Excel でプログラムを書く

萩谷 昌己
（東大・情報理工）

入門的プログラミング教育のためにExcelプログラミングを提唱する．実際に，Excelを用いて素因数分解，エラトステネスの篩，ソーティング，動的プログラミングなどの各種のアルゴリズムを実装し，その可能性と限界を議論する．また，その経験に基づいて Excel に対する拡張の提案を行う．以上に先立って，Excel プログラミングを考える契機となった東大前期課程における情報教育について概観する．

背景と動機

東大駒場（前期課程）の情報に関する科目は二つある．一つは一年生夏学期の文理共通の必修科目である「情報」であり（川合慧監修「情報」東大出版会），この授業ではプログラミングは教えない．もう一つの「情報科学」は，一年生冬学期の理科のクラス指定（選択）の科目であり，情報の基礎概念に加えてプログラミングについても学ぶ．ここで，いうまでもなく，科目「情報科学」でどのプログラミング言語を用いるかというFirst Programming Languages問題が生じる．

どちらの科目も，学生の評判はすこぶる悪く，存亡の危機に瀕しているとは誇張のし過ぎであろうが，授業内容の改訂を余儀なくされていることは確かである．前者は特に文科からの批判が強く，後者は履修者の少なさに悩まされている．両者の最大の問題は授業内容が過多であることだが，学生の動機付けに失敗していることは否めない．

科目「情報」に関しては，高校における学習の仮定が正しくなかったことが問題点としてあげられているが，やはり，今の学生は実用志向であり，WordやExcelを教えてくれるものとの期待が強いことが大きな問題であろう．並行して開いた全学ゼミの「情報システム利用入門」は，「情報の授業より実用的」と評判がよかったとの報告がある．また，文科と理科で評価に明瞭な差があり，共通化の理想に走りすぎたという反省もある．

科目「情報科学」は，理科の学生を対象としており，他の必修科目との関係で，必修ではなく「クラス指定」という形で提供されている．この科目は，科目「情報」の内容を深化し，情報科学に関する部分を中心に，数値解析やパターン認識に関する入門も含めるとともに，離散数学や論理の基本概念も押さえることを目標としている．

情報の入門的な授業におけるプログラミングに関しては，二つの考え方がある．一つは，プログラム言語・プログラミングを通じて情報科学の基本概念や思考方法を身に付けるという考え方であり，もう一つは，情報科学の基本概念や思考方法を主とし，プログラミングはその学習の手立てとするという考え方である．科目「情報科学」においては，長い議論の末，後者の考え方でプログラミングを教えることとなった．

科目「情報科学」におけるプログラミング言語として何を採用するかに関しても長い議論があった．情報科学を学ぶための補助手段として，対話的な言語処理系が提供されていて授業中に計算結果を自分ですぐに確かめられること，自宅学習を奨めるために多くの種類の計算機上でただで使えること，最新のソフトウェア技術の導入という観点からオブジェクト指向の機能を有することが条件としてあげられた．そして，長い議論の結果，Rubyを（2006年度は部分的に）採用することとなった．ただし，極度にRubyに依存しないよう注意することとし，最終回に世の中の実際のプログラミングについての紹介（学ぶための方策）を教えるという方針を立てた．参考のために，科目「情報科学」の2006年度のシラバスを次ページの表に示した．

第 1 回 対象のモデル化とデータ構造（1）
　対象のモデル化，データ型，配列，レコード，オブジェクト
第 2 回 対象のモデル化とデータ構造（2）
　ハッシュ，リスト，木，スタック，キュー
第 3 回 データと計算（1）
　関数，条件分岐，関数の関数，代入，オブジェクト
第 4 回 データと計算（2）と有限オートマトン
　再帰，繰り返し，有限オートマトン
第 5 回 アルゴリズムと計算量
　ニュートン法，フィボナッチ数，ユークリッドの互除法，単純整列法，併合整列法，ビン整列法，計算量の求め方
第 6～7 回 数値解析
　実数データ表現，色々な誤差：丸め，桁落ち，情報落ち，打ち切り
　数値積分，常微分方程式，連立 1 次方程式，モンテカルロ法
第 8～9 回 パターン認識
　DP マッチング，隠れマルコフモデル
第 10～11 回 言語処理系と仮想機械
　字句解析，構文解析，コード生成，仮想機械
第 12 回 いろいろなプログラミング言語
　プログラミング言語の歴史，コンパイラ，インタープリタ，スクリプト，仮想機械
　命令型，宣言型，オブジェクト指向

