人文科学とコンピュータ　18－3
（１９９３．５．２１）

暦日研究における計算機利用の一形態

―― 『日本暦日便覧』作成の事例報告 ――

湯浅 吉美

成田山仏教研究所

旧暦行用時代の日本の暦日を研究する上で、筆者がどのようにコンピュータ、とくに汎用大型計算機（メイン・フレーム）を利用しているか、またパーソナル・コンピュータとの関係を如何なるものと考えているか、という点について述べる。具体的には、拙著『日本暦日便覧』（東京，汲古書院，1988）作成時に創成したデータ・ファイルの構成と、それを用いての作表や索引作成のアルゴリズムの概略とを紹介しながら、一個人としての計算機利用のあり方を示す。十数年前のハードウェア、ソフトウェア両面の制約の中で、言わば苦肉の策として考え出された手法を回顧することにより、安易なプログラムレス志向に対する疑問を提示したい。

A style of using computers in the study of Japanese Historical Calendars

—— A case report on making "NIHON REKIJITSU BINRAN" ——

Yoshimi YUASA

Naritasan Institute for Buddhist Studies
1 Narita, Narita-shi, Chiba 286, Japan

In my studies of Japanese historical calendars called *kyu-reki*, how do I use computers, especially main frames? And what is my opinion upon the relation, or the difference, between main frames and personal computers? These two points are discussed here. In the concrete, I show the structure of file organized at the time of my publishing *NIHON REKIJITSU BINRAN* (Tokyo, Kyuko-shoin, 1988). I also give an outline of algorithms. Though they were devised as the last resort under the limited circumstances in the early 1980's, you can see my fear about programmingless-oriented use in recent years.

1．はじめに

周知の如く、明治５年に現行の太陽暦が採用される以前、日本では１２００年の永きにわたり、一般に旧暦と呼ばれているところの太陰太陽暦が行われていた。ところが、中国伝来の太陰太陽暦法の下では、年ごとに複雑な暦計算を済ませた後でなければ、毎月の大小日数さえ確定せず、たとえば資料に「七月晦日」と見えても、それが２９日なのか３０日なのか、その年の暦が無ければ判断できない。まして、「七月甲子の日」などと干支で表記されては、当月朔日の干支がわからない限り、何日のことかを知るのも不可能である。その上、これらを簡便な早見表にまとめうるような規則性も無いため、どうしても毎年の暦が必要となる。

ところで、その間、実際に使用された暦は、江戸時代については二百数十年を通じてほぼ完全に、しかも内閣文庫に一括して現物が残っているため、あまり問題は無い。しかしそれ以前となると話は別で、年次の点でも断続的、たまたま残っている年でも通年の遺品は稀、さらにそれが全国の諸文庫に分蔵されているとあっては、実に厄介と言わざるを得ない。ここに実施暦復元の必要性が生ずるのである。

2．暦計算について

さて、実施暦の復元に当たっては、まず最初に、当時の暦法そのものが記された資料に準拠して、計算の方法を理解しなければならない。日本で行用された暦法は、時代の変遷につれて８、乃至９種類あるが、いずれも中国から導入された暦法に基づいており、基本的なロジックに変わりは無く、暦を構成するための骨格となる要素として、常気、経朔、定朔の値を算出することが第一のステップとなる。

a．常気を求める

常気は二十四気とも、また現代では二十四節気とも呼び、よく知られているものとして夏至、冬至、春分、秋分、立春などがある。中国では紀元前数世紀頃から使われており、暦の上で１太陽年を２４等分する定点となる。もっとも、天文学的に厳密には二十四気の間隔が不等であることは当時においても知られていたが、暦法上は便宜的に等間隔として扱い、向こう１年間の二十四気の日時を２４、乃至２６件、計算する

b．経朔を求める

経朔とは暦算上の理論的朔である（朔とは太陽と月の視黄経が一致する瞬間をいい、それを含む日を朔日という）。まず前年１１月の経朔（天正経朔）を求め、それを起点として、造暦する１年間の毎月の経朔を算出する。

