また、「GraphPaperB」では、Excelのシートのルールに合わせ、ゼロ座標を枠の左上に配置し、下方向をプラスとした。これにより、B9のya(i,j)の変換式では、マイナスを取り除いてある。

また、図形に番号で色を指定できるようにした。色番号の一覧は、図８のようにH列、I列に配置してあり、そのプログラムは次のようになる。

```
' B0　番外 図形の色指定のための色番号一覧
  For i = 1 To 7
    Worksheets("data").Cells(i, 8) = i
    ActiveSheet.Cells(i, 9).Interior.ColorIndex = i + 1
  Next i
  For i = 1 To 56
    Worksheets("data").Cells(i + 7, 8) = i + 7
    ActiveSheet.Cells(i + 7, 9).Interior.ColorIndex = i
  Next i
```

このプログラムが二つに分かれており、それぞれの行で右辺の内容が異なるのは、色指定の番号がセルと図形では異なることによる。今回用いているExcel VBAでは、セルの色は「ColorIndex」、図形の色は「SchemeColor」で指定するという違いがある。これに対し、今回は「セル」に色を塗るプログラムを使って、図形に塗る色の番号の表を作成する必要があった。そのため、プログラムを組む際、色指定の番号の違いを解消するよう、上記のように指示する必要があった。

２　他図形の描画

また、この「GraphPaperB」と別のデータを使って描いたのが、図９である[15]。

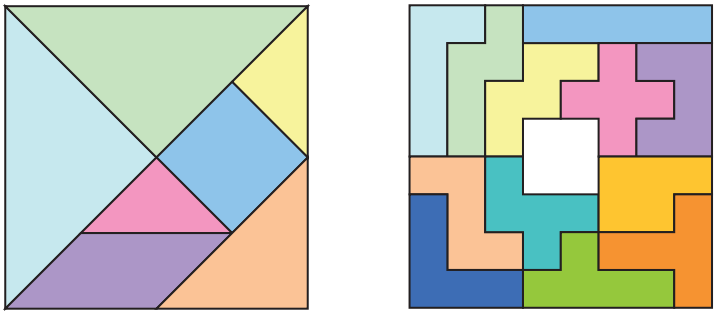

図９　他の図形の例（左：タングラム、右：８×８のペントミノ）

これらの例は、「フリーフォーム」が多様な多角形を描くことができることと、プログラム

[13] ８×８のペントミノの、中央縦横２マスの正方形は空白である。

「GraphPaperB」では作成する図形の数を変えることができることの、二つの性質を利用したものである。つまり、このような図形を描く場合は、プログラムを変更する必要はないことを示した。逆に言えば、汎用性の高いプログラムを組めれば、いちいちプログラムを組まなくても済むのである。

図９の両図を描くために必要なデータをまとめたものが、表４、表５である。

表４　タングラムのデータ（相対的な指示）

枠を分割する数	4													
図形の数	7													
図形の辺の数（最大）	10													
図形番号	1		2		3		4		5		6		7	
辺の数	3		3		3		4		3		4		3	
色番号	41		42		43		44		45		46		47	
	縦	横	縦	横	縦	横	縦	横	縦	横	縦	横	縦	横
指示０：始点	0	0	0	0	3	1	2	2	2	2	1	3	4	2
指示１：１本目の移動	0	4	2	2	1	1	1	1	−1	1	−1	1	−2	2
指示２：２本目の移動	2	−2	2	−2	0	−2	1	−1	2	0	2	0	2	0
指示３：３本目の移動	−2	−2	−4	0	−1	1	−1	−1	−1	−1	1	−1	0	−2
指示４：４本目の移動	0	0					−1	1			−2	0		