2006 年度の反省点は以下のようである．上述したように，授業内容が過多であることに関しては，教えるべき項目を厳選することが必須である．また，プログラミングを情報科学の基本概念や思考方法の学習の手立てとするにしても，プログラミングにかける時間が余りにも少なかったことは反省すべきである．教えるべき項目を厳選しつつ，プログラミング演習の時間を増やすことが重要である．また，科目「情報」と同様に学生のモチベーションが低いことに関しては，情報以外の分野に関連した動機付け，たとえば，シミュレーション，実験データの解析を含めることが重要であろう．

さて，上述したように科目「情報科学」では Ruby を採用することとなったのだが，学生が実用志向であり，**Word や Excel を教えてくれるものとの期待が強く**，シミュレーション，実験データの解析などの情報以外の分野に関連した動機付けが必要であるならば，**いっそのこと，Excel でプログラミングを教えたらどうか**と考えた．これが本発表の動機である．Excel でプログラムを書けば，結果をすぐにグラフにすることができ（gnuplot などは要らない），Word や PowerPoint にすぐに張り付けることもできる．

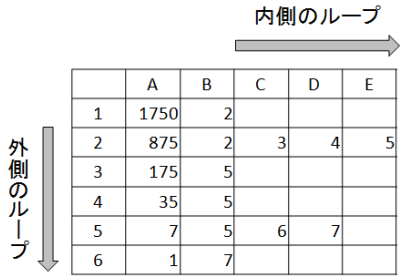
本発表ではまず，Excel プログラミングの提唱を行う．Excel プログラミングとは，Excel のセルに指定される式を用いたプログラミングのことであり，Visual Basic によるプログラミングのことではない．Excel プログラミングは制約指向プログラミングの一種であり，超純粋な宣言型プログラミングということができる．Excel プログラミングの実用性の評価を行うために，科目「情報科学」の題材を Excel で記述することを試みたので，本発表ではその報告を行う．そして，その経験に基づいて，Excel プログラミングの限界の確認しつつ，表計算に対する拡張への提案を行う．

Excel プログラミングの例

本節では，Excel プログラミングの考え方を説明するために，素因数分解の例を示す．右のような（Ruby の）プログラムを考える．これに相当する Excel プログラムの実行結果を次ページの左上に示す．

```
n = 1750
k = 2
while n != 1 do
      while n%k != 0 do k = k+1 end
      n = n/k
end
```

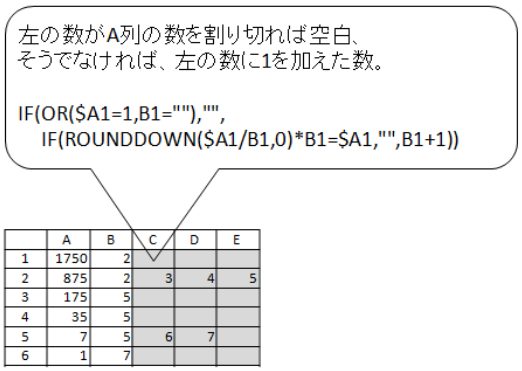
Excelプログラミングでは，基本的に，プログラムの各ループは，行か列の繰り返しに対応する．したがって，Excelプログラミングで自然に記述することのできる繰り返し構造は二