ここまでの暦算は、基本的には整数除算の剰余をとる操作の繰り返しであり、さまで複雑なものではない。入力データというようなものは一切無く、年ごとに異なる変数は「積年」のみである。積年とは甲子夜半朔旦冬至（上元とも称し、中国暦学において想定される天体運行の開始点）から当代までの年数であるから、これとて毎年１ずつ漸増させるだけですむことは自明であろう。

c．定朔を求める

経朔は容易に求まるけれども、「暦算上の理論的朔」では実用にならないので現実的な朔（定朔）を計算する必要がある。実は、この定朔計算こそが暦算の本体ともいうべきもので、相当に複雑な手順となるが、要するに、a．b．で得られた計算値に対して、条件判断と配列操作とを繰り返し、太陽と月との運行の不規則性に由来する補正を施すのである。

かくして常気と定朔とが求まったならば、それらを適当な表形式にすれば、クロノロジカルな意味での造暦は完了したことになる。

3．暦日データの構成

前章でも触れたように、復元暦の作成には本来的に入力データなるものは存在しない。したがって、如何に複雑であろうとも、暦算のロジックさえ正しくプログラミングできれば、得られた結果を表の形にすることは、いとも簡単にできそうに思われるが、実際にはそうはいかない。というのは、前近代社会の常として、一言でいうならば縁起の善し悪しを理由として、計算結果に手心を加えたケースが頻繁に見られる、という事情に因る。史料によって確認される「手心を加えたケース」（暦日変更と呼ぶ）の全てについて、プログラム中で機械的に処理することは事実上、不可能であるのみならず、それをしていては、計算機を用いて自動的に実行する意義が薄弱になってしまう。さらには、新たな暦日変更が確認されるたびに、プログラム自体の修正を余儀なくされることにもなる。そこでこの仕事においては暦算と作表とを別個に考え、作表に必要な各種の暦日事項を集め、1年分を1件とするデータレコードを構成し、ファイルとしてもつことにした。これを便宜上、暦日データと呼んでいる。暦日変更についてはデータ作成の段階で手作業により、「手心を加えた」データに直した上で作表するわけで、省みれば当時の司暦官の行為そのもののシミュレーションともいいうる。またこうすることにより、月朔干支・二十四気該当日の日付・同じく干支、の3種類の索引を作成できたことは、当初予定しなかった副産物である。

以下、暦日データの構成を紹介する。アルファベット表記の奇妙な和英折衷の語はデータ項目の変数名（ＣＯＢＯＬでは「一意名」という）、コロンの後がその属性で、「数字」は内容的には0～9の数字でも文字項目として扱うことを示す。また「＿」（アンダーバー）は空白を意味する。

ＡＧ－ＴＡＢＬＥ：数値（０１～７５）２桁＊４

日本年号を漢字表示するための数値。２４０以上ある日本年号を通じて、使用される文字種は僅かに７４に過ぎない。これに空白を加えて７５字の漢字表をプログラム内部にもち、この数値をポインタとして、しかるべき漢字を得る。奈良時代に行われた4字年号に対処すべく年号欄が4文字分あるため、2字年号の2字目と4字目とに明に空白を指示しておく。

ＮＥＮ：数字（＿１～３４）２桁

年数。改元当年は新年号元年と表示する。いうまでもなく、元年が「＿１」。

ＳＥＩＲＥＫＩ：数値（０６９２～１８７２）４桁

その年の西暦年。持統天皇6年が「０６９２」、明治5年が「１８７２」。

ＴＳＵＫＩ－ＮＡＭＥ：数値（００，０１＝正月～２４＝閏１２月）２桁＊１３
月の呼び名の並び。閏月の出現が不規則なので、平年も含めて明に指示する。閏月がある場合、１年が１３か月となるが、平年では第１３月に「００」を入れてある。たとえば閏正月のある年のデータは「０１１３０２０３０４０５０６０７０８０９１０１１１２」となる。

