VISUAL BASIC[1)]による応答的プログラミング
――ユーザの成績に応じた問題提示法――

門田　幸太郎*

本稿では，ユーザの特性に応じて，項目の提示方法を変えていくプログラミング法について考察した。ユーザの特性としては，一連の問題に対する過去の成績を用いた。今までの試行数が10回以下ならば，成績に関係なく全問題を提示する。試行数が10回以上ならば，成績が一定の基準に達している問題は提示せず，成績が一定の基準に達していない問題については提示するようにした。提示した問題への反応の正誤を判定し，その結果を書き込み，記録を更新する。これが本プロジェクトの目的である。プログラミング言語としてはVISUAL BASICを用いた。

キーワード：VISUAL BASIC，応答的プログラミング，問題提示法

コンピュータのもつ重要な特質として応答性（responsiveness）をあげることができる。応答性とは，コンピュータ・ユーザの特性に応じて反応するコンピュータ・プログラムの性質をいう。どのユーザに対しても一様に反応するのではなく，ユーザの特性に応じて反応するようにプログラムを組むことができるのである。ここではユーザの特性として，一連の問題についての現在までの各ユーザの正解率を取り上げた。正解率が一定基準に達していない問題のみを選択して提示するようにした。ユーザにとって，一定基準に達していない問題は，まだ理解されていないものとして，毎試行かならず提示し，一定基準に達した問題は，すでに理解されたものとして，試行の間隔をおいて提示する。これにより，ユーザ（学習者）にとって学習すべき課題を選択的に与え，それに集中させることができるようになる。また試行結果を記録することにより，各問題に対する学習者の理解度を検討することもできるようになる。

１　本プロジェクトの全体の流れとプログラムの構成

開始の画面（図４参照）に対して，ユーザが名前を入力する。データ・ファイルdata.txtから，データとして登録されているユーザの人数，問題，正解および各ユーザの過去の記録を含むデータを読み込まれる。入力された名前をもとに，読み込まれた過去の記録を検索する。入力された名前が，新規データの場合は，全問題を提示する。入力された名前が，既存の場合は，過去の試行数の９割を基準として，過去の正答率が基準を越えない問題のみを提示する。「開始」ボタンをクリック

＊立命館大学産業社会学部教授

することにより（図５参照）問題が提示され，各問題に対する反応の正誤を判定し（図６参照），その結果をもとに記録を更新する（図７参照）。以上がプロジェクトの手順の概要である。

次に，プロジェクトのプログラムの構成について述べる。プログラムは，標準モジュールとForm1モジュールとForm2モジュールの３つのモジュールで構成されている。標準モジュールはTypeステートメントからなるモジュールであり，本プロジェクトで利用するレコードを定義する役割を担っている。Form1は２つのプロシージャから構成されているモジュールであり，ユーザ名を取り込む役割を担っている。Form2は４つのプロシージャから構成されているモジュールであり，データの読み込みから問題の提示，入力された答案の正誤判定から記録の更新の役割を担っており，本プロジェクトの中心部をなす。

２　各モジュールのコード

各モジュールのコード記述を中心に，プログラム内容について説明をする。とくに，本プロジェクトの中心部分であるForm2のモジュールについては，フローチャートを併用しながら，プロシージャごとに説明することにする。

２－１　標準モジュール（Moduel1-1）のコード

```
Public Type record
    name As Variant
    trial As Variant
    item（1 To 10）As Variant
End Type
Public Data（1 To 200）As record
```

標準モジュールModuel1-1は，データ・ファイル内のレコードを定義するものである。レコードとは，ユーザごとのデータ・セットを意味する。ここでは，本プログラムをすでに利用したことのあるユーザのこれまでの成績が記録されている。レコードで用いる要素名はnameとtrialとitem（1 To 10）の合計12個である。item（1 To 10）のように，要素名に配列を用いることもできる。いずれも，バリアント型として変数宣言されているので，データ型が，数値でも文字でも対応できる。要素nameはユーザの名前を，要素trialは各個人の今までの全試行数を示す。要素item（1 To 10）のitem（1）からitem（10）は問題1から問題10の各問題に今までの試行で何回正しい答えを出したかを示すものである。配列型のData（1 To 100）をレコード型として変数宣言する。Data（i）のｉはメンバー名といわれ，ｉ番目の人のレコードであることを示している。標準モジュールでデータ・ファイルdata.txtのレコード部分を定義することにより，Form1とForm2のいずれのモジュールでも共通にデータを扱うことができるようになる。

