2．プログラム作成過程分析の心理実験とプログラミング・ツールについて

山梨大学計算機科学科　有澤　誠
Makoto Arisawa

1．はじめに

夏のシンポジウムのテーマに，「いかにしてよいプログラムを書くか」がとりあげられた．現状でのプログラム作成過程がどうなっており，どこを改める必要があり，どのような方向へ流れを変えるべきかを，調べてみることが，第一の目標である．その上で，望ましいプログラミング環境，特にプログラミングツールの姿を定めていこうという第二の目標をおく．

プログラム作成過程の分析の手法として，今のところ二つの方法をとっている．第一は，プログラマあるいはプログラムを日頃書く機会の多い人に対して，アンケート調査やインタビュー調査を行って，当人の口から分析してもらったものをまとめることである．第二は，プログラム作成過程を，声に出してもらって録音をとり，あとでその録音を分析するやりかたである．

本稿では，後者に関する報告をして，同じテーマに興味をもっておられるかたがたのための議論の材料にしたいと思う．

2．心理実験のあらまし

問題解決(problem solving)の過程を分析する心理実験として，A.Newell, H.A.Simonの著書[1]がある．被験者に"thinking aloud"をすすめ，そのようすを録音して分析している．材料としては，予備知識の影響を考えて，魔面算パズルや論理演算などが選ばれている．同一の問題に対して，複数の被験者の思考過程を比較している．

対象をプログラム作成にする場合，同様な心理実験を実施するには，いくつかの問題点がある．適切な被験者を得ること，適切な材料を得ることの二点が，特に厳しい．また，一般に"thinking aloud"ということ自体に，かなり無理な不自然さが入りこむ．

筆者のおかれた状況では，被験者として研究室の学生に依頼せざるを得ない．被験者の人数が限られるから，問題のほうを何通りも用意して，同一の被験者にくりかえして実験を行うことになる．学生間の情報の流通はきわめて速いから，同じ問題を二度使用することさえできない．

学生たちのプログラム作成能力は，必ずしも熟達したものでないから，問題の領域をかなり限定し，少くともその領域に関する予備知識は十分にもたせておかねばならない．計算機システムやプログラミング言語といった環境は，日頃使い慣れたものを使用させることが望ましい．

まず"thinking aloud"については，被験者が同一研究室の学生ということから，二名ずつを組にして，対話(dialogue)の形をとることにした．二人に共同で解くべき問題をもたせれば，自然な対話が行われるので，声を出すことに対する抵抗が少い．次に問題の材料としては，backtrackingを基調とするものに限定した．問題解決の過程そのものが一種のbacktrackingであることも関係がないわけではない．しかし，再帰的手続きの形で制御構造が統一的に扱えるために，目先は変っていても本質的には同一の問題になることから，多数の問題を用意しやすいことが大きい．

言語としては，構造的Fortran言語 Star[2]を用い，FACOM 230-45S (160KB) のOS-II上で動かす．Starは，再帰的手続きが使用でき，実行時プロフィールが出てくるので，この実験のための言語として，十分使用に耐えると判断した．

Knuthの文献[3]をbacktrackingの基本として選び，それをStarで記述したもの[4]について，事前に十分教育して予備知識とした．

プログラムの構造は，次に示すような手続きの形をとる．したがって，プログラム作成過程では，この形に問題を形式化するためのデータ構造と，その初期値設定のための手続き，および手続き中の述語Pに相当する論理関数を設定すればよい．Starはデータ構造に関しては弱体でFortranそのものであり，クラスの概念などは使えない．抽象的データ型 (abstract data types) に基づくプログラム作成過程とは異る結果が出るかもしれないが，こうした心理実験の初期の段階では，この点は障害にはならないと考えられる．

```
recursive procedure BT(K)
if (K=KLIMIT) then
    found
else
    I←1
    loop
    until (I=ILIMIT) do
        if (P(I,K+1)) then
            set(X(K+1))
            BT(K+1)
        fi
        I←I+1
    repeat
fi
```