ＴＳＵＫＩ－ＮＩＳＳＵ：数値（０，１，２）１桁＊１３
月の日数の並び。太陰太陽暦では月の大小は年ごとに異なり、規則性が無い。そのため、毎月の大小を与える必要がある。余談ながら、この並びが大小大小・・・と完全に交互で１年を成したことが、歴史上、１回だけある（仁和４年（８８８））。
１＝小月（２９日），２＝大月（３０日），０＝なし（平年の第１３月）

ＴＯＴＡＬ－ＮＩＳＳＵ：数値（３５３～３８５）３桁
年間の日数。月の大小、閏月の有無により、３５３、３５４、３５５、３５６、３８３、３８４、３８５の７通りがある。

ＫＡＮＳＨＩ－ＮＥＮ：集合項目
ＫＡＮ－１：数値（０１＝甲～１０＝癸）２桁
ＳＨＩ－１：数値（０１＝子～１２＝亥）２桁
その年の六十干支。もっぱら表示の用途だけなので、十干と十二支とに別けておき、それぞれの文字テーブルのポインタとする。

ＫＡＮＳＨＩ－ＧＡＮＪＩＴＳＵ：集合項目
ＫＡＮ－２：数値（０１＝甲～１０＝癸）２桁
ＳＨＩ－２：数値（０１＝子～１２＝亥）２桁
ＥＴＯ－２：数値（０１＝甲子～６０＝癸亥）２桁
元日の六十干支。この内、ＫＡＮ－２・ＳＨＩ－２は前項ＫＡＮ－１・ＳＨＩ－１に準ずる。一方、年干支と異なり、行カウント等の計算に用いるため、ＥＴＯ－２として六十干支の番号をも別にもっている。

ＡＧ－ＮＡＮＣＨＯ：文字２桁＊２
南北朝時代において南朝年号を併記するために、その漢字コードをもつ。ルーチンを簡略化すべく、ＫＩＳコードで入力してある。因に、ＫＩＳコードとは、連想記憶２ストローク入力方式と呼ばれ、主としてカタカナ、部分的に英数字をも用いて、２文字で漢字１字のコードを入力するものである。例えば「暦日」は「コヨミニチ」という具合。音訓（の一部）と一致する字も多く、慣れれば使い易いものであったが、カナ漢字変換が当たり前となった今日では全く捨てて顧みられない。

ＳＫ－ＳＴＡＲＴ：数値（１，２，３）１桁
その年最初に出現する二十四気の番号。本来、二十四気は冬至を第１番目とするが、暦年単位で作表処理する際の便宜上、大寒を１とした。
１＝大寒，２＝立春，３＝雨水

ＳＫ－ＤＡＴＥ：数値２桁＊２６

各月２回の二十四気該当日を示す数値。貞享２年（１６８５）の改暦以前はその日の干支（００～５９）、以後は日付（０１～３０）で入力した。貞享改暦以前は暦算の結果をそのまま用いる方が安全であり、以後は実用暦が完全に残っていて日付が確認できる、という事情による。なお、閏月の挿入に伴う月１回の場合と、平年のときの第１３月とには、「９９」を入れて「なし」を明示してある。

ＳＫ－ＲＥＫＩＨＯ：文字４桁＊２

その年に行用されていた暦法名（２文字）を表示するための漢字コード。使用メインフレームである富士通ＦＡＣＯＭ　Ｍシリーズの標準コード体系、ＪＥＦコードを入れた。ＪＥＦコードは、空白（４０４０Ｈ）を除き、ＪＩＳ１６進コードに８０８０Ｈを加えたものとなっている（例：亜（３０２１Ｈ）→Ｃ０Ｂ１Ｈ）。

