易しくない コンピュータ言語学

雑誌：橋梁&都市 PROJECT

2010.1～2010.12　連載記事の原稿

島　田　静　雄

この冊子は、雑誌「橋梁&都市 PROJECT の連載記事の元原稿に使った MS-Word 版から PDF 形式に変換したものです。表紙・目次などの頭の部分と本文とを含め、全 113 ページあります。全体を通して、ハードコピーで見たい方のために用意しました。最初に目次と索引を付けてあります。目次と索引に使った数字は、ページ番号ではなく、「章・節・項」の 3 連番号です。この原稿と殆ど同じ内容をインターネットで閲覧するようにしてあります。パソコンのモニタ画面で閲覧することを考えて、項の単位の文字数を 600 字前後に抑え、通信のアクセス負荷時間も気にならないようにしました。閲覧のときは、小説を読むように最初のページから順に項を閲覧するようにリンクを付けてあります。しかし、目次や索引から見たい個所を探すようなランダムなアクセスができます。この便を考えて、ページ番号ではなく、「章・節・項」番号をリンク情報にしました。インターネットの WEB サイトは、下記です；

http://www.nakanihon.co.jp/gijyutsu/Shimada/shimadatop.html

目　　次

用 語 索 引

0. はじめに

科学の一分野に言語学があります。もともとは文科系の学問とされていました。コンピュータの利用に欠かせないプログラミング言語が研究されるようになって、理科系の人も興味を持つなど、この学問の範囲が広くなってきました。そもそも、学問とは、何かの対象に知的な興味を持つことに始まります。言語学は、例えば「日本語は面白いよ」が始めにあって、幾つかの見方に深入りをしていく過程を系統的に整理して構成する全体を、大雑把に指します。言語学の定義は辞書に書いてありますが、そこに書いてない全体もすべて理解しようとなると単純な学問ではありません。学問的な研究と言うときは、対象とした問題の本質的なものを取り出して、何かの法則を発見するか、仮説として提案する態度を言うようです。余分なものを取り去る操作を捨象と言い、抽象は本質を取り出すことを言います。工学的な見方をすると、純粋化、精製化です。悪口を言うと、科学的態度と言うのは、美味しそうなところのつまみ食いです。

言語学が何の役に立つのか、のような世俗的に問いかけに対して、純粋科学的な方法論は捨象ですので直接の答えが出ません。反対向きが具象です。現実の複雑な問題に正面から取り組む応用科学を実学と言い、それをさらに応用するのが技術です。具体的に言語学と関係する実用技術は、作文と話し方です。例えば、英作文と英会話がそうです。コンピュータ言語学に関係する実用技術は、作文がプログラミング、会話にインタフェースと当てることができます。これらは、他人から教わる部分が多いとはいえ、自分の責任で努力することが必要です。これらの技術を磨くには、元に戻って、言語学的な知識が有ると役に立つ、と答えることができます。

言葉は日常的な道具です。筆者の専門は橋梁工学ですが、その説明に日常的な言葉を道具として使います。筆者が、日本語に関することに手を染めたのは、コンピュータを使った自動製図に、漢字で説明を書き込みたい、とする研究(1974)に始まりました（図 1.1)。その当時は、日本語ワープロなどが無かったこともあって、周りの人の多くは奇異に感じていたようです。筆者は教育現場に籍を置きましたので、知識の伝授だけでなく、文書の書き方や口頭発表の技術も、系統的に教える必要がありました。コンピュータを使うことは、コンピュータを人に見立てて（擬人化して）その人に語りかけ、コンピュータから結果を聞き出す対話と捉えます。このときの言語がプログラミング言語です。ここから、コンピュータ言語学、英語で言えば computational linguistics が用語して表れるようになりました。これから説明するコンピュータ言語学は、普通の言語学の切り分け方とは異なって、脱線的に見える話題を含めた書き方をしています。そこで、形容詞の「易しくない」を付けることにしました。

図 1.1　自動製図用に開発した漢字のベクトルフォント

1. 言語学が関与する環境

1.1 言語学が生まれた経緯

1.1.1 狭い言語環境に居ると言葉数が少なくて済む

日本は、徳川時代、ずっと鎖国で管理されていたことと、島国であることもあって、庶民レベルでは他の言語圏の人との接触がほとんどありませんでした。地域によって方言の違いがあっても、実質的な言語構造は共通です。筆者の経験で言うと、一つの狭い言語環境の中に居れば、言語学的な発想は生まれなくて、必要になるのは実践的な言葉遣いの約束です。最も原始的な言葉遣いは小さな子供に見られ、日本語の環境では単純に名詞を並べます。それも、正確でないことがあります。代名詞の「あれ、それ」だけで済むことも少なくありません。これが舌足らずです。大人は、それを聞いて、「てにをは」や動詞を補って言い換え、子供が言いたいことを確かめます。これが、いみじくも言語教育になっています。大人の世界でも、声に出して丁寧に説明しない場面が起こります。相手は、これを遠慮や慎み深い態度であると好意的に解釈して、言外の意味を察して行動するのが礼に適うと思われています。しかし、現実の話し合いでは、書き言葉で補うなどの手間を省くと、説明が不足し、それを確かめる方法に欠け、思わぬ誤解が起きることがあります。そこで、相手に失礼にならないように、正確に意思を伝える言葉遣いが必要になります。双方の言語環境が異なると、美徳と考え易い言葉の省略は、対話が無いことと同じです。言葉を使わないか、言葉を補うために、喜怒哀楽の感情表現を過度に交える態度は動物的ですので、これを抑えます。特に、怒りの感情はしばしば暴力行動を伴う危険があります。

1.1.2 言語は文化的な侵略の武器になること

日本の外では、ヨーロッパのように、多種類の民族・言語・宗教・習慣の違いを身近に経験します。自分の言語と他の言語との違いによる相互理解に相違が無いように、注意する方法（マナー）が必要です。自分の言語を守った上で他言語を理解する保守的な態度を持たないと、自分の側の言語や文化を失います。海外言語を無防備に取り込んで知識をひけらかす人は、文化的な奴隷です。アメリカナイズと言う言葉は、文化的にアメリカの奴隷化になることを、皮肉な意味で使いました。植民地支配は、支配者側の言語も強制しますので、他方の固有文化の破壊を伴うのです。国力を背景とした自国優位の先入観を持った言語学者は、被支配側の言語を差別する間違いを犯すことがあります。**時枝誠記**(1900-1967)は、時枝文法に名を残す言語学者ですが、日本の朝鮮支配のときに、朝鮮語を抹殺することに加担した経歴がありますので、偏った科学的な態度があったと考えています。あからさまな武力による侵略と支配が無くても、一方の言語が他の言語に影響を与えることがあります。日本語は中国語に影響を受けましたが、中国語が日本語から影響を受けたことは、多く無いと言えるでしょう。英語はフランス語の影響を色濃く受けていますが、フランスでは意図的に英語を使うことを避けるナショナリズムがあります。コンピュータ言語の世界で見ると、マイクロソフト社のオペレーティングシステムが普及した経過は、言語支配が成功した例と言えます。

1.1.3 外来語の輸入は語彙を増やすこと

言語環境が異なる場合、物の名前などは、実物を介すれば、相互に用語を翻訳しても理解の相違が大きくなりません。一対一に用語が対応するときは問題が有りません。しかし、一方に有って、他方に無い場合は、無い側は造語が必要です。ほぼそのまま使うのが借用語です。外来語は言葉の輸入です。元からある語と競合が起こります。どちらも、語彙が増えます。日本では、外来語はすべて名詞として扱います。用語の対応が有っても不十分なことがあります。例えば、日本語では兄と弟と二つの用語を使い分けるのに対して、英語では brother 一つですので、日本語の感覚ではどちらを指しているのかを気にすることが起こります。抽象的な事象を表す言葉は、もっと深刻です。同じ意義の用語が無い場合もあり、また、有っても微妙に理解の違いが起こることがあります。日本古来の言葉は和語と言いますが、抽象的な用語が多くありません。抽象的な用語は、外来語、主として漢語に頼ります。明治以降、ヨーロッパ系の言語が外来語として入ってきましたが、和語に無いものはカタカナ語で使うか、相当する漢語を当てます。熟語を作る造語の機能は、漢字を使うと圧倒的に便利です。そのため、専門用語に、日本で造語された和風漢字熟語が増えました。言葉の輸入があると、同じものに複数の言い方が利用されるようになります。日本語の環境では、漢字の読みに音と訓とがあり、音も漢音と呉音があります。これにカタカナ語が加わるのが現状です。英語も、フランス語とドイツ語の影響を蒙って、複数の言い方が見られます。

1.1.4　言語と民族とは関連を持つこと

言葉は、口伝えを聞いて覚えることが基本です。文字からではありません。赤ちゃんが言葉を覚えるのは、主に母親からの口伝えです。アイヌのユーカラは、母親から娘へと何代にもわたって正確に伝承されてきました。赤ちゃんには先天的な言語習得能力（遺伝形質）があります。しかし、言葉を覚えることは獲得形質の方ですので、個人差があります。乳幼児期の言語環境で、その人の言語がほぼ固定されます。したがって、同じ言語の家族の集合は、大なり小なり血縁関係が濃く、その集合が民族です。多くの民族が分布し、それぞれに固有の言語を持つのは、歴史的に民族間の交流が狭かった時代の名残です。一つの民族、一つの言語圏であっても、地域で方言差があり、地域間の距離が大きいと方言差も大きくなります。この差は、発声の違いが主です。日本では東北と九州との方言差が大きいのです。多くの方言が集まる場所では、相互の影響を受けて変質し、一種の標準化が起こります。標準語は東京言葉が元になったものです。不思議なことに、東北の北に在る札幌言葉も標準語です。しかし、微妙な差があります。その一つは、東京弁は助詞の「が」を鼻濁音で発音することです。ラジオやテレビの普及は、良くも悪くも官製の標準語を普及させましたので、一面では方言文化の衰退を招きます。

1.1.5　声を文字で表すことは難しいこと

もともと、言葉（ことば）は、声に出して相手に伝え、耳で聞いて理解します。文字は、音を図形化して記録する目的で使います。音を記録する文字を**表音文字**と言い、英語では letter と使います。英語の character も日本語では文字と訳し、区別が分かりません。こちらは、英字(letter)・数字(digit)・記号(symbol)の集合をまとめて言う言葉です。コンピュータが扱う文字集合は character set と言います。漢字は、音と意味とを持つ**表意文字**です。英語で言うときは Kanji character と言い、カナは表音文字ですので Kana letter と使い分けます。世界を見渡すと、文字を持たない言語が未だあるようです。日本語も、古い時代に漢字を輸入して使うようになりました。日常会話では、文字は必須の道具ではありません。文字を見るときは、声に出すか、頭の中で音に直して理解しています。そのため、文字の読み方に約束を決めておかないと、正しく言葉が伝わりません。日本語の場合、漢字を当てるとき、音の方に重点を置くか、意味の方に重点を置くかの区別があります。前者が訓読み、後者は音読みです。例えば、**言語**（げんご）は字音語の言い方、**言葉**（ことば）は訓読みで使いますが、実は同じことを表しています。これが、日本語を複雑にしている一つの原因です。訓読みは、聞いて紛れることが少ないのですが、音読み熟語は同音異義語が多くなりますので、眼で見て意味を確認しながら理解します。話し言葉で音読みの熟語を言うとき、しばしば訓読みの言い換えで説明することが行われます。

1.1.6　音声を直接扱う研究は難しいこと

日本語のワードプロセッサ（ワープロ）でカナ漢字変換が利用されるようになって気が付いたことは、言葉は音として頭の中で発想されていることです。カナ漢字変換の判断ミスによる誤字・当て字が増えたのです。人口衛生（人工衛星）と変換しても、本人は気が付かないことが起こります。逆に、文字並びを見るとき、頭の中で声に直して理解しています。そのため、日本語の言い方を知らないと、日本語ワープロを使うことができません。英語圏の人が英文を日本語に直したいとき、英語で使える英和辞書で日本語を探し、その発音を日本語ワープロで漢字に変換しても、それが正しい日本語変換になる保証がありません。これは、他の言語ごとにある専用ワープロの使い方すべてに当てはまります。言語学は、文字に書いた文書を媒体として間接的に研究することで妥協しなければなりませんので、音（おと）の扱いに抜けが起こります。それでも良いのが、一例としてのコンピュータ言語学です。

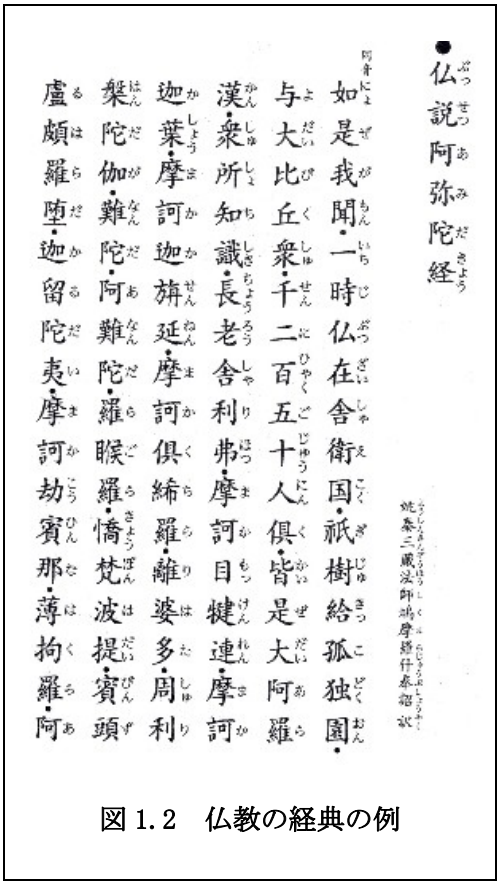
1.1.7　複数の言語環境の場では公平な言語学が必要になる

二つの言語環境に別々に属する人が接するとき、両方の言語に達者な人が居ないと、正確な相互理解に欠けが起こります。このとき、言葉の分類や分析が必要です。この作業を含めた全体が、言語学です。言語学者の**ソシュール**(Ferdinand de Saussure, 1857-1913)が多言語環境のスイス人であるのは、公平な立場で複数言語を研究する環境に居たことの関係があると思います。言語の翻訳作業では、紛れのない規則を決めておくと、正用誤用がはっきりします。これが言葉の**定義**です。そのための道具の一つは辞書です。辞書というと、和英・英和辞書のように、例えば、「犬-dog」と対応させる単純な対訳を想像します。しかし、これでは不正確なことが起こります。犬とはどういうものかを説明するのが定義です。定義を説明する形式の辞書には**事典**と当てます。普通の辞書は**辞典**を当てます。音が同じですので、話ことばの中では、**ことてん**と**ことばてん**の言い換えをすることがあります。専門用語の説明辞書は事典の性格を持ちますので、英語では dictionary ではなく glossary の方を使います。

1.2　道具としての文字

1.2.1　言葉を音のまま研究対象にすることは難しい

言葉は音ですので、その場限りです。音としての言葉を研究するには、音を記録し、再現する道具が必要です。テープレコーダが手軽に利用できるようになったのは戦後ですが、それを言語学の研究に直接利用したいとしても、研究の焦点を絞り込むことが難しい面があります。したがって、間接的ながら、文字に表した言葉を研究することに頼ります。そうしておいて、文字で記録した言葉を読むことで、擬似的に音声を再現します。しかし、元の音声の特徴を正確に再現できません。日本の仏教の経典は、元の梵語（ぼんご）を漢字で表記したものを中国経由で輸入したものですが、漢字に補助的な振り仮名を付けて読んでいます（図 1.2)。元の発音と意味とは全く分かりません。そもそも、どのように読むのかも分からない、古代エジプトで使われた化石言語もあります。他の言語で書かれた文献を解読して意味を理解するだけならば、音に拘る必要がありません。古い時代、日本が中国文化を輸入したとき、また、明治以降、西洋文明を輸入したとき、文字は、意味の理解を優先し、日本風に読みました。中国語の語順は日本語と違いますので、漢文の学習は、返り点を付けて日本語風に読み替えます。考えてみれば、不思議な理解方法です。これは、その言語を話す人との交流が無いためです。英文和訳も、返り点をつけませんが、眼で文字並びの前後を判断しながら語順を替える言い方に変えます。戦後、アメリカ軍の駐留で英語（アメリカ英語）を話す機会が多くなって、それまでの英語（イギリス英語）の学習方法に大きな影響を与えることになりました。


●仏説阿弥陀経

如是我聞・一時仏在舎衛国・祇樹給孤独園・
与大比丘衆・千二百五十人倶・皆是大阿羅
漢・衆所知識・長老舎利弗・摩訶目犍連・摩訶
迦葉・摩訶迦旃延・摩訶倶絺羅・離婆多・周利
槃陀伽・難陀・阿難陀・羅睺羅・憍梵波・提賓頭
盧頗羅堕・迦留陀夷・摩訶劫賓那・薄拘羅・阿

姚秦三藏法師鳩摩羅什奉詔訳

図 1.2　仏教の経典の例

1.2.2　文字の並べ方にも言語による違いがあること

文字は、発声順に並べて書きます。その並べ方は、図形的に横書きと縦書きとがあります。日本語は縦書きを主に使ってきました。縦の文字列を右上から並べます。漢字は文字単位が図形的に四角形ですので、右上から始める縦書きの規則を使って一列に一文字を当てると、一行に書く文字並びが右から左に並びます。戦前の書き物・看板・石碑の銘などに見られます(図 1.3)。欧米語は横書きで、左から右へ文字を並べます。しかし、アラビア語は、右から左に文字を並べる横書きです。ただし、数字は例外的に左から右に書きます。

図 1.3　戦前の絵葉書（部分)、納屋橋の表題は右から左に書いてあります

1.2.3　漢文訓読という文体が表れたこと

文字は、その言語の発声順に並べるのが自然です。しかし他言語の中身（意味）を文字だけから理解したいとき、語順を変えた翻訳文字並びにすることがあります。日本語の環境で習う漢文は、元が中国語です。漢字は日本語の中に取り込まれていますので、熟語単位は、ほぼそのまま理解できます。中国語の翻訳、つまり意味を日本語の音声で理解するように変換するときは、単語の語順を変える返り点をつけ（図 1.4）、「てにをは」を補い、動詞などの活用語尾を加え、必要に応じて振り仮名を付ける程度の最小限の作業で済ますことをします。このように変換した文書の読み方が**漢文訓読法**であって、文語体の一種です。**読み下し**とも言います(図 1.5)。考えて見ると、パズルを解くような変わった読み方です。

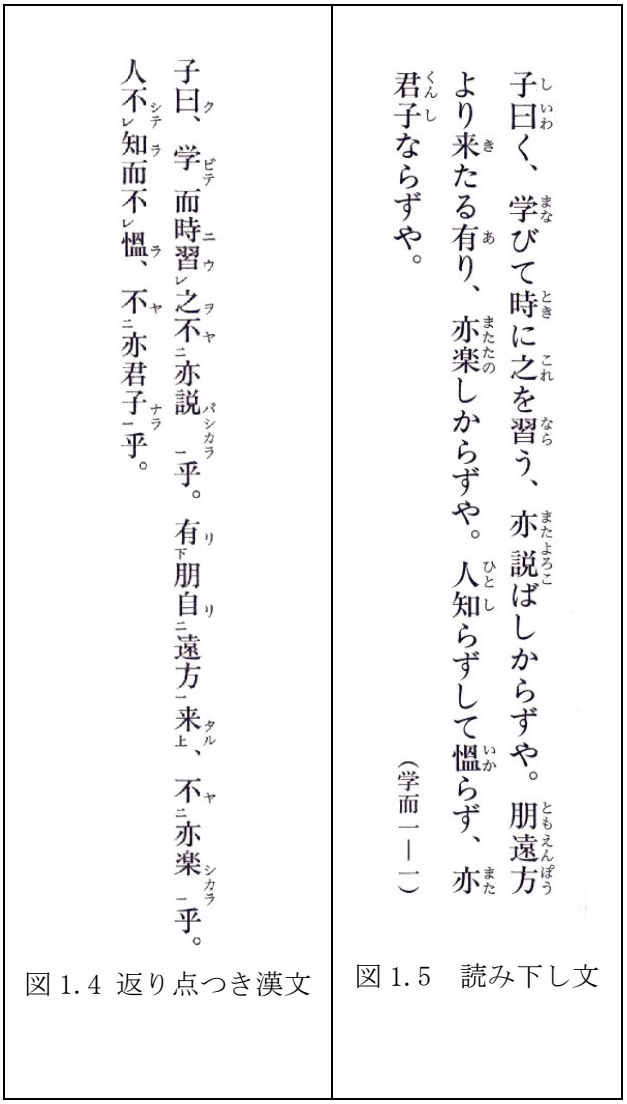


図 1.4 返り点つき漢文　　図 1.5　読み下し文

1.2.4　英訳には翻訳調の文体があること

英語は、文字種と語順との二つが日本語とは異なりますので、翻訳は二段階で行われます。一つは単語単位を日本語に直すこと、二つに語順を変え「てにをは」と「活用語尾」を補います。英単語に相当する日本語が無いとき、読みをカタカナ語にして済ますことが増えました。元の英語を知っているときは、何とか意味を理解できますが、そうでないときは困りものです。語順を変える言い方は、**翻訳調**と言う独特の言い回しを生みました。特に、英語の関係代名詞を訳すとき、語順を変える個所で表れる「…するところの…」「…するものとする」などの言い回しがそうです。英語を日本語流に習った年代の人には抵抗がないのでしょうが、庶民レベルの言葉遣いとはズレています。

1.2.5　文字は形を持った道具と考える

声に出す言葉は、実体がありません。眼で見る形にする媒体が文字です。文字は、道具や部品の性格があります。言葉を文字の並びに直すと、言葉の研究を具体的に物を扱うように進められます。結果として、言語の研究のほとんどは、文字並びの解析で済ませます。話し言葉は、文法規則を保とうとする保守的な性質があると同時に、時代と共に、変質も受けます。「昔はこういう言い方をした」と言うのがそうです。新しい言葉は、若者の間で創造されることが多く、流行語として広まることもあります。しかし、言葉の骨格、つまり文法構造は、案外なことに丈夫です。言葉を物理的に捉える分野を、ソシュールに始まる**構造言語学**と言うのがそうです。文字を道具と見るとき、全く新しい言語体系を創作する場合も出てきます。**エスペラント**がそうです。数学式も言語です。プログラミング言語も、そのような人工言語です。この人工言語は書き言葉として使いますが、これを見るとき、頭の中で声に直して読んでいることに気が付いているでしょうか？　これらの言語を構造的に見ると、英語の文法構造を色濃く持っていますので、英語の**方言**(dialect)の性格があります。言語構造のモデル化の一つは、主語(Subject)・述語(Verb)・目的語(Object)の語順です。日本語はＳＯＶの順、英語と中国語はＳＶＯの順です。構造違いの言語を習うとき、苦労の種（たね）となるのです。

1.2.6　部品に分ける単位が品詞であること

音の並びを文字に直して並べるとき、その最小要素と組み合わせについての議論が出てくるのは半ば必然です。しかし、音を文字並びに直すこと自体に幾つかの問題があります。アルファベットやカナは表音文字ですので、一般的に読み方を表すときにはこれらを使います。しかし、正確な表現法にはならないことが多いので、発音記号を別に使い、声の出し方を口の開け方や舌の位置などを図解で示すことがあります。日本語ではカナを読みの補助に使うのですが、「ゐ・ゑ・くぁ」を「い・え・か」と単純表記するようになって、声に出す読み方も簡単になってきました。江戸は yedo、円は yen と発音していました。日本語では、アルファベットのＲとＬを区別しませんが、巻き舌の言い方（べらんめえ）はＲの音だと言います。英語の表記は、意味のある音の並びの最小単位を単語とした分ち書きをしますので、個別の単語の性格を品詞(part of speech)に分ける考え方も自然です。品詞の考え方は、欧米言語に接するようになって日本語の辞書にも採用されるようになりました。しかし、品詞分類の歴史は比較的新しいものです。漢字を輸入して使った日本語の文字表記法は、中国語と同じように、文字並びを分かち書きしませんので、品詞の区切りを間違えることが起こります。「かねおくれたのむ」「べんけいがなぎなたをもつ」などがよく例に出されます。言語学的な分け方は、**形態素**(morpheme)の方を使います。単語の境界判別を行い、意味を持つ品詞の並びに分解することを形態素解析と言います。これをコンピュータにやらせようとする研究は、**長尾真**(1936-)に始まりました。

1.2.7　実用文書の書き方に応用する研究が必要

コンピュータを使い、学問として取り組む形態素解析は、文字並びをデータとします。形態素解析の応用目的の一つに、日本語から英語への機械翻訳があります。日常的に使われる言語（これを自然言語と言います）を研究対象にすると、或る程度の法則性を発見できるのですが、例外もまた多く出るため、解析の成功率が下がり、実用的な機械翻訳ができません。これを逆に考えます。形態素解析が成功するような文書の書き方は、どうあるべきかを提案する研究です。文学的に書かれた文章は、感情表現やレトリックがありますので、読み手はそれを楽しみます。機械翻訳は、このような文章データが苦手です。学術論文を海外に発表するときの道具として、機械翻訳が実用になると助かります。学術論文は、曖昧な表現を避け、論理的な整合性を持たせるように作文しますので、機械翻訳の対象とすることができます。機械翻訳が成功しない個所は、コンピュータが論理的に判断できない個所です。したがって、機械翻訳は、元の文章に何かの欠陥があることを指摘する校正ツールになります。一般の人を対象とした作文の実用書は、書店で普通に見られます。作文添削は、眼で見て人が判断しなければなりませんが、自分の書いた文書を自分で添削するのは限界があります。欧米の大学では、technical writing がほぼ必修の教養科目に位置づけられていて、学術論文などの書き方について、物理的な約束と文書の書き方の基本を教えています。効果的な添削は、別の人がするのがよいので、対面授業は意義があります。学術誌の投稿規程は、ごく基本的なことしか書いてありませんので、より実践的な実用文書作成を自分で心がけなければなりません。参考として下記のＵＲＬを参照して下さい。

「実用文書のまとめ方」http://www.nakanihon.co.jp/gijyutsu/Shimada/bunsyo/DocumentIndex.htm

1.2.8　言葉は危険な武器にもなること

文字は、為政者側が権力を保つときの手段として利用してきた歴史があります。一般大衆は、文字を正しく読み書きできない文盲であっても、日常生活で特に困ることはありませんでした。特権階級は、言葉の知識で武装することで優位を保つこともしました。日本ではこれに外来語、特に漢語を使いました。言葉は文化の一翼を担いますので、文字の読み書きができる人は学があるとされ、さらに、お習字が上手であると、力関係だけでない尊敬を受けます。昔の武士は、刀のような武器を持つことを許された階級ですが、漢学と礼節を素養として弁えることをしました。文武両道を極めると言うのがそうです。この論理を裏から見ると、文字、つまり、言葉も武器になることが見えてきます。攻撃に代えた口撃は、的を射た言葉の言い換えです。面と向かって口喧嘩をするのは直接的です。国際間では、メディアを介して、間接的でも激しい遣り取りが行われています。ジャーナリストは、「ペンは剣よりも強し」と言う言葉を好み、権力におもねない態度を矜持（きょうじ）と自己弁護します。しかし、裏を返せば、ペンと言う武器を持った特権階級ですので、礼節の無いヤクザ的な危険人物も存在します。メールを介した誹謗中傷が社会問題化していますが、言葉が危険な武器として使われていることの実証です。

1.3　標準化と多様化

1.3.1　差別用語の扱いが難しい

一つの大きな言語環境であっても、狭い範囲での方言違いは普通に見られます。人間集団には、その集団だけで通用する言葉や符丁も生まれ、これが他の集団との差別にも使われます。職業別の専門用語がそうです。仲間内だけで通用する用語や言い回しは、やや暗い影をもつことから**隠語**と言います。人間社会は、動物的に他者を攻撃する悲しい性質がありますので、言葉違いを身内と他者、味方と敵とを区別する手段の一つに使います。スパイ・間諜・隠密・密偵などは、相手側の集団の中に入って情報収集をする人を言います。優秀なスパイである要件の一つは、相手側の言葉に堪能であることです。これらの行為を言語操作と見ると、意図的な言葉の標準化です。それも、狭い言語環境の範囲への標準化です。差別用語は、それを使うことが相手に対して敵意や軽蔑を投げかけることになる言葉と解釈します。しかし、相手を不利にする使い方ではない差別化もあります。相手に対して礼に欠ける言葉がそうです。それらを使うことを制限することも、標準化の一面です。狭い方への標準化は、保守的な性格がありますので、好意的に見れば、その言語環境の文化を保存する上で意義があります。

1.3.2　官製の標準化があること

言葉の標準化は、社会の秩序と平和を保つ手段として必要です。言葉は、多様化する性質があります。多種類の言葉があると、孤立する集団ができ、相互の意思疎通の障害になりますので、共通の理解を図る方法が必要です。言葉の標準化は、互いに妥協することで自然発生的に成立する場合の他に、権力者側が意図的に制御する場合もあります。日本語の環境で言えば、**常用漢字**の制定、**正書法**の提案がそうです。ラジオやテレビで話される言葉は、全国で共通に理解できることを目的にした**標準語**を使います。これは良い面があると同時に、官主導の立場を色濃く持ちます。官主導の標準化は、それを提案するときと利用するときに幾らかの優越感と正義感とを持ちます。その正義感に染まると、自由で文化的な背景を抑える負の側に作用し、それが行き過ぎると、はた迷惑で、お節介になることがあります。ＮＨＫ、朝日新聞社、岩波書店などを嫌う文化人が少なくないのですが、嫌う理由の一つは、公平さを言う割りには、官を背景にするような権力的な臭いを感じるからと思います。

1.3.3　人工言語は標準化が意識される

普通の会話に使っている言語を自然言語と言うのに対して、人工言語は、便利さを目的として工夫される性格を持ちます。自然言語は多様性が有り過ぎますので、人工言語は、なるべく多様性を排除し、言葉数を抑える標準化をします。日本人の多くは、数学式を特別な記号表現と見ますが、そもそもの出発は、話し言語を記号に変えた書き言葉の一種です。そのため、数式を書くことに代えて、文章で表現する方法も実用されていて、数式を見るとき、声に直して言うこともします。数学者同士は、数式を言語として論理を展開するので、一般の人は非常に特殊な集団と見ます。しかし、啓蒙的、教育的には、なるべく文章で説明するのが勝ります。数式記号をコンピュータに教えるコンピュータ言語の代表がFORTRAN です。事務処理言語の COBOL は、数式を文章で記述する方法を使います。数式処理のソフトウエアもその範疇です。幾何学は、歴史的には、言葉での説明が主です。幾何学に代数学的な方法を使うようになったのは、幾何学の長い歴史から見れば、ほんの最近です。図形も書き言葉で表そうというのがグラフィックス言語です。欧米風の科学技術の取り組み方が、言葉による表現に拘ることの感覚は、日本人の理解方法とは違います。日本のマンガは、今や世界的に有名になりましたが、文字を順に並べるような規則が緩やかな二次元的な表現を使います。漢文を読むとき、返り点などを助けにするとしても、視点を前後させて理解する方法は普通ではありません。文字ではなく、図を媒介とする情報伝達の道具は、ピクトグラム(pictogram)があります。日本では、東京オリンピックのとき、世界から集まる多くの言語のお客さんに、言葉を超えて正確な情報を伝える目的に開発されたのが始まりです。こちらは文字扱いではありませんので、この報文では扱いません。筆者の専門だけでなく、コンピュータ言語を使う場面が増えています。この人工言語を覚えるには、少し遠回りに見えますが、我々が使っている日本語の性格を言語学的に見ることが役に立ちます。次章から、この課題を取り上げます。

2.　日本語文書の構造

2.1　読みの仮名表記

2.1.1　音節の単位と文字単位が異なること

日本語は、音声を文字で表す最小単位が仮名です。音を表すので表音文字ですが、アルファベットに有るような、子音単独を区別する文字がありません。文字教育の最初は、仮名を教えます。明治以前は、「いろは 48 文字」、義務教育では「アイウエオ 50 音文字」になりました。この数には濁音と半濁音を含みません。仮名は片仮名と平仮名の二種の字体があって、使い分けができる便利さがあります。戦後の義務教育は、平仮名を先に教えるようになりました。英語の単語とは異なり、日本語の文字並び、特に仮名文字を隙間無く並べると、品詞の区切りが分かり難くなりますので、分かち書きも工夫します。意味を持つ最小単語並びに分ける一つの方法が形態素解析です。音声としての最小単位を、**音節**(syllable)と言い、一つの母音と複数の子音で構成されます。息継ぎや間も、音のない音節です。片仮名は、日本語の一音節単位を一文字で書き表しますが、子音構成が一つです。濁音（が・ざ・だ、など）は濁点を付け、半濁点（ぱ、など）は丸を付けた別の文字単位です。日本の和歌を仮名文字で書くときは、濁点文字を使いません。読むときに濁ります。和語は、半濁音（パ行）がないのですが、平安時代はパ行の発音があったそうです。銀行のＡＴＭコンピュータの画面で固有名詞を入力するときは、濁点「゛」と半濁点「゜」が別単位の文字扱いですし、姓と名との間に空白文字も使います。「キャ・キュ・キョ」の拗音、「ッ」を添える促音は、小文字を当てます。そして「ン」は撥音を表し、別格扱いです。電報文はこのように書きました。

2.1.2　漢字は一文字一音節であること

漢字は、中国語の環境では一文字一音節一意です。日本に輸入されたとき、中国語の発声を仮名で近似的に表す方法を工夫しました。しかし、仮名二文字以上を使わないと漢字一字の音節を表すことに無理があります。漢字の音表現を仮名に置き換えると、音読み・訓読み共に仮名文字数が増え、音節の数え方が変わります。口語体の文章は、耳で聴いて分かる言葉並びにしますので、眼で見ながら読む文書に直すと字数が多くなります。意味を理解するとき、仮名の並びから漢字相当の音節単位を区切り、さらに品詞に分別することが、日本語形態素解析の考え方です。漢字が一字で一音節を表すことから、元の中国語の漢字の読み方は、音楽的なリズムを持ちます。二字熟語や四字熟語は、二拍と四拍のリズムに乗ることと、一字よりも意味を限定します。日本語として使うとき、同数の片仮名では足りません。漢字一字に片仮名二字を当てると収まりがよいので、二字の漢字熟語の音読みは仮名四文字を当てると馴染みます。欧米のアルファベットで表した語の読みをカタカタ表記にすると、文字数、つまり音節単位が増えます。個別の子音にも仮名を当てるからです。これを省略語にする造語法は、仮名４文字を使うとリズムがよく、「マイコン・パソコン」などのように利用されます。耳で聴いた音を仮名文字で表すと、元のスペル（綴り）が全く分からないことも起こります。日本語の音をローマ字表記にする方法にヘボン式があります。ヘボンが女優のオードリーヘップバーン(Hepburn: 1929-1993)と同じ綴りであるとは、直ぐには想像できないでしょう。

2.1.3　文字選択が多いので厄介であること

日本語（和語）の読みに漢字を当てると、意味を限定する利点があります。例えば動詞の「とる」に当てる漢字は、ワープロの仮名漢字変換を通すと、何種類かの候補がでてきます。眼で見る文書にするときは、適切な漢字を使うと意味を補う利点があります。中国語の熟語を、訓読するときの漢字を当てるのを目安にします。「写真を撮る」と言うときは、撮影が意識された選択です。動詞の意味を限定する方法は、英語の動詞にもあります。他の語と組み合わせて使うことで意味が限定されますので、この組み合わせ単位を"phrasal verb"（**句動詞**）と言います。英語の翻訳のとき、単語単位で意味を当てる方法を**逐語訳**または**直訳**と言います。そうすると、複数の単語の集合で意味を表す慣用句は、意味不明の訳になることがあります。この点では、中国語の漢字熟語の作り方は、日本人にとっては合理的です。英語の慣用句は、構成単語単位が性別や時制で変化することに加え、並び順も変えることがあります。句動詞に使う前置詞は、位置が変わることがありますので、英文和訳はパズルを解読するような技術が必要になります。単語並びの順番は、言語構造では非常に重要な要素です。英語や中国語は、単語の位置で品詞と意味が限定されるのに対して、日本語では、助詞の「てにをは」を付けることで、語の並びを変える自由度が大きいのが特徴です。

2.2　文体と口調

2.2.1　書き言葉も声に直して読むこと

文字に書いてある言葉を読むとき、頭の中で声に直しています。言葉を文章で表すときの様式（スタイル）が**文体**です。これをそのまま声に出して言うとき、そのスタイルの**口調**と言います。文語体・文語調などと使い分けます。声に出すときは、アクセント(強勢)とイントネーション（抑揚）も重要な要素です。漢字は、中国語を発声する環境では**四声**の区別がありますが、日本語の書き言葉にすると、この区別ができません。また、例えば kwa の音を「くぁ」と発声するのを、「か」で言うような単純化もあって、同音異義語が増えました。日本語の環境で一方的に話すときの言い方には、漢文調、演説調などもあります。対等な立場での会話の言葉遣い（話し言葉）とは異質です。明治時代の作家、二葉亭四迷が始めたとする**言文一致**のスタイルが**口語体**です。正確に言いたいときは、**文章口語体**と言います。作家のペンネームも「くたばってしまえ」のもじりだそうです。聞いて判り易い言葉遣いの技術を話術と言います。言文一致のスタイルの研究には、当時の落語の口演筆記を参考にしたとされています。文字表記は約束ごとですので、必ずしも文字通りに発声するのではありません。例えば、主語を表す「わ」の発音は、書き言葉では接尾辞に「は」と書く決まりです。読み方の変化は音便（おんびん）として表れ、それが書き方に影響を与えています。逆に、ワープロで熟語の仮名漢字変換をするとき、例えば、学校は「がくこう」ではなく、「がっこう」でないと出てこない規則（アルゴリズム）になっています。

2.2.2　格式を持たせる言い方があること

日本語の話し言葉は各地で方言差もありますので、同じ言い方でも違う意味に取られることがあります。例えば、東京弁は、「ひ」を「し」と発音しますので、「おしさま・しのまる」になります。したがって、言文一致の文章言葉も、くだけ過ぎた表現にならないような、適度な約束が必要です。文章口語体の特徴は、文末の表現法での違いで二つあります。「である調」と「です・ます調」です。前者は、文語の影響を残し、やや格式ばった表し方であって、話し言葉で使うよりも、学術論文のような書き言葉に見られます。後者は、「である調」を丁寧な会話の物言いにしたスタイルであって、耳で聞いて理解できるように、意識的に訓読みを使うようにします。字音語は、漢字を眼で確認しないと誤解されることがありますので、口語体での文章は、字音語の熟語に訓読みの説明を補うこともします。そうすると、欠点として字数が多くなります。丁寧な言い方は、くどくなり易いので、簡潔で要点を押さえた書き方の工夫が必要です。簡潔が過ぎると省略に繋がります。書く本人は内容を理解していますが、読み手は必ずしも同じレベルにはいませんので、第三者に見てもらうことが必要になります。これが校閲です。文学作品は、文章に表れない部分を読者が感性で補う楽しみ方もします。文学作品の評論や解説は、行き過ぎると、読者にとってお節介になります。人間社会では、上に立つ人の言葉数が少ない曖昧な態度に対して、下のものが気を回すのも見られます。第三者を間に入れるか、書面にして、具体的に話を進めます。筆者は、これを公家風（くげふう）と言うことにしています。しかし、現代は説明責任が問われる時代ですので、誤解が起きないように直接話しを交わすルールが必要になっています。

2.2.3　話し言葉で文書も書くこと

筆者の文章を読んで下さる方は、ほとんど、当然ながら日本語の環境に居ると考えています。上で説明したように、文書を黙読しているときであっても、頭の中では音に直して理解します。読み方が分からない漢字や文字並びの個所で、読みが中断し、どう読むのかを考えることが起きます。筆者の文体は、話し言葉で文章化するようにしてあります。句読点は、声に出すときの自然な息継ぎを意識して、比較的短い文で区切ってあります。漢字は音読みと訓読みが混じりますので、読みを助けるために、括弧で振り仮名を付けることもしてあります。自分で書いた文章を何度か読み返して、音の繋がりが不自然に感じる個所を修正しています。読みを助けるときの重要なキーが、句読点だけでなく、漢字の使い方、送り仮名の付け方にもあることが、経験的に分かってきました。常用漢字の範囲で使う熟語は、素直に読めます。しかし、人名や地名などの固有名詞には、常用漢字外も表れますし、また特別な読みもありますので、そこで読みの中断が起こります。カタカナ用語は、元の言語の意味とスペルを知っていないと、突っかかります。専門用語で、当用漢字にない漢字をひらがなに置き換える場合も抵抗を感じます。単語の切り分けができ難い、長いひらがな綴りも読みのためらいが起こります。第三者に見てもらう、その第三者をコンピュータとし、コンピュータに作文添削と知識を教えるとことが、筆者の研究目的の一つです。

2.3　語順と向き

2.3.1　文字並びを時系列の記録として理解する

声に出す言葉は、物理的には音ですので、時系列(time series)データです。音は一過性の事象であって、その場限りで何も残りません。言葉を聞いているとき、幾つかのまとまりのある音の単位を順に頭の中で整理していって、或る区切りでその全体を総合的に理解します。頭の中がどのように働いているかは分かりません。短い時間間隔のデータを記録しておく短期記憶の機能を使うようです。論理の流れが素直でないと、記憶順と取り出し順が狂い、理解に混乱が起こります。音声を文字並びで記録したものが文書です。文字並びにまとまりの有る最小単位が、語（単語、word）です。単語並びでまとまりの有る意味単位を構成するのが句(phrase)です。句を構成する語順は言語ごとに固有の決まりがあります。そして、句の並び順も言語ごとに決まりがあります。これを、日本語ではＳＯＶ（主語・目的語・述語)、英語と中国語はＳＶＯの順であると大枠で区別します。説明を補うと、主語・目的語・述語と言うと、それぞれが単語一つと考え易いのですが、実際は複数の語の集合で構成しますので、主語部・述語部のような大枠で考えます。日本人が英語を耳で聞いて理解することが負担になる理由は、音の種類が異なることを聞き分けることと、語順の入れ替えが即座に判断できないからです。文書にして理解することが好まれるのは、時間の束縛がないので、この負担を軽くできるからです。

2.3.2　先回りで言葉を予測していること

耳で聴いている言葉は、時間の流れで理解していく過程があります。単語が発声された瞬間に、何が続くかの幾つかの予測が働き、次に出てくる言葉で予測を修正し、文が一区切りしたところで文全体の理解が完結します。予測ができるのは、語順に規則があるからです。日本語で使う漢字の熟語は、漢字２語以上で構成された最小の句単位です。慣用的な言い方、慣用句は、全部を聴き終わらない内に、最後までの言葉が予測できて、早めに判断ができます。日本語の語順は、動詞が最後に来ますし、肯定・否定も文末になりますので、文が完結するまで待たないと、全体の意味の取り違えが起きます。文章単位が短い方が良いと言う提案の理由は、ここにあります。下手な作文や演説は、最初の言い出しから脱線的な別の話題に入るなど、文の切れ目が論理的になっていません。年配者の発言などは、いつ終わるか、何を言いたいのかが分からないので、回りが迷惑することが起こります。英語は、動詞が早いうちに表れ、肯定・否定も最初に分かります。相手に何かを依頼するのが命令文です。英語は文頭に動詞を使いますので、聴く側は、それなりに早めの予測が働きます。日本語では、動詞が最後に現れますので、文末にくるまで、命令文であることも分かりません。

2.3.3　文書は眼で見て先読みしていること

声に出して話す場合、話す側も聴く側も、文字数相当の時間を取られ、端折ることができません。話すときの速さは、標準として１分間 300 字程度です。現代はせせこましい時代になりましたので、どこの言語でも、昔に較べて話し方の速度が上がっています。文字に書いた文書を黙読すると、速度がかなり上がります。文書によるコミュニケーション（情報伝達）は、会話時間を節約できることと、時間に束縛されないことの二つの理由で、現代の多忙な環境では重要な手段です。電話・面会・会議に代表されるような口頭でのコミュニケーションが、必要以上に時間を取られ、非能率になることがあるのは、多くの人が経験しています。文書の構成が定型的、または慣用的であると、要点の拾い読みができます。その手掛かりを与えるのが、表題(title)や見出し(caption)です。この他にも種々の技法が提案されていて、この全体が実用文書の書き方です。小説などの文芸作品を読むときは、相手を意識することなく、自分の時間ペースで楽しむことが、相手のあるコミュニケーションと違うところです。詩歌などは、何度も読み返し、朗読や歌も発声して楽しみます。探偵小説は、手掛かりを伏線として先に挙げておいて、謎解きまでの過程を楽しみにします。何度も読み返して楽しむこともします。答え、つまり、話の筋が分かっていても、また分かっているので、話の展開を楽しみます。先読みの効かない文書は、聴くことも読むことも疲れます。マンガは、文字を省く二次元的な図で情報を伝達しますので、文字に較べて伝えたい内容の質も量も変わります。しかし、感覚的に理解する速度は上がります。かなり順番を無視した見方もできます。しかし、コマの並べ方は話しの時系列に載せます。欧米のマンガは左上から始まり右に続きます。日本では、絵巻物の歴史もあって、右から縦書きの習慣があります。この違いは、海外では面白がられています。日本のマンガ本を欧米で複製出版するとき、左右を裏返しにすることがあります。

2.3.4　言語理解は複数の発声の同時進行がないこと

聖徳太子は、七人の人の請願を同時に聞き分けたと伝えられています。しかし、普通の対話の環境では交互に話しを切り替えます。時系列として見るとき、言語の発声は、複数の事象の同時進行はありません。同時通訳がありますし、テレビでは二ヶ国語放送もあります。両方を同時に聴き分けることができるのは特殊な能力と言えるでしょう。ラジオやテレビでは、音声のバックグラウンドに音楽を流すこともしますが、時系列の種類が違います。これに対して、音楽では並列同時進行があります。日本で多くの人が合唱をするとき、すべて同じ歌詞で、音程も旋律も同じです。伴奏があるときも発声を誘導するように同じ音程と旋律を奏でます。音程を代えた和音（ハーモニー）にすることや、伴奏が別の旋律を奏でるような歌い方は地についていません。

2.3.5　音楽の文書化が楽譜であること

音楽の歌詞と旋律は、多くの人が楽しみます。これも時間的な音の流れです。旋律を紙に書く方法が工夫されています。西洋音楽の楽譜の書き方は、輸入技法です。音の高さを区別する言い方は「ドレミ～」ですが、英字記号に、Ａ～Ｇを当てます。これに、日本語では「イロハニホヘト」を当てました。ハモニカ、大正琴の楽譜は、イロハで書いてあります。楽譜が無くても、歌を楽しむことはしますが、普通の日本人は絶対音感が鋭くありません。今はカラオケ全盛ですが、以前、流しの音楽家は、客に合わせて伴奏しました。歌う人のテンポに合わせますし、音痴の客には音の高さに合わせて、元の歌の調を変えて伴奏する特殊技能を持っていました。楽譜を眼で追うときは、頭の中で音を再現しています。楽譜と文書とは時系列としての音を紙などに記録する手段ですが、大きな違いが一つあります。楽譜は、複数の時系列の同時進行を記録します。大編成のオーケストラや小編成の合奏集団用の総譜は、音楽専門家が使う文書です。アマチュアが音楽を楽しむときにも見たいので、小型本にした出版物として売られています。音楽レコードを耳で聴いて楽しむときは、音の発現順をたどり、それなりの長さの時間を取られます。楽譜を見る楽しみ方は、拾い読みや先読みができることが、文字文書の読み方と同じ方法です。例として、図 2.1 を示します。筆者の趣味が入るのをご容赦頂きますが、モーツァルトのオペラ、ドンジョバンニから、ツェルリーナのアリアの始め部分です。オーケストラの指揮者は、歌手の声も含め、複数の楽器演奏の同時進行を聞き分ける技能に恵まれていないと勤まりません。

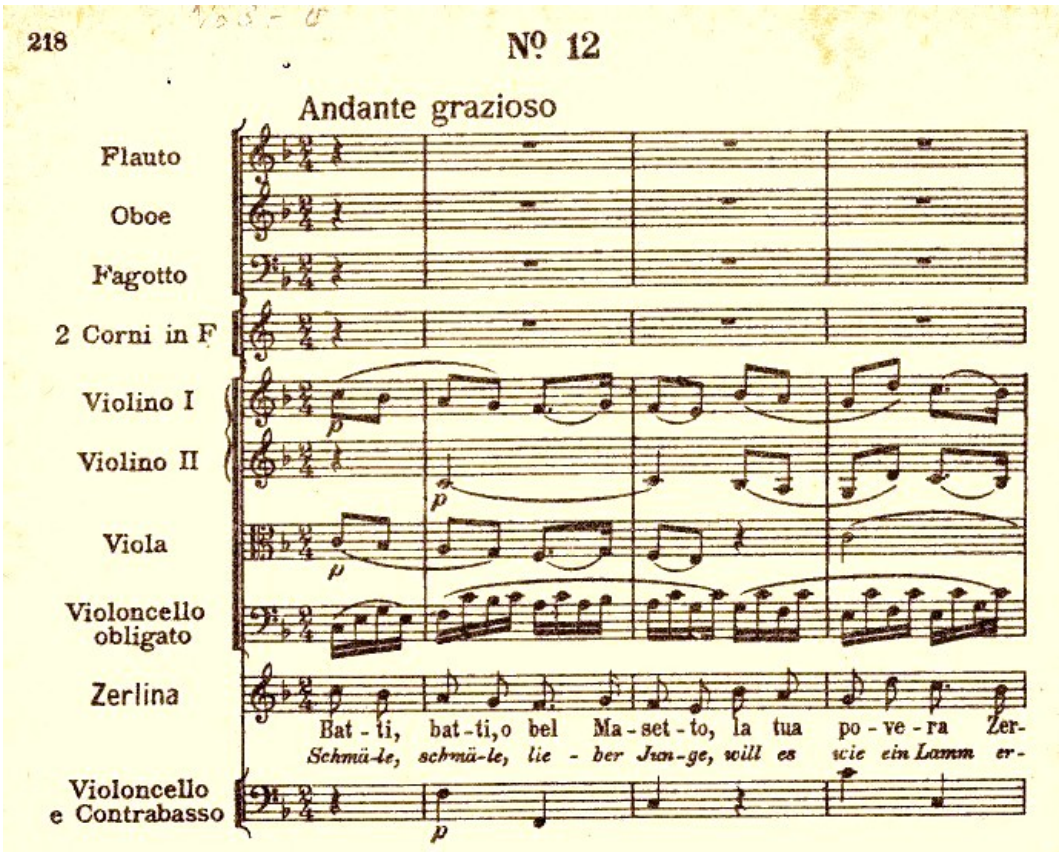


図 2.1　音楽の総譜は複数の時系列を並列に書き表した文書とみる

2.4　標準的な語順

2.4.1　語順で重要なのは動詞の位置

一つの文単位の語順は、一つの言語環境に居るときは、無意識に使っている規則です。そのため、他言語、特に英語表現と較べるときは、意識しておかないと間違って理解が起こります。日本語はＳＯＶ，英語はＳＶＯと言うのが語順です。語順は、大きな分類では、主語部（Ｓ）と述語部（Ｖ）の二つです。述語部は、動詞を基軸にして、幾つかの追加の語を組み合わせます。これを目的語（Ｏ）と括ります。伝えたいことを過不足なく一つの文にまとめるとき、昔の軍隊では「いつ・どこで・誰が・何を・どうした」と言うように教育しました。この並びの、後３つがＳＯＶ、前二つが状況説明の語句であって、補語と言い、英語では attribute です。補語の語順としての位置は、説明を補う意義がありますので、かなりの自由度があります。文章をまとめるときの基本として５Ｗ１Ｈを満たすことが言われます。これは日本独特の規則だそうです。文単位の構成規則ではなく、文章全体の構成をまとめるときの心覚えにします。目的語は他動詞が取り、普通は**物**を表す名詞ですので、英語では object と言います。英語の object は、物だけでなく、人や動物も含める意義の抽象語です。和語で発音して「もの」と言うと「ひと」を含みません。文語では「者」を「もの」に当て、人を表します。漢字の「物」は、人物・動物・植物のような熟語で現れるように、意義としては object に通じる抽象語です。日本語の文法書で目的語と訳したのは、当を得た名訳だと思います。自動詞は目的語を取りませんが、状況説明の補語は、必要に応じて使います。

2.4.2　直接目的と間接目的の語順

他動詞は目的語を取ります。これが一つの場合と二つの場合があります。英語の文法を習うと、直接目的語と間接目的語の二種を区別します。助詞の「～を」を付ける語（**直接目的**）が主です。「～に」を付ける語が（**間接目的**）です。「犬に餌をやる」と言う句は、語順を変えて「餌を犬にやる」と言っても間違いません。英語は主語を必ず立てますが、主語なしの動詞句の構成で言う語順は、「give dog food」「give him dog」が標準です。日本語では「～に～を」の順と覚えます。「～を」をとる語は、英語では文の最後に来ます。論理的に考えると、直接目的語は、なるべく動詞に近い位置にある方が自然です。英語は、目的語の語順を変えるとき「give food to dog」「give food to him」のように前置詞を補わないと、意味的におかしくなります。コンピュータ言語で、或るメモリの内容を別の場所に移す copy や move の他動詞は、英語流の語順を誤解すると間違った処理になります。「move A B」の形で書いてあるのは、B を A に移すのです。物の動きと文の並びが逆順です。この英文を日本人が見ると、A を B に移すと誤解します。話の筋で直接目的の B を省くことができるとき、「moveto A」の表し方があって、グラフィックスのコマンドに見られます。代数式の「A=B」は代入文と言い、データの流れは B から A の向きです。処理の向きが移動対象と逆順になることは、英米人でも困るようでして、その分、文法書の定義がくどくなっています。

2.4.3　状態と性質を説明し補足する語が形容詞と副詞

文の主役は、名詞と動詞です。名詞は主語と目的語に使います。単独に名詞だけを使うのではなく、幾つかの説明を補足する語を従えて名詞句、またはずっと長い文として名詞節を作ることもあります。名詞を補足する、または修飾する語が、形容詞の大枠に分類する語です。形容詞は、名詞の直ぐ前にきますので、形容詞が現れたら名詞が続くとの予測が働きます。ただし、ラテン系の言語は、形容詞を名詞の後に続けるのが標準です。副詞は、動詞を修飾する語ですので、同じく、動詞の直前に置きます。こちらも、副詞が現れたら動詞がくるものとの予測が働きます。英語もこの語順が標準ですので、構文として主語に続く語順が標準です。副詞を文末に補足的に使うことがあります。このときは、コンマで切って繋ぐ使い方をみます。日本語の形容詞と副詞は似た使い方をしますので、同じ語幹で、例えば「美しい花」「美しく咲く」のように使い分けます。これを活用する語であると言い、動詞的に「花が美しい」のようにも言います。音声口語体では、動詞的な文末は座りが悪いので、「美しいです」のような言い方をします。文章口語体では、この言い方も馴染みませんので、筆者は形容詞そのものを使うことを避けて別の言い換えをしています。形容詞と副詞とは、定量的な表現が曖昧になりますので、論理的な文章ではできるだけ使わないようにします。新聞、ニュース報道、法律に絡む文章、学術論文は、量的に定義ができない限り、意識して形容詞と副詞を省くか制限します。コンピュータのプログラミング言語に、形容詞と副詞は一切ありません。楽譜は、調子を指示する語、例えば図 2.1 にある grazioso のような形容詞の記入がありますが、テンポの Andante には大体の定量的な基準があります。