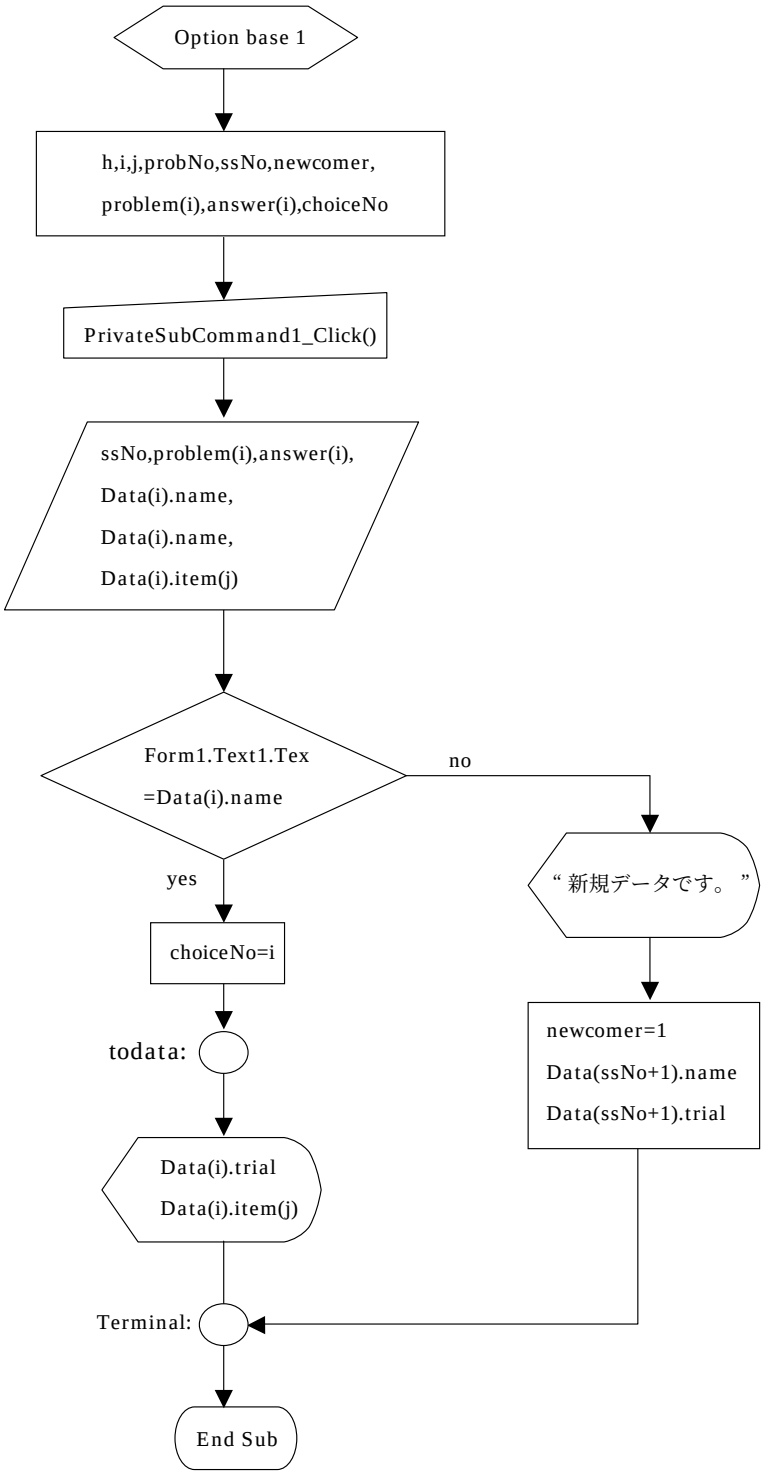


図1　変数宣言とCommand1_Click（）プロシージャのフローチャート

２－２　Form1モジュールのコード

```
Private Sub Form_Load ()
  Dim msg As Variant
  msg = “名前をテキストボックスに入れてコマンドボタンを押してください”
  MsgBox msg, vbOKOnly
End Sub

Private Sub Command1_Click ()
  Form2.Show
  Form1.Hide
End Sub
```

Form1のモジュールは次のForm2を実行する準備の役割を担っている。Form_Load（）のプロシージャは，メソッドといわれる。一般にプロシージャはユーザの操作，すなわちイベントによって起動されるのに対して，メソッドは，イベントによらずにプログラムの読み込みと同時に起動される。ここでは，メッセージ・ボックスに「名前をテキスト・ボックスに入れてコマンド・ボタンを押してください。」と手続きの説明が表示される。ユーザはテキスト・ボックスに名前を入力して，「データ読込み」のキャプションが付いているcommand1のコマンド・ボタンをクリックすることになる。 Command1_Click（）のプロシージャはプログラムを実行してコマンド・ボタンをクリックした場合，Form1.HideでForm1を表示しないようにし，Form2.ShowでForm2を表示するようにする。これによりプログラムで画面を切り替えることができる。Form1の実行により，Form2でユーザ名を取り込むことができるようになる。

２－３　Form2モジュール

Form2モジュールは，「開始」，「問題」，「確認」，「終了」の４つのプロシージャからなる。Command1_Click（）のプロシージャは「開始」のキャプションがついたコマンド・ボタンをクリックした場合のプロシージャを定義したものである。Command2_Click（）のプロシージャは「問題」のキャプションがついたコマンド・ボタンをクリックした場合のプロシージャを定義したものである。Command3_Click（）のプロシージャは「確認」のキャプションがついたコマンド・ボタンをクリックした場合のプロシージャを定義したものである。Command4_Click（）のプロシージャは「終了」のキャプションがついたコマンド・ボタンをクリックした場合のプロシージャを定義したものである。