ＰＲＯＣ－ＳＷ：文字（＊，＿）１桁

ＳＫ－ＤＡＴＥの項で述べたように、貞享改暦の前後でデータの入力方法が異なり、それにつれて、通過するルーチンも当然、別になる。その切り替えのためのフラグ。

＊＝貞享改暦以前（～１６８４），＿＝貞享改暦以後（１６８５～）

ＫＡＩＧＥＮ－ＦＬＡＧ：数字（０，１，２）１桁

改元の日付を表示するために通るルーチンを切り替えるフラグ。

０＝その年には改元なし，１＝１件，２＝２件

ＫＡＩＧＥＮ－ＤＡＴＥ１：集合項目

ＫＡＩＧＥＮ－ＭＭ１：数値（０１＝正月～２４＝閏１２月）２桁

ＫＡＩＧＥＮ－ＤＤ１：数字（＿１～３０）２桁

ＫＡＩＧＥＮ－ＮＳ１：文字（Ｎ，Ｓ，＿）１桁

改元日付のデータ。ＭＭ１は月名（テーブルのポインタとして用いるので数値扱い）、ＤＤ１は日付（そのまま出力に送るので文字扱い）。ＮＳ１については、通常は空白で、南北朝時代の両朝年号併記のとき、その別を示すために転記する文字を切り替えるフラグになる。

ＫＡＩＧＥＮ－ＤＡＴＥ２：集合項目

ＫＡＩＧＥＮ－ＭＭ２：数値（０１＝正月～２４＝閏１２月）２桁

ＫＡＩＧＥＮ－ＤＤ２：数字（＿１～３０）２桁

ＫＡＩＧＥＮ－ＮＳ２：文字（Ｎ，Ｓ，＿）１桁

改元日付のデータ。前項に準ずる。南北朝時代など、ごく稀に１年の内に２件の改元日付が存在するケースがある。

ＫＧ－ＨＤＲ－ＮＵＭ：数値（０，１，２）１桁

ごく些細な表現上のコダワリのためのフラグ。

ＪＵＬＩＡＮ－ＤＡＴＥ：数値７桁

その年の元日のユリウス通日。ユリウス通日は、西暦紀元前４７１３年１月１日を元期とする通し日数で、暦法の相違に煩わされることなく、ある特定の一日をポイントできるため、天文学などの分野で用いられている。和暦の研究でも西暦日への換算や七曜の算出を数式化する際、重宝なものである。因に拙著の暦日表の最初の日、持統天皇６年元日は西暦６９２年１月２４日で、ユリウス通日は１９７３８３４日。

以上のデータ項目から成る１年分１６０バイト（末尾に３バイトのＦＩＬＬＥＲ項目が付く）を１レコードとして、順編成データセットを構成している。

【再定義項目について】

なお付言すると、例えばＡＧ－ＴＡＢＬＥのように、「～桁＊～」となっている項目は再定義して添字付きで参照される。再定義というのは、あるデータ項目を別の一意名と形式とでも記述しておくＣＯＢＯＬ言語の手法で、ＡＧ－ＴＡＢＬＥの場合、２桁ずつ４個のＡＧ－ＴＢＬとして再定義することにより、ＡＧ－ＴＢＬ（ｎ）という添字付きの形で扱えるようになる。ＢＡＳＩＣ言語などで「配列の宣言」（ＤＩＭ文）と称するものに相当するが、再定義の場合、必要に応じて元の項目名で全体を一括して参照できる。ただし厳密にいうと、その場合は文字項目扱いとなるわけで、「数値２桁＊４」なる書き方は再定義した上でのＡＧ－ＴＢＬのデータ形式である。