2.5　語順の分解と組み立て

2.5.1　単語は多様に使い分けること

科学的な研究方法を言うとき、動作を意味する言葉として分析や解析が表れます。意義としては、観察事象をバラバラに分けることです。漢字の「科」は、共通する性質を手掛かりにして種類に分ける意味を持っています。言葉の研究も同じ方法を使います。語を品詞の色眼鏡で分類し、音の最小単位に分解すると、文の構造が分かった気になります。しかし、現実に話したり書いたりするときは、語を並べて合成し、意味を伝えています。注意深く言葉を選ぶとしても、語の品詞分類を意識しません。人間社会では、商売のように、具体的な物を前にして話しをしますので、物の名前、つまり、名詞を使う頻度が高くなります。しかし、名詞は並べて言うと、先に出る名詞が後の名詞に対して形容詞的に振る舞います。形に表せない抽象的な事象を言う名詞、動作を言う名詞は、言語環境が異なる場合、対応する語がないことがあります。外国語を日本に輸入するときは、言葉としてはすべて名詞扱いをしておいて、元の意味を考えて、動詞として使うときは活用語尾「～する」、形容詞として使うとき「～な」、副詞にするときは「～に」の助詞を付ける方法を使います。このように使う名詞を、「**する名詞**」、「**なに名詞**」と区別する言い方があります。漢字の熟語も、元はと言えば輸入語です。意味は字形を見れば分かりますので、使い方を誤ることはありません。英語では同じ綴りの語で種々の品詞の使い方をしますが、語順で使い分けます。「する名詞」的な使い方には、do を付けるのが当たります。命令文では be も見ます。

2.5.2　動詞も多様に使い分ける

動詞は、言語種類間の差が大きい品詞です。外国語を正確に覚えるとき、動詞の使い方に重点を置きます。日本語では活用を覚え、英語は時制と不規則動詞を覚えます。人称でも変化します。和語、つまり、元からある日本語の動詞語幹の言い方は、同じ意味の漢字の中国語発音とは別ですが、便宜的に漢字を当てて和語的に読ませます。これが訓読みです。この使い方が日本語を複雑にしている一つの理由です。例えば、「走行（そうこう）する」と言うのはする名詞的な言い方、「走（はし）る」は訓読みの使い方であって、意味は同じです。前者は、中国語からの輸入語の性格を持っていて、格式を持たせた書き言葉に使う傾向があります。眼で用語を確認して同音異義語と区別しなければなりません。後者は、話し言葉で使っても意味を取り違えることはありません。日本語の動詞は活用します。動詞を形容詞的、または副詞的に次に続く語句を修飾する使い方があって、連体形・連用形としています。文の中の語として機能は、形容詞や副詞になっています。連体形で使う文は、英語文法で言う関係代名詞的な使い方であって、先行詞との語順が逆です。つまり、動詞の意味が強いのですが、修飾語としての使い方もします。動詞として挙げておいた言葉が、他の品詞の性質に変わる使い方は、品詞分類に拘ると、理屈を付けるときに困るのでしょうが、日本語では、上で説明した「する名詞」「なに名詞」に見るように、多様に語尾を変えて、目的に合わせた品詞に変えます。動詞も例外ではありません。

2.5.3　数式は英語文章の一種であること

明治以降、欧米からの文化を輸入したとき、文字記号を使った数式表現は、日本語の環境には無かったので、そのまま、つまり借用語の扱いをしました。四則計算は商業活動では必須ですので、それを言葉で説明する言い方は、どの言語にもあります。数に代えて、文字や記号を使って計算手順を表すことが代数の原義です。日本人にとっては、代数式は、最初から特別扱いをしています。欧米人は、語を記号表現に代えたものとして、元の語に直して声に出して読み、また、それを文としても書きます。（因みに、ＵＳＡのような表し方は頭字語（acronym）と言い、元の語に直して言うこともします。）数式記号も、日本語の語順と英語の語順とは向きが違います。日本語では、例えば「A 足す B は C」のように言いますが、英語は「C is equal to A plus B」が普通です。これを記号化して「C=A+B」とします。割り算を言葉で言うときは、二通りの語順があります。「A を B で割って C を得る」操作を説明する句は、「we divide A by B getting C」または「A is divided by B getting C」が普通です。ところが「we divide B into A getting C」の言い方もあるのです。A と B の語順が反対です。最後の言い方は日本人には馴染みが無いのですが、欧米ではごく普通の言い方だそうです。コンピュータ言語の COBOL には、この二つの文があって、前置詞 by（日本語の助詞で相当）と into（に相当）で使い分けます。この違いは、直接目的語を A にするか B にするかと言うことと、それを踏まえた語順が関係しています。このように比較してみると、日本語は助詞を使うことで、語の組み立て順に影響されずに、正確な意味表現ができる便利さがあります。日本語の文は、幾つかの条件を先に並べ、最後に動詞で締めます。

2.6　漢字熟語の語順

2.6.1　眼で見て確認する読み方をする漢字

漢字は日本語の中に取り込まれて、日本語の重要な語彙を構成しています。しかし、元は中国からの輸入文字ですので、日本古来の和語の読みとの同居が多くの問題を起こしています。その一つは、漢字を日本風に読むときの読み方が何種類もできたことです。加えて、多くの同音意義の熟語があります。正確さを確保するには、文書に書きますが、そのまま話し言葉に使うと、やや不自然になることと、同音異義語の解釈違いで、間違って理解されることがあります。文書に書いたものを読むときは、頭の中で音声に直して理解しています。英語でも、acronym は、読み方が複数になることがあります。文字を介さない対話の中では、それまでの状況から、どの意義の熟語であるかを瞬時に判断しています。これが、第 2.3 節で説明した予測と修正です。文書を字形によって意味を補う読み方をする方法は、日本独特です。ところが、最近のコンピュータのプログラミング言語は、英字の大文字と小文字とを別の文字としてコンピュータに認識させることが増えています。人の方では、眼で字形を確かめながら読み書きする必要があります。これは同音異義語が使われるようになったことと考えることができます。そうなってくると、日本語の文書の書き方は、情報伝達には最先端の方法です。振り仮名を付けると、音も、不十分ながらも記録できます。音声を記録する表音文字だけの表記法は、情報伝達の日常道具として不便になってきました。ただし、絵文字(pictograph)となると、読みようがありません。

2.6.2　二ヶ国語を混ぜて使っていること

漢字の熟語は、単独には名詞扱いですが、前の第 2.4 節の始めで説明したように、「する名詞」「なに名詞」の使い方があります。例えば、「運動」の熟語は、構成する漢字に動詞の意義がありますので、「運動する」と素直に使います。一方、例えば「優美」は「なに名詞」の性格がありますので、「する」を付けると意味が通りません。不思議なことに、漢字の辞書には、品詞種別の説明がありません。知り合いの中国人に尋ねたところ、漢字を品詞別に分類する習慣は無い、とのことでした。品詞分類の考え方自体、歴史が浅いことを考えれば当然のことです。日本人が、漢字個別に動詞・形容詞または副詞の種別を判断している理由は、日本語の読みに漢字を訓読みで当てる習慣が定着しているからと思います。同じ文字を使っていても、音読みと訓読みの違いは、全く別系統の言語、つまり、中国語と日本語の違いです。日本語（和語）には抽象的な意義を表す言葉がありませんので、語彙を補うには中国語の熟語を借用語として使う必要があります。しかし、読みは日本風に訛った（なまった）音読みです。日本語の文書は、読みの異なる漢字がまぜこぜに使われていますが、この不思議な構造を違和感なしに眼で読んでいます。欧米語も同じ様に日本語の中に名詞として借用しています。こちらは音を仮名で書きます。これも、元の言葉が動詞であれば「する」を付け、形容詞や副詞に使うときに「な・に」を付けた使い方をします。電子レンジを使うようになって「チンする」の言い方も出てきました。読み手が困るのは、元の欧米語の知識が無いときです。

2.6.3　熟語の作り方の規則が二つあること

漢字を２字使う熟語の作り方は、本家の中国では規則性があります。動詞に使う漢字は（この言い方は筆者の品詞分類法です)、熟語に構成するときに、「動詞・動詞」とするか「動詞・目的語」の順に並びます。先に出る語が第一義で、続く方が意味を補います。語順から言うとＶＯを踏まえています。中国語を真似て、日本で造語した熟語を和製漢語と言い、特に、中国人には通用しないものを言います。例として「落馬・離島・水防」を挙げましょう。落馬は、中国人の理解は馬の方を落とす、または馬が崖から落ちる意義に理解します。日本語では、人が馬から落ちる意義で使いますので、日本語の辞書に説明が載ります。離島も、人が島から離れる意義で中国人は理解します。もともとの日本語は、連体形で使う「離れ島」であったのですが、「れ」を送らない書き方を音読みにして熟語化しましたので、言葉を乱した使い方です。水防は、洪水のときに活躍する水害防止を詰めた和製漢語です。中国語風の言い方ならば防止水害から防水となり、文としての文字並びです。似た使い方は、ひけし（火消し）があります。漢字を当て、送り仮名を取ると、火が消えた意味に取られます。情報は敵情報告を詰めた熟語であって、森鴎外が始めた和製漢語と言われています。中国流の語順は報告敵状です。このように見ていくと、漢字を音読みにするか訓読みにするかを助けるには、適切な送り仮名をつけることです。日本語では、動詞に連体形や連用形があり、これが修飾語として前に出ます。送り仮名を省いて熟語に作るのが和製漢語です。耳で聞いても分かる文章口語体の書き方は、送り仮名を適切に使う必要があります。このとき、送り仮名は漢字の後に付けますので、眼で見て文字の先読みをしなければなりません。

2.7　作文指導法

2.7.1　言葉は発想と連想で出てくる

言葉は、基本的に耳で聞いて、真似て発音することで覚えます。文字から覚えるのではありません。音声として言葉を聞いているとき、瞬間的・直感的に理解しています。意味を理解するには、僅かですが時間がかかります。レトリック的な言い方は、表面上の言葉の意味ではなく、その裏を理解しなければならないからです。言葉を文法構造に分解するような言語学的な方法を踏まえているのではありません。それを気にしていると、発声の速度に理解の速度が追いつきません。言葉の組み立てには、大枠の法則があります。しかし、実際に話され、また作文される自然言語は例外だらけです。そのため、自然言語の研究は、統計的または確率的な成果で満足しなければなりません。言語学の研究態度は、既に発声されて文字になった、言わば、死んだ文の構造を解析します。これから話そうとする、または、文章で表そうとする、生きた（英語で言う live;ライブな）言葉の組み立てについては、何の役にも立ちません。話し言葉は、感覚的な発想と連想を瞬間的に声に出します。言葉に出るのは、それまでの経験以上のことは出ません。幾つかの記憶から連想が膨らんで次の言葉が出てきます。豊富な言語経験があるのが役に立ちます。おしゃべりな人は、話し出すと止まりません。同じことを何度も聞かされて迷惑することもあります。書き言葉に表すときの作文指導は、「思った通りに書きなさい」と言う以外に助言のしようがありません。既に書いたものは、後から批評を加えることができます。実用文書の作文指導は、伝えたいことが始めに分かっていますので、文単位としての書き方を定型的に提案することができ、また、全体文章のまとめ方に重点を置くことができます。

2.7.2　言葉を反芻して修正していること

声に出して話す言葉は、説明が不足する場合もありますし、しょっちゅう間違えます。話している本人は、気が付けば、その場で言い直しをします。昔の軍隊言葉では、「もとい」と言って言い直しました。聞いている側が理解できないとき、問い返すのが良いのですが、相手の話しの流れを中断させますので、マナーが必要です。細かな間違いをあげつらうのが、揚げ足取りです。公式の場で発言するとき、揚げ足を取られないように言葉を選ぶと、当たり障りのない話になり、いわゆる官僚の答弁調になります。対話のときは、質問と意見とを峻別しなければなりません。質問は、相手の答えを要求しますので、質問と応答とが対です。英語を略した Q and A も見ます。意見の方は、必ずしも相手が応答することを期待しません。自己主張ですので、目立ちたがり屋に見えることがあります。匿名の投書は、一種の意見表明です。自分を隠して言葉の攻撃になることがありますので、マナーとしては褒められた方法ではありません。学術論文の覆面審査ではよく起こります。公開の学術発表の場では、年長者が知ったかぶりで、いじめに当たるような、温かみのない、一方的に意見を述べることがあります。声に出す言葉は一過性ですので、文書にして、それを書いた本人が読み返すのは、言わば反芻です。他人に見てもらうのは意見をもらうことです。修正の責任は本人が取ります。他人が書いた文書に自分で手を加え、それを断りなしに自分の文書にするのが盗用です。したがって、引用の方法にもルールがあります。組織として学術論文作成時に、参加した全員を著者名に入れることが要求された時代がありました。

2.7.3　英語の素養が必要であること

外国語を習うことの目的は、海外旅行などでその地の言葉での交流もありますが、日本では、その言語で書かれた文書から知識を吸収することに重点がありました。この学習法は、音声の方を幾らか軽く扱います。例えば、英語の学習の定型的な応用は、日本語に直す英文和訳です。このとき、文法の知識を学習しておいて、辞書を利用して単語の意味を調べます。逆に、日本語から英語に直す英作文、つまり和文英訳が必要になることは非常に限られていて、受験英語のような特殊な場面に現れる程度でした。コンピュータを使うときの人工言語がプログラミング言語です。プログラムを作成することは、一種の和文英訳です。コンピュータ言語は、英語の影響を色濃く持ち、英語圏の人は、これを英語の特殊な方言(dialect)として扱います。日本人がプログラミングを学習するときは、英語の基本的な知識があって、見本とするプログラムの実物を理解することから始めます。これは英文和訳の段階に当たり、使われているキーワードの知識と文法をマニュアルで調べます。これは辞書の利用に当たります。プログラミングは英語方言を使う英作文に当たります。何を言いたいのかの目的別に単語の選択や表現法を選びますので、遠回りのように見えても、まず、論理的な日本語での作文技術を踏まえた上の話しです。

3.　名詞の話し

3.1　言葉を覚える過程

3.1.1　実物を見て名前を覚える

幼い子供が言葉を覚えていく過程で使う最も基本的な言い方は、眼に見える物を前にした「これ何？」の質問です。親は、それに対して物の名前を教えます。同じことは、海外旅行をするときの基本的な言い方に使います。現地の言葉を覚える方法は、「これ何？」に当たる言い方を一つ覚えて、会話の糸口を作ります。英語ならば「What is it?」フランス語は「Que'est-ce que c'est?」です。相手は名前を教えてくれます。眼の前に物がないとき、絵本を介して子供は物の名前の言葉、文法的に言えば**名詞**、を覚えます。このとき、今までに見た経験の有る物に興味を示します。大きくなって、知的興味を満たすために参考にするのは、挿絵（イラスト）入りの百科事典です。使い方は、ランダムにページをめくって、絵に引かれて読む楽しみの他に、「見たことは無いが、話しに聞いた何とかとは？」のように、名前から実体を知ろうとするときに使います。Duden は、挿絵入り辞書の先駆的なドイツの出版社として知られていて、各国語版があります（図 3.1）。物についての大まかな分類概念が先にあって、詳しい名前や言い方の区別を知りたいときに便利です。内容は、専門性が強いのが特徴です。書棚に置いておきたい辞書として、フランス語では Petit Larousse illustre も楽しめます（図 3.2）。インターネットで検索できる時代ですが、ハードカバーになった絵入りの辞書は長く愛用できます。

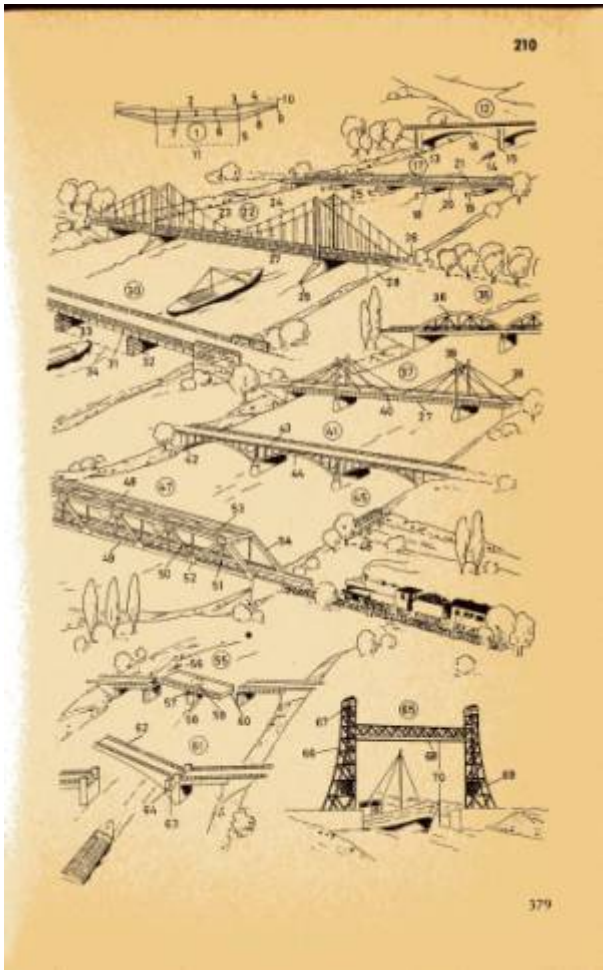


図 3.1　専門性の高い絵入り辞書(Duden)

図 3.2　Petit Larousse illustre'のページから

3.1.2 象形文字の価値を認識すること

漢字は中国で生まれ、工夫の積み重ねが何千年も続けられてきた文字です。この造語法は、物の形に似せて作る部分（偏など）と、音を表す部分（旁：つくり）を組み合わせて意味を表します。漢字は、眼で見て分る象形文字(hieroglyph)の性格がありますので、読み方が分らなくても意味を伝える手段に使うことができます。読みは、中国でも時代と地域によって変化があり、日本では日本風に変化します。しかし、音声による話が通じなくても、双方で漢字を知っていれば、筆談で意思を伝え合うことができます。漢字を覚えることは、絵を媒介として言葉を覚える方法でもあります。筆者の経験をお話します。パリからスイスに向かう列車の中で、前の席に座ったご婦人が、筆者が日本人であると知って、漢字を趣味として勉強していると話してくれました。「木」の字を二つ並べると「林」、三つ並べると「森」を表すことを面白がっていました。そこで、「林」は人工的に木の集合を作った場所、「森」は自然状態で木が集合した場所の意味を持つ、と説明したところ、非常に喜んで、納得もしてくれました。漢字は、字形を見た瞬間に意味を直感的に理解できる便利さがありますので、文書にしたものは、漢字だけを拾い読みしても大意が分ることがあります。中国のテレビ放送で漢字の字幕が表示されるのを見るとき、このような理解方法もしています。

3.1.3 読み易さを助ける漢字かな混じり文書

日本語の文書は分かち書きをしませんので、語の切れ目を眼で確認して、頭の中で声に直して読んでいます。言葉は時系列ですので、後戻りをしないと理解できない書き方があると、読み難いと感じます。部分的には、漢字は、その送り仮名を見て読み方（音と訓）を修正することをしています。これが、先読みか、後戻りか、に当たるのですが、ほとんど意識しない瞬間で分ると早く読めます。文章の書き手は、読まされる側が読み間違えないように、気配りをすることが必要です。英語文書は、語単位で分かち書きをすることと、語順で語の用途が決まりますので、コンマとピリオドの使い方に文法的な規則があります。日本語の文書では、「てにをは」の助詞を付け、送り仮名を付けますので、語の用途が分り、語順を変える自由度があります。日本語の句読点のうち、特に、読点（､）の使い方には規則がありません。書いてある文書は、頭の中で声に直すのですが、このとき、句読点や括弧を読みません。そこでは、音を切るか息継ぎに使う場所です。括弧で括った部分は、主文とは別に、声を落とす気分で読むか、読み飛ばしをします。この読み方のとき、図形としての漢字が重要な働きをしています。このことは、無意識ではありますが、漢字に焦点を当てた字面解析（じずらかいせき）をしています。この理解方法を、コンピュータにやらせようとする研究が、意識的な字面解析です。成功率は惨めなのが現状ですが、それは元の文章が良くないことの方が主な原因です。パソコンが普及してきましたので、パソコンの利用を考えて、文書作成技術を見直すことを提案してもよい時代になりました。比喩的に言えば、パソコンを、全く言葉を知らない幼児と見て、辛抱強く、言葉を覚えてもらうにはどうするか、になるのです。

3.2 外来語は名詞扱いとする

3.2.1 漢字は第一義的には名詞扱いである

古い時代、日本語は文字を持っていませんでしたので、漢字を中国から輸入して使い、仮名は、それを日本風に改良して作った表音文字です。物扱いをする第一義の言葉は名詞です。輸入品ですので、元の言い方、つまり音読みで使います。ただし、発音は、文化の輸入時期の違いで、漢音と呉音が混ざりますが、日本風になるのは仕方がありません。日本語の動詞と形容詞は活用しますので、音の変化がない語幹部分に同じ意味を持つ漢字を当て、全体を日本風に読ませました。これが訓読みです。日本語の音が先にあって、それに漢字を当てることも行われましたので、読み方がクイズなるようなものも使われます。「不如帰」「時鳥」と書いて、「ほととぎす」と読ませるのが一つの例です。地名は、昔から和語の読みが元からあって、それに漢字を当てたことから、同じ漢字綴りでも読み方の違いがあります。人名の姓は、地名と関わりを持つことが多いので、同じ音で違う字を当てることも、また、同じ字を使っても読みが異なることがあります。鉄道駅名は、文書では字数の節約ができる漢字で書いてあっても、駅名の表示板はひらがなで書いてあります。これが地名を正確に覚える助けになります。道路の案内標識の地名は、字数の節約と、図形として瞬間的に理解できる漢字を主に使います。道路交差点に、その場所の地名がローマ字表記もされてあるのは助かります。

3.2.2 漢字を覚えるのは大仕事であること

表音文字の仮名書きをすると字数が増えます。しかし、字の種類は少なく済みます。漢字は、画数が多いことと、書くときは文意に合わせて漢字を選び、読むときは読み方と意味の理解が必要ですので大仕事です。江戸時代の庶民の教養は「読み・書き・そろばん」でした。このうち、書きは、仮名の「いろは」が必修でしたが、それで役に立ちました。また、耳で聞いて分る言葉を使いましたので、かな書きでも用が足せました。武士階級は漢学を修めることが素養でしたので、その延長として漢字のお習字を書道として修めました。（話しを脱線させて、筆者の経験をお話します。知人が、家業を継ぐために大学を辞める、と挨拶がありました。家業は何？と尋ねたところ、お寺だとのことでした。その次があって、勉強のし直しの最初がお習字だと言われて納得しました。）漢学に原点のある音読みの熟語を混ぜた物言いは、字を眼で見ないと分りません。庶民レベルで難しい漢字の熟語を音で覚えるのは、語りを何度も繰り返して聞く演芸を通してでした。現代のラジオとテレビは、耳で聞いて分る言葉を使います。英語は、耳で聞き分ける言葉の性格を持っていますので、それをカタカナ語にして使っても元のスペルが想像できて、意味が分るからです。しかし、英語の素養が低い庶民レベルでは、迷惑な言い方です。逆に、聞いて分らない言葉の代表が、お坊さんの読むお経です。一方的な物言いであって、意味が分らなくても、聞いて有り難がる習性がここで育ったのではないかと思います。演説調の物言いは、権威付けのため、音読みの漢字熟語を並べる傾向があります。本人は得意なのでしょうが、論理的な内容が少ないと、返って教養の無さが露呈します。

3.2.3 漢字を悪者扱いすることもあること

漢字の歴史は何千年も続いていますが、時代の流れの中で悪者扱いされることも起こります。しかし、長い年月に渡って使われ、記述されてきた文字資料が継承できるようなシステムを工夫する必要があります。これは保守的な考え方ですので、革命思想に染まると攻撃の対象になります。中国の文化大革命が大きな傷を残したのは気の毒です。表音文字体系を主張する過激な漢字廃止論は、さすがに受け入れられませんでしたが、漢字の使い方を簡単にしようとする主張には、それなりの評価ができます。しかし、国が指導して規則を作るのは問題があります。その理由は、規則を決める委員会などで、委員の恣意が入ることと、弾力的な運用を許さない権威主義的な雰囲気も生まれるからです。日本では、常用漢字に無いことを理由として、専門用語に漢字かなを併用する熟語（例えば剪断をせん断、剪定をせん定とする）や、音を合わせて別の漢字に直す、などが採用されました。これは、漢字の個性的な使い方を壊しています。ワードプロセッサが無かった時代、書いてある文字は読めても、それを正しく書くにはかなりの知識が必要でした。作家がいい加減な略字を書いても、出版社や印刷所で正しい活字を選択してくれました。ただし、言葉に敏感な作家は、この干渉を嫌いました。今のワードプロセッサは、読みを知っていれば、仮名漢字変換で幾つか候補の文字を表示してくれますので、自分の判断で常用漢字にない漢字も使えるようになりました。つまり、条件は変わりますので、規則化してしまうと時代の動きに合わせる対応ができなくなります。

3.2.4 カタカナ語の扱いの経緯

漢学を基礎においた本格的な国語の辞書は、大槻文彦（1847-1928）の言海が最初です（図 3.3、図 3.4）。これは語の並べ方がアイウエオ順です。庶民の向け辞書の語並びは、いろは順が普通でしたので、画期的な編集でした。そのため、索引に「いろは順」も付けてあります（図 3.5）。漢字かな混じりの書き方は、日本語の中でしっかりと取り込まれています。漢字は、もともと輸入語であって名詞扱いをしていますので、漢字の位置にカタカナ語を使うことは文法規則に違反しません。このとき、不思議な表現方法も見られます。例えば、洋琴と書いておいて、振り仮名をピアノと付けるのです。書き手は、ピアノだけで済ませたいのです。そこで、洋琴と書いてピアノと読む方法を提案し、これが普及すると、めでたく和製漢字熟語として認知され、洋琴と書いてもピアノと読みます。これが、漢字が意味を伝える機能を最大限に生かした和製漢字造語の一つの作成方法です。しかし、読み手は、「ようきん」の読みより元の発音で読む方が自然ですので、文字としての洋琴を使わなくなり、カタカナ語が残ります。明治以降、多くの欧米語、特に専門用語を漢字の熟語に直して使いました。英語を知らないことが普通であったころ、この翻訳は意義がありました。戦後は、この過程が間に合わなくなったことと、英語の知識が増えましたので、カタカナ語のままで使うことが多くなりました。新村出(1876-1967)の編になる広辞苑は、元の欧米語を知らない人向けに、国語辞書にカタカナ語も含めるようになりました。広辞苑に載ったカタカタ語は、日本語として認知されたと見なすことができます。固有名詞もイラストも含めてありますので、辞書というよりも百科事典の編集です。

図 3.3　小言海の見出し表紙の部分

図 3.4　小言海の奥付、明治 37 年(1904)

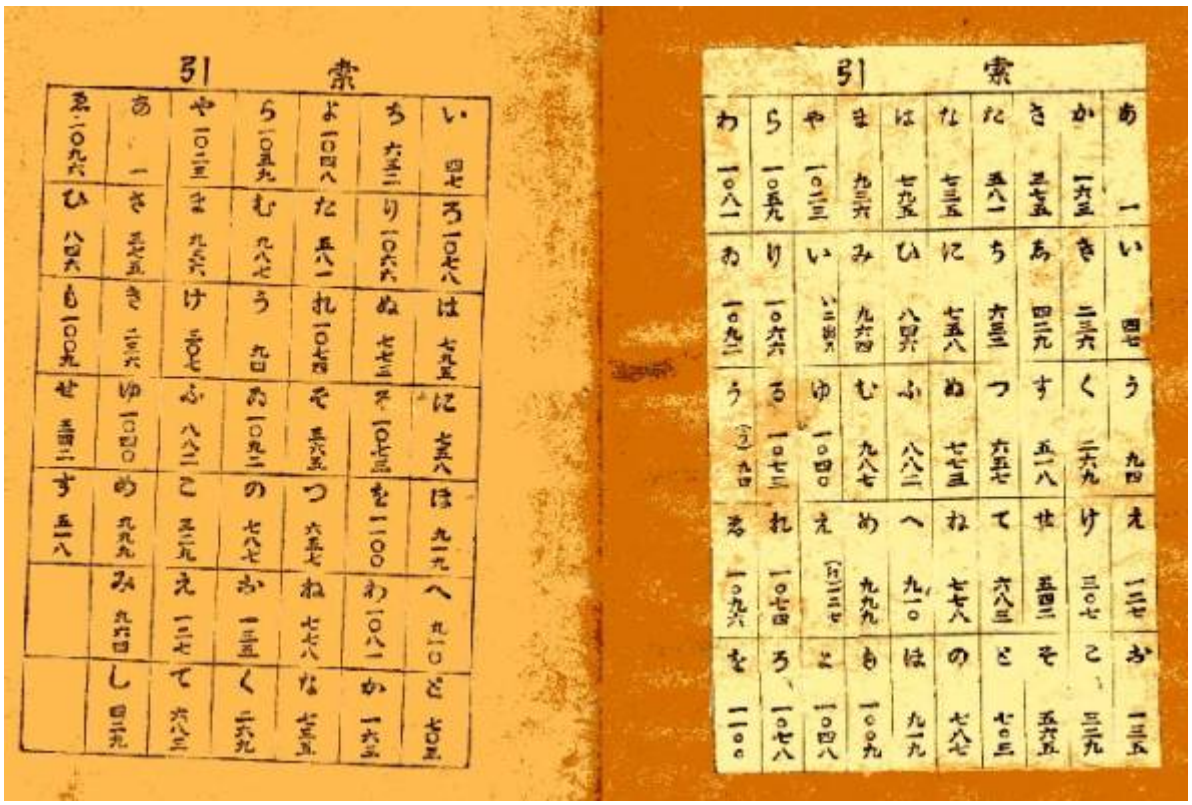


図 3.5　言海の索引、「あいうえお」順と「いろは」順も付けてあります

3.3 階層的な構造で使う名詞

3.3.1 同じものが複数あるときの総称が普通名詞

果物（くだもの）や野菜などの買い物に行って、同じ物が複数並んでいるとき、その個々を特に区別しない名前が**普通名詞**です。人を表す、やや具体的な用語「父・母・兄・弟・姉・妹」などが普通名詞です。個別には氏名がありますので、こちらは**固有名詞**と言い、一つしかないものです。個人が愛用する物に愛称を付けることもします。アルファベットで表す商品名の愛称、例えば自動車の TOYOPET（トヨペット）などは、他の同類と区別する品種名ですが固有名詞の感覚で使います。しかし、複数の商品をまとめて表す普通名詞扱いも起こります。しかし、s を付けて複数形にすると、固有名詞の使い方に矛盾します。この解決は、物質名詞並みに扱います。日本語では名詞に単複の区別をしませんが、助数詞（個、枚、本など）と数詞（一二…など）を使い分けることで、日常用語として困ることはありません。ただし、正確な言い方に精通しているとは限りません。英米人が日本語を覚えるときに、助数詞の種類が多いことと、物による使い分けに苦労するようですが、物質名詞と割り切って(a cup of water)並みに覚えるようです。

3.3.2 集合名詞の理解までには途中経過があること

幼児は、物の名前（**普通名詞**の方）と**固有名詞**と、二つの言い方の区別が分らなくて、すべて固有名詞にして言います。「猫ちゃん」「お猿さん」と言うのがそうです。生き物だけでなく、身の回りに在る日用品にも「ちゃん」を付けることもします。日本語では丁寧な物言いに、「お」を付ける習慣があります。女性が特によく使います。子供に物を大事に扱うことを教えるとき、物にも命が宿っているような話し掛けをすることに理由付けをすることができます。幼児に知恵が付いてくると、同じ名前で複数有る普通名詞の言い方と固有名詞との区別が付いてきます。普通名詞に付ける名前の約束は**定義**です。「象の花子さん」と言うときの「象」が普通名詞、「花子さん」が固有名詞です。この命名の過程を**宣言**と言います。普通名詞は、同じ概念の物（生き物も含めます）が複数あります。英語は、単数・複数を区別する言い方ができる名詞を**可算名詞**(countable noun)と言い、冠詞 a、複数形にする s 、動詞も活用形を変える、などを使い分けます。複数の物事の集まりをまとめて言う用語があって、文法では**集合名詞**です。代表的な言葉に family があります。漢字の熟語は家族です。日本語（和語）の環境では、集合名詞が未発達の古い時代に漢語が輸入されましたので、「族」に当てる和語の訓読みがありません。日本語（和語）では、「父母兄弟姉妹」は訓読みで使います。しかし、兄弟の言い方は集合名詞的であって、漢語の熟語を音読みで使います。科学と言うときの科はグループに分ける意義を持つ漢字です。科学的な方法の一つが分類学であって、あらゆる分野で応用されています。その創始者であるリンネ(Linne,1707-1778)は分類学の父と言われています。動植物を大枠から細分するまでの分類では、個々の集合名詞を表す漢字は、「界・門・類・綱・目・科・属・族・種」などを付けます。これらは、いずれも和語の語彙には無かった（抽象的な）言葉です。

3.3.3 抽象名詞の理解までにも段階があること

日本語の中での集合名詞、例えば家族は、構成の全体を指します。英語では、中身を一つ二つと数える意義で使う場合と、全体集合を単位とする何かの分類法があって、その単位で数える場合とがあります。英語の集合名詞は、単数形のままであっても、また複数の意義を持つ場合も、動詞の単数形と複数形とを使い分けます。この使い方は、話し手または書き手の生（なま）の意思と習慣を反映しますので、文法書の鵜呑みと受け売りは禁物です。例えばデータ(data)は複数形の語であって、単数形の datum があるのですが、動詞の単数形で受けることが普通に見られ、datum に別の定義を持たせる使い方もします。情報の語に当たる英語の information は**不可算名詞**とするのが常識です。しかし、前後の文脈から複数にしたいことがあります。これを**物質名詞**扱いで数えることをします。英語の水(water)は物質名詞ですので、(a cup of water)のように言うのに倣って(two pieces of information)と言えます。文法用語で言う名詞は、第一義として「実体の有る」物の名前を指す言葉、と説明があります。国文法の用語は**体言**です。「実体の有る」ことの説明用語に「具象」があって、和語的に読むときは「形（象）を備える（具）」と当てます。この反意が抽象です。形が有りませんので、説明が大変です。名詞の第二義として、**物事**の状態を表す名前をもつ**抽象名詞**があります。物事は「ものごと」と読ませ、漢字の物と事を当てた和製熟語です。音読みでは事物と使います。物は形が有り、事は有りません。したがって、事を指す名前を説明して、相手が同じ理解を持つようにするには、用語についての定義が必要です。また、その定義を理解する知的な、また感覚的な環境を共有する必要があります。

3.3.4 代名詞は名前を省略する言い方に現れる

物の名前、または言い方が分らないとき、または、名前の代わりに使う言葉が代名詞です。日本語には「あ・こ・そ・ど」を付けた「あれ・これ・それ・どれ」、「あの・この・その・どの」が当たります。「あれ・これ」と「あの・この」は this と that が当たります。複数形の these, those の区別に当たる言葉は、「ら」を付けた「あれら、これら、それら」があります。英語で言うならば、定冠詞を付ける言い方であって、話している対象を**限定**します。英語で定冠詞 the と不定冠詞 a の使い分けは日本人には難しいのですが、英語の native speaker が一々理屈を考えているわけではありません。ドイツ語の定冠詞群 (der des など) は、名詞の単複、性別、主語目的語などの用途、などで複雑に変化します。そこで、ドイツ語の native speaker 以外は、ごまかしてｄだけで済ますことをします。そう考えて、英語の the はドイツ語からの借用語を単純化したのだとする説があります。日本語の感覚から説明するならば、定冠詞は「あの・この・その」と言える状況で付けると覚えます。人称代名詞は、本人の名前を言う代わりに使います。日本語の言語習慣には、欧米語の人称代名詞のような限定的な言葉がありませんので、種々の言い方があります。「彼・彼女」は、翻訳語に使われて一般化しましたが、恋人の言い換えの意味を持ちます。日本語の環境では、個人名を直接使うのが普通です。

3.3.5 物の名前は属性を含めて理解している

果物の名前、例えば「リンゴ」を言うとき、実物を前にして「リンゴとはこう言うもの」とする共通認識を踏まえます。これが**概念**です。個別に性質を言う項目を**属性**(attribute)とします。その項目は、もし実物や絵が無くて、言葉だけで説明しようとなると、言いようが無くて困ります。実物の概念があると、色が赤い・少し大ぶり・値段が高い・産地は長野県・品種名は紅玉、などの説明が理解できます。これらが属性です。この全体集合を、コンピュータで整理するようになって、**情報**の言葉で括り、さらに個別の中身を**データ**というようになりました。人の場合に当てるとき、個人情報と言い、その中身は、名前（氏名）をキー(key)項目として、性別・生年月日・国籍または本籍・現住所・家族構成、などなどのデータです。属性それぞれに、上で挙げたような集合名詞的な見出しの言葉があり、説明に使う言葉は修飾語の性質があります。コンピュータで扱うデータは、見出しの言葉を変数名または配列名にします。属性の中身は、言葉つまり文字データよりも、相互に数量的に比較ができる数値データに直します。これを**符号化**(encode)と言います。元のリンゴの属性を具体的に言う「赤い・大ぶり」は、形容詞または形容詞的な言い方ですので、曖昧さを避けて数値化する方法も工夫しなければなりません。氏名などは文字データですが、文字コードを媒介として順序を決めることができます。都道府県名は、文字並びそのままをデータにできますが、配列の寸法を抑える目的から、符号化した短い名前または数字に変えることもします。これらの全体に、総称的な集合名詞の**データベース**(database)が使われます。

3.3.6 コンピュータ言語の中での名詞は定義と宣言で決める

プログラミング言語は、英語の構文法を色濃く持っていますので、プログラミング言語を理解するとき、英語の習慣から説明すると納得が得易いところがあります。名詞は、構文上の位置として主語と目的語に現れます。主語は、プログラマ本人か、コンピュータか、です。構文は、コンピュータを擬人化して対話の相手とする考え方です。プログラマの意思をコンピュータに知ってもらう目的語は、定義と宣言とで決めます。名詞を主語に立てるのは、宣言文と、数式の代入文の左辺がそうです。コンピュータに指示するのは命令文ですが、ここには動詞が使われます。プログラム文の中で、目的語（object）として使う名詞に当たるのが**変数**です。変数の中身は属性です。英語の object は、物と事との両方の意義を持つ抽象名詞です。日本人は物だけを意味すると短絡的に考え易いので理解の混乱を起こします。英語の用語「object oriented program」は、日本語に訳し難いので、カタカナ語のオブジェクトを使い、oriented に指向を当てるのですが、この全体を直ぐに納得するのは難しいでしょう。プログラム文の中での名詞、ここでは変数になりますが、「変数とはこういうものだ」とする**定義**を、ハードウエア的に、変数の型として前もって決めてあります。高級なプログラミング言語は、プログラマが定義することができます。変数は可算名詞の性格をもち、**配列**が集合名詞に当たります。また、**定数**は固有名詞の性格を持ちます。名詞は個々を識別する固有の名前が必要です。名前を付けて登録することで、始めて中身を参照できます。これが**宣言**です。集合名詞も含め、成分を個別に識別する方法が必要ですので、中身を順序数(index)で区別します。集合は、何層かの階層で仕分けする分類が必要になるとき、多次元の配列に構成します。C 言語の**構造体**と**共用体**は、プログラマが変数構成を定義と宣言とをして利用するデータ構造です。集合名詞として全体を一つの名前で単数扱いをすることも、また個別の中身を扱うこともします。C++言語になると、集合名詞のデータを扱う概念に class も使います。

3.4　修飾語として使う名詞

3.4.1　「何とか」の「何とか」と言う言い方

日本語では、名詞を形容詞的に使うときに「の」で繋ぎます。英語では人や動物の所有格(possessive case)を作るとき、’s(アポストロフィー・ エス)を付け、それ以外は前置詞の of を使うと習います。語順が変わるのが面倒なところです。所有と訳したので、持ち物の意義を感じるのですが、日本語には「長野のリンゴ」、逆に「リンゴの長野」の言い方もあります。所有の意味がなく、修飾的また説明的な言い方に使います。品詞分類を考えるならば、名詞を形容詞的に活用させるときの語尾と見ることができます。「の」を省く「長野リンゴ」は、熟語的な語構成で、一つの名詞扱いにしたものです。「りんごの長野」の言い方は、「リンゴが有名な長野」のような名詞句を短く現したと見れば、前半は形容詞句です。英語の場合、’s や of を、冠詞などと合わせて**限定詞**(determiner)としています。日本の短歌では口調を取るために「の」を連続して使う言葉遊びが見られます。落語のじゅげむ（寿限無）は、最後の名前「長助」を修飾する名詞の連続です。「の」が幾つもありますが、「の」を省く言い方も見られます。日本語ワープロ MS-Word では、「の」が幾つも並ぶ表現が見つかると、文章校正を警告してくれます。

3.4.2　階層化した集合名詞の位置を区別する

動物や植物の分類階級（科・属・種など）を表す漢字をつけた名前は、集合名詞です。あまり意識しませんが、住所表記は、グループ分けされた集合名詞の構成である固有名詞を形容詞的に繋いで使います。最後に使われる名詞に目的語の性格を持たせ、その前にある名詞は、その順で、後に続く名詞の修飾をします。前にある名前は、後の名前に対して集合名詞の意義も持ちます。筆者の住所表示は、以前は次のような具合でした（実際はこれと違います）愛知県名古屋市名東区猪高町大字上社字鋳物師洞一丁目１０番地上社住宅５棟６階 11 号。ここで、「県・市・区・町・大字・字・丁目・番地・棟・階・号」が階層分類に使う集合名詞名で、「愛知・名古屋・名東…」が成分（属性）の固有名詞です。区切りが分るように書けば、階層分類名を省くこともしますし、逆に「の」を間に挟む言い方や書き方も間違いにはなりません。「愛知の名古屋の名東の…」または「愛知県の名古屋市の名東区の…」のようです。特に、手紙の宛名を書くとき、上の例では、数字部分に、「１の 10 の５の６の 10」のような省略記法を使います。欧米流の一般的な住所表記の習慣は、逆順です。したがって、語の切れ目が分る書き方が必要です。このような名詞（数詞も含め）の並びで、個別の機能が何であるかを分類する文法規則を考えるとき、所有格では説明しきれません。

3.4.3　日付と時刻の書き順も集合名詞の意識で使う

数字は、大小の位取りが分る集合名詞を補助に使って言います。漢字では十百千万億兆…がそうです。欧米語も位取りの語があります。しかし、100 以下の数の言い方は、日本語から見ると、かなり乱れています。英語で 13～19 の言い方は、-teen ですが、一と十の文字並びは逆順です。ドイツ語は、21 を１と 20 の語順で「eins und zwanzig」と言います。フランス語では、70 から 100 までの言い方が特殊です。日付の年・月・日の数字並びの方法も、日本語の書き順と欧米流の書き順が逆です。欧米流は、集合名詞的には小さいグループから書き、限定詞に of や in の機能を持たせた/（スラッシュ）を間に挟む使い方をします。この書き方は、集合名詞の書き順となっていますので、割り算記号的に考えて、部分と全体の意義には合います。日本では割り算記号に解釈するとき、分子の方が分母よりも小さい表現方法に慣れていますので、例えば 3 月 4 日を 4/3 と書いてあると混乱を起こします。欧米語では、例えば 3 から 10 までの並びを表すときにも 3/10 の書き方をします。日本風では、そもそもスラッシュを使う習慣がありません。日付も時刻も大きい数グループから書き始め - (ハイフン)で繋ぎます。ハイフンを取って連続数字並びにしても、桁数を揃えれば、整数の大小順に矛盾が起きません。こちらの書き順の方が合理的ですので、国際標準 ISO でも、この方式を提案しています。

3.4.4 英語では順序を変えることがある

日本語で「の」を挟んで繋ぐ言い方は、英語では of で逆順に繋ぐことに当たります。意義的には階層的な集合名詞の順です。そのため、of 以外の前置詞(in, on, at など)を使い分けて、所有の意義よりも詳しい意味上の位置関係を表す使い方が見られます。和文英訳をするとき、「の」を of で置き換えるのではなく、適切な前置詞を選ぶと良い英文になります。of を使う場合は所有格の意義が強くなるのですが、むしろ部分と集合の関係を表します。意味的にそう扱うことができるとき、of に続く名詞は修飾的な作用になりますので、語順を変える表現も使います。例えば、パソコンの画面は、机の上を意味する「top of desk」が「desk top」、さらに詰めて１語の「desktop」が作られました。「database」は、軍事用語的に、データの基地を意味した base of data が始まりです。これが data base さらに１語の database になりました。日本語流の語順にすると、形容詞的な意義が強くなります。一種の妥協として、無生物的な集合名詞に、of を付けて後ろに繋ぐ方法と共に、’s(アポストロフィー・ エス)を付けて前に出すことも許しています。Government’s decision, company’s history などに見られます。Map of Japan と Japan’s map はほぼ同義です。しかし形容詞にした Japanese map は、意味的には曖昧です。ドイツ語ではアポストロフィーを省いた s を挿入して１語に繋ぐ方法を使います。

3.4.5 名詞を限定する方法

日本語で「の」を連続させるとき、最後に来る名詞が主題とする目的語です。修飾語が入り組んでいると、「一単位の名詞句に別の名詞句や動詞句が入れ子状に組み合わせること」があります。括弧で括った部分の言い方も、最後のことの前までが、主題とする目的語を修飾する動詞句です。英語にすると、ことを先行詞にした関係代名詞を使い、逆順に書きます。前項の例題の desktop を限定的に言うとき the top of the desk のように the を２ヶ所に付けます。最初に現れる the が主題とする目的語を限定し、前置詞を伴う方は、話題の対象とする名前を限定する使い方ですので、日本語の「あの・その」に当たる言い方です。パソコンで使う desktop は、言外に「あなたが使っている」の意味がありますので、固有名詞的に考えれば、冠詞を省いても誤解されません。

3.4.6 プログラミング言語は書き順の約束を覚える

プログラミング言語では、集合名詞全体の中で一つのデータ位置を参照する方法が必要です。その準備段階（宣言）では、大きい集合名を先に決めてから中身の小さい集合順に名前を並べる書き方を使うのが普通です。Fortran や Basic では、配列名が集合名詞の性格を持ちますが、中身は同じ変数型の集まりですので、中身を個別に区別するときは、順序数の添え字(index)を使います。２次元の配列を表すとき、Fortran または Basic で A(K,J)と書いてある配列名では、メモリの使い方の並び順は、最小の連続集合単位が K です。K 個ごとの集まりで J 個並べます。K,J を集合単位名とすると、英語の書き順と一致します。一方、C 言語では A[J][K]と宣言し、右端からデータを詰めます。この約束の違いは、二次元的に並べたデータをメモリ上に構築するときに、気を付けなければなりません。Fortran では、マトリックス（行列）の成分を表すとき、縦の列並び(行数 K)を連続並び単位として横に L 列繋ぐ構造です。C 言語では、横書きの K 文字並びを単位として、縦に L 行繋ぐイメージです。

3.4.7 構造体を扱うときは集合名詞を意識する

C 言語は、型の違うデータの集合を扱うことができます（定義がある）。構造体がそうです。struct のキーワードを付けて書く名前は、集合名詞であるとコンピュータに知らせます。それに続けて波括弧{}で括った名前の集まりが、その順で並びますが、同格の構成要素を表し、型の違うデータを扱うことができます。そうすると、集合名詞全部をまとめて参照するときと、構成を個別に参照したいときの書き方を決める必要があります。日本語の感覚で参照する方法が自然です。「の」に当たる限定詞にピリオド(.)を使います。名前の付け方は恣意的なところがあります。コンピュータ言語の中での名詞（変数）別の名前に直して参照することはしませんが、同じメモリ領域のデータを、型の違う別の名前で参照したいことがあります。Fortran では equivalence 文があります。C 言語では共用体(union)の使い方があります。名前を宣言することで、メモリに実際のデータ領域が確保されます。しかし、使わなくなったとき、そのメモリ領域を別の目的に当てて、効率的に運用したいことがあります。英語の用語では delete と erase の区別があります。Delete は完全抹消の意味です。Erase は、データの中身を消去することですのでメモリ領域そのものは残ります。８ビットマイコン時代の BASIC のコマンドには ERASE がありました。メモリ領域の全体寸法が小さいので、不要になった配列領域を開放し、別目的に使うことが目的でした。

3.5　形容詞から作る抽象名詞

3.5.1　形容動詞の品詞分類は評判が良くない

英語の品詞分類法に倣って、日本語の文法も形容詞の語彙を分類しています。細かな違いまで気にしだすと、分類法は細分化する傾向があります。学校文法では形容動詞の分類があります。和語では、語尾を「イ」とする（例えば；美しい）を形容詞、「カナ」を語尾とする（例えば；静かな）を形容動詞と言うのが紛らわしいところです。名詞、特に外来語を形容詞にする場合は「～な」と使いますので、ナ形容詞の言い方を使い、普通の形容詞をイ形容詞と分類するのが便利です。これは外国人に日本語を教える現場の経験から生まれた言い方です。外来語は、日本語に輸入されると名詞扱いが第一義ですが、元の意味に修飾語の性格があると、「ナ」を付けて形容詞に、「ニ」を付けて副詞に使います。そのような熟語名詞をナニ名詞と言います。

3.5.2　用言の言い方は品詞分類法と矛盾する

日本語の形容詞は活用する語（用言）であるのが特徴とされています。しかし、その機能を見ると、名詞の修飾が本命であるのに、動詞的な言い切りに使う場合、名詞にする場合、そして副詞に使う場合があります。例えば「美しい花・花が美しい・美しさ・美しく踊る」のようです。漢字の熟語は、例えば「優美」は、「優美な・優美である・優美さ・優美に踊る」と使い分けます。欧米語をカタカナ語にして使う場合も同じです。ただし元の言語を知っていて、それが修飾語の意義を持つときに限ります。例えば「ビューティ」「ビューティフル」に当ててみれば納得できるでしょう。「ナ」を付ける使い方を上の「イ形容詞」で見れば、「美し」の語幹に付ける語尾の種類が違っています。ただし、例外も幾らかあって、「大きい・大きな」「小さい・小さな」の言い方がありますが、独立した別の語ではないので「～な」の方の活用が不完全です。この現実がありますので、「イ形容詞」でも「～な」と言える場合があるときは、「～な」の方を使うことを避けるような作文指導をすると、文体の統一が取れます。日本語の形容詞を活用して使うと言う説明は、欧米人にとっては評判が良くありません。形容詞は、あくまでも名詞の修飾に使う語の意義だからです。英語の形容詞は、語尾に「-ly,-ness」を付けた別の語にして、副詞や名詞を作る例が多く見られます。フランス語では、男性名詞と女性名詞、単複、さらにリエゾンでも変化をします。これらは、活用とは別ものです。

3.5.3　「だ・です・である」は be 動詞の日本語版

夏目漱石の「我輩は猫である」の表題は、英語の「I am a cat」に対応します。上で挙げたナニ名詞は、単独に使うと前後の意味関係が取れませんが、語幹に「だ・です・である」を付けると用言の終止形になって落ち着きます。ナ形容詞も語幹に「だ・です・である」を付けられます。英語の be 動詞は、日本人にとって理解の難しい動詞です。英語の参考書を見ると、説明が山ほど出てきますが、あっさりと「だ・です・である」に当てると覚えると楽になります。動詞は多様に活用する語ですので、be 動詞も複雑に変化するのです。理屈を考えるのも悪くはないのですが、基本的には慣れで覚えるしか方法がありません。一方、イ形容詞は終止形があります。ところが、話し言葉では、終止形のままで「花が美しい」で止めると何となく落ち着かないので、「花が美しいです」のような言い方があります。これがやや奇異に受け取られるのは、終止形の用言が二つ重なっているためです。筆者は、このような場合には、別の言い方に直すようにしています。例えば「花が美しく咲いています」のようです。

3.5.4 「サ」を付けて程度を表す名詞にする

和語の形容詞は、名詞にするとき、語幹に「さ」を付けます。ナニ名詞も「さ」を付ける言い方がありますが、元の語幹だけでも抽象名詞になります。サを付ける方は、物事の状態を表す抽象名詞です。形容詞と副詞が表す意味は、主観的・感覚的に理解しますので、個人によって受け取り方が変わります。長さ・広さ・大きさ・高さ・速さ・重さ・暖かさ、などの言い方は、程度を意味する抽象名詞です。比較を言う言葉は、何かの基準をもとにした絶対比較と相対比較で言っても、曖昧さは避けられません。「曖昧さ」という言い方自体は、曖昧を測る基準がありませんので二重に曖昧です。したがって、客観的・論理的に定義するには数量化して比較ができるようにしなければなりません。このとき、高度・温度・速度・震度・密度のように「～度」を付けた用語を使います。学術論文や法律文書では、数で指示のない形容詞や副詞を使いません。数での大小比較は、その客観的判断を助けます。判断の分け方は、二分法と三分法があります。二分法は、yes/no、或る無し(1/0)を使い分けます。欧米人はこの方法を好み、どちらでもない条件を避ける傾向があります。日本では、三分法を良く使います。上中下、大中小に分類すること、じゃんけんでは勝ち負けの二分類に、相子または引き分けを含めます。数で言うと、大・小に等しいの条件(>, =, <)を加え、数字に直すときは (1, 0, -1)に分けます。コンピュータで数値計算をして実数値の大小比較をすると、等しくなる条件が成立することは実際には殆んど起こりません。等しい条件を設定したいとき、或る数値の幅を閾値（**しきい値**）に設定して、数値を**丸め**、整数化して比較します。コンピュータ言語では、丸めは応用技法ですので、分るような説明は文法書には載りません。正確さにこだわる理論家は、このような扱いが誤差を黙認するごまかしであると見なして嫌います。コンピュータを頭から信用している一般の人は、数値の実用的な処理に、丸めの技法が使われていることの認識がありません。

3.6 名詞から作る動詞

3.6.1 外来語はスル名詞から作る

言語間の違いが最も大きく現れるのは動詞です。外国語を覚えるとき、この章の最初に説明したように、物の名前は実物を介して覚えることができます。動詞はすぐには覚えられませんので、外国語を習うときには、丸暗記の方法しかありません。基礎として文法規則を覚えることが大切です。しかし、どの言語にも不規則に変化する動詞があります。理想を言えば、その言語を正しく話す native speaker について正しく手ほどきを受けるのが最善です。外国語にあって、日本語にない動詞を借用語として取り込むことについては、日本語は便利な方法があります。それは、元の言語を名詞として取り込んでおいて、「～する」を付けます（2.5 節参照）。古くは中国からの漢字の熟語、近代では欧米語の読みをカタカナで表しておいて「～する」と使います。このように使うことができる名詞は、元の言語で動詞の意義を持つものです。このように動詞に使うことができる名詞を**スル名詞**と言います。この用語は、**ナニ名詞**と共に、外国人に日本語を教えるときの実践的な経験から生まれたものです。語幹に当たる名詞は、抽象名詞です。この名詞を直接目的語にして、助詞「を」を付けて「～をする」の言い方も自然です。複数のスル名詞を並べるときは、「～をする」と使わなければなりません。「する」はサ行変格活用の動詞で、短く「サ変」とも言います。英語では do に当たるのですが、do を含む動詞句は、日本語の「する」ほどの多様さでは使われません。なお、口調の関係で「～ずる」と付ける動詞があります。