フローチャート（図１参照）により，Command1_Click（）プロシージャの全体の流れについて説明する。

Form2では，プロシージャに先立って，Option Base文と変数定義文がある。Option Base文は

配列の添字の最小値を決定するものである。通常，Visual Basicでは配列の添字の最小値はあらかじめ0と定められている。この最小値を1とするためにOption Base文が用いられる。変数定義文をここに配置することにより，これらの変数を共通変数として，Form1とForm2とのモジュールで用いることができるようになる。変数h，i，jはプログラムの中で用いられる作業用のカウンター変数で整数として定義されている。newcomerは，Form1のテキスト・ボックスに入力されたユーザ名がデータ・ファイルに含まれている既存のものか，含まれていない新規のものかを判別する場合に用いられる。入力されたユーザ名が既存の場合，newcomerはディフォルト（既定値）の0のままであり，新規の場合，newcomerは1となる。変数problemNoは問題数を示す整数であり，変数ssNoは既存のデータに登録されている個人のデータ・セットの数，すなわち，今までにこのプロジェクトを利用した人の数を示す整数である。choiceNoは，プログラムを利用する人が既存のデータの何人目にあたるかを示す整数である。既存のデータに含まれていない新規ユーザである場合はchoiceNoの値はデータ・セットの数に1を加えた値となる。problem（1 To 100）は問題を読み込むための配列である。ここでは，問題数は全部で10問であるが，100問まで設定することができる。もちろん，上限の100を増大することにより，より大きな問題数に対応することもできる。answer（1 To 100）は問題に対応する正解を読み込むためのファイルである。この問題と正解のデータを換えることによって，別の問題，正解セットのもとでもこのプログラムを利用することができる。

２－３－１　Form2モジュールのCommand1_Click（）プロシージャのコード

```
Option Base 1
Dim h, i, j As Integer  'カウンタ
Dim probNo As Integer  '問題数
Dim ssNo As Integer  'ssの数
Dim newcomer As Integer  '新規データの判別
Dim problem(1 To 100) As Variant  '問題データ
Dim answer(1 To 100) As Variant  '正解データ
Dim choiceNo As Integer  '選択されたユーザの番号
Private Sub Command1_Click()  '開始
  Label4.Caption = Form1.Text1.Text        ①
  Command1.Visible = False        ②
  CurrentY = 3000
  newcomer = 0
  probNo = 10
  Open "h:MyVB6\New\data.txt" For Input As #1  'data.txt データ読み込み
    Input #1, ssNo
    Print "問題数は"; probNo; "個です。", "データ数は"; ssNo; "人分です。"
```

```
    For i = 1 To probNo
      Input #1, problem(i)
    Next
    For i = 1 To probNo
      Input #1, answer(i)
    Next
    For i = 1 To ssNo
      Input #1, Data(i).name, Data(i).trial
      For j = 1 To probNo
        Input #1, Data(i).item(j)          ③
      Next
    Next
  Close #1
  For i = 1 To ssNo
  If Form1.Text1.Text = Data(i).name Then choiceNo = i: GoTo todata          ④
  Next
      MsgBox "新規データです。"
      newcomer = 1
      Data(ssNo + 1).name = Form1.Text 1.Text
      Data(ssNo + 1).trial = 0
      Print Data(ssNo + 1).name, Data(ssNo + 1).trial
      GoTo terminal
todata:
  Print
  Print "今までの試行数は"; Data(i).trial; "回でした。"  '過去の試行数
  Print
  For j = 1 To probNo
    Print"（"& j &"）", Data(i).item(j)     ⑤
  Next
  Print
terminal:
End Sub
```

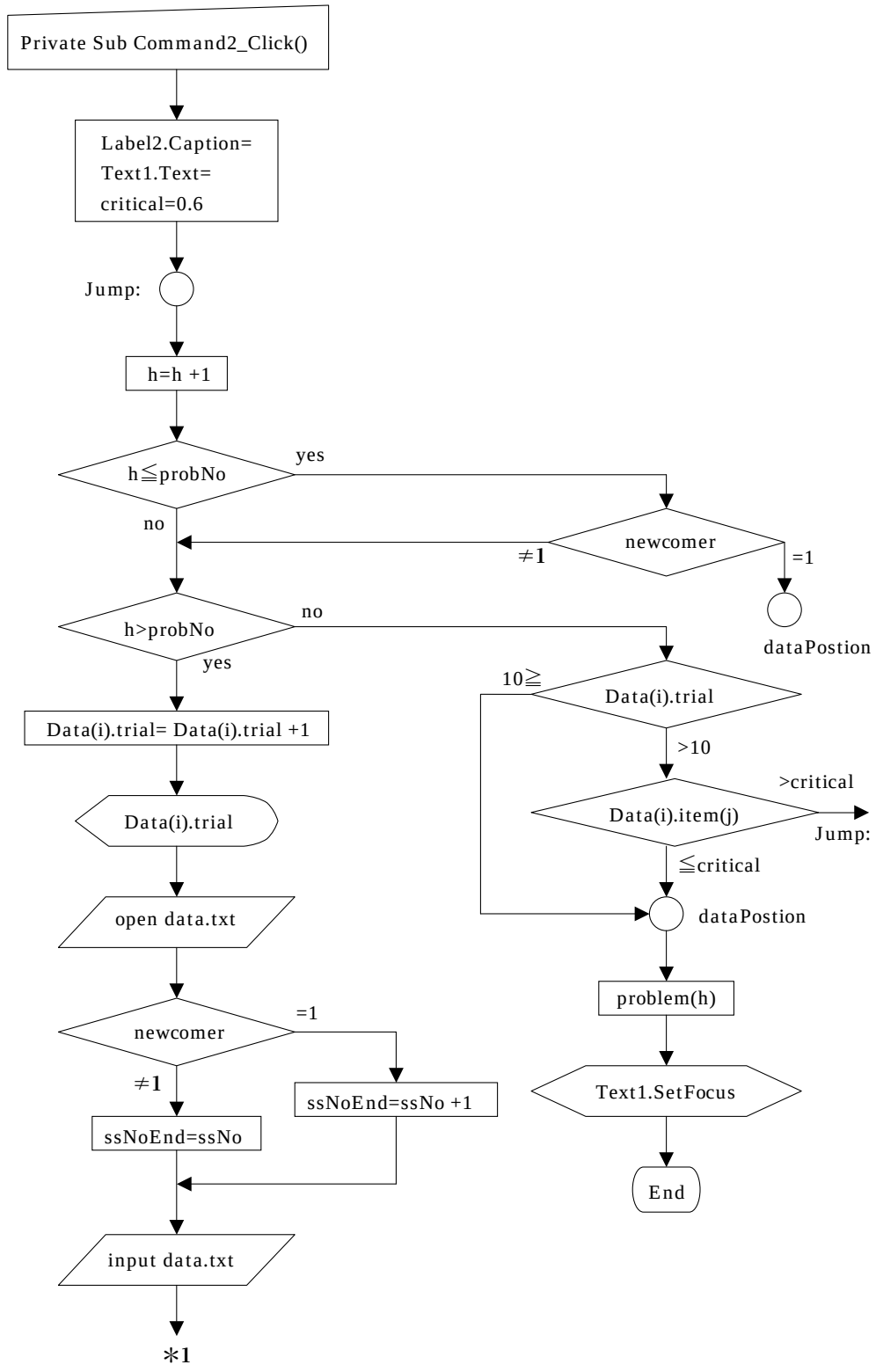
Form2モジュールのCommand1_Click（）プロシージャは「開始」のコマンド・ボタンをクリックした場合に対応するものである。これによりデータ・ファイルdata.txtが読み込まれる。data.txt