表５　８×８のペントミノのデータ（相対的な指示）

枠を分割する数	8																							
図形の数	12																							
図形の辺の数（最大）	12																							
図形番号	1		2		3		4		5		6		7		8		9		10		11		12	
辺の数	6		8		10		4		12		8		8		6		10		8		6		8	
色番号	41		42		43		44		45		46		47		48		49		50		51		52	
	縦	横	縦	横	縦	横	縦	横	縦	横	縦	横	縦	横	縦	横	縦	横	縦	横	縦	横	縦	横
指示０：始点	0	0	0	2	1	3	0	3	1	5	1	6	4	0	5	0	4	2	6	4	4	5	5	7
指示１：１本目の移動	0	2	0	1	0	2	0	5	0	1	0	2	0	2	0	1	0	1	0	1	0	3	0	1
指示２：２本目の移動	1	0	2	0	1	0	1	0	1	0	3	0	2	0	2	0	1	0	1	0	1	0	3	0
指示３：３本目の移動	0	−1	0	−1	0	−1	0	−5	0	1	0	−2	0	1	0	2	0	2	0	2	0	−1	0	−1
指示４：４本目の移動	3	0	2	0	1	0	−1	0	1	0	−1	0	1	0	1	0	1	0	1	0	1	0	−1	0
指示５：５本目の移動	0	−1	0	−1	0	−1			0	−1	0	1	0	−2	0	−3	0	−1	0	−4	0	−2	0	−2
指示６：６本目の移動	−4	0	−3	0	1	0			1	0	−1	0	−2	0	−3	0	1	0	−1	0	−2	0	−1	0
指示７：７本目の移動			0	1	0	−1			0	−1	0	−1	0	−1			0	−1	0	1			0	2
指示８：８本目の移動			−1	0	−2	0			−1	0	−1	0	−1	0			−1	0	−1	0			−1	0
指示９：９本目の移動					0	1			0	−1							0	−1						
指示１０：１０本目の移動					−1	0			−1	0							−2	0						
指示１１：１１本目の移動									0	1														
指示１２：１２本目の移動									−1	0														

V　おわりに　プログラム教育の必要性の確認

本稿では、プログラム教育に適した題材を小学校算数の教科書から選び、プログラム作成の前提となる考え方から、プログラムを組んで動かすまでの作業を一通り説明した。また、このプログラムは、作成に必要な考え方や手順を理解するための教員向けの教材としてだけでなく、教員が授業準備に使えるよう、また授業の教材としても使えるようにした。

一方、本稿を読めばわかるように、プログラムという技能を習得し、維持するために必要とする労力や時間は少なくない。また仮に習得したとしても、教員がプログラムを使わないとできない作業は、今日ではおそらくないはずである。また、使ってできなくはない作業も限られたものであり、その頻度もそう多くはないはずである。

日常的に使う技能、行う作業なら繰り返す中で慣れていくであろうし、慣れてしまえば何とかなるものである。しかし、めったに使わないのであれば、そもそも慣れる機会がないし、また忘れてしまうであろう。このことから、「めったにしない」ことを行うために必要なのは、忘れていることを何かを見ながら行う「再現力」であり、そのために必要なのは「専門性の高い技能」や「記憶力」よりも、「記録力」や「読解力」の方であることが分かる。

また、本稿で示したように、プログラムには英単語と数式が用いられている。この数式には数学記号が使われていることから、抽象的な数学的概念を理解していることが必要なことが分かる。つまり、これらのことから、プログラムにおける「論理性」を理解するには、最低でも中学校程度の数学と英語の理解が必要であることが確認できる。これに対し、『小学校学習指導要領』の「総則　第３　教育課程の実施と学習評価」では、「必要な論理的思考力を身に付ける」ことが目標とされている。つまり、小学生の段階では、プログラムの技能そのものを身につけさせる必要はないのである。

これらのことから、小学校の段階では、プログラムよりも使用頻度の高い、日常的に使うもので論理性を養うよう、まずは論理的に聞く、読む、話す、書くことができるようにするための訓練に重点を置いた、日本語教育が必要であろうと考えられる。

参考文献一覧

宇多賢治郎（2003）「スカイライン分析と分析用ツール『Ray』の紹介」『産業連関　－イノベーション＆IOテクニーク－』、第11巻第２号、環太平洋産業連関分析学会。

