# テキストデータベース管理システムSIGMA第2版について

有川, 節夫
九州大学理学部基礎情報学研究施設 | 九州大学総合理工学研究科情報システム学専攻

篠原, 武
九州工業大学情報工学部知能情報工学教室

武谷, 峻一
九州大学工学部中央計数施設

大島, 一彦
福岡大学理学部応用数学教室

他

https://doi.org/10.15017/1468164

出版情報：九州大学大型計算機センター広報. 20 (6), pp.517-581, 1987-11-25. 九州大学大型計算機センター
バージョン：
権利関係：

# テキストデータベース管理システム SIGMA第2版について

有川 節夫[1),5)]，篠原 武[2)]，武谷 峻一[3)]，大島 一彦[4)]，原口 誠[1),5)]，白石 修二[4)]
酒井 浩[5)]，宮原 哲浩[5)]，山本 章博[5)]，井上 仁[5)]，川崎 洋治[5)]，湯浅 寛子[5)]

目 次

1 はじめに……518
2 SIGMAの使用例 ……520
2.1 起動と終了，簡単なファイル操作……520
2.2 古今和歌集の検索 ……527
2.3 文献型データの検索 ……530
2.4 文献型データにおける並べ換え……534
2.5 索引，逆引き辞書の作成 ……535
3 SIGMAの利用法 ……538
3.1 キー入力に関する注意……538
3.2 LOGファイルの利用 ……541
3.3 TSSコマンドの呼び出し……545
3.4 ファイルの操作 ……545
3.4.1 領域と区域 ……545
3.4.2 MEMO区域とSIGMA区域 ……547
3.4.3 作業区域，LOG区域，索引区域 …547
3.4.4 バックアップ区域……548
3.4.5 SIGMA領域の作成・消去・アクセス権の設定，領域の拡張 ……549
3.4.6 外部領域……550
3.4.7 ファイル名規約 ……550
3.4.8 パスナンバーの設定と解除 ……551
3.4.9 ワイルドカードの使い方 ……552
3.4.10 COPYとMOVEの違い ……554
3.4.11 フロッピーディスクとSIGMA間でのファイルの送受信 ……555
3.5 検索コマンド(SEARCH)の機能 …555
3.5.1 特 長 ……555
3.5.2 区切り語と仮想レコード ……556
3.5.3 キーワード……556
3.5.4 論理式 ……557
3.5.5 検索件数の表示とファイルの指定 ……557
3.5.6 検索の原理と論理式の評価……558
3.5.7 否定の取り扱い ……559
3.5.8 検索結果の表示と再ファイル化(REFILEコマンド)……559
3.6 ソートコマンド(SORT)の機能 …560
3.6.1 特 長 ……560
3.6.2 パタンとパタンリスト ……561
3.6.3 切り出し仕様 ……562
3.6.4 再ファイル化仕様……565
3.7 置き換えコマンド(REPLACE)の機能……565
4 コマンドレファレンス ……567
5 おわりに……578
参考文献 ……578
付 録 ……580
コマンド一覧 ……580
PROFILEサブコマンド一覧……581

昭和62年9月29日受理

1)九州大学理学部基礎情報学研究施設，2)九州工業大学情報工学部知能情報工学教室
3)九州大学工学部中央計数施設，4)福岡大学理学部応用数学教室
5)九州大学総合理工学研究科情報システム学専攻

## 1. はじめに

パソコンやワープロの急速な発達に伴って，われわれの身のまわりには機械可読な文書が大量に作られ，テキストファイルとして蓄えられるようになっている．このようにして集積されるテキストファイル（の山）をもっと有効に活用し，また，ワープロを打つ感覚で手軽に（数メガバイト程度の）大規模データベースを作成したいという要望もでてきている．

著者らは，1981年に SIGMAという名前の情報検索システムを開発し，九州大学大型計算機センター（九大センター）で公開している [3,4]．情報検索システムといえば，INSPECデータベースを高速に検索する AIR[21]が著名であるが，SIGMA はこれとは違い，陽には構造を持たない一次元の文字列データ，すなわち，テキストファイルを対象としたものであり，上で述べた要望を満足するものである．大規模な文章（たとえば，ある作家の全集を機械可読にしたもの）を対象にした語の用法の調査や，数万件程度の文献データベースの構築・検索を簡単に行うことができる．

日本語テキストの増大や，テキストファイルの大規模化・大量化に対処するために，著者らは最近 SIGMA を全面的に改訂し，第２版を九大センターの FACOM M-780 OSIV/F4 MSP上に実現した[8]．SIGMA に取り込めるデータセットは，テキストファイルであれば，拡張子やレコード形式は何でもよい．また，フロッピーディスクのファイルも簡単に取り込むことができる．

SIGMA は，ファイル全体を一度だけ先頭から一字ずつ走査するという一方向逐字処理に基本を置いており，文献データベースの作成や検索，自然言語の解析などに活用されている．また，SIGMA は，高度の汎用性を有しており，多数のテキストファイルを管理・検索・編集する機能を持っているため，システムが管理するファイルの総体は，データベースとして機能する．このようなデータベースをテキストデータベースという[10]．この意味で SIGMAはテキストデータベース管理システムであるといえる．

今回の改訂の主要な点は，日本語テキストへの対応とファイルシステムの改良である．SIGMA は，ファイルの高速処理および共有データベースの管理のため，独自のファイルシステムによってファイルを管理している．そのファイルシステムの改良により，大規模なテキストファイルを効率よく維持できるようになった．SIGMA 第２版では，1バイト文字（半角の英字など）と２バイト文字（漢字など）が混在する日本語テキストをそのまま処理する[20]．よって，日本語処理のためのオーバーヘッドは特になく，漢字を含まないファイルの処理が遅くなったわけではない．漢字を含むファイルも，含まないファイルも同じ速度で処理することができ，第１版に比べて，処理は大幅に高速化された．例えば，SEARCHコマンドは１CPU 秒で２メガバイト（全角で百万字）程度のファイルを逐字検索することができる．今回の改訂により，大規模テキストデータベースの管理や，自然言語の解析，文献検索などへの応用範囲が拡大される．

(1)SIGMAシステムの特徴

SIGMA システムは，文献の蓄積・検索，論文の作成，自然言語の解析，書類の整理，その他データの収集・加工など，研究者の日常的な活動を支援する目的で開発した研究者向き情報システム[2,14]である．現在では上で述べたように，テキストデータベース管理システムとして位置づけられるものである．SIGMA における基本的なデータ処理方法である一方向逐字処理および文字列パタン照合アルゴリズムについては，文献[1,5,6,7,9,15]などを参照されたい．

また，個人用のデータベース以外に，グループで共有するためのデータベースを構築することもできる．さらに，くずかごの考え方によるファイルの保護や，ファイル化された交信記録をコマンドプ

ロシジャとして実行する機能などを持つ.

(2)利用例

SIGMA は，きめの細かい検索，特に自然言語の解析などに威力を発揮している．たとえば，九州大学教養部の樋口忠治教授は，S.Fischer書店刊Thomas Mann全集全13巻（約1万1千ページ，約40万行）を入力したテキストデータベース「トーマス・マン・ファイル」（約26メガバイト）を作成し[10,11,13]，種々の文脈中における語の用法の解析に活用している [12]．また，和歌集[19]を図1のような形式でテキストファイル化しておけば，様々な検索を容易に行うことができる（2.2 参照）．

九州大学農学部昆虫学教室の多田内修助手は，1900年以後の国内の主要昆虫学雑誌について，各文献中の昆虫の分類単位を一つのレコードとした「昆虫学データベース（KONCHU）」（現在，約1万2千件）を構築し，種々の検索に役立てている[23]．

トーマス・マン・ファイル及び昆虫学データベースの詳細については，それぞれ，本広報別稿[13]，[23]を参照されたい．

文献型データベースを構築するときには，図2のような形式のテキストファイルを，エディタかワープロで作成して，SIGMA システムへ取り込むだけでよい．従って，データベースの作成・維持は極めて容易であり，パソコン（と時間か予算）さえあれば，個人で数万件程度の文献データベースを構築し，他の利用者に提供することも簡単にできる．しかも，検索は逐字的に行うため，転置ファイル方式の情報検索システムで用いられているストップワード（キーワードに指定できない語）を特に設定する必要はないので，検索の精度は高い（2.3 参照）．よって，標題にofが4回以上現れる文献を検索することなども可能である．また，ファイルを一度読む間に複数の質問を同時に処理できるため効率はよい．

(3)本稿の構成

本稿では，SIGMA システム第2版の使用法を説明する．2章では具体的な使用例を中心にして SIGMAの使用法を簡単に説明し，3章で詳しい使用法を述べる．4章はコマンドレファレンスである．なお，豊富な使用例と包括的な解説をのせたマニュアルも作成して，SIGMA システムのファイル（S.'A71095B.MANUAL'）にしている．SIGMA システムから参照されたい．

```
@
        (T)ふるとしに春たちける日よめる$                (A)在原元方$
#0001¥年の内に春はきにけり　ひととせをこぞとやいはん　ことしとやいはん$
@
        (T)春たちける日よめる$                          (A)紀貫之$
#0002¥袖ひぢてむすびし水のこほれるを　春立つけふの風やとくらん$
@
```

図1．　テキストファイル（和歌）

```
$
(TI) Efficient String Matching: An Aid to Bibliographic Search
(AU) Aho,A.V., Corasick,M.J.
(SO) Comm.ACM,Vol.18,pp.330-340,(1975)
$
(TI) テキストデータベース管理システムＳＩＧＭＡ第２版について
(AU) 有川ほか
(SO) 九州大学大型計算機センター広報 Vol.20,No.6,(1987)
$
```

図2．　テキストファイル（文献型）

## 2. SIGMA の使用例

本章では，SIGMA の使用法について，例を中心に説明する．以下本稿では，利用者の入力の部分には下線を施して表す．利用者による入力が復改のみである場合は，記号↙を用いて表している． SIGMAでは，プロンプト（入力を促すメッセージ）に対して復改のみを入力すると，実行が次の状態へ移り，次のプロンプトが出される．説明のために左に番号をつけ，実行例の表示の省略は……で表した．

### 2．1 起動と終了，簡単なファイル操作

使用例１（TSS セッションの開始から終了まで）

```
    JCET005 SYSTEM READY
(1) LOGON TSS A79999A
    + PASSWORD ?
    password
    KDS70001I A79999A  LAST ACCESS AT hh:mm:ss ON yy.ddd
    KEQ56455I A79999A LOGON IN PROGRESS AT hh:mm:ss ON month day, year
    JOB NO = TSUnnnn CN(01)
    KEQ56951I NO BROADCAST MESSAGES
    READY
(2) SIGMA
     THE CREATION OF SPACE IS STARTED.
     THE NUMBER OF BLOCKS MUST BE FROM 10 TO 4776 .
(3) NUMBER OF BLOCKS:=50


    NUMBER OF BLOCKS      =   50

    NUMBER OF WORK   FILES=   20
    NUMBER OF LOG    FILES=   20
    NUMBER OF INDEX  FILES=   20
    NUMBER OF BACKUP FILES=  900
    MASTER KEY:=9999


        1144000 BYTES ARE AVAILABLE.

    THE CREATION PROCESS IS COMPLETED.
(4) SIGMA)KEYIN
     TO EXIT TEXT INPUT MODE, TYPE ;;; AT THE PROMPT K).
(5) K)私は，今日初めてSIGMAを使いました．
    K)とても速いので感動しました．
    K)私もテキストデータベースを作ろうと思います．
    K);;;
(6) DO:?
    FOLLOWING COMMANDS ARE AVAILABLE:
       CATENATE, COMMENT, COPY, DDIRECTORY, DELETE, DIRECTORY, ECHO,
       END, GET, HELP, HEXDUMP, KEYIN, LIST, LOAD, LOOK, MOVE, NICKNAME,
       PASSNUMBER, PROFILE, PUT, REFILE, REPLACE, SAVE, SEARCH, SORT,
```

```
      SPACE, TSS
    USE "HELP" COMMAND TO GET BRIEF DESCRIPTION OF THESE COMMANDS.
    (EXAMPLE)  HELP SEARCH ... DISPLAYS THE BRIEF DESCRIPTION OF SEARCH.
    USE "END" COMMAND TO EXIT DO-STATE.
(7) DO:LOOK
    私は，今日初めてSIGMAを使いました.
    とても速いので感動しました.
    私もテキストデータベースを作ろうと思います.
(8) DO:END
(9) SIGMA)END
    READY
(10) LOGOFFC
    RETURN CODE : 0000
    CPU TIME(   0.40SEC.) USE TIME(    4MIN.) REGION SIZE(1744KB)
    INPUT(     13LINES) OUTPUT(       48LINES) EXCP(      343TIMES)
    SESSION CHARGE(TSUnnnn,hh:mm:ss)                              16YEN
    TOTAL CHARGE SINCE mm/dd/yy (EXCEPT THIS SESSION'S)       xx,xxxYEN
    KEQ56470I A79999A LOGGED OFF AT hh:mm:ss ON month day, year+
    KEQ54220I SESSION ENDED
```

使用例1の説明

(1) SIGMA を使用するために，LOGON コマンドを用いてTSS セッションを開始する．“A79999A”は課題番号である．正しいパスワードを入力すると，TSS セッションが開始され，READY 状態のプロンプト“READY” が表示される．READY 状態でTSS のコマンドを入力できる．

(2) SIGMA を起動するには，READY 状態で“SIGMA” を入力する．

(3) SIGMA を使用するためには，MEMO領域という個人用の作業場が必要である．最初にSIGMA を使うときは，システムが作成するMEMO領域のブロック数とマスターキーを尋ねてくる．50ブロックで1メガバイト程度の領域を確保する．またマスターキーは，4桁の正の整数で指定する．マスターキーをつけないときには，復改のみを入力する．ここでは“9999”を指定した．マスターキーは，領域の初期化や拡張，ファイルのパスナンバーを忘れたときに必要となる（パスナンバーについては3．4．8を参照されたい）．一度MEMO領域を作れば，以降SIGMA を使用するときにこの操作は必要ない．

(4) SIGMA 状態のプロンプト“SIGMA)”が表示されるとSIGMA のコマンドを入力できる．ここでは，KEYIN コマンドを入力している．MEMO領域には，名前をつけてファイルを保管できるMEMO区域や，一時的にファイルを置く作業区域などがある．KEYIN コマンドはこの作業区域にファイルを作るコマンドである．MEMO領域のほかに，共有のファイルを置くためのSIGMA 領域と，データセットなどの集まりである外部領域がある．SIGMA 領域には，名前をつけてファイルを保管できるSIGMA 区域と，そのバックアップをとるSIGMA バックアップ区域がある（SIGMA バックアップ区域に関しては3．4．4を参照されたい）．

(5) KEYIN コマンドでは，プロンプト“K)”に続いて端末からテキストを入力できる．入力の終わりは，プロンプト“K)”に続いて，“;;;”（半角のセミコロン三つ）で指示する．

(6) SIGMA 状態でコマンドを入力するとDO状態になり，プロンプト“DO:” が表示される．SIGMA では，“K)”以外のプロンプトが出ているときに，“?” を入力すれば，ヘルプメッセージを見ることができる．ここでは，DO状態で使用できるコマンドが表示される．“?” という文字列を実際に入力したいときは“??”とすればよい．SIGMA には SIGMA状態とDO状態の二つの状態があるが，ここではどちらも同じものと思っていてよい（詳しくは3．2 を参照されたい）．

(7) LOOKコマンドを使って，(5) で作成した作業区域にあるファイルの内容を表示する．

(8) DO状態を終わるには，END コマンドを使う．

(9) SIGMA 状態で，END コマンドを使いSIGMA を終了させている．DO状態で ENDコマンドを投入するとSIGMA 状態になり，SIGMA 状態ではSIGMA が終了する．

(10) TSS セッションを終了するためには，LOGOFFあるいはLOGOFFC コマンドを用いる．LOGOFFC を用いた場合，CPU 時間などのシステム使用量とともに使用負担金，累積負担金が表示される．

使用例2（簡単なファイル操作）

ファイル操作の簡単な例をあげる．KEYIN で作成したファイルに名前をつけたり，オペレーティングシステムが管理するテキストファイルをSIGMA へ取り込んだり，ファイルの一覧を見たりする．また，SIGMA システムの環境の設定をする．

```
    READY
(1) SIGMA
(2) SIGMA)LOOK
    私は，今日初めてSIGMAを使いました．
    とても速いので感動しました．
    私もテキストデータベースを作ろうと思います．
(3) DO:KEYIN
     TO EXIT TEXT INPUT MODE, TYPE ;;; AT THE PROMPT K).
    K)きのうは雨でした．
    K);;;
    DO:KEYIN
     TO EXIT TEXT INPUT MODE, TYPE ;;; AT THE PROMPT K).
    K)きょうは曇りです．
    K);;;
    DO:KEYIN
     TO EXIT TEXT INPUT MODE, TYPE ;;; AT THE PROMPT K).
    K)あしたは晴れでしょう．
    K);;;
(4) DO:DIR W

          TOTAL =   4
     1490 SECTOR(S) AVAILABLE
(5) DO:LIST W.1
    あしたは晴れでしょう．
    DO:LIST W.2
    きょうは曇りです．
    DO:LIST W.3
    きのうは雨でした．
    DO:LIST W.4
    私は，今日初めてSIGMAを使いました．
    とても速いので感動しました．
    私もテキストデータベースを作ろうと思います．
(6) DO:MOVE W.4 HAJIMETE.TEXT
(7) DO:LIST
    FILE:=?
    TYPE THE FILENAME YOU WANT TO LIST.
(8) FILE:=HAJIMETE.TEXT
    私は，今日初めてSIGMAを使いました．
```

```
    とても速いので感動しました.
    私もテキストデータベースを作ろうと思います.
 (9) DO:DIR W

            TOTAL =   3
     1490 SECTOR(S) AVAILABLE
(10) DO:PUT TENKI.ASHITA
(11) DO:DIR W

            TOTAL =   2
     1490 SECTOR(S) AVAILABLE
(12) DO:PUT TENKI.KYOU
     DO:DIR W

            TOTAL =   1
     1490 SECTOR(S) AVAILABLE
(13) DO:GET TENKI.ASHITA
(14) DO:DIR W

            TOTAL =   2
     1490 SECTOR(S) AVAILABLE
(15) DO:LOOK
     あしたは晴れでしょう.
     DO:LIST W.2
     きのうは雨でした.
(16) DO:TSS
(17) TSS:LISTC
     IN CATALOG:SYS1.EDFUCAT1
     A79999A.A.DATA
     ........................
     A79999A.LIBSMALL.TEXT
     ........................
     A79999A.ZZZ.TEXT
(18) TSS:↙
(19) DO:COPY X.LIBSMALL.TEXT LIBSMALL.TEXT
(20) DO:DIR *.TEXT
     PASSNUMBER:=↙
     HAJIMETE.TEXT
     LIBSMALL.TEXT

            TOTAL =   2          PREFIX =
     1478 SECTOR(S) AVAILABLE
(21) DO:PROFILE
(22) PROF:DISPLAY
     ****** SYSTEM CONSTANTS ******
     NEWLINE CHARACTER =    !
     CONTINUE CHARACTER =   -
     TRIPLE DOT CHARACTER = .
     LINE SIZE =             78
     BELL MODE =            ON
     PREFIX OF MEMO =
     PREFIX OF SIGMA =      A79999A
     SYSOUT CLASS =         U
     HELP LEVEL =           1
(23) PROF:PREFIX
```

```
(24) MEMO OR SIGMA(M OR S)?S
(25) PREFIX:=?.2
     SIGMA領域のファイルを参照するときのプレフィックスを入力してください.
     初期値は課題番号です.
     復改のみを入力すると，プレフィックスは課題番号になります.
(26) PREFIX:=A71095B
(27) PROF:↙
(28) DO:DIR S
     PASSNUMBER:=↙
     DRAFT.TEXT
     DRAFT1.TEXT
     HELP
     KOKIN
     KOKIN.NUM
     MANUAL
     RIFISLIB.ENG.TEXT
     RIFISLIB.JPN.TEXT

             TOTAL =   8          PREFIX = A71095B
      6561 SECTOR(S) AVAILABLE
(29) DO:END
(30) SIGMA)END
     READY
```

使用例２の説明

(1) SIGMA を起動するために，READY 状態で“SIGMA” を投入する．使用例１でMEMO領域はすでに作っているので，今度は作る必要はない．

(2) SIGMA 状態のプロンプト“SIGMA)”に引続きSIGMA のLOOKコマンドを入力している．LOOKは作業区域の内容を見るためのコマンドである．

(3) SIGMA 状態でコマンドを入力すると，DO状態になる．ここでは，３回 KEYINコマンドを使って，作業区域に新しく３個のファイルを作っている．作業区域は図１(a) のような構造（深さが20のスタック）になっていて，この区域に新しくファイルを作ると，古いファイルは順次下に押し下げられて最新のファイルが最上部（トップという）にくるようになっている（図１(b),(c),(d) ）．作業区域のファイルは，上から順にW.1,W.2,...,W.20と名前がついている．作業区域のトップのファイル W.1 はW でも参照できる．作業区域のファイルの個数が20個を越えたときは，一番底のファイルがバックアップ区域と呼ばれる所に自動的に移動する（３．４．３参照）．

| | |
|---|---|
| W.1 | 私は，今日… |
| W.2 | |
| W.3 | |
| W.4 | |
| W.5 | |
| ⋮ | ⋮ |
| W.20 | |

図１(a)

| | |
|---|---|
| W.1 | きのうは雨… |
| W.2 | 私は，今日… |
| W.3 | |
| W.4 | |
| W.5 | |
| ⋮ | ⋮ |
| W.20 | |

図１(b)

| | |
|---|---|
| W.1 | きょうは曇… |
| W.2 | きのうは雨… |
| W.3 | 私は，今日… |
| W.4 | |
| W.5 | |
| ⋮ | ⋮ |
| W.20 | |

図１(c)

| | |
|---|---|
| W.1 | あしたは晴… |
| W.2 | きょうは曇… |
| W.3 | きのうは雨… |
| W.4 | 私は，今日… |
| W.5 | |
| ⋮ | ⋮ |
| W.20 | |

図１(d)

(4) DIRECTORY コマンドで作業区域のファイルの数を見ると，最初に作ったファイルと合わせて4個あることがわかる．領域の使用可能なセクタ数も表示されている．1セクタは 714バイトである．ここでは，DIRECTORY コマンドの省略形DIR を入力じている．このように，SIGMA のコマンドの入力は，他のコマンドと区別できるところまで入力すればよい．例えば，DIRECTORY には，DI,DIR,DIRE,...,DIRECTORY のいずれを用いてもよい．各コマンドの最短省略形については，4章および付録を参照されたい．以後，コマンドを入力する際に省略形を用いることがある．