3.6.2 スルを含む不思議な文

「～する」と使う動詞の語幹は、和語にはない概念を表す抽象名詞を多く使いますので、「～する」を含む文書そのものは、日常の言葉とは離れた硬い文章になります。規則や規約を書いた文書には、「～するものととする」「することとする」のような表現を見ることがあります。前にある「～する」はスル名詞を受けるのですが、後ろの「する」は動詞の命令形を意識した使い方になっています。日本語の動詞は命令形を持ちますが、英語のような明確な命令形を使う構文の習慣がありませんので、命令の意思を伝える表現に困ります。文語的な言い方の命令形に「するべし」「するべからず」があって、漢字の「可」と「不可」を訓読みで使いました。最初のするに「べし・べからず」をつければ済むのですが、これを使わない言い方に、「ものとする」「こととする」を当てたと想像しています。また、この言い方では、ものまたはことを先行詞とした関係代名詞の言い方になっていますので、英語を習ったインテリは不思議な文とは感じないようです。

4. 動詞の話し

4.1 品詞に分けるときの動詞

4.1.1 日本語の動詞は活用させて別品詞としても使う

動詞は、言語間（日本語、英語など）で違いが大きい品詞です。言語間で比較するとき、単語単位で品詞に切り分ける方法は、必ずしも適当ではなく、動詞句または述部のように大きな枠組みで考えることの方がよい、と筆者は考えています。言語学的な品詞分類の考え方は、歴史が高々100年と浅いので、漢和辞典は漢字を品詞で分類していません。中国語は、動詞の意義がある一字の漢字を、熟語構成や文中の位置で、動詞や名詞に使い分けます。英語の動詞も、同じ使い方をします。人称・単複・時制・能動受動・肯定否定の別などで変化させます。これが**活用**(conjugate)です。文字のスペルが変化し、be, have などと合わせて２語以上で構成することもしますが、その全体が動詞です。日本語動詞の**活用形**とは、変化をしない語幹を軸に、別の文字や語を繋ぐ目的をもって送り仮名が変化する、そのときの個別の形を言います。つないだ一かたまりは、別の品詞の性質にもなります。文書に書くとき、語幹に漢字を当てると意義を限定できます。送り仮名を余分に付けたり、また省いたりすると意味を正しく表すことができません。これが送り仮名の付け方の約束を構成します。ただし、不規則動詞のサ変とカ変は、語幹も仮名書きをします。「仕分ける」の「仕」は、音読みの「し」を当てますが、「する」の連用形ですし、「つかえる」の訓読みが別にありますので、仮名書きの方が理屈に合います。「くる」も漢字の「来る」をあてない方がよいでしょう。

4.1.2 動詞の基本形があること

動詞は、これが動詞の基本だ、とする形があります。日本語は**終止形**です。欧米語は**不定形**の訳を当てます。日本語の動詞終止形は、語尾に仮名文字 50 音表の「う、く、す、つ…る」行を取ります。試しに 50 音の仮名一文字に「る」を付けてみれば、ラ行と濁音半濁音を除く、ほぼすべてに和語の動詞の意味があります。仮名で書けば２字ですが、発声単位としては一音節の語幹を持っています。面白いことに、フランス語の動詞の不定形は、語尾に″r″の文字を持ちます。ドイツ語の動詞の不定形は語尾が en で揃っています。動詞は活用させて使いますが、どの言語にもかなり整然とした規則があります。もしそうでなければ、個別に覚えなければ言葉を使えません。それぞれの言語を母語とする人(native speaker)は、そのような不便を感じないで言葉を覚えた人です。別の言語を習うときは、理屈で覚えようとしますので、規則が整然としていれば覚え易いのです。フランス語やドイツ語に較べれば、英語は動詞単語の文字構成は規則性が低い、と言っても良いでしょう。特に、不規則動詞の一群を別に覚えなければなりません。つまり、日本人にとって、正しい言葉遣いを覚えるのが厄介な言語です。

4.1.3 日本語の文法用語にこだわらない

日本語動詞の活用形の内、**連体形**は、名詞（体言）を修飾する形として使うので、そう言うことになっています。しかし、その使い方は、後に続く名詞の修飾句を作っています。機能としては、順序を逆にした、英語の関係代名詞(that または which)を使って名詞を修飾する使い方です。「～すること」「～するもの」の構文は、「what～」の構文に似た使い方です。「～するとき」の使い方は、「when～」の副詞句か、分詞構文の使い方と似ています。連体形は、終止形と同じです。修飾する名詞を主語とする意義があることを考えると、同じ語尾形を使うのは理屈に合います。筆者は、連体形の後が長い名詞句であるとき、意識して読点（、）で区切る書き方をしています。一方、**連用形**は、後に続く動詞など（用言）を繋ぐのでそう言うのですが、この活用形は、動詞を名詞化しています。英語との類似は、ing を付けた動名詞と考えると納得し易いと思います。例えば、「行き」と「帰り」は連用形ですが、「行き帰り」と繋ぐ使い方は名詞を単に並べたのであって、「行き」が「帰り」を修飾する意味はありません。名詞を並べると、前の名詞が後の名詞を修飾する熟語も作ります（第 3.4 節参照)。例えば「書き方」は、両方の漢字を訓読みする和語的な熟語名詞です。英語にすると「manner of writing、または writing manner」です。連体形で繋ぐ「書く方法」は、音読みの熟語名詞「方法」を和語で修飾したと見れば、副詞句を含めた全体を名詞句にした「the way how to write」のニュアンスが当たるでしょう。ここでの how は、上に上げた what と似た使い分けです。実は、英訳のとき、how と what の使い分けは、英語の学習者はよく間違えるそうです。さらに、送り仮名を使わない音読みの「書法」のような漢語の熟語名詞にすると、形式を持たせた硬い文章の顔になり、耳で聞くと、同音異義語の別の熟語と間違えます。

4.1.4　音節が短い動詞

日本語の動詞の表記法は、語幹に漢字を当て、活用部分に送り仮名を使います。語幹に当てる漢字は、中国語でも同じ意味か、それに近い漢字です。しかし、その読みは、中国語とは全く別の訓読みです。どの言語でも、よく使う基本的な、音節の短い動詞があります。英語では「do, have, make, take, get, put, set」などです。日本語（和語）では、「みる、とる」などがあります。ワープロの仮名漢字変換をすると、語幹に当てる候補の漢字が幾つか示されますので、場面に合った漢字を選びます。字が違っても、意義的には同じ動作に源があることが多いので、耳で聞く分には不便を感じません。前後の文脈で、慣用的な動詞句として使うからです。言葉を覚えることのかなりの部分は、慣用的な言い方を覚えることです。書き言葉にするとき、漢字の選択を間違えると恥をかきます。逆の場合があります。例えば「見る」「診る」「観る」「視る」「看る」は「ミル」と読みます。これらは、眼で文章を読むとき、表意文字の特徴として文章の意味を補う役目があります。しかし、声に出して話そうとすると、読みが分からないことも起こります。フリガナを付ける習慣が少なくなりましたが、それを補うのが、常用漢字の読みの約束です。先の例では、最初の二つだけに「ミル」の読みが許されています。

4.1.5　動詞に使う漢字がある

漢字一字で動詞の意義を持つものは、常用漢字の範囲で約 800 字あります。筆者は、これを**動詞用漢字**と便宜的に分類しています。その内、約 600 字は訓読み（くんよみ）があります。訓読みで使う動詞は、耳で聞いて分かり易く、軟らかい文体になります。文字並びが「訓読み」の語構成は漢字が二字並びます。「訓」の字は音（おん）でしか使いません。一方、「音読み」と書いてあると（おんよみ、おとよみ）どちらか？で迷います。書き手は、読み手が迷わないようにする義務があるのですが、その文書に関係している人の間（専門）では何となく約束があります。書き言葉で動詞用漢字を純粋に動詞として読ませるときは、一字の後に送り仮名を付けます。黙読しているとき、続いている仮名を瞬間的に判断して音訓を読み分けます。「～する」と続く熟語動詞は、通常は音で読みます。音だけで使う一字の動詞用漢字には、「～ずる」と使う字と（表 4.1）、「～する」と使う字があります（表 4.2）。漢字は熟語に構成する造語能力が格段に高いので、二字の組み合わせで使う音読み熟語のスル名詞は多く知られていて、慣用的な語彙の数でも約 6000 語あります。

表 4.1　一字で「～ずる」と使う動詞用漢字

漢字	音読み	訓読み	備考
案	アン		
演	エン		
応	オウ	（こたえる）	*2
感	カン		
興	キョウ	おこる	*1
禁	キン		
吟	ギン		
献	ケン		
滅	ゲン	へらす	*1
殉	ジュン		
準	ジュン		
信	シン		
生	ショウ	いきる	*1
談	ダン		
通	ツウ	とおる、かよう	*1
転	テン	ころがる	*1
点	テン		
動	ドウ	うごく	*1
任	ニン	まかせる	*1
念	ネン		
判	ハン	（わかる）	*2
封	フウ		
変	ヘン	かわる	*1
弁	ベン	（わきまえる）	*2
報	ホウ	むくいる	*1
命	メイ	いのち	*3
銘	メイ		
論	ロン		

*1）　訓読みの使い方もある動詞
*2）　常用漢字では訓読みを使わない
*3）　訓読みは動詞ではない

表 4.2　一字で「～する」と使う動詞用漢字

漢字	音読み	訓読み	備考
愛	アイ		
圧	アツ		
介	カイ		
害	ガイ	（そこなう）	*1
議	ギ		
給	キュウ		
制	セイ		
対	タイ		
託	タク		
適	テキ		
発	ハツ		
罰	バツ		
服	フク		
没	ボツ		
律	リツ		
列	レツ		

*1）　常用漢字では訓読みを使わない

4.2　動詞が作る複合語

4.2.1　複合動詞の作り方

例えば「打ち合わせる、取り替える、組み合わせる」など、前後が動詞であるものを**複合動詞**と言います。「打ち・取り・組み」が連用形で、後の動詞の意義を補います。前の語は、名詞化した動詞が修飾語になったと考えても、不都合ではありません。先行する語を形容詞の語幹にした「美し過ぎる、高過ぎる、静か過ぎる、優秀過ぎる、近寄る、遠のく」などがあります。名詞を動詞に直接繋ぐ言い方で複合動詞的な造語があります。その動詞が目的語や補語にする名詞を必要とするとき、普通は助詞「を」を語の間に置きます。助詞を取って繋ぐときは、動詞を連用形にして名詞化して繋ぎ、この複合名詞に「スル」を付けて動詞にします。例えば、「山を歩く→山歩きをする→山歩きする（自動詞）」「栗を拾う→ません。日本語の文法では熟語名詞、熟語動詞の用語も使いません。これらの造語は、一括して**複合語**と言うようです。

表 4.3　複合動詞に使う頻度の多い動詞用漢字（頻度数は参考値です）

漢字	訓読み	英語との対応	頻度数
替	かえる	to, for	21
返	かえる、かえす	back	38
越	こえる	over	14
出	でる、だす	out	106
取	とる	take, get	35
直	なおす	back	20
張	はる	fill, over	14
向	むく、むける	toward	13
寄	よる	in, at, by, aside	17
分	わかる、わける	off, between, into	15
下	さげる、くだる、くだす、おろす	down	18
過	すぎる、すごす、あやまつ	through	12
回	まわる、まわす	around	25
開	ひらく、あく	open	
掛	かける、かかる	put, set	38
換	かえる、かえす	to, for	
帰	かえる、かえす	go, come	
行	いく	go, come	
合	あう、あわせる	co-, in, into	102
込	こむ、こめる	set, put	
始	はじめる	pre-	
止	とめる	at end	
持	もつ、	take, have	
上	あがる、あげる、のぼる	up, on, over	92
切	きる	off, cut	36
渡	わたる	accros	16
登	のぼる	up	
入	いる、いれる、はいる	in, into	31
抜	ぬく	out of	29
付	つく、つける	set,put	97
揚	あげる	up, on, over	
落	おちる	down	22
離	はなれる	away	
立	たてる、たつ	upward	58

4.2.2　連用形で止めて名詞で使う複合語が多い

ここでの複合語は、送り仮名を持つ和語の動詞を組み合わせる造語法の説明です。慣用が定まったものは国語辞書に載ることもあります。しかし、語の組み合わせには、書き手や話し手にかなりの自由度がありますので、慣用されていても辞書に載らない複合語が山ほどあります。例えば「組み合わせる」と使うと動詞ですが、連用形で止める「組み合わせ」は名詞です。前を名詞と見て「組合わせ」と表記しても慣用的には正しく読めます。「組み合い」の使い方もあります。送り仮名をすべて省いた「組合」があります。最後が、典型的な和製漢字熟語です。音で読むと「そごう」ですが、この読みは使いません。音読みで使う中国語風の熟語を和製漢語と言い、中国語の語構成とは違うことが起こります。音訓まぜた言い方を「重箱読み・湯桶読み」と言い、読み手を悩ます問題の一つです。複合動詞の造語法で作られる複合語であっても、動詞としての使い方がなく、連用形で止めて名詞で使う用語が多く見られます。また、この言い方は、軟らかく聞こえ、言わば口語的です。他動詞として使うことができるときに「をする」「する」を付けます。例えば「建て売り、馬鹿騒ぎ」などです。動詞の部分を別の活用形で使うと不自然になります。「雨降り」は、主語が無生物で自動詞を使う合成語ですので、「する」を付ける言い方ができません。送り仮名、特に後ろの動詞のそれを省くと、読み誤り（誤読は漢語的な言い方）を起こすか、意味不明になるか、別の意味に取られます。

4.2.3　英語は句動詞の使い方がある

動詞の意味を持つ語を他の語と組み合わせる使い方は、すべての言語で見られます。英語では、前置詞や副詞を組み合わせて一単位の動詞として使う語を**句動詞**(phrasal verb)と言います。「set back, put up with」などです。複合動詞は、句動詞と似たところがあります。「上がる・下がる・入る・出る」などは複合動詞によく使う動詞です。英語の句動詞によく使う前置詞「up, down, in, out」と対応しています。表4.3に示した動詞用漢字は、複合動詞を構成する語によく使われる漢字を調べたものです。或る辞書的な資料から、漢字の出現数を数えた数を、頻度数として参考に示しました。英語の句動詞の構成と対応させるように、動詞と前置詞を思いつくままに書き出してみました。英語の句動詞は、前置詞を使い分けて意味を限定します。ただし、目的語を間に挟むなど、構成単語の並びが固定していないことが複合動詞とは違うところです。英語の接頭辞、例えばco-の付く動詞「cooperate, collaborate」の意味を考えると、一語に繋ぐ複合動詞の構成になって、漢字「合」を使う場面と似ています。

4.2.4　漢字２字のスル名詞は構成則が二種ある

漢字の本家の中国では、漢字一字ごとに固有の一音節の音があることと、同じ一音節でも、四声の区別があります。文中の位置で、一字で動詞として使う意義があっても、意味を補うように**二字熟語**にして使うと安定します。この熟語を輸入したものがスル名詞であって、音読みに「～する」を付けて動詞にします。これを句動詞と見ることもできます。この語構成は、中国語の習慣では最初の語に動詞の意義を持つ語を置き、次ぎの語にその動詞の目的語になる名詞、意味を補う動詞、補語的な語、などを繋ぎます。この構成法は、中国語の構文(ＳＶＯの順)と関連した（ＶＯ）の順です。この造語法を真似て作る和製のスル名詞があります。多くは、複合動詞の送り仮名を省いて作ります。複合動詞の語順は、前の語が名詞的、後の語に動詞用法を置きます。これは、日本語の構文が（ＳＯＶ）であることと関係しています。複合動詞の送り仮名を省くと、この熟語になります。「訓読み」から送り仮名を省いた「訓読」は、音で読ませる典型的な和製スル名詞です。また別の例として「読み解く」から「読解」にします。これらは中国語流の語順構成の方法と逆です。しかも「解読」の用語もあります。どちらも和製漢語のようです。

4.3　英語の be 動詞に当たる動詞

4.3.1　動詞を静的な描写に使う

普通の人は、改まって、この言葉は動詞だ、と区別するような文法的な分類を考えて、言葉を読んだり聞いたりはしません。日本語の物言い（ものいい）には、情景描写、つまり静的な事象を言う傾向が強いので、動的な事象を言う、純粋な言い切りの動詞表現をあまり使いません。ちなみに、前の文で使った「物言い」「言い切り」は、複合動詞の連用形を名詞にした言い方です。日本語の構文は、動詞が文の最後にきます。動詞の前に状態を説明する語を使いますので、文が長くなると、主語と述語とが対応しないことも起こります。書き言葉には、動詞なしの体言止めも普通に見られます。状態を表す動詞の代表は、英語では be 動詞です。口語でこれに当たるのが「だ、です、である、～ます、～ている」と考えることができます。古くは「～て居る」と当てました。前三つは名詞を受け、「我輩は猫である」の使い方が当たります。「です・ます」は、丁寧語の言い方です。「ます」は、口語文法では助動詞です。

4.3.2　「だ」は助動詞であるのか？

文法を研究して品詞分類にこだわると、文字並びを切り刻んで、最小文字並びに、なにかの品詞を当てはめたくなります。口語文法では、「だ」を助動詞に分類しています。異論も、したがって幾つかあります。英語の be と have は、受身形や完了形にするときには、他の動詞と組み合わせた使い方をします。英語の辞書を見ると、動詞と助動詞と、品詞が二つ書いてあります。単語の使われ方の区別で品詞の種類を分けています。「こういう使い方（形）をするときは、この意味になる」と割り切ります。「～だ」「です」は言い切りの形になっていますので、機能は全体が動詞です。「だ」の後に体言を付ける言い方がありませんので、助動詞に分類したのでしょう。過去形は「た」を語尾につけた「だった」「でした」です。「～である」は、「ある」が動詞です。古くは「在る」を当てることがあり（有り）ます。理屈を考えれば複合動詞です。終止形と連体形とは同じ語尾ですので、「～である」は後に体言を繋ぐ言い方ができます。「～である」を取ってしまうと体言止めの文です。そうすると「～で」は連用形の語尾変化に相当します。動詞としてみれば、語幹も変化しますので、「する」と同じような変格動詞です。語幹に漢字を当てることができません。「くる」は語幹に「来」を当てることもしますが、活用で漢字の読みが変わるのを避けるために、仮名書きするのがよいでしょう。謎解きのようになりましたが、「です」は複合動詞であって、「で」が連用形、「す」はサ変動詞の終止形です。「～であります」とも言えます。これは三つの動詞の複合形です。「だ」「だった」は名詞を受ける単独の動詞の使い方ですので、be 動詞を単独に動詞として使う言い方に当たります。「た」は過去形を作りますが、「だた」を音便で「だった」と使います。

4.3.3　別の動詞を受ける使い方

「～ます、～ている」の二つは、前に置く動詞を連用形で繋ぎます。ただし、「～ている」のときは音便変化を起こします。外国人が日本語を覚えるときの難関の一つです。「～ます」は丁寧語にする動詞です。単に動詞の言い切りで文を終えても意味は大体同じです。例えば、「夏山を歩く」「夏山を歩きます」「夏山を歩いています」の微妙な言い方の違いを、意味的にも文法的にも説明したいところです。過去形の言い方は「夏山を歩いた」「夏山を歩きました」「夏山を歩いていました」になります。うがち（穿ち）過ぎるかもしれませんが、「た」を添えることは、英語で完了形を作る時に have を使うことと対応しています。ただし、語順は違います。このように並べると、「～ています」「～ていました」が英語の進行形に似た言い方であり、静的な情景描写です。ただし、動詞の「読む」を使うと、音便で「読んでいます」「読んでいました」になります。このときの「て」と「で」の機能は、上の項で説明した「だ」の連用形です。英語の進行形は be 動詞に現在分詞を繋ぎますので、「て」が be 動詞に当たっています。言い切りの動詞部分が「いる」ですので、三つの動詞の複合動詞になります。「～ます」「～ました」は、単純な、現在時制と過去時制に当たりますが、習慣的な動作を説明する言い方です。言い切りの表現「歩く」「歩いた」は、名詞句の感じを受け、連体形を使った言い切りと見ることもできます。日本語では、「見て、聞いて、食べて、遊んで、暮らして、（居）います」のように「て」で区切って並べる言い方があります。これは「見ています。聞いています。…」のように、複合動詞を並べた言い切りの文の連続を詰めたものです。「て」を be 動詞的な連用形と見ると、最初の文形は、名詞化した連用形を並べた形になっていて、最後の「います」を修飾した一つの文です。英語に直すならば、「I am doing to look, to hear, to eat, to play and to live」になるでしょうか。

4.4　状態の違いを表す言い方

4.4.1　「相」と言う用語はあまり使わない

もともと言語学は輸入学問ですので、英語の文法用語 aspect は、カタカナ語にして使うか、漢字用語の「相」を当てました。言語学の参考書では幾つかの説明があります。よく分からないのですが、意義は、動詞が表す時間的な状態を言うようです。これを具体的に言うとき「現在・過去・完了・未来・進行形」の動詞形で区別します。英語は、過去形と完了形で書き方の区別があります。英語の完了形の文、例えば「I have been in Tokyo」を「東京に行ったことがある」と訳すのだと習います。意味は、「過去の或る時点で、東京に行き、或る期間そこにいて（継続)、今は居ない」という時間経過と経験的事実を言います。これが、この英文の完了形の「相」です。日本語では過去形と完了形の区別がありませんので、相の用語自体もほとんど使いません。この理由には、前節の始めで説明したように、動詞が表す動作は、主に静的な描写で説明するように使うからです。

4.4.2　日本語動詞の「時制」は未然形と已然形の区別がある

日本語文の時制表現は、過去とそうでない場合（非過去）の二つです。しかし、時間の捉え方からみると、動詞に三通りの区別があります。「動作をしない状態、している状態、済んだ状態」です。動作をしない状態とは、「未だしていないが、これからするかもしれない」ので、「いまだしからず」の訓読みを持つ**未然**形の活用があります。否定形を作る「ない」は、未然形に繋ぎます。している状態は**終止**形です。用語として終止は意味的には正しくないのですが、文の終わり（止め）に使いますので、従来からの用語の約束です。主語を立てるとき、「～は」と付けると、事象の状態を静的に言うか、近未来を言うか、習慣的な動作を表します。「～が」を使うと、その時点で動的な状態を説明する言い方が強く出ます。済んだ状態の時制は過去です。終わったと言う意味の漢語の已を使って**已然形**を当てます。口語文法では**仮定形**に言い換えています。「～れば」「～たら」と繋ぐ条件文に使います。時制的な見方をすると、過去に何かが起きた、または前に決まっていたことを条件としますので、意義的には過去時制の解釈が当たります。英語の if を使う条件文も、動詞は過去形にします。

4.4.3　自動詞・他動詞の区別が重要である

日本語の動詞は、**受身**・**使役**・**可能**の状態変化を区別する形があります。形を使い分けるとき、語幹に同じ漢字を使い、送り仮名で区別しますので、活用語尾だけを見ると複雑です。しかし、活用形の違う独立の動詞に分類すると、規則性が分かります。言葉を話すとき、自動詞か、他動詞かの区別を何で判断しているかは微妙です。自・他の区別は、動詞の語幹に使う漢字から判断できることがあります。自動詞に使う漢字（行く、走る、咲く、照る、など)、他動詞に使う漢字（取る、送る、読む、書く、など)、そして、どちらにも使う漢字（曲がる・曲げる、落ちる・落とす、など）があります。同じ漢字を使いますが、(見る・見える、分かる・分ける）は意義的には別動詞です。できれば、別の漢字を当てたいところです。現状では「視る、判る」と使うのは常用漢字の使い方に違反します。

4.4.4　「態」「形」を使う用語の方が分かり易い

日本語の動詞は、英語の文法用語 voice を訳した「態」を付けて、状態を区別する分類を使う方が普通です。能動態・受動態などの言い方です。用語としては硬いので、「形」も使わない「受身」(和製漢語で訓読みします)のように使うのが一般的です。動詞の態は、**自動詞**と**他動詞**とで機能が違ってきます。日本語は、フランス語やドイツ語にあるような、人称、性別で名詞と動詞の使い方を区別しません。実は、主語が人や動物のような**生物**（通常は人）であるときと、**無生物**であるときで言い方に違いがあります。これを便宜的に**人称**としておきます。無生物を主語に立てて自動詞を使うとき、論理的には、受身・使役・可能の態がありません。無生物を主語に立てて他動詞を使うとき、使役の言い方も、論理的には有りません。使役の受身形は作れます。動詞用漢字が同じであっても、自動詞と他動詞の使い分けがあるものは、別の動詞と見なければならないことがあります。自動詞の使役形が、同じ漢字を使う他動詞と紛らわしいことが問題になるのです。例えば「朝日が照る（自動詞)」と「朝日が山を照らす（他動詞)」があります。もし自動詞「照る」の使役形を作るならば「照らせる」だからです。「雨が降る（自動詞)」と「何かが雨を（人または物に）降らす（他動詞)」があります。自動詞の使役形は「降らせる」です。「茂る」は、主語に植物を考えて生物扱いをすると、使役形の「茂らせる」は、許容できる言い方としてもよいでしょう。この論理的な使い分けは、書き手や話し手の感性で決まります。聞き手や読み手が、この違いが分かることも感性に依存しています。

4.4.5　自動詞の受身形が無い理由

自動詞は、主語自体が何かの動作をすることを言う言葉ですので、動作相手、つまり目的語を使いません。したがって、相手から同じ動作を返される言い方、つまり、受身形がありません。ところが、欧米語では再帰動詞(reflexive verb)の形があって、主語の動作の対象を主語に向ける他動詞的な言い方、例えば「I enjoy myself」があります。日本語では自動詞で言う言い方が、英語では再帰的に他動詞を使って受身形で使うこともあることに違和感を持ちます。これは、英語の動詞（例えば move）は、自動詞にも他動詞にも使いますので、動作の対象を明らかにしたいときの言い方に現れます。話は少し変わりますが、コンピュータのプログラミングの技法として、**再帰的プログラミング** (recursive programming) があります。或るサブプログラムの中で、自分自身のサブプログラム名を呼ぶように作る技法です。欧米人の感覚では、素直に理解できるようですが、日本人には発想できない構文です。

4.4.6　丁寧に言うとき自動詞の受身形がある

自動詞は、「～を」でつなぐ目的語を取りません。しかし、日本語では、自動詞を受身形で使う場面が二つ有ります。まず、丁寧な物言いには、自動詞を受身形で使います。例えば、「行く」は、相手（人）が行く（自動詞）動作を丁寧に言う「行かれる」と受身形を使います。「（人が）本を読む」は他動詞です。目的語を主語にした「本が読まれる」は文法的に合います。これも「雅子さまが本を読まれる」のような、文法的には不合理な受身形も使います。この言い方は、主語に人を立てるときにだけ例外的に使いますので、論文などの文章に現れることはありません。二番目、自動詞の受身形には、主語が人であるとき、迷惑の受身があります。「雨に降られる」「釣った魚に逃げられる」などです。「紅葉が夕陽に照らされる」のような文学的な言い方もあります。もう一つ、日本語の自動詞の言い方に、例えば「道を走る」があります。一見すると、「道」に「を」を付けますので目的語と誤解します。紛らわしいのですが、「走る」が自動詞ですので、こちらの「を」の使い方は補語です。したがって「この道は自動車が走られる」の受身形は文法的には違反した言い方です。「この道は自動車が走れる」は、**可能**の言い方ですが、無生物を主語としていますので、これもおかしいと感じるべき言い方です。理屈からは、「自動車が走ること、ができる」です。尤も、その自動車を運転している人がいますので、レトリック的に、自動車を擬人化した文学的な言い方と大目に見ることもできます。

4.4.7　使役は自分に代わって誰かにしてもらうこと

使役の動作を表す「せる・させる」を付ける元の文は、主語が必ず生物です。主語（人）の動作を別の人にしてもらうことを、主語が言う言い方が使役です。無生物が主語であれば、使役形の態はありません。英語では let, make, have を**使役動詞**に分類することをします。文語的な言い方に「～をして～だれそれに～をせしむ」があります。この文は、第三者が間に居る言い方です。直接に動作を促すことが命令です。使役は、結果的には命令ですが、間接的な状態描写を言う静的な言い方です。「歩かせる・読ませる」と言うとき、相手がその動作をするか・しないかの自由度があります。日本語動詞の活用形に、強制の意味を持つ命令形がありますが、単独で他の活用形と区別できませんので、言い方を工夫して命令の意思を伝えます。現実に人と人が接する修羅場では、間接的な物言いは、誤って相手に伝わることがあります。命令形の言い方は英語では、主語を省いて動詞を文頭に置くことで命令文ができます。主語を立てて助動詞の shall を使う言い方もあります。

4.4.8　可能形は主語が生物のときに使う

可能の言い方を正確に言いたいとき、「～することができる」と使います。「する・しない」は、意思で選択しますので、主語は生物です。「スル名詞」または動詞の連用形を名詞で使う形を動詞に使うとき「～ができる」と使います。「できる」は自動詞です。「～が」を取る語は補語です。英語では助動詞 can に動詞を繋ぎます。「が」を省く言い方は舌足らずです。和語の動詞は、已然形に「る」を付けて可能に意味を表すことができますので、これを**可能動詞**として独立に扱うこともします。しかし、態の一つの形とする方が分かり易いでしょう。ここまでの説明を、表 4.4 にまとめます。

表 4.4　主語の「無生物・生物」の区別と「自動詞・他動詞」の区別で取りうる態

主語	自動詞				他動詞			
人称	受身	使役	可能	使役＋受身	受身	使役	可能	使役＋受身
無生物	×	×	×	×	○	×	×	×
生物	○	○	○	○	○	○	○	?

4.5 「態」の違いの表し方

4.5.1 送り仮名で助動詞の役目をさせる言い方

「受身・使役・可能」の状態を言い分けるとき、日本語の動詞は送り仮名で区別し、語幹の漢字は同じですが、活用形の違う別の動詞形にします。英語では「助動詞 be・動詞 let・助動詞 can」を使う構文に相当します。口語文法では、送り仮名の「(さ) せる」「(ら) れる」は、状態の区別に使う助動詞に分類しています。筆者は、品詞に細分することにこだわるよりも、送り仮名を含めた全体として動詞で扱う方がよいと考えています。自動詞は受身形が無いと説明しましたが、物理的には受身形を作れます。無いと言うのは、そのように使うのは論理的に成り立たないと言うことです。この使い分けは、書き手、または話し手の感性で判断することです。コンピュータに電子化した文書を理解させようとするとき、文学的で感性に関わる正誤の判断はできません。学術論文では、曖昧な表現を使わないようにしますので、コンピュータが判断できる文章解読の規則は必要です。この仕分けには、主語、目的語の性別に相当する生物・無生物の分類も必要です。

4.5.2 送り仮名の付け方で態を決まる

一つの動詞が表現する状態の全体が活用です。状態は、時制（現在・過去など）と態（受身・使役・可能）に大きく分けます。その中をさらに細分類するときに活用形を使います。日本語動詞の時制は過去と非過去の二つですし、過去形の作り方も単純です。動詞の活用形には未然、已然の区別がありますが、時制を区別する言い方とは関係しません。動詞の使われ方（態）は、語幹に繋ぐ送り仮名で決まります。送り仮名は多様な文字並びの組み合わせですので、それをさらに文法的に細分して説明することもします。「(さ) せる」「(ら) れる」は助動詞である、と言うのがそうです。実際に動詞を使うとき、ここまでが動詞で、ここから助動詞である、のような区別をしないで、繋いだ全体を一単位の動詞として扱います。一つの動詞の基本形から出発して、幾つかの態に使い分けると、動詞は別の顔を持ちます。元の基本形の動詞を含めた一つグループを構成すると考えるのがよいと筆者は思っています。可能動詞と言う分類を立て、これだけを別動詞扱いとする方法を取りません。このとき、同じ動詞用漢字を使っていても、別のグループになることがあります。その例を、幾つか例示します。このとき、主語の「生物・無生物」、動詞の「自動詞・他動詞」の違いで、論理的に使わない態があります（表 4.4)。「照る・照らす」「走る・走らす」は動詞用漢字が同じですが、動詞の意義から見れば別の動詞であって、それぞれ、自動詞と他動詞の対ですので、比較の例題としました。

4.5.3 「照る・照らす」の使い分け

文例として「朝日が照る」の自動詞を使う文と「朝日が山を照らす」の他動詞が作る文を比較して、下の表 4.5.1～表 4.5.3 の態の違いによる活用形を見て下さい。送り仮名に「ら」を含む動詞ですので、意図的に例題にしました。この主語の朝日は無生物です。したがって、自動詞「照る」は基本形の使い方だけしかありません。表 4.5.1 は、物理的に語尾変化をさせて受身・使役・可能などを表したのですが、論理的にこの態はありません。一方、「照らす」は他動詞ですので、主語に人を立てて「人が物を光で照らす」と使うことができます。ここでは主語を生物と無生物に分けたことで、使える態と使えない態の区別が出ます。無生物は意思を持ちません。他動詞の「照らす」は、途中に壁があれば、照らすことができませんので、可能態の出番があります。これが「可能動詞」の使い方です。「照る」の使役形と同じですが、主語が無生物ですので、自動詞の使役ではないことが演繹されます。「使役＋受身」は、幾らかくどくなりますので、表 4.5.2 では「？」を付けました。「可能＋使役」「可能＋受身」の送り仮名構成も、物理的にこの言い方ができるにしても、意味が限定できませんので、使いません。

表 4.5.1 語幹の漢字が同じで態の違う使い方（主語が無生物・自動詞・照るの場合）

自動詞	語幹と（送り仮名）	活用種別	活用形（未然・連用・終止・連体・仮定）	
（基本形）	照（る）	五段活用	ら、り、る、る、れ	○
受身	照ら（れる）	下一段活用	れ、れ、れる、れる、れれ、	×
使役	**照ら（せる）**	下一段活用	せ、せ、せる、せる、せれ	**×**
可能	照（れる）	下一段活用	れ、れ、れる、れる、れれ	×
使役＋受身	照らせ（られる）	下一段活用	られ、られ、られる、られる、られれ	×
可能＋使役	照れ（させる）	下一段活用	させ、させ、させる、させる、させれ	×
可能＋受身	照れ（られる）	下一段活用	られ、られ、られる、られる、られれ、	×

表 4.5.2　語幹の漢字が同じで態の違う使い方（主語が無生物・他動詞・照らすの場合）

他動詞	語幹と（送り仮名）	活用種別	活用形（未然・連用・終止・連体・仮定）	
（基本形）	照ら（す）	五段活用	さ、し、す、す、せ	○
受身	照らさ（れる）	下一段活用	れ、れ、れる、れる、れれ、	○
使役	照らさ（せる）	下一段活用	せ、せ、せる、せる、せれ	×
可能	**照ら（せる）**	下一段活用	せ、せ、せる、せる、せれ	×

表 4.5.3　語幹の漢字が同じで態の違う使い方（主語が人・他動詞・照らすの場合）

他動詞	語幹と（送り仮名）	活用種別	活用形（未然・連用・終止・連体・仮定）	
（基本形）	照ら（す）	五段活用	さ、し、す、す、せ	○
受身	照らさ（れる）	下一段活用	れ、れ、れる、れる、れれ、	○
使役	照らさ（せる）	下一段活用	せ、せ、せる、せる、せれ	○
可能	**照ら（せる）**	下一段活用	せ、せ、せる、せる、せれ	○
使役＋受身	照らさせ（られる）	下一段活用	られ、られ、られる、られる、られれ	？

4.5.4　「走る・走らす」の使い分け

主語が人であるとき、走る・走らすの自動詞と他動詞の対は、すべての態があります。この例も、送り仮名に「ら」を使う動詞ですので、意図的に例題にしました。可能形の作り方は、五段活用の動詞は已然形に「る」を付け、それ以外は「られる」を付ける違いがあります。「れる・られる」は受身を作る助動詞ですが、可能を作る助動詞の使い方があるので混乱します。自動詞「走る」の可能形は、已然形に「れる」を付けた言い方「走れれる」と言っても通じます。したがって、「走れる」を別動詞として立てて、慣用の可能の言い方は、可能動詞形を使うことにしたのです。自動詞「走る」の使役形は、機能的には他動詞の「走らす」と同じです。助動詞の「(ら）れる」は、受身と可能を作る二つの機能がありますので、「走らせ（られる)」と「走らせ（れる）の言い方が、表 4.6.1 と表 4.6.2 にあります。この混乱を避けるには、「（ら）れる」を可能態に使うことを制限した言い換えがよいでしょう。それは、文字数が増えますが、「走ることができる」「走らすことができる」です。もう一つ、「走らす」の使役形「走らさせる」を不思議な言い方であることに気が付くでしょうか？　機能としては使役の言い方がダブっています。と言うことは、ひるがえって、「走らす」の動詞を使うまでもないと決めても良いのです。同じように、「書く・書かす」「聞く・聞かす」「取る・取らす」などは、「書く」「聞く」「取る」で済ますことができます。英語には使役動詞の区分を使います。日本語では「書かす」「聞かす」「取らす」は使役の意義がありますので、使役動詞と言えなくもないのですが、可能動詞ほどには使い勝手が良くなりませんので、これらの動詞形を使わないようにすると文が判り易くなります。

表 4.6.1　語幹の漢字が同じで態の違う使い方（主語が人・自動詞・走るの場合）

自動詞	語幹と（送り仮名）	活用種別	活用形（未然・連用・終止・連体・仮定）	
（基本形）	走（る）	五段活用	ら、り、る、る、れ	○
受身	走ら（れる）	下一段活用	れ、れ、れる、れる、れれ、	○
使役	走ら（せる）	下一段活用	せ、せ、せる、せる、せれ	○
可能	走（れる）	下一段活用	れ、れ、れる、れる、れれ	○
使役＋受身	走らせ（られる）	下一段活用	られ、られ、られる、られる、られれ	○

表 4.6.2　語幹の漢字が同じで態の違う使い方（主語が人・他動詞・走らすの場合）

他動詞	語幹と（送り仮名）	活用種別	活用形（未然・連用・終止・連体・仮定）	
（基本形）	走ら（す）	五段活用	さ、し、す、す、せ	○
受身	走らさ（れる）	下一段活用	れ、れ、れる、れる、れれ、	○
使役	走らさ（せる）	下一段活用	せ、せ、せる、せる、せれ	？
可能	走らせ（れる）	下一段活用	れ、れ、れる、れる、れれ	？
使役＋受身	走らさせ（られる）	下一段活用	られ、られ、られる、られる、られれ	？

5.　形容詞・副詞・助詞の話し

5.1　修飾に使う語の分類

5.1.1　品詞分類法は恣意的な手段であること

日本語の言葉並びの構成を分類するとき、品詞に分けることは、学問的な手段として定着しているように見えます。**品詞**の用語は、英語の a part of speech を訳したものです。英語の場合、分かち書きをした語の単位（単語、word）の機能を分類するときに便利です。品詞分類の考え方は歴史が浅く、そのため、学術用語も説明的です。もし学術用語化するのであれば、speech-part→speechpart のような集合名詞になるかも知れません。品詞分類法には幾つかの説があります。英語の大枠は８品詞（名詞・代名詞・動詞・形容詞・副詞・前置詞・接続詞・感動詞）です。日本語の場合、分かち書きをしない文字表記ですので、単語に区切る方法を最初に考えて、それから品詞に分類します。生物の種を分類する方法は客観性があります。しかし、言葉の品詞分類法は、研究者の恣意的な（思い付きの）考え方です。誰もが納得している方法ではありません。文の中で最も明確に分類できる品詞は名詞です。しかし、動詞や形容詞から作られて名詞的に使っている語や、修飾語を含めた長い名詞句も見られます。文字並びを、細かく分割して最小の品詞要素に分ける方法を考えるとき、活用のある語は、変化しない語幹部分と送り仮名とがあるので厄介ですが、これを１語に分けます。機能が良くわからない文字並びには、**助詞**と言う品詞、例えば（が・の・に・を）を立てます。この章は、文中の使い方で、ある文字並びが、機能で見て、形容詞になるか副詞になる、と言う見方で説明します。

5.1.2　形容動詞とは不思議な分類名であること

大枠的な品詞名として、形容詞と副詞の、機能としての違いは、形容詞は名詞を、副詞は動詞を修飾する語と区別します。語幹部分が同じで、修飾対象が名詞か動詞かの違いで語尾が違う語があります。英語では「beautiful, beautifully」、日本語では「美しい、美しく」のように使い分けます。日本語の（美しい）は形容詞に分類しますので、用言（動詞・形容詞）を修飾する副詞としての使い方（美しく）の形を形容詞の連用形とします。「静かな、静かに」の対も修飾対象で使い分けますが、「静か」を学校文法では形容動詞と命名しています。なぜ、この分類を立てたかを推測すると、動詞の活用形との類似を当てたからと思います。「美しい」は、見かけ上、連体形と終止形とが同形です。「静かな」は文末の終止形に使うときは「静かだ」とするからです。外国人に日本語を教えるときに、形容詞が動詞なみに活用すること自体が論理的に理解できないので、「**イ形容詞**（例えば；美しい)」と「**ナ形容詞**（例えば；静かな)」の分類を使います。「ナニ名詞」は、「～な」「～に」と付け替えるだけで形容詞と副詞の使い方になる名詞です。「な」と「に」の品詞は何か、と問われると困ります。学校文法では、形容動詞の活用語尾としています。「～だ」と付ける活用語尾を英語の be 動詞の機能と同じと見ると（第 4.3 節参照)、形容動詞の品詞分けを必要としません。ここで、困ったことが起こります。ナ形容詞の語幹「静か」に「だ」を付けると、文の終止形になります。イ形容詞の語幹「美し」に直接「だ」が付きません。そのため、「美しいです」の不思議な言い方も、何となく許容されています。これは終止形の言い方二つ繋がっているためである、と意識できるでしょうか？　筆者は「美しいのです」と使います。

5.1.3　和語の形容詞の語幹に当てる漢字がある

日本語（和語）の形容詞の表記法は、前章で説明した動詞と同じように、語幹に漢字を当て、活用部分に送り仮名を使います。語幹に当てる漢字は、中国語でも同じ意味か、それに近い漢字です。しかし、その読みは、中国語とは全く別の訓読みです。語幹の訓読みに当てる形容詞用漢字（この言い方は筆者の造語です）の数は、常用漢字で読み方を決めているもので約 140 字あります。この内、ナ形容詞（形容動詞）として使う字数は約 10%です（表 5.1、表 5.2)。二字を並べて音読みのナニ名詞に構成する漢字は、形容詞用漢字を組み合わせる方法で造語されます。常用漢字で、訓読みを使わない約束の漢字を、表 5.3 に示します。尤も、なかには、慣用的に訓読みで使う漢字もあります。漢字二字を組み合わせた「ナニ名詞」の熟語は非常に数が多く、語彙数を数えたことはありませんが、数百はあると思います。試しに幾つかの国語辞書で「優美」を引いてみて下さい。①名詞・形容動詞と品詞説明があるもの（角川：新国語)、②「～な、～に」の注記があるもの（三省堂：新明解)、③何も書いてないもの（岩波：広辞苑)、など、様々です。コンピュータ用の辞書を作るならば、「ナニ名詞」の品詞項目を立てるのがよいと思います。

悪	安	暗	偉	易	永	鋭	円	煙	遠
汚	温	快	怪	悔	懐	楽	寒	甘	緩
丸	危	忌	貴	久	強	恐	恭	狂	狭
近	苦	軽	激	潔	賢	険	厳	古	固
厚	広	硬	荒	高	黒	細	酸	若	寂
弱	臭	醜	柔	渋	重	暑	小	少	詳
新	深	親	辛	甚	正	清	青	惜	赤
浅	善	疎	粗	早	憎	速	尊	多	太
大	嘆	淡	短	暖	恥	遅	著	長	珍
痛	低	等	鈍	軟	難	熱	濃	白	薄
煩	卑	悲	美	貧	怖	芳	乏	忙	明
優	勇	幼	欲	涼	良	冷	麗		

（読みは省きます）

表 5.1　イ形容詞の語幹に使う訓読み漢字

哀	穏	華	確	滑	愚	健	嫌	巧	幸
懇	惨	静	鮮	平	豊	朗	和		

語幹に当てる訓読み（参考）
アワれ、オダやか、ハナやか、タシか、
ナメらか、オロか、スコやか、イヤ、
タクミ、サイワい・シアワセ、
ネンゴロ、ミジめ、シズか、アザやか、
タイら、ユタか、ホガらか、ナゴやか

表 5.2　ナ形容詞の語幹に使う訓読み漢字

英	佳	寡	概	完	寛	敢	簡	閑	頑
奇	宜	巨	虚	凶	緊	堅	謙	顕	康
剛	豪	酷	雑	邪	淑	俊	純	順	徐
昭	冗	迅	斉	拙	塑	壮	俗	妥	惰
駄	泰	稚	貞	漠	般	微	敏	普	遍
朴	凡	慢	漫	密	妄	猛	愉	唯	幽
悠	隆	累	烈	廉					

（読みは省きます）

表 5.3　音読みでナニ名詞用熟語に使う漢字

5.1.4　「～的」を助辞として使うこと

和語の「て・に・を・は」などは助詞に分類し、仮名で使います。中国語で同じような使い方をする漢字を助辞（助字）と言います。「的」の字は、日本に中国人留学生などが増え、中国語の情報が多くなった 1980 年代以降、眼にすることが多くなりました。その使い方は、和語の助詞「の」とほぼ同じで、例えば「日本的習慣」のように使います。漢字の「的」は、訓読みで「まと」と読み、その意義で使う「目的」などの熟語に使うことが本義です。中国語風の使い方は、例えば「科学的」のように、やや硬い文章表現に使われていました。日本語の語感では「日本の習慣」と言うのが軟らかです。「的」を音で「テキ」と読ませると、英語の romantic のような形容詞語尾にある -tic と似た語感に通じますので、このように造語すると形容詞的な意味に取るようになりました。「～的」とすると「ナニ名詞化」し、ここに「日本<u>的な</u>習慣」の言い方が受け入れられるようになりました。「的」の解説に、国字の使い方として一部の漢和辞典（角川）にも載るようになりました。中国語の語感で言えば、「の」の意義が二つ並びます。

5.1.5　口調はよいが曖昧さのある言い方

「～の」は、名詞を形容詞の機能に変えて、後の名詞を修飾するときの日本語の助詞です。「～な」「～に」は、形容詞と副詞の機能に変える仮名ですので、それと同列に扱う接尾辞と見ることができます。しかし日本語文法では、助詞に分類していません。「日本の習慣」「日本的習慣」「日本的な習慣」と並べてみると、語感として微妙な違いがあることに気が付きます。最初の二つの言い方は、ほぼ同じに理解されます。「日本的な習慣」のように「的な」を使う表現には微妙なニュアンスがあって、「似ているが、すこし違うところがある」意味を持ちます。意味としては「日本風の習慣」「日本流の習慣」と同じです。つまり、「的な」を多用すると、曖昧さが残りますので、論理の通った文章を作成するときには注意しなければならないでしょう。

5.2　形態素解析の位置づけ

5.2.1　形態素を意識するか・しないか

言葉は、個別の用語（名詞、動詞など）に説明を補う（修飾する）語句を付けると、意味がはっきりします。修飾語は、意識して形容詞・副詞の品詞名を付けて分類します。日本語文は、助詞を付けることで、語句全体の機能と目的とをはっきりさせることができます。助詞は、英語との対応を考えると、機能は前置詞(preposition)と似ています。しかし、位置が逆になりますので、後置詞(postposition)と言うことがあります。主語・述語・目的語の並び順は、助詞を付けることで、英語や中国語に較べてある程度の自由度があります。文字並びから意味を理解するには、意味の最小単位の語（**形態素**：morpheme）に分けることと、その組み合わせで構成する最小の語単位（単語:word）に分けます。形容詞と副詞とは、語形が変化しない部分（語幹）と送り仮名とに分けることができます。そうすると、例えば「美しい・美しく」は、「美し」が語幹で、「～い」「～く」と付け替えることで、機能は形容詞と副詞の使い分けができます。「美し」を意味単位の一つにすることができます。そうすると、「い」「く」は何か？の質問に対して説明に困ります。そこで、形容詞も活用する語（用言）であるとしたのが日本語の品詞分類法です。つまり、「い」「く」は活用語尾であるとしたのです。品詞としての形容詞が、機能として副詞にも名詞にもなるのは困ります。そこで、「い」「く」は、品詞に構成する形態素である、と説明するようになりました。これは、言語学の用語ですので、文法書では使いません。しかし、形容詞・副詞・助詞の話しをするとき、形態素の説明を避けて通ることができません。

5.2.2　形態素解析が現れた経緯

品詞分類法は、辞書を作るときなど、文字並びで意味のある言葉の単位に分けるとき、自然に必要となった方法です。日本語の表記方法は漢字と仮名を組み合わせますが、分かち書きをしません。英文のような、最初から語単位（word）に分けてある言語表記とは違いますので、単語に切り分ける方法を先に考えて品詞を決めます。大局的に見ると、規則性、つまり文法があって、形容詞と副詞は、修飾対象（被修飾語句）の前に置き、助詞は後に付ける語（後置詞）です。動詞と形容詞は活用語尾が変わりますが、それを含めて品詞に分類しています。これが混乱の始まりです。分かち書きをしないで仮名文字だけを並べると、語の切れ目が分からなくなり、別の意味に取られることが起こります。仮名文字並びの言葉遊びは昔からありました。話し言葉を耳で聞いているときは、単語の切れ目、意味上の切れ目などの僅かな息継ぎやイントネーションを判断して、意味単位を正確に理解します。尤も、東北弁では「サ行」が「さ・す・す・せ・そ」と聞こえ、東京弁は「ひ」を「し」で発音することで取り違えが起こることがあります。現実には仮名文字だけで実用文を作ることはないのですが、電報文が例外的に使われていました。良く引用される例には「カネオクレタノム」を「金送れ、頼む」の意が「金をくれた、飲む」と判断される笑い話です。仮名文字並びから漢字交じりの意味ある文字列に直すことは、日本語ワープロを開発するときの問題の一つとして、長尾真(1936-)によって、1980 年代前半から新しく形態素解析の研究が始まりました。

5.2.3　表記と発音を区別する

形態素の説明の前に、文字の書き方についての説明が必要です。言葉は基本的に音ですが、それを文字で表す規則が、書き方（**表記**）であり、それを声で再現することが言い方（**発音**）です。表記と発音は、言語ごとに対応の規則があります。細かくみれば、１対１に対応しません。アルファベットは表音文字ですので、かなり正確な書き方ができます。しかし、例えば、フランス語やドイツ語ではアクセント記号付きの母音文字を余分に使い、普通の母音文字と発声が微妙に違うことを反映するように書き（表記し）ます。英語は、この面倒な方法を使いませんので、逆に、読み方を個別に覚えなければなりません。アメリカ英語は、イギリス英語よりも簡単な表記が増えています。日本語では仮名が表音文字ですので、仮名を使えば、一応、正確に発声を文字で表すことができますが、実用的には表記の規則（かなづかい）を決めてあります。主語を立てる「～は」、目的語を表す「～を」などがそうです。漢字は、中国語の言い方を単純化した言い方に変化してきましたので、例えば、「過」は、以前「クァ」と振り仮名で表記したのを「カ」で済ますようになりました。漢字仮名交じりの文書を声に出して読み、その読みをすべて仮名文字で書いたものを、再度読むとなると、同音異義語が増え、読みから元の漢字を再現することができない問題が起こります。もう一つ、分かち書きしない仮名文字は、電報文の誤読の問題が起こります。この二つの問題の研究が、形態素解析の第一の課題です。

5.2.4　ワープロの作業環境と使い方があること

日本語のワープロを利用する最終目的は、誤字や当て字の無い文書の作成です。この作業のとき、ハードウエアとしてのパソコンを操作する技術、特にキーボードを見なくても正しい文字のタイピングができる**ブラインドタッチタイピング**技能(blind touch typing)の習熟と、ワープロのソフトウエアを使いこなす技能を覚えることが必須です。この作業を便利にする隠れたソフトウエアツールが、**仮名漢字変換**のソフトウエアです。このとき、二段階の文字変換処理が行われます。言葉は、頭の中で音として発想されます。その音を表音文字、日本語の場合は、仮名で表します。パソコンのキーボードを操作するとき、直接に仮名で入力する方法もありますが、アルファベット並びを利用したローマ字変換をして仮名に直す方法が普通になりました。これが第一段階です。この部分は、**フロントエンドプロセッサ**(front end processor)と言うことがあります。第二段階が、仮名文字並びから、適切な漢字を選択する仮名漢字変換です。作成された文字並びが正しいことを最終的に判断するのは、人の側です。ある程度まで知能を埋め込んだソフトが利用できるにしても、コンピュータが人間の能力を上回ることはありません。したがって、完全無欠さをコンピュータソフトに期待するのではなく、程ほどの機能で妥協しなければなりません。妥協の線引きをどこに置くかは、文書作成の環境と関わります。

5.2.5　ヘボン式と日本式

パソコンの操作でローマ字入力を使って仮名に変換するときは、日本語をローマ字で表記する方法と関係を持ちます。これには、**ヘボン式**と**日本式**とがあります。米国人のヘボン(J. C. Hepburn)が、耳で聞いた日本語の発声を、ローマ字で表記する方法として提案したので、ヘボン式と言います。日本式とは、仮名の 50 音表を物理的にアルファベット表記にする方法です。二つの方式の違いは、主に「シ」、「ジ」、「フ」 の扱いに現れます。「ハヒフヘホ」 をヘボン式では「ha, hi, **fu**, he, ho」、日本式は「ha, hi, **hu**, he, ho」と当てます。これは、いみじくも、発音と表記の関係の典型です。和英辞書にローマ字で日本語の見出しを使うときは、ヘボン式が普通です。専門用語の和英辞書には、日本式を採用しているものがあります。理論にこだわる学者は、日本式を提案する傾向があります。この方式は長母音にアクセント記号 ^ のついた母音文字が必要ですので、パソコンの入力には向きません。漢字に変換するとき、長母音「おー」は「おう」入力する（表記の）約束です。パソコンでは、fu, hu どちらの方式でキー入力をしても「フ」が出る設計です。50 音表のラ行は「ra, ri, ru, re, ro」を使います。パソコンでは「la, li, lu, le, lo」の入力は小文字の「ァィゥェォ」の文字に変換されます。ここまでの変換は、**コード変換**と言うものであって、１対１の変換表があれば逆向きの変換もできます。ただし、仮名文字からローマ字表記に直す方向の変換を使う場面はありません。

5.2.6　完全無欠な形態素解析を期待しない

仮名漢字変換は、上のコード変換とは質の違う変換です。１対１の対応ではなく、複数組のコード間の変換になるからです。漢字は、音読みと訓読みの区別があり、逆向き変換で見ると同音異義語の選択があります。どれとどれとが対応するかの判断は、最終的には人が決めます。それを助けるのが仮名漢字変換のソフトです。ある長さの仮名文字の並びを形態素であるとして、変換候補を絞り込むことができれば、変換の能率が上がります。人為的に形態素単位、例えば熟語単位、で変換候補を探すことが最も基本的な方法です。この章の表題の形容詞・副詞・助詞は、語幹と仮名の組み合わせがありますので、仮名部分の扱いが形態素解析で重要になるのです。しかし、一般のユーザがいつも形態素を意識した作業をすることは無く、一つの意味を構成する文字並び（文節）が繋がった連文節の仮名文字並びが、変換の対象です。そうであると、文字並びを形態素に分解する処理を先行させた、形態素解析が必要になります。平たく言えば、電報文を意味のある仮名漢字混じりの語に並び変えることです。実用的なワープロソフトは、多くの研究グループが競って製品を発表してきました。その成果が積み重なって、便利さは向上してきました。しかし同時に、欠点も指摘されるようになりました。ワープロソフトは、一般ユーザの多用な要求を満たすように努力が重ねられてきました。商業ソフトは、ソフトウエアの中身を公開しません。文書の校正は、最終的にはユーザの責任で行いますので、親切と思って組み込まれた人工知能(AI)の機能が、はた迷惑やお節介に過ぎることも起きます。AI 機能は、ワープロを使い込んでいくと、ユーザのクセを取り込んで、変換効率を上げるように、内部の組み込み辞書の用語選択順位が変わります。この機能はリセットができませんし、辞書の取替えもユーザレベルでは手が出ません。用語の質の違う別の文書を作成するとき、前に作成した文書の用語が優先される AI 機能は、手戻りの労力を増やすのです。要するに、ワープロの仮名漢字変換は、完全な自動化ができません。幾らか不便であることを納得の上で、ワープロと付き合う度量が必要です。