3. 実験の中間結果のまとめ

本稿を書いている時点までに，6名の被験者の相異る組合せに対して，計6回の実験を行っており，近日中にさらに新しい組合せで3~4回の実験を実施する予定である．実験時間は60分で，録音はそこまでで打切る．しかしその後自宅でプログラムを完成させて，コーディング，テスト，デバッグの段階を経て，必ず完成させることになっている．

6回の実験で使用した問題は次の通りである．

(1) インスタント・インサニティ・パズルを平面化したような，4枚のタイルの重ね合わせパズル

(2) 頂点を2色に彩色した立方体4個を，ドミノ条件を満たすように2×2×1の直方体に並べるパズル

(3) きき手じの方向が北北東，東北東，東南東，…，北北西であるような駒についての，8-Queenパズルの変形．

(4) 3つの面にマグネットをうめこんだ立方体4個を，マグネットがひきつけあうように2×2×1の直方体に並べる市販のパズル．

(5) 3×3×3の3次元座標空間の頂点を表す27枚のカードを規則に並べるパズル．

(6) 1×1×3の直方体9個を組合わせて，各片につけてある印がサイコロの目のようになるように整える市販のパズル．(使用した片は手製のもの)

また，被験者の組合せは，U-V, W-X, Y-Z, LHW, V-Y, X-Z のようになっている．もともとの文献[3]がインスタント・インサニティ・パズルを例題にしていたので，はじめのほうの問題では，類似性の強いものを用意した．8-Queenについても，既にStarプログラムに関する話題をとりあげてあったので，予備知識をもっていると考えてよい．

6回の実験のいくつかでは，60分の間にプログラムの設計がかなりすすんだも

のもあり，またほとんど暗中模索の状態で終り，完成したプログラムがこの60分の議論とはまるで無関係な様相を呈したものもある．その中で，中間的な性格をもつ(5)の実験について，付録に詳細をあげておく．(例題のつもりである．)

付録1は，例題の説明，付録2は例題の場合の対話の一例(はじめのほう15分程度と，おわりのほう15分程度)，付録3は，その後完成した(はずの)プログラムリスティングである．

実験を行っていて気がついた点をいくつか挙げてみる．

- 学生の組合せによって，対話がスムーズにすすむこともあり，別々に考えこんでしまうこともある．十分考えを煮つめてから動きだすタイプと，むしろ早い段階から試行錯誤にのりだすタイプとがあって，うまいコンビになることもあれば，相性が合わないこともある．
- 学生の場合，たとえ留年生であっても，上級生とみなされて遠慮が生じる．(入学時から同期であれば，浪人などによる年令差は遠慮を生じない．) こうした場合，必ずしも対話がスムーズにすすまない．
- 問題を理解するための時間は，具体的な「もの」をよそおうか否かで大きく差がちる．パズル・ピースをよそえてあると，慎重派のタイプでも試行錯誤につぶれかすく，プログラムにもそれが反映してくるように思える．
- 実験の中心は，与えられた問題をどう理解し，データ構造としてどう表現するかである．StarやFortranの，3次元以内で正整数の添字しか許さない配列しか使えない，という制約がどのくらい影響をもつか調べてみたい．次年度には，あらかじめPascalプログラミングの教育をしておき，Pascalについて同様な実験を実施したい．(ただし，使用できる2つのPascal処理系には，一長一短があって，Starほどのavailabilityがない点が悩みである．)
- backtrackingという制御構造のわくを限定し，しかも毎回新しい問題を用意するという方法をとったため，どうしても問題の難易にバラツキが出る．60分前後でプログラム設計の見通しがたちそうなものを選んではいるが，被験者のコンビネーションや，学習履歴のようすによって，こちらの予想通りになかなか事がすすまない．
- 今回の実験では，被験者になってくれた学生の総合的な学力がほぼ一定の水準であったため，二人の内の一人がほとんど全部やってしまうといったケースは生じなかった．ただし，自宅でプログラムを書く段階では，たまたまーすが多忙になってしまったので，もう一人が大部分ひきうけたことはあったらしい．
- このような実験は，バッチ処理システムでの環境で実施するだけでなく，タイムシェアリング・システムでの環境でも実施して，比較検討することが望ましいと思う．将来，そのような方向も考えてゆきたい．