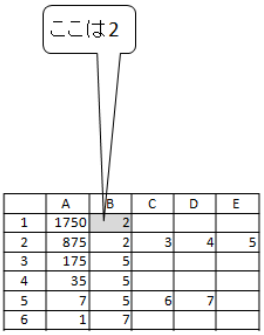
	A	B	C	D	E
1	1750	2			
2	875	2	3	4	5
3	175	5			
4	35	5			
5	7	5	6	7	
6	1	7			

重ループに限定される．しかし，以下に示すように，たとえ二重ループに限られていても，工夫すれば様々なプログラムを書くことができる．

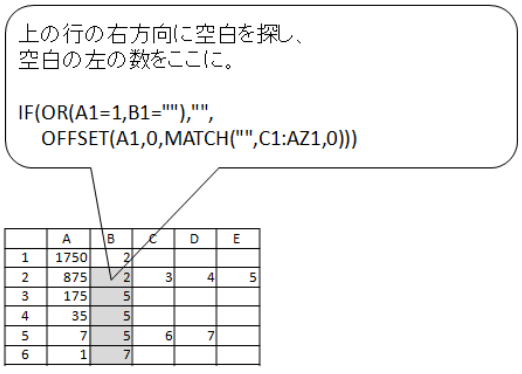
上の Excel プログラムは，セル A1 を入力としており，以下に示すように，四種類のセルに対して式を指定することによって記述されている．C1 から右下の領域のセルには，「左のセルの数が（同じ行の）A 列の数を割り切れば空白，そうでなければ，左のセルの数の 1 を加えた数」を表す式が指定される．セル B1 には定数 2 が指定される．B2 から下のセルには，「上の行の右方向に空白のセルを探し，空白のセルの左のセルの数」を意味する式が指定される．最後に，A2 から下のセルには，「上のセルの数を右のセルの数で割った結果」を表す式が指定される．



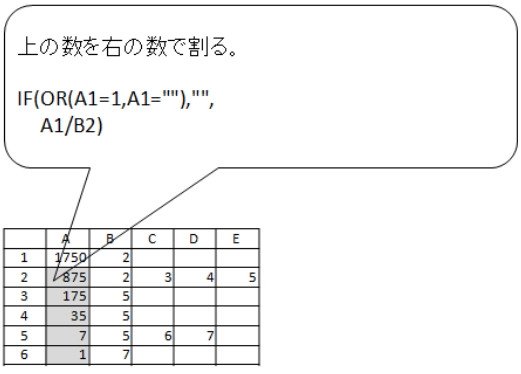
	A	B	C	D	E
1	1750	2			
2	875	2	3	4	5
3	175	5			
4	35	5			
5	7	5	6	7	
6	1	7			



	A	B	C	D	E
1	1750	2			
2	875	2	3	4	5
3	175	5			
4	35	5			
5	7	5	6	7	
6	1	7			



	A	B	C	D	E
1	1750	2			
2	875	2	3	4	5
3	175	5			
4	35	5			
5	7	5	6	7	
6	1	7			



	A	B	C	D	E
1	1750	2			
2	875	2	3	4	5
3	175	5			
4	35	5			
5	7	5	6	7	
6	1	7			

それぞれの種類のセルに指定する式は上図に示してある．ここで，Excelにおいて記号$は絶対参照を表すことに注意しよう．記号$は列と行のどちらにも指定することができる．たとえば，A1はセルA1を絶対的に参照している．$A1は列のみを絶対的に参照している．したがって，たとえばセルC1の式に$A1が含まれていて，その式をセルD2にコピーすると，$A1の部分は$A2に置き換わる．同様に，A$1は行のみを絶対的に参照する．A1は通常の相対参照であり，たとえばセルC1の式にA1が含まれていて，その式をセルD2にコピーすると，A1の部分はB2に置き換わる．

Excel が提供する関数のうち，本稿で特に重要な関数は MATCH と OFFSET である．たとえば，MATCH("",C1:AZ1,0) という式は， 領域 C1:AZ1（セル C1 からセル AZ1 までの領域）の間に空白文字列（""）を探し， その位置を返す． 上の例では 1 が返る． OFFSET(A1,0,…) という式は， セル A1 を起点として指定された行数（0）と列数（…）のオフセットのセルの内容を返す． したがって，