は４つのカテゴリーにより構成されている。１番目のカテゴリーは，レコードとして記録されているユーザの人数を示すssNoという整数である。２番目のカテゴリーは問題文を入れるためのproblem（1 To100）という配列である。problem（1）が“将門の乱”であり，problem（2）が“南北朝合一”となり，以下順にproblem（10）の“平城遷都”まで割り振られる。３番目のカテゴリーは，先の問題に対応する正解を入れるためのanswer（1 To100）という配列である。answerの要素数も問題文のための配列と対応しているので，ここでは当然10である。４番目のカテゴリーは標準モジュールで定義されているレコードに相当するものである。12個の要素のうち，最初の要素はユーザ名，２番目の要素はそのユーザの今までの試行数，３番目は第１問への今までの正答数を，４番目は第２問への今までの正答数を，そしてレコードの最終である12番目の要素は第10問への今までの正答数を示す。

このように，「開始」ボタンをクリックすることにより，data.txtに含まれている個人のデータ・セットの数と問題文とそれに対応した正解が読み込まれ，各個人の名前と試行数と各問に対するこれまでの正答数が読み込まれる。次に，Form1のテキスト・ボックスに入力されたユーザ名と一致するものがデータ・ファイルdata.txtのレコードData（i）.nameの中にあるかどうかが判定される。入力された名前に一致するデータがある場合，それを選び出し，一致するデータがない場合，新規データとして取り扱う。既存のデータがある場合，過去にこのプロジェクトを何回利用したかという試行数と各問題に対して正答を何回出したかを表示する（図６参照）。一致するデータがない場合，メッセージ・ボックスで「新規データです。」と表示し，データ・セットの最後に新しいデータとして，ユーザ名と試行数を割り振る。ユーザ名としてはForm1のTex1.Textに入力された名前を，試行数としては０をそれぞれ入力し，それを画面表示する。

先の標準モジュールでのメッセージ・ボックスの指示で，ユーザはテキスト・ボックスに自分の名前を入力することになる。まず，①でデータ・ボックスに入力された名前がLabel4. Captionとして，ラベル領域に表示される。②はcommand1のコマンド・ボタンを表示しないようにさせる。「開始」のコマンド・ボタンは，ファイルを読み込んで質問を提示するための準備をするものであって，一度クリックされるとその後，利用する必要はなくなる。もし間違ってクリックすると，その時点までの試行結果が消されてしまうことになる。また，「問題」のコマンド・ボタンと混同する可能性も考えられる。このため，②で非表示の設定をしておく必要がある。CurrentY=3000で，これ以後用いられるPrint文で表示されるＹ座標を設定する。probNo=10で，問題数を10とする。probNoはパブリック変数として定義されているので，この値は他のプロシージャでも用いられることになる。

データを読み込むために，Open文でデータ・ファイルdata.txtを開く。まず，ユーザの登録数ssNoが読み込まれる。先のCurrentYで定めた位置に問題数probNoとデータ数ssNoを表示する。次に，問題文が10個とそれに対応する正解が10個読み込まれる。最後に，レコードが読み込まれる。③では，ｉについては1からssNoまで，jについては１からprobNoまで，FOR…NEXT文でデータを読み込む。これによってｉ番目の人のレコードを名前Data（i）.nameと各個人がこれまで行った試行数Data（i）.trialと各問題に対する過去の正答数Data（i）.item（j）の記録を読み込むことになる。


Private Sub Command2_Click()
Label2.Caption=
Text1.Text=
critical=0.6
Jump:
h=h +1
h≦probNo
yes
no
newcomer
≠1
=1
dataPostion
h>probNo
no
yes
10≧
Data(i).trial
Data(i).trial= Data(i).trial +1
>10
>critical
Data(i).item(j)
Jump:
Data(i).trial
≦critical
open data.txt
dataPostion
problem(h)
newcomer
=1
≠1
ssNoEnd=ssNo +1
Text1.SetFocus
ssNoEnd=ssNo
End
input data.txt
＊1