4. おわりに

今回報告する実験結果は，まだ系統だてたものでなく，統計的な処理を前提としたものでもない．今後，より系統だったものにしていくために，プログラミング・シンポジウムの席で，興味をもっておられるかたがたからのご意見をいただいて，参考にさせていただくことを期待して，報告のための時間をいただくことにした．この実験に関連したことについて，どんなことでもコメントがおありなら，いつでも筆者にご連絡いただければ幸いである．

末筆ではあるが，このような実験の必要性を示唆いただいた和田英一先生と，被験者および資料整理に協力してくれた，岩谷正宗，神田雄一，滝沢，中山久雄，薗井等，柳原博之の諸君とに，心からお礼を申しあげる．

本実験は，文部省科学研究費補助金課題番号 458615 の一部として実施している．

参考文献

1) Allen Newell, Herbert A. Simon: Human Problem Solving. Prentice Hall, Englewood Cliffs. NJ. 920+xiv pp. (1972)

2) M. Arisawa, M. Iuchi: Debugging Methods in Recursive Structured FORTRAN. Software-Practice and Experience Vol. 10 (1980) 掲載予定. (Yamanashi-CS-79-008)

3) D. E. Knuth: Estimating the Efficiency of Backtrack Programs. STAN-CS-74-442. Math. of Computation Vol 29, 121-136 (1975)

4) 有沢: 連載 FORTRAN プログラム ⑩ バックトラッキング. bit 11-10, 44-50 (1979)

5) Gerald M. Weinberg: The Psychology of Computer Programming. Van Nostrand Reinhold, NY. 288+XV pp. (1971)

6) B. W. Kernighan, P. J. Plauger: Software Tools. Addison-Wesley, Reading MASS. (1976)

7) E. Wada, K. Kakehi, M. Takeichi: An Analysis of Program Making. in W. M. Turski ed. Programming Teaching Techniques. North-Holland. (1973), 127-137.

8) 有沢: リレー連載・パズル ⑳ 三次元ゲーム. 数理科学 No.199 (1980-1) 掲載予定.

9) 牛島和夫: Fortran プログラミングツール. 産業図書. 東京. 241+viii pp. (1979)

付録1　例題の説明

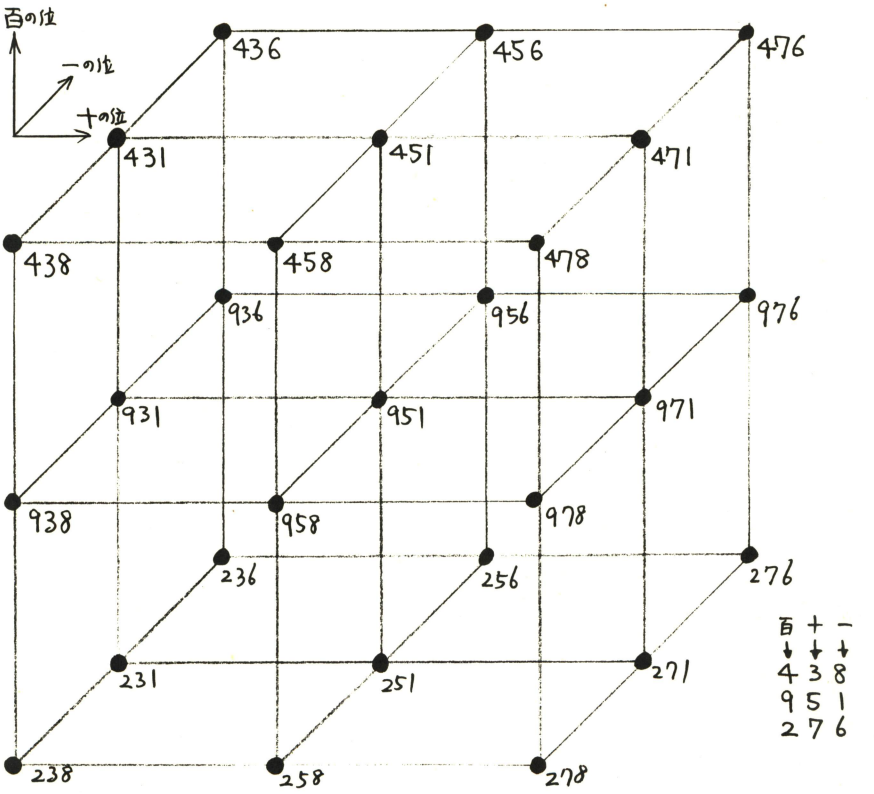
27枚のカードがある．どのカードにも相異る3ケたの数が書いてある．百の位の数字は4,9,2の3種，十の位の数字は3,5,7の3種，一の位の数字は8,1,6の3種である．従って，すべての組合せは3×3×3=27となり，それぞれに対応するカードが1枚ずつあるわけである．この27枚のカードの数は，図に示すように，3次元の座標空間に埋めこむことができる．