(5) LISTコマンドを使って，それぞれ内容を見ると新しい順に並んでいるのがわかる．

(6) MOVEコマンドを使って，W.4 のファイルをHAJIMETE.TEXT に移している．MOVEコマンドを使った後は，W.4 にもとのファイルは残らない（図1(e)）．

(7) LISTコマンドにオペランド（コマンドに引き続いて入力すべきもの）をつけずに入力すると，プロンプト“FILE:=”が表示される．このように，コマンドのオペランドの入力が必要になるとシステムはプロンプトを表示する．ここでは，“?”を入力してヘルプメッセージを見てみる．

(8) ファイル名としてHAJIMETE.TEXT を入力すると，その内容を表示する．

(9) (6) で作業区域のファイルを移動したので，DIRECTORY をとってみると，作業区域には3個のファイルがあることがわかる．

(10) PUT コマンドでW.1 をTENKI.ASHITAに移動する．PUT コマンドは，MOVEの第1オペランドをW.1 に固定したものと考えることができる．このコマンドにより，W.1 はなくなるので，それより下のファイルが自動的に上に詰められる（図1(f)）．

(11) DIRECTORY をとってみると，PUT コマンドにより，作業区域のファイルがさらに減っているのがわかる．

(12) さらにPUT コマンドで移動してDIRECTORY をとってみると作業区域のファイルが1個になっているのがわかる（図1(g)）．

(13) 今度は，GET コマンドを使ってTENKI.ASHITAをW.1 に移す．作業区域の中のファイルは順次下に押し下げられる．GET コマンドはMOVEコマンドの第2オペランドをW.1 に固定したものであると考えることができる（図1(h)）．

(14) 作業区域のDIRECTORY をとってみると，作業区域のファイルの数が増えているのがわかる．

(15) それぞれのファイルの内容を表示する．LOOKはLOG 区域のトップのファイルを見るためのコマンドであり，LIST W.1と同じ働きをする．

| W.1 | あしたは晴… |
|---|---|
| W.2 | きょうは曇… |
| W.3 | きのうは雨… |
| W.4 | |
| W.5 | |
| ⋮ | ⋮ |
| W.20 | |

図1(e)

| W.1 | きょうは曇… |
|---|---|
| W.2 | きのうは雨… |
| W.3 | |
| W.4 | |
| W.5 | |
| ⋮ | ⋮ |
| W.20 | |

図1(f)

| W.1 | きのうは雨… |
|---|---|
| W.2 | |
| W.3 | |
| W.4 | |
| W.5 | |
| ⋮ | ⋮ |
| W.20 | |

図1(g)

| W.1 | あしたは晴… |
|---|---|
| W.2 | きのうは雨… |
| W.3 | |
| W.4 | |
| W.5 | |
| ⋮ | ⋮ |
| W.20 | |

図1(h)

(16) SIGMA ではシステムを終了しないでTSS コマンドが使用できる．そのためには，SIGMA コマンドのTSS を入力すればよい．
(17) コマンドTSS によってプロンプトが“TSS:”に変わり，TSS コマンドを入力できる．LISTC コマンドにより，所有するデータセットの一覧が表示される．その中にLIBSMALL.TEXT というファイルがあることがわかる．
(18) プロンプトが“TSS:”の時に復改のみか，“END” を入力すれば，DO状態に戻る．
(19) 外部ファイル（オペレーティングシステムが管理しているデータセット）をSIGMA に取り込むためには，COPYコマンドを用いる．データセット名の先頭に“X.”をつければ参照できる．データセットLIBSMALL.TEXT の複製を作り，LIBSMALL.TEXT という名前のMEMOのファイルにした．
(20) DIRECTORY コマンドでMEMO区域のファイル名を表示する．DIRECTORY コマンドでは，ワイルドカード“*” が使える．“*” は任意の文字列とマッチする．この例では，“.TEXT” で終わるファイル名の一覧を表示する．また，ワイルドカードを使ったファイル名を指定すると，パスナンバーを入力しなければならない．パスナンバーを指定しないとパスナンバーがついていないファイルだけ，パスナンバーを指定するとそのパスナンバーと一致するファイルと，パスナンバーがついていないファイルの一覧が表示される．パスナンバーについては，３．４．９を参照されたい．
(21) SIGMA には，システムが使用するためのいくつかの定数がある．そのシステム定数を変更するにはPROFILE コマンドを用いればよい．PROFILE コマンドを入力すればプロンプトが“PROF:” に変わる．PROFILE コマンドに関しては４章を参照されたい．
(22) まず，DISPLAY サブコマンドを用いてシステム定数を表示してみる．PROFILE のサブコマンドも他のサブコマンドと区別できるところまで入力すればよい．
(23) SIGMA 領域のプレフィックスが初期値の課題番号A79999A であるので，それを変えるためにPREFIXサブコマンドを入力する．
(24) プレフィックスには，MEMO領域のプレフィックスとSIGMA 領域のプレフィックスがある．ここではSIGMA 領域のプレフィックスを変えるので，S を入力する．
(25) “?.2” と入力すれば，レベル２のヘルプメッセージを見ることができる．“?” だけを入力すれば，PROFILE コマンドで設定されたヘルプレベルに対応するメッセージが出る．
(26) プレフィックスとしてA71095B を入力する．
(27) “PROF:” のプロンプトから“DO:” に戻るにはEND サブコマンドか復改のみを入力すればよい．
(28) SIGMA 区域のDIRECTORY をみてみる．“S” だけを指定した場合には，ワイルドカードを使わなくてもパスナンバーを入力しなければならない．ここでは，復改のみを入力したのでパスナンバーがついていないSIGMA 区域のファイル名が表示される．
(29) END コマンドを入力してSIGMA 状態に戻る．
(30) END コマンドを入力してSIGMA システムを終える．

（注意）SIGMA 区域のファイルを参照する際の注意
以下の使用例では，SIGMA 区域のファイル名に対してS.KOKIN などの指定をしているが，そのためには使用例２の(21)から(26)のように，PROFILE コマンドでSIGMA 区域のプレフィックスをA71095B に設定しておかなければならない．この設定がない場合は，S.'A71095B.KOKIN' などと指定しなければならない．なお，ファイル名の規約に関しては３．４．７を参照されたい．

## 2．2　古今和歌集の検索

和歌の研究では，たとえばある景物のよみこまれた歌をすべてリストアップするというような作業は，ほとんど手仕事で行われているのが現状で，大変な労力を要している．ここでは古今和歌集[19]の一部（巻第一から巻第八まで）をテキストファイル化して検索を行ってみた．これにより，

秋ののにをく白露はたまなれや　つらぬきかくるくものいとすぢ

のように，「露」と「たま」の二つを含んでいる歌を瞬時にリストアップすることなどができる．

使用例3（古今和歌集におけることばの検索）

```
(1) DO:LIST S.KOKIN

                古今和歌集巻第一


                    春哥上

    @
          (T)ふるとしに春たちける日よめる$          (A)在原元方$
    #0001¥年の内に春はきにけり　ひととせをこぞとやいはん　ことしとやいはん$
    @
          (T)春たちける日よめる$                    (A)紀貫之$
    #0002¥袖ひぢてむすびし水のこほれるを　春立つけふの風やとくらん$
    !

(2) DO:SEARCH
(3)       RECORD DELIMITERS
    D01:=@
    D02:=↙
          ITEM DELIMITERS
    D02:=$
    D03:=↙
(4)       KEYWORDS
    A01:=こそ...けれ
    A02:=露
    A03:=つゆ
    A04:=玉
    A05:=珠
    A06:=たま
    A07:=(T)
    A08:=¥
    A09:=↙
(5)       LOGICAL FORMULAS
    Z01:=A1
    Z02:=A2,A3
    Z03:=(A2,A3).A7
    Z04:=(A2,A3).(A4,A5,A6).A8
    Z05:=↙
(6) REPORT(Y/N)?Y
(7) FILE:=S.KOKIN
       RETRIEVED TEXTS
    QUESTION 01 (Z01) =          10          10
```

```
     QUESTION 02 (Z02) =         24          24
     QUESTION 03 (Z03) =          1           1
     QUESTION 04 (Z04) =          4           4
     TOTAL             =         34          34
     CPU (SEC/1000)    =         54          54
     FILE:=↙
 (8) DO:REFILE
 (9) REPORT(Y/N)?N
(10) QUESTION:=1
     NEW RECORD DELIMITER:=@
     NUMBERING(N/Y)?N
     OUTPUT-FILE:=T
     @
          (T)ならのいそのかみでらにて郭公のなくをよめる$
                                                    (A)そせい$
     #0144¥いその神ふるき宮この郭公　こゑばかりこそむかしなりけれ$
     @
          (T)これさだのみこの家の哥合によめる$          (A)大江千里$
     #0193¥月みればちぢにものこそかなしけれ　わが身ひとつの秋にはあらねど$
     @
      ..............
(11) QUESTION:=4
     NEW RECORD DELIMITER:=@
     NUMBERING(N/Y)?N
     OUTPUT-FILE:=KOKIN.KEKKA
     QUESTION:=↙
     DO:LIST KOKIN.KEKKA
     @
          (T)西大寺の邊の柳をよめる$                    (A)僧正遍昭$
     #0027¥浅緑いとよりかけて　白露を珠にもぬける春の柳か$
     @
          (T)はちすの露をみてよめる$                    (A)僧正遍昭$
     #0165¥はちすばの　にごりにしまぬ心もて　なにかはつゆをたまとあざむく$
     @
          (T)*題しらず$                                 (A)*よみ人しらず$
     #0222¥はぎの露たまにぬかんととればけぬ　よしみん人は枝ながらみよ$
               [ある人のいはく、この哥はならのみかどの御うた也と]
     @
          (T)是貞のみこの家の哥合によめる$              (A)文屋あさやす$
     #0225¥秋ののにをく白露はたまなれや　つらぬきかくるくものいとすぢ$
     @
(12) DO:.S.KOKIN.NUM
        RETRIEVED TEXTS
     QUESTION 01 (Z01) =         10
     QUESTION 02 (Z02) =         24
     QUESTION 03 (Z03) =          1
     QUESTION 04 (Z04) =          4
     TOTAL             =         34
     QUESTION:=2
     QUESTION 02 (Z02) =         24
     DO:END
     DO:LOOK
       #0027  #0165  #0188  #0199  #0209  #0221  #0222  #0223  #0224  #0225
       #0247  #0257  #0258  #0259  #0260  #0261  #0270  #0273  #0291  #0306
       #0307  #0369  #0373  #0375
```

使用例3の説明

(1) 検索の対象とするファイル S.KOKINの内容をLISTコマンドで表示している．題詞，作者にはそれぞれ頭に“(T)”，“(A)”を付し，歌には4桁の通し番号をうった．これら一つ一つの切れ目を示すため，すべて末尾には“$”がつけてある．歌の後に注がついているときには，それを［ ］でくくっておいた．また，題詞や作者が前の歌と同じで省いてあるものにもそれを補っておいた．その箇所には“*”をつけてそれとわかるようにしてある．題詞，作者，歌，注でひとまとまりとして“@”で区切っておいた．なお，表示を中断するために割り込みキーを押したので，“!” が表示される．

(2) SEARCHコマンドにより検索を行う．

(3) SEARCHで検索の対象とするのは，陽には構造を持たない1本の文字列（テキストファイル）であるため，仮想的なレコード構造を設定して検索を行う．レコード区切り語(RECORD DELIMITER)とよばれる文字列にはさまれた部分が1つのレコードとみなされる．また，項目区切り語(ITEM DELIMITER)とよばれる文字列は，1レコード中での質問の評価を行う単位（これを項目とよぶ）を指定するものである．この例では，一つの歌を1レコードとし，題詞，作者，歌，注をそれぞれ1項目とするため，レコード区切り語には“@”を，項目区切り語には“$”を指定した．

(4) プロンプト“A01:=”，“A02:=”，… に続いて，検索するキーワードを登録していく．

(5) キーワードを登録したAi（これをキーワード変数とよぶ）を用いて質問式を構成し，プロンプト“Z01:=”，“Z02:=”，…に続いて登録する．“...” （半角のピリオド三つ）は任意の文字列（空でもよい）を表す．質問式Z1は，「こそ……けれ」という形の係り結びを含むものをすべて検索せよ，という意味である．また，“,” は「または」を，“.” は「かつ」を表す．したがって，質問式Z2は「露」または「つゆ」を含むレコードをすべて検索せよ，という意味であり，質問式Z3はそれを題詞の中に含んでいるようなレコードを検索せよという意味になる．質問式Z4では「露」（あるいは「つゆ」）と「玉」（あるいは「珠」，「たま」）の両方を含む歌をすべて検索することになる．

(6) 検索件数（各質問に対してヒットした件数）を表示するかどうかを指定する．“Y”(YES)を指定した．

(7) “FILE:=”に続いて，検索を行うファイル名を入力する．ここでは，“S.KOKIN” を入力した．検索を行って各質問にヒットした件数と処理に要した時間を表示している．左列はそのファイルに対する結果であり，右列は今までの累積である．

(8) 検索結果をテキストとして見るためには，REFILEコマンドを用いる必要がある．

(9) 各質問に対してヒットした件数を表示するかどうかきいている．ここでは，検索したばかりでわかっているので “N”(NO)とした．

(10) 何番の質問に対する検索結果を再ファイル化するのかきいているので“1” を指定した．続いて，再ファイル化する際のレコード区切り語，結果に番号を打つかどうか，出力先のファイル名をきいてくるので，それぞれ，“@”，“N”，“T”（T は端末装置を表す特殊ファイル名）を指定した．検索の結果が端末画面に表示される．ここでは「こそ……けれ」という形の係り結びを含む歌（の候補）がリストアップされている．

(11) 質問4に対する検索結果を，KOKIN.KEKKA というファイルに出力している．その内容はLISTコマンドを用いて見ることができる．

(12) 質問2に対する検索結果のように件数が多いときには，歌に付した通し番号のみを切り出せれば便利である．“.S.KOKIN.NUM”と投入すると，通し番号のみを切り出して作業区域のトップW に出力する．これは，S.KOKIN.NUM というファイルにこのような作業をする SIGMAのコマンドが書かれて

おり，この内容を実行したからである．通し番号は Wに書かれているので，LOOKコマンドを用いて見ている．ファイルに書かれたコマンドを実行する方法の詳細については，3.2 を参照されたい．

## 2．3 文献型データの検索

文献型データの検索の例を挙げる．図書データを対象に著者名などで検索したり，必要な書誌事項のみを切りだしたりする．

使用例４（文献型データの検索）

```
(1) DO: LIST S.RIFISLIB.JPN.TEXT
    $
    (NO) G45/1
    (TI) 新システム設計論
    (AR) Gill,A.
    (TR) 森政弘　合田周平
    (JP) 朝倉書店
    (MS) J 1965 P217
    $
    (NO) Ko73/1
    (TI) 自動制御装置
    (AR) 近藤文治　秋葉光俊
    (JP) 日刊工業新聞社
    (MS) J 1967 P254　（第７　工業電子装置
    シリーズ）
    $
    (NO) Ts91
(2) !
(3) DO:SEARCH
(4)      RECORD DELIMITERS
    DO1:=$
(5) DO2:=↙
(6)      ITEM DELIMITERS
    DO2:=!
    DO3:=↙
(7)      KEYWORDS
(8) AO1:=(AR)
    AO2:=Hopcroft
(9) AO3:=(AR)...Hopcroft
    AO4:=(TR)...野崎昭弘
    AO5:=↙
(10)     LOGICAL FORMULAS
(11) ZO1:=A1
     ZO2:=A2
     ZO3:=A3
     ZO4:=A4
(12) ZO5:=A1.A2
     ZO6:=A3.A4
(13) ZO7:=Z3.Z4
(14) ZO8:=^(A1.A2)
     ZO9:=Z4.Z8
(15) Z10:=↙
(16) REPORT(Y/N)?Y
(17) FILE:=S.RIFISLIB.JPN.TEXT
```

```
                RETRIEVED TEXTS
QUESTION 01 (ZO1) =      347      347
QUESTION 02 (ZO2) =        3        3
QUESTION 03 (ZO3) =        3        3
QUESTION 04 (ZO4) =        4        4
QUESTION 05 (ZO5) =        3        3
QUESTION 06 (ZO6) =        0        0
QUESTION 07 (ZO7) =        3        3
QUESTION 08 (ZO8) =      493      493
QUESTION 09 (ZO9) =        4        4
TOTAL             =      493      493
CPU (SEC/1000)    =       98       98
(18) FILE:=↙
(19) DO:REFILE
     REPORT(Y/N)?N
(20) QUESTION:=5
(21) NEW RECORD DELIMITER:=!$
(22) NUMBERING(N/Y)?Y
(23) OUTPUT-FILE:=T

     $0000000001
     (NO) H863/1
     (TI) 言語理論とオートマトン
     (AR) Hopcroft,J.E.　Ullman,J.D.
     (TR) 野崎昭弘　木村泉
     (JP) サイエンス社
     (MS) J 1971 P285　（第６　サイエンスラ
     イブラリ・情報電算機）

     $0000000002
     (NO) A211/3(1)
     (TI) アルゴリズムの設計と解析　第１
     (AR) Aho,A.V.　Hopcroft,J.E.　Ullman,J.
     D.
     (TR) 野崎昭弘　野下浩平
     (JP) サイエンス社
     (MS) J 1977　（３５　サイエンスライブラ
     リ・情報電算機）

     $0000000003
     (NO) A211/3(2)
     (TI) アルゴリズムの設計と解析　第２
     (AR) Aho,A.V.　Hopcroft,J.E.　Ullman,J.
     D.
```

```
    (TR) 野崎昭弘　野下浩平
    (JP) サイエンス社
    (MS) J 1977　(３６　サイエンスライブラ
    リ・情報電算機)

    $
(24) QUESTION:=9
    NEW RECORD DELIMITER:=#
    NUMBERING(N/Y)?N
    OUTPUT-FILE:=BUNKEN1
(25) QUESTION:=↙
(26) DO:LIST BUNKEN1
    #
    (NO) B18/2
    (TI) 問題解決の理論；人工知能の基礎
    (AR) Banerji,R.B.
    (TR) 南雲仁一　野崎昭弘
    (JP) 産業図書，東京
    (MS) J 1974 P202
    #
    (NO) H863/1
    (TI) 言語理論とオートマトン
    (AR) Hopcroft,J.E.　Ullman,J.D.
    (TR) 野崎昭弘　木村泉
    (JP) サイエンス社
    (MS) J 1971 P285　(第６　サイエンスラ
    イブラリ・情報電算機)
    #
    (NO) A211/3(1)
    (TI) アルゴリズムの設計と解析　第１
    (AR) Aho,A.V.　Hopcroft,J.E.　Ullman,J.
    D.
    (TR) 野崎昭弘　野下浩平
    (JP) サイエンス社
    (MS) J 1977　(３５　サイエンスライブラ
    リ・情報電算機)
    #
    (NO) A211/3(2)
    (TI) アルゴリズムの設計と解析　第２
    (AR) Aho,A.V.　Hopcroft,J.E.　Ullman,J.
    D.
    (TR) 野崎昭弘　野下浩平
    (JP) サイエンス社
    (MS) J 1977　(３６　サイエンスライブラ
    リ・情報電算機)
    #
(27) DO:SEA
        RECORD DELIMITERS
(28) DO1:=!
    DO2:=↙
        ITEM DELIMITERS
(29) DO2:=↙
        KEYWORDS
    A01:=#
    A02:=(TI)
    A03:=(AR)
    A04:=↙
        LOGICAL FORMULAS
(30) Z01:=,
    Z02:=↙
    REPORT(Y/N)?N
    FILE:=BUNKEN1
    FILE:=↙
(31) DO:REF
    REPORT(Y/N)?N
    QUESTION:=1
    NEW RECORD DELIMITER:=!
    NUMBERING(N/Y)?N
    OUTPUT-FILE:=BUNKEN2
    QUESTION:=↙
(32) DO:LIST BUNKEN2

    #
    (TI) 問題解決の理論；人工知能の基礎
    (AR) Banerji,R.B.
    #
    (TI) 言語理論とオートマトン
    (AR) Hopcroft,J.E.　Ullman,J.D.
    #
    (TI) アルゴリズムの設計と解析　第１
    (AR) Aho,A.V.　Hopcroft,J.E.　Ullman,J.
    D.
    #
    (TI) アルゴリズムの設計と解析　第２
    (AR) Aho,A.V.　Hopcroft,J.E.　Ullman,J.
    D.
    #
    DO:
```

使用例４の説明

(1)　ファイルS.RIFISLIB.JPN.TEXTの内容を表示させている．

(2)　画面での出力を，割り込み（BREAKキー）で中断させている．

(3)　SEARCHコマンドを用いて検索を開始している．SEARCHの対象とするファイルは陽には構造をもたない１本の長い文字列（テキストファイル）である．検索を行う場合は，区切り語を与えることにより，レコードと項目という二つの構造を指定する．ここでは，SIGMA 領域にある図書データS.RIFISLIB.JPN.TEXT を対象に検索を行う．文献型データの場合，文献１件が１レコードであり，１レコー

ド中の書名，著者名などが項目である．レコードを与えるための区切り語をレコード区切り語，項目を与えるための区切り語を項目区切り語という．区切り語には任意の文字列を指定することができる．区切り語はあわせて99個まで指定することができる．

(4) 最初にメッセージ“RECORD DELIMITERS” が表示される．この状態で，プロンプト“D01:=” (01は区切り語の通し番号) に従って，レコード区切り語を登録する．S.RIFISLIB.JPN.TEXT では，各文献が“$” で区切られているので，レコード区切り語として“$” を登録する．

(5) レコード区切り語を1語登録すると，次のレコード区切り語の登録を促すプロンプトが表示される．レコード区切り語の登録を終了する場合は，復改のみを入力する．

(6) メッセージ“ITEM DELIMITERS” が表示される．この状態で，プロンプトに続いて項目区切り語を登録する．プロンプトの数字は，レコード区切り語の番号の続きである．ここでは，項目区切り語として，代用復改記号“!” を入力している．S.RIFISLIB.JPN.TEXT では，各項目の最後は必ず改行されているので，項目区切り語として復改記号を指定している．復改記号を指定する場合は，必ず代用復改記号を用いなければならない．代用復改記号の初期値は，“!” であり，PROFILEコマンドで変更可能である．項目区切り語の登録が終了している場合は，復改のみを入力する．

(7) メッセージ“KEYWORDS” が表示される．この状態で，検索に利用するキーワードをプロンプト“Ai:=” (i は99までの数) に続いて登録してゆく．キーワードは99個まで登録できる．Aiをキーワード変数と呼び，キーワードはAiに登録されているという．