４．作表のアルゴリズム

ａ．暦日表の作成

本体を成す暦日表の作成は、アルゴリズムとしてはすこぶる単純なものである。プログラム内部に作業領域（working-storage)として確保した６０行７列のテーブルに日付を埋めてやり、それを１行ずつ出力に送る操作を６０行分、繰り返せばよい。すなわち、左上隅（１行１列）に元日を意味するデータ「０１０１」を入れ、日数のカウンタをプラス１して２行１列に正月２日「０１０２」を入れ、また日数カウンタをプラス１して・・・、というように行方向に進んでいく。この間、常に当月の日数と比較しながら進み、それを越えたところで月の方をプラス１して、日付を１に戻す。こうして６０行終了したならば、行カウンタをイニシァライズし、列カウンタをプラス１する。７列目まですべての記入が終われば、出力に移る。この度は先に列方向へ進み、１行１列から１行７列まで出力領域への転記を繰り返し、書き出したなら、２行目に進む。概要を示せばこれだけの作業である。他に二十四気についても処理しなければならないが、これも似たようなことで、上述のテーブルを埋めていく過程で、常に当日がテンポラリの二十四気該当日か否かをチェックして、該当日ならばその行・列のデータに二十四気の番号を書き込み、次の二十四気データを比較対象のテンポラリに読み込んでやる。

以上のようなネスティングループによる表操作はＣＯＢＯＬの場合、ＰＥＲＦＯＲＭ文を用いることで容易に実現できる。もちろんＦＯＲＴＲＡＮのＤＯループやＢＡＳＩＣのＦＯＲ～ＮＥＸＴループ等でも同様である。

実際のプログラムでは、このほかにヘッダ行の処理など、いくつかの細々した手順を踏んでいるが、基本的なロジックは以上に尽くされており、あとは如何に見易い日本語表示に変換するか、と

いう問題になる。この点についてもすでに触れたように、年号に使用する漢字・月名・十干・十二支など、日本語表示すべきものの日本語文字直定数テーブルをプログラム内部にもち、それぞれを指示する数値データをポインタとしてそれを参照し、出力領域に送り出す、という方法で解決している。この方法は今日では迂遠に相違ないが、出現する文字が予め分明で、数的にも限られているならば、存外有効な手法といえるのではないであろうか。

b．索引の作成

拙著には「月朔干支索引」・「二十四気日付順索引」・「二十四気干支別索引」という３種の索引が収録されている。次にこれらの索引作成のアルゴリズムについて述べる。いずれも同趣のロジックから成り、基本部分は上述ａ．のルーチンを流用したものである。つまり、６０行７列のテーブルを満たしていく作業を行いながら、必要なデータに当たった時だけ、それを出力ファイルに書き出しておき、後でそのファイルを入力ファイルとしてソートすればよい。

たとえば「月朔干支索引」の場合、テーブルを埋めながら、日付が「０１」の時のみ、西暦年・月名・月朔干支から成る出力レコードを書き出す。このレコードは僅か８バイトで、レコード数は１４６００件ほどである（１２か月＊１１８１年間＋閏月数）。こうして月朔干支データのファイルを創成した後、それを入力としてソートプログラムを起動する。ソートの第１キーは「月名」、第２キーは「月朔干支」で、出力ファイルに書き出す。しかる後、それを用いて適当な形式の表にして印刷出力とする。

他の２種の索引でも要領は同じで、切り出すデータとソート時のキーが異なるだけである。マッチングしたならば、西暦年・日付（月日）・当日の干支・二十四気名から成る出力レコードを書き出す。このファイルは１レコード１２バイトで２８３００件ほどになり、２種の索引作成に共通に使用される。ソートに当たっては「二十四気日付順索引」ではソートキーを二十四気名・日付の順とし、「二十四気干支別索引」の場合は二十四気名・干支の順とする。

以上でアルゴリズムの大概は理解されたことと思う。

５．おわりに

本稿の内容は少しく古めかしく、回顧的であるが、その後、現在に至るまで、暦日研究の分野で計算機が有効に活用された例は多くないので、敢えて発表させていただいた。今後、これを汎用性のあるデータベース・システムに発展させることも検討中なので、機会を改めて報告したいと思う。結びに代えて、計算機利用の現状につき、筆者が日頃抱いている感想を述べておこう。