問題は，この27枚のカードを環状に並べて，どの隣り合う2枚のカードをとってき，3つの数字のうち2つの数字が同じになるようにすることである．

注意すべき点は，この問題は，図の格子をグラフとみたてたときの，Hamilton径路(閉路)を求める問題ではないことである．すなわち，たとえば図中では隣接していない436-476は，問題の条件では隣接しているとみなしてよい．従って格子点を2色にぬりわけて，解が存在しない証明をしてはいけないことになる．

なお，この問題に関しては，文献[8]がある．

実験に際しては，27枚のカードのほか，数字を書きこんでいない格子図を書いて与えた．また，文献[4]の中に出てくるプログラム・リスティングは，機能省めいめいが持参しており，文献[3]ともども，いつでもとりだして見ることができる．ほかに，メモをとるための大きな白紙が数枚用意してある．



付録2　例題の場合の対話の一部

（問題の説明）

A　ええと

B　6と4と2，7と5と3，8と6と1．組合せが27通りあるから，まずこのデータ構造を作るら．これだけでもできそうじゃん．

A　できそうな気もするな．

B　なんか楽にできそうじゃん．データは．

A　いやデータというのは．ん．これ直接使ってだろ．

B　うん．ええと，でこう2つ比べるとこうも．なんかうまくできそうじゃん．

A　ええと座標にしたらどうなるんだ．百の位3つだから，あそこに4,3,…

B　データを作るにはどうしたらいいのかな．

A　いやその前に，この座標ですよ．この座標とひと目盛というのは，どういう関係かな．

B　まあいいや．書いてみよう．

A　3つだら百の位だろ．まん中に十こうとう．27桁だからすべてを比べるんだけど，たとえば

B　たとえば

A　ここにちゃんとふっとばしていけばいいんだな．ええとこれは9．

B　9と4と2

A　この通りすればいいのかな

B　これをこの通りを9と4と2としていくの．この角を．

A　9の通り．

B　4の通り．2の通り．

A　うんそうだな．9,4,2．逆でもいいな．

B　どっちでもいいよ．7,5,3．

A　次が…　わからん．8か．

B　ああ，9,7,8で．

A　これが7，これが3，8,6,1，ええと．あここは，あれちょっとおかしいな．

B　向きはどうなっているんだ．

A　こういう面とこういう面とこういう面と．3つの面があるから，5とまずいんじゃないんすか．こっちが百だから2,4,9．あれ，ええと．

B　おかしいな．

A　7,5,3．8,6,1．だから．

B　つかみにくいね．

A　あ，これでいいのか．こっちが十だから，これが7,5,3．ちがう8,6,1．

B　それでいいじゃん．

A　よかった．だから，ここが表わすとしたら．

B　だからたとえば，ここはいくつだ．

A　4，5，8，な．4,5,8．　ころころ4,5,8．

B　4,5,8

A　9,5,8

B　9,5,8

A　このつながりが…　そうか，すべてつながりがあるのだが，これをたどっていけばよいのか．そうするとこれだと

B　たどっていくとか．ああそう，一筆書きだな．環になるちゃこんは一筆書きか．

A　だから．

B　そこ backtrack じゃん．そこ．

A　ここから始める．ここに座ってくればいいんだな．深さは27．へたすりゃ，スタック・オーバフローになってしまう．

B　27とは限らんじゃない．27か．27じゃないよ．深さちゃうのは3じゃん．3ったってちがうのか．3つのうち1つをちがってそっ合えばいい．それが深さになるんじゃない．