(8) ここでは，キーワード変数A1にキーワード(AR)を登録している．

(9) キーワードに文字列“...” (トリプル・ドット) を用いている．トリプル・ドットは任意の文字列を表す．“(AR)...Hopcroft” は，(AR)から始まって，Hopcroftで終わる任意の文字列を表す．

(10) キーワードの登録が終わると，メッセージ“LOGICAL FORMULAS” が表示される．この状態で，検索の質問を表す論理式をプロンプト“Zi:=” (i は99までの数) に続いて登録してゆく．Zi (i は99までの数) を式変数と呼び，論理式は式変数Ziに登録されたという．論理式は，キーワード変数と式変数と，それを論理演算子 (“.” (and)，“,” (or)，“^” (not)) などで結んだ式が利用できる．論理式は99個まで登録できる．

(11) 式変数Z1にキーワード変数一つだけからなる論理式A1を登録している．Z1を用いた質問で，「A1に登録されたキーワード(AR)が出現する項目を含む」レコードが検索される．

(12) 式変数Z5に論理式A1.A2を登録している．式変数Z5を用いた質問は，「A1に登録されたキーワード(AR)と，A2に登録されたキーワードHopcroftの両方が出現するような項目を含む」という意味になる．Z6についても同様である．

(13) 式変数Z7に，登録済みの式変数を用いて，論理式 Z3.Z4を登録している．式変数Z7を用いた質問は，「Z3を用いた質問を満たし，かつ，Z4を用いた質問を満たす」という意味になる．Z6を用いた質問の意味との違いに注意されたい．この意味の違いは，式変数の評価方式から生じている．論理式の評価方式については，3.5.6を参照されたい．

(14) 論理式に，“^” を用いている．Z8を用いた質問の意味は，「A1，A2に登録されたキーワードがともには出現しない項目を含む」になる．これは，Z5を用いた質問の論理的な否定にはならない．また，“^” の結合力は“.” より強いので，“(“,”)” を用いないと，(^A1).A2と解釈される．

(15) 論理式の入力が終了したときは，復改のみを入力する．

(16) 検索件数 (各質問に対してヒットした件数) を表示するかどうかを指定する．“Y” (YES)を指定した．

(17) 検索を実行するファイルを指定している．今回の検索の対象のファイルは上で表示したS.RIFI SLIB.JPN.TEXT である．ファイルを指定すると，検索が実行される．ここでは，(16)で“Y” を指定しているので“RETRIEVED TEXTS” の表示のあと，検索件数が表示される．次の表示

```
QUESTION 01 (Z01) =        347        347
```

は，質問番号1番は式変数Z1を用いた質問であり，ヒットしたレコードは 347件（“=” の次の数字）であることを示している．最も右の数字は，同じ質問による検索を複数のファイルに対して行った場合の検索件数の累計を表す．また，

```
TOTAL                =        493        493
CPU (SEC/1000)       =         98         98
```

においてTOTAL の行は，質問番号1番から9番までの少なくとも一つの質問にヒットしたレコードの件数を表す．また，CPU の行は検索に要した時間を表す（単位は1000分の1秒）．ヒットしたレコードの位置を，質問ごとに記録した索引ファイルが，索引区域のトップに積まれる．

(18) 同じ質問を用いた検索を別のファイルに対して行うかどうかを尋ねてくる．これ以上検索したいファイルがない場合は復改のみを入力する．

(19) 上の検索で，各質問ごとにヒットしたレコード全体からなる新たなテキストファイルを作成することを再ファイル化という．コマンドREFILEにより再ファイル化を実行している．再ファイル化は質問番号ごとに行うので，最初に検索件数を再表示する必要があるかどうかを尋ねてくる．必要ならば“Y” を，不用ならば “N”を入力する．

(20) “QUESTION:=”につづいて，再ファイル化を行う質問番号を指定している．ここでは，質問番号5は式変数Z5を用いた質問であった．

(21) 検索時に複数指定できたレコード区切り語は，再ファイル化の段階で一つに統一しなければならない．その場合，統一後のレコード区切り語は，検索時のものを用いる必要はない．しかし，同じものを用いる場合でも，必ず指定しなければならない．ここでは，“!$”を指定している．“!” は代用復改記号である．

(22) 再ファイル化の結果を出力するときに，各レコードに通し番号を打つかどうかを尋ねてくるので，必要ならば“Y” を，不用ならば “N”を入力する．

(23) “OUTPUT-FILE:=” に続いて，再ファイル化の出力先のファイル名を指定する．ここでは，端末装置を表す特殊ファイル名 “T”を指定しているので，再ファイル化の結果は端末に表示されている．レコード区切り語が “$”から“!$”に変わっていることに注意されたい．

(24) 一つの質問に対する再ファイル化が終了すると，次に再ファイル化が必要な質問番号を尋ねてくるので，同じ要領で繰り返せばよい．ここでは，質問式9番に対する再ファイル化の結果を，レコード区切り語を“#”に統一した後，BUNKEN1という名のMEMOファイルに出力している．

(25) これ以上再ファイル化の必要がない場合は復改のみを入力する．

(26) LISTコマンドで再ファイル化の結果を表示させている．

(27) 上で作成したファイル BUNKEN1から，書名と著者名を抜きだしたファイル BUNKEN2を作成する．SIGMA では，SEARCHコマンドによってこれが可能である．ここでは，SEARCHコマンドの省略形SEA を用いている．

(28) BUNKEN1 は，“#” をレコード区切り語と指定すれば，文献データとしての構造をもつ．しかし，復改記号をレコード区切り語と指定すれば，1行を1レコードとみなすことができる．SIGMA では，区切り語の指定を変えれば，一つのテキストファイルに異なる構造を与えることが可能である．

(29) ここでは，項目区切り語を与えていない．この場合には，１レコード１項目になる．
(30) 論理式の登録の際，論理記号だけを入力すると，省略形と解釈される．ここでは，

Z1:=A1,A2,A3（すべてのキーワード変数の論理和）

なる登録がなされたものと解釈される．他の論理記号を用いた場合については３．５．４を参照されたい．今回の検索は，１レコードにつき１項目であったから，Z1を用いた質問の意味は，「キーワード#，(TI)，(AR)のどれかが出現するレコードを検索せよ」になる．
(31) 直前の検索結果を再ファイル化するために，REFILEコマンドの省略形 REFを用いている．検索のときは１行１レコードと見なしていたので，再ファイル化するときレコード区切り語は，検索時と同じ復改記号を指定しなければならない．
(32) LISTコマンドを用いると，BUNKEN2 は意図したファイルになっていることを確かめることができる．

## ２．４ 文献型データにおける並べ換え

この２．４ と次の２．５ では，今回の改訂で強化されたSORTコマンドの主な機能を，いくつかの例により紹介する．まず，ここでは，表形式で表現されている文献型データを対象にした例で，ソーティングの基本的な機能を示す．次の使用例５は，文献型データにおいて著者名を第１キー，出版年を第２キーとしてレコードの並べ換えを行ったものである．

使用例５（レコードの並べ換え）

```
(1) DO:SORT
        RECORD DELIMITERS
(2) DO1:=$
    DO2:=↙
        KEYWORDS
(3) KO1:=F/'(AR)'/U/''/,G/5KJ/,O(BO)
(4) KO2:=F/'(MS)'/,G/4N/,O(BO)
    KO3:=↙
(5) REFILE UNIT:=R
(6) NEW DELIMITER:=$
(7) INPUT-FILE:=S.RIFISLIB.JPN.TEXT
(8) OUTPUT-FILE:=↙
(9) LISTING(N/Y)?Y
(10)    EXTRACTED KEYS
                1965
    近藤文治  1967
                1968
    甘利俊一  1973
      ……
    野崎昭弘  1975
      LISTING FINISHED
      CPU (SEC/1000) = 581
(11) INPUT-FILE:=↙
(12) DO:LOOK
     $
     (NO) Ko982/(3-3)
     (TI) 医学サイバネティックスの展開
     (AR) 阿部裕　古川俊之　Bellman,R.　大島
     正光　渥美和彦
     (ED) 北川敏男
     (JP) 学習研究社，東京
     (MS) J 1973 P231　（第３巻　講座　情報
     社会科学；生命と情報　Ⅲ）
     $
     (NO) Ko982/(11-2)
     (TI) 創造活動と思考革命
     (AR) 稲山貞登　道家達将
     (ED) 北川敏男
     (JP) 学習研究社，東京
     (MS) J 1975 P198　（第11巻　講座　情報
     社会科学；知識と思考革命　Ⅱ）
     $
       ……
     DO:
```

使用例５の説明

(1) SORTコマンドを投入する．
(2) レコード区切り語に“$” を指定する．“$” で区切られた文字列を１つのレコードとみなして，各レコードのソートを行う．

(3) 第1キーの切り出し仕様を指定する．“(AR)”を見つけた(Find)後“(” まで (Until)の間（切り出し範囲）で，漢字 (KanJi)5文字を切りとる (Get)．この場合，“(AR)”に続く文字列において最初に出現する漢字からキーの切り出しを始め，5文字分全部漢字が見つかるか，または5文字以内で漢字以外の文字が現れると切り出しが終わる．これで，最長5文字までの漢字の著者名を切り出すことになり，3文字や4文字など5文字に満たない著者名も切り出される．オプション(Option)として，キーが切り出されなかった場合（切り出し範囲内に漢字がなかった場合）には，そのレコードを全体の最後(BOttom)に置くことにする．

(4) 第2キーの切り出し仕様を指定する．“(MS)”を見つけた(Find)後，半角の数字 (Numeral)を最長4文字可能な限り取ってくる (Get)．これで発行年を切り出す．これも，オプション(Option)として，キーが切り出されなかった場合には，そのレコードをリストの最後(BOttom)に置くことにする．これにより，まず第1キーの著者名でソートし，次いで第2キーの発行年でソートすることになる．

(5) 再ファイル化仕様（後で指定するファイルにどのようなレコード単位で書き込むか）を指定する．“R” または復改のみを入力すると，レコード単位で再ファイル化を行う．

(6) (5) でレコード単位での再ファイル化を指示したので，新しいレコード区切り語を入力する．ここでは，前と同じ“$” を指定している．

(7) ソートの対象となるファイルとして，SIGMA 領域にある S.RIFISLIB.JPN.TEXTを指定している．

(8) 再ファイル化の出力先に，作業ファイル Wを指定している．なお復改のみを入力すると Wを指定したことになる．

(9) 切り出されるキーを，そのつど順次表示するように指定している．“N” または復改のみを入力すると表示しない．

(10) (7) の指示によって，切り出されたキーが次々に表示されている．第1キー（著者名）が欠落しているときは，漢字を表すシフトコードの関係で，第2キー（発行年）が右にずれている．

(11) 再び入力ファイルが要求される．ここでは，復改のみを入力してSORTを終了させている．まず，小さなテストファイルに対して，(9) のLISTING を“Y” としたソートを行い，その結果を確かめて目的のファイルのソートに移るとよい．

(12) LOOKコマンドで作業ファイルW の内容を表示している．全角文字列のソートも辞書を使わず漢字コード順で行なう．第1水準漢字は，ほぼ音読みの順であり，この例では阿（ア），穐（アキ），…の順番となっている．したがって，漢字列をソートする場合には，あらかじめ別にふりがなの項目を設けておいて，そこからキーを切り出すなどの工夫が必要である．なお，全角文字列の先頭にはシフトコードが入り，このコードは半角の英字コードよりも小さい．よって，切り出されたキーに全角と半角の文字列が混在する場合の昇順ソートでは，全角文字列がキーのレコードは半角文字列がキーのものより先に出現する．

## 2．5 索引，逆引き辞書の作成

前節のソーティングではレコードを対象としたが，論文の推敲や言語解析などでは，単語を単位としたソーティングが必要となる．ここでは，一般のテキストデータベースを対象として，単語の頻度表と索引，および逆引き辞書の作成を例をあげて示す．

単語の頻度表は，いわゆる言語解析の目的で単語の使用統計をとるだけでなく，頻度1の単語の綴りを調べることによって，綴り文字の間違い検出や論文の推敲にも役に立てることができる．また，逆引き辞書は語尾をそろえて右からソートした辞書で，同じ語尾変化の単語がまとめられるため，語

尾処理などの言語解析に利用できる.

使用例６（単語の頻度表作成）

```
    DO:SORT
        RECORD DELIMITERS
(1) DO1:=↙
        KEYWORDS
(2) KO1:=G/15KK/,O(SE)
(3) REFILE UNIT:=#K1
    INPUT-FILE:=S.DRAFT.TEXT
    OUTPUT-FILE:=↙
    LISTING(N/Y)?↙
       CPU (SEC/1000) = 343
```

```
INPUT-FILE:=↙
DO:LOOK
69 ファイル
35 コマンド
34 システム
   ……
 1 ロシア
 1 ワイルドカード
DO:
```

使用例６の説明

(1) 復改のみを入力しているので，レコード区切り語をまったく登録しないことになり，ファイル全体が一つのレコードとみなされる.

(2) 切り出し仕様として，全角のカタカナ(KataKana)15文字を指定し，最長15文字までの連続した全角カタカナ文字列を切り出す(Get) ことにする．さらにオプション(Option)として，切り出されたカタカナ文字列を一つずつ(SEparately)そのまま一つのキーとするよう指定している．使用例５のように“SE”を指定しない通常のソートでは，各キーの切り出し仕様に基づいた切り出しが一つのレコードで１回だけ行なわれる．そして，一つの切り出し仕様に複数の Gがあれば，これらで切り出される文字列がすべて連接されて，一つのキーとなる．これに対して“SE”を指定すると，指定された切り出し区間の中で，G で指定された文字列を，可能な限り何回も切り出す．このとき，切り出された文字列が重なることはない．そして，切り出された文字列は，一つ一つがキーとしてソートされる．したがって，文章中の単語を拾いだす場合などに便利である．なおこの場合，切り出された文字列一つ一つをキーとするため，第２キー以降は指定できず，プロンプトも出ない.

(3) 再ファイル化仕様として“#K1” を指定し，第１キーを頻度で再ソートして，頻度にキーが続く形でファイル化するよう指示している.

使用例７（索引の作成）

```
    DO:SORT
        RECORD DELIMITERS
(1) DO1:=. !
    DO2:=↙
        KEYWORDS
(2) KO1:=G/15KK/,O(SE)
(3) REFILE UNIT:=K1L
    INPUT-FILE:=S.DRAFT.TEXT
    OUTPUT-FILE:=↙
    LISTING(N/Y)?↙
       CPU (SEC/1000) = 250
```

```
INPUT-FILE:=↙
DO:LOOK
アイテム                    47,47,50
,50,71,71,81
アクセス                    62
アルゴリズム                1,1,12,5
7,58,58,59,59,84,84
   ……
ワープロ                    2,9,66
ワイルドカード              49
DO:
```

使用例７の説明

(1) レコード区切り語として，全角のピリオド“．”に続く復改コード（“!” は復改コードを表

す代用記号）を指定している．これで，通常の日本語論文では，パラグラフを一つのレコードとみなしたことになる．

(2) 切り出し仕様として，使用例6と同様に全角のカタカナ(KataKana)15文字を指定し(Get)，さらにオプション(Option)にも，同じく“SE”を指定している．

(3) 再ファイル化仕様として“K1L”を指定して，第1キーとその後ろにそれが現れたレコード番号のリストを付ける形でファイル化するように指示している．

使用例8（逆引き辞書の作成）

```
    DO:SORT
        RECORD DELIMITERS
    DO1:=$
    DO2:=↙
        KEYWORDS
(1) KO1:=F/'(TI)'/U/'('/,G/20A/,0(SE,RJ,RE)
(2) REFILE UNIT:=K1#
    INPUT-FILE:=S.RIFISLIB.ENG.TEXT
    OUTPUT-FILE:=↙
    LISTING(N/Y)?↙
```

```
    CPU (SEC/1000) = 3199
INPUT-FILE:=↙
DO:LOOK
                    A 120
                 CIBA 1
               AMERICA 3
                  ·····
             CONVEXITY 1
           UNCERTAINTY 4
DO:
```

使用例8の説明

(1) 切り出し仕様を指定している．“(TI)”を見つけた(Find)後“(”まで(Until)の区間で，半角の英字(Alphabet)20文字を可能な限り取ってくる(Get)．オプション(Option)として，切り出されたカタカナ文字列を一つずつ(SEparately)そのまま一つのキーとするよう指定して，切り出されたキーは右寄せ(Right Justify)して，さらに右から逆順にソートする(REverse)ことを指示している．

(2) 再ファイル化仕様“K1#”は，第1キーとその後ろに頻度をつける形を指示する．

## 3. SIGMA の利用法

### 3．1 キー入力に関する注意

本節では，コマンドやオペランド類の入力方法と，入出力の制御に使われている制御文字について解説する．

SIGMA を起動すると，“SIGMA)”と表示される．この状態（ SIGMA状態という）では，SIGMA システムのすべてのコマンドを入力することができる．コマンドを入力すると，そのコマンドにオペランドが必要な場合にはオペランドの入力を促すプロンプトが表示されるので，それに従って入力すればよい．一つのコマンドの処理が終わると，“DO:” と表示される．この状態（DO状態という）では，SETMEMO とSETSIGMA以外のすべてのコマンドを入力することができる．（ SIGMA状態とDO状態の違いについては3．2 の“LOG ファイルの利用”を参照のこと）

コマンドやオペランド類を1個ずつ入力していくと，システムは次々とプロンプトを出力し利用者からの入力を促すが，半角の空白で区切ることにより1行に続けて入力することもできる．1行に複数の入力をした場合には，途中のプロンプトは出されず，次に入力が必要となったときにプロンプトが出される．

例えば，SAMPLE.TEXT のコピーをSAMPLE2.TEXTに作るときは，次の(1)～(3)のいずれでもよい．

(1) DO:COPY
    FROM-FILE:=SAMPLE.TEXT
    TO-FILE:=SAMPLE2.TEXT

(2) DO:COPY SAMPLE.TEXT
    TO-FILE:=SAMPLE2.TEXT

(3) DO:COPY SAMPLE.TEXT SAMPLE2.TEXT

ただし，空白が意味をもつような文字列を入力するときは1行に続けて書くことができず，続けて入力した文字列も含めて一つの入力とみなされる．例えば，あるファイル中の“テキスト”という文字列を“文章”に置き換えようとする場合，(4) は正しい入力であるが，(5) と (6)は，“テキスト 文章” という文字列を置き換えるという指示になってしまい誤りである．なお，空白が意味を持つのは，REPLACE コマンドの置き換え文字列の入力時やSEARCH，SORTコマンドの区切り語，キーワードの指定時などである．

(4) DO:REPLACE
    XO1:=テキスト
    YO1:=文章
    XO2:=

(5) DO:REPLACE テキスト 文章
    YO1:=

(6) DO:REPLACE
    XO1:=テキスト 文章
    YO1:=

コマンド名の入力に際しては，他のコマンドと区別のつく部分までを入力すれば十分である．たとえば，L で始まるコマンドはLISTとLOAD，LOOKの三つがあるが，LISTに対してはLI,LIS,LIST のいずれでもよい．また，LOOKに対してはLOO,LOOKのいずれでもよい．

コマンド類の入力時に復改のみを入力すると，状況に応じて次のように解釈される．

(A) 入力が無効であるので再入力を促す．

(B) 入力の終わりを示した．

(C) 省略値を示した．

（例）

(A) DO:↙
DO:LIST
FILE:=SAMPLE.TEXT
これは，テスト用のテキストです．
DO:REP　　（REPLACE とすべて入力する必要はない）
X01:=テキスト
Y01:=文章

(B) X02:=↙
INPUT-FILE:=SAMPLE.TEXT
OUTPUT-FILE:=SAMPLE2.TEXT

(C) REPORT(N/Y)?↙

(C) のREPORT(N/Y)?のような(N/Y)?における選択は，省略すると前の文字が選ばれる．この場合はN が選ばれる．プロンプトが(N OR Y)? のように選択枝がORで結ばれている場合は省略値はなく，復改のみを入力はどちらも選ばれない．詳細は各コマンドに対する説明を参照されたい．

利用者が入力した文字列は，通常はそのままコマンドやオペランド類として解釈されるが，次に述べるいくつかの文字は特殊な処理をするための制御文字として使われているので注意が必要である．

(1) ハイフン (-)

文字列の末尾が “-” であるとき．その行の入力がまだ続くと解釈され継続処理が行われる．KEYIN コマンドによる端末からの入力のときには継続処理はいくらでも行うことができるが，他のコマンドやオペランド類の入力中には継続した全長が500 文字までという制限がある．

“-” が，継続文字としてのシステムの初期設定であるが， PROFILEコマンドによって他の文字に変更することができる．

次の(2)～(4)は，プロンプトが “K)” 以外のときに有効である．

(2) 感嘆符 (!)

SIGMA システムでは，復改記号も一つの文字として扱っている．復改記号は端末から入力することができないので，復改記号を表すために “!” を用いている．入力文字列に含まれる “!” は復改記号で置き換えられる．

代用復改記号のシステム初期設定は “!” であるが，これはPROFILE コマンドによって他の文字に変更することができる．代用復改記号として指定した文字は，自動的に復改記号に置き換えられるので，通常使わない文字に変更されたい．

(3) 疑問符 (?)