OFFSET(A1,0,MATCH("",C1:AZ1,0)))

という式は，「領域 C1:AZ1 の最も左の空白の左の数」を表す．この例や次のエラトステネスの篩の例におけるように，MATCH と OFFSET を組み合わせの表現力は意外に強力である．

Excel プログラミングは宣言的であるので，終了文というものはない．計算に必要ないセルには明示的に空白を埋める必要がある．上の例でも，適当な終了条件を検査し，

IF(OR($A1=1,B1=""),"",…
IF(OR(A1=1,B1=""),"",…
IF(OR(A1=1,A1=""),"",…

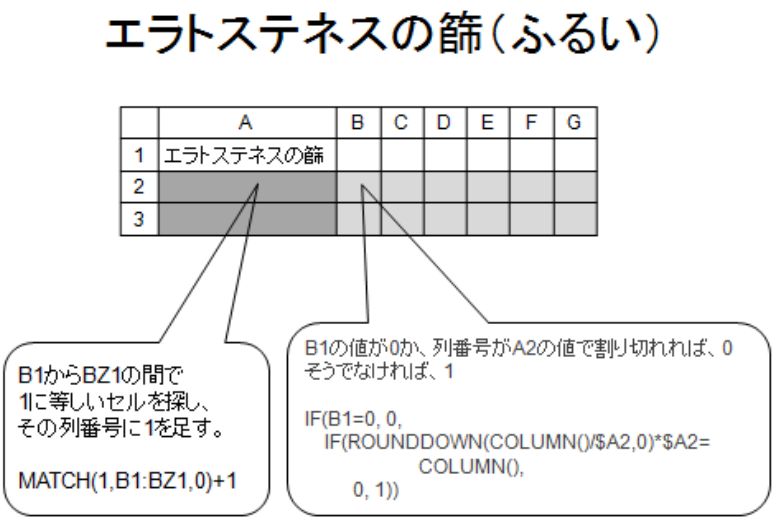
というようにして，空白を埋めるようにしてある．適切な条件を漏れなく入れるのは結構面倒である．

科目「情報科学」より

先に述べたように，以上のようなExcelプログラミングによって，科目「情報科学」の様々な題材のプログラムを書くことを試みた．具体的には以下のような題材である

- 簡単な繰り返し計算
  - 階乗関数・フィボナッチ数・2 の冪
  - コラッツの予想
  - 再帰的な階乗関数・ユークリッドの互除法
- エラトステネスの篩
- セレクションソート・並列バブルソート
- スタックを用いたフィボナッチ数
- オイラー法・オイラー法とルンゲクッタ法の比較
- モンテカルロ法
- 有限オートマトン
- 動的プログラミング
- 木構造

エラトステネスの篩のプログラムは右の図に示す．



二種類のセルに対して式を指定することにより，非常に簡潔にエラトステネスの篩のアルゴリズムを実装することができる．その結果を以下の図に示す．Excel ではこのように条件付き書式を用いて計算結果を視覚的に表示することが可能である．

エラトステネスの篩	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
2	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
3	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	1	0
5	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	1	0
7	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0
41	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0
43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