5.2.7　学術論文を書くときのマナーがあること

学術論文を書くときは、その専門に関係する学術用語を断りなしに使います。その用語は、一般の辞書には載っていないか、有っても用語の定義が異なりますので、その専門に関わる学会などで編集される学術用語集に従うのが普通です。そこに無い用語を作者が使うときは、その語が最初に現れた場所で定義と宣言をします。そして、ページ数の多い著作であれば、その用語索引を付録に付けます。啓蒙的な文書は、専門用語の解説を、用語集(glossary)として、同じく付録に付けます。専門に関わる文書をワープロで扱うときは、専門別の用語辞書（シソーラス：thesaurus）をユーザが取り替えできるか、選択順位を変更できるようにしておくカスタマイズの機能があると便利です。パソコンでワープロを使うユーザは、大部分が企業ですが、そこでは人名・地名を使う頻度が高くなるのが特徴です。固有名詞には常用漢字にない漢字があり、また、特殊な読みをすることがありますので、これに対応できるような大寸法の辞書が組み込まれています。専門分野では、或る限られた数の固有名詞しか使いませんので、専門ごとに特徴のある固有名詞が必要です。専門文書をワープロで作成する場面では、一般ユーザ向けの固有名詞用辞書を使わないようにしても不便にはなりません。しかし、参考文献を挙げて引用するときは、固有名詞が必要になるのですが、参考文献集を別作業で作成しておいて、それを利用するように対応できます。

5.2.8　文書の校正に使うときの辞書も要ること

ワープロは、文書を作成する道具です。文書の文字列だけの作成作業に特化したソフトウエアが、テキストエディタです。ワープロの使い方を覚える教育過程は、最初にテキストエディタの使い方から始めると効果的です。現在のワープロは、見栄えのよい印刷を作成することが最終目的です。文字並びについて、簡単な文書校正の機能も備えるようになってきました。英文では、spell check の機能がそうです。英文は単語単位で分かち書きをしますので、辞書の中の語彙に無い英字綴りは、誤りの可能性があることを警告表示してくれます。このとき、イギリス英語か、アメリカ英語かで、辞書の語彙が違います。例えば、「centre・center、harbour・harbor、programme・program」があります。日本語の場合、分かち書きをしませんので、普通には使うことがない仮名文字並びに、誤りの警告を出すのが spell check に当たるでしょう。専門用語を使うときは、同音意義語、または、間違え易い語彙の辞書を別に作成しておいて、そこにある語彙と一致すれば、誤用の可能性を警告させるようにできます。そうすると、例えば、「人工衛星」と書くべき個所が「人口衛生」となる誤用を指摘するには、「人口」と「衛生」とを間違え易い語彙の辞書の方に登録しておきます。差別用語の扱いも、同じ扱いができます。文書の作成作業と校正作業とでは辞書の種類も使い方も別になりますので、校正用のソフトウエアツールは別作業で利用するのがよいでしょう。校正作業は、漢字仮名交じりの文字列を対象とした形態素解析になります。このときの問題には、動詞や形容詞に適切な送り仮名を付けることがあります。

5.2.9　言葉の発声と文書記録との相互交替が重要である

紙に書かれた文書は、声に出して再現することが目的です。黙読をしていれば音声と関係が無いと考え易いのですが、実際は頭の中で音を再現して読んでいます。読みを助ける手掛かりは、句読点のような明瞭な記号と共に、意味のある語の区切りを眼で判断しています。音を文書化し、それから音を再現する例として、第 2 章図 2.1 で楽譜の紹介をしました。音符は、音単位の記号です。音符と音符の間を切って演奏することを基本とし、音を滑らかに繋ぐ記号にタイとスラーを使います。音を出さない指定が休止符です。歌曲で言葉を割り当てるときは１音節１音符ですが、単語の切れ目や息継ぎの個所は、タイとスラーの切れ目が当たり、文としての切れ目は終止符の個所です。文書は、音の高低や長短を文字化しない表記です。発声は標準的な速度がありますし、リズムに乗るような音節の繋がりがあると聞き易く、理解も助けます。表音文字のアルファベットを使う欧米語では、音の並びをそのまま文字に落とせば、ほぼ正確に言葉の記録が得られます。コンマやピリオドを補い、スペルを僅かに修正すれば実用的な文書の作成ができます。単語の分かち書きは、ほぼ形態素単位になります。分かち書きをしないと、読み難くなりますし、音声の再現のとき、別の意味単位になることがあります。欧米語のようにアルファベットを使う言語では、書き言葉と話し言葉の交替で意味が変わる確率は高くなりません。英文で書かれたテキストをコンピュータに読ませて発声させるソフトは、かなり以前から開発されていました。パソコンで利用する実用的なソフトは、1984 年、アップルコンピュータに搭載されたに MacInTalk が最初です。音から文字への変換は、音声認識の技術です。こちらは、物理的な装置の助けが別に必要ですので、文字並びを対象とした言語学の研究からは少し距離があります。

5.2.10　表意文字を使う言語表現は交替が難しい

日本語や中国語の文書表記には表意文字の漢字を使いますので、音を文字に落とすことと、文字を音声で言う読み方とを並列させて覚えることに難しさがあります。日本語では、表音文字の仮名を覚えれば、文書による最低限のコミュニケーション（例えば手紙の読み書きなど）ができます。漢字を使う限り、文書表記法と読み方との間に在る質の違いを埋める手段はありません。漢字廃止論は、ナショナリズムによる漢字文化圏からの独立の要求も背景にありましたが、鉛の活字を使う活版印刷で障害になっていたことが大きな理由でした。東芝が日本語のワードプロセッサ（ワープロ）を始めて開発した 1978 年以前、企業での必須の事務機械の一つは、邦文（和文）タイプライタでした（図 5.1）。文書の発想は音として頭の中で行われ、それを手書きにした原稿を見て、タイピストが操作します。タイピストは、文字を見てその音を再度頭の中で復元し、その音の順に並べられた漢字表で文字位置を確認し、レバーで活字を取り出してプラテンに打つ作業です。文字から音に変換する処理がここにあります。これは一文字単位での変換であって、訓読みの漢字も音の並びで検索しますので、単漢字変換と呼ばれるものです。熟語並びで変換する仮名漢字変換は、熟語に切り分けることと、読みを決めて辞書に当たりますので、人が漢字仮名混じり文を形態素解析する問題、と捉えることができます。

図 5.1　邦文タイプライタ(カタログから採図)

5.2.11　JIS の漢字コード系が決まった経緯がある

形容詞の説明からかなり脱線してきましたが、漢字コードについて、簡単な解説をしておきます。日本語の環境で使うコンピュータ用の漢字コードは、漢字の音読みの順に並べてあります。JIS の漢字コードが決まる前、印刷会社や新聞社では種々の２バイト系コードが使われていました。新聞社では、短時間で大量の鉛の活字を拾う植字作業が大変でしたので、活字列の自動鋳造機（タイプセッター）を使いました。これを制御する文字並びのデータは、特殊な漢字テレタイプライタ、通称で漢テレと呼ぶタイプライタで紙テープに鑽孔しました。キーボードの文字並びは、平面領域を「区」と「点」の名称で分け、一文字を紙テープの２バイトに鑽孔しました。これが用語としての漢字区点コードです。漢テレのタイピストは、原稿を見てタイピングするのですが、打ち間違えをその場でモニタできませんでした。作成した紙テープを自動鋳造機に掛けると、文字並びの順に活字が出てきますので、それを版に並べます。新聞社の中で、活字を鋳造する工場があったのです。試し摺りをするまでは、文字並びの確認ができません。校正作業も手が掛かりました。新聞社ごとに、コード系が少し違う方式であったこと、活版印刷を扱う印刷所、さらには写真を媒介とした写真植字で使う方式もあったこと、などの間で協議の結果、JIS コードが決まりました。この 2 バイト系コードは、通信用コードとして使うことを考えて１バイトの中の７ビットを有効情報としています。さらに、アルファベットなどの１バイト系コードと混在して使っても識別ができるようにすると、文字種として 4000 字程度までしか利用できません。常用漢字の範囲で漢字の利用を制限する分には十分です。固有名詞にあるような特別な漢字を扱うとなると、この字数では不足しますので、その対応を考えたコード系も工夫され、結果的に複数の漢字コード系が利用される混乱が続いています。

5.3　プログラミング言語の中の形容詞

5.3.1　形容詞と副詞は感覚を表す言葉であること

形容詞（ここでは副詞も含めます）は、物事の状態（形容）を説明するときに使う言葉です。その状態とは、大きく分けて二つあります。人の五感、主に眼と耳、を介して理解できる具体的な性質と、精神作用でなければ理解できない抽象的な性質です。人に代わって、コンピュータが形容詞で表される性質を理解できるかどうか、を考えてみると、二つの状態の区別が分かります。前者は数値を介して説明できますが、後者はそれができません。コンピュータは、数の四則演算をさせる道具として開発されたのですが、物事の性質を数に置き換えて処理する道具へと発展してきました。そうするには、コンピュータ側からみると、物事の性質を数の大小で扱う物理的な外部測定装置が必要です。画像処理を考えるとき、ビデオカメラとマイクロフォンが人の眼と耳に代わる装置です。デジタルカメラが取り込む数値情報は、被写体の形と色です。他の人に説明するときに言う形容詞は、和語の語幹の対で言うと、「大小、長短、高低、太細、多少、明暗、黒白」などです。程度の大小を区別する名詞は、「大きさ、長さ、高さ、明るさ」などですが、普通、言葉で言うときは、数値を添えるのではなく、相対的な比較の物言いをします。英語で形容詞を比較級・最上級を使う場面がそうです。デジタルカメラが取り込む形容詞的な情報は、数値化ができます。しかし、「美しい」の形容詞は、人の感情で判断する性質を言いますので、デジタルカメラで扱うことができません。この区別を、筆者は、**属性形容詞**と**感情形容詞**の用語で分けます。人の五感で理解する形容詞的な性質で、甘い・辛い・痛い、などは感覚形容詞と言うこともできますが、これは主観的な形容詞ですので、感情形容詞の方に含ませます。眼で見て理解できる形状を表現する大きい・長いなども感覚形容詞と言えなくもないのですが、こちらは客観的に理解できるので、属性形容詞として独立させることにしました。

5.3.2　論理変数で言い換える方法を使う

数値で言い換えができる形容詞的性質を、その物（object）の属性(attributive)と言います。属性に幾つかの段階があるとき、整数または実数を当てます。二通りの段階でしか区別できない形容詞の対があります。漢字の語幹の対で言えば「美醜、難易、善悪、尊卑」などを表す感情形容詞がそうです。言葉としては、「ない」を付けるか・付けないか「**肯定・否定**」の言い方をします。例えば「美しい」「美しくない」です。「美しくない」と「醜い」とは同義ではありません。反対でもありません。コンピュータでの表現法は、整数の(1,0)を当てる場合と、**論理変数**の対（**真偽**；true, false）を当てる場合とがあります。用語としての、**正誤**、**異同**の意義とも少し違います。数の大小で比較をするとき、同じである（等しい）ときと、どちらが大きいか・小さいか（異なる）に二通りありますので、数は三段階で比較をします。言葉としては「大・中・小」「上・中・下」「左・右・中央」「特・上・並」などがあります。数値で言うときは、「大きい・小さい・等しい」を「正・負」と０（ゼロ）で区別します。

5.3.3　論理計算の扱いが特殊になること

コンピュータは、四則演算の数値計算をさせる道具であると思い勝ちですが、論理演算ができることが大きな特徴です。電卓（電子式卓上計算機）は、論理計算のツールがありません。代数計算式は、数を文字や記号に置き換えて四則演算の手順を表記する方法です。式は、文章を元に記号化した書き方ですので、欧米人は、式を声に出して言うこともします。また、事務処理用の言語 COBOL は、プログラムの管理をコンピュータの専門家以外でもできることを意図して、英文的な表現でプログラムを作文できるように設計したものです。四則演算は記号式でも書くことができますが、ADD, SUBTRACT, MULTIPLY, SUBTRACT のキーワードが使えます。論理的な条件文を構成する IF, THEN, ELSE の語を使い、大小判定にも GREATER THAN, LESS THAN, EQUAL TO を使います。一方 FORTRAN は、数値計算の手順が、代数式の表現をコンピュータが直ぐに理解できるように設計したプログラミング言語です。四則演算の記号に「+,-,*,/」を当てますが、これは、英文キーボードで利用できる記号の範囲で間に合います。論理演算に関しては、ブール(George Boole,1815-1864)の名を冠した**ブール代数**があるのですが、演算記号を別に工夫しなければ文書に表せないこと、また、それが普通のキーボードの文字とは違うこと、とがあって、表記方法に工夫が必要でした。AND, OR, NOT は、そのまま論理演算子として使うと、コンピュータが変数名と間違えますので、「.AND.」「.OR.」「.NOT.」にしました。初期のキーボードには「<, >」がありませんでしたので、「.LT.」「.GT.」のように書きました。幾何学は、文章論理で説明する学問の性格がありますが、これを記号式ですべて表す一般的な方法は未だありません。

5.3.4　ワープロ本体作成のプログラミング

コンピュータの助けを借りて、図を描かせることの研究は、コンピュータ本体開発の初期段階からありました。文字そのものは図形です。これを、二次元的な領域に体裁よく並べると文書になります。体裁を指定するときは、プリンタの機能を考えることになりますが、差し当たって文字の説明を先にしておきます。文字データは、文字コードと同時に、形状の性質（属性）を持たせます。コンピュータ開発の初期段階では活字の寸法は一種類、パイカ（12pt）しか使いませんでした。テキストエディタ本体を作成するプログラムは、文字コートだけを扱うだけでしたので、８ビットマイコンに組み込まれていた簡単なBASIC言語でも作成できました。ワープロになると、文字コードに形容詞的な情報を余分に付けたデータを扱います。現在の時点(2010年)ではオブジェクトと言う概念で扱い、ワープロソフトの開発はオブジェクト指向プログラミングで組まれます。文字寸法（大きさ）は種々のポイント寸法の選択ができますので整数で寸法系列を指定します。文字の書体、文字の色は、デザイン情報です。適当な数値に直して定義します。１文字単位で文字データに必要な属性に「太字:bald体」「斜体:italic」「下線:underline」がありますが、これらは、それぞれ１ビットのデータのon/offで切り替えるようにして使う形容詞的属性です。

5.3.5　線図作成と濃淡図作成のグラフィックスソフト

物の形は、外形線で図に表すことができます。寸法数値を添えることで、正確で具体的な情報となります。工業製図は、物造りに必要な情報を表現する技術として、線描きの技術で作成し、原則として色や濃淡表現を使いません。数学的な関係を図化するときも、曲線で表すと数式の性質の理解に役立ちます。この場合には、絶対的な寸法ではなく、相対的な大小比較が分かるような描き方をします。線図を描く装置は、電気計測器ではレコーダとして製作されています。工業製図に使うコンピュータ制御の作図装置はプロッタと言います。工作機械をコンピュータで制御することを数値制御(NC: Numerical Control)と言いましたが、プロッタは、ペンの位置制御に応用して作図させる装置です。この装置では、濃淡表現には向きませんので、それに代えてハッチング(hatching)の技法を使います。テレビの画面は、以前は白黒、それからカラーのブラウン管を使う時代へと移りましたが、濃淡表現は電子ビームを線描きのように動かして（走査させて）その道筋だけで発色させる装置です。線図表示にも利用されましたが、濃淡図を表示するモニタとしての利用に特化して使いますので、こちらは線図を描くには向きません。マイコンの利用は、カラーモニタでグラフィックスを楽しむ使い方が多くなりましたので、それをサポートするソフトウエアが必要になりました。つまり、グラフィックスソフトは出自が違う二系統、線図用と濃淡図用、があります。現在はレーザプリンタやインクジェットプリンタを使って、濃淡図を印刷できるプリンタができて、プリンタとプロッタとは原理的には同じ装置になりました。文字は図形ですので、文字の作図データが二系統あります。線図用データと濃淡図用データです。コンピュータから見るとき、プリンタとプロッタは外部装置、つまり「物」ですし、図形も形を持つ「物」扱いをするようになり、この全体を扱うソフトウエア開発を「物を扱う」と言う意義を持ったオブジェクト指向プログラミング(object oriented programming)と呼ぶようになりました。物の性質を言うときは、形容詞を踏まえた属性変数名を使い、数値で具体的に指示する方法を使います。

5.3.6　感情形容詞の定量化の研究がある

美しいと感じる感情は、主観的な認識ですので、対象となる物でも、また人によっても評価が変わります。幾何学的な形状について言えば、古典的には黄金比が美しさを表す数値として扱われています。景色が良い設計と言うことを、都市計画の際にどのように取り込むかの研究は、中村良夫が始めた景観工学があります。また、感性工学と言う専門分野もあります。これらは定量化のできない形容詞の性質を、コンピュータで扱うためには、どのような数値化をするか、の研究と考えることができます。しかし、美人コンテストの審査をコンピュータに任せるような研究は、やはり程ほどにしたいものです。

6. 文書の作成技術

6.1 文書に作成する意義

6.1.1 言葉を文書にして残すこと

この章は、最後の第 6.6 節の編集記述言語 markup language(ML)の説明を目的とするのですが、その前に、少し遠回りをしますが「文書をどのように作成するのか？」その一般的な解説から始めます。そもそも、声に出して言う言葉は音の並びです。一過性の現象、つまり、その場限りであって、物理的には何も残りません。社会生活では、文書にしたものを信用の置ける情報とします。テープレコーダなどを始め、種々の電子化媒体が開発されましたので、音声そのものを記録して再現することができるようになりました。しかし、紙以外、眼に見えない媒体を使う記録方法があっても、紙に再現した文書にすることで、始めて保存に耐える記録として残ります。ラジオ・テレビの報道は一過性ですので、うわさ話として話題にできても、紙として残す新聞・雑誌の情報の方を記録として使います。コンピュータの利用は、大量の情報を種々の記憶装置に保存し、処理結果をモニタ上で確認するだけで済ませ、必要に応じて用紙に印刷することが普通になってきました。モニタ上の画像をソフトコピーと言い、これも一過性です。記憶装置内にあるデータは眼に見えませんので、紙に印刷しない限り、存在しない情報です。学術論文の発表を雑誌形式で出すことをやめて、電子出版に切り替えることが多くなりました。しかし、どこかで紙の形で保存をしておく必要があります。こちらを**ハードコピー**と言います。

6.1.2 プログラミングのソースコードも印刷して残すこと

コンピュータプログラミングコードは、コンピュータが実行できるように工夫された言語で書いた文書です。人が読んでも理解できるように、最初に作る元文書をソースコードと言います。コンピュータが理解するのは、機械語に翻訳した、対象とする（オブジェクト）コンピュータ向けの文書（オブジェクトコード）です。これは、読める文字で符号化しても理解が難しい文書です。そのため、ソースコードを公開しなければ、プログラミングの財産を或る程度は保護することができます。ソースコードもハードコピー化しておく注意を怠ると、プログラム本体がブラックボックス化して、修正も改良もできなくなり、結果として知的財産が失われます。紙に印刷しないで、フロッピーディスクに保存したソースコードは、ディスクが読めなくなると消滅したと同じです。コンピュータを扱うときは、コンピュータで扱う文書の読み書き技能を習熟しなければなりません。この全体概念を、コンピュータリテラシー(computer literacy)と言います。具体的な問題の一つが、コンピュータ向けの作文術です。プログラミング言語は、無駄を省いたコンピュータ寄りに特化した言語、例えばC言語、などを使います。しかし、一般的なユーザには、読んで分かる普通の文書形式を持たせた言語がよいのです。その意味では、COBOL がそれを意識した言語です。

6.1.3 記録を残す目的だけの文書がある

コンピュータのファイルには拡張子で(.log)を見ることがあります。ログは航海日誌と訳されていて、時刻を付けて作業を記録した文書です。電話や手紙の受発信記録がログです。日記または日誌は、本来、個人が素養として書くものであって、強制されるものではありません。自分の日誌を残し、これを隠居後に自叙伝(autobiography)に編集するのは政治家では一つの嗜みです。企業において、公的な立場にある人は、その立場での日誌を記録し、その企業の歴史を残します。企業自体が編集する自分の企業の社史や記念誌は、特別に編集の組織を作ることをしなくても、日誌を編集すれば済みます。そのため、役職に在る人に日誌を書くことを義務付けることがあります。「良いことずくめ」だけを記録し、後で不利になりそうな記録を残さないことや、意図的な廃棄をすることも起こります。倫理（モラル）照らせば、あるべき姿ではありません。歴史の記録には、かなりの恣意的な改変があるものです。民主主義の時代は、多くの意見を持った人が合議で何かを決める会議が多くなりました。会議の後では議事録を作成するのですが、内容を確認する儀式を経て文書記録として保存します。これは歴史を記録している意味があります。議事録に載せることを目的とする発言や、逆に、削除したい項目などを巡って駆け引きが行われることがあります。議事録は、全員一致の内容にはならないのですが、契約書や密約の書類、少し物騒なものは血判状があります。当事者の一致が条件の書類ですので、全員の署名が必要です。公的な立場であっても、署名は私的に行われますので、この種の書類は、或る期間の保存の後で公開される場合があります。これは、私的に保存するのではなく、公的な保存図書館に保存するべきですが、これらの扱いが日本では未だ未成熟です。

6.2 言葉の理解から作文へ

6.2.1 作文教育は難しいこと

言葉、それも、話し言葉を覚える過程は、受動的に耳で聞くことに始まります。子供は、文字を使わなくても日常生活には不自由しません。書いたものを読んで中身を理解するには、字形としての文字を覚え、その読み方を学びます。他人が書いたものを読んで理解する段階の次は、自分で作文することに挑戦します。このとき、明確に作文の目的が必要です。コンピュータのプログラミングは、コンピュータに理解してもらう文書を作成することです。義務教育段階の児童生徒は、社会生活の外にいます。覚える方に重点がありますので、相手に何かをしてもらう、その意思表示の目的で作文させる教育をしません。観察したことを客観的に作文することも難しいものです。普通には、日記など、自分を中心とした行動の記録か、感情移入の作文指導です。具体的な指導に困って「思った通りに書きなさい」となります。それを評価する方も、客観的な判定方法を持ちません。大学教育になると、論文やレポートのように、論理的な内容を持たせる作文技術が必要です。これを系統的に教育する科目を、technical writing と言い、日本以外では、ほぼ必修の教養科目です。その理由は、種々の言語環境からの学生が集まりますので、標準的な形式での表現法が重要になるからです。目的意識を持った文書を**実用文書**と言うことにします。これは、論文やレポートもそうですが、日常的には手紙がそうです。特に、企業で使われる手紙を business letter と言い、企業の顔を持って発信されます。企業では、秘書課に当たる部署がこれに当たります。文書作成のような実務に関係した教育は、以前は商業学校のような実業学校が担ってきました。学問の衣を付けた教育の方を有り難がるようになって、作文技術のような実務教育が軽く見られるか、手が回らなくなっています。

6.2.2 話し方の教育も難しい

文字に書いて相手に伝えるのは間接的です。直接的には相手との対話です。このとき、頭の中では伝える内容の論理的組み立てが行われます。その上で、会話(conversation)と討議 (discussion)での話し方が必要です。相手がありますので、マナー（方法）が必要ですが、これを大きく敬語の使い方としています。日本語に取り入れた英語のマナーは、礼儀の意義で使うことが多いのですが、作法の方が当たります。敬語の中身は大きく三つあって、尊敬語・謙譲語・丁寧語です。企業の新入社員の教育でよく扱われるのですが、本来は家庭環境で基礎的なしつけとして身に付ける課題です。会話と討議には、話しの受け渡しの約束（ルール）が必要です。一方的な喋りになる状況は、演説もそうですが、学校教育の場で教師側の発言に見られます。筆者の経験を言えば、お行儀の悪い聴衆も困りものですが、聴く側からの質問が無いことや、無反応であるのも寂しいものです。複数の人が討議をするときには、誰かが司会役をして交通整理をします。上記のマナーやルールに無頓着な人がいると、司会役は苦労します。

6.2.3 文書の作成には三つの要素がある

文字で表すことが作文です。作文教育をするには、「これが正しい書き方である」とする規範が必要です。これが正書法(**orthography**)の原点です。それに先立って、文書全体をどのように作るのかの全体概念を踏まえます。文書の作成には三つの要素があります。第一：正しい**文章**、第二：文書の**書式**、第三：**体裁**です。実用文書を作るときは、明確な目的があります。数枚以内の手紙、お知らせ、お願い程度のものから、ページ数の多い、形のある書物を作る場合も含め、一般には複数の人手を経て文書を作成します。基本的な作業は、上の三つそれぞれに別の人が当たります。文章原稿を書く人、それを決められた書式に編集する人、版組みと印刷に当たる人、体裁を整えて製本を担当する人、最後に発行に当たる人です。日本の書物は、この情報が奥付に載ります。ワードプロセッサが無かった時代、著者は、草稿を 400 字詰めの原稿用紙に手書きしました。この原稿を編集者の所に持ち込めば、これに編集者が書式情報を書き込んで、版組みの人に渡し、残りの作業をそれぞれの専門家が処理してくれました。校正の段階で著者の所に戻るループがあり、校正情報を書き込みます。編集と校正に使う記号などは、標準の約束が日本工業規格(JIS)に載っています。普通の人は書式と体裁については漠然とした知識しか無くても済みました。従来の作文指導は、このような作業環境を前提としていることを理解しておきます。ワードプロセッサが使えるようになって、小部数の簡単な文書は、すべて自分で作成できるようになりました。それは第二、第三の作業も自分でするようになったことです。そうなると、いままで専門家がしていた作業に関わることになりますので、特殊な専門用語を覚え、ワードプロセッサの操作に慣れ、パソコンを使うことに悪戦苦闘するようになりました。

6.3 文章の書き方

6.3.1 文字の物理的な並べ方を理解する

日本語では、文単位の作文ならば、漢字と仮名の使い方と、句読点の使い方が作文技術の主題です。前章までの説明は、文章を品詞に分解した単位で、送り仮名の付け方に関連した文字の並べ方について、正書法の模索をしました。書式と体裁の部分が、英語では typography と呼ばれ、印刷術と訳しています。文書は、幾何学的に言えば、決められた用紙の領域を文字、言わば小単位の図形、で埋めて作ります。書式と体裁は相互に関連があります。ページ単位の体裁は、レイアウトです。書式は、レイアウトと関連付けた構成を言い、表題、見出し、章・節・項の立て方、それらの配置、寸法などです。全体の体裁は、見易さと取り扱いに焦点を置いたデザイン的な要素です。活字の書体、寸法の選択、段落構成、製本などです。文章・書式・体裁の三つの要素は、何を対象として文書を作成するかの目的によって、内容に相違があります。日常的な例には手紙の書き方があります。標準の外形寸法と体裁がありますので、定形外は郵便料金が別です。宛名の書き方は書式に含まれ、大体の決まりがあります。手紙の文章は、拝啓・時候の挨拶・用件・敬具で締め、日付などを加えます。E-mail のソフトは、モニタ上で作業用の書式と体裁部分のテンプレートを提供しています。従来、印刷したい文章は、手書き原稿を印刷屋さんに持ち込めば、書式と体裁とを専門的に処理してくれました。テキストエディタは、手書きに代えて、文字並びの作成だけを目的としたツールです。ワードプロセッサとなると、書式と体裁とを含めた全体を設計し処理するツールです。ワードプロセッサのマニュアルには、多くの印刷関係の専門用語が使われますので、一般ユーザも、それらを理解しなければならなくなりました。

6.3.2 文単位での物理的な構成を理解する

文章の最小単位を、主語と述語の対で作る文とします。これを細かく見ると、句で区切り、さらに品詞に分け、最小単位が文字です。文の集め方で見ると、集合の大きい順に**章**・**節**・**項**でグループ分けをします。文の文字並びは連続させます。用紙幅（縦書きならば用紙高さ）がありますので、入りきらないときに次の行に続けます。眼には改行ですが、論理的には文字並びは連続です。最小の文単位集合を**段落**(paragraph)と言い、眼で見て切れ目が分かるようにします。普通は、行を新しくして（改行)、その書き出しを一文字空けます（indent：インデント)。縦書き時代の用語が字下げです。インデントしない場合には、空白行を入れます。段落の集合で項を立て、項の集合を節、節の集合を章、章の集合で１単位の文書とします。論文やレポートでは、章・節・項に番号を付け、見出しを付けます。手紙などは、見出しや番号をつけませんが、段落の考え方を基本に踏まえます。段落単位で、一つの主張をまとめます。その要点を見出しにできると、全体の文集合の見通しに役立ちます。筆者の作文の物理的な構成では、段落単位は見出しを付けてありますが、１段落で１項としてあって、項番号を省いています。ただし、インターネット版では項番号をつけてあります。筆者の書く１段落の文字数はやや多めですが、その量は、インターネットで閲覧するとき、モニタ１画面に収まる量を目安としています。

6.3.3 言文一致は理想通りには実現できない

文書は、声に出して再現することを目的とする場合と、内容の正確な記録を眼で確かることを目的とする場合とで、書き方に違いがあります。この全体を**表記法**と言い、約束を決めます。表記通りには発声しない場合がありますし、発声を正しく表すこともできません。さらに、日本語では、話し言葉と書き言葉とに大きな区別がありました。書く作法と話す作法とがうまく噛み合わないところがあります。書き言葉の文体としては、口語体・文語体・漢文体、などの区別があります。書き言葉と話し言葉の差を無くす言文一致の努力は、明治以降ずっとなされてきました。近年になって、言語の種類が増え、異なった言語の、適度な使い分けを必要とする時代になりました。これには、英語のような外国語だけでなく、プログラミング言語、さらにはグラフィックス言語、そして、後で解説する編集記述言語などがあります。日本語の環境に限っても、漢字熟語は同音異義語が多いこともあって、眼で文字を見ながら意味を確認する傾向が強く、話し言葉で情報を伝える技術と整合しないことがあります。外来語をカタカナ用語で多く使うようになりました。しかし、語の意味を別に覚えなければ分からない不便さがあります。英語でも話し言葉と書き言葉の違いはありますが、日本語ほどに大きな違いがありませんので、話した言葉を、多少の修正をして、そのまま文書に落として利用することが行なわれています。そのため、**口述筆記**(dictation)と**速記**(shorthand)は、秘書の技能として必要とされています。タイピングの技能と相まって、話し終わった段階で速記録ができていることがあります。

6.3.4　句読点の使い方が難しい

日本語の表記法では、読点「、」の使い方は、英文のコンマの使い方に比べれば、規則が無いと言えます。逆に言えば、自由に使うことができます。英文は、単語間にスペースがあります。日本語は分かち書きをしませんので、語の切れ目が分からないことが起こります。文書の書き手は、語の切れ目を意識していますが、読む側は、中身を理解しても同じようには読まないことが起こります。特に、漢字の並びは、音訓の読み分けが難しいことがあります。前章までの個別の品詞の説明は、送り仮名を含め、漢字と仮名の組み合わせの規則を整理することにありました。一般的な文書を話者が読んで相手に伝えるとき、僅かな息継ぎとイントネーションを聞き分けて、間違いなく理解します。文書本体にはそれを表す方法がありません。私事（わたくしごと）ですが、筆者の場合には、この原稿を含め、幾つかの書き分けをして、読者が読み易くなるような工夫をしています。その一つは、なるべく和語的な言い方をし、音読み熟語を少なくしています。上に挙げた「私事」は、「しじ」と読むのが普通でしょうが、ふりがなが付けられませんので、必要を感じたときには括弧で読みを入れています。また、語の切れ目に読点「、」を使うまでもないとき、半角のスペースを入れることも試しています。仮名文字が並ぶと、単語の切れ目が分からなくなりますので、形態素解析の効率が下がります。これも、書き手がスペースを入れるようにすると簡単に解決します。このような書き方は、日本語から英語に自動翻訳をするときに仮名漢字混じり文を解析するときに必要になる技法です。ワープロで日本語原稿を作成するときは、仮名文字の連続を形態素解析の課題としますが、それとは逆向きの解析です。

6.3.5　外来語の取りこみと表記法

海外文化を輸入して近代化に努力してきた明治以降は、読んで理解することに傾斜した外国語教育が主流でした。外国語で話すことと、外国語で作文することには、やや距離がありました。それは、話したり書いたりして伝える相手が身近にいなかったことが理由の一つです。英語で言えば、英単語を覚え、文法を理解して英文和訳の学習から始めますが、和文英訳は教育し難い環境でした。英語に直しても、「良し悪し」の評価は、native speaker でないとできません。そのような環境での外国語の習得は、実用を離れた教養を深めることに向かいます。そもそも、漢字は、日本にとっては中国から輸入された外国語の文字です。万葉集は、音を表す文字として使いました。漢字の意味の方を使いたいとき、和語の言い方を当てました。これが訓読みであって、日本語の表記法を複雑にしました。しかし、自然発生的に一応のルールができてきますので、それを標準化する試みが出てきました。常用漢字や送り仮名の付け方がそうです。官主導の正書法の提案は、権威主義的な押し付けになり易いので、その臭いを嫌う雰囲気もあります。一般的な初等教育の環境では意義があります。実用文書の場合には「この書き方がいいよ」とする適度な提案や、それを受け入れる妥協も必要です。

6.3.6　眼で見ることを目的とする文書がある

数学・物理・化学の記号式は書き言葉として使います。英字の大文字・小文字、斜体・直立体の区別で別の意味に使うことがありますし、声に出すとき、妙な発音で言うか、元の単語の読みで言う、などをします。コンピュータ言語でも、大文字と小文字とを別文字として扱い、単語の種類を、見て分かるようにする書き方が使われるようになってきました。日本語では漢字と仮名との混ぜ書きをし、漢字の読み方が複数あるので、場面に応じた読み方と書き方をしてきました。欧米のアルファベット系言語は、USA のような頭字語(acronym)を使うと、元のスペルが分からない同音異義語が増え、元のスペルで読むように使うと読み方が表記とは別になります。それを考えると、日本語の仮名漢字混ぜ書きの表記方法は、最先端の手法と言えます。一般的な小説や雑誌など、読み物としての文書は、話し言葉で書きます。漢字の使い方を常用漢字で制限しても、大きな障害にはなりません。眼の不自由な人には読み聞かせができますし、テープ・レコードなどの別媒体を補助に使うことができます。熟語漢字を使う硬い文書は、同音異義語がありますので、眼で見て理解する使い方をします。漢字の使い方を制限されると迷惑なことも起こります。読んで理解することも考えますが、誤って理解される、または、書き手が意図した読み方と違って発声されるのを避ける工夫が必要です。これが、文書の書き方の規則（正書法）を提案する理由の一つです。

6.4　書式

6.4.1　書式は用紙上の構成方法

書式(form)は、用紙の外形も含め、文章・図などの必要項目の構成方法(structure)を指します。英語の用語説明では、form について structured document とあって、構成や組み立ての意味を含みます。用紙の寸法に合わせて、余白や文字の寸法、その配置などに注意を払いますが、こちらは体裁であって、デザイン的な要素です。文書は、伝えたい用件に応じた書き方があります。例えば、履歴書や、何かの届けや申請をするとき出す書類は、書式の決まった印刷用紙が用意されていて、必要なところに書き込めばよいようになってきました。元々は、個人が手書きで書類を作成するのが原則でした。しかし、人によって形式が異なるのでは書類を受ける側が扱いに困ります。一般の人は代書屋と呼ぶ書類作成の専門家の手を借りて、必要充分な書類形式に整えなければなりません。印刷された申請用の用紙は、手書きの形式を踏襲して書式が作られています。代書をする人は、以前は司法書士を指していて、それなりの専門的な知識に加えて、文字をきれいに書ける素養が必要でした。昔の武家社会では、右筆または佑筆（ゆうひつ）と言う文書の専門家が殿様に仕えましたが、現代風に言えば秘書に相当する官職です。コンピュータのモニタ画面には、擬似的な作業用の領域を作ります。これを通称で**ウインドウ**と言います。プログラミングの用語は**フォーム**です。

6.4.2　書式の英語にフォーマットもある

書式の英語には**フォーマット**(format)もあります。こちらは、データの方の、外見や構造の意味で使います。数を文字に書いて表すとき、種々のフォーマットを使い分けます。小数点を持つ実数形式、整数を使った分数で表す、指数表現を使う、などがあります。コンピュータ内部のメモリやファイルなどの記録媒体には、ビット並びで記録されます。これは眼に見えませんが、ビット並びの物理的構造もフォーマットと言います。数は、文字並びで表すときと、ビット並びで表すときとがありますので、相互の変換が必要です。これを**書式変換**と言います。数をコンピュータに伝える**入力**、数の内部データを文字として印刷する**出力**の操作は、プログラミングでは面倒な部分です。プログラミングの教育は、この部分を覚えることが重要な基礎課程です。

6.4.3　印刷物の書式の項目は版組み指定に使う

印刷物の書式項目は、活字の寸法・字体・組み方です。活字のデザイン上の種類をフォントと言います。出版社は、自社固有のフォントを使うことで個性を主張しています。日本語の印刷では、書式は、例えば、「A4 版・横書き・明朝体・8 ポ・25 字詰め・50 行・二段組み」などのように決めます。学術雑誌では、その雑誌固有の印刷仕上がりの書式が決められています。少し前までの学術雑誌は、著者の原稿に基づいて雑誌の発行機関の責任で活字を組みました。しかし、数式などが多い理工系の文書原稿は、組み方に特殊技能が要求され、校正も手が掛かります。コンピュータを利用して、著者が完成文書として準備することを DTP(desktop publishing)と言い、これで作成した図形原稿をそのまま写真製版することが普通になりました。この方式で作成する学術雑誌は、その雑誌の権威の象徴でもあった書式と体裁が失われ、掲載論文の実質的な内容で評価されるようになってきました。

6.4.4　広義の書式には文章の文体も含めます

書式には、文章の項目構成、つまり、章・節・項の番号付け、の約束も含みます。作文教育で「起承転結」の構成を法則のように主張するのを見受けます。これは、段落の論理的な構成方法と内容とを関連させた規則と解釈するとよいでしょう。文書全体は、頭に置く項目（目次、前書きなど）と、終わりに置く項目（後書き、索引など）が別にあります。ページの付け方は、本文とは別にするのが原則です。体裁と関係しますが、表紙・裏表紙を別用紙、別ページにする場合と、しない場合とがあります。小ページ数のレポートは、全体を通したページ番号を付けます。雑誌と単行本の区別は、雑誌扱いをする場合、表紙からページ番号を始めます。使用する文字の種類が違うとき、例えば、英語版と日本語版の区別は書式の違いとします。書式には、文体も含めます。口語体・文語体、「です・ます調」「である調」などの選択の幅があります。英語では米国英語か英国英語の違いも、単語のスペル違いがありますので、書式の問題とします。

6.5 体裁

6.5.1 文書全体のデザインが体裁である

第三の要素である体裁は、用紙の綴じ方や製本のような物理的な外観を指します。前項の文章・書式と重複する事項も多いので、レイアウト(layout)と言うときは、書式に関連して、ページ単位のデザインを言います。図などを含め、文字並びの幾何学的デザインです。手紙を書くとき、用箋や封筒の質やデザインを選ぶことは、体裁の問題です。企業が発送する手紙は、企業の顔を代表するように固有のデザインをした専用の用箋（レターヘッド付き）や封筒を使います。ワードプロセッサを効率的に利用するとき、あらかじめ決めておいた定形的なレイアウトを保存しておいて、伝えたい本文を書き込めば簡単に清書が完成するような方法があります。このときに使われる雛形の書式とレイアウトを**テンプレート**(template)と言います。伝えたい具体的な項目に落ちがないようにする視覚的な方法です。その必要項目を、5W1H と言うことがあります。昔の軍隊では「いつ、どこで、だれが、なにを、どうした」と報告させる訓練をしていました。これに「なぜ、どうやって」を加えると 5W1H(what, when, where, who, why, how)に対応します。

6.5.2 綴じ方にも規則がある

綴じ方も体裁です。大福張は上綴じで、紐を付けてぶら下げます。縦書きで書かれた和書類は右綴じ、英文や横書きの日本語のレポート類は左綴じになりますが、綴じは書式と連動しています。ページの切り方にも決まりがあって、左綴じは見開きの左ページを偶数ページ、右綴じは右を偶数ページにします。印刷と製本には専門の技術集団がありますので、書物の奥付に印刷と製本の企業名が載ります。質のよい製本には特殊な道具や技術が使われますので、一般のユーザの手に余ります。簡単な書類管理は、ホッチキス止めか、パンチで穴開けをしてファイルに綴じます。用紙の綴じる側には、綴じ代を見込んで相応の余白を取ります。文書の体裁は、保存と検索などの管理技術と密接に関わります。官公庁の文書の寸法は、JIS の B 列から A 列規格の採用に変りましたが、これは大きな変化を伴います。例えば書架やファイルキャビネットなどの外形寸法が変り、ひいては家具調度のデザインにも影響が及ぶからです。

6.5.3 大きな寸法の用紙は折り方を決める

工業製図は、大版の用紙に描きます。使うときは広げて見ますが、他の書類と共に保存するときには、その書類の外形寸法に合わせた折り畳み方と綴じ方が必要です。これは文書管理に関係しますので、工業製図の規格には参考事項として載っています。地図も大きな用紙を使います。山登りをする人は、地図を見易く広げられるような折りにして、必要な箇所が見えるように納めるケースを使います。長い用紙は短冊に折りますが、折りを嫌うときは巻物(scroll)で使います。日本の折り紙技法では鋏を入れないで折ることが一つの条件ですが、欧米では適当な切れ目を入れて、変化のある折り方と開き方の工夫が試みられています。欧米の書店で売っている都市地図に良くみられます。

6.5.4 実質で扱う文書は体裁を無視する

文章原稿は、文書の実質をまとめますので、最初、体裁を考えません。さらに、書式も省きます。原稿は、編集者のところで書式とレイアウトを指定する記号や文字を書き込んで、組み版工程に回します。この書き込みの英語が **markup** です。**テキストエディタ**は、文字並びだけを編集するツールです。文字データだけを保存するファイルを**テキストファイル**と言います。日本語の環境では漢字と仮名を使う場合もテキストファイルとします。英語の文書は、英字・数字・記号だけを考えればよいので、文字種が少なくて済み、コード系が簡単になります。こちらを区別して**アスキーファイル**と言います。英字だけを使って電報を送受信する装置が**テレタイプ**(tele-typewriter)でした。電子メール(e-mail)は、テレタイプ利用の延長に位置していますので、基本的には書式情報を含んだデータを扱いません。通信回線を使って文書を送信し、送信側と同じ書式とレイアウトが受信側で再現できるようにするときは、書式情報を加えたテキストファイルを送ります。この情報は、文字と記号とを組み合わせます。これが組み版言語(markup language: **ML**)です。実を言うと、この章の始めに述べたように、ML を説明することがこの章の目的ですが、その説明の前段階として、文書作成の三要素の解説から始めたのでした。

6.5.5　単純なプリンタは書式が固定されている

単純なプリンタの代表は、手動で操作するタイプライタです。英文用と邦文用があります。英文用は電動の装置がありましたが、邦文用（図 5.1 参照）は電動化が難しく、それに代わる装置として**ドットマトリックスプリンタ**が開発されました。カーボン紙を挟んで、数枚程度までの複写が同時に取れるプリンタとして利用ができます。ちなみに、電子メールで CC と書いてあるのは Carbon Copy を表し、自分の控えに自分宛に送信することを指定します。現在ではコンピュータからの出力にレーザプリンタやインクジェットプリンタが使えます。こちらは同時に複数枚の複写ができません。したがって、部数を必要とするときは、同じ操作を複数回繰り返すか、高速の複写機、または、インクを使う高速の印刷機を持つ専門工程に頼みます。元に戻って、単純なプリンタは、活版印刷に比べると、書式の選択に制限があります。フォントは、原則として一種類です。このプリンタに文字データを送信して印刷するときは、フォントは、プリンタ側で持っています。文字コードを送信すると、指定する文字を順に打ち出す形で印刷が得られます。プリンタの一行当たりの文字数とページ当たりの行数が決まっていました。文字並びが行末に来れば、自動的に改行させることができます。しかし、ページの最下行での自動ページ変えの制御はありませんでした。テキストエディタのモニタ画面は、この単純なプリンタの印刷画面を表示するように設計されています。このとき、印刷制御に使う信号は、復帰(CR: Carriage Return)、行送り(LF: Line Feed)、タブ(TAB: Tabluator)、空白(SP: Space)、後退(BS: Back Space)です。これらは文字図形を持ちませんので、**ファンクションコード**(function code)と言います。テキストファイルは、ファンクションコードと**文字コード**(character code)としか含まないファイルです。なお、後退はテキストファイルでは機能を持ちません。

6.5.6　文字を図形として扱うプリンタ

説明が後先になりましたが、印刷の物理的な処理は、最小の図形単位である活字の寸法を基準として、用紙の平面領域を、この単位で埋めます。これが**版組み**(type setting)、幾何学的設計が**割り付け**(layout)です。書式情報は、印刷仕上がりの体裁を指示するデータを指します。文字単位では、フォントの種類・上付き・下付き・太字・斜体・アンダーライン、などです。フォントには、文字幅について、等幅フォントと、プロポーショナルフォントの区別があります。文字並びでは、センタリング（左右揃え）・右寄せ・左寄せ・均等割り付け、などが行単位での書式情報です。レイアウトは、眼で見なければ分かりません。ワードプロセッサは、モニタの画面上で印刷のレイアウトを確認して、プリンタにデータを送り（送信し）ます。ページ全体をビット並びの図形データに直して送るときは、上の項で説明した単純なプリンタとは別原理の装置、例えば、レーザプリンタを使います。印刷情報をファイルに保存するときは、二種類の選択があります。文字コードと書式情報とを含めたデータ形式か、図形化したドット形式のデータです。書類を再編集できるようにファイルに保存したいときは前者の方式がとられます。図形データ化したファイルは再編集ができません。これは書類を改変されては困るときに行われる一つの方法です。

6.5.7　モニタをプリンタの擬似装置として使う

モニタは、用紙の無駄使いを抑える目的で使われるようになったのが最初です。これが Character Display です。テレビのブラウン管を利用したのが始まりですが、文字の解像度が悪いので、専用のディスプレイ装置が開発されました。普通のプリンタと同じ 1 行 80 文字（半角）が読める解像度を持たせたモニタが、横・縦のビット数 640×480 です。これを Graphics Display としても使うようになったのは自然の成り行きであって、同時にカラー化も進みました。当初は、文字とグラフィックスとの表示は、ソフトウエア的に別扱いをしました。これはプリンタとプロッタとが、原理の異なる別々の外部装置であったことも理由でした。オペレーティングシステム(OS)がウインドウズに変ったことと、プリンタとプロッタの装置原理上の違いがなくなりましたので、文字表示も図形の集合としての扱いになりました。モニタは、レーザプリンタの擬似装置として使うようになりました。ソフトウエア的に見れば、モニタもプリンタの一種です。装置を制御するソフトウエア（デバイスドライバ）を介して印刷イメージをモニタに表示します。このソフトウエアが、つまりワードプロセッサです。ここで、モニタで見たままの書式とレイアウトで用紙に印刷できることを謳った用語が WYSIWYG (what you see is what you get)です。文書原稿は、ファイルに保存し、それを読み込んで表示させます。このとき、文字コードと同時に、書式情報も必要とします。この方法のとき、書式情報をバイナリーデータとして持たせる方法と、文字と記号の組み合わせで挿入する方法とがあります。前者の方法が、一太郎、MS-Word などです。後者の場合、この文字並びが、組み版言語、または編集記述言語(Markup Language: ML)です。

6.6 組み版言語

6.6.1 編集用の記号文字付きの文書を使う

コンピュータ用語で"hyper"を接頭辞に使う hyper-text (HT)が造語されたのは 1965 年頃です。日本語に訳し難いのですが、「超」の字を付ける感覚の用語です。野口悠紀夫(1940-)が造語した超整理法があります。コンピュータを利用し、通信回線を介して文書情報を遣り取りするとき、特別な構造のテキストを使うことから造語されました。これは、原稿の文字テキスト（plaintext と言います）に、編集記述言語(ML)を組みこんでありますので、HTML 文書と言うようになりました。これに先立って国際標準化機構(ISO)は、編集記述言語の構想をまとめて 1986 年に規格を発表したものが SGML (Standard Generalized Markup Language)です。HTML は、この実用版です。これと考え方は同じですが、Donald Knuth(1938-)が開発した Tex(1978 年頃)、それを発展させた LaTex は、数式の版組みを特に指向した編集記述言語です。HTML の方は、数式を扱うことができません。MS-Word には数式を、特別な言語形式ではなく、グラフィックスイメージで作成して挿入する数式エディタがあります。筆者の場合、数式はグラフィックスイメージとして作成したものを HTML 文書に貼りこむようにしています。したがって、テキスト部分のフォントとの整合性が悪く、統一した体裁が得られません。より上位のバージョンが XHTML です。X は extended(拡張の意味)です。こちらは単純な数式も編集できることになっていますが、まだ一般的ではありません。

6.6.2 編集用のソフトウエアを使う

新聞・雑誌の出版社は、編集作業を事務所の環境で行いますが、印刷・製本は専門工場で行わせるのが普通です。組み版の作業は、鉛の活字を使った伝統的な方法が電子組み版に代わってきています。一般ユーザは、自分のパソコンでも同じような組み版と印刷ができるようになりましたが、これが DTP です。組み版は、英語で typesetting です。古典的な編集作業は、編集者が作者の原稿用紙に赤鉛筆などで組み版の指示を記号で書き込んで組み版工場に送ります。ここで使われる記号体系が編集校正記号です。この英語が markup です。コンピュータを利用した電子組み版は、モニタ上で完成したページのイメージをグラフィックスで表示しておいて、GUI の環境で書式とレイアウトを整える作業になりました。そのため、編集記述言語を眼で見ることはありません。編集記述言語は特殊な構成ですので、これを含む全体のファイル構造は、モニタに文字並びを表示する編集用ソフトウエアと密接な関連があります。最も単純なソフトウエアが、通称で言うテキストエディタです。ソフト名は Windows 系では NotePad、日本語名では「メモ帳」です。特別な編集記述言語ではなく、改行・復帰などのファンクションコードだけを認識して文字を表示します。一太郎、WordPad、MS-Word などは、ワードプロセッサに分類され、編集記述言語をバイナリー形式で組み込んだファイルを使います。LaTex 用文書, HTML 文書は、編集記述言語もアスキーコードで挿入したファイルを使いますので、テキストエディタを使って中身を見ることができますし、編集もできます。重要なことは、通信回線に載せて送信できることを考えていることです。

6.6.3 文字コードの問題が発生したこと

編集から印刷までの作業をコンピュータ化するとき、閉鎖的・独占的な (exclusive)環境で利用しているときには、文字コードと字形との対応を自前で決めて使います。マイクロコンピュータを利用したワードプロセッサ専用機が 1980 年代に多数発表されましたが、各社のファイル仕様が別々でしたので、ファイルに納めたフロッピーディスクを介して文書での情報交換が簡単にはできませんでした。編集と印刷作業を分離し、電子化した原稿を別の工房で利用しようとなると、コード系の違いは文書の再現の大きな障害になります。加えて、文字データを通信回線で送受信するときの問題もからみます。E-mail での文字化けがそうです。コード系の標準化の努力がなされてきましたが、ソフトウエアでの文字コード変換機能が強力になってきましたので、一般ユーザの眼の届かないところで実質的な標準化が進んでいます。しかし細かな点で、文字コードと字形の違いもあります。一つの例として、日本語用のパソコンでは、アスキーコードの半角バックスラッシュは、"¥"で表示されます。プログラムソースコードをモニタで見るときやプリントをするときに文字が変ります。

6.7　コンピュータリテラシーの教育

6.7.1　学問ではなく技能の教育として捉える

筆者は、或る私立大学の文科系情報学科の非常勤講師として、５年ほど、プログラミング入門の講義と演習の経験をしました。カリキュラムの内容をどのように組み立てるかの模索がありました。学生の知識レベルはまちまちです。書店に行けば、コンピュータ関係の書物が溢れていますので、既にかなり実用知識に詳しい学生もいます。そのような学生も含めて、対面授業の場でないと基礎的な知識の穴埋めができないことを納得させる内容が必要でした。プログラミングを学問として説明するのではなく、作文術として捉えることにしました。そう考えると、技術の問題としての三つの要素を考えることができます。それは、「**道具・技法・技能**」です。道具はコンピュータのハードウエア、技法はソフトウエア、つまりプログラム、技能は使い方のノウハウです。道具としてのパソコンは、講義室で学生数だけ用意されています。キーボードのタイピングは基礎的な技能ですが、これは個人で習得してもらうことにして、必要な情報と技能習得の目標は開示しました。プログラミングはコンピュータ向けの作文ですが、何かの目的が必要です。一般的なプログラミング言語、例えばBasicやC言語を使うことは、入門授業には向きません。そこで、ワープロの使い方の基礎を教え、それを HTML 文書に直し、学生間でリンクを構成するまでの必要知識と技能を埋めることを計画しました。プログラミング文書の例題として、お料理のレシピの文書作成を含めることにしました。レシピは、料理手順の解説書、つまりプログラム文書の性格があるからです。英語で書いたレシピもありますので、作文術について日英の比較ができることも含みにできました。

6.7.2　最初にテキストエディタの使い方を教える

テキストエディタは、タイピングの練習画面を兼ねて基本的な使い方を教えます（ここでNotepadを使います)。例文は日本語、英語二種類を用意します。分量としてはA4版１枚程度です。新聞記事を主に使いますが、見出し、要約、段落構成がはっきりしているものを選びます。重要なことは、段落の概念を考えたタイピングです。テキストエディタの作業画面は、メニュー項目が少なく、ツールバーもありません。設定では、書式のメニューに「右端で折り返す」があります。初心者は原稿の行単位で改行するタイピングをします。チェックを入れると、ウインドウの横幅にあわせて自動改行された表示になることを納得してくれます。