A　こっちにいって

B　解は何通りもあるんじゃない．

A　うん，そうだな．

B　そうすると出発点は決っているのかや．なんでもいいだら．

A　とにかくどこにおくか．組立てパズ

ル．

B　だから，データ構造作ったあと，比べていくんじゃないすか．

A　そりゃそうだ．

B　データ構造作って．だけんそれで解がいくつもでるかな．でるかな．でるか．

A　まず第一段のデータは，こいすべて27だろ．

B　で，あと出発点見つけてさ．あ，やっぱ深みは27か．こうやってって．で，あとここらへんで止まっちゃうこともあるな．

A　あるね．あ，ないか．

B　いや，こうきて，こうきて，こうきて．とまっちゃうじゃん．止まらんかな．どうなるかな．ここととここはくっついているから，

A　4と7とよ，2と7とよ

B　だから．

A　これるな．

B　これるか．前に一回来たものは，なんか覚えて，覚えてないと困るじゃんね．

A　うん，そうか．

B　なかなか，なかなか．

A　ちょっと単純じゃないな．

B　はじめは単純と思ったけど．

A　ここにたどりなくなってしまう．

B　だからこれなんか覚えておかなきゃ．FALSE，TRUEで，ここへ来ちゃだめだと．

A　うん，いえるな．

B　そいで，ここだけは，出発点だけは一番最後にきてTRUE．だからそれをFunction文でやっておけばいいのかな．

A　点なんだな，要するに．

B　あっ，だけどさ．もしかしてさ．こうたどっていく場合，次にくるもんが定っているから．

A　3つしかない．

B　そうね．それは配列で定められるかな．無理かな．

A　データでいれると．

B　うん，データでいれる．うまくいんなきゃね．

A　次はこっちと．9，5，8で4つある．

B　あ，4つあるね．

A　3つのところもある．

B　うん．あ，全部4つじゃない．

A　ここら辺じゃん．

B　うえ．ここに隠れているんだもん．

A　だってわかれないよ，直接．

B　あ，そうかそうか．かどか．かどは3つか．

A　かどは3つで，こういうところは4つか．

B　なんかやりにくいね．

A　そうすると．

B　かどが3つで，他のところが4つか．じゃかどは，その他のものを判別する何かいるやんね．

A　まず

B　かどはどうなっているんだ．座標は2，7，8．

A　になるけど，この位置を変えたらまた変ってくるんだよ．

B　位置を変える．あっそうか．

A　2，4，9だろ．9，4，2にしたら．

B　結局この考えるとめんどくなっちゃうじゃない．何か．だから頂点が何通りもあるんだろ．位置を変えたら狂ってくるのか．何かこれを考えると複雑．

A　27枚だから，あのデータ27枚だからそんなにでてくるわけ．9，5，1．9，5，1というとどこだ．この．ん

B　ん．うえ．

A　9だろ，5だろ，ここだ．

B　それにくっつくのな，9，5，1．…2，5，1というのか．2，5，1．

A　2，5，1　　とい，たらおかしいな．

B　どこだ，いったい2，5，1は．