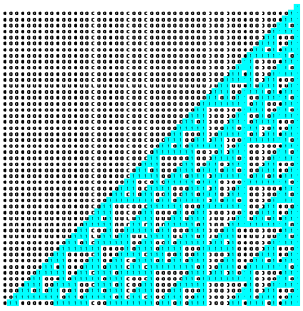
CALL	5						
NEXT	3	CALL	4				
NEXT	3	NEXT	2	CALL	3		
NEXT	3	NEXT	2	NEXT	1	CALL	2
NEXT	3	NEXT	2	NEXT	1	NEXT	0
NEXT	3	NEXT	2	NEXT	1	NEXT	0
NEXT	3	NEXT	2	NEXT	1	PLUS	1
NEXT	3	NEXT	2	NEXT	1	PLUS	1
NEXT	3	NEXT	2	NEXT	1	RETURN	1
NEXT	3	NEXT	2	PLUS	1	CALL	1
NEXT	3	NEXT	2	PLUS	1	RETURN	1
NEXT	3	NEXT	2	RETURN	2		
NEXT	3	PLUS	2	CALL	2		
NEXT	3	PLUS	2	NEXT	0	CALL	1
NEXT	3	PLUS	2	NEXT	0	RETURN	1
NEXT	3	PLUS	2	PLUS	1	CALL	0
NEXT	3	PLUS	2	PLUS	1	RETURN	0
NEXT	3	PLUS	2	RETURN	1		
NEXT	3	RETURN	3				
PLUS	3	CALL	3				
PLUS	3	NEXT	1	CALL	2		
PLUS	3	NEXT	1	NEXT	0	CALL	1
PLUS	3	NEXT	1	NEXT	0	RETURN	1
PLUS	3	NEXT	1	PLUS	1	CALL	0
PLUS	3	NEXT	1	PLUS	1	RETURN	0
PLUS	3	NEXT	1	RETURN	1		
PLUS	3	PLUS	1	CALL	1		
PLUS	3	PLUS	1	RETURN	1		
PLUS	3	RETURN	2				
RETURN	5						

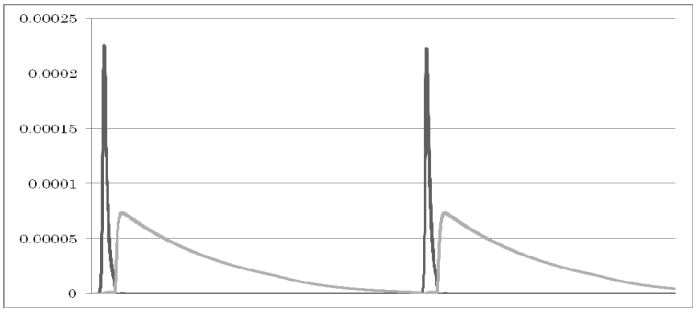
その他の例のいくつかについて簡単に触れる．ソーティングに関しては，セレクションソートの実装も行ったが，並列バブルソートは Excel プログラミングに非常にマッチしたソーティングアルゴリズムである．有限オートマトンの実装では，OFFSET を用いることにより遷移テーブルを簡潔に参照することができた．動的プログラミングでは二枚のシートを使った．一枚目のシートで表を計算し，二枚目のシートでバックトレースの計算を行っている．スタックを用いたフィボナッチ数の結果を左に示す．この例は Excel の処理系の限界に近く，式の書き方を間違えると，式のネスティングの深さの上限に抵触してしまう．具体的に，以下のような式が各セルに指定されている．

```
IF(OR(ISNUMBER(E4),ISNUMBER(C4)),IF(D4="CALL", IF(E4<2, E4,
E4-2),IF(AND(B4="CALL", NOT(C4<2)), C4-1,IF(AND(D4="RETURN", B4="NEXT"),
C4,IF(D4="NEXT", IF(F4="RETURN", G4, E4),IF(D4="PLUS", IF(F4="RETURN", G4+E4, E4),
"")))))),IF(E4="CALL", IF(F4<2, "RETURN", "NEXT"),IF(AND(C4="CALL", NOT(D4<2)),
"CALL",IF(AND(E4="RETURN", C4="NEXT"), "CALL",IF(E4="NEXT", IF(G4="RETURN",
"PLUS", E4),IF(E4="PLUS", IF(G4="RETURN", "RETURN", E4), "")))))
```