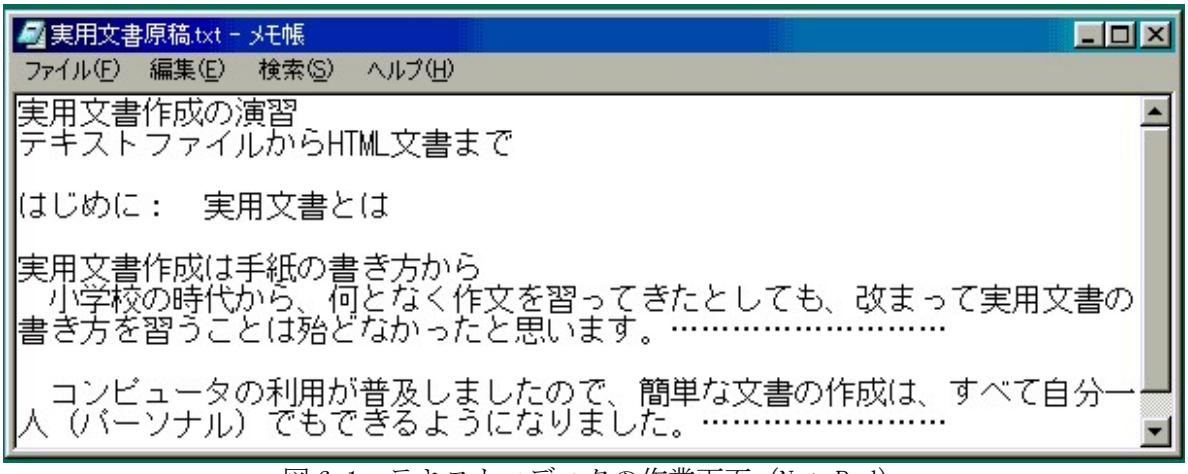


図 6.1　テキストエディタの作業画面（NotePad）

6.7.3　ワードプロセッサで書式を整えさせる

次に、テキストエディタの原稿をそのままワードプロセッサ（ここでは MS-Word）に取り込んで、見てくれのよい書式にすることを試みさせます。フォントを変更すること、表題をセンタリングすることなどで、表示方法の工夫ができることがテキストエディタと異なる機能であることを理解してもらいます。これを、「文書のお化粧をする」と説明しています。段落構成を理解してテキストエディタの原稿ができていないと、Word 上のレイアウトが悪くなりますし、次に説明する HTML 文書の体裁にも関係してくることを理解してくれます。

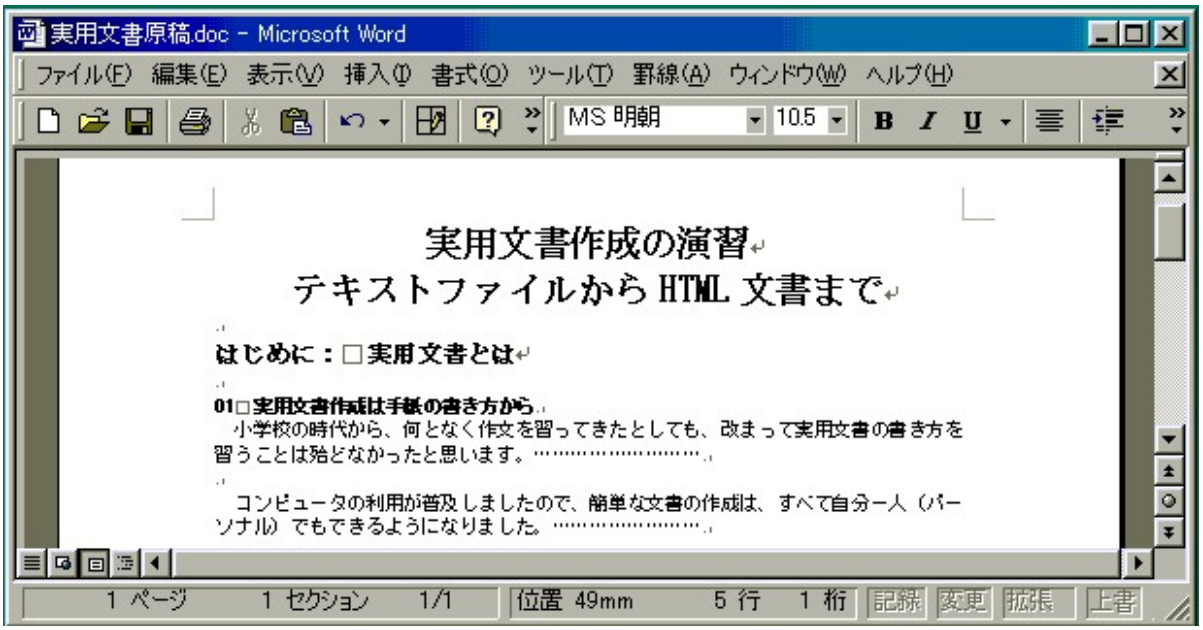


図 6.2　ワードプロセッサの作業画面（MS-Word）

6.7.4　Word の原稿から HTML 文書に落とす方法を教える

MS-Word で表示されている画面から、「ファイル→Web ページとして保存」をクリックすると、ファイルの拡張子が(.htm または.html)に代わってそのまま HTML 文書に変更することができます。これをファイル名エクスプローラからクリックすると、Off-Line の環境でインターネットエクスプローラが起動して、MS-Word の画面とほぼ同じレイアウトで文書が表示されます。モニタ上のウインドウ幅を変えると、段落単位で自動折り返し表示が効くことを確かめることができます。この画面で、メニューの「ページ→ソースの表示」をクリックすると、テキストエディタが起動し、タグで区切られた編集記述言語の付いた文書を見ることができます。実は、この方法で作成したテキストは、不必要なタグも自動的に挿入されますので、ファイルの寸法が非常に大きくなって、読み難くなっています。そのため、教師側で必要最小限のタグを使った HTML 文書の例題を示して、テキストエディタを使っても HTML 文書が作成できることを理解してもらいます(図 6.3)。少し経験のある学生は、自宅で市販ソフト、例えばホームページビルダーを使って Web ページを作成しますが、そうでない作成方法があることも納得してくれます。ここまでは基礎演習です。ここからテキストの例題にお料理のレシピを使った HTML 文書の作成に、各自で挑戦してもらいます。見本は提示しますが、自由な例題を選ぶことができます。義務として、最小限二つのファイル（日英）を作成してもらいます。これは、次項で説明する、ファイル間のリンク設定の演習に応用するためです。

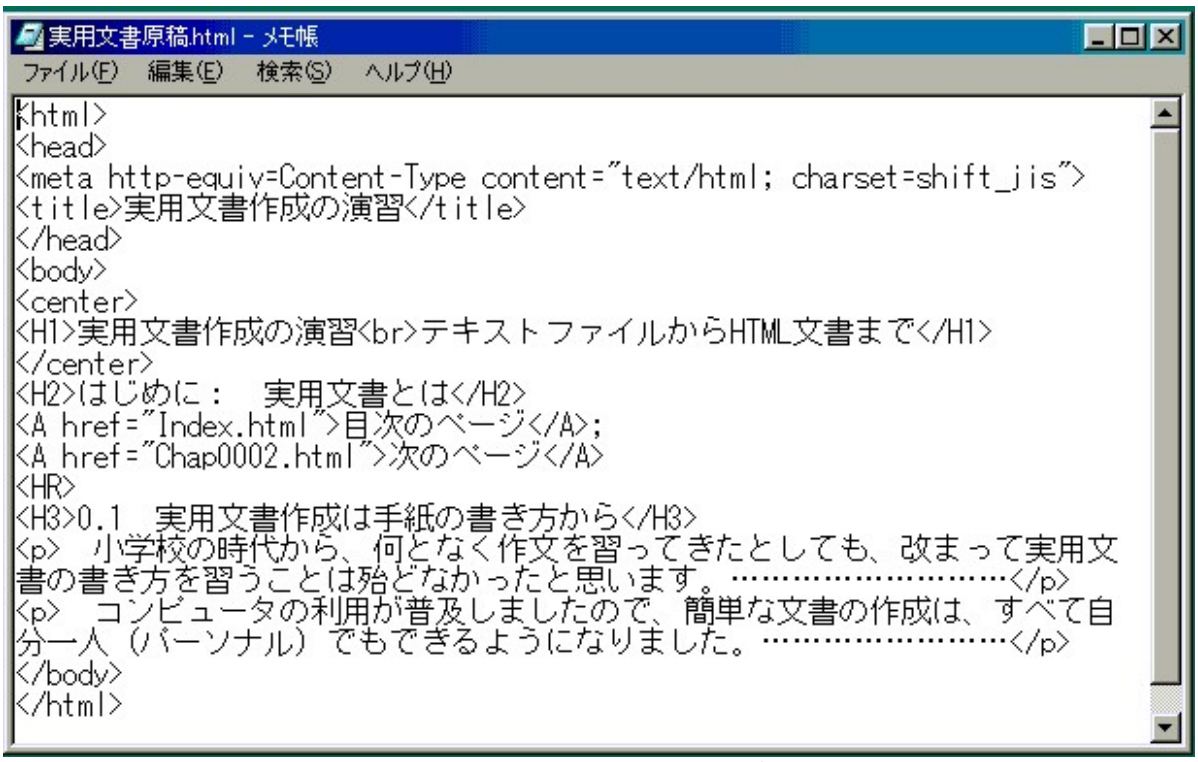


図 6.3　HTML 文書のソースコードをテキストエディタで表示させたもの

6.7.5　文書間のリンク方法を実習させる

学生が自前で作成した複数の HTML 文書は、個人名をつけた同じフォルダに納めますが、ここまでは独立したファイルになっています。この、自分のファイル相互にリンクを付けるタグを追加し、ファイルの切り替え表示ができるように、テキストエディタでクリックする見出しを付けさせます。これが旨く行った段階で、教師側で用意したホーム相当のフォルダに、学生ごとの名前で先のフォルダを転送し、すべての学生が他の学生の作品にもリンクできるようなローカルのネットワークを構成します。リンクが確実に動作することを教師側が確認したところで、レポートが受理されたと判定して、この学生の授業単位を与えることを約束しています。

6.7.6　期末試験は常識の確認に当てます

毎週の授業と演習では、出席の確認を兼ねて、10 問程度の簡単な常識問題のクイズを出し、解答を説明して自己採点をするようにしています。この一つに「1 インチの長さは何センチか？」があります。コンピュータ技術はアメリカ主導型で開発されてきましたので、長さの単位に、陰に陽にインチ(2.54cm)の寸法が使われています。活字のポイント数も１インチが 72 ポイントになっていることも、関連した常識です。期末試験では、この問いを一つのキーに使っています。これが正解でない学生は、多くの場合、出席率も低いので、レポートでの単位を認定しても成績はＡとしません。これは学生の方でも納得してくれました。

7. 外国語としてのプログラミング言語

7.1 英語の常識が必要

7.1.1 日本人は二段階の外国語の勉強をする

現代の社会環境は、コンピュータを公的にも私的にも広く利用しています。コンピュータそのものを、単なる道具と見ることから始まって、パートナーとして擬人化して付き合おうとするまでの、幅があります。ワープロやインターネットの閲覧に使う道具と見れば、全体、つまりハードウエアとソフトウエアの中身について最小限の知識に止めておいて、使い方を覚えるだけで済ますことができます。コンピュータに何かの仕事をしてもらうには、プログラミング文で作文して、それをコンピュータが理解して実行できるようにする手続きが必要です。この過程（プロセス）は、コンピュータを擬人化し、例えば、児童に接するように教育することと捉えることをします。コンピュータの側には、比喩的に言えば保護者が居て、児童に対する日常的な知識は教育済みであるとします。この保護者は、マイクロソフト社のようなコンピュータのオペレーティングシステムの提供者であって、かなり手ごわいパワーペアレントです。プログラミング文は特別な文章構造です。プログラマは、それを外国語として覚え、その外国語を使って作文する技術を学習し、コンピュータにその言語で話しかけ、仕事をしてもらいます。コンピュータはアメリカ主導型で研究・開発されてきた経緯がありますので、プログラミング言語の多くは、アメリカ英語の方言(dialect)の性格があります。英語風で書かれたこの方言は、英米人であっても、特殊な単語と文法を持った言語として覚えなければなりません。ただし、英米人は、日常言語が英語ですので、マニュアルなどの英文説明書（レファレンス）は、英語の言語習慣を常識とした省略があります。しかし、日本人は、英語そのものが外国語であるハンディキャップを持ち、その上で特殊な方言を学習しなければなりません。

7.1.2 英語のマニュアルを訳しただけでは分からない

コンピュータ全体の勉強は、英語で書かれた元の参考書を日本語に翻訳したものを見ることから始まります。元の英語版の説明は、声に出して理解できる文章で書かれています。プログラミング言語の方は、発声を目的とした言語ではありません。英米人は、日常言語と少し違う方言として発声することもしますし、方言の元にしている言葉も理解しています。専門的な意義で限定して使うときの方言は、元の言葉または同義語(synonym)の意味と違う場合があります。スペルを部分的に変える場合もあります。ASCII のような頭字語(acronym)に造語することもあります。同義語は、単語としては別の言葉であっても、意味的には同じである場合の言い換えに使います。一つの言語習慣の環境では、特に説明しなくても感覚的に理解できる場合があります。しかし、英語環境内での言い換えは、日本人に直ぐには分かりません。専門用語が別に増えたと解釈することもあって、説明の本質を理解することができないことが起こります。例えば、コンピュータに仕事をしてもらうことはプログラムの実行です。英語表現は、これを動詞の命令形で使います。その動詞には、call, compute, do, execute, go, perform, proceed, run などを使い分けています。これらは、日常 普通に使う言葉ですので、英語の用語説明には載りません。それを受けた日本語の用語説明にも載りません。英語の知識が少ない初心者ユーザは、ここで最初の混乱を味わいます。

7.1.3 一つのプログラミング言語に執着しない

メタ言語(meta-language)と言う用語があります。「言語を作る言語」という意味です。プログラミング言語のメタ言語は、日本人から見れば英語です。一つのプログラミング言語を覚えると、他の言語は全く別言語として覚えなければならないと感じます。しかし、英語の環境では、方言違いと理解しますので、表現違いの書き方が混在していても、定義に矛盾しなければこだわりません。種類の違う言語は、利用目的に合わせて実質で使い分けます。バージョンの改定や、グラフィックス言語などの機能の追加に伴って、言葉の種類が増えて行きます。言語の提案者は個性を主張する傾向がありますので、言い換え版も増えます。そうすると、標準化した言語を決めたいとする動きになります。さらに、主導権を握るための水面下の闘いが起こります。日本語の環境では、或る一つのプログラミング言語にマニアックに執着するプログラマが出る傾向があります。劣勢側に与する（くみする）破目になると、不便を強いられます。こちらは、或る程度の知識を持ったプログラマが味わう混乱です。

7.1.4　外国語を覚える定石

日本は島国ですので、他言語を話す外国人との話し言葉での交流の機会は、多くありませんでした。言葉を覚えるのは、耳で聞いて、相手との対話が基本です。明治以降、海外文化の吸収は、主に、欧米言語で書かれた書物の文字並びからでした。プログラミング言語の勉強は、話し言葉としての勉強を必要としません。見本のプログラミング文書を理解する一つの定石は、単語の意味を覚え、文法を理解し、実際にコンピュータを使って実行して納得することから始めます。このときの参考書は、単語の辞書と文法書に当たる language reference です。この勉強方法は、従来の英語教育で言えば、**英文和訳**の段階です。しかし、このマニュアルが素直に理解できない理由があることを、前々項で説明しました。プログラミングは、プログラミング言語を使う作文、言わば、**和文英訳**に当たります。このとき、何をしたいかを日本語で発想して、作業計画を論理的に組み立てます。和文英訳が正しくできたことの当否は、その作文をコンピュータが正しく理解して実行したことを確認することで行います。プログラミングに挑戦する前に、対象としている専門について、かなりの知識が必要ですし、さらに、日本語でも正しい文書が作文できる教養が重要です。元になる日本語での発想が貧弱であれば、良いプログラムもできません。

7.1.5　用語の意味と使い方に注意が必要であること

市販のコンピュータ関係の参考書を見るとき、英単語を日本語訳にしないで、カタカナ用語にしたものの意味を理解することに、多くの努力が必要です。英単語をそのまま書くと(spell out と言います)、未だ手掛かりがありますが、USA などのように頭文字だけで構成した**頭字語**(acronym)が厄介です。ありきたりの単語も、専門によっては特別な定義を持たせて使うことがありますので、英米人にも説明が必要です。コンピュータ関係で使う用語は、３種類を使い分けます。

１：人に説明するときに使う専門用語；コンピュータ技術はアメリカ主導型で開発されてきましたので、用語の大部分は元がアメリカ英語です。漢字を当てて分かるように翻訳した専門用語も工夫されています。しかし翻訳まで手が回らないので、カタカナ語で使う例が増えます。元のスペルが分かれば何とか意味をたどることもできますが、そうでないときは何も分かりません。日常会話で普通に使われている単語は専門用語ではないのですが、日本語の環境では特別な用語になることがあるのが厄介です。これらの英語用語は、日本ならば日本語、中国ならば中国語のように、自国の言語に翻訳した用語との対訳**辞書**(dictionary)が必要です。さらに、これが、どのような場面で、どのような意味であるかの**用語説明**(glossary)も必要です。こちらは、辞書または辞典ではなく、事典と書き分けます。英語のコンピュータ辞書は、glossary の形で編集されていますので、日常、普通に使う単語は含まれていません。日本語で編集されるコンピュータ辞書は、事典として編集されています。見出しに、漢字用語だけでなくカタカナ語も使いますが、元の英語の綴りも載せて辞書としても使えなければ片手落ちです。

２：コンピュータを人に見立て、コンピュータ本人が理解する用語；この代表がプログラミング言語です。こちらは元の英字スペルで使わなければなりません。これが英語の方言ですので、日本人にとっては厄介な代物です。アイコンとマウスを使うインタフェースは、言葉が分からなくてもコンピュータを使うことができる方法です。これが **GUI**(Graphical User Interface)です。文字を入力することでコンピュータに知らせる方式は、GUI と区別するため、後から **CUI**(Character User Interface)の用語ができました。古くは、Man-Machine-Interface とも言いました。事務処理では女性が多く働きますので、Man を使わない User Interface の方が、抵抗がないようです。

３：コンピュータを使う立場での、専門ごとの固有の用語；これも、日本語に翻訳した用語と、元の言語との関係が分かる事典が必要です。英語だけでなく、ドイツ語が原義である用語もあります。なるべく統一した用語にしたいとする努力は、専門ごとの団体や学術的な学会で試みられています。定義が決まった用語を、変数名などに利用します。これらの専門用語は、専門を大分類とした中での**下位語**です。同じ単語を別の専門で使うことがあります。英語の例で言うと、beam は物理学で光の束として使う場面と、構造力学の梁の意味とがあります。したがって、一般的な利用で使う場面では、専門別の**上位語**も加える必要があって、集合名詞の階層構造を構成します。

7.2　言語が使われる環境

7.2.1　対話する場所を意識すること

話し言葉は、場所と場面によって言葉遣いが変わります。身分制度が厳しいと、上下関係で言い方が変わります。民主主義の社会では、極端な敬語や謙譲語を使うことはそぐいません。しかし、仮に身内であっても、乱暴に過ぎず、丁寧な言い方が望ましいのは当然です。ただし、言い方次第で丁寧にも乱暴にも聞こえますので、書き言葉では表現できない作法があります。人と人との対話では、双方で了解済みのことと思い込んでいることは、あえて言葉に出すことをしないことがあります。また、レトリックもあって、表現と意味することとが違うこともありますので、相手が同じように了解しているとは限りません。これが時として誤解を生みます。コンピュータとの対話は、この面倒な使い分けがありません。この対話では、デフォルト(省略：default)は、決められた初期値があることを意味していて、最初から無いのではありません。ユーザがコンピュータと付き合うとき、かなり多くのデフォルト値は、ユーザの眼に触れません。コンピュータの使い方が難しいことの理由の大部分は、デフォルトの部分の意味と機能とが分からないことにあります。コンピュータの利用が一般的でなかった1950年代までは、コンピュータを使う場所が閉鎖的な環境でした。そこで使う言葉は、主に専門用語だけでした。パソコンが一般事務に使われ、また、家庭での利用にも普及するように環境が変化してきました。当然、用語も変わってきます。

7.2.2　コンピュータ側が理解する語彙を知っておく

コンピュータは、一種の電子機械装置であって、初期にはmachineとも呼ばれ、手動で操作するキー・ボタン・スイッチなどが並びました。コンピュータの心臓部が CPU(中央演算処理装置)であって、ここは手動で直接操作のできない電子的なスイッチの集合です。データの入力用と計算結果の出力用の機械装置が大きな空間を占めていました。この全体を手動スイッチなどで制御することは物理的に不可能ですので、キーボードからの文字信号の並びを工夫して、スイッチ操作に代える通信方法が工夫されました。初期のコンピュータは、電報の送受信に使うテレタイプライタ(Tele-Type-Writer: TTY)を使いました。コンピュータを起動させると、オペレーティングシステムが実行されます。単にシステムとも言い、これもプログラムの一種です。システムは、「あらかじめ決めてある」単純な文字列の入力を待ちます。この文字並びをコマンド（命令語：command）と言います。これによる実行が済むと、再び待ち状態に戻ります。これをコマンド起動型(command driven)のシステムと言い、DOS(Disk Operating System)がその代表です。また、CUIとも同義です。コマンドの機能は、個別に実行形式のプログラムファイルになっています。この機能が、コンピュータ側の持つ知識に当たります。一方、Windowsのシステムは、GUIが使われます。実は、ユーザがモニタ上のアイコンやメニューをクリックして選択すると、コマンドの文字列を内部的に転送するように変更したものです。したがって、Windowsのシステムでは、マウスを使わなくても、すべてキーボードからの文字列入力でコンピュータを制御できます。コマンドをキーボードから入力する場所は「ファイル名を指定して実行」です。

7.2.3　マウスとキーボードの併用作業は避けたい

パーソナルコンピュータ（パソコン）は、欧米ではプロ級のタイピング技能を持った女性秘書のツールとして普及してきた経緯があります。タイピング作業は文字入力ですので、キーボードから手が離れる操作が間に挟まる作業手順を嫌います。マウスだけを使う文字入力、タッチパネル式文字入力は、銀行のATMの操作画面に応用されますが、操作性はよくありません。プロ級タイピストが扱う特殊操作は、CTRLキーと他のキーとの組み合わせで入力するキーショートカットを使います。SHIFTキーは、キーボードの左右にあって、小指を使う位置にあります。CTRLキーも、欧米で使うキーボードは、左右に、SHIFTキーの近くにあって、小指を使う位置にあります。キーボード全体の寸法は、手の寸法と関係しますし、キーを見ないで操作するBlind Touch Typingを考えた簡素なレイアウトにします。図7.1は、IBM社が提案したキーボードのデザインです。これらは、アルファベットを使う言語環境から育った習慣で提案されたことを理解しておきます。最近では、携帯電話でメールの原稿を作成するときは、片手だけで文字入力ができるような入力方法も工夫されました。また、電子辞書や関数電卓は、小寸法のキーボードが付き、それなりの便利さもありますので、これらを使い分ける時代になりました。

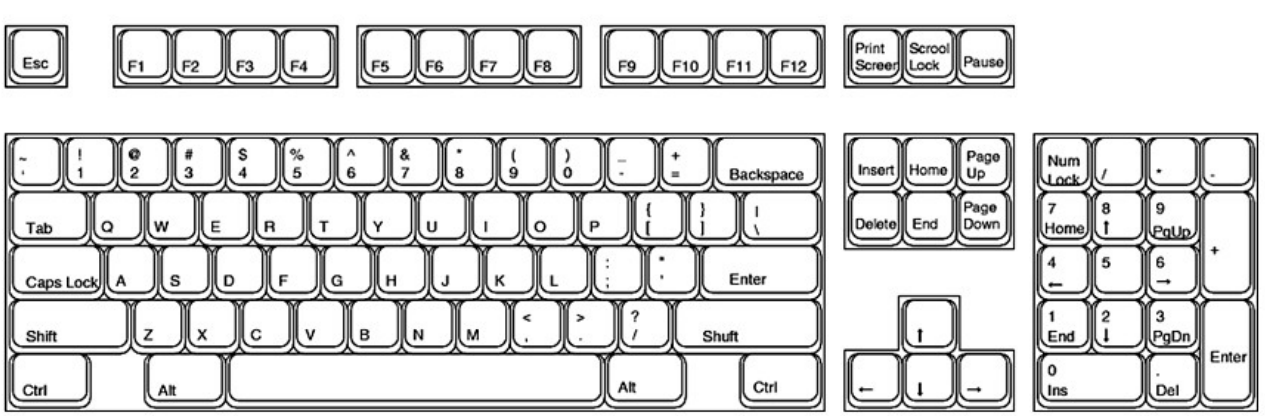


図 7.1 IBM 社が提案した 101-key keyboard のキーレイアウト

7.3 動詞の意義と使われ方

7.3.1 自動詞と他動詞とを意識すること

システムが CUI の環境にあるとき、ユーザは、コマンド用の単語の正しいスペルと、それがどのような実行を伴うかを知っていないと、何も仕事ができません。コンピュータ側で理解できて、あらかじめ用途を決めてある単語を、**予約語**(reserved word)と言うことがあります。コマンドは、コンピュータ側が持っている知識を引き出すために、ユーザがコンピュータに語りかける文字並びです。コマンドは、英語の習慣から言えば、動詞の意義を持つ予約語を先頭に置いた、コンピュータに対する命令文です。この動詞は、原則として**他動詞**です。日本語では、動詞が文末にきます。また、命令文の標準的な言い方がありません。命令的な言い方は、「誰かにしてもらう」使役の表現があります。コンピュータを無生物として主語に立てる構文は、命令の意義を持つ使役の相を取ることができません（第４章、表 4.5 参照)。翻って、この章の最初に述べたように、コンピュータを擬人化すれば、他動詞を命令文として使うことは、矛盾しません。コンピュータ言語で使われる動詞は、**自動詞**としての使い方をしません。コンピュータが勝手に（自動）動作を始める機能は無くもないのですが、英語の自動詞の go, run を使ってプログラムを制御させるコマンドを使う例があります。このときは、ユーザを主語に考え、例えば使役動詞の Let を命令形で使ってコンピュータに何かをさせる構文を考えています。しかし、通常は Let を使いません。プログラミング言語のマニュアルには、動詞的に使う語の通称を、コマンド以外に、アクション(action)、マクロ(macro)、メソッド(method)、プロシージャ(procedure)、ステートメント(statement)、などと言い分けています。コマンドと言うときは、単独に実行できる小単位のプログラムの性格を持ちます。動詞的に使う他の語は、或るプログラム言語の中で、他の条件と関連があるときに使い、部分プログラムの性格があります。

7.3.2 目的語の並べ順で混乱が起こる

他動詞は、基本的に**直接目的語**(object)を一つ必要とします。そのときの環境によっては、目的語がデフォルトになっていて、動詞用単語一つだけを使う場合があります。英語では、同じスペルの語を名詞にも動詞にも使います。命令文として使うときは、誤解しないように、do や execute など、明示的に命令を示す動詞を文頭に補助的に付けることもありますが、普通はこれも省きます。「やり・もらい」のときに使う**授受動詞**は、二つの目的語を取ります。日本語の標準は、直接目的語は助詞の「を」を取り、間接目的語は「に」を取ります。日本語では、目的語の語順を変える言い方であっても、助詞が助けますので、意味を間違えません。英語の授受動詞に当たる move, give などでは、目的語の並び順の標準形は、他動詞→間接目的語（に）→直接目的語（を）の順です。例えば、「give him the book」です。しかし逆順にする言い方もあって、そのときは前置詞が入り、「give the book to him」です。この使い分けは、言語習慣として覚えます。目的語以外に、説明を補う語や句があると、見掛け上、複数の目的語が並びます。文章で表すときは、前置詞などと組み合わせますので誤解しません。プログラム文にするときは、前置詞を省くこともしますので、語の並び次第では論理的な関係の理解が混乱することが起こります。この例は、代数式の表し方を説明する後の項で取り上げます。

7.3.3　階層的な環境という概念がある

或るプログラムが実行状態にあって、コマンドを入力するとプログラムの状態が変わり、そこで、さらにユーザに対して別系統のコマンドやステートメントの入力待ちになるような階層的な状態にあるとき、環境(environment)と言う概念を使います。最も上位、(または基底) にある環境を構成するのがオペレーティングシステムです。Windows の環境から、DOS の環境に移行すると、DOS のコマンドを実行できます。何かのプログラミング言語を使って実行形式のプログラムを作成するときは、作業を効率的に進めるため、**統合開発環境**(IDE: Integrated Development Environment)に移行して、この環境での作業を**プロジェクト**(project)と言うことがあります。環境が変わると、固有のコマンドを別に持つことがあります。そのコマンドが実行されると、さらに別のコマンドを利用する環境に移ります。GUI では、メニューをクリックするとサブメニューの候補が現れ、さらにサブサブメニューの選択が要求されるような状態です。CUI の場合には、コマンドの単語に続けて、複数の引数が必要になることに対応しています。この単語並び全体が文(statement)であって、プログラムの最小構成です。文の区切り記号は、改行が普通ですが、区切りに別の符号文字(delimiter)を使うこともあります。BASIC 系のプログラミング言語は、コロン「:」が使えます。一方、C言語では、セミコロン「;」だけが区切り記号であって、改行はスペース（空白）扱いです。改行して、論理的に次の行に１文が継続しているときの表記法も決められています。一つの文単位の実行が済めば、当面の環境に制御が戻ります。

7.3.4　コマンドやステートメントの集合をファイル化する

コマンドやステートメントを実行する環境は、文字列を解読して、内部データに変換してから所定の処理単位に制御を渡す文字処理プログラムを、内部的に持っています。これを**インタプリタ**(command interpreter)と言います。8 ビットのマイクロコンピュータ（マイコン）に搭載されていた BASIC インタプリタは、使い勝手が良かったので、結果的にマイコンの爆発的な普及からパソコンの大衆化へと発展した歴史になりました。インタプリタ形式のプログラムは、対話的(interactive)に利用できる即応性が便利ですが、HELP の機能が使えない環境であった時代は、コマンドを忘れたユーザが、キーボードとモニタを前にして立ち往生することがありました。そこで、コマンド文字並びの集合を、別作業でテキストファイルに作成しておいて、このファイルをコンピュータに読ませ、まとめて実行させる方法も使います。このファイルを**バッチファイル**(batch file)と言います。大元のオペレーティングシステムの環境にあるとき、実行形式のプログラムファイルの拡張子は(.exe)です。拡張子に(.bat)が付いたバッチファイルは、これも実行プログラムファイルであるとシステムが認識します。この中身は、複数のプログラム単位をさらに集めたプログラム（集合）の性格がありますので、**メタプログラム**と言うことができます。

7.3.5　数値計算を表す文書表現形式が二通りある

コンピュータの基本的な使い方は、数値計算です。電卓（電子式卓上計算機；calculator）は、最も単純なコンピュータであって、すべての操作が手動のキー入力です。プログラミング言語を使うときは、数値計算の手順を文書に書き、それをコンピュータが解読して計算します。そのときの文章スタイルは、歴史的に言うと COBOL 流と FORTRAN 流があります。計算手順を文章で表現する方法と、代数式で書く方法です。数学で言う代数式とは、「**数**に**代**えて」記号と文字とで計算手順を表す方法です。基本的には文章表現順に並べます。また、式をそのように読みます。例えば、代数式「Y=A+B」は、「Y is equal to A plus B」のようです。COBOL 流は、動詞 Add を頭に置いた命令形でコンピュータに指示する「Add A to B giving Y」です。全く異なった表現方法に見えます。代数式の場合は、「Let Y be equal to A plus B」が正式の言い方であって、頭に置く命令形の動詞 Let が省略されたと解釈します。数学では「Y=A+B」と「A+B=Y」とは同じ意義があって、等式と言います。FORTRAN では「Y=A+B」を代入文(assignment statement)と言い、右辺の(A+B)の部分を式(expression)と言います。COBOL 流、FORTRAN 流、共に、一つの文で表しますが、処理は、計算と代入の二つの単位から構成されています。処理の流れで見たとき、計算結果を Y へ代入する処理は、データの移動です。文章並びで見ると、FORTRAN では右から左への逆順、COBOL では左から右への素直な順の移動です。素直と言うのは、電卓でキーを押す順と同じだからです。日本語の表現で言うと「Y は A 足す B である」「A 足す B が Y になる」に当たります。後者の言い方が、論理的には素直です。語順並びと処理の流れとが逆になるのは英米人も混乱するようです。

7.3.6　動詞の代わりに演算子記号を使う場合

算術の四則演算（加減乗除）の英語は、動詞の Add, Subtract, Multiply, Divide です。COBOL は、これらの語が予約語であって、演算を指示しする命令文として使います。FORTRAN では、動詞を演算子記号「+, -, *, /」で表します。演算子を使う式の表現は便利ですので、COBOL でも、また、ほとんどのプログラミング言語で使うことができます。ただし、英米人は、式の中にあるこれらの演算子記号を声に出して言うときは、動詞に直して言う習慣があります。コンピュータは、四則演算のほかに論理演算と比較演算ができることが、電卓と異なります。COBOL は、最初から予約語を使う文章表現で演算処理を指示していました。テレタイプライタのキーボードのコード系(TTY コード)では、<, >, ^などの活字の字形が使えなかったこともあって、FORTRAN では演算子記号で表す方法に工夫が必要でした。例えば、COBOL では、AND, OR, GREATER THAN などと使うところを、FORTRAN では「.AND.」「.OR.」「.GT.」のように演算子を決めました。IBM 社が、図 7.1 に示した"101-key keyboard"を標準化したことから特殊記号<, >, ^なども使えるようになりました。

7.3.7　割り算の COBOL 文が二通りある

日本人は、割り算の代数式「C=A/B」を、言葉では「A 割る B は C」のように言います。COBOL も、この代数式表現を使うことができますが、文章で表すときには二通りの書き方があります。

書き方１：　Divide A by B giving C
書き方２：　Divide B into A giving C

下の書き方２の方は、日本人には馴染みがありません。おまけに割り算の被除数・除数の位置関係が書き方１と逆です。英米人は、どちらも普通に使うのですが、この区別は、前置詞を含めた句動詞(phrasal verb)に一単位の動詞機能を持たせる言語習慣からきています。目的語を動詞と前置詞の間に挟んで語順を変える言い方は、逐語訳をしている限り、理解できないでしょう。句動詞は、前置詞違いで別の動詞の機能になりますので、直接目的語（を）の捉え方が異なります。

7.3.8　プログラマが決める動詞的な用法がある

上の割り算の説明で使った、Divide は予約語です。一つの文ですが、コンピュータの処理は、演算と結果の格納との二つの単位から構成されています。引数の文字記号を３つ使いますので、この文に先行して A, B, C の名前と型の約束（宣言）をしておき、さらに A, B には数値を代入しておく処理が必要です。つまり、Divide は、動詞の命令形の意義を持ちますが、コマンドとはしません。プログラミングをするとき、プログラマ側は、何かの処理名を決めて、動詞の命令形の形で使いたいことがあります。日本語で言うと「何々する」とする動詞的に使うときの「何々」に当たります。この目的のためのプログラミングが、**サブプログラム**の作文です。命名はプログラマの裁量です。例えば、上の割り算処理をサブプログラム化したいとき、Visual Basic ならば、次のように書きます；

```
Public Sub WARIZAN(A as Integer, & _
      B as Integer, C as Double)
        C=A/B
End Sub
```

これを別のプログラム単位から呼び出す場合には、Visual Basic では二通りの書き方が使えます。

```
書き方１：　Call WARIZAN(ValueA, ValueB, ValueC)
書き方２：　WARIZAN ValueA, ValueB, ValueC
```

書き方１は、予約語の Call を文頭に置きますので、WARIZAN が予約語ではないことが分かります。書き方２では、予約語であるかないかは、すぐには分かりません。予約語は、プログラミング言語本体が持っている単語です。しかし、商品としての IDE の環境では、ユーザの便宜を図る目的で、定数などがライブラリ化されていることがあって、断りなしに使える名前があります。例題のプログラムを見て勉強しようとするとき、その名前がどこで宣言されているのかが分からなくて、結果的にプログラムの構成が理解できないことも経験します。

7.3.9　単語の文字数に長さの制限があること

ユーザがコンピュータに指示を出す文字並びは、なるべくなら、その場面に合った説明的な文章になるのが理想です。コンピュータ側では、文字並びの解読処理が必要ですので、その作業効率を上げるための規則を作ります。その一つは、単語の長さを制限することです。一つの単語として扱う名前は、標準として英字を頭に置いて、英字と数字との文字並びで表します。数学では、変数を表す文字に、英字またはギリシャ文字１字を当てるのが普通です。これを大文字・小文字・斜体などで使い分けます。多種類の変数名を必要とするときは、下付きの数字や文字を付けます。初期の FORTRAN は、7 文字以上の長い単語名を扱うことができませんでした。したがって、短い変数名であっても何とか意味を表すことができるときは、方言として理解できます。そうでないときは、暗号のようになります。そのため、この文字並びの単語が何を表しているのかの説明（コメント文）を丁寧に作成することが推奨されていました。COBOL では、変数名などの意味が分かるように、記号化しないで、長いデータ名を直接使う方法を推奨しています。複合名詞を使いたい要望が増えてきましたので、一語に繋ぐとき、間に記号を置く約束を持つプログラミング言語も増えてきました。COBOL ではハイフン「-」が使えます。ただし、単語の最大文字数が 32 までの制限がありました。最近のプログラミング言語では、文字解読処理の部分の機能が拡充されて、かなり長い文字並びの単語も使えるようになりました。

7.4　データの定義と名前で宣言する儀式

7.4.1　変数に固有の名前をつけるとき

コンピュータが処理の対象とするデータは、動詞の目的語です。動詞名の方は、プログラミング言語の方で決めてありますが、変数名はプログラマが決めます。データ種類としては、定数と変数です。数のデータには型の区別があって、大別して整数と実数です。これに文字データを含めます。文字は文字コードがありますので、数に準じた扱いができるからです。プログラミング言語側は、利用できる数と文字について型の**定義**を持っています。平たく言えば、この型が使えますと言う約束ですので、型名が予約語です。例えば、INTEGER, DOUBLE などです。どのような型が使えるかは、プログラミング言語ごとに固有です。整数・実数の言葉自体は、単数形の**普通名詞**です。プログラミングでは、プログラマが定数名または変数名を決めて（**宣言**して）から、その名前で処理に組み込みます。その名前は**固有名詞**です。配列名は**集合名詞**を宣言することに当たります。その中身の要素は、個別に固有名詞を決めるのではなく、番号(インデックス：index)を付けて区別する方法を使います。宣言をした時点で始めて、その定数または変数の実体が、メモリの中に確保されます。したがって、使わなくなったときは、不要の宣言をする命令を持つことがあって、そのメモリ領域を別の目的に使うことができます。宣言なしに或る名前を使ってプログラム文を書くと、その名前は存在していない、または使い方が間違っていると言う警告エラーが通知されます。論理的な正確さを確保するためには、宣言の手続きが必須です。しかし、一般のユーザがプログラミングに挑戦するときには面倒な儀式です。

7.4.2　二次元配列は階層的な関係にある集合を表す

数値計算を主題としてプログラミングをする場合、連続した性質のある関数を、飛び飛び(離散的：discrete)の座標で扱います。そのデータ並びを格納する領域を、配列名で宣言します。普通は一次元的な並び構造を扱いますが、マトリックスを格納するときには二次元配列を使います。三次元以上の配列を扱う場面は多くありません。二次元以上の配列であっても、メモリの使い方は内部的には一次元の並びです。これを視覚的に表で表すとき、行と列の要素をどの順に並べるかの約束をします。これは、表の二次元的な座標位置(index)の番号付けと表記の約束に関係しますので、プログラミング側で定義を持っています。数値計算を主目的としたプログラミング言語の FORTRAN, BASIC では、例えば A(M,N)と表記します。M 行 N 列の要素は、メモリ内では列の並びを単位とします。列の要素数が N 個ですので、論理的には N 個単位で区切られます。一方、C 言語では、配列を A[M][N]と書くと、マトリックスの要素は、行方向の M 個を連続単位としてメモリ内で並びます。これは、横書きの文字並びを格納するときには扱いやすい方法です。一次元配列を集合とすると、二次元配列は集合の集合と考えることができます。配列全体に名前を付けて宣言しますが、個別の要素は、データ型がすべて同じですので、行または列に名前を割り当てることをしないで、インデックスを使って参照します。

7.4.3　集合の集合と見出しの名前

上で説明した数表と対照的な表の使い方はEXCELの表に見られます。例として個人の住所表記のデータを作成することを考えて見ます。データ要素は、姓名、男女の区別、年齢、住所、電話番号、E-mailのアドレスなどのように型の違うデータの集合が行方向に並びます。住所で言えば、例えば、「愛知県 名古屋市 名東区…」のような並びで表します。「県 市 区…」が階層の見出しに使われる集合の名前、「愛知 名古屋 名東…」はデータですが、固有の名前です。これを、表計算ソフトEXCELで使うような表の形にまとめることを考えます。列の見出しに「県、市、区、…」と書きます。英語の言い方に当てると、見出しは単数形の「prefecture, city, ward, …」にします。住所表記では、県の見出しの列に、愛知が列方向で重複して現れますので、別に県名を集めた表を作るとします。この表の見出しは、日本語ならば「都道府県」とするでしょうが、英語方式では prefectures のように複数形を見出し語にします。これは、集合名詞の意義です。英語では普通名詞を使うときには、単数・複数を神経質に区別し、それを受ける動詞形も変えます。県単位で見ると、県内には複数の市がありますので、県名は単数形で表記しても、市名の集合を扱う集合名詞の性格があります。「都道府県」と言うときは、県の集合を言うときの言葉です。これは集合名詞のさらに集合を表す名前です。英語の場合、単数形の prefecture を市の集合を意味すると解釈させると、複数形のprefecturesは集合の集合を意味します。これをprefectureの collection（コレクション）であると説明することがあります。日本語では、関東・中部・関西の地区名は、幾つかの県の集合(collection)の意義ですので、個別のコレクションに名前を付けて区別しています。

7.4.4　下位の集合の構成要素に同名が現れることがある

市名に続く区名は、さらに町名や番地を下位集合として持ちますので、これも集合名詞の性格があります。番地やマンションの棟号はさらに下位の集合であって、数字を当てますが、こちらは配列に作成したインデックス番号に当たります。区名は、その市では一意の名前であっても、全国的に見れば同名があります。例えば、港区は、東京、名古屋、大阪にあります。したがって、市名までを省略して港区と使うときの注意が必要です。コンピュータの用語では、名前が一意に区別できる範囲のことを、scope（スコープ）と言います。それがglobalであるか、localであるかを決めます。全国区と地方区のような意義です。その名前がlocalであるときは、それが所属する集合の一段階上の集合名を形容詞的に付けます。その名前は、一段階上の集合名のmember（構成要素）です。階層的な関係で繋がった集合の中で、ある個所のデータ名を明示的に表す書き方には種々の方式があります。階層的な関係にあるときは、日本語では「何の何」のように「の」を挟む言い方をします。「愛知の名古屋の名東」のような言い方は、大きな集合から小さな集合への順です。しかし、欧米では住所の表記が日本と逆順です。英語では「の」に当たる前置詞「of」を使って、例えば「Nagoya of Aichi」の言い方もします。しかし、擬人化して所有格にするアポストロフィを付けて、「Aichi's Nagoya」の言い方も許容することがあります。

7.4.5　C言語には構造体というデータ構成法がある

EXCELまたはACCESSを標準的に使って表の形式にデータを格納することを想定して、このデータをプログラミング言語で扱いたいとします。エクセルの表一行分は、レコードと言うことがあり、一単位のデータ集合です。データ型は列要素によって異なるとします。C言語では、型がそれぞれ異なったデータの集合を一単位で扱うように、この全体に名前を付けた**構造体**を宣言して利用することができます。配列に構成して使うこともできますので、複数行の表形式のデータを扱うことができます。構造体単位で参照するときと、中身の構成要素（メンバ：member）を参照するときとの区別が必要です。プログラミング文では、日本語の「の」の使いかと同じように、表記する方法に種々の工夫があります。標準では、集合の大きい方から小さい方の順です。これに、小数点を挟む方法が使われるようになりました。例えば「Aichi.Nagoya」です。また単独に命名する必要があるときは、小数点を省いて、覚えやすいAichiNagoyaの造語法も使います。この場合、英字並びは大文字と小文字とを使い分けますので、眼で見て語の使い方の正否を判定するようにもなってきました。しかし、E-mailのアドレス表示などでは、英語の習慣では逆順表記「Nagoya.Aichi」が普通です。これらの表記法の使い分けは、英米人も混乱するようですので、その分、マニュアルの説明がくどくなります。C言語では、構造体のメンバを参照するとき、「Aichi->Nagoya」のような書き方の約束も定義されています。なお、Visual Basicには、構造体に相当するデータ型の定義がありませんので、特別なデータコントロールを介して、データ材料とする元のEXXELまたはACCESSの表を利用します。

7.5　形容詞の扱いが必要になったこと

7.5.1　GUI の環境を構成するプログラミング

オペレーティングシステムが Windows のような GUI を採用するようになって、コンピュータは、数値計算の道具としての使い方から、擬似的な装置（シミュレータ：simulator）の使い方が標準になりました。グラフィックスモニタ上に種々の擬似的な装置（オブジェクト：object）の図柄を表示して、主としてマウスを使って選択をすることでコンピュータとのインタフェースを取ります。ここで使う用語のオブジェクトは、元の英語の意義の「物」の意であって、動詞の目的語と同じ原義です。日本語に訳した文法用語の目的語は巧みな訳語ですが、動詞が対象にする「物」が原義です。ただし、対象には人も生物も、また抽象的な事象も含みます。グラフィックスモニタ上のオブジェクトは、それを表示する作図データと、どのような機能があるかの定義を利用して、ユーザインタフェースのプログラミングを追加しなければなりません。Visual Basic 6.0 (2010 年時点)では、従来の方法で作成したプログラミングのソースコードを**標準モジュール**と言い、フォーム単位でオブジェクトの扱いをするソースコードを、**フォームモジュール**として別に追加する方式になりました。簡単な処理は、フォームモジュールだけで済ますことができます。オブジェクトの種類が多くなりましたので、Visual Basic の参考書は、オブジェクトを使うプログラミング（オブジェクト指向プログラミング）を主題とするようになって、基本的な標準モジュールのプログラミングの説明が丁寧ではなくなりました。

7.5.2　オブジェクトの定義と宣言の儀式がある

基本的なオブジェクトの図柄は、システムの画面でも使いますので、オペレーティングシステムが標準として定義を持っています。特殊なオブジェクトは、別のライブラリにまとめてありますので、フォームモジュールの中で、それを選択する宣言をします。オブジェクトの定義は、図形としてのデータではなく、文字並びで記述されています。オブジェクトを利用するには、専門的で複雑なプログラミングをして宣言しなければなりません。一般ユーザにはかなりの負担ですので、作業を便利にするツールが統合開発環境としてベンダーが用意しています。Visual Basic では、フォームモジュールの作成は GUI の環境で行うことができます。このとき、特殊オブジェクトのライブラリからの参照方法、オブジェクトの宣言方法のソースコードは、裏で自動的にフォームモジュールに書きこまれ、ユーザからは直接見えないようになっています。最も基本的な図柄は、ディスプレイ装置そのものを擬似的に表示する「物」であって、通称でウインドウと言う枠図形です。プログラミングの用語では、普通名詞扱いをする**フォーム**(form)と使います。モニタ上に複数のフォームを表示できますが、考え方は、大小種々のキャラクタディスプレイ、グラフィックスディスプレイを表示していると考えます。フォームは種々のデザインの図柄を表示する入れ物です。マウス操作に反応する機能を持つ個別の図柄も通称はオブジェクトとくくりますが、普通名詞で**コントロール**または**コンポーネント**と当てています。さらに、機能の特徴を具体的に表す、データ型名に相当する名前が**定義**されています。ラベル、ボタン、ピクチャーボックス、テキストボックスなどがそうです。プログラマ側では、個別に**固有名**をつけて**宣言**した上で、プログラムの中で参照します。

7.5.3　オブジェクトの性質を表すデータをプロパティという

オブジェクトには、種々の性質、つまり情報と機能とを持たせます。これを**プロパティ**(property：属性)と総称します。プログラミングでオブジェクトを参照するときには、プログラマが一意の名前(name)を宣言します。実際の画面上では見出し(caption)を使います。図形としてのデータは、寸法、位置、色使い、などの指定が必要です。デフォルトの文字や数値が決められていますが、プログラマが指定することも、また値を参照することもできます。このとき、階層的な関係にあるデータ集合の呼び方に、前節で説明した、名前の付け方の約束が関係します。寸法や色は形容詞的な性質ですが、客観性がありますので、数値データとして扱うことができます。画面に見えるか見えないかは、(yes/no) で判断する形容詞の性格がありますので、論理数の(true/false)または、その整数表現の(1,0)がデータです。数値化できない形容詞的な性質は、扱いません。例えば、オブジェクトの見てくれに当たるグラフィックスデザイン的な要素は、機能として第一義的な重要性はありませんが、ユーザに好まれるように、システム側の開発者が凝る傾向があります。アイコンの図柄は、もともと、文字が分からなくても利用できる分かり易い図柄がよいのですが、最近は種類が増え過ぎて、何を意味しているのかが分からないことも起こっています。フォームの上段には、アイコンを表示するツールバーと、文字で表記を選択するメニューバーとがあります。ツールバー無しでも作業を進めることができるようになっているのが基本です。

7.5.4　乱数は適当な値を決める感覚形容詞の数値とする

シミュレーションをするとき、適当な数値を代入して結果を見る方法があります。**モンテ・カルロ法**(Monte Carlo Method)は、乱数を使って任意の条件を代入して結果を統計的に判断するときに使います。これは、言葉で言えば、「適当に」または「適当な」に当たります。コンピュータは、原因と結果とが正確に一対一に対応するように使います。理論にこだわる学者は、解析的な式で表すことを好み、確立統計的な手法を嫌う傾向があります。アインシュタインもそうであって、「神はサイコロを振らない」の言葉を残しています。実践技術の問題として、**試行錯誤**の言葉があるように、予測の立てられない事象の解析には、適当な選択をする場面が現れます。これは、適当な選択をする数値的な方法に乱数を使うことでモデル化します。この方法は、数値実験と言うことがあります。感覚に関係した形容詞または副詞をコンピュータが扱う例と考えることができます。ただし、数学的に厳密に言えば、これは擬似乱数と言い、非常に周期の長い、規則性のある関数として作られています。

8. 論理を表す文と式

8.1 文章論理と計算論理

8.1.1 文章で説明するときの言葉遣いの約束を考える

社会生活の場では、相手に何かを説明して間違いなく理解してもらう説得の言葉遣いが必要です。論理学(logic)は、その考え方と方法を研究する学問です。その歴史は、ギリシャ時代にまで遡る、非常に古いものです。中学校で習う初等幾何学の証明方法は、典型的な論理学の応用です。初等幾何学の定理、例えば「三平方の定理：ピタゴラスの定理」の理解については、普通、定理から得られた代数的な公式を応用していて、証明の手順を理解することから遠のいています。この定理の証明には何通りもの方法が研究されてきました。注意することは、古典的な証明は、言葉で論理的に**帰納**(induction)する方法を使うことです。下で説明する記号論理学と区別する意味で、形式論理学(pure logic)と言います。筆者は、**文章論理学**の用語が適当であると考えています。初等幾何学の面白さは、結果の美しさもそうですが、むしろ、補助線を使うなど、一種の修辞学（レトリック：rhetoric）の方法を使う証明過程に知的な興味を持つ人が多いのです。論理学の主題は、論理の構成方法の分析です。論理を踏まえて、具体的な言葉遣いを提案することが、**正書法**(orthography)です。表 8.1 は、普通に使う言葉遣いの中、論理学用語との対応を例示しました。文章論理を、記号に置き換えて表す方法を**記号論理学**(symbolic logic)または**数理論理学**(mathematical logic)と言います。ただし、普通の代数計算用に式を構成する方法とは違いがありますので、特別な記号を使います。表 8.2 は、決まり切った言い回しを記号に置き換える、その幾つかの表し方を並べました。これらの記号式は、眼で見て理解するのですが、声に出して言うとき、その元になった言葉を使うことが普通です。表 8.1 と表 8.2 の左欄にある見出しには、あまり眼にすることのない、特殊な論理学用語（連言、選言、内含など）を使っています。明治以降、欧米の学問を日本語に翻訳するときに、先人が工夫して提案した和製の漢字熟語です。現代ならば、カタカナ語で済ますところです。

8.1.2 コンピュータ相手に言葉で説明する

普通の言葉で、相手に説明して間違いなく理解してもらいたいとき、論理学の用語を踏まえた言葉遣いをします。コンピュータを擬人化し、何かの作業を依頼するときの文書、つまりプログラム文書は、誤解が生じない、論理的な構文だけで構成します。コンピュータによる処理は、数値計算に使うことが主な目的ですので、文科的な学問である論理学とは無縁であると考え易いところです。しかし、意外なことに、プログラミングは一種の作文ですので、論理学的な素養が必須です。演算子を使う数学的論理式で論理手順を説明すると、従来の文章論理学の説明に比べて明快になります。この章は、論理式の考え方を説明することに当てました。プログラミング文書では、レトリック的な要素を省く表現になります。表 8.1 の分類の中にある、推論(8)、理由(9)、許容(10)、命令・禁止(11)、説明文(13)、意見陳述(16)はありません。なお、定義文(12)は、プログラミングの場合には代入文のように、値を確定させる使い方が当たります。

8.1.3 ブール代数の登場が新しい論理学を開いた

ジョージ・ブール（George **Boole**、1815-1864）は、イギリスの 数学者・哲学者です。今日のコンピュータの論理演算を理論的に支える、記号論理学の創始者とされています。ブール代数が扱う数は、従来の数値とは違い、true/false（真と偽）で区別するような、二種の顔だけを持つ量とした定義であって**ブール値**（論理数）と言います。コンピュータメモリの１ビットの状態を表すときに使いますので、便宜的に(1,0)の二値で扱います。論理の繋がりを言葉で言う方法を、代数式を扱うときのように、ブール値の変数記号と演算子記号の並びで置き換えます。基本的な算法は、二つのブール値に変数記号Ｐ，Ｑを当てるとして、ＰとＱとの間に、一つの演算子を介して演算させ、その結果もブール値で得る表し方を考えます。普通の数学算法は、二数間の演算則は４通り（加減乗除）しかありませんが、ブール代数の場合には、実は 16 通りの約束を考えることができます。もう一つ、数値計算の場合と大きく違う論理演算があって、変数一つを対象とした否定の演算則があります。１ビット単位のブール値の演算則を基本として、ビット並びの**集合**を考えて、この集合単位での論理演算にもブール代数を応用します。コンピュータの論理演算は、文字型または整数型の変数をブール値の集合名詞扱いをして、ビット位置単位で論理演算をさせる使い方をします。集合間の論理演算則を、説明図に表したものを**ベン図**(John Venn, 1834-1923)と言います。表 8.4 は 16 通りの演算則をベン図で説明したものです。