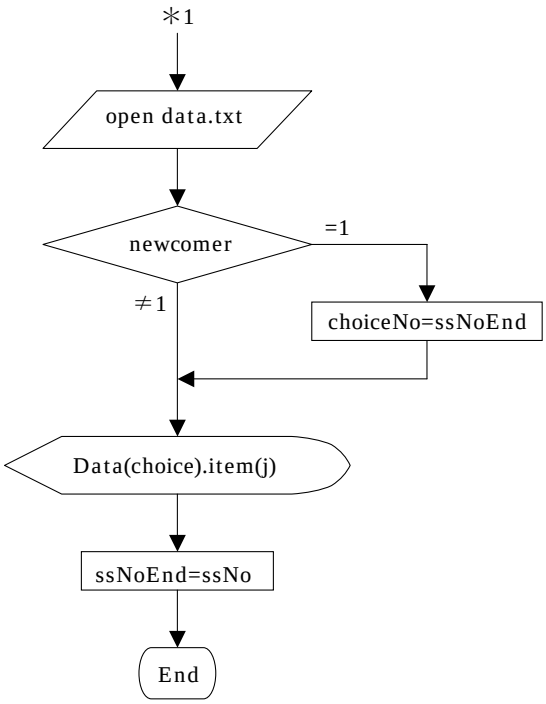


図２　Command2_Click（）プロシージャのフローチャート

④では，③で読み込まれたデータの中から，①で入力されたユーザのデータを検索する。一致するデータが見出されたときには，その値をchoiceNoに保存し，そのデータを⑤のPrint文で表示する。もし，④で該当するデータが見出せない場合は，先述のように，メッセージ・ボックスに「新規データです。」と表示し，変数newcomerの値を1としておく。さらに，新規データなので，既存のレコード数ssNoの次のメンバーとして，Data（ssNo+1）とする。要素nameには，Form1のテキスト・ボックスにテキストとして入力された名前を保存し，今までの試行数を示す要素trialは0としておく。

Command2_Click（）は「問題」と表示されたコマンド・ボタンをクリックした場合のプロシージャを定義するものである。ここでは，一定の基準にしたがって問題の提示を繰り返す。提示の基準は過去の試行数が10回を越えない場合は全問を提示する。過去の試行数が10回を越えた場合，その試行数の一定比率，ここでは90％を越えた問題は飛ばして，越えない問題のみを提示する。問題が最終問題まで行った場合，各問題に対する正誤の結果を既存の結果に書き込んで，さらに更新された結果を表示し，データ・ファイルdata.txtを更新する。

フローチャート（図２参照）でCommand2_Click（）の手続きの流れを説明する。「開始」というキャプションがついたCommand2のコマンド・ボタンをクリックすると，結果を示すラベル領域と答案を入力するテキスト・ボックスがクリアになる。基準となる正解率criticalが0.9と設定される。問題数のカウンターであるhを増加させる。このカウンターの値が問題数probNoを越えていないか

どうかが判定される。越えていない場合はさらに新規データか否かが判定される。newcomer=1の場合，つまり新規データの場合，過去の成績はないので，問題提示の手続きのためにdataPositionに飛ぶことになる。新規データでない場合，再度カウンターhが問題数probNoを越えているか否かが判定される。この判定は問題文が問題数まで提示されたか否か，つまり最後まで提示されたかどうかを判定するためのものである。条件式が満たされない場合は，過去の試行数が10回以下か否かが判定される。10回以下なら，dataPositionに飛んで，全問題が提示される。10を越える場合，その問題に対する過去の正答数Data(i).item(j)が，試行数Data(i).trialの９割以下か否かが判定される。９割を越える場合は，その問題を飛ばして次の問題に移るべく，Jump:へ飛ぶことになる。９割以下の場合は，その問題を提示して，テキスト・ボックスを入力待ちの状態にして，プロシージャを終了する。

カウンターhが問題数probNoを越えている場合，記録更新の手続きに移る。まず，試行数を1増やし，その結果を画面に提示する。次に，データ・ファイルdata.txtの書き込みが行われる。書き込みの順序は，標準モジュールで説明したデータ・ファイルの構成と同じで，登録ユーザ数，問題文，正解とレコード部分の順である。レコード部分は，ユーザ名，試行回数，各問いに対する正答数である。

データ・ファイルの更新が完了すると，新規データの場合はssNoEndを選択された番号choiceNoとして更新したデータ，正答数の記録部分を画面に表示する。最後にメッセージ・ボックスに「記録更新完了。」と表示して，このプロシージャを終了する。