なお，再帰をどのように実現するかは Excel プログラミングの課題であり，本稿の最後でも簡単に触れる．

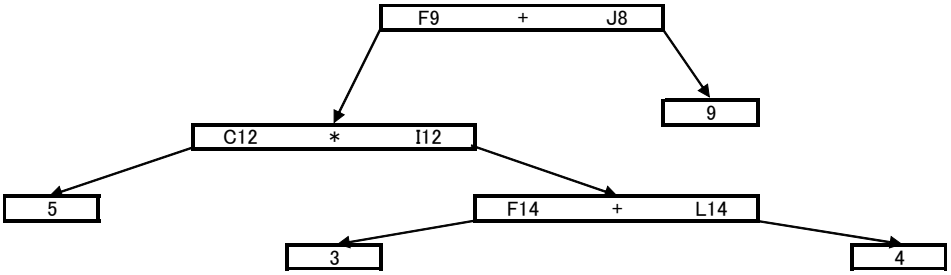
科目「情報科学」とは関係ないが，セルオートマトンやチューリングパターンのシミュレーションなども行ってみた．左下は有名な 110 番のエレメンタリセルオートマトンの実行結果である．右下は，BZ 反応のシミュレーション結果をグラフにしたものである．





条件付き書式を用いることにより，計算結果をこのようにきれいに表示することができる．また，右の例のように，かなり重たい数値シミュレーションも可能である．

科目「情報科学」に戻る．最後に下図のような木構造を扱うことを試みた．



二枚のシートを用いている．片方のシートでは，セルの名前を用いてポインタを表現している（線には意味はない）．もう片方のシートでは，関数INDIRECTを用いて木構造が表す式の値を計算した．正直のところ，このような方法で複雑なアルゴリズムを実装するのは難しいと思われる．

Excelの限界と拡張の提案

本節では，以上の経験に基づいて，Excelの限界について簡単に議論した後，Excelプログラミングの観点から，Excelの拡張について提案する．

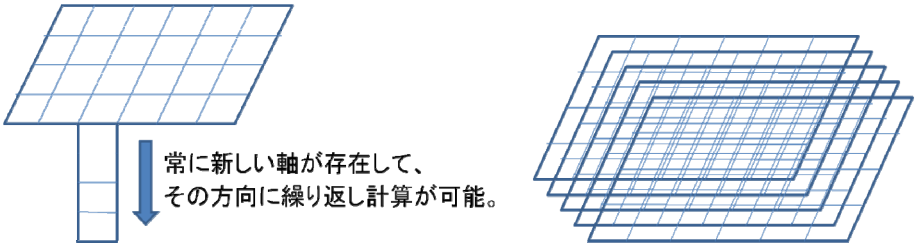
Excelでは，各セルの式は，そのセルのみに対する制約しか表現することができない．もちろん，Excelにとってこの簡潔さは重要だが，もう少し豊富な制約記述方法が欲しいと思った．たとえば，空白の大域的な条件などを書きたい．

また，Excelではセルのデータ構造が貧弱である．たとえば，一つのセルに数の組を書くことができれば，より多くのアルゴリズムを自然に表現することができると考えられる．

セルの式のコピーがマニュアルでしかできない．Excelプログラミングでは，ある領域のセルに同じ式を設定しておく必要があるが，これを手作業で行うのは煩雑である．また，あらかじめ式を設定しておく必要があるため，計算結果が設定された領域をはみ出してしまうこともある．何らかに仕組みで，計算の途中で必要に応じて動的にセルの式のコピーが行われると便利である．

基本的にExcelのセル空間は2次元のみである．複数のシートがサポートされているが，セル空間は3次元ではない．各シートは個別にしか扱うことができない．たとえば，行列の表示はできるが，行列の計算はできない．

以上のような考察に基づいて，いろいろな拡張の方向を考えることができるが，n 次元への拡張が最も魅力的であろう．動的に次元を拡張することができれば，再帰的な計算も可能になる．この場合，軸をどのように指定するかが大きな課題であろう．少なくとも，シートを加味して，3 次元のセル空間を実現してはどうだろうか．