表 8.1　論理に対応する日本語

	論理学の用語	日　本　語　で　の　表　し　方　の　例	備　　考
1	肯定 存在 部分肯定	…である、…だ、(英語の be 動詞相当) …がある（無生物)、…がいる（生物) …だけがある、…もある、	用言の終止形一般
2	限量詞 代名詞	すべて（の)、なんでも、みな、つねに、いつも、 或る、任意の、少なくとも、もの、こと、その、おなじ	
3	否定(NOT) 存在否定 部分否定	…ない、…でない、違う、 …がない、必要がない、 とは限らない、必ずしも…ない	
4	否定の限量詞	なにも…、まったく…、全然…、必ずしも…	
5	連言(AND)	…と…、および（及び)、かつ、さらに、ならびに したがって、そこで、そして、そのうえ、それゆえ、ただし、なお、…して、…（し）たり、 ところで、だが、けれども、でも、ところが（逆態) しかし、だが、のに、(逆接) さて、つぎに、では、	連用中止形は文の連言とする。いわゆる「て」フォームも含む。逆接、逆態も論理的には連言。
6	選言(OR)	…か…、あるいは、いずれか、あるいは、または（又は)、それとも、 もしくは（若しくは)	普通は、排反的な意味(exclusive)とする。
7	内含(If-Then) ・条件文の前件 ・条件文の後件	（もし）ならば…、…（する）と、なら、ば、れば、たら …（の）場合、…のとき ときだけ、の限り、…のみ、…だけは、だけである だから…、ゆえに…	 双条件文
8	推論	ゆえに必ず、ゆえにおそらく、当然である、必然である、必ず、みなす、そうだ、 ようだ、らしい、だろう、可能である、あり得る、妥当である、はず、ちがいない、しかない、ありえない、あき(明)らかである	必然的真 可能
9	理由	ために、ためには、だから、…（する）ためである、ので、なぜなら、…からである、…からだ	
10	許容	よろしい、（し）てもよい、…てよい、可能である、できる、 れる、られる	
11	命令 禁止	（し)なければならない、すべきである、必要である、(し)てはならない	
12	定義文	英語の be 動詞に相当する「である」式の文が、定義文。動詞を定義するとき以外に、「もの」や「こと」などの代名詞を使わない。	定義文は双条件文にする。言いかえと混同しない
13	説明文	読む、書く、走る、歩くなど、動作動詞を使う普通の文が説明文。	普通名詞を使うと具体性が低くなる
14	比較文	比べる対象との類似点をあげるやり方（含意、対等）と、相違点をあげるやり方（大小、逆、反対、矛盾）がある。部分的に同じ場合のように範囲が曖昧な場合もある。	「…のような」「… らしい」は、修辞学の直喩
15	副詞、形容詞	少々、しばらく、すぐに、早く、十分に、大幅に、大きい、小さい、非常に、間もなく、急激に、いろいろな、高い、古い、 大勢、至急、よく、たまに、	二値論理に向かないので、数量化して使うのがよい。
16	意見陳述	考える、思う、好き、嫌い、心地よい、痛い、苦しい、眠い、見える、聞こえる、甘い、辛い、臭う、	感覚に関する用語は、主観的な判断である。

表 8.2　種々の論理記号(日本語ワープロで表現できる条件での表記です)

論　理　学　の　用　語	記　号　表　記	演算子	英　文	和　文
否定	~~P~~,　￢P,　～P	NOT, ！	not	…でない
選言、論理和	P∨Q、P∪Q	OR, ｜	or, and/or	または
排他的選言、排反		XOR	exclusive or	
連言、論理積	P∧Q,　P＆Q,　P∩Q	AND, &	and	かつ
内含、含意	P⊃Q,　P→Q		if～then	ならば
可逆的な内含、対等、同値	P≡Q,　P←→Q,　P～Q	EQV, =	equivalence	（双条件文）
非両立	P｜Q			
備考：ブール値の集合に対して行わせる論理和と論理積の記号は、カップ(∪)とキャップ（∩）を使う習慣になっていて、そのように声に出して説明します。記号の（∨）と（∧）は、「または、かつ」と読み変える言い方をします。記号が意味する演算が直観的に分かり難いのが難点ですが、筆者は、∧と∩とは、英語の And のＡと形が似ていることをヒントにして理解しています。				

表 8.3　二値の論理演算則

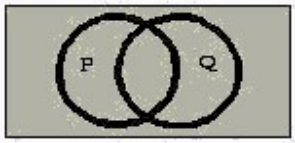
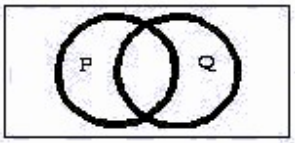
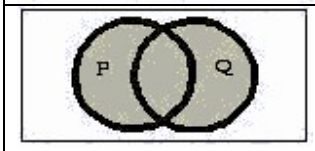
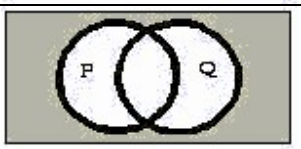
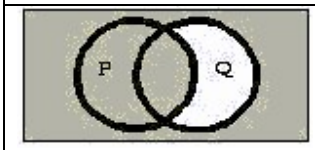
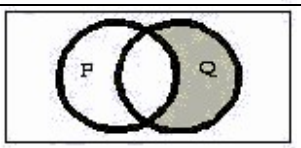
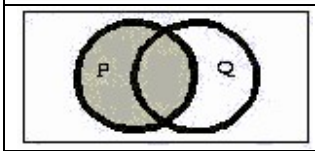
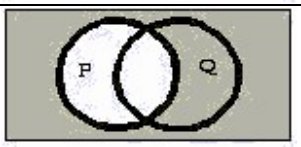
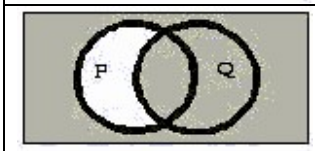
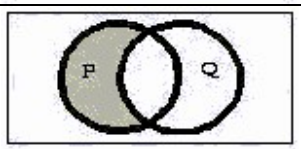
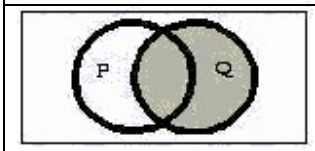
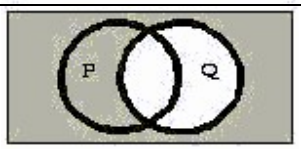
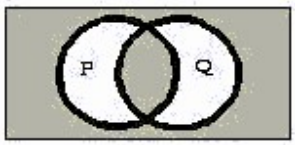
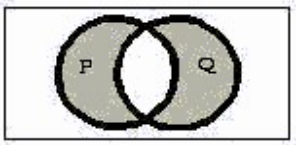
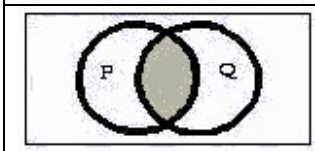
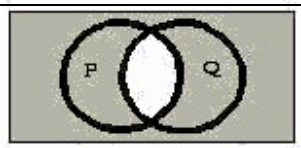
番号	Pの取る値	1	1	0	0	定　義	文章での表し方	備考
	Qの取る値	1	0	1	0			
	演算表記	演算の結果						
1		1	1	1	1	トートロジー	常に真を返す	対称
2	P∨Q	1	1	1	0	論理和、OR	PとQの両方	対称
3	P⊂Q	1	1	0	1	逆向きの内含	QならばPである	
4		1	1	0	0	・・・・・・・・	［常にPである］	
5	P⊃Q	1	0	1	1	内含、IMP	PならばQである	
6		1	0	1	0	・・・・・・・・	［常にQである］	
7	P≡Q	1	0	0	1	対等、EQV	PとQとは同値である	対称
8	P∧Q	1	0	0	0	論理積、AND	PおよびQである	対称
9	P｜Q	0	1	1	1	非両立	PとQとは両立しない	対称
10	P≠Q	0	1	1	0	排他的論理和、XOR	PとQとは仲が悪い	対称
11		0	1	0	1	・・・・・・・・	常にQの否定である	
12		0	1	0	0	（P⊃Q）の否定	「PならばQである」ではない	
13		0	0	1	1	・・・・・・・・	常にPの否定である	
14		0	0	1	0	（P⊂Q）の否定	「QならばPである」ではない	
15		0	0	0	1	（P∨Q）の否定	PでもQでもない	対称
16		0	0	0	0	矛盾	常に偽を返す	対称

［説明］
１：演算子の記号は、日本語ワープロで利用できるフォントを使いました。
２：プログラミングで使うフォントはアスキーコードですので、定義の中の英字表記を使います。
３：論理和(OR)は、文章表現では「PまたはQまたは両方」という表し方をすることがあります。
４：排他的論理和(XOR)は、文章表現では「PまたはQどちらか」の意味で、両方を含みません。
５：内含(implication)は、分かり難い論理演算ですので表 8.1 の方で理解するとよいでしょう。
６：等しいと言う表現をP≡Qと書きましたが、英字では＝記号を代用すると混乱が起こります。
７：番号で後半 16～9 の演算は、前半 1～8 の演算の否定形になっています。
８：備考の欄にある対称とは、演算に使う変数の順序を逆にしても同じ結果が得られることです。
　これは交換則がある演算です。
９：上の表を拡張して、PとQとを否定で使う論理演算則を数え上げることができます。しかし、
　その結果は、上の 16 通りのどれかと一致します。**ド・モルガンの法則**は二つあって、否定形にして論理積と論理和を計算すると、上の表の(9)と(15)と同じ結果になることを言います。

8.1.4　論理演算子の記号については種々の提案がある

算術の加減乗除の演算子記号は、日本語ワープロの環境ならば（＋－×÷）と使えます。プログラミングで使う英語は、キーボードにある記号だけを使うことで制限を受けますので、（＋－＊／）を当てます。16 通りある論理演算則のすべてに演算子記号を当てると丁寧です。しかし、論理和（∨）と論理積（∧）の二つと、否定記号の三つがあれば、他の演算則はすべて誘導できます。ただし、日本語のワープロの環境では、記号文字の上に横棒を付ける方法で否定を表現する方法ができませんので、表 8.2 の否定の表示記号は、中央に横棒を入れる方法にしています。英語の環境は、キーボードにある記号文字種の制限がありますので、例えば Visual Basic にあるように、予約語扱いをした文字並び(Not, And, Or, Xor, Eqv, Imp)を、そのまま演算子記号に使う方法が主に採用されます。C 言語の場合には、(||, &&, !) を演算子記号に使います。説明が後先になりましたが、否定の演算は特殊です。これは、論理変数一個を使うからです。プログラミングの感覚で説明すると、二値を使う演算は、引数を二つ使う関数に組み立てることができます。同じように否定の演算は、引数を一つ取る関数と考えるとよいでしょう。コンピュータが数を内部的に表すときは、ビット並び、つまりビットの集合とします。この集合間の演算記号に、論理和（∪：カップ）と論理積（∩：キャップ）を使います。ただし、表 8.4 では∨と∧を使って説明しました。

表 8.4　二値を持つ論理値の集合間の論理演算則を表したベン図

	(1) ＰとＱのトートロジー		(16) ＰとＱの矛盾
	(2)　Ｐ∨Ｑ ＰとＱの論理和 図形演算として見ると 足し算（Ｐ＋Ｑ）です		(15) （Ｐ∨Ｑ）の否定
	(3)　Ｑ⊃Ｐ 「ＱならばＰである」		(14) （Ｑ⊃Ｐ）の否定
	(4) 「Ｐである」		(13) 「Ｐの否定である」
	(5)　Ｐ⊃Ｑ 「ＰならばＱである」		(12) （Ｐ⊃Ｑ）の否定 図形演算として見ると 引き算（Ｐ－Ｑ）です
	(6) 「Ｑである」		(11) 「Ｑの否定である」
	(7)　Ｐ≡Ｑ 「ＰとＱは同値である」		(10) 論理的選言、排反 exclusive or
	(8)　Ｐ∧Ｑ ＰとＱの論理積 図形演算として見ると 掛け算（Ｐ×Ｑ）です		(9)　Ｐ｜Ｑ ＰとＱの非両立
備考：表の左右のベン図は、互いに否定の関係になるように配置しました。			

8.1.5　交換則が成立しない演算則は理解が難しい

あまり気にしませんが、算術の四則演算の中、引き算と割り算は、交換則(commutative law)が成立しない演算です。日本語の言い方では、助詞を使うことで語順と演算順とを間違えることがありませんが、英語の環境では注意が必要です。前章 7.3 節の後半に、COBOL での割り算の文章表記が二通りあることを説明しました。引き算の場合にも、subtract は他動詞ですので、前置詞の遣い方、または受身形にすると目的語の並び順が変わります。例えば、(A-B) の代数式は、逆順に「subtract B from A」の構文が普通の言い方です。和製英語の感覚では A is subtracted by B の語順の方が素直に理解できます。論理演算では、内含(implication)が交換則の成立しない演算です。数学論理で扱うことは少ないのですが、文章論理では、「if A then B」までの構文単位で一つであって、その演算結果が真であるか偽であるかの判定を言うときです。Ｐ,Ｑを、条件を言う文単位（命題）として「ＰならばＱである」が真（成立）しても、逆順の「ＱならばＡである」が必ずしも成立しない（偽）ことがあります。この区別は、定理などの証明をするときに重要です。コンピュータ言語の中で使う「if Ｐ」までの構文は、二分岐選択の判断に使い、内含の意義ではないことに注意します。While, When、Select も、分岐先を選択する使い方です。内含の演算子（例えば Imp）を特に決めなくても、「論理積・論理和・否定」の演算を組み合わせて同じ処理ができます。

8.2　比較を表す表現

8.2.1　数の大小関係を判定する式がある

我々は、式と言うと代数式だけと考えますが、英語の **equation** は、等号(=)で左右の式を結び付けた**等式**だけを指します。変数記号と、等号を除いた演算子記号を組み合わせた表現を **expression** と言い、これが**式**の正しい和訳です。コンピュータ言語で扱う式は、三通りあります。**代数式**(algebraic exp.)、**論理式**(logical exp.)、**比較式**(comparison exp.)です。代数式に使う変数は数であり、加減乗除の演算子を使い、式の評価も数で得ます。論理式は、変数に論理値（ブール値）を使い、論理演算子を使い、式の評価も論理値で得ます。比較式は変数に数を使い、比較演算子を使って、結果を論理値で得ます。比較式の場合にも、文章での言い方が対応します（表 8.5)。比較演算子は、Ａ，Ｂが数値を表す変数であるとして、例えば（Ａ＜Ｂ）のように表します。(5)と(6)とは交換則が成り立つ演算です。それ以外は交換則が成り立ちません。Ａ，Ｂを入れ替えても、否定になりません。表 8.5 で言えば、(1)と(4)、(2)と(3)とが互いに否定の関係にあります。したがって、比較を表す式では、否定演算を組み合わせて演算の順番を工夫します。これが、論理式を扱う場合のレトリックです。

表 8.5　比較演算子(visual Basic)と対応する文章語句

	演算子	和文	英文	意味と使い方
1	<	未満、より小さい	less than	６未満と言うとき、６を含めない
2	<=	以下	less than or equal to	６以下と言うとき、６を含める
3	>	より大きい、を超え	greater than	６を超えと言うとき、６を含めない
4	>=	以上	greater than or equal to	６以上と言うとき、６を含める
5	=	等しい	equal to	代入文と間違えないように使う
6	<>	等しくない	not equal to	
備考：関係演算子と言う場合もあります。				

8.2.2　文字列の比較と検索に使う演算子が提案された

コンピュータを利用するとき、フォルダの中のファイル名を検索する方法に、トランプゲームの、通称で言う**ワイルドカード**(wildcard character)を使う方法が一つの常識となりました。部分的な文字並びが一致するファイル名を検索するとき、例えば"*.txt"を入力すると、拡張子".txt"の付くテキストファイルをリストしてくれる機能です。これを**曖昧検索**(fuzzy searching)と言います。これに使う特別な比較演算子が定義されるようになりました。Visual Basic では、表 8.6 に示す演算子です。

表 8.6　特殊な比較演算子(Visual Basic)

	演算子	意味と使い方
7	Is	二つの文字列が同じであるか否かを比較する。数値の比較「＝記号」に対応する
8	Like	文字列のパターンマッチングに使う

8.3　普通の文章で使う言い回し

8.3.1　代名詞を使わない文書はくどくなる

公式的な文章の言葉の表現形式、つまり、言い回しは、伝えたい事柄を正しく理解させるため、定型的になります。しかし、文学作品では、同じ言葉を繰り返して使うことを避けます。言葉で表すことを意図的に省いて、読者に想像させるような作文技法も工夫され、それが文学的価値に評価されます。日常の日本語では、主語や目的語を省いても、誤解されない場合もあります。これは、文単位で見ると、文法的には不完全な構文です。英語では、文としての形式を整えるため、省略を補う代名詞を使うことが普通に行われます。公式な文書では、くどいようですが、同じ物や事を、代名詞を含め、別の言葉で言い換えることをしません。一つの文の中で、代名詞を使わない説明をしようとすると、英語では関係代名詞で繋ぎます。「…すること、…するもの」の言い方が多くなるのは、これが一つの理由です。文が変わって、同じ物や事を引き継いで言いたいとき、代名詞で引用しないで、もう一度 元の名前を使います。法律文書は英文も邦文も、文単位が長く、素直に読めないのは、このことが一つの理由です。プログラミング言語では、変数は、必ず一意の変数名を宣言してそれを使います。代名詞の定義はありませんが、名前が長くなるとき、別名(エイリアス：alias)を宣言して使うことをします。契約書などでは、当事者名などに「甲、乙…」などと言い換える表現を見ますが、これが alias です。

8.3.2　助動詞の使い方

規格、基準、仕様など、公文書的な色彩のある文書は、文章の末尾の言い回しが非常に大切です。プログラミング言語の解説書や教科書を読むとき、また逆に作成するときも、英語では文の始めの方に出てくる動詞と助動詞が文の意義を決定します。日本語では、動詞が文末にきますので、文末まで丁寧に読むようにしなければなりません。表 8.8 と表 8.9 は、JIS Z8301 を参考に作りました。

表 8.7　定型的な文章末尾の表し方

意義	日本語の言い回し	対応する英語
(1)指示又は要求 (requirement)	…する。…とする。…とおりとする。 …しなければならない。…による。 …しない。…してはならない。用いる。 示す。 （箇条書き）…すること。…のこと。	… shall, … shall not, …is to, … is required to, … has to, … it is necessary, … it is not allowed,
(2)推奨 (recommendation)	…ほうがよい。…するのがよい。 …するとよい。…することが望ましい。	… should, … it is recommended, … should not,
(3)許容 (permission)	…してもよい。…差し支えない。	… may, …is permitted, … need not, … is allowed,
(4)可能性 (possibility)	…することがある	… can, …cannot, … be able to, … it is possible, …be unable to

解説

(1)　should の日本語訳は「…すべきである」と解釈していますが、英語本来の意味ではこのような命令的、義務的、または強制的な意味がありません。should, would, could などは丁寧な言い回しに使うことを考えれば、この言葉の感覚が分かると思います。

(2)　支持または要求を表す日本語では、ぶっきらぼうになるのを和らげるために「…示すものとする。…するものとする。…用いるものとする。」のような表現をよく見受けます。これは、英語の関係代名詞がある文の翻訳調ですので、少し英語の素養がある人は不思議な日本語とは感じなくなっています。短く「…示す。…する。…用いる。」とします。

(3)　二つ以上の選択肢があるとき、それらをすべて示した上で、そのどれかを推すのが推奨です。選択の範囲を明示しないで、「原則として…する。」という言い回しは、例外を認める表現になります。単独で使うと、「どうやってもよい」と解釈できますので、指示または要求になりません。「原則」は「例外」と対に使います。そうすると推奨または許容と同じことになります。

(4)　日本語の感覚では、英語の may と can との意味上の使い分けが難しい面があります。may は、他者に指示をするときの言い回しです。can は、自分の自主的な判断で、するかしないかを決めるときの宣言的な言い回しです。

表 8.8　定型的な論理語句の使い方

日本語の語句	対応する英語	意味と使い方
…及び…、 …と…	and	二つの事項の並列は、“ＡとＢ”、“Ａ及びＢ”の様に用い、三つ以上のときは、“Ａ、Ｂ、Ｃ及びＤ”の様に用います。
など、その他	etc., and others, and so forth	“など”、“その他”を最後に付けるときは、コンマだけで区切ります。“Ａ、Ｂ、Ｃ、Ｄなど”
又は	or	例：“Ａ、Ｂ、Ｃ又はＤ”。排他的な意義で使います。
並びに	(A and B) and (C and D)	左の論理式を文章で表現するとき、“Ａ及びＢ、並びにＣ及びＤ”とします。誤解を生じないように、なるべくなら別の表現を工夫するのがよいでしょう。
若しくは	(A and B) or (C and D)	左の論理式を文章で表現するとき、“Ａ及びＢ、若しくはＣ及びＤ”とします。誤解を生じない様に、なるべくなら別の表現にします。なお、“あるいは”は用いません。
	A and/or B	“Ａ又はＢ、若しくは、ＡとＢの両方”という意味で、英文で見かけますが、日本語では該当する語句がありません。英文でも正式の表現では使わないとされています。
…場合 …とき …時	in case of, if … as … when	限定条件を示すのに“場合”と“とき”とを用いますが。限定条件が二重の場合、大きい条件に“場合”を用い、小さい方の条件に“とき”を用います。 “時”は、時間を限定するときに用います。
…から…まで	from …to …	時、場所などの起点と終点をはっきりさせるのに用います。“ＡからＢまで”と云うとき、ＡもＢも含めます。例：“東京から名古屋まで”

8.4　プログラミングに使われる修辞学

8.4.1　無生物を主語とする文は動詞を取らない

論理的に正しい文章を書くとき、無生物を主語に立てる文は、自動詞も他動詞も使うことはできません。無生物が、あたかも意識を持っているかのように動作をすることはありませんので、自動詞を取りません。また、無生物が何かに働きかける他動詞を取ることも、ありません。無生物の状態を言う文は、be 動詞に形容詞を繋ぎます。しかし、無生物であっても、レトリックとして擬人化する言い方は、普通に使います。自然現象、例えば「陽が昇る」「風が吹く」「雨が降る」などがそうです。この言い方の裏には一種の宗教観があります。日本では、太陽神、風の神、雷神など、複数の専門家集団の神が居て、それぞれが自然現象を司る、とするアニミズム(animism)があります。「雲が雨を降らす」のような他動詞を使う言い方も、不自然ではありません。ユダヤ教、キリスト教、イスラム教などの一神教は、唯一の万能の神が、自然現象を含め、すべてを制御しているとする宗教観ですが、日本的な感覚からは納得できないのです。コンピュータは人工的な装置ですので、コンピュータを動かす主語は人です。間接的に人を介して動かすと、コンピュータを擬人化したことと同義になりますので、他動詞を使った命令文でコンピュータを制御することは、文章論理的には矛盾しません。しかし、コンピュータ本体が無生物であることも意識にありますので、物理的また幾何学的な性質も扱います。その代表的なものがデータ型です。そして、これを扱う代表的な動詞は、英語では be 動詞です。定義文、宣言文、代入文は、コンピュータが自分の判断でデータを発生も取り込みもしなくて、人の側でデータを扱うからです。ただし、be 動詞を表（おもて）だって使うことは少ないのですが、イコール記号が代わりに使われるのが普通です。

8.4.2　数値計算の判定には三分法も利用している

コンピュータの論理処理は、ブール値を基本に使いますので、結論（真偽）が明快に得られます。これを判定の二分法と言うことにします。論理の進め方は、論理数学の演算則を採用しますので、論理式を使った過程に修辞学的な言い換えがあっても結論は、変わりません。そのため、プログラミングの文構造としての分かり易さを図るため、幾つかの論理学的な演算則が応用されます。よく使うものは、「ド・モルガンの法則」と「否定の否定は肯定」を使った論理式の書き換えです。数値計算の結果を利用するとき、或る位取りのところで四捨五入のような丸め処理をします。位取りの最後の桁には±0.5の曖昧さが含まれます。丸めの幅のことを**閾値**（しきいち：threshold)）と言うことがあります。例えば、実数の集合を整数に丸めると、数値は（正負と０）の三通りに分類されます。これは、三分法です。この方法は、理論と言うよりも実践技術ですので、数学の参考書に説明が載ることはありません。会計計算では、丸めの規則が厳格に決められていますので、計算結果の正誤も明快に判定できます。

8.4.3　文章論理には否定と逆の言い方があること

文章論理には、用語として**逆**と**否定**との使い分けがあります。その例として「良薬 口に苦し」で説明します。これは「良い薬（Ｐ）であれば苦い薬（Q)」と言う意義ですので、論理式は「Ｐ⊃Ｑ」の内含(IMP)です。(Ｐ)、(Q)、(Ｐ⊃Ｑ）それぞれは命題単位です。内含の論理式は交換則が成立しません。「苦い薬（Q）であれば良い薬（Ｐ)」と言うのは前の文の**逆**であって、論理式は「Ｑ⊃Ｐ」です。一方、「良い薬（Ｐ）であれば苦くない（！Q)」は、最初の文の**否定**です。「良い・良くない」「苦い・苦くない」の組み合わせでできる内含は８通りあります。これらの相互の関係を表す論理学の用語に**対偶**と**対等**があります。文章論理では、論理の当否を「真、偽」ではなく、「**ほんと**、**うそ**」で仕分けます。「良薬 口に苦し」のように、命題が単独に示されているときは、すべて「ほんと」扱いをして論理を展開します。途中または結果が「うそ」になる論法が詭弁です。

8.4.4　日本の実社会では三分法が良く使われる

文章表現の中での否定の使い方と、論理学での否定の使い方とは同じ意義になりません。論理演算では、否定は反対に変換する意義です。逆とは違います。例えば、「行かないこともない」と言うのは、部分肯定です。「必ず行く」意味ではありません。ここでは「行く」と言う動作が複数あって、集合の概念があるからです。英語の環境では「yes・no」を使う二分法の表現を基本に使いますが、これは集合であっても二種類しか考えない場合です。日本風は三分法も多く使われ、「yes でも no でもない」保留の意思表示も使います。言葉としては、「上中下、左中右、勝ち負けと引き分け、じゃんけんの相子、数値の正負ゼロ」などがあります。欧米の言語習慣からみると、日本風の三分法は曖昧な判断であるとして嫌います。しかし、二分法を「正誤」または「正義」に沿うか沿わないかの判断に応用すると、実社会では深刻な対立になることが起こります。宗教観や信条の違いで「正義」の基準も異なりますので、妥協が成立し難いのです。日本風の三分法であると、争いの場の解決方法に、「引き分け、相子」を使うことができますので、「仲直り」の手段を探すことができます。

8.4.5　仮説を認めることには慎重であること

修辞学(rhetoric)は、言葉で相手を説得するための技術を扱う学問です。代表的な方法が三段論法です。「大前提(major premise)」「小前提(minor premise)「結論」の三つの命題(proposition)から成る推論(inference)規則です。ギリシャ時代のアリストテレスによって演繹法(deduction)に準拠して整えられました。「大前提」には法則的に導き出される一般的な原理を置き、「小前提」には目前の具体的な事実を置き、「結論」を導き出します。よく見られる例は下です。

大前提：すべての人間は死すべきものである。
小前提：ソクラテスは人間である。
結論: ゆえにソクラテスは死すべきものである。

コンピュータを擬人化して説得する技術に使っても、コンピュータが納得する感覚はありませんが、論理の展開に間違いが起きないようにプログラムをした、その処理結果は信用が置けるとして、人に対して説得する資料が得られます。一般の人は、「コンピュータで計算した」と言うだけで頭から信用するのですが、実は、英語の premise には仮説(hypothesis)の意味があります。大前提も、また小前提も、仮説に基づく理論や法則を使っている場合が普通ですので、二分法的に正誤の拠り所に使うことには慎重でなければならないのです。一旦 仮説を正しい認めてしまうと反論ができません。コンピュータを利用する説得も、実は、かなり生臭い駆け引きが隠されています。

9. データベースと文字処理

9.1 図書館の利用

9.1.1 データベースは軍事用語であったこと

この章は、コンピュータに言葉を注意深く扱わせると言う見地からのデータベースの解説です。そもそも、コンピュータのソフトウエアとハードウエアを含めた全体開発の研究は、軍事利用を目的として、国家の手厚い保護と機密保持の下に進められてきた経緯（いきさつ）があります。高速計算を必要とする場面は、例えば、敵の軍用機（現在ならミサイル）を撃墜する高射砲の制御をするとき、軍用機の位置と速度などの測定データを使って、狙いとタイミングを素早く計算して弾丸を発射しなければなりません。科学技術計算以外のコンピュータ利用は、大枠としての事務処理です。オフィスオートメーション(office automation: OA)の言い方があります。"office, file, folder"などの用語は、事務処理の方から転用して使われるようになりました。事務処理の中の重要な課題の一つがデータそのものの管理です。**データベース**(database: DB)は、アメリカ軍の軍事用語に起源のある用語です。軍用機専用の飛行場を air base と言います。軍事作戦は、データを収集して分析に利用します。この基地と言う意義です。ハイフンで繋ぐ data-base を経て、１語の database と使うようになりました。

9.1.2 図書館学は情報科学であること

何かの書物資料を探したいとき、**図書館**(library)を利用することを思い付くでしょうか？　伝統的な管理方式は、**書物**（＝書籍、図書、本、文書）を財産扱いしますので、分類番号を付けて整理・保存をします。目的の書物を探す（**検索**：retrieve）手助けに、**図書カード**(library card)が整備され、種々の**目録**(bibliography)も備えます。この全体の専門家が、図書館**司書**(ライブラリアン：librarian)です。もともと、**情報学**または**情報科学**(information science)は、大枠として文化系の専門に分類されていて、**図書館学**(library science)は、その一分野に位置づけられています。一方、コンピュータに関する学問の方は、computer science に分類され、主に理工系の専門の性格を持たせていますが、情報科学の性格もあります。ところが、日本では、コンピュータのようなカタカナ語を学問名にすることを避けるため、これを**情報工学**と言う大枠に含めたことことから、混乱が始まりました。コンピュータを利用するあらゆることを、**情報処理**と総称するようになりましたので、情報をコンピュータと読み替えるような解釈が通るようになってしまいました。図書館学にコンピュータの利用も必須になってきましたので、**図書館情報学**(library and information science)と言うようになりました。

9.1.3 情報などの用語の意義

情報は、information の訳に当てた和製漢語です。最初に使ったのは、軍医でもあった森鴎外(1862-1922)である、とするのが定説です。これは、「敵情報告」の中の二字を取ったものです。諜報の方が information に近い漢語ですが、スパイ活動に関連した意義で使いました。**資料**とは、その情報元である文書本体です。**一次資料**とも言います。英語の source（**ソース**）が当たります。**データ**(data)は、英語での説明には collection of facts としています。こちらは実体から得られた別形態の**二次資料**です。英語の fact は、**事実**を**記録**したもの、とします。記録は**主観**または**意見**が入っていないとします。データは、測定装置で得られる、眼に見えない電気信号なども含みます。情報は、データとほぼ同義です。ただし、何かの言語を介して文書化し、眼に見える形態に変えたものを指します。情報はデータの集合であるとの解釈もします。これには、英語の名詞表記の習慣からくる区別が関係しています。data は datum を複数形にした名詞ですが、(複数の) 記録の集合を１単位として表したいときは、単数形で利用するようにもなりました。"information"は、不可算の集合名詞です。どちらも、複数の種類を表すときは、物質名詞のように、例えば two pieces of information のように使います。コンピュータが情報を扱うとき、眼に見える文字や数字も、内部では不可視のデジタル化したデータに直します。データの元にする書物や文献など、実体を蓄積保存する施設が図書館(library)です。図書以外の実物保存の施設が**博物館**(museum)などです。**アーカイブ**(archive)は、コンピュータのファイル管理で眼につく用語になりました。こちらは**保存図書館**の意味からきています。**レポジトリ**(repository)の用語も見ます。保管場所と言う意味です。コンピュータ用語の場合には、データベースの集合をさらにデータベースにまとめた、より大きな集合を扱う意義で使うようになりました。英語では**メタデータベース**と言います。ファイル集合をさらにファイル化する概念をメタファイルと言います。考え方は、階層的な分類で、一段上の分類になる集合の呼び名（一般名詞）です。

9.1.4　文書作成の次に保存と利用を考える

貴重な宝物は、権力または財力がないと保存しておくことができません。宝物を金銭に換算して価値を判断することも行われますが、所持して保存するための責任費用と考えることもできます。貧乏な庶民レベルでは、持っていても扱いに困ります。文化財は、庶民にとっては実質的な価値がありませんので、放っておくと失われます。革命思想は、既成の権力や財力に対する一種の嫉妬ですので、歴史的に見ても、中国の文化大革命のように、略奪や破壊の標的に多くの文化財が狙われました。書物も、焚書（ふんしょ）の古事で知られていて、例外ではありません。現在では出版物が溢れていますが、基本的には、製本された書物を財産扱いします。したがって、保存を目的とした保存図書館(アーカイブ)は、閉鎖的に管理された書庫を持ちます。従来、図書館が扱うのは厚手の表紙（ハードカバー）を持って製本された書籍や、文化財的な価値のある文書が主な対象でした。雑誌は、責任のある機関が発行したものを、製本して、始めて管理の対象にしました。しかし、多様な出版物が溢れるようになって、混乱が始まりました。簡易な製本の雑誌やペーパーバッグ本（ソフトカバー）の扱いが、まず問題でした。**公共図書館**は、一般の人の利用も考えますので、従来の古典的な管理に加えて、一般的な資料を扱うことの対応に悩むようになりました。

9.1.5　書物の利用形態は三種類

個人が書物を保存する場合は、所蔵点数としては知れたものです。その実質的な利用の形態を、筆者は三種類に分けています。

- 一度読めば済む一過性のもの（小説・新聞・雑誌など)、
- 何度も利用するもの（辞書・参考書など。愛読書も含めましょう)、そして
- 消耗品的に使うもの（教科書・テキスト・コピーした資料など）

趣味的に保存するものは、利用とは言えません。俗に言えば、見せびらかしか、自己満足が目的です。利用状態では、書き込みなどをして汚すこともします。物を大切に扱うことは美徳です。汚して使うのは消耗品扱いをすることと同じですので、長く保存する対象に考えません。研究者が退職などで資料や蔵書を処分しなければならなくなったとき、私物を図書館などに寄贈する例もあるのですが、受け入れ側が迷惑することも少なくありません。利用を考える場合は、図書館を介して公的に分類・登録した上で私的に借り出す形にしてあれば、この問題の大半は解決されます。共通に利用できる小規模の図書室でもあれば、一過性の利用ならば実質的には便利です。しかし利用者（ユーザ）の利用方法（マナー）で問題が多く発生します。個人が書物を排他的に保存したがる理由の一つが、このマナーにもあります。盗難や紛失もありますが、コピーが便利でなかった頃、かみそりでページが切り取られる被害に悩まされたものです。共通に利用できる図書の整理と保存は、書庫を設けてそこに書架を置きます。利用者が書庫に自由に入れて、借り出すまでもなく、見るだけで済むような管理方式を**開架式**と言います。これに対して、書庫への立ち入りを制限し、司書を介して書物を借り出す方式や、閲覧室を設けて閉鎖的に利用する方式が**閉架式**です。庶民レベルの書物の利用は、書店での立ち読みと、貸し本屋形式の利用です。古書店は、借用の対価を払って入手し、利用が済めばそこに売りに出すことで、利用のループを構成しています。考えてみれば、合理的なシステムです。日本の書店は、開架式の展示が普通です。インターネットを介するカタログ販売は閉架式と言えます。欧米の書店は閉架式に近いため、立ち読みは追い出されますし、買わなければ読めないような包装もあります。

9.1.6　データベースは共同利用が本来の目的であること

データベースの開発と利用は、文字の扱いが主ですので、文科系専門の性格があります。コンピュータが開発されると共に直ぐに始まっていたのですが、理工系専門の数値計算への利用に比べると、地味な扱いを受けていました。データベースは、あれば便利です。データベースの利用を、自分の専門の研究に利用したいとする発想は自然です。**文献調査**(literature survey)は、必然的にデータベースの利用に繋がります。現在(2010)、個人レベルでパソコンを利用して、私的にデータベースを作成するための道具（ツール）も多くなりました。しかし、それを構築することを研究課題として取り上げるときは、目的意識が明確でないと焦点がぼけます。専門ごとに、データベースの中身に対する要望が異なるのは当然です。本来、データベースの構築は、閉鎖的に自分で作成し、自分の研究に利用することが目的ではありません。この種の私的な試みは、殆んど失敗しています。1980 年代からほぼ４半世紀の間、大学での研究は産学協同が罪悪視されていました。研究成果を実社会で試行して評価され、それを研究にフィードバックして改良するような協力活動ができませんでした。開放的に、共同でデータベースを作成し、共同で利用する。その方法の提案がようやくできる時代になりました。

9.2　文書の整理

9.2.1　倉庫を別に設ける習慣がなかったこと

公私に関わらず、我々は、日常的に、何かを探す作業と、保存と再利用を目的とした整理作業に多くの時間を取られます。典型的には家事がそうです。整理を合理的にするためには、保存と廃棄のための空間を別に必要とします。欧米では冬の寒さが厳しいので、家屋が閉鎖的です。冬籠りを考えて、物を溜め込む倉庫の空間を別に持つ習慣があります。日本建築は、開放的です。押入れを個別の部屋に設けますが、それでは足りないとき、収納用の家具を持ちこみますので、その分だけ部屋が狭くなります。夏炉冬扇と言う言葉があります。収納場所が別に無いので、夏冬 別に使う道具が一部屋に同居した状態を言います。公共的な建築物は、いわゆる箱モノとして当事者が欲しがりますが、建築家も含めて、倉庫の空間を最初から準備しておく設計習慣がありません。その空間があったとしても、そこを機能的に活用する、つまり、整理する技法も、経験的に覚えます。大学の研究者は個室を欲しがるのですが、公的な空間を私的に使っています。公的な整理法と私的な整理法との使い分けが必要です。図書館は、書物を公的に保存整理する機関です。企業内では、個人が私的に書物を整理保存するのも大切です。しかし、原則として、公的な場所に私物を持ちこむべきではありません。企業では、それを厳しく管理するため、制服を義務化し、ロッカールームを別に用意することなどが見られます。

9.2.2　整理法を教える組織的教育がなかったこと

一般に、整理が良いと言うのは、行儀作法的な素養が関係します。合理的な方法を教育システム化することは殆んどありませんでした。超整理法の言葉は、野口悠紀雄(1940-)の造語に始まりました。内容は、コンピュータを利用する整理法について、個人的な経験をまとめたものです。トヨタ自動車の工場管理でカンバン方式が注目されましたが、一種の整理方法として見ると納得できます。整理には上手・下手がありますので、一種の**技術**です。技術の中身は、三つに分けて考えることができます。これを筆者は、技術の三要素「**道具**・**技法**・**技能**」としています。現代風に言えば、「ハードウエア・ソフトウエア・インタフェース」です。専門家がいても不思議ではありませんし、そうあるべき時代になりました。書類の整理で言えば、例えば、書庫・書架・ファイリングキャビネットが道具、分類法が技法、そして、書類の扱い方が技能に当たるでしょう。公的な環境では、前の二つは規則を決めることができます。専門家としての資格が、例えば図書館司書です。欧米の管理職につく女性の秘書は、職業としての位置が社会的に認められています。この秘書がコンピュータユーザの一大勢力です。マイクロソフト社が Office の名称を付けたソフトウエア製品は、秘書の希望を反映するように発展してきました。企業の秘書の仕事は、かなりの部分が整理です。また、女性の方が整理を面倒がらずにやってくれます。道具の中に、パソコンが加わったので、この取り扱いも秘書の素養として必須になりました。

9.2.3　データベースの管理はメモの整理である

メモ用紙やノートに書きとめたものは情報です。私的な利用ならば、頭の中に記憶できればメモに取る必要がありません。メモの内容を後で取り出す手続きを考えておかないと、メモは生きません。データベースは、判り易く言えば、メモを組織的に集めたものです。ある内容のメモを取り出す方法を、コンピュータを使って便利にする工夫が、**データベース管理システム**(database management system: DBMS)です。狭い意味で言うデータベースは、二次資料的なメモやデータを単に集めたものを指します。広義に解釈するときは、データ更新と検索のソフトを合わせた全体を指します。これを物理的に扱うとき、カードが媒体として良く使われます。メモを集め、自分で整理するのは大変ですので、誰か、つまり専門家に整理してもらうことを考えると、これが一つの職業、例えば秘書になることが分かると思います。企業において、あらゆる雑用を取り仕切る部署は庶務課や総務課などの名称がつきますが、平たく言えば雑用係り、整理係りです。大学の研究者は、秘書を個人的に採用している人もいますが、大部分の人は自分で雑用をこなさなければなりません。そのために費やされる時間と費用とが本務の教育と研究を圧迫します。整理は、専門家を育成して、その人にやってもらうのが合理的です。自分の資料の整理は人任せにできないとして、自分用の二次資料の整理を目指してデータベース化に挑戦しても、使い物になりません。結論から言えば、私的に作成し、私的に利用するだけのものはデータベースではありません。複数の人が、同じまたは個別に異なる要望をもって、二次資料を必要とするときに、データベース化が役に立ちます。

9.2.4　カードを利用する整理方法

もともと、図書館は、書物本体（一次資料）の整理保存と共に、データベースの機能も持つ施設です。コンピュータ化が進む以前は、データベース化と検索の作業を、ライブラリアンが行っていました。本体（一次資料）の所在が分かればよいのです。図書館では、新聞、雑誌、パンフレットなどは、保存に向く設備があるか、書物なみの体裁にしなければ、ある期間の公開展示が済めば廃棄します。言わば、捨てる規則です。保存と廃棄とは、整理を挟んで表裏の関係にあります。レコードやテープなどの新しい形式の資料は、取り扱い規則がなければ、業務に組み込むことができません。役所的に分掌を決めて運営をする場合には、弾力的な対応ができません。こちらは、研究者、個人、民間の機関などが、いわば私的なボランティア活動で支えることでバランスが取られてきました。どのような蔵書があるかを探す手立てに、図書館ごとに図書カード(library card)が準備されてきました。図書カードは、カード型データベースの原形の意義があります。蔵書目録(bibliography)や抄録集(abstract)は、二次資料です。目録類は、固有の性格の有る蔵書が作成の対象です。学術論文などは、雑誌の形式で出版されますが、これを年単位などで製本して、始めて図書館の管理に組み込まれます。こちらは専門性が高いので、どの図書館に何が何号から何号まであるという情報を含めた学術雑誌総合目録が利用されています。ただし、専門ごとの内容（contents）である論文個々の情報は扱いません。

9.2.5　カード型データベースの利点と欠点

ユーザが図書を探したい（検索）ときに利用する、図書館側で用意する図書カードは、一つの原簿から二種類作ります。著者名検索用と、書名検索用です。これらは、二次資料の位置づけです。開架式図書館では、ユーザが直接 書物（一次資料）を眼で確かめて探すことができます。しかし、目的の書物が見つからないとき、最初から無いのか、誰かが借り出しているかは分かりません。小規模の図書室ならば、貸し出し管理用ブックカードを使う方法が実践的です。図書カードの内容を、そのままデータベースのファイル構造としたものがカード型データベースです。現在の考え方は、先に電子化した単純なデータベースに登録しておいて、それを元にカードを作成します。パソコンで利用できる表計算ソフトの歴史は 1982 年に始まりました。EXCEL の発表は 1985 年です。現在(2010 年)の時点での EXCEL は、そのままで電子化したカード型データベースのツールとしても利用できます。私的な利用ならば、EXCEL が便利です。カードに作成して二次資料を管理することをモデルに考えれば、インタフェースを覚えることは難しくありません。ただし、レコード点数として 65K の上限があります。本格的なデータベースとしてのリレーション機能が使えません。個人が私蔵する書物は点数も多くありません。所有者は、どこに何が在るかを知っていますので、改めてカードを作ったり、EXCEL などで管理したりすることは、ほとんどしません。公的な図書館では、所蔵点数が桁違いに多くなりますので、カード型データベース式の管理は問題があります。カード単位は、追加と削除はできますが、記録内容を部分的に更新することは、実際作業では殆んど不可能です。言わば、静的な利用です。

9.2.6　二足のわらじを履くような管理をする

図書館で、実体を持つカードや、それを集めた目録書を作ることを時代遅れと考えると間違えます。眼に見える形（ハードコピー）で残すことが重要です。便利さを追求して、すべてを電子化することは、危険を伴います。簡単に書き換えができることは、前の記録事項が失われることです。電子化資料は、保存と再利用に記録媒体が必要ですが、例えば、昔の磁気テープやフロッピーディスクが読めなくなる深刻な被害も受けますし、一瞬の処理で全滅することもありますので、バックアップなどの安全対策が大きな問題です。とは言っても、カードや目録類の作成は、図書管理では大きな労力が必要です。ここにコンピュータを利用することでの合理化が図られてきました。これには、従来からの保守的な運営管理を引き継ぎながらも、ネットワークの利用など、先端的な取り組みもする、言わば二足のわらじを履くような、不経済とも見える管理方式を取っています。その理由は、図書管理が国内だけの問題ではなく国際的な連携が必要だからです。最初に電子化データを作成しておけば、従来からのカードや目録の作成など、眼に見える資料にすることに、弾力的な対応ができるようになってきました。論文誌の個別の項目（contents）は、専門と関わりますので、こちらの検索サービスまでは手が回りません。これには、私的にデータベースを扱う個人や機関が協力する形でバランスが取られています。

9.3　文書の分類法

9.3.1　カードを使う私的な技法

整理の実践的な技術のうち、カードを補助的な媒体(道具)として使う技法が種々工夫されています。初期のコンピュータは、データの作成にパンチカードを使っていました。それなりの便利さもありましたが、大量の紙資源と広い保存場所も必要ですので、これがディスクの利用などに代わり、現在では使われません。年賀状は、住所と氏名の情報を記録したカードです。名刺は企業の顔を持つ個人情報のカードです。これらは、他人からもらって集まるものであって、積極的に収集しません。寸法が標準化されていますので扱い易くなっています。これらを私的に整理する方法には、種々の工夫があります。最も単純には、整理の手間を省いて、適当な箱に放り込んでおきます。筆者は、これを**乱れ籠**方式と言っています。**ゴミ箱**とは違います。整理が必要になったときに、乱れ籠の中を調べます。少し整理を考えるときは、複数の乱れ籠を使い分けます。日常的には、ゴミの分別などがそうです。経験的に理解できると思いますが、分別、つまり分類は面倒なものです。メモに代えてカードに書き込んで知的な作業に使うアイディアは古くから試みられていました。川喜田二郎(1920-2009)の名前のついたKJ法、梅棹忠夫(1920-2010)の京大カード利用法は、それなりの評価があります。しかし、カードを作成して使う技法は、情報管理には基本的に不便です。整理というのは、当面の仕事の後始末をして、次の仕事に備える作業ですので、言わば、後向きです。ただしコンピュータを利用する場面で、カードで整理することを意識下においた整理モデルを考えると、理解し易くなります。一般的に言えば、メモを別に作ることは、それを積極的に利用する場面が、いつになるかが決まっていないと、制作意欲が長続きしません。

9.3.2　分類に使うコード体系を理解する

書物を分類するとき、その内容によって分類番号などを決めて、書架に並べます。日本の図書館で利用するのは**日本十進分類法**（NDC: Nippon Decimal Classification）です。これはアメリカで使っていた方法を参考にしたものです。ヨーロッパでは種々の言語がありますので、こちらは**国際十進分類法**（UDC: Universal Decimal Classification）が使われています。日本、アメリカ、ヨーロッパで、それぞれ固有のコード体系です。一般の人は、数字を見ただけでは分類内容が分かりません。書店で見るように、判り易い見出し語（**キーワード**：keyword）を使えばよいのですが、多種類の言語環境で使う分類法では、各言語に精通していないと正しい分類ができません。司書は必ずしも専門内容が分かるとは限りませんので、数字を使う分類コードが汎用的です。一般書籍は、作者側が分類番号を付けてくれません。司書が困るのはこれも一つです。数字を使う分類法は、一つの憲法に相当する取り決めですし、今までの歴史がありますので、簡単に変更を提案することができません。したがって、これと併用する形で、使い易い分類方法を実践的な管理に試みることも行われます。その一つは、読んで分かるキーワードを使うことです。学術論文では、作者の方で分類を助けるキーワードを付けることが普通の習慣になりました。

9.3.3　シソーラスの作成が準備作業として必要

データベースを作成するときは、二次資料に分類を付けて検索の手掛かりにします。これに、図書の分類方法を転用できますが、詳しい分類法には不向きですし、分類が無い場合が起こります。学術論文では、表題に内容の手掛かりになる学術用語が使われていれば、その用語を見出しに使うことができます。デパートやスーパーマーケットでは、商品の種類別に売り場が分類されています。この場面には、番号や記号の方を補助的に使い、名前や用語を見出し語として使います。これがキーワードの材料です。ここから、コンピュータの利用を前提として、言葉の吟味が必要になり、標準として使う名詞用語をキーワードとして決めます。そうすると、集合名詞・普通名詞・固有名詞などの区別、階層的なつながりを意識しなければなりません。これは、いみじくも分類を考えることです。第３章で名詞の話を取り上げました。その中の3.3節で、階層的な構造で使う分類用の一般名詞について説明しました。データベース構築では、**シソーラス**(thesaurus)を先に作ります。これは、分類語彙辞典と訳しています。採録する語彙がキーワードであって、対象としている専門ごとの用語や学術用語に基づきます。データベースのプログラミングの参考書では、キーワードとシソーラスについての解説は殆んど載っていません。シソーラスは、キーワード間に**上位語**(BT: broader term)・**下位語**(NT: narrower term)・**関連語**(RT: relational term)の階層的な関係を定義してあるのが、単なる類語辞典とは異なります。形態素解析に使う目的で多くの用語を単に集めた辞書とは、編集の考え方が異なります。前項の十進分類法の編集は、階層構造別にキーワードを並べ、それを表す数字コードの辞書です。

9.4 プログラミングする前の予備知識

9.4.1 ファイル装置との関連が重要である

データベース開発の初期の時代は、ハードウエアとして、大量のデータを保存して、任意に読み書きできる記録媒体（ファイルシステム）を開発することから研究しなければなりませんでした。磁気テープは、大量の記録ができる媒体ですが、テープを巻き戻して頭から通して読み書きします。データを最後に追加することはできますが、途中への書き込みと書き換えはできません。これを**順編成ファイル**と言います。全体のデータ保存には向きますが、検索の効率が低いので、データベースの利用には向きません。この解決が、ドラム式またはディスク式の記録媒体の利用です。こちらはランダムにレコード位置を選択できますので、**ランダムアクセスファイル**と区別しました。データの検索は、何かの規則の下に或る範囲のデータを取り出す操作と考えることができます。大量のデータをすべてメモリに取りこんでから探す作業ができませんので、外部記憶装置の効率的な利用を計画します。そのためには、データの方も、検索に使うキーワードを含めたファイル構造の工夫が必要です。外部記憶装置を制御するプログラムが必要ですので、この検索プログラム部分を**データベースエンジン**(database engine)と言います。これを利用するときの言語に **SQL**(Structured Query Language ：**構造化問い合わせ言語**)が標準として提案されました。ハードウエアとソフトウエアとが密接に関連しますので、この全体をデータベースシステムとします。現在のパソコンのディスク容量はギガ単位ですが、初期のコンピュータで利用できる外部記憶装置は大きくなかったこともあって、科学技術計算用とは別のコンピュータシステムとして計画されていました。

9.4.2 ソフトウエアの機能を理解すること

マイクロソフトの **ACCESS** は、パソコン上で、本格的な**リレーショナルデータベース**(relational database)の、作成から利用までが一通りできる、**欲張った**ツールです。データを納めたファイルであると同時に、検索に利用するプログラム部分を持たせることができます。ただし、常にできるようには準備されていませんので、目的に合わせたプログラミングが必要です。リレーショナルを考えなければ、EXCEL は、カード型データベースのツールとして、そこそこの役にたちます。データベース用の元データは、ファイルにまとめなければなりません。ファイルの読み書きと検索の効率を上げるため、プログラムと密接な関連を持たせたデータ構造の設計が必要です。ACCESS では、ファイル本体のほかに、データベース管理用プログラムとの接続情報を集めた全体を、拡張子(.mdb)を付けたファイル名で扱います。これは、統合開発環境(IDE)を構築していますので、データだけを集めた単体のファイルではありません。このファイルを開くと、ここから種々の処理の実行プログラム（作成してあれば）を間接的に呼ぶことができます。利用形態を工夫したいとなると、別のプログラミング言語から ACCESS のファイル部分を使うことができます。ただし、最初からは準備されていません。ACCESS では、自前でプログラミングができるように、Visual Basic の簡易版が利用できるようになっています。マイクロソフトの Office ソフトウエアグループには、**Visual Basic 6.0** 本体(2010 年現在のバージョン、VB6 と略記します)を含んでいませんので、この簡易版を **VBA**(Visual Basic for Applications)と言います。

9.4.3 ユーザインタフェースはオブジェクト指向プログラミングで作成する

コンピュータが開発された初期の頃は、ユーザインタフェースの考え方が未成熟でした。ユーザとコンピュータとの接点（インタフェース）は、プログラム文の中の Read 文と Write 文で代表されます。SQL も、このインタフェース用言語です。パソコンは、計算機械の概念を越えて、一種の擬似的な装置(シミュレータ)の使い方が普通になりました。対象物（オブジェクト）の概念を広く使います。データベースを扱う ACCESS や EXCEL などは、他のソフトウエアから利用の対象（オブジェクト）として考えて、そのデータを利用する技術が開発されてきました。これを OLE(Object Linking and Embedding)と言います。データベース本体も、文書レコードだけでなく、映像、音声なども扱うので、これをオブジェクトと総括し、これらを扱うデータベースを、オブジェクト指向データベース(Object-Oriented Database)と言います。ACCESS のユーザインタフェースは、フォーム、ボタン、ラベルなどを使うます。これもオブジェクトというので、混乱します。インタフェース用のオブジェクト用に、基本的な操作用フォームと、VBA(Visual Basic for Applications)が組み込まれていますが、これは Visual Basic 6.0 (VB6)の簡易版です。したがって、逆に、VB6 でプログラミングして ACCESS を OLE として利用する方法があります。これも**オブジェクト指向プログラミング**です。

9.4.4　プログラミング言語の拡張が図られたこと

Visual Basic(VB)は、従来の方法でプログラミングする、いわゆるBasic流言語のソースコードの**標準モジュール**と、オブジェクトを扱うソースコードを**フォームモジュール**として別に追加する構成に変わりました。他のプログラミング言語もオブジェクト指向を取り入れるようになりましたが、モジュールを別にするVBの言語設計の方が理解し易いでしょう。これによって、単純なインタフェースはフォームモジュールの作成だけで済むようになりました。ACCESSやEXCELには、フォームモジュールの簡易版が内部的にプログラミングできるようになっています。これにVBAの機能を追加すれば、かなりの応用ができます。しかし、VBAを使いこなすには、ACCESSの使い方を覚えると同時に、Visual Basicのプログラミングを理解しておかなければなりません。これは、一般ユーザが、自前でデータベースを作成して利用する目的に向かうとき、道草的に見える勉強を強いられることになります。VBAをコンパイルして実行形式で組み込んだ小単位のプログラムが**マクロ**です。他のプログラマが作成したバージョンでは、ソースコードが眼に見えない形で組み込まれることがあって、ハッカーが悪さを仕組むことにも使われる危険をはらみます。

9.5　注意して言葉を選ぶ

9.5.1　構造：″structure″の用語

コンピュータの参考書には、structureの用語を良く見ます。代表的には、structured programmingがあって、**構造化プログラミング**と訳しています。この反義語はspaghetti codeです。木構造は、tree networkの訳です。プログラムのソースコードを文書として見るとき、論理的な流れが一筋に繋がるように構成することを言います。文単位の構成に階層的な関係が分かるように、インデントを使って視覚的区別できるようにします。論理的な構造で説明するときは、具体的にはGoTo文を使わないプログラミング技法とされます。階層の区切りを明示的に表す方法は、HTML文書にみるように、タグ記号を使います。HTML文書はプログラム文の集合の性格があって、その文を読み取って実行するプログラムが閲覧ソフト､例えばインターネットエクスプローラです｡データベースの扱いでは**SQL**として使われていて、Sはstructuredです。なぜ、この修飾語が付いているかについての説明を、筆者は見たことがありません。以下は筆者の解釈です。″structure″は、論理的に見て、一方向に整然とした、システマティックな組み立てでモデル化するときに言うようです。種々の可能な選択があるにしても、最後に一つのゴールに収斂するような一つの道筋に構成するときに使います。網目構造は道筋としての選択が一意に構成できません。リレーショナルデータベースを構築するとき､分類項目を並列にした一単位の表ではなく､複数の表の集合に分け、それらの論理的な繋がりを定義します。ここに構造の概念を含みます。何かの検索要求は、この構造の特性を踏まえた手順を擬似的なプログラミング言語で論理的に組み立てて指定します。EXCELで作成した表は、リレーショナルな構造を持ちませんので、検索要求は論理的な検索**構造の組み立て**を必要としません。選択肢を多く用意することは親切に見えますが、モニタの画面でアイコンが並んでいると、何から作業を始めればよいかで迷います。ユーザが道案内（例えばHELP機能）を必要とする場面は少ないほどよいのです。