文字列が “?” で始まるとき．

? または ?.n (n=1～) を入力すると，そのときのプロンプトに応じたヘルプメッセージが表示される．n は，出力されるヘルプメッセージのレベルを表す．レベル１は英語，レベル２は日本語のメッセージである．ふつうレベルは１か２であるが，プロンプトによっては（SORTコマンドのキーの切り

出し時など）レベル３のメッセージも用意されている．n=3 を指定したときに，レベル３のメッセージがない場合は，レベル２のメッセージが出力される．

なお，？一文字のときのメッセージのレベルは，PROFILE コマンドによって設定された値である．？の後の文字が，ピリオド＋数字でない場合は入力エラーとなる．

先頭が ?で始まる文字列を入力したいときは，?を重ねればよい．

(4) ピリオド（.）

文字列が“.” で始まるときには，後に続く文字によって次の処理が行われる．

(A) 以前に端末から入力した文字列を再入力する．

(B) 端末からの入力をファイルからの入力に切り替える．

(C) 入力がファイルからのとき，その内容を表示するかどうかを切り替える．

なお，先頭が “.” で始まる文字列を入力したいときは，“.” を重ねればよい．

(A) 入力文字列の履歴がとられているので，以前に入力した文字列を再利用することができる．

.= を入力すると直前に入力した文字列が再入力される．

.=.n（n=1～10）を入力すると n個前の文字列が入力される．

.=.L を入力すると入力文字列の履歴をみることができる．

なお，これらの文字列の履歴はとられない．また，コマンドやオペランド類を１行に続けて入力したときには，その行すべてが一つの履歴としてとられるのではなく，各コマンドやオペランド類ごとにとられる．

（例）

| | |
|---|---|
| (1) SIGMA)LIST A-LONG-FILE-NAME-1 | (1) A-LONG-FILE-NAME-1の内容をみる |
| This is a test file 1. | (2) 入力文字列の履歴をみる（LISTとA-LONG |
| (2) DO:.=.L | -FILE-NAME-1の履歴は別々にとられている）． |
| 2 LIST | (3) ２個前の入力文字列（LIST）を再入力す |
| 1 A-LONG-FILE-NAME-1 | る． |
| (3) DO:.=.2 | (4) １個前の入力文字列（A-LONG-FILE-NAME |
| (4) FILE:=.= | -1）を再入力する． |
| This is a test file 1. | (5) 入力文字列の履歴をみる（.=や.=.Lなど |
| (5) DO:.=.L | の履歴はとられていない）． |
| 2 LIST | (6) “COPY” の入力で履歴が１個ずれるので， |
| 1 A-LONG-FILE-NAME-2 | .=.2は“A-LONG-FILE-NAME-1”をさしている． |
| (6) DO:COPY .=.2 NEW-FILE | よって，A-LONG-FILE-NAME-1のコピーをNEW- |
| DO: | FILEに作ることになる． |

このように，長い文字列を再入力するのに便利である．また，LOG ファイルで特に効力を発揮する．

(B) SIGMA システムでは，コマンドやオペランド類の入力を端末からだけでなく，ファイルからも行うことができる．ファイルからの入力は，

.filename　または　./.filename

によって行う．

.filename と ./.filenameの違いは，filenameからの入力時に，プロンプト込みで入力するかどうかである．“./.”をつけると，そのファイルにはプロンプトが書かれていないとみなされる．

ファイルからの入力についての具体的な利用法は，３．２ の“LOG ファイルの利用”を参照され

たい.

(C) .#を入力すると，現在の状態がエコーオン（入力がファイルからであるとき，そのファイルの内容を表示する）であればエコーオフ（ファイルの内容を表示しない）に，エコーオフであればエコーオンの状態になる.

## 3．2 LOG ファイルの利用

SIGMA システムでは，利用者との交信記録（ LOG）がMEMO領域の LOG区域に自動的に作られる（領域や区域については，3．4．1を参照のこと）．こうしてできたファイルを LOGファイルという．LOG ファイルは，他のファイルと同様に文字列の形をしたもので，内容はシステムが端末に出すプロンプトとそれに対する利用者の応答の記録である． 3．1 で述べたように，コマンドやオペランド類の入力をファイルから行うこともできるので， LOGファイルをそのまま，あるいは変更を加えて一種のコマンドプロシジャとして利用できる．SEARCHやSORTコマンドではキーワードや論理式，キーの切り出しなどをきめ細かに指定できるが，定型的なテキストを扱う場合それらを毎回キーボードから入力していたのでは効率が悪い．それらをファイルから入力すれば便利である．以下， LOGファイルとその利用法について述べる.

SIGMA システムには SIGMA状態とDO状態があり，プロンプトはそれぞれ“SIGMA〉”，“DO:” となっている．SIGMA システムが呼び出されると，まず SIGMA状態になり，END 以外のコマンドを投入すると， LOGがとられ始める．SIGMA 状態でのコマンドの処理が終わると，DO状態になる．この状態では，SETMEMO とSETSIGMA以外のコマンドが使用できる.

DO状態で ENDコマンドを投入すると，ここまでが1回の交信として LOG区域のトップに置かれ，SIGMA 状態になる．ここで ENDコマンドを投入すればSIGMA システムが終了するし，他のコマンドを投入すれば次の交信がとられる（図1参照）．システムを終了するまで SIGMA状態に戻らなければ， LOGファイルは1個しかできないが，途中で ENDコマンドにより SIGMA状態に戻ればその度ごとに LOGファイルができる．次に述べる LOGファイルの再利用という点からみれば，あるまとまりごとに SIGMA状態に戻ったほうがよいだろう.

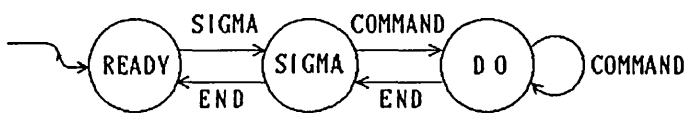



図1． SIGMA システムにおける状態遷移

次に，こうしてできた LOGファイルを再利用した例をあげる.

（例1）

| | |
|---|---|
| (1) READY | (1) TSS の READY状態で SIGMAを起動する. |
| SIGMA | (2) プロンプト“SIGMA〉”が表示される. |
| (2) SIGMA〉LIST | ここから LOGがとられ始める. |
| (3) FILE:=W.2 | (3) 作業区域の2番目のファイルの内容をみる. |
| This is a text. | |
| (4) DO:END | (4) DO状態になる．END コマンドを投入する. |
| (5) SIGMA〉LIST L | (5) 状態はSIGMA 状態になり，今までの交信 |

```
    DO:LIST          ┐
    FILE:=W.2        ┘ Lの内容
    DO:END
(6) DO:.L
(7) DO:LIST
    FILE:=W.2
(8) This is a text.
(9) DO:END
(10) DO:MOVE L LOOK2
(11) DO:END
(12) SIGMA)END
```

記録がLOG 区域トップにおかれる．LISTコマンドでLOG ァイルの内容をみる．
(6) 入力を端末から先ほどできた LOGファイルに切り替える．
(7) ファイルから入力するとき通常は内容が表示される．
(8) 作業区域の２番目のファイルの内容が表示された．
(9) ファイルから ENDが入力され，入力はもとの端末に戻る．
(10)今できた LをLOOK2というファイルに移動する．
(11)END コマンドを投入する．
(12)交信内容が再び LOG区域のトップにおかれ，SIGMA 状態になる．END コマンドを投入して，SIGMA を終了する．

上の例で，“MOVE L LOOK2”により LOGファイルを LOOK2というファイルに移動している．よって，次からは“.LOOK2”により作業区域の２番目のファイルをみるコマンドとして利用できる（なお，作業区域の１番目のファイルをみるコマンドはLOOKである）．

ファイルからの入力の切り替えは .filename や ./.filename の入力によって，プロンプトが“K)”のときと次の (3)を除く任意の時点で行うことができる．ここで用いられるファイル名には，通常の SIGMAのファイル名の他に，特別な働きをする次の４つがある．

(1) .K　以後の入力を端末に切り替えると同時に，新しく LOGファイルの作成を開始する．この中で入力を別のファイルに切り替えることもできる．
(2) ./.K　働きは (1)と同じだが，システムからのプロンプトは LOGにとられない．
(3) .K.1　次の１行だけを端末に切り替える．このときの交信は記録されない．この時点で入力を別のファイルに切り替えることはできない．
(4) .E　現在の入力を終了させ，もとの入力の流れに戻す．その際，現在の入力が端末からのものであれば，その LOGファイルを閉じて LOG区域のトップに置く．

（例２）

```
(1) SIGMA)REPLACE
(2) X01:=.K
(3) X01:=A
    Y01:=A
    ·····
    X26:=Z
    Y26:=Z
(4) X27:=.E
(5) X27:=↙
(6) INPUT-FILE:=SAMPLE1.TEXT
```

(1) ここから LOGがとられる．
(2) ふつうREPLACE コマンドで置き換える文字列は直接入力するが，置き換え文字列だけを保存するために.Kにより新たに LOGを開く．
(3) プロンプトにしたがって文字列を入力している．ここでは，全角の英字を半角文字に置き換えようとしている．
(4) .Eの入力により端末からの入力を終えると，LOGが閉じられ，LOG区域のトップにおかれる．

```
    OUTPUT-FILE:=W                          (5) 入力がもとの端末に戻るが，文字列の指
    REPORT(Y/N)?N                           定をやめたいので，復改だけを入力している．
    INPUT-FILE:=↙                          (6) 入出力ファイルや置き換え結果の報告の
(7) DO:END                                  有無を指定し，置き換えを終わる．
(8) SIGMA)LIST L                            (7) END を入力し，SIGMA 状態に戻る．
    DO:REPLACE                              (8) 今とられた LOGファイルの内容をみる．
    X01:=.K                                 ┐
    X27:=                                   │
    INPUT-FILE:=SAMPLE1.TEXT                │ L の内容
    OUTPUT-FILE:=W                          │
    REPORT(Y/N)?N                           │
    INPUT-FILE:=                            │
    DO:END                                  ┘
(9) DO:LIST L.2                             (9) 置き換え文字列を指示したファイルは
    X01:=A                                  ┐ L・2 の内容   L からL.2 に移動している．
    Y02:=A                                  │
    ·····                                   │
    X26:=Z                                  │
    Y26:=Z                                  │
    X27:=.E                                 ┘
(10) DO:REP                                 (10)REPLACE コマンドを入力する．
(11) X01:=.L.2                              (11)全角の英字を半角に置き換える．そのた
(12) X01:=A                                 めに ".L.2" と入力して先ほどの LOGを利用
     ·····                                  する．
     Y26:=Z                                 (12)L.2 の内容が表示されている．
(13) X27:=.E                                (13)ファイルの最後に ".E" と書かれている
(14) X27:=0                                 ので，これによりファイルからの入力が終了
     Y27:=0                                 し，端末からの入力に戻る．
     ·····                                  (14)さらに，全角の数字を半角の数字に置き換
     X36:=9                                 える指定をする．
     Y36:=9
(15) X37:=↙                                 (15)置き換え文字列の指定を終わる．
(16) INPUT-FILE:=SAMPLE2.TEXT               (16)置き換えを行うファイルを入力する．
(17) OUTPUT-FILE:=.=                        (17) ".=" を入力したので，出力ファイル名
     REPORT(Y/N)?N                          は元のファイル名と同じになる．
     INPUT-FILE:=↙
(18) DO:END                                 (18)END と入力すると，SIGMA 状態になる．
     SIGMA)
```

図2は，例2の入力の流れを示したものである．図において，実線は端末からの入力であり，点線はファイルからの入力である．".L.2" は LOG区域の2番目のファイルからの入力を指示しているの

で，実線②における交信記録が入力されることになる．また，最終的に SIGMA状態に戻ったときの LOG区域には上から順に③，①，②での交信記録が作られていることになる．

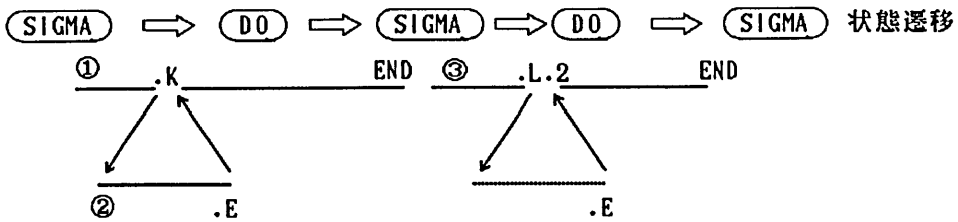



図2．　SIGMAシステムの入力の流れ

上の例では，実際に実行してできた LOGファイルを再利用したが，LOG ファイルは通常のテキストであるので，REPLACE コマンドや外部エディタなどで修正・作成することができる．作成は， LOGファイルにならって行えばよいが，次の点に注意する必要がある．

(1) プロンプトと利用者の入力部分を区別するために，システムはプロンプトのチェックを行っている．よって，ファイル中にもプロンプト込みで書かなければならない．なお，実行中にプロンプトが合わなければエラーとなり，そのファイルからの入力は終わる．

(2) ファイルからの入力を終了し前の入力に戻すために，ファイルの終わりにプロンプトと.E（プロンプトが“DO:” ならば，END でもよい）をつけなければならない．

(1) のプロンプトのチェックに関しては，次のように制限を少し緩めている．

(3) プロンプトの先頭に空白がある場合には，それを無視して書いてよい．

(4) プロンプトに数字がふくまれているときは，その部分をチェックしていない．よって，例2における REPLACEコマンドの文字列の入力に使ったファイルは図3(a) のような形をしていてもよい．また，“./.filename” により，プロンプトのない図3(b) のようなファイルから入力を行うこともできる．ふつうは，“.K” により図3(a) の形でLOG がとられるが，“./.K” により図3(b) のような形でLOG をとることができる．

```
X01:=A
Y01:=A
.....
X01:=Z
Y01:=Z
X01:=.E
```

```
A
A
...
Z
Z
.E
```

図3(a)　(b)

ファイルを見やすくしたり，実行中にその内容を表示したりするために，次のような制御文字をファイルに入れることができる．

(5) 行の先頭に %を書くことによって，その行を注釈行とみなすことができる．また，ファイル中に空行があってもよい．

(6) 行の先頭に #を書くことによって，# の後の文字列をメッセージとして出力することができる．また，# の後に ON(OFF)を書くことによって，そのファイルの内容を出力するかどうかを任意の時点で指示することができる．# の働きはECHOコマンドと同じである．

次の例は，LISTコマンドと同じ機能をTYPEというファイルで実現している．

（例3）

```
DO:LIST TYPE
```

```
%これは，LISTコマンドと同じ機能をもつTYPEコマンドである          (注釈文)
#OFF                                                    (エコーオフにする)
DO:LIST
#表示したいファイル名を入力してください                    (メッセージを表示)
FILE:=.K.1                                              (端末から1行入力)
DO:END
DO:.TYPE
表示したいファイル名を入力してください
FILE:=SAMPLE.TEXT
これはテスト用のテキストです
DO:
```

ここでは簡単な例しか示さなかったが，LOGファイルをいくつか組み合わせることによりさらに複雑なコマンド・プロシジャを作成することができる．なお，課題番号A71095B の SIGMA領域に LOGファイルの例をおいているので，各自コピーして参考にされたい．なお，ファイル名はすべて“LOG.”で始まる．

### 3．3　TSSコマンドの呼び出し

SIGMAシステムの中から，TSSコマンドを呼び出すことができる．ただし，LOGOFF,PROFILEなど一部のコマンドは使用できない．TSSコマンドを呼び出すには，まず SIGMA状態かDO状態で“ TSS”と入力して，“TSS:”というプロンプトに続いてコマンドを入力すればよい．たとえば，所有しているデータセット名の一覧を表示するときは，次のようにすればよい．

```
DO: TSS
TSS:LISTC
IN CATALOG:SYS1.EDFUCAT1
A79999A.A.DATA
A79999A.ABC.FORT77
.....................
A79999A.ZZZ.TEXT
TSS:
```

また，“TSS:”のプロンプトからDO状態に戻るには，プロンプトに続いて“ END”か，復改のみを入力すればよい．

### 3．4　ファイルの操作

SIGMA は，ファイルの高速処理および共有データベースの管理のため，独自のファイルシステムを有し，それによってファイルを管理している．また，オペレーティングシステムが管理しているデータセット（外部ファイル）も簡単に参照できる．SIGMA が扱うファイルは，SEARCHコマンドの索引ファイルを除いてすべてテキストファイルである．

#### 3．4．1 領域と区域

SIGMA ではファイルを領域と呼ばれる場所に保管している．領域には，個人用のMEMO領域と，データベースを共有するための SIGMA領域，オペレーティングシステムが管理しているデータセットや端

末・プリンタを SIGMAのファイルとして扱うための仮想的な外部領域がある（図４(a)）．MEMO領域と SIGMA領域は，ファイルを保存するための機能と， SIGMAの利用者の作業環境としての機能を持っている．各領域は次のような構造になっている．さらに細かな解説は，後の節で行う．

(a) MEMO領域には，ファイルの置き場所（区域と呼ぶ）が５つある．（図４(b)）

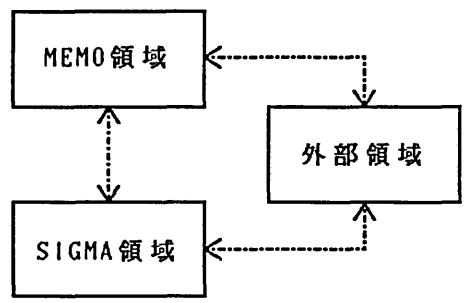
(1) MEMO区域(略号MEMO)：個人が保存したいファイル（MEMOファイルと呼ぶ）を保管する場所である．ファイルには名前をつけて保管する．区域内のファイル数は最大 463個である．

(2) 作業区域（W)：個人の作業場所であり，作業の対象となるファイルを置くための場所である．底のあるスタック構造をしている．区域内のファイル数は最大20個である．ファイル数が20を越えると，底のファイルがバックアップ区域に移動する．

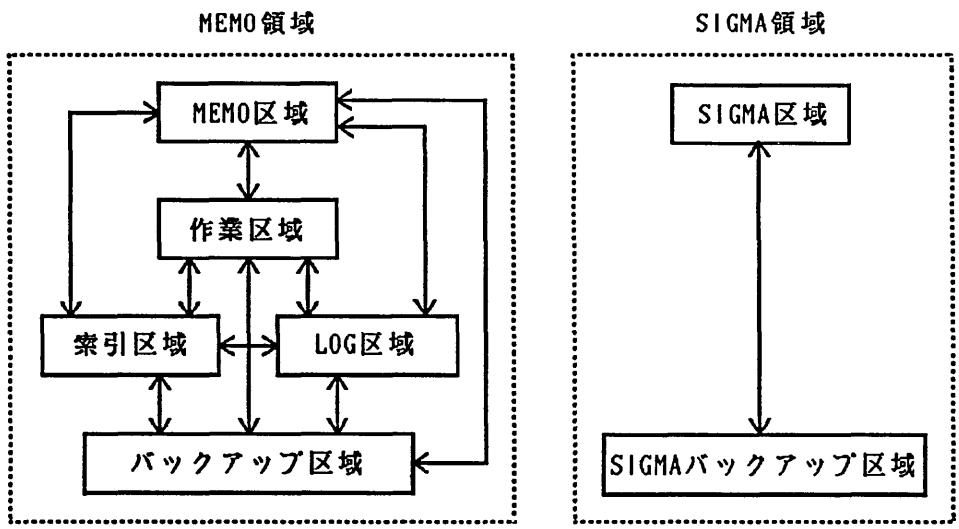
(3) LOG区域（L)：LOGファイルが置かれる場所である．作業区域と全く同じ構造である．

(4) 索引区域（I)：検索コマンドの索引ファイルが置かれる場所である．作業区域と全く同じ構造である．

(5) バックアップ区域（B)：上の４区域で不用になったファイル（バックアップ・ファイルという）を置くための場所である．底のあるスタック構造をしている．区域内のファイル数は最大 900個である．ファイル数が 900を越えたり，MEMO領域全体の容量が不足すると，底のファイ




(a)




(b)　　(c)

図４．　SIGMAファイルシステム
（破線の矢印はCOPYによるファイルの複製が可能であることを表し，
実線の矢印はMOVEによるファイルの移動も可能であることを表す）

ルが再利用のために消えてしまう．

(b) SIGMA領域 は，2つの区域から構成される．（図4(c)）

(1) SIGMA区域 ：複数の利用者で共有するための名前つきのファイル（SIGMA ファイルと呼ぶ）を保管する場所である．区域内のファイル数は最大 463個である．

(2) SIGMA領域のバックアップ区域 ： SIGMA区域で不用になったファイル（バックアップ・ファイル）を置くための場所である．底のあるスタック構造をしている．区域内のファイル数は最大 900個である．ファイル数が 900を越えたり， SIGMA領域全体の容量が不足すると，底のファイルが再利用のために消えてしまう．

(c) 外部領域は1つの区域と特殊ファイルから構成される．

(1) 外部区域 (X)：オペレーティングシステムが管理しているデータセットが置かれている場所である．外部区域にあるファイルを外部ファイルと呼ぶ．

(2) 特殊ファイル：端末をファイルとして取り扱うための端末ファイル(D,K,T)，プリンタをファイルとして取り扱うためのプリンタファイル(P)の二つがある．

SIGMA では，ふだんは作業区域のファイルを用いて，検索，置き換え，ソートなどを行い，個人用に整理したいと思ったときに，名前をつけてMEMOファイルとして，MEMO区域に保管する．他の利用者と共有したいファイルは， SIGMAファイルとして， SIGMA区域に保管する．逆に，他の利用者の持っているSIGMAファイル を利用することもできる．MEMO領域のファイルは，それを所有する利用者が文字通りメモ帳として使うものであるから，他の利用者との共有はできない．不用になったファイルは，各領域のバックアップ区域へ移動する（ファイルの除去という）．したがって，誤って除去してしまったファイルを復元することも可能である．

### 3．4．2 MEMO区域と SIGMA区域

MEMOファイルとSIGMAファイルは保存用ファイルである．MEMO区域と SIGMA区域では，コマンドの出力オペランドにMEMOファイルとSIGMAファイルの名前が用いられたときに，次のようなファイルの操作がなされる．例えば，

COPY DATA COPY-OF-DATA

と入力するとき，出力オペランドCOPY-OF-DATAと同じ名のMEMOファイルが存在していない場合には，DATAと全く同じ内容のファイルCOPY-OF-DATAが作成される．すでに，COPY-OF-DATAという名のファイルが存在している場合には，そのファイルが除去されたあと，新たにCOPY-OF-DATAという名のファイルがつくられる（ファイルのオーバーライトという）．

DELETE COPY-OF-DATA

COPY DATA COPY-OF-DATA

と入力したときと結果は同じになる．なお，コマンドの誤入力などの理由で，バックアップ区域へ移動された古い方のファイルを回復したい場合には，

MOVE B COPY-OF-DATA

と入力すればよい．このとき，再びファイルのオーバーライトが生じ，入れ替わりに新しい方の（複製の方の）COPY-OF-DATAがバックアップ区域に移動される．

### 3．4．3 作業区域， LOG区域，索引区域

これらの区域は底のあるスタック構造をしている．底のあるスタック構造とは，ファイルを一列に

積み重ねて管理する構造であり（２章図１参照），ファイルの基本操作は次のようになっている．
(a) 区域のファイルの名前は，区域の略称に “.” と上からの順位（何番目のファイルか）を用いる（３．４．７参照）．作業区域の上から２番目のファイル名は W.2である．通常， W.1は Wと略記する．
(b) 区域に新たなファイルを作成・移動するときには，そのトップに作成・移動しなければならない．このとき，作業区域にあったファイルの順位は一つずつ繰り下がる．したがって，作業区域のファイルは，作業区域に入ってきた順に並んでいる．
(c) 区域にあるファイルの移動は，自由に行うことができる．ファイルが移動された場合，移動されたファイルより下にあるファイルの順位は繰り上がる．
(d) 新たなファイルの追加によって，区域のファイル数の制限(20)を越えた場合は，底にあるファイルがバックアップ区域へ移動する．