終わりに

本稿の最後に，本稿のアプローチによるプログラミング教育が有効かどうかを考えたい．Excel によるプログラミング教育は，いわば，**プログラミング言語なしのプログラミング教育**であるということができる．プログラミング言語がなくとも，繰り返しや分岐などのプログラミングの基本的な考え方を教えることができる．また，以上で述べたように，各種の基本的なアルゴリズムについて教えることも可能であることがわかる．しかも，Excel という実用的な環境を用いることは，実用志向の学生を動機付けることになる．たとえば，プログラミングやアルゴリズムに関する基本的な概念の解説に交えて，偏差値の計算とか，実験結果の統計処理などを織り交ぜたらよいだろう．

Excel プログラミングによる教育が実際に可能かどうかについては，学生がセルに対する式の書き方を習得できるか，特に，相対参照と絶対参照を理解できるかがキーポイントであると考えられる．ただ，このポイントの理解は，Excel を実用的に使いこなす上でも必須であるので，注意深く動機付けを行いつつ丁寧に説明すれば，十分な理解が得られるのではないかと考えられる．

もちろん，Excel プログラミングはプログラミング教育の最初のステップに過ぎない．Excel プログラミングを通してアルゴリズム的な考え方に慣れることは，本格的なプログラミング教育への準備になるだろう．

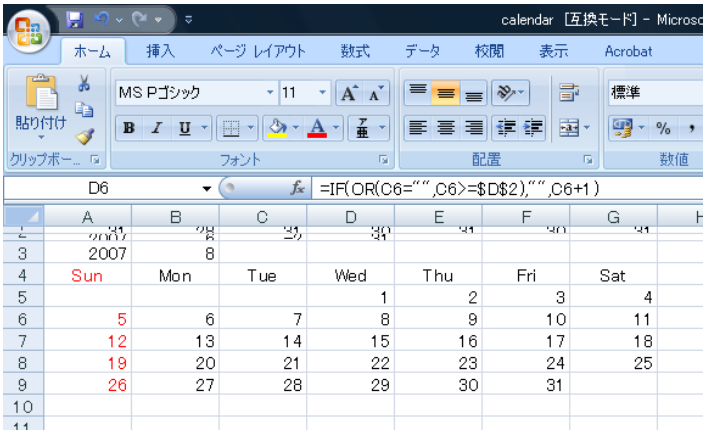
なお，最後に付け加えるが，実は最近では発表者も Excel プログラミングを日常茶飯事的に行っている．小規模の単純な計算は，プログラミングを書かずに，Excel を使ってしまうのである．

質疑応答

山下：本発表の例で Excel を使った気分になるのか．
萩谷：数値計算など，普通の Excel らしい例もたくさんある．
並木：昔，同様のことをセンター試験に際して考えたことがある．手でやらないといけないことがある．テクニックが必要である．普通のアルゴリズムに対してすっきり適用できないのではないか．
萩谷：たしかに，セルの式のコピーをマニュアルで行うは煩雑である．ただし，エラトステネスの篩や素因数分解のように，自然なものもたくさんある．もちろん，一般にアルゴリズムをセルの式に分解するのは難しい．
並木：Excel は現場での評判はよい．
萩谷：プログラミング言語全くなしでプログラミングの教育ができるのはよい．できる人はどんどんやるのではないか．
大日向：Excel で構造計算は現場でもやっている．2003 年に Excel は関数型言語だという Microsoft の論文がある．Excel は無料ではない．
伊地知：早稲田で Excel を教えているが評判はよい．かなりプログラミングをしている気になる．

最後の最後に

シンポジウム中に，本シンポジウムの課題であったカレンダーを実装したのでその証拠を残しておく．



	A	B	C	D	E	F	G
3	2007	8					
4	Sun	Mon	Tue	Wed	Thu	Fri	Sat
5				1	2	3	4
6	5	6	7	8	9	10	11
7	12	13	14	15	16	17	18
8	19	20	21	22	23	24	25
9	26	27	28	29	30	31	
10							