9.5.2　データ構成にはシソーラスを踏まえる

データベース用ファイルの物理的構造は特殊です。このデータ入力を１から始めるファイル作成機能はACCESSの基本です。他の方法で作成されたデータからACCESS用に変換して取り込むこと（**インポート**）が、実際の作業では多くなります。特に、EXCELで作成したファイルからの追加処理が便利です。このとき、データベースの扱いに適するようなデータの並べ替えや、細かな修正が必要です。このためには、さらに遡って、データベースとしてのレコード単位の構造を設計しておかなければなりません。EXCELでデータを作成するとき､考え付くあらゆる項目を列の要素とし､データ単位を行とした表(EXCELの用語ではワークシート）を作ります。このままのスタイルで、ACCESS用の表（ACCESSの用語ではテーブル）に転用するのが最も単純な考え方です。表の項目は、ユーザ側で決めますが、このとき、見出しの名前・項目ごとの名前の扱い・集合名詞・普通名詞・固有名詞などの区別、階層的な繋がりを意識しなければなりません。特定の語をキーワードとして決めておき、キーワード間について分類と階層関係も決めておきます。これはシソーラスの考え方です。ファイルを作成しただけの段階では、ファイルの一覧が見られるだけの使い方しかできません。検索機能もありません。

9.5.3　キーワードの絞込み

検索をするときのキーワードは、基本的にはシソーラスに載せた語を使います。キーワードを知らないと、データベースの利用はできません。一般の人は、シソーラスもキーワードも意識することなしにデータベースを使おうとしますので、データベースの機能を引き出せないことが起こります。また、プログラムの設計者の方でも、ユーザ側の専門的な要望が理解できないことがあって、ヒットしない死蔵レコードができることがあります。この原因の一つにデータ入力のときに文字を間違っている場合もあります。この解決には種々の工夫があります。思い付いた語を検索語に使う方法が**自然語**検索です。文字並びの解読処理を組み込みますので、検索効率は下がります。住所や氏名など、固有名詞を検索語にすると、一字違ってもヒットしません。そのため、他の条件を補助とした連想(association)で検索をしなければなりません。このとき、分類として決めておいた、より大きな概念のキーワードを使って探索します。したがって、このキーワードが使えるような全体のデータ構造を設計すると同時に、このキーワードが使えるようなユーザインタフェースを設計しなければなりません。ここに文字処理が必要です。思い付くすべての語を集めてキーワードにするのではなく、言葉を選んで、数を制限し、階層関係を決め、分類からの落ちや例外の救済を図ります。

9.5.4　固有名詞を扱うときの例

筆者の専門は橋梁工学です。橋梁の実態（facts）を知りたいときの検索語に、固有名詞としての橋梁名を使うのですが、氏名と同じように同名が数多くでる場合も、また意図した橋梁がヒットしないことも少なくありません。在ることが分かっていても、文字違いで見つからないこともあります。例えば、或るデータベースを検索したとき「城ヶ島大橋」と入力してもヒットしませんでした。実は「城ケ島大橋」で登録されていました。「ケ」の字が大文字か小文字かの違いです。ワイルドカードで検索できるようにする、例えば「城＊」を検索語に使いたいときは、文字処理を組み込む必要があります。橋の名前が分からなくて、橋梁の所在地から検索をしたいとして、都道府県名を使うとします。しかし、県境にある場合、管理者がどちらの県にあるかによって検索漏れも起こります。このときは、階層的に一段上の地区名（関東・中部・関西など）を使います。上位語で検索すると大量のヒット件数になることがありますので、そこから下位語を選択して絞りこみます。データベースを構成するとき、分類項目に使うキーワードに文字違いが無い様に注意することが必要ですが、これが次項で説明する正規化の目的の一つです。

9.5.5　リレーションを確立するために正規化をする

データベースの構築が、理科系の感覚よりも、文科系の感覚が必要になる理由は、言葉（文字並び）の扱いがあるからです。橋梁について言えば、項目の主な見出しは橋梁名です。これは、人名なみの固有名詞扱いです。しかし、幾つかの橋桁単位を持つことがありますので、家族構成全体を一単位とするような集合名詞の性格があります。戸籍謄本のような管理用の書類があります。橋梁台帳がそれに当たります。専門的には個別の橋桁単位を対象としますので、ここから個人情報なみのデータが必要になります。架け替えなどで構造形式が変わることもあります。したがって、橋桁形式別に見出しを付けると、複数の同じ橋梁名が並びます。人名と同じように同名で別の場所の橋梁もあります。同じ橋梁でも、市町村合併などで住所表記の変更があって、記録と現実が一致しなくなることも起こっています。データベースの個別の項目は、変更される可能性のあるものが含まれます。この対応を便利に、かつ、確実にするように、データベースの項目の構造を設計しなければなりません。これを助けるのが**正規化**です。具体的には、項目を単一の表にまとめるのではなく、複数の表に分割し、重複を避け、相互に関連（リレーション）を付けることです。一か所の表で修正をすれば、すべての修正が済むような方法ができるように、表間のリレーションシップ(relationship)を設定します。これが、**リレーショナルデータベース**の言葉の始まりです。

9.5.6　何をしたいかの問題意識を持つことから始める

データベースは、現在では二つの使い方の区別があります。一つは、図書や文献の所在情報だけでなく、知的興味を満たすため、要約的な情報も扱うものです。インターネットの検索サービスがこの代表です。もう一つはビジネス活動に使うものであって、閉鎖的な環境で使います。商品管理や名簿のデータを扱うものであって、中身の正確さが要求され、頻繁に書き換え要求があることが特徴です。こちらの方を指向したものが、データベースのプログラミングの主要な課題になりました。知的興味を満たすためのデータベースは、データの追加はありますが、削除や書き換え作業を重要に考えなくても済みます。こちらを指向したデータベースの利用が、一般ユーザにとっては身近な課題ですし、これの解説がこの章の主題です。研究・開発の活動は、論文やレポートの作成が一つの区切りです。この活動を支える準備段階に文献調査(literature survey)があります。自分の研究の客観的な位置づけをし、研究主題を絞り込む場合に、関連のありそうな文献を探します。論文やレポートの最初、introduction 部分は、他の論文とどこが同じで、どこが違うかを、参考文献を挙げて説明します。筆者の経験から言うと、データベースに構築したかった参考文献は、1960 年代、委員会活動の一環として、吊橋の耐風安定性の文献抄録(二次資料)を対象としたものでした。この時代はコンピュータの機能が低かったこともあって、公的な活動委員会資料用としての文献抄録集は、簡易印刷で作成し、電子化はできませんでした。橋のデータそのものを保存して利用したい希望は、長い間、気持ちの上では持ち続けていましたが、個人的な研究用に資料のリストを作成するとしても、それをデータベース化する気にはなれませんでした。データベースは、私物として作るのではなく、複数の人が利用できることが目的だからです。データベースにするならば、研究の最初は、プロトタイプのデータベースの作成から始めます。多くの人の協力を得て、これを育てる形で内容の充実を図ることが理想です。一般ユーザが、パソコンで本格的なリレーショナルデータベースが作成できるようになったのは、ACCESS バージョン１の発売(1992)からです。しかし、かなり経験豊かなプログラマでなければ、このツールを使いこなすことはできません。また、プログラマ側では、ユーザが対象としている専門についての知識がありません。したがって、ユーザとプログラマとの協力が理想です。その際、ユーザ側がしなければならないことは、

①：データベース化したいデータを集めること、
②：それをどのように使いたいかのインタフェースの希望をまとめること、
③：データ量を充実させるためのユーザ協力の手続きを決めること、そして、
④：全体の継続的な管理をどこに置くかを決めることです。

9.5.7　挫折と失敗の歴史がある

データベースを作成したいとして、研究者が私的に取り組んだものは、殆んど役に立ちません。それは、利用の目的を意識しないで、単に作成することだけを目的とすることが多いからです。教育に利用することを考えると、かなり具体的な構想を立てることができます。公的な資料の整理は、それを使うことを目的とした担当の組織が行うか、外部に委託して、本務の仕事と切り離すのが効率的です。このとき、作業のマニュアルを作ります。これは、本務の仕事と関連しますので、全くの他人任せはできません。何を目的としたデータベースを作りたいのか、その需要が一般的に多いものは、名簿の管理です。データベースの参考書で扱う例題の大部分は、大なり小なり、これを扱っています。個人レベルでは、年賀状の宛名書き用に応用されたソフトを使うのがそうです。この内容は、氏名と住所などの個人情報です。量も多くないので、EXCEL を利用しても役に立ちます。企業が私的に作成した個人情報が流出し、いわゆる名簿屋の間でこのファイルが売買され、ダイレクトメールや電話勧誘などで迷惑を蒙る例が増えました。しかし、考えて見れば、最も目的に合った利用方法です。商品管理などを目的としたものも、比較的需要が多いので、企業では担当者を当てることができます。自前でプログラミングできなければ、専門のプログラマに頼むこともできます。蔵書管理を例題とした、本格的なデータベースの作成例題は、アメリカの参考書にはありますが、日本ではあまり見かけません。これは、例題に使うような点数の多いデータベースは、書店などで閉鎖的に使う、やや専門的な性格があるからです。個人の環境で作成する意義はありません。いずれも、利用目的がはっきりしているものは成功する確率が高くなります。現在では、新聞社が自社の記事情報の検索に自前のデータベースを持つようになりました。1980 年代、データベースの開発は、訳の分からないまま一時ブームになりました。しかし、利用する具体的な展望に欠けていたため、急速に熱が冷めて、これを育てる継続的な努力をすることに、上層部の理解が及びませんでした。その中で、日本経済新聞社だけが、地道な開発をして実用レベルに昇華させました。他の企業は、気が付いたときには手遅れに近い実力差がついてしまいました。つまり、システム構築の開発と研究は、それほど簡単にはできないのです。

10. データベース言語 SQL の解説

10.1 データベースの概念の変化

10.1.1 データ集合と捉えるようになったこと

用語としてのデータベースは、当初、図書や文献の所在情報をまとめて、管理と検索の目的で使われたのが始まりです（第 9 章参照)。データ集合を検索して取り出し、モニタ画面で見たり、それを印刷したりして視覚的に表すときは、普通、**表**の形にします。具体的には EXCEL を使う場面を考えるのがよいでしょう。その発展として、表の形にして整理できるデータの**集合**（セット：data set）があるとき、その管理にデータベース技術を応用するように進化してきました。こちらの技術は、図書や文献の管理を目的とする使い方と少し違う、**ビジネスモデル**などに応用しますので、第二世代のデータベースと言うことにします。どちらの場合も、データベースを個人が閉鎖的に使うのではなく、複数のユーザが独立に利用できる**環境**(environment)に構成することが特徴です。データベースの内容を、ユーザが閲覧だけに使う場合を第一世代型とすると、ビジネスモデルで使うデータベースは、ユーザがデータを直接扱うことを許し、データ内容が変化していくことを考えます。項目によっては、勝手に変更できないように**管理**(management)を考えることが、データベースの利用技術を複雑にしている原因です。

10.1.2 モデルと言う言葉

上の段落で使ったビジネスモデルのように、何かの現実的な事物を対象として研究をするとき、非現実的な**モデル**（model）に置き換えることをします。物理モデル・力学モデル・数学モデル・論理モデル…などです。**模型**と訳すと、具体的な形を持つ意義があります。データベースの中身は、視覚的には数字や記号も含めた文字並びの集合ですが、一固まりの集合に意味があるデータ単位が**論理モデル**です。集合名詞としてのデータ一単位を、外部の記憶媒体に保存するときの**物理モデル**の形態が**ファイル**です。データベースはファイルの集合です。コンピュータのメモリ領域に取り込んで処理の対象とするデータは、実体が眼に見えませんので、**抽象モデル**または**仮想モデル**です。データ型や配列などは、数学を踏まえた抽象モデルです。プログラミングは、すべての作業を仮想の世界を扱います。或る事物をコンピュータの処理に載せるようにモデル化することは、学問的に言えば**仮説**を立てることです。この仮説が現実によく合うかどうかのテストをする仮想の実験が**シミュレーション**です。コンピュータを利用する利点の一つは、仮説が有効であるかどうかの実験と評価が仮想世界の中でできることです。或るデータベースの構築ツールを、現実問題に採用するか否かの判断は、最初、小規模の**プロトタイプ**（試作）モデルから始めます。したがって、場面に応じてモデルを変える弾力的な発想も重要です。

10.1.3 環境・システム・管理と言う全体概念を使う

データベースとして扱いたいデータ全体は、量が多く、コンピュータの作業用内部メモリ領域に「全体を取り込んで処理することはできない」とする前提を考えます。大量のデータを記録できる外部の記憶装置（ハードウエア）とコンピュータとの間で効率的なデータ受け渡しをする方法（ソフトウエア）が必要です。どこか別の場所にその記憶装置を置いて、そこと結ぶ通信方法（ソフトウエア）と通信装置（ハードウエア）を考えることが、関連する重要な課題です。プログラミングは、これら外部装置(デバイス:device)の機能を考えなければなりません。デバイスが変わる度にユーザのプログラムを書き換えなければならないのでは困りますので、その対策として、コンピュータからは、ソフトウエアとしての**デバイスドライバ**(device driver)を介して装置（物）を制御するプログラムを使います。**データベースエンジン**は、デバイスドライバの機能を持ちます。デバイスドライバは、デバイスのメーカー側が作成し、それをコンピュータの OS(オペレーティングシステム)に繋ぎます。**システム**(system)言う用語が出てきますが、多くの要素が一体となって一つの機能を実現させる構成を言います。機能を実現させる状態が**環境**です。静的な状態と動的な状態とがありますが、俗に生きている（**アクティブ**：active）と言うときは、何かのプログラムが実行状態にあることを指します。**スリープ**が中止状態にあることは感覚的に理解できるでしょう。この環境の中で、ユーザのプログラム（ソフトウエア）からは、OS 側の**仮想のデバイス**（ハードウエア）を使います。仮想に代えて抽象の用語も使います。DOS(Disk Operating System)とは、ディスク装置の違いを吸収する意義で命名されたものです。大量のデータを保存し、ランダムにデータの読み書きをする記憶媒体として、ディスクを使うことが必須です。しかし、中身のデータ書き換えが必要になるとき、データ量が増えると予定場所に入りきれないことがあります。データ量が減るか、そのデータを削除すると、無駄な領域ができます。これらがファイル管理の対象です。

10.1.4　共同利用をするためネットワーク技術が必要である

データベースの元になるデータファイルは、多くの人が共同で利用できる共有の環境に置かれます。これを**サーバー**（親元）としましょう。子に当たる個別の一般ユーザ（**クライアント**：client）は、個人のパソコンを通信回線で親元と繋いでデータベースを利用しますので、**ネットワーク**（網目構造：network）技術が必要です。用語としては、net に代えて、蜘蛛の巣の意味がある**ウエブ**(web)も一般化してきました。ユーザは、何かのプログラミング言語を介して自分のコンピュータのメモリにファイルのデータを取り込んで使いたいとします。ファイル構造が分からなければ、ファイルの利用ができません。しかし、データベースのファイルは、複数の人が利用しますので。読み書きがユーザレベルで簡単にできると、データ内容の保護に問題が起こります。親元のデータベースプログラミングでは、ハードウエアとソフトウエアを管理する全体を**DBMS**(Database Management System)と括ります。これを通信回線で繋ぐ、別系統のハードウエアとソフトウエアの傘下に置きます。これを**NET Framework** と言うようになりました。この全体を見通した上で、個別の問題を解決しなければなりません。この手の総論と各論の議論に、欧米人は凝る傾向があります。**ソフトウエア工学**（software engineering）がそうです。この議論に特別な用語や頭字語が現れます。それも日本語に訳し難いので、カタカナ語が氾濫することになります。例えば**パラダイム**(paradigm)があります。上で説明した全体の見通しを構築する考え方を言います。このとき、前に説明したモデルや環境と言った概念が使われます。プログラミングは技術ですので、気の短いユーザは、手っ取り早く結果を欲しがります。多くの参考書は、それに迎合するように、どうしても細部の(各論的な)プログラミングコードの解説や使い方の説明に偏る傾向があります。

10.1.5　ファイル構造が特殊になること

データベースの材料データを構築するファイルは、DBMS と関連して物理的には複雑な構造です。このことは、単純なデータベースとしても使うことができる EXCEL で経験できます。EXCEL のデータファイルは、拡張子(.xls)が付きます。「パソコンの OS；実行プログラム EXCEL.EXE；*.xls ファイル」の全体が、言わば DBMS です。メインフレームであるコンピュータの能力が低かった時代は、データベース専用に一台のメインフレームを使いました。ファイルを単純なテキストエディタで開いて中身を見ても、意味のあるデータ並びとして利用できません。また、実行単位の EXCEL.exe 本体のバージョンが異なると、ファイル自体も読めなくなる被害を受けます。同じことは ACCESS でも経験します。ユーザレベルからは、元のデータベースのファイル内容を直接読み書きすることを制限する、間接的な方法が考えられています。中身が見えませんので、**カプセル化**(encapsulate)などと言います。データベースの用語では、中身を問い合わせる手続きが**クエリ**(query)です。ユーザ側が心得ておく最小限の知識は、データの論理的な構造です。通常、パソコンのユーザレベルで最も汎用性の高い論理構造は、表の項目をテキスト形式にしたものです。EXCEL は、拡張子*.CSV の付いたファイルが、テキスト形式のファイルとして読み書きする機能があります。EXCEL では、一つの表の行数と列数の最大寸法に制限がありますので、一つの表を一つのテキスト形式のファイルで扱うことができます。データベースは、部分的には表の形で利用するとしても、全体として行数と列数の制限がありませんので、或る検索条件のもとにデータ数を絞り込んでから、テキスト形式に変換して利用する方法が必要です。つまり、データベースの利用では、問い合わせ(query)のソフトウエア技術が必須です。この中身は高度に専門的ですので、一般ユーザの手に余ります。したがって、本格的にデータベースの構築と利用を計画するときは、現状では、マイクロソフト社かオラクル社のソフトウエアシステムを使うことしか選択の自由度がありません。

10.1.6　個人の閉鎖的利用を考えたソフトもあること

データベースは、個人が閉鎖的に利用するものではありません。データファイルの構築・管理・利用は、別々の人が当たることを考えて計画します。しかし、個人的な利用目的でデータベース的な扱いをすることはパソコンの環境では普通です。ファイル名の検索や、ワープロソフトの編集メニューには文字検索機能があるのがそうです。データベースの開発のデザイン段階では、プロトタイプ的なデータベースシステムを、個人的な環境で構築することから始めます。このときのツールとして、原初的にはテキストエディタなどで作成する文字並びの集合データを使います。この発展として、EXCEL(1985 発表)が、また、パソコンが高機能化したことで ACCESS（1992 発表）が利用できるようになりました。パソコンを使う程度の一般的なユーザがデータベース的な利用をしたい希望が多いのは、住所録などを作成して宛名書きに利用することです。比較的使い易いソフトが市販されています。中程度の図書館の図書管理にも、使い勝手のよい市販のデータベースソフトが見つかります。この作業で必要になるのは文字探索に使うキーワードですが、特にシソーラスを意識するまでもありません。

10.2　問い合わせの儀式

10.2.1　SQL の語源

SQL の提案は、1970 年に遡るのですが、当初、シークエル(SEQUEL: Structured English Query Language)と呼ばれていて、「English」とわざわざ断る説明が入っていました。1970 年頃のコンピュータ言語は、コンピュータに指令する命令語（コマンド）の構文は、日本語で言う助詞の「がのにを」に当たる前置詞を省く使い方が普通でした。ただし、COBOL(1959)は、例外的に予約語の中に前置詞も含めた英語風に構成するプログラム文を使います。SEQUEL は、COBOL と同じ設計思想を主張する命名であったと想像します。しかし同名の省略語が既にあったこともあって、SQL に落ち着いたとの説明がされています。ここで、structure の用語が使われていることが、日本人には理解し難いところです。日本語では単純に構造と訳すのですが、英語の同義語は design, composition を含んでいます。つまり、単純に検索語を集めた言語集合ではなく、或る機能を組み立てるように設計(design)された言語、と言う意義を含む言葉遣いです。文法構文(composition)が英語だよ、と言う意義を含めて、当初は English が入っていたのだと想像できます。

10.2.2　言語本体を設計するときは仕様書が要ること

SQL の言語仕様の骨格は私的な標準(de facto standard)に始まり、1987 年頃から ISO/JIS で規格化(de jure standard)が図られてきました。規格は、種々の要求を受けて改訂されていきます。しかし、規格そのものは、時代を先取りするのではなく、かなり保守的な提案をします。規格は憲法のようなものですので、規格の文書本体は、参考にする価値があります。プログラム言語のコンパイラを発売するベンダーは、ユーザの使い勝手が良いように拡張した機能を追加した製品を出すこともします。そうすると、バージョン違いや、他のベンダーの製品で利用するとき、正常に機能しないことが起こります。規格に忠実な範囲でプログラミングすると、幾らか不便ではあっても、ソースコードの互換性が高くなりますので、プログラムを管理する場合には重要な態度になります。規格の元（オリジナル版）は英語版 ISO です。JIS は、それを翻訳した日本語版です。この二つを比べると、前の、第 7.1 節最後の段落で説明したように、人に説明するときに使う用語の定義(1)と、コンピュータが理解する方の予約語の定義(2)との違いを確認することができます。(1)の方の英語用語で、日本語に訳し難くて、カタカナ語で使う語があります（例えばスキーマ)。元の英語の定義を翻訳しただけでは、日本語の環境では説明になりません。また、紛らしいのですが、予約語(reserved word)の意義の語彙の集合を、キーワードと説明することがあります。こちらは、データベースの検索目的に使うキーワードと違う語彙の集合であることに注意します。

10.2.3　SQL を埋め込み言語として使う

SQL の構文は、プログラマが作成するホスト（主）プログラム文の中に組み込んで使うことを目的としています。言語の仕様を元に、プログラムモジュールのライブラリが必要であって、ライブラリを繋ぐという宣言がどこかで行われます。このライブラリを使うとき、つまりサブプログラムを呼ぶ手順のとき、プログラマが現在使っているプログラミング言語の定義には無い、別の予約語を使う書式になります。採用しているプログラミング言語が変わっても、この書式のまま使う方法が考えられました。これを**埋め込み言語**(embedded language)と言います。**スクリプト言語**(scripting language)とも言います。プログラミング言語が変わると、組み込み言語の書式も変えるのでは混乱します。SQL は、この変更無しで使うように提案された言語です。しかし、これを利用する元の言語のコンパイラは、必要なライブラリを組み込むようにしなければなりません。これは、コンパイラのベンダの方で解決してもらいます。ユーザにとって面倒なことは、元のプログラミング言語の、バージョンの改訂を伴うことです。埋め込みは、プログラム文書だけでなく、実際にはプログラムコードを混ぜて使うリンカーレベルで必要です。二つ後の段落で説明しますが、クラスはサブプログラムの性格を持ちます。これをオブジェクトと解釈して「ホストのプログラミングコードに埋め込む」と言う用語 **OLE** (Object Linking and Embedding) を使います。具体的な例で言うと、ホストのプログラミング言語が Visual Basic であるとして、これから EXCEL または ACCESS で作成したファイルをそのまま組み込んで（embedding して）利用できるようにするソフトウエア技法です。実は、この組み込みを逆にして使えるようにしたのが、EXCEL VBA, ACCESS VBA です。VBA は Visual Basic for Applications の頭字語です。Visual Basic 本体の簡易版になっていて、専用に使う予約語があります。ACCESS の場合には、本体は間接的に SQL を使っていますが、VBA の中から明示的に SQL 関係のコマンドも利用できます。

10.2.4　埋め込み言語のスタイルは特別ではないこと

何かの文書の中に別仕様の言葉を混ぜる方法は、普通に見られます。典型的な例は、文芸小説です。状況説明の文と、引用符で括った会話の文が混ざります。会話の部分だけを並べる作品がドラマの脚本です。また、文単位で見ると、日本語では漢字が中国からの外来語ですので、熟語には音読み熟語と訓読み熟語が混ざります。さらにカタカナ語を混ぜると、主体性の無い、妙な文書になることがあります。種類の違う文構造を混ぜる方法がプログラミング言語でも採用されるようになりました。プログラマは、一つの言語だけの利用にこだわらず、種々の言語にも弾力的に対応しなければなりません。英語の環境では、言語違いは方言違い程度の認識です。コンピュータを擬人化し、その人に読んで理解してもらうには、方言違いの用語の定義と宣言の儀式が必要です。コンピュータ側が、その混合文を解釈して実行する方法が二通りあります。一つは、コンパイラを介して実行文に直す方法、二つ目がインタプリタ方式です。後者の方法は、即時性があります。代表的にはインタネットの閲覧ソフト(**ブラウザ**：browser)が解読する HTML 文書がそうです。これも、プログラム文と解釈できます。HTML 文書に種々の機能が追加できるように文書の仕様が拡充(extended)されてきました。スクリプト言語に Java Script, VB Script が使えるように進化してきました。それを解読する閲覧ソフトの機能が対応していないと読めません。したがって、迷惑なことに、ユーザの方で使わない機能が追加されても、頻繁にバージョンの改訂が提案されます。かなり大幅な改訂が加えられたものが、XHTML 版です。X は extended の意義です。しかし、改訂で、使えなくなる機能があると悲劇が起きます。

10.2.5　クラスと言うプログラム単位が考えられたこと

何かのプログラミング言語を使って少し込み入った処理をしたいときは、一般的にはサブプログラム(sub-program)の集合を作ります。小単位のステートメントやサブプログラムの集合を**モジュール**(module)と言い、モジュールの集合が一単位のプログラムです。プログラミングの作業性を効率化するには、全体を一つのモジュールに作成するのではなく、適度な独立性のある単位に分けます。この考え方はソフトウエア工学(software engineering)の一つです。データベースは外部のディスク装置を利用しますので、このライブラリはオブジェクト指向で作成され、中身はカプセル化されます。このオブジェクト指向のモジュール単位を**クラス**(class)と言うようになりました。この中身は、装置の使い方に関係のあるデータ定義（属性、**プロパティ**：property）と、装置を制御する命令(**メソッド**：method)とを含めますので、ソフトウエアではあっても、物(object)扱いをするようになりました。OS が Windows になったことで、本体装置を制御する別の擬似的な装置を GUI の画面に表示し、それを扱うプログラミングが必要になりました。この擬似的な装置を通称（普通名詞）でコントロールと言い、個別には、より具体的な普通名詞（フォーム、ボタン、ラベル、テキストボックスなど）を使います。ユーザレベルでは、それぞれに固有名詞を当てて宣言して使います。Windows のシステムでも使うコントロールの定義は、コンピュータ側（システム）が標準として持っています。しかし、特殊なコントロールは、サードパーティがクラスライブラリとして提供しますので、そのライブラリを参照する宣言が必要になります。この作業全体を統合開発環境 **IDE** が管理します。

10.2.6　定義と宣言との区別を使い分ける

コンピュータを使うときは、利用する単語の、**定義**(definition)と**宣言**(declaration)とを意識しておかなければなりません。言語の大本の定義は、規格や仕様書で決めます。その言語を理解するソフトウエアが、コンパイラやインタプリタです。このソフトウエアを実行状態にするには、定義済みのコマンド（プログラム名）を入力することですが、これが、プログラムの実行をコンピュータ向けに宣言しています。実行状態になると環境が変わり、コンピュータ側は、そこで利用できる定義済みの別の予約語を理解するように変わります。ユーザが変数などに付ける名前は固有名詞の性格を持つ一種のニックネームですが、これをコンピュータ側に知らせる宣言文を、プログラムコードの最初に置きます。コンピュータに作業を指令するときの文の基本形式は、予約語として決まっている動詞が先行した命令文です。SQL では、これを、二つのカテゴリに分けています。**データ定義言語**(またはスキーマ定義言語、DDL: data definition language)と、**データ操作言語**(DML: data manipulation language)です。まず DDL について言えば、これは、データベースを最初に構築するデザイン段階で必要な処理です。CREATE 文がそうです。言葉を厳密に使う場合には、定義済みのデータ型を利用して、データ変数の名前を宣言することです。一般のユーザは、データ定義と宣言とが既に決まっているデータベースを利用しますので、特に意識する必要がありません。プログラマが具体的にデータ内容を問い合わせるプログラミングをするときには、このデータ定義と宣言を理解しておくことが必要です。

10.2.7　一般ユーザにはデータベースを読み取り専用で使わせる

データ操作言語 DML の方は、データベースのファイルデータを直接読み書きする処理用の構文です。動詞の意義を持つ予約語で主要なものは、次の４つ；INSERT, UPDATE, DELETE, SELECT です。最初の３つは、ファイル内容の書き換えを伴いますので、ビジネスモデルのデータベースで必要です。しかし、図書文献のデータベースの場合には、一般ユーザには閲覧だけの使い方をしか許さないように制限し、データの追加や変更は、権限を持った責任者だけが当たるようにします。つまり、SELECT で始まる文だけが問い合わせ処理の骨格です。SELECT は他動詞ですので、目的語（を）の他に、条件を補う補語的な句が必要です。規格の英語は、文法用語の predicate であって日本語では**術語**と当てています。英語の構文では、主に前置詞を使う句と、大小比較、論理演算に関係する形容詞的な関係を表す記号も含めています。

10.2.8　目的語は検索対象のキーワードです

データベースを利用するユーザは、検索語（キーワード）の入力が要請されます。登録されているキーワードと一致する正確な綴りを使うことが要請されるのですが、実際にはかなり厳しい条件になりますので、部分的に一致する文字並びを使う方法が工夫されています。トランプゲームの**ワイルドカード**(wildcard character)、通称で言うババを使う方法がそうです。規格の用語では LIKE 演算子と言います。複数のキーワードの組み合わせるときは、(AND,OR,NOT)を組み合わせた論理式の形を使います。補語は、検索範囲を指定する句または文の集合です。ここには、前置詞が使われます。外国人にとって日本語の助詞である「がのにを」の正しい使い方が難しいのと同じで、日本人にとって英語の前置詞の使い方に悩まされます。動詞と組み合わせて意味を限定する使い方を句動詞(phrasal verb)と言います。動詞が省略されて前置詞だけで使う代表的なものに"for"があります。「for next 文」は繰り返し制御に普通に使うキーワードです。これは、動詞の「do for …」または「do loop for …」などが元になった使い方です。前置詞"for"に範囲の意義がありますので、英米人には他動詞 do を省くことに違和感を持たないようです。

10.2.9　親元のプログラミングが面倒であること

一般ユーザがネットワークを介してデータベースを利用する場面を考えると、少なくとも二つの情報を扱います。質問項目（クエリ）と回答項目（レポート）です。これをホスト側のプログラミングで扱うときは、該当する回答が複数になれば、ユーザ側には表の形で見せるのが便利です、一般ユーザは、プログラミングの実際には深入りする必要がありません。プログラマ側ではデータ構造を扱うときの用語を理解しておきます。データの中身がすべて同じデータ型であるときは、二次元配列を考えることができます。しかし、EXCEL や ACCESS の表で見るように、列方向は同じ型のデータが並びますが、行方向に並べる要素のデータ型は、同じにならないのが普通です。また、一つの表ではなく、複数の表に分けて作成しておいて、相互に関連(relational)を付ける方法も必要になります。ユーザが見る表の形式は、元のデータベースを設計したときの表形式とは別のデザインにしたいことがあります。ユーザ側は、回答項目を自分のパソコン側に写して中身を利用する方法を考えます。多くの場合、ファイルにして保存するか、そのまま印刷して取り出します。親元（サーバ側）が準備するプログラムは、一般ユーザに比べれば遥かに専門に詳しいプログラマが計画し、ユーザの要求を勘案して、ファイル装置へのアクセスと、データ入出力制御、そしてレポートの発行とを含めます。パソコンの性能が格段に進歩してきましたので、一般ユーザが自前でもデータベースを設計することもできるようになりました。例えば ACCESS がそうです。そうなると、多くの勉強も必要になりました。一般ユーザは、プログラミングに専念するには限界がありますので、プログラマに協力してもらう必要があります。そのときの教養の一つが、専門用語の共通理解です。データベースに関連した用語には、ISO/JIS の SQL 規格で定義のあるものがありますが、他の規格で使われている用語も断りなしに現れ、また、マイクロソフト社が自社のマニュアルに使う用語などがあります。これらの用語のうち、幾つかを次節で解説します。

10.3　幾つかの用語の意味

10.3.1　シソーラス関連

データベースの参考書で、シソーラス(thesaurus)を解説したものをほとんど見ません。考え方は、検索に使うキーワードを系統的に集めた辞書です。特に検索用の単語を意識しないで、普通に思いつく単語（これを自然言語と言います）を使う検索機能が強力になりました。しかし、或る決められた個数の検索語だけを使うと効率がよいのは当然です。この単純な選択方法が、プルダウンメニューの利用に見られます。選択可能なキーワードの一覧から選択します。検索に使う自然言語に揺れがあると、検索効率が下がります。例えば、「計算機、メインフレーム、コンピュータ、コンピューター、パーソナルコンピュータ、パソコン、マイクロコンピュータ、マイコン、計算器、卓上電子計算器、電卓、…」など、思いつく用語をすべてキーワードに登録するのではシソーラスになりません。これらの言葉を階層構造に構成します。その用語として、上位語、下位語、関連語の区別を付けて相互参照の語を示します。同義語が複数あるときは、その中の代表となる語を決めて、そちらを使うように提案します。英語では単数形と複数形で見掛け上は別の語ですが（例えば datum と data)、別の意義に使う場合を除き、単数形を使うことが暗黙の提案になっています。二語以上の連続でキーワードにする語は、一語にします。例えば database は、元は２語ですが、ハイフン（－）や点(・)などを介する data.base、data-base と使うと、記号に文字列解読で特別な機能を持たせることもありますので、この使用を避けます。また、大文字・小文字も区別しません。日本語の環境では、英字に半角と全角の区別があります。また、カタカナ・ひらがなの区別を、するか・しないかも大きな問題です。固有名詞の扱いも大きな問題です。橋梁のデータベースを計画するとき、橋名などで検索するときにこの問題に悩まされます。例えば、城ケ島大橋と城ヶ島大橋とは「ケ」の字の大小違いで探索に漏れる例があります。固有名詞を検索語に使うときの問題は、根本的な解決策は無いと言えます。

10.3.2　スキーマ

スキーマ(schema)は、scheme から派生した語ですが、英語環境でもかなり特殊な専門用語です。翻訳し難いので、JIS もカタカナ語のままで使い、解説はありません。スキームには計画の意味があります。スキーマはデータベースを構成する要素をまとめて言う用語です。視覚的に理解するときは表の形にしますので、具体的には表と同義と考えるのがよいでしょう。しかし、データベースの対象を広く考えると、つまりオブジェクト指向データベースであると、いつも表にして表せるとは限りませんので、抽象的に言う概念にスキーマを当てたのだと思います。あまり評判の良い用語ではありません。規格の定義では、スキーマの集合をデータベースとしています。リレーショナルデータベースは、複数の表の集合を使うからです。しかも、SQL の規格ではテーブル(table)の用語も使います。表は行(row)の集合であるとし、行は列要素の集合です。行単位がデータベースの一つの集合単位であって、ACCESS では FIELD です。プログラミングでは配列で扱う単位です。しかし、一般に個別の要素のデータ型が同じではないので、プログラミング言語では扱い難いデータです。COBOL は最初からこの種のデータ集合を扱う設計になっていて、SQL との相性がよい言語です。C 言語では構造体(struct)を使って型の違うデータの集まりを一単位とし、その単位で配列を宣言することができます。古典的に使う BASIC 言語では、構造体の定義がありませんでした。データベースのファイル構造を直接扱うようなプログラミングができません。SQL を埋め込み言語とし、それ介して、間接的にデータを扱う方法が工夫されます。最近(2010)の Visual Basic は、独立した言語製品として販売されなくて、Visual Studio の中に含まれるようになりましたが、構造体が使えるように改訂されています。一方、EXCEL のワークシートは、印刷して使う表の原稿用紙をモニタに表示しています。データベースを作成する前のデータを整理するときに使い易いツールです。ただし、複数の表間に、データベース的なリレーション関係は構成できません。

10.3.3　モジュール

モジュールは集合名詞です。プログラム単位または一単位として扱うデータ集合の一般的な呼び方（普通名詞扱い）です。プログラムもプログラム文の集合で一単位となるからです。モジュール単位は、それを組み込んで、より大きな集合に構成する要素の意義を持ちます。ハードウエアの構成要素もモジュールと言うことがあります。特定の目的と機能とを表す名前を付けて区別します。プログラム単位を言うとき、モジュール構成に何のプログラミング言語を使っているかを区別することがあります。例えば、Visual Basic で、フォームモジュール、標準モジュールのような言い方がそうです。クラスも、モジュールの意義の用語です。使い方（インタフェース）とその機能さえ知れば、中身の詳しい内容を知るまでもないとき、情報隠蔽（information hiding）になります。クラスの中で処理対象とするデータには、外部から値を取りこんで使うものがあります。クラスは、オブジェクトの仕様やデータ型や定義をもっていますが、実体はありません。したがって、クラスの中で扱うデータも、定義の段階では実寸法を持っていませんので、抽象データと言うことがあります。実体を作成するときは、プログラマが名前を付けて宣言する儀式が必要です。そのオブジェクトの実体を**インスタンス**(instance)と言います。これも日本語に訳し難い単語です。英語の「for instance」を「例えば」と訳しますが、具体的に例を挙げる、その実体の方を意味する用語です。データも、宣言を経て実体を作成します。そうすると、利用しなくなったとき、そこで使われていたメモリ領域を開放して、その領域を別目的に使うようにすれば、メモリを効率的に使うことができます。削除するまでもなく、別名で利用する方法も考えられます。このときは、クラスの定義に合った引き継ぎ的な使い方が要請されます。これに**継承**(inheritance)と言う用語が使われます。クラスモジュールの中で、処理コードの部分は、そのまま残して使います。

10.3.4　カーソル

カーソル(cursor)の原義は、測量用望遠鏡の視野に入れて目標点を合わせる目的の十字線や、計算尺の目盛を読む移動線のことを指します。キャラクタディスプレイを使ってタイピング作業をするとき、文字の表示位置を示すための、点滅する縦棒マークをカーソルと言います。GUI の画面では、マウスの移動に合わせて動く矢印マークなどのアイコンもカーソルと言うようになりました。SQL が提案されたのは 1970 年です。二次元的に表す表の、或る特定の行と列の位置、つまり**座標**のことをカーソルと定義しました。**座標値**に当たる方はパラメタと言い、場面に応じて、ID 番号などを使います。GUI の操作環境では紛らわしい用語になりましたので、SQL を使う場面には現れなくなりました。

10.3.5　トランザクション

トランザクション(transaction)は、ビジネス用語です。リエゾンなしに、トランスアクションとも言いますが、米語の辞書は、濁る方の発音で載っています。ビジネスモデルを扱うデータベースの場合、複数のユーザが同じデータ項目を読み書きすることが起きることがあります。この交通整理的な処理を時系列的に行わせるときの用語です。日本語では取引ですが、あまりしっくりした訳語になりませんので、カタカナ語で使っています。図書や文献のデータベースを一般のユーザが利用するときには、トランザクションを考える必要がありません。

11. グラフィックス言語の解説

11.1 設計と製図

11.1.1 設計図に要求される事柄

この章の主題に入る前書きとして、設計と製図、および作図装置の話しを省くことができません。工業製品の製作は、未だ存在していない形状と寸法とを頭の中で想像し、それを**設計図**に表す作業から始めます。設計作業では、図面を作ることが必須ですので、**設計・製図**と繋いで言うこともします。設計の英語は design です。カタカナ用語の**デザイン**と使うと、世間的には、見取り図的な図を描くことと同義に捉えるようです。コンピュータを使う場面では、CAD (Computer Aided Design)と使うようになりました。これも、CAD のソフトウエアと言うときは、作図のソフトウエアと同義に使われています。一方、芸術活動では、フランス語の**デッサン**(dessin)を使います。日本語では下絵、素描などと当てます。工業製品の製図は、製作作業に必要な幾何学的な形状と寸法を、間違いなく伝えることを目的とします。製作時の作業目的に合わせて、必要十分の条件を満たさなければなりません。また、作業目的に直接関係しない情報を、意図的に省くか、重複記入も避けます。したがって、設計図面単位は、作業目的に合わせて複数の図をセットにして構成します。一つの図面単位で、何を表示し、何を削除するかは、製作対象ごとに異なります。そのため、言葉として、特に**工業製図**(engineering drawing)と断ります。製図の描き方については規格が制定されていますし、その技術教育が重要です。形状と寸法の情報が不十分であるものは、仮に見栄えの良い**イラスト**(挿絵:illustration)であっても、参考にはなりますが、製作現場で利用する図面としての実務的な意義はありません。漫画的な挿絵を埋め草的に描くなどは、子供向けには大目に見ても、図の作成者の技術レベルや品位が低く見られます。

11.1.2 芸術的な制作では図なしで作業にかかることもする

設計図は、それを元にして実物を製作することが目的です。文書に使う説明図は、通称でイラストと呼ばれ、理解を助けることが目的です。工業製図とは異なり、イラストは、全体の図の幾何学的形状や寸法にこだわりません。用紙上のレイアウトを考えて、縦横の寸法を適当に調整することも行われます。科学技術の測定値や数式などをグラフ化すると、全体の性質を感覚的に理解できます。これに対して、芸術作品を作成すること自体を最終目的とする場合があります。絵画や彫刻は、基本的に個人作業の一品制作です。制作者の個性的で自由な発想で直接に作業したことが評価されます。デッサンで構想を下書きする場合もあります。しかし、そのデッサンは、全く同じ作品を複数作ることが目的ではありません。今では見られませんが、紙芝居屋が使っていた絵は、一品制作されていました。日本の浮世絵は、刷りの技法を使って、複数の職人の作業工程を経て、複数の同じ作品を出版していたことが大きな特徴です。浮世絵は多色摺りですが、瓦版は速報を目的としてモノクロ版で刷られていました。日本の漫画は、個人作業で作品を発表することを超えて、工場のような環境で作品を制作するプロダクションシステムも見られます。手描きで見栄えのよい作図をすることは、或る程度の才能に加えて、技能を磨く修行が必要です。工業製図は、図が下手であっても、表現している内容に落ちが無ければ実用になります。見栄えの良い図であるのに越したことはないので、設計者は製図を専門とする図工に図面を描いてもらう分業が普通に行われていました。きれいな作図ができる道具に、コンピュータを応用する製図機械を使いたい要望が出るようになったのは自然の成り行きであって、この全体がコンピュータグラフィックス(computer graphics)の専門分野を構成するようになりました。

11.1.3 コンピュータを図工に見立てる

コンピュータグラフィックスには、作図装置(デバイス)が必要です。装置をコンピュータで物理的に制御するときに、グラフィックス言語を必要とします。これは、コンピュータを擬人化した図工として、その人に語り掛ける言語構造として工夫されています。その言語またはそれを使ったプログラミングをファイルに記録して保存しておけば、図をいつでも正確に再現できます。修正もできるようにしておけば、設計作業全体の合理化に繋がります。このデータファイルの方をグラフィックスの**メタファイル**(metafile)と言います。図の再現には専用のソフトが必要です。この処理は専門性が高いので、一般の人が眼にすることが少ないのですが、コンピュータグラフィックス技術の本流です。カラーのグラフィックスモニタの性能が向上すると共に、モニタをキャンバス代わりに使う対話型の作画技法も開発されてきました。また、ハードコピーを作ることが目的ではなく、テレビで見せるような図の使い方が普及してきました。世間的にはこちらの方に興味が集まっています。

11.2　図を描く装置

11.2.1　レコーダからプロッタまでの開発の経緯

地震のような**時系列**(time process)の振動現象を記録して、科学的な解析に応用する目的の装置を、広く、**レコーダ**と括ります。測定原理の部分を**センサ**(sensor)と総称します。電気・電子技術の進歩に伴って、センサとレコーダとを別装置とするようになりましたが、古典的な地震計は純機械式の一体化した装置でした。いつ起こるか分からない地震波を長時間待機して連続記録する部分は、エジソンの発明したドラム式蓄音器のような外観をしています。一般的なレコーダは、ロール状の長い巻紙に**ペン**などで時系列の波形を描き出す装置が主流です。記録紙を大量に消費しますので、磁気テープなどを使い、一旦電気電子的に記録しておいて、それを再生してモニタで観察するか、必要な個所を選らんで用紙に描きだす方法を採るようになりました。**アナログ計算機**は、今では見ることがなくなりましたが、その出力用に、X-Y 二方向の座標軸方向のアナログ入力信号で関数グラフを作成する **X-Y レコーダ**が使われました。**デジタル計算機**の呼び方は、アナログ計算機との対比で言ったのですが、こちらの計算結果のグラフィックス出力用に開発されたのが**プロッタ**です。学術研究に応用する場合は、作図領域として A4 版(210×297mm)の用紙寸法でも役に立ちます。しかし実務、それも地図の作画や製図に応用することを目的とすると、A0 版(841×1189mm)の用紙寸法が使える大型のペン描きプロッタが必要です。この開発と研究過程がコンピュータグラフィックスの一つの柱です。

11.2.2　マイコンの利用が新しいグラフィックス分野を開拓した

1970 年代の末から発売された８ビットのプロセッサを使うマイクロコンピュータ（マイコン）は、テレビの画面をモニタ代わりに使う方式でした。当時のテレビ用のブラウン管は解像度が低かったので、文字を表示するとしても、英字で横 40 字×縦 25 行程度でした。このマイコンがテレビゲームの火を付け、さらに、モニタ上だけで観察するコンピュータグラフィックス技術の、研究と開発を刺激しました。こちらは、質の良いハードコピーを作成する目的を持ちませんので、製図を目的とするコンピュータグラフィックスの使い方とは距離がありました。ディスプレイ専用の、当時としては解像度の高い、640×480 ピクセルのモニタが開発されたことが一つの契機となって、テレビゲームのようなホビーだけでなく、実務にも積極的に利用されるようになりました。このモニタ画面のハードコピーには、ドットマトリックスプリンタとの相性がよく、キーボードには専用のキーが予約されていました。現在のパソコンのキーで PrtScr と表示されているのが、その機能を引き継いでいます。ただし、こちらはモニタの画像をクリップボードに転送するようになっています。テレビゲームと実務との接点になった、その古典的な使い方は、フライトシミュレータです。元々は、高価な航空機の操縦と操作を、実機で訓練に使う代わりに研究開発されたのですが、テレビゲームとして発表されて人気を博しました。

11.2.3　濃淡図作成用の作図装置も要望されたこと

工業製図、また、印刷物に使うイラストは、原則として線図(line drawing)で描き、写真や水墨画のような濃淡表現を使うことができません。油絵のような色違いの塗りつぶし技法も使いません。これを補う作図の技法に、枠で決めた領域を斜線で埋めるハッチング(hatching)が使われます。絵画ではペン画がそうです。**プリンタ**は、文字の印刷を目的とする装置です。原理的には、文字は小単位の塗りつぶし画像です。昔の英文または邦文タイプライタに代表されるような、一文字単位の文字図形の活字を順に打ち付けて印刷する簡易なプリンタは、標準的な事務機械でした。この装置の重要な機能の一つは、カーボン紙を挟んで、複数枚の複写が取れることです。ドットインパクトプリンタは、複写機能が使えますので、今でも固有のマーケットを持つ装置です。プロッタとプリンタとは、一枚単位のハードコピーを作る装置です。複写が必要なときは、同じ作業を複数回行います。ゼロックスに代表されるレーザー方式の印刷機とインクジェット式のカラー印刷機が使えるようになりましたので、プリンタとプロッタとで作図原理上の区別がなくなりました。文字は小単位の図形ですので、文字集合全体をコントラストの強い濃淡図形として扱うようになりました。しかし、新聞社や出版社のように、大量の部数を高速で作成するには、版を別に作成して印刷する専門の装置を使います。印刷に使う版にインクを載せ、それを紙に転写します。細い線の集合が滲んで(にじんで)線として区別できなくなることを防ぐように、表面に細かな凹凸がつくような版を作り、印刷インキが液状に広がって全体が塗りつぶされないようにします。写真のような濃淡図は、俗に**網かけ**と言う方法で大小・粗密の細かな点の集合にした図に撮り直して、コントラストを擬似的に再現させます。初期のレーザプリンタは、濃淡図の印刷の質が良くなかったのですが、最近のプリンタは画質をソフトウエア的に調整しています。

11.2.4　作図のソフトウエアが二系統あること

プリンタが文字の印刷に、プロッタが線図の作成に特化していた頃は、装置の動作原理の違いに合わせて、書き出し命令が異なっていました。プロッタの開発はプリンタよりも後です。プリンタを使う標準的なコンピュータの機器の構成から見れば、追加のハードウエアとソフトウエアが必要です。現在(2010)ならば、これがオブジェクト指向プログラミングである、と説明するところです。プロッタは種々の会社が開発し、また、ビットイメージを扱うＣＲＴモニタを、プロッタに作図する前の観察（モニタ）用にも、また表示装置（ディスプレイ）としても使うなど、装置の種類が増えました。作図のソフトウエアには、線図用(line drawing)と、濃淡図用(painting)の区別がありました。そうすると、ユーザレベルでは、装置違いで別のソフトウエアを使うため、プログラムの書き換えが必要になる問題が顕在化してきました。この解決を図ることが、標準化した**グラフィックス言語**を提案することと、その受け皿としての**デバイスドライバ**の利用です。しかし、この問題は簡単には解決できませんでした。それは、作図原理として、線図を主に考えるか、ビットマップ(bitmap)で作図領域全体を構成する方法とでは、作図データの準備も、また作図方法も異なるからです。製図では文字も記入する必要があります。これも、小単位のビットマップとして扱う場合と、線図で書かせる方法（図 1.1 参照）とがあります。

11.2.5　オブジェクトの考え方が生まれたこと

文字は小単位の図形です。プリンタにテキストを書き出すことと、線図で図を描くこととは、図形の並べ方や描き順に固有の習慣があります。グラフィックスの見方をすると、決められた用紙領域を図形単位で埋めていくことです。そこで、図形を部品化する考え方が生まれます。通常、円・矩形などは単位図形（オブジェクト）と考えることができます。これを、より細分化すれば、線分の集合と見ることができます。線分は、図形を構成する最小部品であると考え、これもオブジェクト（物）扱いができます。部品を組み合わせて、複雑な図形に構成する考え方が structured graphics の用語として現れ、転じて、オブジェクト指向グラフィックス(object oriented graphics)の概念が使われるようになりました。ここでの object は、図形を構成する過程として、部品も完成図形も指します。作図のプログラミングは、object oriented programming と総括して言います。作図のソフトウエアは、幾つかの部品図形のメニューを揃え、これを選択し、変形や移動の操作を加えるような機能を持ちます。図形制作過程のデータをファイルに記録したものがグラフィックスのメタファイルです。このように計画すれば、図の修正はファイルの中の部品単位で行わせることができます。CAD のソフトウエアは、この方法を採用しています。作図作業は、グラフィックスモニタ上で対話的に進めることができますが、裏でこの作業過程がメタファイルを目的として記録されるようになっていることが、単純なお絵描きソフトと異なります。メタファイルのファイル構造として、テキスト形式とバイナリー形式が選択できることがあります。汎用性があるのはテキストファイルですが、ファイルの寸法が大きくなる欠点があります。完成図形のビットマップデータのファイルが**グラフィックスファイル**です。全体をビットマップ形式*.bmp でファイル化すると、余白部分のデータも必要になるので、ファイル寸法が大きくなる欠点があります。この無駄を省く圧縮形式のファイル構造が研究され、*.jpg、*.gif、*.tif などが利用されています。

11.2.6　幾何モデルの考え方も生まれたこと

手作業で自由な図形を描くと、全く同じ図形を何枚も作ることはできません。個性的な画家のオリジナル作品は、独創的なアイディアと技術に対してと、一品物であることで価値を評価しています。彫刻も同じです。工業的に作成される物は、同じ形状を複数の人手を介して複数作成しますので、形状の情報を幾何学的に決めます。これを**幾何モデル**と解釈するようになりました。工作機械は幾何学的な原理で動きを制御します。加工したい形状が自由な形状に見えても、数学原理を踏まえて形状を定義しておかないと製作ができません。設計段階で成品の形状を図面で表すとき、正確に実物を写すように描くのではなく、幾何モデルを描きます。正確な尺度（縮尺など）を使うと、描き難いこともありますので、多少のデフォルメが許され、寸法や注記を記入して正確な情報を補います。プロッタは機械的な装置です。ペンに代えて工具を動かすコンピュータ制御の工作機械と同質の装置です。コンピュータ制御で図形を描かせる CAD の場合と、コンピュータ制御で材料を加工する CAM(Computer Aided Manufacturing)の場合とでは、同じ形状を対象としますが、別々にプログラミングをすることに無駄があることが問題になりました。この解決の一つの方法が、同じ幾何モデルを定義して、製図にも製作にも使うことであって、これを CAD/CAM と総括して言うようになりました。特に、立体的な形状の場合には、透視図を作成して、実物を製作する前に種々の検討ができることが大きな利点です。模型を作って検討することも行われますが、それに代わるグラフィックスシミュレーション技術が便利になりました。

11.3　グラフィックスの言語設計

11.3.1　デバイスドライバの発想

コンピュータグラフィックスは、典型的な、装置依存のプログラミングを使います。線図を描くことを目的としたプロッタは、ペンの移動を電気機械式に制御します。ペンの動きは、方眼状に設定した座標単位でステップ移動をさせます。ペンを用紙上に接しておけば線を描きます。縦横座標軸方向には真っ直ぐな直線を描くことができますが、斜めの線は、ミクロに見れば階段状の軌跡を描きます。マクロに見てギザギザが気にならないように描くためには、方眼の精度を細かくすることと同時に、ソフトウエア的に直線に近似させるようにペンの移動軌跡を設定します。実用的なプロッタの精度は 0.1 mm 程度です。どの作図装置を使うかによって、同じ図を描かせるプログラミングの命令語は、それぞれ異なった仕様を持ちます。コンピュータのオペレーティングシステム(OS)の環境が DOS 系であったとき、ユーザは、作図装置のメーカー側で用意したグラフィックス用サブルーチンを生（なま）で使いました。カルコンプ社(Calcomp)のプロッタは、線を描かせる命令語が PLOT(X,Y,IP)の書式であって、パラメータ IP でペンの上下位置の状態を指定します。この他に、作業開始と終了、ペン種の選択などを含めた数種類のコードが基本です。一方、CRT の画面に線図を描かせる場合のコードは、テクトロニクス社(Tectronix)では MOVETO(X,Y), LINETO(X,Y)の対を提案しました。こちらの命令コードの方が直観的に分かり易いこともあって一般化し、Windows API でも採用しています。装置が変わると、ユーザ側でサブルーチンを書き換える手間がかかります。この手間を最小にする方法が二つあります。一つはメーカー側に統一した標準化したグラフィックス言語に書き直してもらうことですので、ユーザが簡単には対応できない問題です。もう一つは、ユーザが自前のデバイスドライバ相当の中間言語を作成し、メーカー側の用意したサブルーチンを間接的に呼ぶようにします。筆者が私的に開発したグラフィックス用の命令を表 11.1 に示します。動詞の意義を持つ語がメソッド、形容詞的な性質を宣言するのがプロパティと解釈するのがよいでしょう。