また，

```
COPY DATA W
```

と入力すると，ファイルDATAの複製が，作業区域にあるファイルの列のトップに積み重ねられる．たとえば，コマンド実行前に W.2であったファイルは実行後には W.3で参照しなければならない．

W.3 の内容に名前RESULTを付けてMEMO区域に保管したい場合は，

```
COPY W.3 RESULT
```

と入力すると， W.3と同内容のMEMOファイルRESULTが作成される．一方，

```
MOVE W.3 RESULT
```

を用いると， W.3の内容がMEMOファイルRESULTに移動し， W.3以下の作業区域にあるファイルの順位が繰り上がる．なお，

```
MOVE RESULT W.3
```

は，許されていないので，特に除去したファイルの回復の際には注意が必要である．
(e) さらに，区域によっては，次のような機能が付加されている．
(e-W) 作業区域のトップのファイル Wに対しては，MOVEとCOPYのオペランドを次のように Wに固定した特別なコマンドが用意されている（４章参照）．

```
SAVE file = COPY W file          PUT file = MOVE W file
LOAD file = COPY file W          GET file = MOVE file W
KEYIN = COPY K W
```

(e-L) LOG区域では，DO状態からSIGMA状態へ戻ったときに， LOGファイルがトップに積み重ねられる（３．２ 参照）．
(e-I) 索引区域では，SEARCHコマンドを使用したとき，索引ファイルがトップに積み重ねられる（３．５．８参照）．

### ３．４．４ バックアップ区域

バックアップ区域は，MEMO領域と SIGMA領域にあり，それぞれの領域のくずかごとして機能する．底のあるスタック構造をしていて，３．４．３の(a),(b),(c) と，次の(d'),(e-B)の機能を持っている．
(d') バックアップ区域は，ファイル数が制限( 900)を越えると，スタックの底にあるファイルが実際に消去される．さらに，領域全体で容量が不足した場合には，その領域のバックアップ区域のファイルが底から順に実際に消去されて，必要な空きが確保される．
(e-B) MOVEの２番目のオペランドをB.1 かS.B.1 に固定したDELETEが用意されている．また，MEMOフ

ァイルとSIGMA ファイルのオーバーライトが生じた場合，オーバーライトされた古い方のファイルはバックアップ区域のトップに積まれる（3．4．2参照）．さらに，作業区域，LOG区域，索引区域のファイル数が制限(20)を越えた場合，スタックの底にあるファイルがMEMO領域のバックアップ区域のトップに積まれる（3．4．3参照）．

これらの操作はコマンドの実行中にすべて自動的に行われる．バックアップ区域には，容量に余裕がある限り，除去されたファイルが保存されていることになって，操作ミスによるデータの損失を防いでいる．例えば，MEMOファイルDATAを誤って除去した場合には，

MOVE B DATA

によって回復が可能である（3．4．2の最後の注意も参照）．

### 3．4．5 SIGMA 領域の作成・消去・アクセス権の設定，領域の拡張

SIGMA 領域は複数の利用者の共有データベースであるので，アクセス権を持っていれば，他人のSIGMA 領域に書き込むことも可能である．図5において，破線で囲まれた部分が1人の利用者のMEMO領域とSIGMA 領域を表している．USER2,USER4 はMEMO領域しかもたない利用者である．矢印はアクセス権（READ/WRITE）を示しており，矢印の向きに，ファイルの転送が可能である．下図では，USER2 はUSER1 のSIGMA ファイルをリード・アクセスもライト・アクセスもできるが，USER3 のSIGMA ファイルはリード・アクセスしかできない．

利用者は，同時に二つ以上のSIGMA 領域をアクセスすることはできない．例えば，

COPY S.'A79998A.TEST.DATA' S.'A79999A.TEST.DATA'

のように，課題番号A79998A のSIGMA 領域から，A79999A の課題のSIGMA 領域へファイルを直接複製することなどは禁じられている．

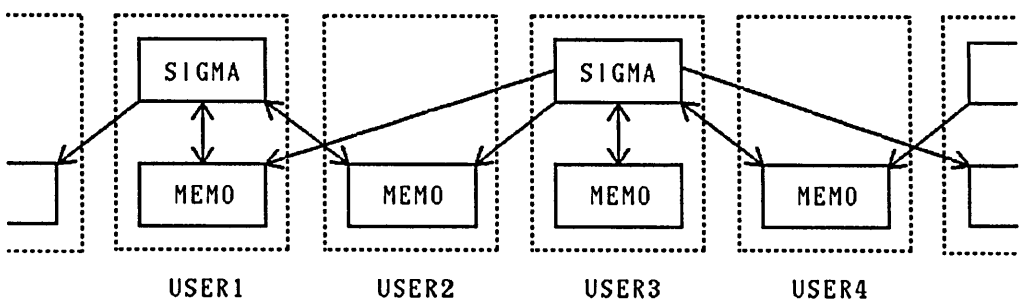



図5．　SIGMA領域の共有状態

SIGMA 領域を作成するには，READY モードで，

SIGMA SPACE

と入力すると，MEMO領域を作るかSIGMA 領域を作るかを尋ねてくるので，SIGMA と入力する．以降SIGMA 領域の大きさやマスターキーを尋ねてくるので，それに答えれば作成は終了する．SIGMA 領域は，SIGMA87.DATAという名前のデーターセットに固定してあるので，その消去は READYモードで

DEL SIGMA87.DATA

とすればよい．

SIGMA 領域のアクセスを他の利用者に許可するのは，TSS のPERMITコマンドを使えばよい．たとえば，ユーザーA79999A に書き込みを許可するのは， READYモードで，

PERMIT SIGMA87.DATA ID(A79999A) ACCESS(UPDATE)

とすればよい．なお，SIGMA 領域を作成した時点では，すべてのユーザーが読み出し可能になっている．

MEMO領域もSIGMA 領域も拡張できる．SIGMA システムでは，バックアップファイルを実際に消去することによって，領域の再利用をするが，バックアップ区域やSIGMA バックアップ区域にファイルがなくなったときには，新しくファイルが作れなくなる．このときは，不要なファイルを除去するか，領域の拡張をする必要がある．拡張をするには，READY モードで，

SIGMA EXTEND

とする．続けてMEMOかSIGMA かを入力し，SIGMA のときは，その課題番号を入力する．以降，拡張するブロック数とマスターキーを入力すればよい．ただし，SIGMA 領域の拡張には，そのデータセットに対して，ALTER 権が必要である．

### 3．4．6 外部領域

SIGMA の利用中に，オペレーティング・システムが管理しているデータセットを読む場合は，それが外部区域にあるファイル（外部ファイル）であるものとして取り扱うことができる．外部ファイルの内容をMEMO領域に転送するには，COPYまたはLOADコマンドを用いればよい．ファイル名は，“X.”の後にデータセット名をつけて表わす．例えば，データセット名がDATA.DATAであるようなファイルを作業区域へ転送したいときは

LOAD X.DATA.DATA

とすればよい．また，データセットの行末に“-” があれば，その行に次の行が継続される．“-”はPROFILE コマンドで変更可能である．

外部領域には，特殊ファイルとして，端末ファイルとプリンタファイルが用意されている．この二つのファイルについて説明する．

一般に，SIGMA のコマンドが投入されると，あるファイルからテキストが読み出され，何らかの処理がなされ，ファイルに書き込まれる．データが書き込まれたファイルは次に書き込まれるまで，そのデータを保存している．また，SIGMA では端末からテキストを読み込んで処理をしたり，処理の結果を端末へ書き出したりする．したがって，端末もテキストの読み書きという点だけから見れば，ファイルと同じ機能を持っている．そこで，端末もファイルとして，コマンドのオペランドに使えるようにしている．このファイルを端末ファイルという（ファイル名はD,K,T のいずれでもよい）．例えば，

COPY DATA K

と入力すればDATAというファイルの内容が端末に表示される．プリンタファイル（ファイル名 P）も同様の考え方によっている．もちろん，プリンタファイルは，データの書き込み専用であり，データの読み出しはできない．

外部領域にあるファイルに対してはMOVE系のコマンドは使えない（3．4．10参照）．また，自動的にバックアップ・ファイルを作る機能もない．

### 3．4．7 ファイル名規約

ファイル名は，最大40文字の英数字および記号からなる文字列であるが，いくつかの制約があるので，まとまりごとに分けて説明する．

(1)MEMO領域のMEMO区域

'$X_1.X_2.\cdots.X_n$'($n \geq 1$)の形をした任意の英数字および記号の列である．ただし，各$X_i$は2文字以上である．PROFILE コマンドで指定するMEMO区域のプレフィックスを$X_1.\cdots.X_m$($1 \leq m < n$)とすれば，$X_{m+1}.\cdots.X_n$で'$X_1.X_2.\cdots.X_n$'を表すことができる．'$X_1.X_2.\cdots.X_n$'を完全名と呼び，$X_{m+1}.\cdots.X_n$を部分名と呼ぶ．たとえば，'EXAMPLE10.TEXT'，BUNKEN.DATA.1-2，(PARENTHESIS).TEXT などは，正しいファイル名であるが，'W.A1.DATA'，M.123.TEXT，ABC.A.DATA などは正しくない．

(2)MEMO領域のスタック区域

W,L,I,B かあるいは，その後にピリオドをつけ数字を並べたもの．たとえば，W.10,L.2,I.1,B.900などは，ファイル名として正しい．それぞれ，作業区域の10番目，LOG 区域の2番目，索引区域の1番目，バックアップ区域の900 番目のファイルを表している．しかし，W.2.2,L.-1などはファイル名として正しくない．また，W.1,L.1,I.1,B.1は，それぞれW,L,I,B と省略して書くこともできる．ただし，DIRECTORY コマンドとDDIRECTORYコマンドにおいてファイル名を入力するときは，他の箇所で入力するのとは異なっている．W.1 のように.1をつけると区域の中の1番目のファイルを指すことになるが，W だけだとその区域にあるすべてのファイルの一覧をとることになる．

(3)SIGMA 区域

S.'$X_1.X_2.\cdots.X_n$'($n \geq 2$)の形の任意の英数字および記号の列である．ただし，$X_1$はSIGMA 領域の課題番号，$X_i$($2 \leq i \leq n$) は2文字以上である．MEMO区域と同様に，SIGMA 領域のプレフィックスを$X_1.\cdots.X_m$($1 \leq m \leq n-1$) とすれば，$S.X_{m+1}.\cdots.X_n$でS.'$X_1.X_2.\cdots.X_n$'を表すことができる．S.'$X_1.X_2.\cdots.X_n$'を完全名と呼び，$S.X_{m+1}.\cdots.X_n$を部分名と呼ぶ．たとえば，S.'A79999A.EXAMPLE.TEXT'，S.ABC-XYZ.TEXTなどは，正しいファイル名である．

(4)SIGMA 領域のバックアップ区域

S.B.'P.n' の形で，P はSIGMA 領域の課題番号，n はSIGMA バックアップ区域の中の番号である．SIGMA 領域のプレフィックスが$P_1.P_2.\cdots.P_n$であれば，S.B.n でS.B.'$P_1$.n'を表わすことができる．(2) と同様に，DIRECTORY,DDIRECTORYコマンドにおいては注意が必要である．

(5)外部区域

X.の後にオペレーティングシステムのデータセット名として許される文字列をつけたもの．たとえば，X.EXAMPLE.TEXT，X.'A79999A.EXAMPLE.FORT77(MEMBER)'などは正しいファイル名である．

(6)特殊ファイル

一文字の名前で，D,K,T,P がある．D,K,T は端末を表わすファイルで，入力の時にはキーボード，出力の時にはディスプレイを表している．またP はプリンターを表し，出力用にしか使えない．プリンターの出力クラスは，PROFILE コマンドで変更できる．また，P.S （S は出力クラスを表わす文字）で，直接出力クラスを指定できる．

MEMO区域やSIGMA 区域のファイルを指定するには，完全名よりも部分名で指定する方が，簡単である．また，プレフィックスをうまく用いると階層化ディレクトリ構造に似たファイル管理もできる．

### 3．4．8　パスナンバーの設定と解除

MEMO区域のファイルと SIGMA区域のファイルには，誤消去防止とファイル保護・ファイル管理のた

めにパスナンバーをつけることができる．パスナンバーは 0から9999までの４桁の正整数で， 1から9999まではパスナンバーで保護されている状態， 0はパスナンバーで保護されていない状態である．ファイルが最初に作られたときは，パスナンバーで保護されていない状態（つまりパスナンバーは 0）である．パスナンバーの設定，解除，変更にはPASSNUMBERコマンドを使う．以下に例を示す．

```
DO:LIST EXAMPLE.TEXT
THIS IS AN EXAMPLE TEXT.
DO:
```

ここでは，LISTをとるときにパスナンバーを聞いてきていないので，EXAMPLE.TEXTはパスナンバー保護されていないことがわかる．このファイルにパスナンバーをつけてLISTしてみる．

```
DO:PASSNUMBER
FILE:=EXAMPLE.TEXT
NEW PASSNUMBER FOR EXAMPLE.TEXT:=123
DO:LIST EXAMPLE.TEXT
PASS NUMBER FOR EXAMPLE.TEXT:=123
THIS IS AN EXAMPLE.TEXT.
DO:
```

ここでは，パスナンバーとして 123を設定している．パスナンバーを入力する箇所では入力した文字は，実際には見えないようになっている．またパスナンバーがついているファイルは，以降参照したり削除したりするときにパスナンバーを尋ねてくるようになる．次に，このファイルのパスナンバーを変更してみる．

```
DO:PASSNUMBER
FILE:=EXAMPLE.TEXT
OLD PASSNUMBER FOR EXAMPLE.TEXT:=123
NEW PASSNUMBER FOR EXAMPLE.TEXT:=1234
DO:
```

今度は，ファイルにパスナンバーがついているので，古いパスナンバーを尋ねた後に，新しいパスナンバーを聞いている．最後にパスナンバーを削除する．

```
DO:PASSNUMBER
FILE:=EXAMPLE.TEXT
OLD PASSNUMBER FOR EXAMPLE.TEXT:=1234
NEW PASSNUMBER FOR EXAMPLE.TEXT:=0
DO:
```

パスナンバーを削除するには， 0を入力するか復改だけを入力すればよい．

また，MEMO領域と SIGMA領域には，マスターキーをセットできるようになっていて（４章PROFILEコマンド参照），パスナンバーを忘れたファイルに対しても，マスターキーを指定すれば参照や削除ができる．

### ３．４．９ ワイルドカードの使い方

SIGMA システムでは，ファイル名の入力にワイルドカードが使える．たとえば，次のような使い方ができる．

DO:COPY A*.TEXT B*.DATA

これは，ファイル名がA で始まって.TEXT で終わるようなファイルを，ファイル名がB で始まって.DATAで終わるようなファイルに複写することを表している．つまり，ファイル名に使われている *（固定文字，変更不可）は，任意の文字列の代わりをしている．上の例で，ABC.TEXTとAAA.TEXTというファイルがあったとき，それぞれBBC.DATA，BAA.DATAにコピーされる．

SIGMA における，ワイルドカードの使い方には，使えない場合を含めて三種類ある．以下にそれぞれについて説明する．

(1) 自由に使える場合

これは，コマンドの中でファイル名を尋ねるところが一箇所だけの場合である（(3) に注意）．DIRECTORY，SEARCHコマンドなどがそうである（SEARCHはファイルを何度も尋ねてくるが，プロンプトの種類としては一つである）．たとえば，次のように使う．

FILE:=AA*BB*CC*DD

これは，AAで始まりBB，CCという文字列をこの順序で含んでいて，DDで終わるようなファイル名を表している．AAZBBBCCDD，AABBCCDDなどは，これにあてはまる．また，先頭や最後にワイルドカードがあってもよい．

FILE:=*AA*BB*

これは，AA，BBという文字列をこの順序で含んでいるようなファイル名を表している．XYZAA.BBCC，AABBなどはこれにあてはまる．

（注意）二つ以上の連続したワイルドカードの指定はできない．また，スタック区域に対するワイルドカードの指定はできない．つまり，A**B，W.*などはエラーになる．

(2) 二つを組にして用いる場合

これは，COPY，REPLACE ，SORTのなどようにファイル名を入力するところが，コマンドの中に二箇所ある場合である．たとえばCOPYコマンドで，

FROM-FILE:=*A*B*

TO-FILE:=*X*Y*

と指定すれば，ZAZBZ ，AABBというファイルは，ZXZYZ ，XAYBにコピーされる．ここではまず，*A*B *の中の *が持つ値を定める．つまり，ZAZBZ に対してはすべてZ ，AABBに対しては前から，空列（長さ0 の文字列），A，Bという値をとっている．これを *X*Y*の中の対応する *に当てはめて，新しいファイル名をつくる． *の値が空列でもよいところに注意が必要である．また二つ目の入力のところでは，(1)と違って *が連続して現れてもよい．つまり， *A*B*に対して *** という指定も可能である．このときZAZBZ に対しては，ZZZ となる．

このように，入力する二つのファイル名には，同じ個数のワイルドカードがなければならないが，例外的に二つ目の入力ではW.L などのスタック構造をしている区域名を指定することもできる．つまり，

COPY *A*B* W　（LOAD *A*B*でもよい）

とすれば， *A*B*に一致するファイル名が ZAZBZとAABBであるとすると，ファイル名としてはAABBの方が先に出てくるので，このCOPYコマンドは，

COPY AABB W

COPY ZAZBZ W

とコマンドを二回入力するのと同じである．したがってこの結果AABBのコピーはW.2 に，ZAZBZ

のコピーはW.1 に作られる.

（注意）一つ目の入力については，(1)と同じ注意が必要である．二つ目の入力については，まず上で述べたように一つ目の入力と *の数が同じでなければならない．また，

COPY *A* **

に対して，ABというファイルはB というファイルにコピーされるが，B という名前のMEMO区域のファイルは存在しないので（実はバックアップ区域である），このときは，¥¥¥.FILE.NAME.ERROR.B というファイルにコピーされる．また，長さが0 のファイル名ができたときは，¥¥¥.FILE.NAME.ERROR というファイルができる．

(3) まったく使えない場合

REFILEコマンドのように出力用のファイルしかない場合と，入力をファイルに切り替えるときなどには，ワイルドカードは使えない．また，外部データセットに対しても使えない．

以下に，ワイルドカードを用いる際の全般的な注意について述べる．ファイル名にワイルドカードを使った場合，常にパスナンバーを尋ねてくるようになっている．ここで，パスナンバーを入力すれば，ワイルドカードに一致するファイル名の中で，同じパスナンバーを持つものとパスナンバーがついていないファイル名を取り出してくる．例えば，MEMO区域にAA，AB，ACという３個のファイルがあり，それぞれパスナンバーが0，1，2 とするとき，

COPY A* B*

を実行すると，“PASSNUMBER:=” というプロンプトが出る．これに対して復改のみか，0 を入力すれば，AAがBAにCOPYされるだけであるが，1 を入力すれば，AAがBAに，ABがBBにCOPYされる．また，領域が持っているマスターキーを入力すれば，すべてのファイルがCOPYされる．DIRECTORY,DDIRECTORYコマンドでファイル名を復改だけしか入力しないときや，SIGMA 区域の区域名 “S” だけしか入力しないときにも，パスナンバーを尋ねてくる．これは，ファイル名として “*” や “S.*” を指定したものとみなされるからである．つまり，パスナンバーやマスターキーを知らないと，パスナンバーがついていないファイルしか見えない．

## ３．４．１０ COPYとMOVEの違い

SIGMA にはCOPYとMOVEというよく似たコマンドがある．ここでは両者の違いを説明する．

LOAD,SAVE,KEYIN,LIST,LOOK はCOPYのオペランドのうちの一方以上を固定したコマンドであるから，COPYとあわせてCOPY系コマンドと呼ぶ（４章参照）．同様にGET,PUT,DELETEもMOVEのオペランドのうちの一方を固定したコマンドであるから，MOVEとあわせてMOVE系コマンドと呼ぶ（４章参照）．ここでの説明は，COPY系とMOVE系のすべてのコマンドについて有効である．

COPYとは，ファイルの複製を作るためのコマンドである．複製元のファイルは消えない（バックアップ区域の底の方にあるファイルの場合は消えることがある）．例えば，

COPY DATA COPY-OF-DATA

と入力すると，DATAというファイルの複製がCOPY-OF-DATAという名前で作られる．コマンド実行後はDATAとCOPY-OF-DATAという全く同じ内容のファイルが二つ存在する．

COPY DATA W

と入力すると，MEMOファイルDATAの複製が作業区域のトップに作られる．コマンド実行後にもファイルDATAは存在している．また，COPYの場合，複製ファイルはどの領域のどの区域にあってもかまわない．領域間のファイルの転送機能をあわせ持っている．

それに対して，MOVEはファイルを移動させるためのコマンドである．移動元のファイル名はなくなってしまう．特に，移動元のファイルのあった区域がスタック構造の場合は，その区域の順位も変わってしまう（3．4．3参照）．MOVEはMEMOファイルやSIGMA ファイルの名前の付け替えという機能もあわせて持っている．例えば，

MOVE DATA W

と入力すると，MEMOファイルDATAが作業区域に移動する．コマンド実行後はDATAという名前のファイルは存在しない．

MOVE DATA DATA2

と入力すると，DATAというファイルの名前がDATA2に変更される．コマンド実行後はDATAという名前のファイルは存在しない．

MOVEは，MEMO領域と SIGMA領域にあるファイルにしか適用できず，移動先のファイルは移動元と同じ領域でなければならない．異なる領域間のファイルのやりとりや外部領域では必ずCOPYを用いなければならない．例えば，

COPY DATA S.'USER1.DATABASE'

と入力すると，MEMOファイルDATAの複製が USER1という利用者の SIGMA領域につくられる．しかし，

MOVE DATA S.'USER1.DATABASE'

許されていない．

### 3．4．11 フロッピーディスクと SIGMA間でのファイルの送受信

MS-DOSフォーマットのフロッピーディスクとSIGMA 間でのデータのやりとりには，パソコンと回線を使う方法と，九大センター二階のFMR-60を使う方法の二種類がある．

(1) 回線を使う方法

直接やりとりができて手軽ではあるが，大量のデータに対しては不向きである．EDITコマンドで，データセットを作成してからSIGMA に取り込む方法は，一行の長さに制約があるので注意をする．したがってSIGMA のKEYIN コマンドを使って直接転送する方が，データの加工をしなくてすむので便利である．端末プログラムに関する資料としては，たとえば[24,25] などがある．

(2) FMR-60を使う方法

FMR-60を使えば，フロピーディスクとデータセット間での転送ができ，またデータセットとSIGMA 間ではSIGMA のCOPYコマンドを使って転送できる．この方法は，回線を使う方法より，速度が速い．SIGMA に取り込めるデータセットは，テキストファイルであれば何でもよく，その名前やレコード形式は自由である．詳しい方法については[17]を参照されたい．