A　下の段の，一番向こう．

B　ここか．だからくっついているよ．

（中　略）

A　まずはデータ．
B　データはすぐできそうだと思う．うん，とりあえずMにデータ1～27までいれるら．
A　小さいほうからかな
B　どっちからかな．データ．
A　いやあ．問題は．
B　何．
A　いいや．まずデータ作ろう．
B　データつくる．これでいいのかな．
A　うん．いや，その作りかたはあとにして，一応Mにはいっているものとして．
B　はいっているものとして，じゃ何やるんや．
A　Functionを作ればいいんだ．
B　Functionを作る．Functionは，ううん2ついっしょちゅうことは，3次元にするとこれは楽になるんだけどな．2次元，あ1次元だと難しいな．Functionえで…．たとえば9,5,1と9,3,1比べるときはどうすればいいのかな．やっぱmod，mod．まず10で割って余り1，また10で割って余り5とる，また10で割って余り…できるら．10で割って余りを比べていけば
A　うん，できるな．問題は何かるしかしな，うんと．
B　何を考えているの．
A　解けるかどうか．
B　解けるかどうか．
A　いやいやそうじゃなくて．一応Functionえいるわけか．
B　え．で2ついっしょだったらTRUEで，ちがっていたらFALSEで返せばいい．もうひとつ．それを，いままで使ったものはどうしておけばいいのかな．いままで使ったデータは，消しとけばいいのか．それかTRUE FALSEでやっておけばいいのか．何を考えているだ．
A　もどれるのか
B　もどれるってどう意味．
A　いや，だからバックトラックできるか．
B　できるんじゃないかと思う．ここでMCが1だったら，はいってはいって1じゃなかったら
A　まずこれを使って，だめだったらば使える状態にして戻ってくるんだろ．前に使ったのまたなるんじゃないのか．
B　え，どうして．もう一回言ってみ．
A　あれ，いいのかな．ひっかかるな．
B　ひっかかっているから直ま人．もう一回いってみ．
A　こうだろ，こう帰るだろ．
B　あまり関係ないんじゃない．3つか4つか比べりゃいいだかよ．
A　だからこれを使用して．
B　使用していってだめだったら．
A　だめになるというのはどういうこと．具体的に．
B　前に使ったのに戻っちゃうとかさ．
A　なくなっちゃうということは，操作していって27個の中に含まれていない．
B　27個に含まれてこない．そんなことないだろ．
A　いや．だから27個だと27個のデータとっとくわけだろ．
B　あ，27個で残りの中にはいっていない．
A　そうだろ．
B　そういうとき戻ればいい．どこまで戻るか．できる．…と思う．
A　そうすると，どれを使ったのかということは，何かとっておかなくちゃだめだ．
B　そうだよ．だから0と1とか．
A　0,1．
B　TRUEとFALSE．Function2つにするのかな．ちがうかな，それともとっときゃいいのかな．あ，0にしておけばいいか．一度使ったのは0．…いいじゃない．