表 11.1　線図を作成することに目的とした基本作図命令のセット（島田）

（1）装置依存の基本コマンド(メソッド)	
DPERAS	引き数はありません。グラフィックス装置を初期化します。プロッタでは新しい用紙を準備し、グラフィックスモニタでは、画面を消去します。
DPWIND xc, yc, h	矩形画面の平面座標系をキャンバス画面に設定します。カメラが二次元の世界座標を視野に捉えるモデルを考えます、レンズ光軸が狙う世界座標が(xc,yc)、キャンバスの横幅に世界座標の横幅が収まるように座標軸を決めます。高さ方向はデフォルトのアスペクト比（標準は 3:4）を持たせます。
DPMOVE x, y	x, y は、平面世界座標系でのペンの位置です。グラフィックスウインドウ上に線を引かずに、描き始めのペン位置を設定します。
DPDRAW x, y	x, y は、平面世界座標系でのペンの位置です。現在のペン位置から、指定された位置まで、グラフィックスウインドウ上に直線を引きます。線の種類は、DPENTX であらかじめ指定します。デフォルトは細い実線です。
（2）装置依存の属性(プロパティ)	
DPENSZ ip	線の太さを選択します。初期値は１（細線）です。システムの仕様によって、線の太さの選択種は異なります。
（3）標準的な応用メソッド（抜粋）	
DPENTX ipen	線の種類（実線・破線）などを指定します。
DPCIRC x, y, r	中心位置(x,y)、半径 radius の円を描きます。線の種類は、その前の設定値が使われます。なお、円の内部の塗り潰しは行いません。
DPMARK x, y, im	指定した位置に記号を描きます。記号の種類を番号で指定します。寸法は、モニタ画面ではピクセル単位の点です。
DPTEXT x, y, ″st″	(x,y)を文字列基線の始端として文字を描きます。線図で描くことを想定していますが、Windows のモニタではシステムの機能を使ってフォントを指定します。
備考	上記の命令セットの単語は、頭に DP を付けた６文字で構成してあります。これは、最初、Fortran でサブルーチンをプログラミングしたことによる制限です。筆者はこの仕様を私的にずっと使っています。

11.3.2　教育利用と実務利用とを別にするのがよいこと

コンピュータグラフィックスの面白さは、自前で作図コードを考えて作図を楽しむことにありますので、プログラミング教育に採用すると人気の高い課題になります。この入門教育のときは、利用できるグラフィックスサブルーチンの中身、つまりアルゴリズムの解説を最小限に抑えて、使い方だけを覚えさせ、グラフィックス作品を評価することから始めます。見てくれのよいグラフィックス作品を作ることにのめり込むと、種々の機能が使える市販のグラフィックスソフトを使いたくなります。この段階に達した時点で、作図のアルゴリズムの幾つかを解説する課程に入ります。この中身は、専門別に固有の問題を多く含みますので、範囲が非常に広くなります。教育課程として、どこに焦点を置くかは、悩ましいところがあります。市販のグラフィックスソフトの開発者側は、多くのユーザの要望に応えるように、応用ソフトの品数を増やす戦略を取ります。しかし、実務、特に科学技術の研究問題にコンピュータグラフィックスを利用する場合には、数ある品数の中の一部だけしか利用しないのが普通ですし、また、作図の目的に合ったソフトが無いことも起こります。したがって、全体のソフト構成を軽くするため、基本的なソフトを選択して、自前で目的に合うグラフィックスのプログラミングをする必要があります。表 11.1 は、そのことを意識し、特に、教育利用を目的とした基本作図命令の最小セットです。

11.3.3　オブジェクトの考え方を吟味する

作図を、部品の集合で構成する考え方は、単位図形を物（オブジェクト）扱いをすることです。モニタ上で図形作成の過程を観察するとき、オブジェクト指向プログラミングを扱います。図形は、形がありますが、手に持てる実体ではありません。立体を表す幾何モデルを図に描くとき、この幾何モデルは仮想(virtual)世界の抽象的な物です。使用材料の性質などは捨象します。多面体の幾何モデルは、頂点・辺・面で構成するとしますが、これをさらに抽象化した**存在概念**が、幾何学で言う点・直線・面です。こちらは、本来、図に描くことができません。幾何学の説明に図を描くのは、理解を助けるための便宜的な方法です。さらに言えば、例えば、平面幾何学の直線は向きを考えません。図形として描く直線は、向きを考えることもしますので、描き順や、直線の左右を区別する情報も必要です。図形としての直線を正確に**定義**したいとき、座標系を考えて、代数的に直線の式、例えば「y=ax+b」を考えるとしましょう。これは、オブジェクトとしての直線のプロパティを、二つのパラメータ(a,b)で**定義**したことになっています。名前を付けて、a を勾配、b を初期値のように言うこともします。具体的にはこれに数値を充てることが**宣言**です。ここで、数と言うのも、実体のない抽象的な概念であって、代数学は、数に代えて記号を使います。眼に見える実体を持たせるときに数字を書きます。数値計算を扱うプログラミングを開発することを課題とすると、数学で言う数を、コンピュータのデータとして正確に扱うことができない問題が持ち上がります。コンピュータ言語では、数を実数型・整数型のような区別をします。これはビット並びで数を表す**定義**です。数を文書で表すときは、眼に見える文字並び、つまり図形、で表しますが、そうすると、この文字並びは図形オブジェクトの性格を持ち、印刷領域のどこに書くか、の扱いが必要になります。ワードプロセッサは、オブジェクト指向でプログラミングしたソフトウエアであると考えることもできます。文字処理用の関数やサブルーチンがあります。これをメソッドと言い換えてもよいわけです。文字を表すデータ型(バイト型)があります。また、文字列(string)を型として扱うプログラミング言語もあります。フォントなどがプロパティに当たります。

11.3.4　図形要素に型の考えを使うアイディア

やや高度な数学を応用したいとき、数の型に複素数型が使えるプログラミング言語があります。複素数は代数的に扱われますが、平面座標系の点の座標、または、原点からその点までを繋ぐ、向きの定義を持った平面ベクトルと解釈することもできます。一つの複素数は、二つの数の集合ですが、演算規則は、一つの変数扱いができます。この考え方を拡張して、点・直線・平面も、固有の型を持った変数として定義するアイディアが生まれます。例えば、二直線の交点は、直線の型を持った二つの変数記号 L1, L2 を宣言しておきます。交点は二直線の共通部分ですので、論理積の演算記号を & と約束しておいて、式の形にした(L1 & L2)の演算で求めます。点の型を持つ変数記号を P と宣言しておいて、結果を P に代入するプログラム文を、「P = L1 & L2」と書けば、交点計算が得られるようにします。この構文構成を筆者は幾何言語と言うことにしました。使い易くするため、全体のプログラミングの言語設計を Basic 風にして、これを Geometry Basic、詰めて、G-Basic としました。幾何学の大きな分類として平面幾何学と立体幾何学があり、次元的に見れば２次元と３次元です。１次元の幾何とは言いませんが、物差しの目盛位置で数の大小を説明するときに使う line geometry の用語があって、英語の初等幾何学の教科書に見られます。つまり、単独の数は、１次元幾何学の点の型としても考えることができます。

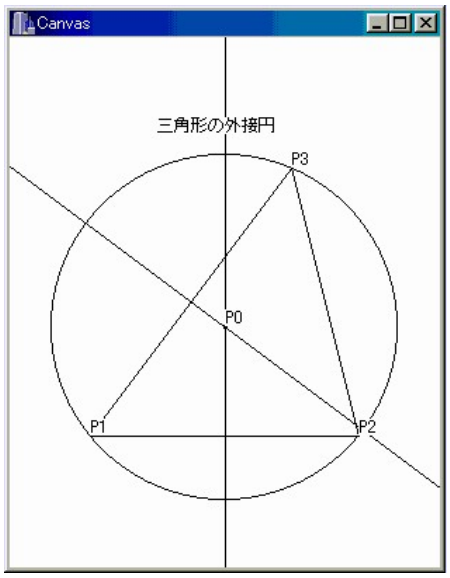
11.3.5　図形要素としての型の提案

幾何学的な性質を考えに入れて図形を描くときは、例えば、或る点から直線に垂線を引くとするとき、垂線と直線とが交わる交点の座標計算が必要です。ここで、幾つかの図形要素に数学的な算法と関連つけるように型の定義を決めることにしました（表 11.2)。考え方としては、前項で説明した複素数の扱いを拡張して、複数の数の集合を一つの変数名として、見掛け上の代数計算式に使うようにしたものです。ここでの代数式に使う演算子は、普通の代数式で使う加減乗除記号の他に、論理積・論理和、さらには二つの図形要素間に直線を引く演算なども定義しました。これらの演算子に行わせる演算は、幾何学的な意味合いを持たせました。二次元、三次元の型の変数は、代入文で左辺に使われたとき、その変数の作図コマンドとなるように G-Basic を設計してありますので、作図用のコマンドは表 11.1 以外には特に増やす必要がないようにしてあります。図 11.1 は、三角形の外接円を求めるプログラムと、その結果の作図を例示したものです。短い文ですので、論理を追いかけることは簡単であると思います。

表 11.2　図形要素を変数扱いをするときの G-Basic の型

番 号	型 の 名 前	寸 法	内　　容	型宣言文
1	整数	1	8 バイト長	DEFINT
2	実数	1	8 バイト長	DEFSNG
3	文字列	2	16 バイトの固定長	DEFSTR
11	二次元の点	2	x,y（点の座標）	DEF2PT
12	二次元の直線	3	a,b,c （直線式　ax+by+c=0）	DEF2LN
13	二次元の円	3	x,y,r （円の中心と半径）	DEF2CR
14	二次元の矩形	4	x0,y0,a,b （2a×2b の寸法）	DEF2BX
15	二次元の線分	4	x1,y1,x2,y2 （始点終点の座標）	DEF2ED
16	二次元の変換行列	6	2×3 のマトリックス	DEF2TR
21	三次元の点	3	x,y,z （点の座標）	DEF3PT
22	三次元の直線	6	x0,y0,z0,u,v,w（点の座標と向き）	DEF3LN
23	三次元の平面	4	a,b,c,d(面方程式 ax+by+cz+d=0)	DEF3PL
24	三次元の球	4	x0,y0,z0,r　（球の中心と半径）	DEF3SP
25	三次元の直方体	6	x0,y0,z0,a,b,c（2a×2b×2c）	DEF3BX
26	三次元の線分	6	x1,y1,z1,x2,y2,z2 （始点と終点）	DEF3ED
27	三次元の変換行列	12	3×4 のマトリックス	DEF3TR

備考：寸法とは、実数が何個で構成されているかの数を云います。実数一個の記憶単位は、標準で倍精度を扱う 8 バイトを指します。



三角形の外接円を求める G-Basic プログラム

```
10 REM       三角形の外接円を求める
20 REM === 三点を通る円を求めること ===
30 CLG : REM 作図領域をクリアする
40 DEF2PT P : DEF2ED E : DEF2CR C : DEF2LN L
50 LET P1,P2,P3=-200,-200,200,-200,100,200
60 E1=P1@P2 : E2=P2@P3 : E3=P3@P1
70 L1=LBSEC(P1,P2) : L2=LBSEC(P1,P3) : P0=L1&L2
80 R0=DIS(P0,P1) : LET C0=P0,R0 : C0=C0
90 PRINT "C0=";C0
100 DPTEXT -100,250,"三角形の外接円"
110 LET X,Y=P0: DPTEXT X,Y,"P0"
120 LET X,Y=P1: DPTEXT X,Y,"P1"
130 LET X,Y=P2: DPTEXT X,Y,"P2"
140 LET X,Y=P3: DPTEXT X,Y,"P3"
```

図 11.1　三角形の辺の二等分線の交点を求め頂点との距離を求めて外接円を描かせる

11.3.6　標準化したグラフィックス言語の提案は難しい

一般論として、装置が変わっても、プログラム本体の変更が最小で済むようにする、標準言語 GKS (Graphical Kernel System)の提案が ISO で試みられ(1978)、JIS 化もされました。しかし、現実には実用化には発展しませんでした。理由は幾つかあります。コンピュータグラフィックスは、利用したい対象や専門に応用するときは、コンピュータグラフィックスに掛ける手間を程々に抑えます。図 11.1 に見るように、ここではグラフィックスに関係するコマンドが、直接目に見えないようにして、ユーザがプログラム文を書く手間を省くようにしました。OS が Windows のような GUI 環境になって、グラフィックスの利用がシステムとしても必要になったことで、システム自体がグラフィックス装置の機能を持つようになりました。この場合でも、フォームを始め、幾つかの図形オブジェクトを表示しますが、その作図命令をユーザが扱わなくても済みます。この機能に対応させる場合でも、筆者は、内部的に自前の中間言語の中身を変えるだけの最小の変更で済ますようにしました。Windows の環境では、差し当たって抽象的なグラフィックス装置を考えます。具体的に外部のグラフィックス装置を使うときには、ファイルにデータを書き出してからそのファイルから別作業として描き出し処理を行わせます。システムが利用しているグラフィックスモニタも、外部オブジェクト扱いです。これを使うときは、画面を準備して、描きだすプログラム単位（モジュール）を作成します。これをクラスモジュールと呼ぶようになりました。グラフィックスモニタの解像度も種々の選択がありますので、ここにデバイスドライバを介します。これは、システムから間接的に呼びますので、ユーザが直接関与しなくても済みます。要するに、標準化を試みると、何段階かの擬似的な装置を介するプログラミングをします。この考え方全体をオブジェクト指向プログラミングと総称します。具体的に汎用のプログラミング言語を使うのはプロ級のプログラマの環境ですが、基本的なグラフィックス機能だけを使う保守的な態度が望ましくなります。

11.3.7　計算幾何学を広く捉える

ここで設計の問題を蒸し返します。形状の設計では、幾何に関わる計算が非常に多いことに注意を向ける必要があります。簡単な幾何学的な問題を数値的に扱うとなると、案外なことに代数計算が非常に面倒になる場面が多くなります。このことを扱う分野を**計算幾何学**(computational geometry)と総称するようになりました。この学問が扱う問題として、ボロノイ図が有名ですが、もっと一般的に、コンピュータを使って幾何に関する数値計算をすること全体と考えるのがよいでしょう。例えば、今ではプログラミング言語の中で、三角関数を普通に使うことができますが、これを精度よく数値計算する技術は、古典的な計算幾何学と考えることができます。アルゴリズムの説明などは教養課程に加えておきたいところです。幾何に関わる代表的な数値計算に、二線分の交点座標を求めることを例としましょう。集合の考え方で見ると、線分は無限に多い点の集合です。交点は、形式的には集合の論理積で得られますので、論理式で表現する言語を提案することができるわけです。数値計算法を吟味すると、計算結果が一意に求まるときは、非常に特殊な場合だけであることが分かります。平行である場合、線分の延長で交点がある場合、接する場合、など、幾つかの起こりうる条件を吟味しなければなりません。したがって、これを扱うサブプログラムは、三角関数のように一意の解が得られるような関数の形を取ることができません。このことが、実は計算幾何学の面白さでもあるわけです。なお、計算幾何学についての解説は、下記の URL に載せてあります。

http://www.nakanihon.co.jp/gijyutsu/Shimada/Computational%20geometry/index.html

11.3.8　グラフィックス処理向けのメソッド

Windows の画面に表示するウインドウと言う枠図形をプログラミング言語ではフォームと言います。この内部に表示する文字や図形は、ウインドウの枠内に収まるように制御されています。これはグラフィックスでは基本的な処理であって、**シザリング**(scissoring)または**クリップ**(clipping)と言います。処理原理は、矩形領域をはみ出す線を描かないようにすることであって、考え方は、矩形と直線との論理積です。この処理では矩形を構成する線分と直線との交点計算を、基本的なサブルーチンとして使います。この他に、線分の長さを求めることのような簡単な処理が多数必要です。どのような種類があるかは、表 11.3 を参照して下さい。これらは、ソフトウエアとして商品化にするには単純過ぎるのですが、利用頻度は非常に多いものです。グラフィックスのプログラミング言語には、表 11.3 の処理に対応できるような単純なメソッドを準備することはありません。これらは、図形処理のプログラミングをするときには、内部的に応用されていて、眼にすることはありませんが、コンピュータグラフィックスの教育では、どこかで基礎的なアルゴリズムとしての知識を埋めておくことが望まれます。

表 11.3　幾何の計算で準備しておく基本処理（メソッド）の例

配列の基本計算		
1	VMOVE(A,B,N)	配列 A を配列 B にコピーする：B=A, または B=-A
2	VMULT(A,SC,B,N)	ベクトルをスカラー倍する: B=sc×A
3	VADD(A,B,C)	二つのベクトルの和
4	VSUB(A,B,C)	二つのベクトルの差
5	VRNDM(N)	区間(0,1)で乱数を発生させる
平面幾何の計算		
1	VN20(A,B)	二つの点間の距離を求める
2	VN21(A)	二次元ベクトルの長さを計算する
3	VN22(A,B)	二つの二次元ベクトルのスカラー積(内積)
4	VN23(A,B)	二つの二次元ベクトルの外積
5	V2PLL(VA,VB,P,IMOD)	二つの平面直線の交点
6	V2PEE(EDA,EDB,VRP,IMOD)	二つの平面線分の交点
7	V2PEL (ED,EL,VRP,IMOD)	平面線分と直線との交点を求める
8	V2PTRX(TRX,PA,PB)	点を座標変換する
9	V2PTRM(ROT,DIS,DISX)	二次元のマトリックスとベクトルとの積を求める
１０	V2PTRR(RA,RB,P,IMOD)	二円の交点を求める
１１	V2LPP(P,Q,VL,IMOD)	二つの点を通る直線を求める
１２	V2LTRX(TRX,AL,BL,IMOD)	二次元の直線を座標変換する
１３	V2LBPP(PA,PB,VL,IMOD)	二つの点を通る線分の垂直二等分線を求める
１４	V2ELE(VL,EDA,ED,IMOD)	線分の、直線に対して正の部分を求める
１５	V2ELB(VL,BOX,ED,IMOD)	直線と矩形との交差で求まる線分を求める
１６	V2EEB(EDA,BOX,ED,IMOD)	線分と矩形との交差で求まる線分を求める
１７	V2ELR(VL,CIR,ED,IMOD)	直線と円との交差で求まる線分を求める
１８	V2EER(EDA,CIR,ED,IMOD)	線分と円との交差で求まる線分を求める
１９	V2ERR(CIRA,CIRB,ED,IMOD)	二つの円の共通接線となる線分を求める
立体幾何の計算		
1	V3MMM(T1,T2,T3)	二つの３×３行列の積を求める： T3=T1*T2
2	V3MTRS(T1,T2)	３×３行列の転置行列を返す
3	VN30(PA,PB)	三次元空間の二点間の距離を求める
4	VN31(A)	三次元ベクトルの長さを計算する
5	VN32(A,B)	二つの三次元ベクトルのスカラー積(内積)
6	VN33(A,B,C)	３×３の行列の行列式を成分ベクトルから計算
7	VN34(T)	３×３の行列の行列式を計算する
8	VNR31(VA,VN,DL)	三次元のベクトルの長さと単位ベクトルを作る
9	VV03(A,B,C)	二つの三次元ベクトルのベクトル積(外積)
１０	V3PTRX(TRX,PA,PB)	点を座標変換する
１１	V3PTRM(ROT,DIS,DISX)	三次元のマトリックスとベクトルとの積
１２	V3PFL(FAEQ,VL,P,IMOD)	面と直線の交点を求める
１３	V3LPP(P,Q,VL,IMOD)	空間の二つの点を通る直線を求める
１４	V3LFF(FAEQ,FBEQ,VL,IMOD)	面と直線の交点を求める
１５	V3ELS(VL,SPH,ED,IMOD)	直線と球との交差で求まる線分を求める
１６	V3EES(EDA,SPH,ED,IMOD)	線分と球との交差で求まる線分を求める
１７	V3ELB(VL,BOX,ED,IMOD)	直線と直方体との交差で求まる線分を求める
１８	V3EEB(EDA,BOX,ED,IMOD)	線分と直方体との交差で求まる線分を求める
１９	V3EEF(EDA,VF,ED,IMOD)	線分の、面に対して正の部分を求める
２０	V3FTRX(TRX,FAEQ,FBEQ,IMOD)	面の座標変換を行なわせる
２１	V3SSS(SPA,SPB,SPC,IMOD)	二つの球面の交差で求まる円を最大径とする球を求める
２２	V3SFS(VF,SPA,SPB,IMOD)	球と面との交差で求まる円を最大径とする球を求める
２３	V3ROTX(VL,ANG,TRX)	回転軸の直線と回転角度を与えて変換行列を作る
２４	V3ROTA(Z,Y,X,CEN,TRX)	座標軸に平行な回転軸で回転させる変換行列を作る

12. 実用文書作成と話し方

12.1 機械翻訳の見方

12.1.1 相手の言語で発信することの難しさがある

世界には多くの種類の言語があります。異なった言語間で間違いなく情報交換をするためには、互いに相手側の言葉を理解する必要があります。物の名前などは、実物（具体的な物）を前にして相互の言葉の確認ができます。しかし、抽象的で論理的な内容の相互理解は難しいところがあります。一般に外国語を覚える方法の順は、最初、一対一に対応する単語を覚えます。次に単語の並びで文を構成する約束、つまり文法を覚えます。さらに、文の並びで構成される文章の全体で意味を理解します。ここまでは、相手側の言語を理解する段階です。自分側から相手側に伝えたいとき、相手側の言語に直します。自分が伝えたいことを文書にする方法が作文、口頭で言う方法が話し方です。英語では英作文と英会話です。戦前までの日本は、英語を母語とする人(native speaker)が殆どいませんでしたので、学習効果を正しく助言してもらう方法に難しさがありました。戦後、米兵相手の売春婦が使う即物的英語をパンパン英語と言い、品位の無い英会話として軽蔑しました。日常会話での実用的な英会話には、昭和 21 年から 5 年続いた NHK のラジオ講座が評判になりました。講師の平川唯一の個性的な教育法から、カムカム英語と言いました。その後、海外旅行をする人が増えましたので、英語に限らず、各国語用に簡単な会話に役立てる本が書店で見られるようになりました。もう一歩進めて、公的な場で使う品位のある実用的な書き方と話し方を学習する教育環境が必要です。こちらの方は、重要視されていますが、具体的にシステム化した教育の場が無いのが現実です。英文の学術レポートを日本人が作成しても、評価がいま一つ得られない理由に、話し方と作文技法の稚拙さがあります。こちらは、教養のある native speaker または、それなりの経験豊かな専門家に添削してもらう必要があります。そのような人材がいつも身近には居ませんので、ここにコンピュータの助けを借りる**機械翻訳**(mechanical translation)と**人工知能**(AI: artificial intelligence)が要望されるようになりました。

12.1.2 日本語と外国語との比較で相互の特徴が分かる

日本語を外国人に教えることの課題は、戦時中にもありました。戦後は、外国人の留学生を日本に受け入れるようになってから重要になりました。それに伴って、外国人から見て、日本語そのものの構成について多くの発見があることも分かってきました。例えば、「な形容詞・い形容詞」、「なに名詞・する名詞」の切り分けは、国語文法にはありませんでした。作文も話し方も、その原稿は人が創作するのであって、コンピュータにその能力はありません。人の方で、何かのヒントを与え、それに応答するように仕組むと、あたかも知能があるように振る舞うことができます。この先駆的なプログラムは、ワイゼンバウム(Joseph Weizenbaum, 1923-2008)が 1966 年に開発した ELIZA です。詳細は省きますが、当時、擬似的にもコンピュータと対話する機能を持たせたプログラムは、新しい衝撃でした。パソコンが普及したのは 1970 年代の後半からですし、自然言語処理が一般化するのも 1980 年代の半ばからでした。ELIZA は英語同士での擬似的な対話を構成しますので、機械翻訳とは異なります。翻訳は、元の構文を分析して、対応する別の言語の単語と入れ替え、単語並びも変えるのですが、あらかじめ辞書と構文のデータをコンピュータ側が準備しなければなりません。逐語的な処理では意味を必ずしも正確に伝える文になりません。ここに論理的で、かつ知的な解決法が必要になります。

12.1.3 まず正しい日本語の作文と話し方が重要

実用文書と言う言い方があります。履歴書、借金証文、手紙は、典型的な実用文書です。英語では business letter（商業通信文）です。文芸作品のような自己満足的な作文と対比する形式の書き物を指します。案外なことに、実用文書の書き方を系統的に、また、公的に教育する場はありません。言わば、見よう見真似で覚えます。企業に入社した新入社員には、言葉遣いや礼儀作法の教育をする例が多いのですが、これが、実用的な話し方です。これらは、本来、自分の責任で覚える個人的な教養です。科学技術が関係する場では、論文やレポートが実用文書です。この書き方に関しては、英語では technical writing の用語があり、日本以外の大学では一つの教養単位になっています。多くの言語環境から学生が集まることから必要になる科目です。日本では言語環境が一つですので、その必要性の意識が育ちませんでした。口頭で発表する場では、一方的な講演形式と、会議で交わす対話にルールが必要です。こちらも系統的な教育の場がありません。この教育の場を考えるときは、日本語も、これを外国語の一つとして客観的に扱う態度が必要です。ここからコンピュータの出番も考えます。

12.2　話し言葉と書き言葉

12.2.1　自然言語は基本的に話し言葉である

自然言語と言うときは、普通、声に出す話し言葉です。音声は一過性の現象ですので、物理的には何も証拠が残りません。口頭での取り決めや約束であっても、これを尊重することは道義です。社会が複雑になると、口頭での情報交換（コミュニケーション）が必ずしも巧く機能しなくなりますので、文字を媒介とする方法が必要です。狭い社会交流の場では、特に文字に書くまでもないので、世界的に見ると、文字を持たない言語が幾つかあるそうです。対等な間で交わす話し言葉は、無礼にならないような、或る標準的な言葉遣いの約束ができます。一般庶民の使う言葉がそうです。江戸言葉は、基本的に耳で聞いて分かる言い方です。落語や講談は、話芸と言いますが、庶民が正しい話し方を覚える場にもなっています。外国人が日本語を覚える方法は、寄席に通うのが一つの良い選択とされています。音読みで使う漢字の熟語は、漢学を基礎教養とするインテリ階級や為政者側が使うことが多く、明治時代以降は、特に海外の用語を日本語化するときに大量に工夫されました。これらは、耳で聞いても直ぐには分からないので、庶民にとっては評判の良くない言葉です。これを皮肉る表現は、落語に多く見られます。戦後はカタカナ語が増えました。これも庶民レベルでは意味不明な用語が多くなりました。使われる頻度が多く、意味の理解が定着し、国語辞書にも載ると、晴れて日本語として認知されたとするようです。外国語の単語は、日本語ではすべて名詞として取り込まれます。その使い方のとき、「なに名詞・する名詞」の切り分けをしています。このときは、元の外国語の品詞が関係しています。

12.2.2　文字表記が問題を複雑にする

女性は、男性に比べて話し言葉を早く覚えます。女の子はおしゃまが多いのですが、これは、子供を育てる環境で必要になる、言わば本能的な能力であるようです。話し言葉は、それ記録して再現する手段に文字を使います。基本的には音を表す**表音文字**を使います。古い時代、日本でも文字を持っていませんでしたので、中国から**表意文字**の漢字を輸入したことから表記法の混乱が始まりました。最初は万葉仮名のように、中国語の漢字の読みで、日本語（和語）の音に近い文字を当てました。中国語では、漢字の読みは四声の区別がありますが、一字一音です。和語の環境ではこの区別を生かすことができないことと、発声の簡略化もあって、同音異字の語が多くなります。したがって、漢字の意味に左右されない、音だけを表す、画数の少ない仮名が工夫されました。古い時代、日本では、仮名文字は女性が主に使いました。源氏物語、土佐日記は、仮名表記であって、女性向けの作品です。ハングルも、女性が覚え易い表記法の意義がありました。仮名文字表記は、意味の取り違えも起こりますので、意味を補う目的で表意文字の漢字を部分的に当て、これを和語風に読ませる方法を採ります。これが**訓読み**です。例えば、動詞の「みる」に当てる漢字は「見・観・診・視・看」を使い分け、眼で見て確認しながら文字を読みます。もともとが当て字の使い方ですので、読み方の分からない表記も頻繁に起こります。日常的には人名や地名に多く見られます。例えば、不如帰と書いて「ほととぎす」と読むことは一種の知識ですが、これを漢字の教養とするのは少し行き過ぎです。知らなかったことは恥ではありません。

12.2.3　表記と発声とは一対一に対応しない。

話し言葉を文字並びに落とし、それを見て発声することは、二回の変換です。話し言葉は、**抑揚**(intonation)、強弱・高低**アクセント**(accent)、**なまり**(dialect)の違いがありますが、書き言葉に表すことができません。文字表記が発声を変えることもあります。ワードプロセッサ（ワープロ）を使って文書を作成するとき、ローマ字入力で仮名に変換するときは、ほぼ一意に変換されます。仮名並びから目的の漢字に変換するときは、幾つかの操作上の規則があります。例えば、発声では「めーじ・しょーわ」が普通ですが、「めいじ・しょうわ」と入力し、幾つかの同音熟語の辞書表示から「明治・昭和」を選びます。中国語用のワープロでも、四声の区別を入力できませんので、同じように辞書表示から選択する使い方です。ワープロ操作では、人は発声する文字並びを頭の中で発想し、モニタで文字遣いの正誤を確認しています。話したことを、すぐに文字に書くことを**速記**(shorthand)と言い、**速記術**(stenography)の用語があります。文字書きを早くするため、独特の記号表記で書いて、後で正しい文字に書き直します。欧米言語は、表音文字を使っていることもあって、話し言葉と書き言葉に大きな違いがありません。秘書の技能として速記術と英文タイピングは必須です。速記者に代えて、パソコンを使って話し言葉をそのまま文字並びに変換することの音声技術は、未だこれからの課題です。逆向きに、テキストファイルになった文書を読み上げて音声に直す方の技術は、表音文字のローマ字を使う英語では、古くからパソコンの環境で実用されていました。日本語での応用は、発展途上です。

12.2.4　言語の研究は書き言葉を材料とする

人工知能は、コンピュータに 人間並みの知能を実現させようという試みの総称です。この言葉の発祥は 1956 年です。人と人との対話に使う基本的な道具は言葉です。便宜的に、何かの人工言語を使ってコンピュータに人（ここではユーザ）が情報を伝え、コンピュータがそれに対して知的に見える情報を返してくれる、その仕組みを研究し、応用することのシステム全体が人工知能です。この手段は、コンピュータが理解できる記号体系、文字表記、つまり書き言葉、に頼ります。コンピュータは人ではないのですが、間接的には、その人工知能の開発者（デベロッパ）が裏に居て、その人に対して何かの言葉を語りかけ、それをコンピュータの処理を介してユーザに返事をするような仕組みを考えます。そうすると、コンピュータを擬人化することになります。コンピュータ側からユーザに言葉を返す方法を加えると、擬似的に対話が成立して、無生物であるコンピュータが、あたかも知能を持つような振る舞いをさせることができます。AI システムは、ユーザとデベロッパの言語環境と同時に、コンピュータ側に仕込む知識レベルの、専門と選択の幅で制限を受けます。AI を広く捉えれば、コンピュータの利用技術すべてを含めることになりますが、通常は、幾つかの限定的な問題に分けて研究開発されています。その一つが、日本語の自然言語処理です。実践的な応用が**正書法**の提案であって、ユーザに対して論理的に正しい作文になるように添削をすることです。それを踏まえて和文英訳の自動化へと繋げます。元になる日本語の論理構造が不完全のままでは、和文英訳そのものが不完全になるからです。ここで、かなり深刻な問題が発生します。それは、二つの言語間で対応する単語がないときです。和語では、抽象的な概念を表す語彙が少ないのです。漢字には抽象的な語が多くありますので、明治以降、欧米に学ぶとき、漢字の組み合わせで大量の用語を工夫しました。これらの用語は、字形を眼で確認しながら音読みします。コンピュータに用語を理解させるときは、読みを必要としないと考えていましたが、対話に使うとなると、発声のことも考えたコンピュータ向けの正書法を工夫しなければなりません。

12.2.5　複数の言語間の翻訳には中間言語を使う

世界には多くの言語があります。話し言葉の段階で、二言語間の対話に、間を置かないで同時通訳を立てるとき、例えば日本語と英語とでは、日英・英日で別の通訳者が当たります。そうすると、Ｎ種類の言語を使う会議の環境を考えると、通訳者はＮ×（Ｎ－１）人が当たることになります。この不便を避けるため、国際会議では公用語を英語とすることが多く、英語以外の言語環境の人は、自国語と英語との通訳をすればよいので、通訳者の数は２Ｎで済みます。このように使う共通言語を中間言語と言います。英語自体も、イギリス英語とアメリカ英語の区別があり、多様な方言や習慣がありますので、中間言語に使う英語は、必ずしも英語を母語とする人たちの話す英語と同じではない標準化が意識されます。英語であっても、曖昧さが生じないように、また、論理的な整合性を持たせ、レトリック的な表現を避けます。ここまでは話し言葉を考えています。機械翻訳はコンピュータに話しかけ、コンピュータから答えを得る二段階の翻訳をします。ここで、コンピュータが理解する中間言語を英語に設定します。この英語は、文字コードを使った書き言葉です。英語を文学的な研究対象にすると、非常に奥行きが広く、かなり難しい課題もあります。しかし、英語自体はとっつき易い性質があって、論理的な表現に適した面がありますので、翻訳を扱うときの中間言語として適しています。コンピュータを介して多言語間の機械翻訳を計画するときも、二度手間に見えますが、英語を介して二度の翻訳をさせます。したがって、日本語ならば、日英間の翻訳の精度を上げることだけに注意を払います。

12.2.6　賢さと頭の良さとは別概念であること

俗に、子供に対して「賢い」と言うのは最高の褒め言葉です。尤も、「悪賢い・ずる賢い」の種別もあります。良い意味での賢さの一つの理想は、一休さんのような、ユーモアと頓智を備えた対応ができることです。頭の良さは潜在的な能力であって、可能性の指標に使う概念です。これは、本人の責任ではなく、両親から受け継ぐ遺伝形質に関わります。賢さは獲得形質です。頭が良いけれど、賢くないインテリもどきが増えています。コンピュータを擬人化して見るとき、頭の良し悪しに当たる能力は、ハードウエア的な性質を言い、メモリの大小と演算速度の速さ、それに蓄積されたデータの多さで評価します。賢さは、ソフトウエアの良し悪しで評価します。大量のデータ、つまり知識があったとしても、それを活用する知恵が無ければ賢いとは言えません。人を対象とした教育現場では、知識の詰め込み、つまり、どれだけ多くの知識を受動的に覚えているかで評価することが普通です。知識を応用する能動的な能力が知恵です。コンピュータに悪さを仕込む悪賢いハッカーと、それ阻止する側との闘いは、一種の知恵比べです。つまり、賢さを良い方向に応用するのが理想です。したがって、悪い方にも応用する十分の知識があった上で、それを抑える倫理が要望されるのです。

12.3　実用文書のまとめ方

12.3.1　実用文書は相手に読んでもらう目的がある

社会生活では、重要な用件は文書に作成することが決まりです。口頭で用件を伝達すると、「言った・聞いて無い」などのトラブルが起こり易いので、簡単な用件であっても、念のために書き物にして相手に渡します。このときに作られる文書は基本的には一通ですので、自分の記憶のために写し、または控えを作ります。用件を伝えたい相手が複数になると、小部数単位の印刷物に作って配ります。雑誌などに投稿する文書は公文書ではないのです。受理されれば多くの人に読んでもらえ、見栄を満足させる喜びがあります。実用文書は、相手に渡すことを考えて作成する、形式を整えた文書です。この定義に沿う文書の範囲は広いのですが、その原点は手紙です。ちゃんとした手紙が書けるのは素養の一つです。書店の実用書のコーナーには「手紙の書き方」の類いのものが並んでいます。社会活動では、個人に代わって企業の顔で文書をやり取りします。そこでは秘書課とか文書課のような部署で形式を整えた手紙などの文書を扱いますが、英語では business letter です。欧米の秘書は、business letter の作成ができることを必須の教養としています。そして、このための現代的な基本技能が、ワードプロセッサを使いこなすこと、そのためにはパソコンの知識が必要になる、という教育的筋書きに繋がります。より専門的で高等な研究を目指す大学や研究機関では、レポートや論文を作成する機会が増えますが、これらも実用文書に含めます。レポートや論文の作成については、どこかで一通りの実務的な教育をする場が欲しいところです。具体的な実務教育を軽蔑的にプラグマティズム（実用主義）といって、観念的な教養教育よりも一段低く評価してきた背景には、中国から輸入された儒学の影響があるようです。

12.3.2　実用文書の三要素

実用文書を作るには三つの技術要素があります。第一は、正しい**文章**が書けること、第二が**書式**、第三が全体としての**体裁**です。この三つの要素は相互に関連していますが、話の内容を整理するためのキーワードとして選びました。文書を作るときには多くの人手を経ますが、基本的な作業は上の三つそれぞれに別の人が当たります。文章原稿を書く人、それを決められた書式に編集する人、そして、体裁を整えて印刷製本を担当する人です。ワードプロセッサなどが無かった時代、原稿と言えば、400 字詰めの原稿用紙に手書きで文章を書くことが作文でした。この原稿を編集者の所に持ち込めば、残りの作業はそれぞれの専門家が処理してくれました。そのため、普通の人は書式とか体裁については漠然とした知識しかなくても困りませんでした。従来の作文指導は、このような作業環境を前提としていたことを理解しておきます。ワードプロセッサが使えるようになって、小部数の簡単な文書は自分で作成できるようになりました。それは、第二、第三の作業も自分でできるようになったことを意味します。そうなると、いままでは専門家がしていた作業に関わることになりますので、特殊な専門用語を覚え、ワードプロセッサの操作に慣れ、パソコンを使うことに悪戦苦闘するようになりました。

12.3.3　書式と体裁の知識が要ること

実用文書は、伝えたい用件に応じた、一般的な書き方の標準があります。これが書式と体裁です。例えば、何かの届けや申請をするときに役所の窓口に出す書類は、現在ではあらかじめ形式の決まった印刷用紙が用意されていて、必要なところに書き込めばよいようになっています。元々は、個人が手書きで書類を作成するのが原則でした。しかし、人によって形式が異なると書類を受け付ける側の扱いに困りますので、一般の人は代書屋と呼ぶ書類作成の専門家の手を借りて、必要充分な書類形式に整えなければなりませんでした。印刷された申請用の用紙は、書式が手書きの形式を踏襲して作ってあります。代書のできる人を代書人と言います。以前は司法書士を指していて、それなりの専門的な知識に加えて、文字をきれいに書ける素養が必要でした。昔の武家社会では、右筆または佑筆（ゆうひつ）と言う文書の専門家が殿様に仕えましたが、現代風に言えば秘書に相当する官職です。文章を考えるのは個人の発想ですので、他人に任せることはできません。したがって、書き物にすることも、個人の責任です。責任ある長の立場に在る人は、公的な日誌的記録を書き残すことが一つの常識です。これは、本来 強制されるものではありません。現代は多忙になりましたので、この素養が意識されなくなりました。論文発表は幾らか私的な自己顕示欲があって作成します。誰かに書いてもらった原稿を自分が作成したようにすることが増え、これによる道義的な問題が多く発生しています。実用文書と対立的な位置にあるのが詩歌小説の類いです。漢詩・短歌・俳句の作成には一応の作法があります。これも、書式と体裁が関わると考えることができます。そこでは筆と墨とを使い、「書を能くする」ことも素養として必要でした。

12.3.4　自分用に文書を残すこと

相手を意識しないで、自分の記憶のために私的な書き物として残すことも日常的に行なわれています。隠居したら、それまでの日記などを元に、自叙伝(autobiography)や回想録的な書き物を書くことは一つの理想です。本人以外の人が、或る人を顕彰する目的で記念誌的な文書を発行することもあって、多くの人が原稿を分担することがあります。しかし、その人が生存中の場合には公表したくない情報が省かれることも多いので、資料としての価値は高くありません。日記は他人に見せることを意識しないで、自分に不利なことであっても、書き残すいさぎよさを持たなければなりません。現代はせわしくなりましたので、手帳にメモ風の記録を残す程度で済ますことが多くなりました。手帳を持ち歩き、予定などを書き込みます。ところが、税務や汚職の調査などで手帳が押収されて不利になる危険を避けて、意図的に手帳などを廃棄することも行われているようです。アメリカの大統領であったニクソンは、ウオーターゲート事件で失脚しましたが、自分に不利となる証拠を処分しませんでした。これは、上に立つ人の矜持（プライド）を保つ態度として、評価されている面があります。一般企業、大学もそうですが、上に立つ人が修める素養を帝王学と言います。民主主義社会では、選挙や持ち回りで長の職に立つ事例が増えました。それに伴って、帝王学的な素養に欠けると思われる事例も増えてしまいました。小説や随想などの形を取る書き物は、一般人は趣味的な扱いと考えることができます。私的な書き物は、形式にとらわれることのない、個人の自由な感情の移入が許されます。自分の経験を元に作成する文学作品は私小説と言われ、作文では入門的な手法です。日本の作文教育は、どちらかと言うと自由な書き方の方を推奨してきました。

12.3.5　ここでは文章作成についてだけを扱います

実用文書は、個人的な感情や心で思ったことを省きます。文学的な表現も抑え、伝えたい具体的な項目に落ちがないようにします。その項目を、英語では 5W1H で表します。昔の軍隊では「いつ、どこで、だれが、なにを、どうした」と報告する訓練をしていました。これに「なぜ、どうやって」を加えると 5W1H(what, when, where, who, why, how)に対応します。作文教育で、文章作法を書いた多くの書物では起承転結の構成を一種の法則のように主張しますが、実用文書に応用するときには注意が必要です。それは、「転」のところの扱いです。今までの論理の流れから、それるような構成に成り易いからです。「転」を省いた起承結の三段落構成は「まえがき・本論・結論」のように筋の通った構成になります。書いた文章が良い文章であると判断するときに使う基準は、一般論で言えば、誤字・宛て字が無いこと、特殊な用語や言葉遣いがないこと、主語述語の構成が正しく、文章に捩れがないこと、などです。日本語は英語に比べると文法が少し曖昧なところがあります。中間言語に英語を意識し、英語に翻訳し難い構文は原則として論理的な構成から見ると悪文になります。文章を直すことを添削といいます。ワードプロセッサの機能が向上してきましたので、文字並びを物理的に判断して校正することが便利になりました。ただし、同音異義的な仮名漢字変換のミスは、人の眼でしか見つけられません。一方、用語や言い回しを吟味することを推敲と言います。この全体が文体であり、個人の特徴が表れます。実用文書では、個性を前面に出さないようにする客観的な基準を決めることがあります。

12.4　エピローグ

12.4.1　筆者の試み

筆者は長らく大学に職を得ていました。大学は、研究と教育をすることを目的に掲げています。研究の方は、先進的な課題に興味を持つのは当然です。一方、教育の方は、毎年新しい新人を受け入れ、保守的とも言える地味な基礎教育を繰り返さなければなりません。これを怠ると、研究に協力してもらう戦力維持に支障が出ます。レポートの作文や発表の仕方、つまり、書き方と話し方の教育は、最も基本的な教育課題ですので、手前味噌にはなりますが、筆者は、かなり意識してきました。筆者が雑誌「橋梁と都市 Project」に連載した書き物の大部分は、筆者が若い時に作成したテキスト原稿の焼き直しです。教育授業の標準は、100 分をコマ単位として一学期 15 回ですので、筆者の連載は一年間 12 回を区切りにするように章の構成を計画しました。中身は、時代が変っても大きく変りませんが、時代に合わせた改訂も意識してきました。

12.4.2　コンピュータの利用を考える時代になったこと

コンピュータの利用が普通になりました。この影響は、人相手の書き方と話し方にコンピュータを介在させる間接的な方法も必要になり、コンピュータを擬人化した相手にした、書き方と話し方が知識として必要になりました。コンピュータが人相手に情報を伝える手段は、従来のように紙に印刷するだけでなく、モニタ上で閲覧する方式が普及してきました。これを**ハードコピー**(hard copy)と**ソフトコピー**(soft copy)と区別します。モニタの画面は狭く、一過性ですので、多くの情報を見ることには向きませんが、即時性の使い方に適しています。今年(2010)の始め、アップル社が発売して携帯端末 iPad は、静的な利用を主体ととしてきた印刷出版業界に大きな衝撃を与えています。実は、この予測は以前からありました。筆者は、それを念頭において、実用文書の書き方について、模索と実験とを行なってきました。そのことの要旨を以下にまとめ、この「易しくないコンピュータ言語学」の結び（エピローグ）にします。

12.4.3　耳で聞いて分かる文章にしたこと

筆者の文体は「です・ます調」つまり、文を声に出して言っても不自然にならないように注意しています。原稿を眼で見ながら発声するとき、漢字を訓読みするか、音読みするかで迷わないようにするには、送り仮名を適切に使うことにあることも分かってきました。読みに迷う場合が想定されている個所では、ワープロ原稿では括弧で読みを入れることもしています。英語に原点のあるカタカナ語は、できるだけ日本語も添えるようにし、そうでない場合は元のスペルを付記しています。漢字を一字使い、その前後が仮名であれば、基本的に訓読みです。二字の漢字が並び、前後に仮名があれば、普通は音読みの熟語です。三字以上の漢字が並ぶとき、読み方の混乱が起こります。人が読んでいるときには正しく読める場合であっても、テキストファイルをコンピュータが読んで発声させるときに間違いが起こる例が知られるようになりました。日本語は分かち書きをしない習慣です。我々の言葉遣いでは、僅かですが息継ぎに間（ま）を入れます。文が長くなるときは、意識してコンマ（読点）を使います。筆者の文章では、主語を立てる「は」の後には、必ず読点を入れています。二字以上の漢字が並んで一つの熟語になるとき、連音または連声で発声し、そうでないときは、僅かの息継ぎがあります。これをワープロで書き言葉にするとき、筆者は半角の空白を使うことも試みています。具体的な書き方の資料は、インターネット上で「実用文書のまとめ方」に載せました。

12.4.4　段落校正を意識してあること

筆者の原稿は、Ａ４版の用紙に印刷して利用できる体裁を考えて MS-Word で作成しています。一ページは、全角で約 2000 字、これを大体三つの段落(paragraph)で構成しますので、一段落は 600 字前後です。段落の区切りは改行です。このようにしてあると、行末での自動改行が効いて、モニタで閲覧するとき、見易い文書になります。また、次のページに段落単位の文章が続かないようにするため、最初に書いた文の削除や挿入などの操作も加えます。一般的なレポートや論文では段落に見出し(caption)を付けませんが、筆者の元原稿には要旨の意義を持たせて番号なしの見出しを付けました。この段落単位は、モニタの画面に程よく収まる分量になります。段落単位は一つの主張をまとめる単位です。このため、原稿の段階で段落単位の作文をしておいて、全体文章の中で、順番を変えることもします。段落の概念は、欧米の文書作成作法では常識になっているようです。編集記述言語 HTML では、章・節・項の見出し部分をヘッダー(header)と言い、<H>、</H>のローマ字を使って区切ります。そして、パラグラフの区切りに<P>、</P>を使います。この P がパラグラフから来ています。日本語の作文指導で、パラグラフの意義と使い方を丁寧に触れていることは見ません。論文などの構成は、章・節・項単位に番号を付けます。実は、項は複数の段落から構成するのが基本です。筆者の元原稿は項番号を付けませんが、インターネットで閲覧する目的に整理したものは、項番号を付けてあって、索引をするときの手掛かりになるようにしてあります。

12.4.5　英語に訳すことを意識してあること

前の 12.2 節では、他言語に翻訳するとき、中間言語に英語を使うことを解説しました。これを意識して、翻訳調にならず、読み易い日本語にするための文章作成作法について、筆者は幾つかの試みをしてきました。筆者の文体は「です・ます調」です。主語は文頭にきますが、主語であることを示す助詞「は」の直後に読点を入れます。日本語では動詞が文末にきます。主語と動詞が離れ過ぎると、文の理解が難しくなりますので、意識して文単位を短くしています。述語の「です」で終わる文は、英語では be 動詞に当たり、「ます」で終わるのはそれ以外の動詞を当てることになるのが経験的に分かりました。この文体では、形容詞の終止形の使い方が幾らか不自然に聞こえます。例えば「美しいです」のように、終止形が重なります。形容詞は名詞に繋げるのが自然だからです。また「です・ます調」は、否定文が少し長くなるのが欠点です。一方、少し公的な言い方の「である調」では、be 動詞を当てる使い方が基本になりますので、動詞の言い切りのとき、文末表現が苦しくなります。単純に動詞の終止形で済ますこともできます。しかし、ぶっきらぼうに響くことを嫌った「することである・するものである」の言い方が増えます。規則を書くとき、「するものとする・することとする」の言い回しが眼につくのも、心理的には同じ作用の結果です。

12.4.6　感覚的理解が必要になる語を省くこと

実用文書は相手に読んで理解してもらうことですので、自分側が感覚的、感情的に理解していることを、相手も同じように理解するとは限りません。五感と感情に関わる形容詞、副詞、動詞は、自己主張の意義がありますので、客観的な比較ができる言い方以外は、省きます。感嘆詞的な「すごい・格好良い」などの形容詞は、若者が好んで使いますが、何も実質的な意味がありません。動詞の「思う」は日本人が好む語です。これは英語では「I think」に当たり、自己主張の言い方です。論文やレポートは、作者の自己主張の意義があります。客観性を出そうとして「思われる」の受身表現で丁寧に言い方にする例を見ますが、英語の think を受身形で使うことはありません。これらの項目は、自分の原稿を、日を置いて何べんも読み返していて発見されます。

12.4.7　文書の発表形式を三通り準備すること

結論から言うと、筆者の著作は、読者に対して三通りの利用形態を考えています。第一は、従来通りの雑誌形式に搭載する報文です。科学技術系の雑誌は、専門別に大学や協会の図書館で保存され、その気になれば古い記録を探すことができます。第二は、インターネットを利用する方法です。筆者の連載原稿は、連載が完了した時点で、段落単位で HTML 文書に落とし、索引、目次とリンクさせ、関連を付けた段落単位で発信しています。写真などは、元になる雑誌原稿ではモノクロでしか載せられませんが、インターネットに載せる場合にはカラー版が使えます。この利用はモニタでの閲覧が主な目的です。科学技術関係の書籍は、文芸書とは違って、頭から順に読むのではなく、関連する個所の拾い読みも普通です。したがって、読者の立場ではハードコピーにして全体を見たい要望もあります。そこで、第三の形態として、連載全体を PDF 形式に落とし、読者側でダウンロードしてハードコピーが得られるようにしてあります。最近は、PDF 形式のファイルを持ち込めば、小部数であっても印刷・製本サービスをしてくれる街中の印刷屋さんがありますので、自宅でコピー機を持たなくても済むようになりました。

12.4.8　コンピュータ専門家との連携が必要であること

文書を作成することは、その時点での記録を残す意義があります。これをアーカイブ(archive)の用語で表します。一旦作成すると、内容の変更を殆どしません。文芸書は、それでもよく、著作権で他者の再利用を保護しています。しかし、かなりの著作は、発売された時点で購入しなければ二度と見ることができません。一方、教育目的の書き物や実務のデータなどの資料は、他者の再利用が普通ですし、内容の弾力的な変更も必要です。従来の印刷出版の形態であると、動的な対応を殆ど考えません。電子化した文書は、書き換えが弾力的にできます。そのことに対応するように、文書を作成する時代になってきました。印刷物であると、校正ミスは改訂版を出さなければ修正ができません。正誤表を後から出すとしても、それを原本に加えることを読者がすることは、殆どできません。このことについては、電子化文書は素早く対応できます。ただし、筆者の経験から言うと、コンピュータのシステムが変更されると、読めなくなる被害にも遭いました。これは資源の管理の問題です。したがって、こちらの方のコンピュータ専門家との連携が、これからの文書作成、保存、利用の課題です。

12.4.9　検索サービスが今後の課題であること

インターネットの利用は、Google や YaHoo などの検索サービスが強力になりました。しかし、検索方法を工夫すれば、希望の情報が得られると考えていることが、必ずしも現実的では無くなっています。検索のヒット数が多くなると、順位の下の方まで項目に眼を通すことができなくなります。したがって、専門ごとに、お助け的な窓口で情報を整理してくれるサービスが要望されるようになっています。従来から、図書館などは検索サービスをしてきましたし、それを助けるデータベースサービスの機関も活動しています。これをコンテンツサービスと言います。室内の環境でインターネットの閲覧をするときは、かなりの字数の情報をパソコンのモニタ画面で見ることができます。iPad のような携帯端末ではさらに画面が狭いので、そこで利用できるコンテンツをどのように準備するかが問題です。二つ上の段落で説明した第二の利用形態は、このことを意識したコンテンツと考えています。いずれにしても、モニタの画面は狭いので、全体の関係が分かる書物の利用も一定した需要があります。これらのことは、つまるところ、コンピュータと言語との複雑な絡み合いで動いていると考えることができます。

12.4.10　文書の保存は紙に限ること

文書の保存のことを考えるとき、物理的な記録媒体の寿命が問題になります。文書の場合、和紙に墨で書いたものは、虫除けや防湿に注意すれば千年の保存に耐えることが歴史的に証明されています。一方、新聞紙などの酸性紙が自然に変質してボロボロになっていくのは実感されていると思います。銀塩の写真記録は百年の歴史がありますので、マイクロフィルムによる記録の保存は、現状では最も信頼性が高いものです。コンピュータの記録媒体である磁気テープやディスクの歴史は未だ新しく、歴史の試練を経ていません。また、規格が目まぐるしく変ってきましたので、信頼性は高くありません。磁性材料は年月とともに磁性がドロップアウトして読めなくなりますので、定期的なバックアップが必要です。CD などの光ディスクはドロップアウトが無いと言われていますが、まだ歴史的な評価が定まっていませんので保存媒体としては慎重に成り行きを見る必要があります。「橋梁と都市 Project」は、2010 年度を持って休刊しますが、新しい形態の雑誌サービスへの飛躍として、時代を先取りする方法を模索しています。コンピュータの記録媒体が読めなくなることの被害はかなり現実的です。大学などで、研究の基礎データが読めなくなるのは、知的財産の喪失ですので非常に重大なことなのですが、眼に見える建物や器財などだけを財産と考えていると評価を誤ります。現在は、ソフトウエアがハードウエアと同等、もしくはそれ以上に価値を持つ時代になってきました。

終わりに

筆者は大学に居て、多くの学生と顔を合わせてきました。これは、大きな喜びです。科学技術が関係する環境で重要な課題に技能の伝承があります。技術移転(technology transfer)とも言います。英語でいえばノウハウです。技能を文書に書いて残せることができれば、コンピュータがそれを理解することができます。これを信奉したのが人工知能やその応用としてのエキスパートシステムです。しかし「自転車の乗り方」「泳ぎ方」という自習書はありません。技術の伝承と教育とは、人と人と顔を合わせる交流が欠かせません。このとき、言葉で丁寧に説明する技術も大切と考えています。