２－３－２　Form2モジュールのCommand2_Click（）プロシージャのコード

```
Private Sub Command2_Click()  '問題 ── 提示と記録更新と再提示
  Dim critical As Single
  Dim ssNoEnd As Integer
  Label2.Caption = ""
  Text1.Text = ""
  critical = 0.9
jump:
  h = h + 1
  If h <= probNo And newcomer = 1 Then GoTo dataPosition
  If h > probNo Then
    Data(i).trial = Data(i).trial + 1
    Print
    Print "試行回数は"; Data(i).trial; "回になりました。"
    Print
    Open "h:\MyVB6\New\data.txt" For Output As #1  '記録更新
```

```
    If newcomer = 1 Then
      ssNoEnd = ssNo + 1
      ssNo = ssNo + 1
    Else
      ssNoEnd = ssNo
    End If
    Write #1, ssNo
    For i = 1 To probNo
    Write #1, problem(i)
    Next
    For i = 1 To probNo
      Write #1, answer(i),
    Next
    For i = 1 To ssNoEnd
      Write #1, Data(i).name, Data(i).trial,
    For j = 1 To probNo - 1
      Write #1, Data(i).item(j) ,
    Next
      Write #1, Data(i).item(probNo)
    Next
    Close #1
    Open "h:\MyVB6\New\data.txt" For Input As #1  '更新記録表示
    If newcomer = 1 Then choiceNo = ssNoEnd
    CurrentX = 3200: CurrentY = 3700
    Print " 更新後の正答数 "
    For j = 1 To probNo
      CurrentX = 3000
      Print " ("& j &")", Data(choiceNo).item(j)
    Next: Print
    Close #1
    MsgBox (" 記録更新完了。")
    End
  End If
  If Data(i).trial <= 10 Then GoTo data Position '10試行以下の場合は全問提示
  If Data(i).item(h) > Int(Data(i).trial *critical) Then GoTo jump
```

```
dataPosition:
  Label1.Caption = "（" & h &"）" & "" & problem(h)
  Text1.SetFocus
End Sub
```

まず，Command2_Click（）のプロシージャだけで用いられれる変数の宣言をする。criticalは問題提示の基準となるもので単精度の実数として宣言される。ssNoEndはデータ・ファイルに含まれるレコードの総数を示す整数である。Label2のラベル領域のキャプションとテキスト・ボックスの表示をクリアにする。ここでは，変数criticalの値を0.9としておく。プログラム制御のためのラベルをjump:としておく。問題提示のためのカウンターhにh+1を代入する。これにより，hの値を1増加させることができる。提示する問題の番号がデータ・ファイルに保存されている問題数probNo以下で，かつnewcomer=1つまり，新規データの場合，プログラムの制御はラベルdataPositionに移行する。もし提示する問題の番号が問題数を越えたならば，そのユーザの試行数Data（i）.trialに1が加えられ，その結果が画面に「試行回数は○回になりました。」と表示される。今回の試行の結果を書き込むために，データ・ファイルdata.txtが開かれる。ここで，ユーザが新規の場合，データ・ファイルを更新するときのカウンターの終了値ssNoEndと新しいユーザ数ssNoを従来のユーザ数に１を加えた数とする。また，ユーザ名がすでに登録されている場合，従来のユーザ登録数ssNoがそのままデータ・ファイル更新のためのカウンターの終了値ssNoEndとなる。データ・ファイルの更新はまずユーザ登録数の書き込みから始まる。次に問題文と正解のデータが書き込まれ，続いて，各ユーザの名前と試行数，各問題に対する正答数が書き込まれて，データ・ファイルが閉じられる。その後，更新された記録を表示するため，再びデータ・ファイルが開かれ，更新した記録内容のみがCurrentX=3000の位置に表示される。その後，データ・ファイルが今一度閉じられる。メッセージ・ボックスに「記録更新完了」と表示され，プロシージャが終了することになる。ここまでが，先の問題提示のためのカウンターがデータ・ファイルの問題数を越えた場合の手続きである。カウンターが問題数を越えない場合は，ユーザが試行数が10以下か否かが判定される。以下の場合，制御はdataPositionのラベルの位置に移る。試行数が10を越える場合は，さらに，過去の正答数が試行数の一定の基準内におさまっているか否かが判定される。基準を越えた場合は次の問題に移り，基準以内なら問題を提示することになる。

Command3_Click（）は「確認」と表示されたコマンド・ボタンをクリックした場合のプロシージャを定義するものである。ここでは，テキスト・ボックスに入力された内容，すなわち答案が正しいか否かを判定する。正答の場合は，ラベル領域（Label2.Caption）に正解と表示し，フォームに問題番号とともに「正答！！」と表示する。誤答の場合は，同じラベル領域に「誤り」と表示し，正解を表示する。

フローチャート（図３参照）でCommand3_Click（）の手続きの流れを解説する。テキスト・ボックスに答案が入力された状態で，「開始」というキャプションがついたCommand3のコマンド・

ボタンをクリックされる。この内容Text1.Textが対応する問題番号の正解と同じか否かの判定がなされる。もし入力された答案answer(h)と同じでないならば，Label2のラベル領域のキャプションに「誤り」と表示し，問題番号とともに正解を表示する。もし入力された答案answer(h)と同じならば，Label2のラベル領域のキャプションに「正解」と表示する。ユーザが新規のユーザならば，