A　や，それと2つの意味の
B　2つの意味の
A　だから要するにこれは．
B　まるっきり使わないんだら，1回使ったらな
A　いやそうじゃなくて，その添字だけでやるんじゃないか
B　おお，添字だけ並べればいいだから
A　だからこれをとっておく必要もない．
B　あれ，難しい．
A　ちがう．難しくない．要するに順列生成だよ．
B　順列か．
A　要するに27枚全部使ったのが解になるんだろ．27枚．
B　やっぱり順列か．
A　配列の添字のパターン27個並べて，前に出て…
B　順列というわけにもいかんから組合せか．何通りもできちゃうな．いいかな．
A　うん．何通りもできるけど，そのうち役のある．
B　ものもかなりある．
A　そう．だからいいんだ．
B　本質的に順列とはちがう．
A　だからさ．作ると．
B　まあいいですよ．Functionが楽にできそうだな．
A　Functionの形態を決めればいいんだ．Function．
B　このDOループの中でやればいいよ．これでいいよ．ここだけ変えればいいじゃん．こことあとL1とL2のとこだけ．
A　要するにメインをなんだ．先に，ちがう，これを先に．バックトラックのサブを作る．
B　そこはいっしょじゃないの，ほとんど．ここだけちがって，あといっしょじゃないんですか．
A　ちがうよ．
B　ちょっとはちがうけんさ
A　うんとおけるかどうか．
B　やっぱいっしょだよ．これと．これか．いらないでいっしょじゃないんですか．
A　ええと，MCの添字をひとつか．そうすると．
B　うん，添字ひとつと．
A　MC(I)と
B　だけん，2つにしたほうが便利のように．まあいいか．いいよ．
A　おけるかどうかだろ．いいんだろ．おけるかどうかで．
B　ほとんどこれといっしょ．
A　そうするとMCは．
B　だからこれはDOのJ=1,3　まあL1とL2にいれるとして，ううとMC=MC+1と，うん．
A　こんどは2が好だろ．それをMC+1．
B　OD　IF(MC.EQ.2) THEN
A　なら1とする
B　これでいいら．ELSEだったらなんとか＝FALSE．こんなふうにしてやればいいんじゃないんすか．
A　その前に，ええとその配列自体がもうとられたものかどうか．
B　おお，それもいるんだ．
A　なんでとっときゃいいんだろう
B　いやその1回使った配列ね．たとえば1,3,1とはおぼえていなきゃならないからさ．ほうだら2次元使ったほうが楽じゃないかな．あ，2次元じゃなく…あ2次元かな．こうやってMC(1,ロ)あ，わからん
A　いっしょだろ，要するにいっしょ．
B　あ，もうひとつ配列いるのかな．
A　だからこっちでやる場合にとれるかどうか．関数においては．
B　もうひとつFunctionいるのかな．
A　いらんて．答えはXにいれて．

――――(ここまでで1時間経過．録音はここで打切り．残りは自分たちの部屋へ戻って続け，プログラムを完成し，テストランさせた．)――――

付録3 完成した Starプログラム

```
FACOM 230 OS2/VS     STRUCTURED FORTRAN                    V-02 L-01
    EXECUTIONS                SOURCE     LIST

                    0001                EXTERNAL BT
                    0002                COMMON M,X,NN,SKIP,LL,II
                    0003                INTEGER M(27),X(27),NN(5),SKIP
                    0004                LOGICAL LL(27)
                    0005                DATA X/27*0/
                    0006                DATA NN/2,5,20,200,500/
                    0007                DATA LL/.FALSE.,26*.TRUE./
                    0008                DATA SKIP/0/
                    0009                DATA II/0/
              1     0010                CALL TAB
              1     0011                X(1)=M(1)
              1     0012                RECCAL BT(1)
              0     0013                WRITE( ' END')
                    0014                STOP
                    0015                END

TOTAL         3


          FACOM 230 OS2/VS     STRUCTURED FORTRAN                    V-02 L-01       DATE

ECUTIONS                  SOURCE     LIST

     9916       0001                RSUBROUTINE BT(K)
                0002                COMMON M,X,NN,SKIP,LL,II
                0003                INTEGER M(27),X(27),NN(5),SKIP
                0004                LOGICAL LL(27)
                0005                RDEFINE I,K
     9916       0006                IF(K.EQ.27)
      392       0007                THEN
      392       0008                  IF(MC(1,27).EQ.1)
       50       0009                  THEN
       50       0010                    II=II+1
       50       0011                    WRITE(1HO,I7,' FOUND ',27I4)II,(X(I1),I1=1,27)
       50       0012                    IF(II.EQ.50)
        1       0013                    THEN
                0014                      STOP
       49       0015                    FI
       49       0016                    SKIP=0
      391       0017                  FI
     9524       0018                ELSE
     9524       0019                  I=2
   256797       0020                  LOOP
   256797       0021                  UNTIL(I.GE.28)
   247299       0022                    IF(LL(I))
    39029       0023                    THEN
    39029       0024                      IF(MC(I,K).EQ.1)
     9915       0025                      THEN
     9915       0026                        X(K+1)=M(I)
     9915       0027                        LL(I)=.FALSE.
     9915       0028                        RECCAL BT(K+1)
     9889       0029                        X(K+1)=0
     9889       0030                        LL(I)=.TRUE.
    39003       0031                      FI
   247273       0032                    FI
   247273       0033                    I=I+1
   247273       0034                  REPEAT
     9889       0035                FI
     9889       0036                RECRET
                0037                END

1739343
```