## 3．5 検索コマンド（SEARCH）の機能

SEARCHコマンドの機能について詳しく説明する．

### 3．5．1 特長

SEARCHコマンドの第一の特長は，区切り語と，キーワードの集合からパタン照合機械（pmm） と呼ばれる一種の有限オートマトンを作り，それが対象となるファイル，すなわちテキストファイルの上を先頭から末尾までただ一度走査する間にすべての作業を行う点にある．この他の主な特長は次の諸点にある．

(1) 自由なレコード形式に対して適用できる．テキストファイルは単なる文字列であるが，検索の単位となるレコードはレコード区切り語と呼ばれる文字列にはさまれた部分列として，仮想的に設定する．このため，テキストファイルの形式や検索の目的に応じて，自由にレコードを設定することができる．たとえば，段落や文章などは特定の文字列（復改記号や句点）で区切られているので，検索の単位を段落とするか文章とするかに応じて，それらをレコード区切り語に指定すればよい．また，検索時の意味の単位（論理式を評価する単位）も，項目区切り語と呼ばれる文字列を指定することによって設定できる．これを使えば，レコードを項目に区切ってきめ細かい検索条件を設定することができる．

(2) X1...X2...X3の形のキーワードの指定ができる．トリプル・ドット“...”は任意の文字列（空でもよい）を表わす．これによって，キーワードの出現順序に意味をもたせた検索が可能になる．

(3) 複数質問を同時に処理できる．SEARCHコマンドは一度ではあるが，ファイル全体を走査するので，処理に要する時間は必然的に対象ファイルの大きさに比例したものとなる．しかし，最大99個のキーワードを組み合わせて，論理式を99個，結果を得るための質問式（＝論理式）を32個同時に処理できるので，複数質問を同時に処理すれば，質問1個あたりの処理時間は比較的に短いことになる．もちろん，SEARCHコマンドで用いるパタン照合機械は非常に高速であるため，数メガバイト程度のテキストならば実用的時間内に処理できる．

(4) 検索能力が高い．キーワードの出現を単にマークするのではなく，出現回数をカウントするようにし，さらに論理式中ではキーワードの出現回数を用いた演算を可能とした．これにより，通常の論理演算では表わすことができないような質問も処理できる．

### 3．5．2 区切り語と仮想レコード

区切り語にはレコード区切り語(RECORD DELIMITER)と項目区切り語(ITEM DELIMITER)がある．いずれも空でない任意の文字列で，合わせて99個まで登録することができる．レコード区切り語は仮想的なレコードを切り出すためのものである．項目区切り語は，論理式の評価のタイミングを指示するためのものである．レコード区切り語は項目区切り語としても機能する．すなわち，論理式の評価は，レコード区切り語または項目区切り語が検出された時点で行う．ファイルの先頭と末尾には常にレコード区切り語があるものとみなす．したがって，レコード区切り語を全く登録しなければ，ファイル全体が一つのレコードとみなされ，検索結果は1件か0件になる．

レコード区切り語は，“RECORD DELIMITERS”に続くプロンプト“D01:=”，“D02:=”，… に従って登録する．登録を終わったら，次のプロンプトに対して復改のみを入力する．すると，“ITEM DELIMITERS”に続いてプロンプトが表示されるので，項目区切り語を同様に登録する．区切り語の登録が終わると，次のプロンプトに対して復改のみを入力する．すると，キーワードの登録に移る．

### 3．5．3 キーワード

キーワードには，通常の空でない文字列としてのキーワードの他に，トリプル・ドット“...”を含むX1...X2...X3の形のキーワードがある．キーワードの登録は，メッセージ“KEYWORDS”に続くプロンプト“A01:=”，“A02:=”，… にしたがって行う．個数は99個までである．A01，A02，… はキーワード変数と呼ばれ，次項で説明する論理式を構成するために用いられる．各キーワード変数がとる値は対応するキーワードの出現回数である．初期値は0であり，区切り語が検出されて論理式の評価を終えると0にリセットされる．ただし，トリプル・ドットを含むキーワードの出現回数は互いに重

なり合わないものをカウントする．たとえば，ABABCにおけるA...B...Cの出現回数は2ではなく1である．

### 3．5．4　論理式

論理式は，キーワード変数（A01，A02，...），式変数（Z01，Z02，...），整定数を，論理演算子（“,”(or)，“.”(and)，“^”(not)），比較演算子（“〈”，“〉”，“=”，“〈=”，“〉=”，“〈〉”），算術演算子（“+”，“-”，“*”，“/”），カッコ（“(”，“)”）で組み合わせて作られるものである．論理式の登録は“LOGICAL FORMULAS”に続くプロンプト“Z01:=”，“Z02:=”，… に従って行う．個数は最大99個である．Z01，Z02，… を式変数と呼ぶ．なお，論理式を入力するときには，キーワード変数 A01や式変数 Z02などは，A1，Z2と略記してよい．

論理演算子は次のように解釈される．

$\alpha, \beta$ ＝1（$\alpha > 0$ または $\beta > 0$ のとき），0（そうでないとき）

$\alpha . \beta$ ＝1（$\alpha > 0$ かつ $\beta > 0$ のとき），　0（そうでないとき）

$\hat{}\alpha$ ＝1（$\alpha \leq 0$ のとき），　0（そうでないとき）

ただし，真偽値をそれぞれ1，0で表している．

各演算子の優先順位とその意味（自明なものは省いた）は次で与えられる．優先順位は小さいものが先である．

| 演算子 | * | / | + | - | = | 〈= | 〉= | 〈 | 〉 | 〈〉 | ^ | . | , |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 優先順位 | 1 | 1 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 4 | 5 | 6 |
| 意味 | | | | | | ≦ | ≧ | | | ≠ | not | and | or |

また，論理式の登録において，次のような省略形を用いることができる．

Zi:=,　　（すべてのキーワード変数の論理和）

Zi:=.　　（すべてのキーワード変数の論理積）

Zi:=^　　（すべてのキーワード変数の論理和の否定）

Zi:=+　　（すべてのキーワード変数の和）

質問は論理式で与える．また論理式の最後にスラント“/”を置くとその論理式は中間的な式変数としてのみ扱われる．スラントなしの論理式が質問を表わし，最大32個まで登録できる．

### 3．5．5　検索件数の表示とファイルの指定

論理式の登録を終了すると，検索件数（各質問に対してヒットしたレコード数）を表示するかどうかを，プロンプト“REPORT(Y/N)?”に従って指定する．省略時（復改のみを入力した場合）には検索件数を表示する．

最後に，検索すべきファイルを指定するために，プロンプト“FILE:=”に従ってファイル名を入力する．指定できるファイルはMEMO領域，SIGMA 領域にあるものでなければならない（外部データセットや“K”，“T”，“D”（キーボード）などは指定できない）．

検索件数の表示を行う場合には，各質問にヒットしたレコード数とその合計（重複を除く），処理に要した CPU時間（ミリ秒単位）を表示する．SEARCHコマンドを1回投入すると，同じ質問を用いて複数のファイルを検索できるので，検索件数は，各ファイルに対するものと，それまで検索してきたファイルに対する結果の累計とをあわせて表示する．

ファイルの検索を終了すると，再びプロンプト“FILE:=”が表示されるので，さらに続けて検索す

るファイルがあれば，それを指定する．検索すべきファイルがない場合には，復改のみを入力すると，SEARCHコマンドの処理を終了し，プロンプト“DO:” が表示される．

### 3．5．6 検索の原理と論理式の評価

SEARCHによる各ファイルに対する検索は，おおむね次の手順で行われる．ここで，キーワード変数は項目およびレコードごとに初期化されるのに対し，論理式に対応する各式変数はレコードごとにしか初期化されない点に注意してほしい．

(1) 検索すべきファイル（入力ファイル）を開く．

(2) 式変数をすべて0にし，レコードの開始位置を記録する．

(3) キーワード変数をすべて0にする．

(4) パタン照合機械(pmm) の状態を初期状態にする．

(5) 入力ファイルから1文字入力し，pmm を動作させる．これをパタンが検出されるか，入力ファイルの終わりに到達するまで繰り返す．

(6) パタンが検出されると，その種類によって次の処理をする．複数のパタンが検出される場合は(6.1)，(6.2)，(6.3) の優先度で処理する．

(6.1) キーワードの場合は，対応するキーワード変数を1増やす．他にパタンが検出されてなければ(5) へ戻る．他にパタンが検出されているときは，残りのパタンの処理のために (6)をくり返す．

(6.2) レコード区切り語の場合は，式変数を評価する．その結果，真となった質問が1個でもあれば，レコードの索引情報（開始位置，長さなど）を索引ファイルに出力し，(2) へ戻る．同時に項目区切り語がみつかっている場合は，その項目区切り語は無視される．

(6.3) 項目区切り語の場合は，式変数を評価し，(3) へ戻る．

(7) 入力ファイルの終わりに到達した場合には，レコード区切り語を検出した場合と同様に，式変数を評価し，必要ならばレコードの索引情報を索引ファイルに出力する．

(8) 入力ファイルを閉じる．

項目区切り語やレコード区切り語を検出した時点で行う式変数の評価は，Z01，Z02，... の順に行われる．質問を表す式変数は，いったん真になるとそれ以後は真とするが，論理式の最後にスラントをつけた中間的な式変数は各項目ごとに単純に評価を行う．ただし，いずれの式変数も切り出された項目やレコードが空の場合は評価を行わない．したがって，空レコードは検出されることがない．式変数の評価の順序から，Ziの定義にZjが含まれるとき，Ziの評価で用いられるZjの値は，i≦jならば評価前のものであり，i＞jならば評価後のものであることに注意しよう．

項目区切り語を設定した場合には，キーワード変数や論理式（式変数）のもつ意味に注意を要する．キーワード変数は，項目中でのキーワード出現回数である．トリプル・ドットを含むキーワードは項目をまたがって出現していてもカウントされない．たとえば，レコード区切り語，項目区切り語をそれぞれ“#”，“%” とし，次のようなテキストを考える．

```
#A%B#ACB#
```

初めのレコードA%B では，“A” と “B” は項目をまたがって現れているので，キーワード“A...B” はヒットしないが，2番目のレコードACB ではヒットする．論理積についても同様の注意が必要である．式変数が

```
Z01:=A1
Z02:=A2
```

Z03:=A1.A2
Z04:=Z1.Z2

で定義されている場合にZ3とZ4の意味の違いを考えてみよう．質問式はある項目でいったん真になると，それ以後，そのレコード内では真であるので，Z1の意味は「A1が現れる項目が少なくとも1個ある」である．Z3は「A1とA2の両方を含む項目が少なくとも1個ある」という意味になるが，Z4は「A1を含む項目もA2を含む項目もどちらも少なくとも1個ある」という意味になる．このことから，項目をまたがって論理積をとりたければ，Z4のようにいったん式変数に代入しておけばよいことがわかる．

### 3．5．7　否定の取り扱い

SEARCHコマンドでは否定を取り扱うことができるが，通常の検索意図に否定が含まれているからといって，そのまま否定演算子を用いるのは危険である．たとえば，「“われわれ”でない“われ”を含む文章を求める」ためには，適当な区切り語を登録し，

```
     KEYWORDS
A01:=われわれ
A02:=われ
A03:=↙
     LOGICAL FORMULAS
Z01:=^A1.A2
.....
```

として検索を行えばよいように思える．しかも，検索の結果には“われ”は含まれているが，確かに“われわれ”は含まれていないので，一見して正しいようにみえる．しかし，“上の例には，われわれが通常用いている否定をそのまま論理演算子として用いると誤った検索をしてしまう可能性があることが顕著に現われている”のように，“われわれ”と“われ”の両方を含んでいる文章は検索されない．なぜならば，“われわれ”が含まれる場合には^A1は偽となり，それ以外に“われ”が含まれているかどうかに係わりなくZ1は偽となるからである．

上の検索意図を忠実に表わすためには，キーワード変数がカウンターであることを利用して，

```
Z01:=2*A1〈A2
```

とすればよい．ここで，A1を2倍にしているのは，“われわれ”には“われ”が2回現れているからである．

### 3．5．8　検索結果の表示と再ファイル化（REFILEコマンド）

SEARCHコマンドによる検索結果は索引区域のトップの索引ファイルにコード化した形で保存される．これをテキストの形に復元して，端末に表示したり，ファイルに出力したりするためのコマンドがREFILEである．REFILEコマンドは，次に示す例のように，プロンプト“DO:”または“SIGMA)”の状態で使用する．ただし，検索したファイルがREFILEを行う時点で存在していないと，再ファイル化できないので注意が必要である．

```
(1) DO:REFILE
(2) REPORT(Y/N)?↙
      RETRIEVED TEXTS
```

(1) REFILEコマンドを投入する．
(2) 検索件数を表示するかどうかを指定する．この例のように指定を省略すれば表示を行う．表示が不要

```
    QUESTION 01 (Z01) =          8
    QUESTION 02 (Z02) =         10
    TOTAL             =         12
(3) QUESTION:=2
(4) NEW RECORD DELIMITER:=!
(5) NUMBERING(N/Y)?↙
(6) OUTPUT-FILE:=W
    QUESTION 02 (Z02) =         10
(7) QUESTION:=↙
    DO:LOOK
    ..................
     〈〈 検索結果の表示 〉〉
    ..................
```

の場合には，“N” を指定する．

(3) どの質問に対する検索結果を選択するかを質問番号で指定する．

(4) 表示や再ファイル化を行うときにレコードを区切る文字列を指定する．ここでは，復改で区切ることになる．

(5) レコード番号をつけるかどうかを指定する．省略時は番号づけをしない．

(6) 出力ファイルを指定する．ここでは，作業用ファイル“W” を指定している．

(7) さらに別の質問に対する検索結果を処理する場合には，(3) と同様に指定する．復改のみを入力すると，REFILEコマンドを終了する．

上の例では，再ファイル化した結果を作業用ファイルに出力している．この結果は，通常のテキストファイルであるので，SEARCHコマンドを用いて検索を行うなど，自由に取り扱うことができる．ここでは，LOOKコマンドを用いて検索結果を端末に表示している．検索結果を端末に表示する方法としては，この例のようにファイルに出力した後にLISTやLOOKコマンドを用いるほかに，REFILEの出力ファイルに直接端末装置“T”，“D” を指定する方法もある．なお，出力ファイルに“K” を指定すると，検索結果は１レコードずつ中断しながら表示される．このときには，１レコード分の表示が終わるたびに，プロンプト“MORE(Y/N)?” が出力されるので，復改または“Y” を入力すると次のレコードの表示を行い，“N” を入力すると表示を中止する．

REFILEコマンドは常に，索引区域のトップにある索引ファイルをもとに処理を行う．したがって，通常は，直前に実行したSEARCHコマンドの検索結果を処理する．ただし，索引区域は作業区域と同様に底のあるスタック構造になっているので，MOVEコマンドを用いて索引ファイルを移動すれば，直前の検索結果以外でも処理することができる．たとえば，直前の一つ前の検索結果を再ファイル化する場合には，

```
DO:MOVE I.2 I
DO:REFILE
```

とすればよい．

### 3．6 ソートコマンド(SORT)の機能

ここでは，今回の改訂で強化された[25]SORTコマンドの機能の詳細や，キーの切り出し仕様の記述について述べる．

#### 3．6．1 特長

SORTコマンドは，SEARCHコマンドと同様に，レコード区切り語や切り出しパタン（切り出すべきキーを指定するもの）からパタン照合機械を作り，それが対象とするテキストの上を先頭から末尾までただ一度だけ走る間に必要なキーを切り出して，ソートするものである．

通常の，例えば表形式のデータに対するソートでは，単に第１キー，第２キー，…となる欄を指定

すればソートできるが，このSORTコマンドは，テキストファイルを対象としているため，さらに以下に述べるいろいろな特長や機能を備えている．

(1) SEARCHコマンドの項で述べたように，自由なレコード形式に対して適用できる．

(2) 通常のソートと同様に，優先度がついた複数のキーを指定できる．

(3) 1つのキーの切り出し指定（切り出し仕様と呼ぶ）では，仮想レコード中のどの範囲からキーを切り出すかを指定でき（範囲記述部により指定），またどのようなキーを切り出すかを，文字種のパタンで指定できる（読み込み記述部により指定）．キーの切り出しは，指定された範囲に入った直後ではなく，その後で指定された切り出しパタンの最初の文字種が見つかった時点で開始し，以下切り出しパタン全部，または切り出しパタンと異なる文字が見つかるまで行なわれる．これによって，例えば切り出しパタンとして漢字5文字を指定して氏名を切り出そうとした場合，最長5文字までの氏名を切り出すことになり，3文字や4文字など5文字に満たない氏名も切り出すことができる．また，切り出しパタンとして，複数の選択的なパタンも指定でき，この場合は最初に先頭の文字種が見つかったパタンで切り出される（3．6．3 読み込み記述部の例を参照）．

(4) 切り出し仕様では，各種のオプションも指定できる．これには，昇順／降順，キーの左寄せ／右寄せ，左／右からのソート，キーが欠落した場合の扱いなどがある．なお，通常のソートでは，各キーの切り出し仕様に基づいた切り出しが1つの仮想レコードで1回だけ行われる．単語単位のソートを行おうとする場合には，英文などのように単語が空白で区切られているときは，空白を区切り語と指定して1単語を1レコードとすれば，このままでよい．しかし，日本語文のように単語が区切られていない場合には難しい．そこで，切り出し範囲の中で指定された切り出しパタンの文字列を，可能な限り何回でも切り出して，その1つ1つをキーとするモードも付加した．

(5) ソートされた結果をどのように再ファイル化するかに関しても，豊富な組合せを用意した．2．4 2．5 の使用例も同時に参照されたい．

### 3．6．2 パタンとパタンリスト

切り出すべきキーや，切り出しの範囲を指定するためには，パタンとその並びであるパタンリストを用いる．

まず，パタンは次のものからなる．

(1) 固定文字列（シングルクォートで囲まれたもの，ただし，'は''で表す）はパタンである．

（例） 'apple'　'果物'　'John''s melon'

それぞれ，apple　果物　John's melon を表す．

(2) A,N,S,K,AL,GR,RU,NU,SY,HK,KK,KJ　（ピクチャといい文字の種別を表す）

| | | | |
|---|---|---|---|
| A：英字(Alphabet) | (A,B,…,y,z) | N：数字(Numeral) | (0,1,…8,9) |
| S：記号(Symbol) | (!,",#,…) | K：仮名(Kana) | (ｱ,ｲ,…,ﾝ,ﾞ,ﾟ,-) |
| AL：英字(ALphabet) | (A,B,C,…) | GR：ギリシャ文字(GReek) | (A,B,…,φ,ω) |
| RU：ロシア文字(RUssian) | (А,Б,…,ю,я) | NU：数字(NUmeral) | (０,１,…,８,９) |
| SY：記号(SYmbol) | (！,”,＃,…) | HK：平仮名(HiraKana) | (あ,い,…を,ん) |
| KK：片仮名(KataKana) | (ア,イ,…ヲ,ン,ー) | KJ：漢字 (KanJi) | (亜,唖,…龜,龕) |

(3) ピクチャの前に1から99までの数を書いたもの　（そのピクチャを続けた文字列を表す）

（例） 5A　半角の英字からなる5文字を表現する．“apple”，“melon”，“LEMON”など．

2KJ　漢字からなる2文字を表現する．“果物”，“肉類”など．

(4) パタンを *で連結したもの（連続したパタンを意味する）.

（例） 2KJ*'＝'*4KK　漢字２文字の後に“＝”と平仮名４文字が続く文字列を表現する.
“果物＝くだもの”，“肉類＝にくるい”など.

パタンリストはパタンをコンマで区切って並べたもので，それらのパタンのいずれかを示す.

（例） 5A*3K,5AL*3KK　半角または全角で，英字５文字の後に仮名３文字が続いたものを表現する.
“melonﾒﾛﾝ”，“ｍｅｌｏｎメロン”など.

### 3．6．3 切り出し仕様

切り出し仕様は，“K01:=”，“K02:=”，… のプロンプトにより指定し，その各々で一つのキーの切り出し仕様を定める．この仕様で切り出されたキーを，K01:=，K02:=，…の優先順位でソートの対象とする．切り出し仕様は，範囲記述部，読み込み記述部，仕様記述部からなり，それらをコンマで区切って並べたものである．範囲記述部と仕様記述部は省略できるが，読み込み記述部は省略できない.

[ <範囲記述部> ,] <読み込み記述部> [, <仕様記述部> ]

(1) 範囲記述部

範囲記述部はFIND指示子からなる． FIND指示子は次の形をしたものである.

F/ <パタンリスト１> / [U/ <パタンリスト２> /]

FU/ <パタンリスト２> /　　(F:Find, U:Until)

この記述部により，パタンリスト１とパタンリスト２ではさまれた部分が，キーの切り出し対象範囲となる．ただし，パタンリスト１が省略されるとレコードの先頭から，パタンリスト２が省略されるとレコードの末尾までを指定した事になる．また，パタンリストが複数のパタンで構成されている場合，最初に見つかったパタンで切り出し範囲が規定される.

（例）１レコードが次の形をしているとする.

(TI) Efficient String Matching: An Aid to Bibliographic Search
(AU) Aho,A,V., Corasick,M,J.
(SO) Comm.ACM,Vol.18,pp.330-340,(1975)

(a) F/'(AU) '/　によりキーの切り出しの対象範囲は，“Aho”からレコードの最後までとなる

(b) F/'(AU) '/U/'(SO)' により切り出しの対象範囲は，“Aho,A,V., Corasick,M,J.”となる．これにより切り出しの範囲として，文献の著者を指定することになる.

(c) U/'(SO)'/ により切り出しの対象範囲は，次の２行となる.

(TI) Efficient String Matching: An Aid to Bibliographic Search
(AU) Aho,A,V., Corasick,M,J.

また，(TI)や(AU)が（表題）や（著者）となっているレコードが含まれているときは，

(d) F/'(AU) ',' （著者） '/U/'(',' ('/ により著者を指定できる.

(2) 読み込み記述部

GET指示子とSKIP指示子からなり，一つの切り出し仕様には GET指示子が少なくとも一つはなければならない．読み込み記述部は次のものからなる.