たしかに近年のパーソナルコンピュータ・ワークステーション等の進歩と、それに伴うあらゆる分野での利用の拡大とは、十年以上前には予想だにしなかったほど著しいものがある。しかし、とりわけ人文科学における様相を見るとき、はたして地に足の着いた着実なものといえるであろうか。とくに、パーソナル・ユースにおける安易なプログラムレス志向には、冷静に反省を加えるべきであると思われてならない。というのも、そもそもプログラミングという作業が問題解決の手順を記述することである以上、いやしくも研究者が自らの仕事に用いるならば、何を措いても他人任せにできない部分である、と考えるからである。人文系研究者に対する情報処理教育（単なるパソコン操作法ではなく、真の情報科学として）が必要とされる時期に来ているのではあるまいか。ともあ

れ、大型計算機とパーソナルコンピュータとの使い分けの次第を記すので、以て信条の一端を汲み取っていただきたい。

繰り言になるけれども、この仕事が緒に就いた頃のパーソナルコンピュータは、システム自体やＢＡＳＩＣ言語のバグの話が絶えなかった。また、ＣＯＢＯＬなど、ＢＡＳＩＣ以外の高級言語を使うためには、本体価格に数倍する資金が要求された。素人でも安心して使えるのは、むしろ大型計算機だったのである。

とはいえ、小回りのききにくい大型計算機であらゆる作業を進めることは、なるほど面倒なことに違いない。そこで、機の熟するのを待ち、筆者もパーソナルコンピュータを導入した。その役割分担は、

・暦日データのチェック。例えば、入力された西暦年が正しくシーケンシァルになっているか否かの確認など。
・大型計算機用に書かれたプログラムのテストやデバッグ。使用言語を異にしようとも、十分な注意を払ってコンバートすれば、アルゴリズムの検証が可能。

という具合である。

かかる観点から、この仕事全体を通じて、彼此４０本近くのプログラムがパーソナルコンピュータ上で記述され、実行された。しかしながら、ここで銘記すべき事実は、そのいずれもが数十行程度の短小なものである、ということであろう。１本のプログラムでは、必ず一つの事象だけを検査している。その理由は簡単で、偏に論理的整合性の保証を確実ならしめるためである。しかもそれらはあくまでも大型計算機上で実行されるメインプログラムの「縁の下の力持ち」であって、決して表面にまで存在を露にするものではない。

【作業環境について】

最後に、この点をまとめて明記しておく。主要作業は全て慶應義塾大学計算センター（三田）を利用して行った。ハードウェアは富士通ＦＡＣＯＭ　Ｍシリーズ（年々機種更改があった）、ＯＳはＯＳⅣ／Ｆ４、言語はＪＩＳ－ＣＯＢＯＬ（後にＣＯＢＯＬ'85）で、日本語処理はＪＥＦシステムによりサポートされている。とくにコード変換などは、ＪＥＦ配下の「ＡＤＪＵＳＴ」からサブルーチンを組み込んだ。また索引作成過程におけるソートプログラムもＯＳ提供のものを用いている（ＯＳⅣ／Ｆ４　ソートマージ）。ソフトウェアのヴァージョンも適時更新された。

補助作業に使用したパーソナルコンピュータは富士通ＦＭ－１１ＡＤで、この仕事の段階では、いわゆるＤＯＳは用いず、もっぱらＦ－ＢＡＳＩＣver4.0にてプログラミングした。

両者間のデータ交換は８インチ標準フロッピィディスクを媒体とするオフライン・ファイル渡し（要するに、データが記録されたＦＤを持ち運ぶ方法）で、Ｍシリーズ側ではＯＳⅣ／Ｆ４上のユーティリティ「ＪＳＤＦＬＰＤＫ」、パーソナルコンピュータ側では富士通提供の「ファイルコンバータ」を介して、データの授受を実現している。

[1993.4.25成稿]