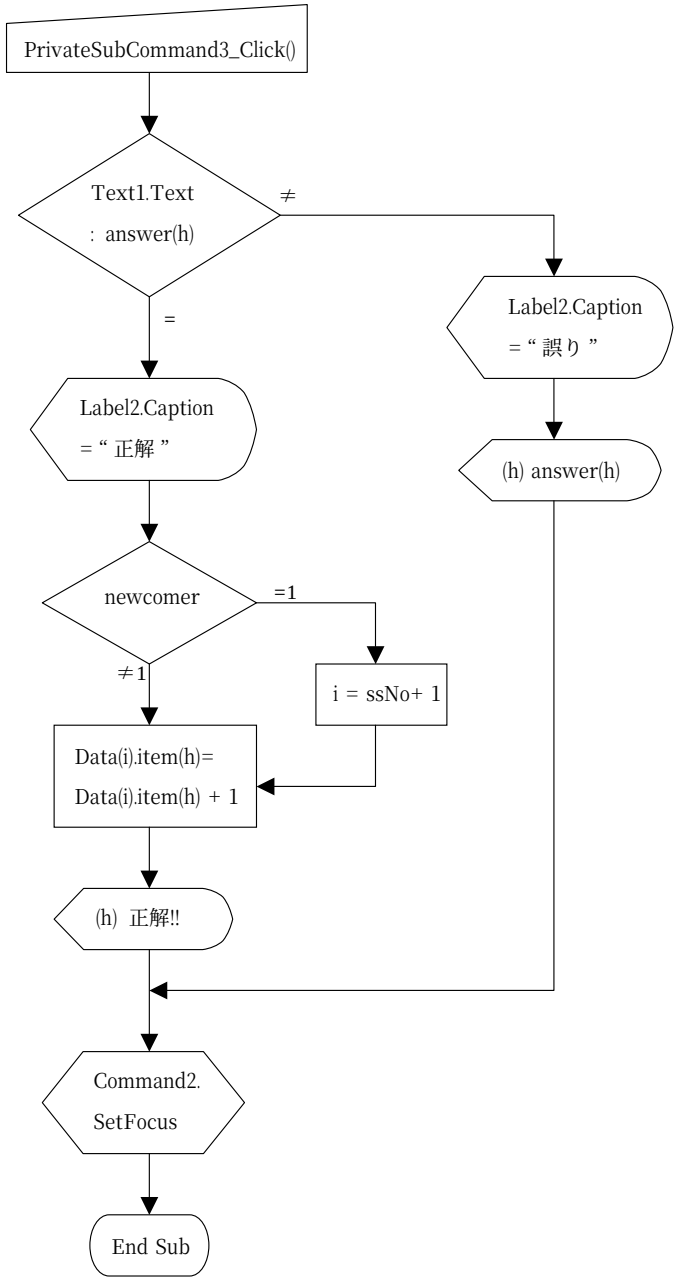


図3　Command3_Click () プロシージャのフローチャート

既存のユーザ総数ssNoに1を加えた値をｉとし，ユーザが既存のユーザならば，その時のｉの値を用いて，ユーザの各問に対する正答数を1増加させる。画面に問題番号とともに「正解！！」という結果の記録を表示し，「問題」というキャプションがついてCommand2のコマンド・ボタンをフォーカスされた状態にする。

２－３－３　Form2モジュールのCommand3_Click（）プロシージャのコード

```
Private Sub Command3_Click（）　'確認
    If Text1.Text = answer(h) Then
      Label2.Caption = "正解"
      If newcomer = 1 Then i = ssNo + 1
      Data(i).item(h) = Data(i).item(h) + 1
      Print "（" & h & "）正解!!"
    Else
      Label2.Caption = "誤り"
      Print "（" & h &"）"; answer(h);"年です。"
    End If
    Command2.SetFocus
End Sub
```

Text1.Textに入力された答案が正解を読み込んだ配列answer（h）に等しいか否かが判定される。等しい場合はLabel2のラベル領域に「正解」と判定結果を表示する。さらにここで，ユーザが新規か否かが判定され，新規の場合はメンバー名が既存のユーザ登録数プラス1の値となる。既存のユーザ名である場合は，Command2_Click（）のプロシージャから与えられたｉの値がそのままメンバー名として利用される。新規，既存それぞれのメンバー名のもとで，正答数の記録であるData（i）.item（h）に1が加えられ，正答数が増加することになる。Form2のフォーム上に問題番号とともに「正解！！」と表示される。Text1.Textに入力された答案がanswer（h）に等しくない場合，Label2のラベル領域に「誤り」と表示され，フォーム上に問題番号とともに正解が今回の試行の結果の記録として表示される。If文の終了後，Command2のコマンド・ボタンがフォーカスされ，新しい問題を表示する準備を整える。

２－３－４　Form2モジュールのCommand4_Click（）プロシージャのコード

```
Private Sub Command4_Click（）　'終了
  End
End Sub
```

Command4_Click（）のプロシージャは「終了」というキャプションがついたコマンド・ボタン

をクリックした場合に起動するものであり，これによってプログラムを問題の途中であっても，いつでも終了することができるようになる。ただし，問題を完了せずに終了した場合，データ・ファイルdata.txtは更新されない。

本プロジェクトのプログラムはコンピュータによるアンケートにも応用できる。質問項目を提示しても，すぐに答えずに次の質問項目にパスする場合がある。質問項目が最後まで進んだ時点で，飛ばした質問項目を点検し，再び提示したりするのにこのプログラムを応用することができる。また，反応を記録する方法については，たとえば，人格テストを行なって，その反応を分析してプロフィールを出す場合にも応用することができる。さらに，試行結果を記録として蓄積する方法は，学力の診断にも用いることができる。問題項目を広い範囲に及ぶものにするならば，どのような分野が得意でどのような分野が不得意なのかを判断することに用いることもできる。

注

1） ソフトはMicrosoft Visual Basic 6.０ Professio-nal Editionを用い，ハードはＮＥＣ系Pentium機を用いた。

参考文献

Microsoft Corporation 1997「Microsoft Visual Basic 5.0 ランゲージ リファレンス Part 1」アスキー出版局