宇多賢治郎（2011）「『Ray　スカイラインチャート作成ツール（2.0j版）』の紹介」、『経済統計研究』、第38巻第４号、経済産業統計協会。

宇多賢治郎（2017）『教育の場で「説明する」ためのパソコン術』、学文社。

宇多賢治郎（2019）「スカイラインチャートなどのグラフ描画プログラムを組む方法」『産業連関』、第27巻第１号、環太平洋産業連関分析学会。

国本温子・緑川吉行＆できるシリーズ編集部（2016）『できる逆引きExcel VBAを極める 2016/2013/2010/2007対応』、インプレス。

国本温子・緑川吉行＆できるシリーズ編集部（2017）『できる大辞典 Excel VBA 2016/2013/2010/2007対応』、インプレス。

東京書籍（2015a）『新編　新しい算数３下　教師用指導書　指導編』。

東京書籍（2015b）『新編　新しい算数４上　教師用指導書　指導編』。

森口繁一（2000）『Excel/Basic基礎指南　－知らないことを知りたい人へ－』、日本規格協会。

文部科学省（2017）『小学校学習指導要領　平成29年告示』。

Appendix. プログラムファイル

このAppendixではプログラム「GraphPaperB」のほぼ全文を掲載し、その元である「GraphPaperA」との違いを示す。

違いを示すため、「GraphPaperA」から「GraphPaperB」にする際、加筆した部分に下線を引いた。

なお、著作権や使用上の注意等の説明、色番号を示すプログラムは省略した。

```
Sub GraphPaperB()
' B1　記号の意味
' n 図形の辺の数，　x() 辺の幅，　y() 辺の高さ，　xa() 横座標，　ya() 縦座標
' 追加：l 枠のサイズ（正方），　m 図形番号，　c() 色番号，　nm() 図形の辺の数

' B2　変数の定義(nm、cを追加)
  Dim x(), y(), xa(), ya()
  Dim nm(), c()

' B3　辺の数の指定（枠のサイズl、図形番号mを追加）
  n = Worksheets("data").Cells(3, 12)
  l = Worksheets("data").Cells(1, 12)
  m = Worksheets("data").Cells(2, 12)

' B4　変数の再定義（m、nm、cを追加）
  ReDim x(n, m), y(n, m), xa(n, m), ya(n, m)
  ReDim nm(m), c(m)

' B5　始点の読み込み（辺の数nm、色c、図形番号jを追加）
  For j = 1 To m
    xa(0, j) = Worksheets("data").Cells(10, (j - 1) * 3 + 13)
    ya(0, j) = Worksheets("data").Cells(10, (j - 1) * 3 + 12)
    nm(j) = Worksheets("data").Cells(6, (j - 1) * 3 + 12)
    c(j) = Worksheets("data").Cells(7, (j - 1) * 3 + 12)
  Next j

' B6　辺データ（相対的な指示）の読み込み（図形番号jを追加）
  For j = 1 To m
    For i = 1 To nm(j)
      x(i, j) = Worksheets("data").Cells(i + 10, (j - 1) * 3 + 13)
      y(i, j) = Worksheets("data").Cells(i + 10, (j - 1) * 3 + 12)
    Next i
  Next j
```

```
' B7　辺データの指示を相対から絶対に変換（図形番号ｊを追加）
  For j = 1 To m
    For i = 1 To nm(j)
      xa(i, j) = xa(i - 1, j) + x(i, j)
      ya(i, j) = ya(i - 1, j) + y(i, j)
    Next i
  Next j

' B8　Excel状に配置するための定数
' ゼロ座標は左上、プラスは下と右
' s　1マスの大きさ，　x0 横の始点，　y0 縦の始点，　l 枠を分割する数
' GraphPaperA では、　s=28.346 、　y0=298
  s = 227 / l
  x0 = 50
  y0 = 50

' B9　辺データを座標に変換（図形番号ｊを追加、yaの計算から「-」を削除）
  For j = 1 To m
    For i = 0 To nm(j)
      xa(i, j) = xa(i, j) * s + x0
      ya(i, j) = ya(i, j) * s + y0
    Next i
  Next j

' B10　フリーフォームの描画（図形番号ｊを追加）
  For j = 1 To m
    With ActiveSheet.Shapes.BuildFreeform(msoEditingAuto, xa(0, j), ya(0, j))
      For i = 1 To nm(j)
        .AddNodes msoSegmentLine, msoEditingAuto, xa(i, j), ya(i, j)
      Next i
      .ConvertToShape.Select
      Selection.ShapeRange.Line.Weight = 3
      Selection.ShapeRange.Line.ForeColor.SchemeColor = 8
      Selection.ShapeRange.Fill.ForeColor.SchemeColor = c(j)
    End With
  Next j

' B11　辺の描画（不要のため削除）
' 削除された箇所は、本稿の３節を参照。

End Sub
```