このプログラムは，主プログラムと，backtracking用サブルーチン BT(K)，データの初期設定用サブルーチン TAB，および臨機条件の判定用関数 MC(I,K) からなる．276-271-208-258-256-251-231-236-238-438-938-958-978-976-971-471-451-951-956-938--931-431-436-456-458-478-476 を第1番目の解として印刷し，以下50番目の解まで印刷して停止している．プロフィールはその時点のものである．実行時間(CPUタイム)は 1分2秒余りである．

```
                    FACOM 230 OS2/VS     STRUCTURED FORTRAN                V-02 L-01

      EXECUTIONS                    SOURCE     LIST

               1      0001                SUBROUTINE TAB
                      0002                COMMON M,X,NN,SKIP,LL,II
                      0003                INTEGER M(27),X(27),NN(5),SKIP
                      0004                LOGICAL LL(27)
                      0005                INTEGER P(3,3)
                      0006                DATA P/2,9,4,7,5,3,6,1,8/
               1      0007                IA=0
               3      0008                DO KA=1,3
               9      0009                  DO KB=1,3
              27      0010                    DO KC=1,3
              27      0011                      IA=IA+1
              27      0012                      M(IA)=P(KA,1)*100+P(KB,2)*10+P(KC,3)
              27      0013                    OD
               9      0014                  OD
               3      0015                OD
               1      0016                WRITE(27I4//)(M(IB),IB=1,27)
               1      0017                RETURN
                      0018                END

TOTAL        136


                    FACOM 230 OS2/VS     STRUCTURED FORTRAN                V-02 L-01

      EXECUTIONS                    SOURCE     LIST

           39421      0001                INTEGER FUNCTION MC(I,K)
                      0002                COMMON M,X,NN,SKIP,LL,II
                      0003                INTEGER M(27),X(27),NN(5),SKIP
                      0004                LOGICAL LL(27)
                      0005                INTEGER COUNT
           39421      0006                COUNT=0
           39421      0007                MC=0
           39421      0008                L1=M(I)
           39421      0009                L2=X(K)
          118263      0010                DO JJ=1,3
          118263      0011                  IF(MOD(L1,10).EQ.MOD(L2,10))
           43237      0012                  THEN
           43237      0013                    COUNT=COUNT+1
          118263      0014                  FI
          118263      0015                  L1=L1/10
          118263      0016                  L2=L2/10
          118263      0017                OD
           39421      0018                IF(COUNT.EQ.2)
           12887      0019                THEN
           12887      0020                  L3=IABS(M(I)-X(K))
           12887      0021                  KK=1
           58581      0022                  LOOP
           58581      0023                  UNTIL(KK.EQ.6.OR.MC.EQ.1)
           45694      0024                    IF(L3.EQ.NN(KK))
            9917      0025                    THEN
            9917      0026                      MC=1
           45694      0027                    FI
           45694      0028                    KK=KK+1
           45694      0029                  REPEAT
           12887      0030                  IF (MC.EQ.0)
            2970      0031                  THEN
            2970      0032                    IF(SKIP.EQ.0)
              48      0033                    THEN
              48      0034                      MC=1
              48      0035                      SKIP=1
            2970      0036                    FI
           12887      0037                  FI
           39421      0038                FI
           39421      0039                RETURN
                      0040                END

TOTAL     1504681
```