Microsoft Corporation 1997「Microsoft Visual Basic 5.0 ランゲージ リファレンス Part 2」アスキー出版局

江藤潔　1997「Visual Basicで始めるプログラミング」講談社

川口輝久・河野勉　1997「Visual Basic 5.0 基礎編」技術評論社

川口輝久・河野勉1997「Visual Basic 5.0 コントロール編」技術評論社

中島省吾　1998「Visual Basic 5.0 入門」スパイク

門田幸太郎　1998「VISUAL BASICのプログラミング法１――その特徴とプログラミングの基礎――」立命館産業社会論集第34巻第３号 pp.119-136

門田幸太郎　1999「VISUAL BASICのプログラミング法２――配列データの操作――」立命館大学産業社会論集第34巻第４号　pp.167-187

門田幸太郎　1999b「VISUAL BASICのプログラミング法３――ファイルの操作；読み込みと表示の基礎――」立命館産業社会論集第35巻第１号　pp.1-13

門田幸太郎　1999c「VISUAL BASICのプログラミング法４――ファイルの操作；読み込みと表示の応用――」立命館産業社会論集第35巻第２号　pp125-141

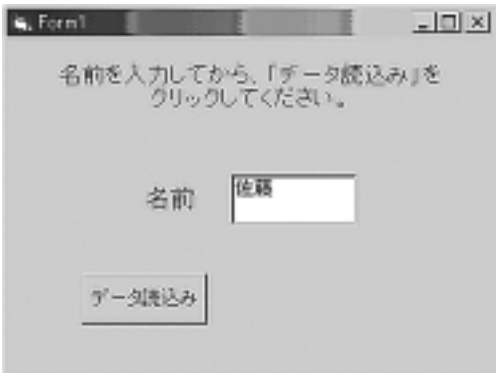


図4　ユーザー名入力のための実行画面

図5　問題開始の実行画面

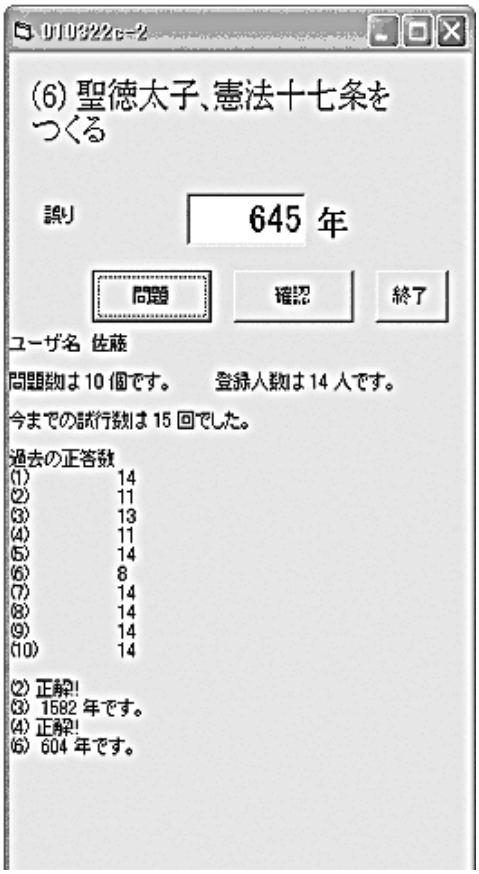


図6　課題の実行画面

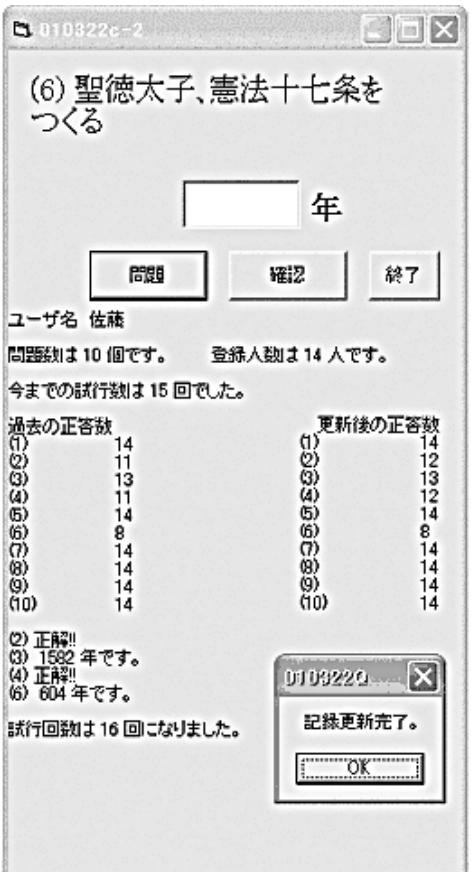


図7　記録完了後の実行画面

Responsive Programming Based on Visual Basic
Selection of Problems According to User Performance

MONDEN Kotaro＊

Abstract: I studied a programming method for selecting problems according to the characteristics of users. Users' characteristics are based on their prior performance. Those who had answered less than 10 sets of problems were given all the problems in a set regardless of their past performance. Those who had answered 10 or more sets of problems were given only the problems they did not satisfactorily answer, that is, their answers did not meet the predetermined standard. If a problem was answered satisfactorily, it was not given to the user again. Upon completion of each section, the performance of the user was updated. Visual Basic was the programming language chosen for my study.

Keywords: Visual Basic, responsive programming, selection of problems

＊Professor of the Faculty of Social Sciences, Ritsumeikan University