(a) r( {GET 指示子 / SKIP指示子} ,…, {GET 指示子 / SKIP指示子} ) (１≦r≦99)

繰り返しの rは省略してもよい．省略すると 1を指定したものとみなされる．この場合は（）は省略してよい．

(b) (a)で構成された読み込み記述部をコンマで区切って並べたもの

GET 指示子は，

G/〈パタンリスト1〉/ [U/〈パタンリスト2〉/]　　(G:Get, U:Until)

の形をしている．なお，〈パタンリスト1〉には固定文字列を含んではならない．

SKIP指示子は，

S/〈パタンリスト〉/　　(S:Skip)

の形をしている．

まず，範囲記述部で指定された文字列の先頭（範囲記述部がない場合はレコードの先頭）をポインタが指している．GET 指示子は現在のポインタ位置から，パタンリスト1の第1ピクチャを捜し，その位置から指示されたパタンを指定された長さ，またはパタンリスト2が見つかるまで切り出す．このとき，ポインタはパタンリスト2の次の文字を指す．この GET指示子のパタンリスト1の文字列の検出は，他の指示子のパタンリストにおける完全一致による検出と異なり，最長前方一致による．すなわち，一般にはパタンリスト1にコンマで区切られたいくつかの選択的なパタンがあるが，そのうち先頭のピクチャに対応する文字が最初に検出されたパタンを選び，そのパタンに従って最も長く前方が一致している所までをキーとして切り出す．切り出されたキーの長さがパタン長に満たないときは，残りすべてを半角の空白で補う．GET 指示子が複数個あるときには，切り出された文字列はすべて連接される．Until の指示があり，パタンリスト1の後部とパタンリスト2の前部が重なった時は，パタンリスト2と重複する部分は切り出されない．

また，SKIP指示子はパタンリストのパタン部分までの文字列（完全一致を要する）を読み飛ばし，その文字列の直後の文字へポインタを移動する．

（例）　テキストを “漢字B12” とする．G/2A,2N/なる GET指示子を適用すると，完全一致であれば2Nに対する “12” のみがキーとして切り出されるが，パタンの先頭のピクチャは Nと Aであり最初に文字 “B” がピクチャA を満足するので，以後2Aに従って GET指示子が働く．次の文字は，数字 “1” だからピクチャ Aの条件を満足せず，1バイトの空白を補い “B ” というキーが切りだされる．

テキストを “Comm.ACM,Vol.18,pp.330-340,(1975)” とする．

(a) G/3A/ により “Com”，G/5A/ により “Comm ” が切り出される．

(b) G/3A/,G/3A/ により “Comm ” が切り出される．

(c) S/'.'/,G/5A/ により “ACM ” が切り出される

(d) G/3A/U/'m'/ により “Co ”，G/3A/U/'o'/ により “C ” が切り出される．

(e) 2(G/3A/,G/S/) により “Com.ACM,” が切り出される．

(f) 2(G/5A/U/S/),S/'pp'/,2(G/5N/) により “Comm ACM 334 340 ” が切り出される．

(3) 仕様記述部

1つの切り出し仕様に対するソーティングの仕様を指定するものである，仕様記述部はOPTION指示子からなる．OPTION指示子は O(〈パラメータ〉) の形である．

パラメータには以下のものがあり（下線は指定がない場合のデフォルト値），指定の順序は任意である．

LJ／RJ　キーが指定された長さに満たない場合，左端揃え(Left Justify)にするか，または，右端揃え(Right Justify) にするかを指示するが，いずれの場合も不足分は半角のブラン

クで補う．１つのキーが複数の GET指示子により切り出される場合は，その各々について端揃えをしてから連接する．なお，桁数が不揃いの数字列に対してRJを指定すると，数値としてのソートができる．

OR／RE　キーの文字単位のソート順序を，通常(ORdinary)の左からか，逆順(REverse) の右からかを指示する．REは，RJと組み合わせると逆引き辞書の作成に便利である．

AS／DE　キーごとのソート順序を，昇順(AScending order) か，降順(DEscending order)により指示する．

IG/TO/BO　切り出されるべき文字列がなかった場合，そのレコードを全体の先頭(TOp) に置くか，最後(BOttom)に置くか，それとも無視(IGnore)するかを指定する．これはキーごとに指定できるため，第１キーに関しては先頭に，第２キーに関しては最後に，などといった指定が可能である．無視(IG)の場合は，すべてのキーについて切り出される文字列がなかったときのみ，無視される．欠落情報のチェックを行う場合には，先頭(TO)を指示するとよい．

CA／SE　読み込み記述部にしたがった切り出しは，通常１レコードに対して１回だけ実行され，いくらかの GET指示子でキーを切り出されると，それらは全て連接されて(CAtenate)一つのキーとなる．しかし，これでは３．６．１で述べたような，特に日本語文に対する単語単位のソートができない．そこで，GET指示子 で指定された文字列を可能な限り何回も切り出して，その各々を別々の(SEparate)キーとしてソータに引き渡す指定ができる．このとき，切り出された文字列が重なることはない．また，“SE”の指定ができるのは，最初の“K01:=”のときのみで，この場合には“K02:=”以降の指定はできない．“K01:=” 以外のところで指定すると，構文エラーとしてチェックされる．この指定は，日本語文の単語単位のソートに便利である．

（例）“Comm.ACM,Vol.18,pp.330-340,(1975)” というテキストに対し，G/4A/,0(SE)によって“Comm”，“ACM ”，“Vol ”，“pp”が切り出される．

以上，指定がない場合をまとめると，通常のキーに関しては，左端揃えで不足分は空白で補い，左からの昇順のソートとして，切り出されるべき文字列がないレコードは無視する．すなわち，

0(LJ,OR,AS,IG,CA)

を指定したのと同じである．

（例）

(a)　F/'(AR) '/,2(G/4A/,S/'.,'/)

２の使用例５のファイルを対象にして，文字列“(AR) ”の後から，半角英字４文字を切り出して文字列“.,”まで読み飛ばす操作を，２回繰り返す．左端揃えで左からの昇順ソートとする．第２著者までの先頭４文字ずつを切り出し，連接した８文字をキーとすることになる．

(b)　F/'(MS) '/,S/'(19'/,G/2N/,0(TO)

同じファイルを対象にして，文字列“(MS) ”の後で文字列“(19”まで読み飛ばし，半角数字２文字を切り出す．左端揃えで左からの昇順ソートとして，見つからなかった（発行年の欠落した）レコードは先頭に配置する．

(c)　F/'（本文）'/U/'（文献）'/,G/10KK/,0(SE)

原稿などの文書ファイルを対象にして，文字列“（本文）”と“（文献）”の間で，片仮名10文字を可能な限り切り出し，その各々をキーとする．

### 3．6．4　再ファイル化仕様

複数の切り出し仕様を指定した後で，これらのキーでソートした結果を再ファイル化する単位の指示をする．これはプロンプト“REFILE UNIT:=” に続いて次のいずれかを指定する．

[R | Ki | Ki# | KiL | #Ki | 〈切り出し仕様〉]

| | |
|---|---|
| R | 最初に指定した仮想レコード単位で，全体を再ファイル化する． |
| Ki | Kiで切り出されたキーそのものを，1レコードとして再ファイル化する． |
| Ki# | Kiで切り出されたキーそのものに，頻度を付けて再ファイル化する． |
| KiL | Kiで切り出されたキーそのものに，それが現れたレコード番号のリストを付けて再ファイル化する． |
| #Ki | Kiで切り出されたキーを，頻度で再ソートして，頻度＋キーの形で再ファイル化する． |
| 〈切り出し仕様〉 | この仕様で切り出される付属情報のみを1レコードとして再ファイル化する．すなわち，キーワードの切り出しと同じ形式でレコードの出力情報の指定ができる．この場合，仕様記述部ではLJ/RJ の指定しか意味をもたない． |

再ファイル化仕様を省略した場合（復改キーのみを入力した場合）は，R を指定したものとみなされる．なお，R 以外の指定をした場合，各レコードの区切りは復改記号となる．

### 3．7　置き換えコマンド（REPLACE）の機能

ファイルの中の指定された文字列をことごとく指定された他の文字列で置き換えることができる．これは通常のエディタの置き換えコマンドを一般化したもので，次のように入力する．

```
DO:REPLACE
X01:=x1
Y01:=y1
X02:=x2
Y02:=y2
.....
Xn:=xn
Yn:=yn
Xn+1:=↙
INPUT-FILE:=filename1
OUTPUT-FILE:=filename2
```

これによって，filename1という名前のファイルの中に含まれている文字列xiをことごとく文字列yiで置き換えたファイルが filename2に作られる．なお，使用にあたって以下のことに注意されたい．

(1) yiは空列（すなわち，Yi:=↙）でもよいが，xiには空列は許されない．

(2) ある文字列xiが検出され，これがyiで置き換えられた時点で，文字列x1, x2, ･･･, xnを探す作業は初期化され，このxiの次の文字から新しい置き換え作業が繰り返される．

(3) 文字列 xi, yiの登録において，xi＝xj（i＜j） となった場合には，始めの方の

```
Xi:= xi
Yi:= yi
```

は無視される．したがって，これを置き換えの指定の訂正に使うことができる．すなわち，第i番目の指定を訂正するには，

Xj:= xi

Yj:= yj

のようにXjに訂正したい文字列xiを，またYjに正しく置き換えるべき文字列yjを割り当てればよい．

(4) REPLACE コマンドは左から右への一方向の逐字処理により置き換えを行う．キーワードに指定した文字列が重複している場合は，左から見てなるべく長い文字列を置き換え，バックトラック（後戻り）は行わない [6]．図6で説明しよう．この例では，左のx2はより長いx1に含まれているので無視される．左のx4は，x3より右にあるので無視される．これは，バックトラックを行わないためである．置き換えられるのは，x1, x3, x5, 右のx2, 右のx4である．

置き換え前　　ああ(X1:あいいい)(X2:いう)(X3:うえお)(X4:おあい)(X2:いう)(X4:えお)

置き換えの指定

X01:=ああいい

Y01:=アアイイ

X02:=あい

Y02:=ＡＩ

X03:=うう

Y03:=ウウ

X04:=うえお

Y04:=ＵＥＯ

X05:=おお

Y05:=オオ

X06:=↙

置き換え後　　あアアイイいウウえオオＡＩＵＥＯ

図6．　文字列の同時置き換え

## 4. コマンドレファレンス

この章では，各コマンドについてそれぞれ説明を行う．各コマンドの説明は次のような表記法に従っている．

(1) コマンドの入力は，見出しのコマンドにつけた下線部だけ入力すれば十分である．つまり，下線部が他のコマンドと区別できる最短省略形になっている．たとえば，CATENATEの入力は，CA,CAT,CATE,...,CATENATEのいずれでもよい．

(2) 入力形式中の〈A〉の形の表記は，利用者が入力すべきオペランドであることを示している．

(3) オペランドが，［A｜B｜C］のように表記されている場合は，選択して入力，ただし省略も可能である．［A｜B｜C］のようにアンダー・ラインが引いてある場合には，省略した時にはその値が選択されることを示す．

(4) オペランドが，｛A｜B｜C｝のように表記されている場合は，選択して入力するが省略はできないことを示す．

(5) 順次入力を求めてくる番号付きのプロンプトにおいては，復改のみを入力すると次の段階のプロンプトに移る．復改のみを入力することを↙のように表記する．

(6) 入力形式中などに現れる半角の感嘆符 !は代用復改記号である．したがって，これは，PROFILEコマンドで変更することができる．

(7) ファイル名に対してワイルドカードを指定した場合は，説明してあるオペランドとは別に，パスナンバーを尋ねてくる．パスナンバーに関しては3.4.8，ワイルドカードに関しては3.4.9を参照のこと．

### 1 CATENATE

機能　：ファイルを連接する．

入力形式：1. DO:CATENATE
　　FROM-FILE01:=〈連接元ファイル名1〉
　　……
　　FROM-FILEn:=〈連接元ファイル名n〉
　　FROM-FILEn+1:=↙
　　TO-FILE:=〈連接先ファイル名〉
　2. DO:CAT 〈連接元ファイル名1〉 … 〈連接元ファイル名n〉 ! 〈連接先ファイル名〉

解説　：SIGMA システムにおけるファイルは文字列であるから，ファイルの連接は文字列の連接である．ファイルを連接する順番は連接元ファイルの入力の順番である．連接元ファイルは最大99個まで登録できる．

### 2 COMMENT

機能　：ファイルにコメントをつける．

入力形式：1. DO:COMMENT
　　FILE:=〈ファイル名〉
　　COMMENT:=〈コメント文〉
　2. DO:COMMENT 〈ファイル名〉

COMMENT:= <コメント文>

解説　：COMMENTコマンドによってつけたコメントは，DDIRECTORYコマンドで表示される．

3　COPY

機能　：ファイルを複写する．

入力形式：1. DO:COPY
FROM-FILE:= <複写元ファイル名>
TO-FILE:= <複写先ファイル名>
2. DO:COPY <複写元ファイル名> <複写先ファイル名>

解説　：ファイルの複製を作るので，異なる領域間での複写も可能である．

4　DDIRECTORY

機能　：ファイルの名前及び属性（ID番号・別名数・大きさ(バイト数)・作成期日・コメント）を表示する．

入力形式：1. DO:DDIRECTORY
FILE:= [ <ファイル名> | <区域名> ]
OPTION:= [A | S | C]
2. DO:DDIRECTORY [ <ファイル名> | <区域名> ] [A | S | C]
ただし，<区域名> は，W,L,I,B,S,S.Bのいずれかである．
[A | S | C] の各選択肢は，次の意味である．
A:すべての属性（ID番号・別名数・大きさ(バイト数)・作成期日・コメント）
S:大きさ（サイズ）
C:コメント
省略するとコメントを除くすべての属性を表示する．

解説　：プロンプト“FILE:=”に続いてファイル名を指定するとそのファイルの属性を表示する．区域名を指定するとその区域にあるファイルの名前及び属性を表示する．復改のみを入力するとMEMO区域を指定したことになる．区域としてMEMO区域か SIGMA区域を指定したときとファイル名にワイルドカードが含まれるときには，パスナンバーを尋ねてくる．そして，同じパスナンバーがついているファイルとパスナンバーがついていないファイルだけを表示する（3.4.9参照）．またID番号とは，ファイルの実体を一意的に表している番号であり，別名をつけたり移動したりしてもID番号は変わらない．

5　DELETE

機能　：ファイルを除去する（消去ではない）．

入力形式：1. DO:DELETE
FILE:= <ファイル名>
2. DO:DELETE <ファイル名>

解説　：ファイルを消去するのではなく，バックアップ区域のトップに移すコマンドである．つまり，“MOVE <ファイル名> B” と同じ働きをする．したがって，誤って除去した場合，“MOVE B <ファイル名>” で復元できる．

6　DIRECTORY

機能　：ディレクトリ（ファイル名一覧）を表示する．

入力形式：1. DO:DIRECTORY
　　FILE:=［〈ファイル名〉｜〈区域名〉］
　　2. DO:DIRECTORY ［〈ファイル名〉｜〈区域名〉］
　　ただし，〈区域名〉は，W,L,I,B,S,S.Bのいずれかである．

解説　：プロンプト“FILE:=”に続いて復改のみを入力するとMEMO区域を指定したことになる．区域としてMEMO区域か SIGMA区域を指定したときとファイル名にワイルドカードが含まれているときには，パスナンバーを尋ねてくる．そして，同じパスナンバーがついているファイルとパスナンバーがついていないファイル名だけを表示する（3.4.9参照）．区域としてW,L,I,B,S.Bを指定すると各区域のファイルの数のみが表示される．

7　ECHO

機能　：入力をファイルから行うときに，そのファイルの内容を表示するかどうかを設定，または端末にメッセージを出力する．

入力形式：1. DO:ECHO
　　MODE OR STRING:=［ON｜OFF｜〈メッセージ文〉］
　　2. DO:ECHO ［ON｜OFF｜〈メッセージ文〉］

解説　：このコマンドは，入力をファイルから行う場合や LOGファイル実行中に，そのファイルの内容を表示するかどうかを設定するために使う．また，メッセージを出力するのに使うと便利である．

8　END

機能　：SIGMA を終了する．DO状態を終える．

入力形式：1. DO:END
　　2. SIGMA)END

解説　：DO状態や SIGMAを終える． SIGMA状態でコマンドを入力してから ENDコマンドでDO状態を終わるまでが一つの LOGファイルとなる．

9　GET

機能　：指定されたファイルを作業区域のトップへ取ってくる．

入力形式：1. DO:GET
　　FILE:=〈ファイル名〉
　　2. DO:GET 〈ファイル名〉

解説　：“MOVE 〈ファイル名〉 W”と同じ働きをする．

10　HELP

機能　：SIGMA で使用できるコマンドや用語の説明を表示する．

入力形式：1. DO:HELP
　　HELP:〈トピック〉

HELP FOR 〈トピック〉:〈サブトピック〉

2. DO:HELP 〈トピック〉 [〈サブトピック〉]

解説 :トピックとしては，コマンドや用語を指定できる．HELPだけを入力すると，参照できるトピックが表示され，プロンプトが“HELP:” となる．ここでトピックを入力すると，それについての説明および指定可能なサブトピックが得られる．このとき，コマンドの入力と同様に他のトピックと区別のつく部分までを入力すれば十分である．ヘルプを終わるためには，復改だけを入力すればよい．

## 11 HEXDUMP

機能 :ファイルの16進ダンプリストを作りファイルに書き出す．

入力形式:1. DO:HEXDUMP

FROM-FILE:= 〈ファイル名1〉

TO-FILE:= 〈ファイル名2〉

2. DO:HEXDUMP 〈ファイル名1〉 〈ファイル名2〉

ただし，〈ファイル名1〉 =16進ダンプを取りたい元のファイル名

〈ファイル名2〉 =16進ダンプを書き出す先のファイル名

解説 :ファイルの16進ダンプを見たいときに利用する．

## 12 KEYIN

機能 :キーボードから入力してファイルを作成する．

入力形式:1. DO:KEYIN

解説 :プロンプト“K)” に続いて入力する． KEYINコマンドによって作成されたファイルは作業区域のトップに置かれる．キーボードからの入力の終了は“K)” につづいて “:::” で指示する．テキスト入力中の復改コードは，復改記号として解釈される．ただし，復改コードの直前の入力文字が継続文字（標準値は -で， PROFILEコマンドにより変更可能）である場合には復改コードは無視される．

## 13 LIST

機能 :ファイルの内容を表示する．

入力形式:1. DO:LIST

FILE:= 〈ファイル名〉

2. DO:LIST 〈ファイル名〉

解説 :“COPY 〈ファイル名〉 D” と同じ働きをする．

## 14 LOAD

機能 :指定されたファイルを作業区域のトップへ複写する．

入力形式:1. DO:LOAD

FILE:= 〈ファイル名〉

2. DO:LOAD 〈ファイル名〉

解説 :“COPY 〈ファイル名〉 W” と同じ働きをする．

15 LOOK

機能　　：作業区域のトップのファイルの内容を表示する．

入力形式：1. DO:LOOK

解説　　：“COPY W D”と同じ働きをする．

16 MOVE

機能　　：ファイルを移動する．

入力形式：1. DO:MOVE
　　FROM-FILE:=〈移動元ファイル名〉
　　TO-FILE:=〈移動先ファイル名〉
　　2. DO:MOVE 〈移動元ファイル名〉 〈移動先ファイル名〉

解説　　：ポインタのつけかえによってファイルの転送を行うので，ファイルの複製は行われない．したがって，異なる領域間での転送はこのコマンドではできない．

17 NICKNAME

機能　　：ファイルに別名をつける．

入力形式：1. DO:NICKNAME
　　FILE:=〈別名をつけたいファイル〉
　　NICKNAME:=〈別名〉
　　2. DO:NICKNAME 〈別名をつけたいファイル名〉 〈別名〉

解説　　：別名をつけるだけなので，元のファイルも別名もファイルの実体は同じもの（ID番号が同じ）である．したがって，異なる領域間で別名をつけることはできない．また，別名のついたファイルを除去してもバックアップ区域に移動せずに，別名数が1減るだけである．別名数が0になると初めてバックアップ区域に移動する．

18 PASSNUMBER

機能　　：ファイルのパスナンバーの設定，変更，解除をする．

入力形式：1. DO:PASSNUMBER
　　FILE:=〈ファイル〉
　　NEW PASSNUMBER FOR 〈ファイル〉:=〈4桁の数字〉
　　2. DO:PASSNUMBER 〈ファイル〉
　　NEW PASSNUMBER FOR 〈ファイル〉:=〈4桁の数字〉
　　3. DO:PASSNUMBER 〈ファイル〉
　　OLD PASSNUMBER FOR 〈ファイル〉:=〈4桁の数字〉
　　NEW PASSNUMBER FOR 〈ファイル〉:=〈4桁の数字〉
　　ただし，〈ファイル〉＝パスナンバーの設定，変更，解除をしたいファイル名

解説　　：パスナンバーを設定することができるのは，MEMO区域のファイルとSIGMA区域のファイルだけである．新たに作成したファイルにはパスナンバーはついていない（0になっている）ので，1,2の入力形式のように新しく設定するパスナンバーしか尋ねてこない．すでにパ

スナンバーがついているファイルに対しその変更をするときには，古いパスナンバーと新しいパスナンバーの両方を尋ねてくる．古いパスナンバーを知らないと変更はできない．また，解除するときは，新しいパスナンバーとして “0” か復改のみを入力すればよい．パスナンバーを設定すると，それ以降のファイルのアクセスにはパスナンバーが必要となる．パスナンバーを忘れたときのためにマスターキーも設定できる（PROFILE コマンド参照）．またパスナンバーの入力部分は，LOGファイルに記録されないので，LOGファイルの実行中でも，LOGファイルとは別に端末に尋ねてくることになる．

## 19 PROFILE

機能　：システム環境の設定，変更，保存，読み込みなどを行う．

入力形式：1. DO:PROFILE
　　PROF:〈サブコマンド〉
　　2. DO:PROFILE 〈サブコマンド〉
　　ただし，〈サブコマンド〉＝［BELL | CONTINUE | DISPLAY | END | HELPLEVEL | LINESIZE
　　| MASTERKEY | NEWLINE | PREFIX | RESTORE | SAVE | SYSOUTCLASS | TRIPLEDOT］

解説　：代用復改記号，継続記号などは記号が固定されていると，検索やソートをおこなうときに不都合が生じることがある．そこで，システム環境のうちいくつかは利用者が自由に設定できるようにした．システム環境を設定するには，プロンプト “PROF:” に続いてサブコマンドを投入し，それぞれのプロンプトに応じて入力すればよい．変更したシステム環境は，SIGMAを終わるとともに失われるが，SAVEサブコマンドでシステム環境を保存しておけば次に SIGMAに入ったとき，その値に設定される．PROFILE コマンドを終るには，END コマンドか復改のみを入力する．以下，各サブコマンドについて簡単に説明する．

(1) システム環境を設定するサブコマンド

(i) 質問型プロンプトの出るサブコマンド

BELL　：ベルモードを切り替える．“BELL MODE ON(Y/N)?” と聞いてくるので “Y” か “N” を入力する．省略すると，“Y” になる．システムが用意する初期値はONである．なお，フルスクリーン型の端末ではベルはならない．

MASTERKEY　：SIGMA 領域，MEMO領域それぞれにファイルのパスナンバーのマスターキーを設定する．まず，“MASTERKEY FOR MEMO OR SIGMA(M/S)?” と聞いてくるのでMEMO領域に対して設定するならば “M” を，SIGMA 領域に対して設定するならば “S” を入力する．“S” を選択した場合，“SIGMA SPACE USERID:=” というプロンプトでSIGMA 領域のある課題番号を聞いてくるのでそれに答える．マスターキーを変えるには，元のマスターキーを知っていなければならない．それを “OLD MASTERKEY:=” に続いて入力する．元のマスターキーがあっていれば，“NEW MASTERKEY:=” というプロンプトが出るので，新しくつけたいマスターキーを入力する．

PREFIX　：SIGMA 区域のファイル，MEMO区域のファイルに対してプレフィックスを設定する．まず，“MEMO OR SIGMA(M OR S)?” と聞いてくるので，MEMO領域に対して設定するならば “M” を，SIGMA 領域に対して設定するならば “S” を入力する．そのあと，“PREFIX:=” のプロンプトでプレフィックスを入力する．省略

値と初期値は，MEMO区域はプレフィックスをつけないことになっているが，SIGMA 区域は利用者自身の課題番号になっている．また，つけたプレフィックスが正しいかどうかの検査を行ってないので注意が必要である．プレフィックスの使い方については，３．４．７を参照のこと．

(ii) (i)以外のサブコマンド

サブコマンドを投入し，それぞれのプロンプトに対して新しく設定したい値を入力すればよい．

| サブコマンド名 | :プロンプト | :指定の制限 | :初期値 | :機能 |
|---|---|---|---|---|
| CONTINUE | :CONTINUE CHARACTER:= | :半角英数記号１文字 | : - | :継続記号を変更する |
| HELPLEVEL | :HELP LEVEL:= | :数字 | : 1 | :ヘルプレベルを変更する |
| LINESIZE | :LINESIZE:= | :1以上255以下の数字 | : 78 | :論理行サイズを変更する |
| NEWLINE | :NEWLINE CHARACTER:= | :半角英数記号１文字 | : ! | :代用復改記号を変更する |
| SYSOUTCLASS | :SYSOUT CLASS:= | :A,O,K,S,U,Q | : U | :出力クラスを変更する |
| TRIPLEDOT | :TRIPLEDOT CHARACTER:= | :半角英数記号１文字 | : . | :SEARCHコマンドのキーワードの指定に使うトリプルドット記号を変更する |

（注意）1. CONTINUE以外のサブコマンドでは，値の入力を省略すると変更せずにプロンプト“PROF:”に戻る．CONTINUEサブコマンドで入力を省略すると継続処理を行わなくなる．

2. フルスクリーン型の端末では，行サイズの指定は無効である．

(2) その他のサブコマンド

次に示すサブコマンドにはオペランドがない．サブコマンド名のみを入力すればよい．

DISPLAY ：システム環境を表示する．

END ：PROFILEを終わる．

RESTORE ：一時的に変更してあったシステム環境を保存されている環境に戻す．

SAVE ：変更したシステム環境を保存する．保存した環境は，次に SIGMAに入ったときの初期値になる．

## ２０ PUT

機能 ：作業区域のトップのファイルを指定の所に置く．

入力形式：1. DO:PUT
FILE:= <ファイル名>
2. DO:PUT <ファイル名>

解説 ：“MOVE W <ファイル名>” と同じ働きをする．

## ２１ REFILE

機能 ：SEARCHコマンドで検索した結果を再ファイル化する．

入力形式：1. DO:REFILE
REPORT(Y/N)? [Y | N]
QUESTION:= <質問番号>
NEW RECORD DELIMITER:= <新レコード区切り語>
NUMBERING(N/Y)? [N | Y]

OUTPUT-FILE:= <ファイル名>

ただし，REPORT(Y/N)? [Y | N]　　NUMBERING(N/Y)? [N | Y]

Y:検索件数を表示する．　　N:検索結果に通し番号をつけない．

N:検索件数を表示しない．　　Y:検索結果に10桁の通し番号をつける．

解説　：SEARCHコマンドによる検索結果はコード化されて索引区域のトップに置かれている．これを端末などに出力して見るためにはテキストファイルの形に復元する必要がある．この再ファイル化を行うのがREFILEコマンドである（3．5．8参照）．再ファイル化してできたファイルは，ヒットしたレコードを〈新レコード区切り語〉ではさんでつないだものである．“QUESTION:=”で復改のみを入力すれば，REFILEコマンドを終了する．

## 22 REPLACE

機能　：複数文字列の同時置き換えをする．

入力形式：1. DO:REPLACE

X01:= <置き換え前の文字列x1>

Y01:= <置き換え後の文字列y2>

.....

Xn:= <置き換え前の文字列xn>

Yn:= <置き換え後の文字列yn>

Xn+1:=↙

INPUT-FILE:= <置き換えを行うファイル>

OUTPUT-FILE:= <置き換え後のファイル>

REPORT(N/Y)? [N | Y]

ただし，　REPORT(N/Y)? [N | Y]

N:置き換え件数を表示する．

Y:置き換え件数を表示しない．

解説　：INPUT-FILEで指定したファイルの中の，Xiで指定した文字列をYiに同時に置き換えてOUTPUT-FILE で指定したファイルとする．REPORT(N/Y)?で“Y” を入力すると置き換えを行った件数を表示する．なお，Xi,Yiの文字列は，99組まで指定できる．

## 23 SAVE

機能　：作業区域のトップのファイルを指定のファイルに複写する．

入力形式：1. DO:SAVE

FILE:= <ファイル名>

2. DO:SAVE <ファイル名>

解説　：“COPY W <ファイル名>” と同じ働きをする．

## 24 SEARCH

機能　：複数キーの同時逐字検索を行う．

入力形式：1. DO:SEARCH

RECORD DELIMITERS

```
D01:= <レコード区切り語01>
.....
Dn:= <レコード区切り語n>
Dn+1:=↙
     ITEM DELIMITERS
Dn+1:= <項目区切り語01>
.....
Dn+m:= <項目区切り語m>
Dn+m+1:=↙
     KEYWORDS
A01:= <キーワード01>
.....
Ak:= <キーワードk>
Ak+1:=↙
     LOGICAL FORMULAS
Z01:= <論理式01>
.....
Zh:= <論理式h>
Zh+1:=↙
REPORT(Y/N)? [Y | N]
FILE:= <検索されるファイル名>
REPORT(Y/N)? [Y | N]
      Y:検索件数を表示する.
      N:検索件数を表示しない.
```

解説　：1. 二つのレコード区切り語にはさまれた部分文字列を仮想的なレコードとして検索する．項目区切り語は，論理式の評価の時点を指示するためのものである．

2. キーワードには通常の文字列の他にトリプル・ドット “...” を含むX1...X2...X3の形のキーワードも指定できる．トリプル・ドット “...” は任意の文字列を表す．すなわち，A...B...C はA の後にB ，更にその後にC がある文字列を表す（例えば，ABC,ATTBNNC,AABAACなどにマッチする）．トリプル・ドット “...” は PROFILEコマンドで変更可能である．

3. 論理式は，キーワード変数 “Ai” と論理記号and “.” ，or “,” ，not “^” ，括弧 “(” ，“)” を用いて作る．複数の論理式を登録することができる．論理式の指定についての詳細は３．５．４を参照されたい．

4. １回の検索が終了するたびに，プロンプト “FILE:=” に対し，ファイル名を次々に入力すると，同一の検索を異なるファイルに対して行うことができる．検索件数の右の欄はそれぞれの質問の累計件数である．ファイル名の指定にはワイルドカード “*” も使える．ワイルドカード “*” を使った場合，当てはまるファイルすべてについて検索を行い，その結果の累計を表示する．

5. SEARCHコマンドは検索結果をコード化して索引区域のトップに置く．それはREFILEコマ

ンドで再ファイル化することができる（3.5.8参照）.

25 SETMEMO

機能　：MEMO領域の初期化をする.

入力形式：1. SIGMA)SETMEMO
　　MASTERKEY:= <MEMO領域のマスターキー>
　　ALL THE CONTENTS IN THE MEMO SPACE WILL BE LOST
　　ARE YOU SURE(N/Y)? [N | Y]
　ただし，[N | Y] =N:NO, Y:YES

解説　：マスターキーを知っていないと領域の初期化はできない．また，SIGMA 状態でないとこのコマンドは使えない．このコマンド実行後には，MEMO領域の内容はすべて失われてしまうので，使用の際には細心の注意をされたい.

26 SETSIGMA

機能　：SIGMA領域の初期化をする.

入力形式：1. SIGMA)SETSIGMA
　　MASTERKEY:= <SIGMA 領域のマスターキー>
　　ALL THE CONTENTS IN THE SIGMA SPACE WILL BE LOST
　　ARE YOU SURE(N/Y)? [N | Y]
　ただし，[N | Y] =N:NO, Y:YES

解説　：マスターキーを知っていないと領域の初期化はできない．また，SIGMA 状態でないとこのコマンドは使えない．このコマンド実行後には，SIGMA 領域の内容はすべて失われてしまうので，使用の際には細心の注意をされたい.

27 SORT

機能　：1. ファイルを定められた区切り語で区切り，ソートする.
　　2. 索引，逆引き辞書等を作成する.

入力形式：1. DO:SORT
　　DELIMITERS
　　DO1:= <レコード区切り語01>
　　……
　　Dn:= <レコード区切り語n>
　　Dn+1:=↙
　　KEYWORDS
　　K01:= <切り出し仕様01>
　　……
　　Km:= <切り出し仕様m>
　　Km+1:=↙
　　REFILE UNIT:= <再ファイル化仕様>
　　(NEW DELIMITER:= <新区切り語> )
　　INPUT-FILE:= <ソートの対象となるファイル名>

OUTPUT-FILE:= 〈ソートの結果を格納するファイル名〉
LISTING(N/Y)? [N | Y]
ただし，〈切り出し仕様〉については3.6.3参照.
〈再ファイル化仕様〉 = [R | Ki | Ki# | KiL | #Ki | 〈切り出し仕様〉]
R : 元のレコード単位で再ファイル化する.
Ki : Kiで切り出されたキーそのものを再ファイル化する.
Ki# : Kiで切り出されたキーに，頻度をつけて再ファイル化する.
KiL : Kiで切り出されたキー，それが現れたレコード番号のリストをつけて再ファイル化する.
#Ki : Kiで切り出されたキーを頻度で再ソートして再ファイル化する.
〈切り出し仕様〉 : この仕様で切り出される付属情報のみを，再ファイル化する.
[N | Y]
N : 切り出されたキーを順次表示しない.
Y : 切り出されたキーを順次表示する.

解説 : SORTコマンドの機能は複雑なので，詳しくは2.4，2.5の使用例や，3.6の利用法を参照されたい.

## 28 SPACE

機能 : MEMO領域の空き領域を表示する.
入力形式 : 1. DO:SPACE
解説 : MEMO領域の現在の大きさや空き領域に関する情報を表示する.

## 29 TSS

機能 : TSSコマンドを呼び出す状態に切り替える.
入力形式 : 1. DO:TSS
TSS: 〈TSSコマンド〉
2. DO:TSS 〈TSSコマンド〉
解説 : このコマンドを実行するとプロンプトが“TSS:”となり，LOGOFF,PROFILEなどの一部のコマンドを除くTSSのコマンドが実行できる．この状態からDO状態に戻るには，ENDを入力するか復改のみを入力すればよい.

## 5. おわりに

SIGMA システムは次のような段階を経て開発された.

0版: 1979年　ミニコン版プロトタイプ完成（九大理学部基礎情報学研究施設 PANAFACOM U-400）

1版: 1981年　SIGMA 第1版公開（九大センター FACOM M-200）

1983年　高速パタン照合アルゴリズムの開発

1985年　日本語テキスト用のパタン照合アルゴリズムの開発

2版: 1987年　日本語テキスト対応 SIGMA第2版公開（九大センター FACOM M-780）

今回の SIGMAシステムの改訂（SIGMA プロジェクト）は，研究室の関係者のほとんど全員がなんらかの形で参加して行われた．以前から折りに触れて，SIGMA システムに関する話題が，セミナーでも取り上げられていたが，昨年度から本格的に SIGMAプロジェクトが動きだした．

SIGMA プロジェクトは，第1版の開発に携わった担当者から SIGMAシステムの概要，開発のコンセプトなどを SIGMAを知らない世代である学生達が聞くことから始まった．日本語テキストを扱えるようにすることが今回の改訂の重要課題であったが，その他にも改良すべき SIGMA第1版の問題点や新しく付け加えたい機能などについていろいろなアイデアが出されて，次第に改訂の要点が明らかにされていった．たとえば，コマンドの名前・構造の見直し，新しい機能の追加，LOG ファイルをコマンドプロシジャとして利用しやすくすること，ヘルプ機能を作ること，ファイル管理に関する問題などである．今回の改訂ではシステムそのものだけではなく開発環境についても工夫し，FORTRAN77 のソースプログラムにC言語のマクロを記述してプリプロセスした．これによって，プログラムはより見やすくなり，共通ルーチンの定義や組み込みも容易になって，開発の作業効率がかなり向上した[22]．開発作業は，メンバーをファイルシステム班，SORT班，コマンド班，REPLACE 班，端末・HELP班の五つに分けて行った．SIGMA の利用者との会議も持ち，利用者が求めているものは何かを聴く機会も設けた．SIGMA については，今後も新しい機能の追加などの研究開発を行う予定である．

本稿では，テキストデータベース管理システムSIGMA 第2版について使用法を説明した．第2版では，フロッピーディスクなどにテキストファイルとして蓄えられている日本語テキストが広く扱えるようになり，検索やソーティング，置き換えなどの各種機能も強化された．多くの利用者に活用されることを期待したい．九大センターの公用データベース制度[18]により，さらに多彩なテキストデータベースが構築され，また，大学間ネットワーク [16,27]を通じて遠隔地からの利用がより多くなることを望みたい．

最後に，改訂を行うにあたって，有益な助言をいただいた九大教養部樋口忠治教授ならびに九大農学部多田内修助手に感謝します．また，種々の作業に日夜協力していただいた伊藤博氏，竹田正幸氏をはじめとする研究室の諸氏に感謝します．

### 参考文献

[1] Aho, A.V. and Corasick, M.J.: Efficient String Matching : An Aid to Bibliographic Search, Comm.ACM, Vol.18, No.6, pp.333-340, (1975).

[2] Arikawa, S. and Kitagawa, T.: Multistage Information Retrieval System Based on Researcher Files, Proc. 2nd USA-Japan Computer Conf., pp.149-153, (1975).

[3] 有川，篠原，白石，玉越: 研究者向き情報システムSIGMAについて，九州大学大型計算機センター広報，Vol.14, No.4, pp.550-573, (1981).

[4] Arikawa, S., Shinohara, T., Shiraishi, S. and Tamakoshi, Y.: SIGMA - An Information System for Researchers Use, Bull. Inform. Cybernetics, Vol. 20, No. 1-2, pp. 97-114, (1982).
[5] Arikawa, S. and Shinohara, T.: A Run-Time Efficient Realization of Aho-Corasick Pattern Matching Machines, New Generation Computing, Vol. 2, No. 2, pp. 171-186, (1984).
[6] Arikawa, S. and Shiraishi, S.: Pattern Matching Machine for Replacing Several Character Strings, Bull. Inform. Cybernetics, Vol. 21, No. 1-2, pp. 101-111, (1984).
[7] 有川, 篠原: 文字列パターン照合アルゴリズム, コンピュータソフトウェア, Vol. 4, No. 2, pp. 2-23, (1987).
[8] 有川ほか: テキストデータベース管理システム SIGMAについて, 情報処理学会研究報告, Vol. 87, No. 65 (FI-6), pp. 6. 4. 1-6. 4. 8, (1987).
[9] Boyer, R. S. and Moore, J. S.: A Fast String Searching Algorithm, Comm. ACM, Vol. 20, No. 10, pp. 762-772, (1977).
[10] 樋口, 篠原: 公用データベース トーマス・マン・ファイル／SIGMAの公開, 九州大学大型計算機センター広報 Vol. 16, No. 4, pp. 379-393, (1983).
[11] 樋口, 篠原: 公用データベース「トーマス・マン・ファイル」のファイル追加について, 九州大学大型計算機センター広報 Vol. 18, No. 2, pp. 73-79, (1985).
[12] 樋口忠治: 「前置詞」bis の用法について, 独仏文学研究, Vol. 36, pp. 183-210, (1986).
[13] 樋口, 篠原: テキスト・データベース「トーマス・マン・ファイル」の完成と再編成について, 九州大学大型計算機センター広報 Vol. 20, No. 6, (1987).
[14] Inose, H. (ed.): Scientific Information Systems in Japan, North-Holland, (1981).
[15] Knuth, D. E., Morris, J. E. and Pratt, V. R.: Fast Pattern Matching in Strings, TR CS-74-440, Stanford Univ., (1974).
[16] 九州大学大型計算機センター, 利用の手引, ネットワーク編第4版, (1987).
[17] 九州大学大型計算機センターニュースNo. 361, FMR-60によるMS-DOSとホスト(OS IV/F4 MSP)間のファイル転送について, (1987).
[18] 松尾, 二村, 高木: 公用データベースについて, 九州大学大型計算機センター広報, Vol. 15, No. 2, pp. 222-227, (1982).
[19] 佐伯梅友校注, 日本古典文学体系8 古今和歌集, 岩波書店, (1958).
[20] 篠原, 有川: 日本語テキスト用の Aho-Corasick 型パターン照合アルゴリズム, 情報処理学会研究報告, Vol. 85, No. 48 (NL-52), pp. 52. 4. 1-52. 4. 8, (1985).
[21] 篠原, 二村, 松尾: 情報検索システム AIRの改訂について, 九州大学大型計算機センター広報, Vol. 20, No. 3, pp. 211-238, (1987).
[22] 篠原, 川崎, 有川: マクロ機能を用いたFORTRAN, 昭和62年度電気関係学会学会九州支部連合大会講演論文集, p. 501, (1987).
[23] 多田内修: SIGMA による公用データベース昆虫学データベース(KONCHU)の公開とその利用法, 九州大学大型計算機センター広報 Vol. 20, No. 6, (1987).
[24] 平良豊: 日本語対応 TSSグラフィック端末プログラム, 九州大学大型計算機センター広報, Vol. 20, No. 3, pp. 183-192, (1987).
[25] 武政尹士: C言語による日本語 TSS端末エミュレータ, 九州大学大型計算機センター広報, Vol. 20, No. 3, pp. 193-210, (1987).
[26] 武谷, 酒井, 有川: 一方向逐字処理によるキーの切り出しについて, 昭和62年度電気関係学会九州支部連合大会講演論文集, p. 502, (1987).
[27] 全国共同利用大型計算機センターデータベース連絡会編, 全国共同利用大型計算機センターオンライン・データベース利用ガイド第6版, (1986).

## 付録

### コマンド一覧

SIGMA システムでは SIGMA状態またはDO状態において以下のコマンドが使用可能である．ただし，SETMEMO, SETSIGMAコマンドは SIGMA状態の時のみ入力できる．コマンドを入力するときは，下線部だけ入力すれば十分である．

| | | | |
|---|---|---|---|
| 1 | CATENATE | : | ファイルを連接する |
| 2 | COMMENT | : | ファイルに見出しをつける |
| 3 | COPY | : | ファイルの複製を作る |
| 4 | DDIRECTORY | : | ディレクトリを詳細情報とともに表示する |
| 5 | DELETE | : | ファイルを除去する（消去ではない） |
| 6 | DIRECTORY | : | ディレクトリ（ファイル名一覧）を表示する |
| 7 | ECHO | : | ファイルに書かれたコマンドの実行中にその内容を表示するかどうかを設定，または端末にメッセージを出力する |
| 8 | END | : | SIGMA状態を終える，DO状態を終える |
| 9 | GET | : | 指定されたファイルを作業ファイルへ取ってくる |
| 10 | HELP | : | SIGMAで使用できるコマンドや用語の説明を表示する |
| 11 | HEXDUMP | : | ファイルの16進ダンプをとる |
| 12 | KEYIN | : | キーボードから直接作業ファイルに入力する |
| 13 | LIST | : | ファイルの内容を表示する |
| 14 | LOAD | : | 指定されたファイルの内容を作業ファイルに複製する |
| 15 | LOOK | : | 作業用ファイルの内容を表示する |
| 16 | MOVE | : | ファイルを移動する |
| 17 | NICKNAME | : | ファイルに別名をつける |
| 18 | PASSNUMBER | : | パスナンバーを設定したり，変更したりする |
| 19 | PROFILE | : | システム環境を表示したり，設定したりする |
| 20 | PUT | : | 作業ファイルを指定された所に置く |
| 21 | REFILE | : | 検索結果を再ファイル化する |
| 22 | REPLACE | : | 複数文字列の同時置き換えをする |
| 23 | SAVE | : | 作業ファイルの内容を指定されたファイルへ書き出す |
| 24 | SEARCH | : | 複数キーの同時逐字検索を行う |
| 25 | SETMEMO | : | MEMO領域を初期化する |
| 26 | SETSIGMA | : | SIGMA領域を初期化する |
| 27 | SORT | : | レコードをソートする |
| 28 | SPACE | : | MEMO領域の使用可能領域の大きさを表示する |
| 29 | TSS | : | TSSコマンドを呼び出す |

PROFILE サブコマンド一覧

PROFILE コマンドにおいて以下のサブコマンドが使用可能である．サブコマンドを入力するときは，下線部だけ入力すれば十分である．

| | | |
|---|---|---|
| BELL | : | ベルモードを切り替える |
| CONTINUE | : | 継続記号を変更する |
| DISPLAY | : | システム環境を表示する |
| END | : | システム環境の設定を終える |
| HELPLEVEL | : | ヘルプのレベルを変更する |
| LINESIZE | : | 端末の論理ラインサイズを変更する |
| MASTERKEY | : | SIGMAファイル，MEMOファイルのマスターキーを変更する |
| NEWLINE | : | 代用復改記号を変更する |
| PREFIX | : | SIGMAファイルを参照するときのプレフィックスを変更する |
| RESTORE | : | システム環境を読み込む |
| SAVE | : | システム環境を保存する |
| SYSOUTCLASS | : | 出力クラスを変更する |
| TRIPLEDOT | : | SEARCHコマンドでキーワード指定に使うトリプル・ドットのためのドット記号を変更する |