「情報処理」1年文I/IIクラス11-12　＃2

久野　靖*

1995.10.23

設定を多少更新したので、演習の前に「~kuno/setup bin[RET]」を実行してからloginし直しておいてください。

0　本日の目標

前回は、ある程度内容を絞ったつもりだったのですが、結局時間不足になってしまいました。今回は、前回の後半の復習からやって、次の内容をマスターすることを目標とします。

- Muleを使ってファイルの作成、修正ができるようになる。
- ファイル関係のコマンドについて多少覚える。
- アルゴリズムとプログラムの違いについて理解する。
- プログラムを打ち込んで動かすことができるようになる。
- WWW(World Wide Web)の利用。

また、どこを説明しているか分かりにくい、追い付けない、というご意見も頂いたので、節や演習に次のような印をつけます。

- 「*」 — 説明だけするので聞いてください。時間がないようなら飛ばすこともありますから後で読んでおいてください。
- 「☆」 — 今日の必修箇所。ここで止まって演習を行う。
- 「△」 — option。必修だけではひまな人はどうぞ。忙しい人はあとで眺めておいてください。

*筑波大学大学院経営システム科学専攻

1　復習: X-Windowの窓の役割など/*

前回で、login/logout は皆様できるようになったようですが、窓の役割り (図 1) について混乱が見られますので、気をつけてください。

- 「Console」「ecc-as50」など、「ユーザ名@マシン名>」と表示されている窓 (端末窓) は「コマンドを打ち込んで [RET] キーを押すと実行される」窓である。
- 「Mule:....」の窓は「テキストを打ち込んでファイルに入れたり、ファイルに入っているテキストを修正する」窓である。だから [RET] キーを打つとそのまま「改行文字」が入る。

これらの区別を常に意識すること。あと、プリントの実行例で「%」と書かれているのは皆様の画面での「ユーザ名@マシン名>」を表すものとしますので、キーボードから打ち込まないように。

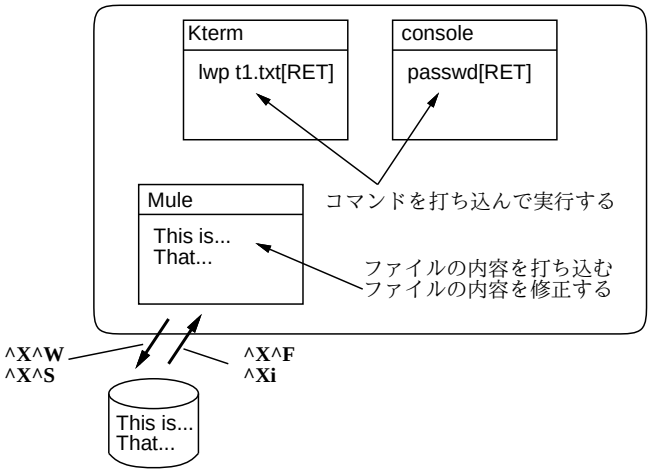

図 1: Kterm の窓と Mule の窓

2　復習: Muleの使い方/*

打ち込んだり保存したりはできるようになったようですが、重要なところなのでしつこく復習させて頂きます。Mule を使う手順は次の通り。

1. 背景メニュー左ボタンで「Mule」を選ぶか、端末窓で「mule &」により実行開始させる。

2. 前回は「いきなり内容を打ち込む」としましたが、それだと^X^Wで保存するときに「*scratch*」という文字列がくっついて表示されてしまう。その場で「^K」を打てば消せるが間違いのもとなので、次のように訂正させていただく。

2.' まず「^X^F」を打ち、ファイル名を聞いて来るのでファイル名を指定して [RET] する。既にあるファイルを指定した場合は、そのファイルが画面に現われて直せるようになる。新しく内容を打ち込む場合にはまだないファイル名を指定する。

3. 以上の準備ができたら普通にキーボードを打つと内容がカーソルの位置にそのまま入る。修正やカーソルの移動はすべて「Control+○」または「[ESC] ○」の形をしている。(前者は Control キーを押しながらもう 1 つのキーを打つ。後者は [ESC] キーを打ってからもう 1 つのキーを打つ。間違えないように。)

^N (next)	カーソルを下に
^P (previous)	カーソルを上に
^F (forward)	カーソルを右に
^B (backward)	カーソルを左に
^A (ahead of line)	カーソルを行頭に
^E (end of line)	カーソルを行末に
^V	1 画面下へ (これらは長いファイルを
[ESC]v	1 画面上へ 修正するとき必要)
[ESC]<	ファイルの先頭へ行く
[ESC]>	ファイルの末尾へ行く
^D (delete)	カーソルのところの文字を消す
[DEL]	カーソルの直前の文字を消す
^K (kill)	カーソルから行末まで消す
	(カーソルが行末にあればその行を消す)
^Y (yank)	^K で消したかたまりを復活

したがって、大量に消すには^K^K... と連打すればよい。また、^Y を使うと^K の連続で消したものが復活でき、さらに^Y を繰り返すと何回でも同じものが追加される (これを利用して歌詞のリフレインなどをコピーできる)。

- 打ち込みや修正が終わったら^X^S でファイルに保存。ファイル関係の指令をまとめておく。

^X^F (find file)	編集するファイル名を指定

^Xi (insert file) 指定したファイルの内容をカーソル位置に挿入
(^I ではなく「ただの i」なのに注意!)
^X^W (write file) ファイル名を指定して書き出し
^X^S (save) 指定してあるファイルに書き出し
^X^C (exit Mule) Mule を終わらせる

^X^S 以外は窓の最下行にファイル名を打ち込んでやる。

3 キーボードをかっこよく打つには…/☆

アンケートの結果、キーボードをかっこよく打ちたいという人はだいぶ多いようなので、少し説明しよう。まず login して Mule の窓を開き、打ち込む用意をする。さてかっこよく、というのは

- 両手の 5 本の指を使って
- キーボードを見ないで (ましてどの字がどこか探さないで)

打てる、という意味だと思うが、こういうのを「タッチタイプ」と呼ぶ。それにはまず、キーボードの上で左手を「ASDF」右手を「JKL;」の位置に置く。これをホームポジションと呼ぶ。では、この状態でハンカチをキーボード全体にかぶせてキーが見えないようにする (代わりに図 2 を見るように)。

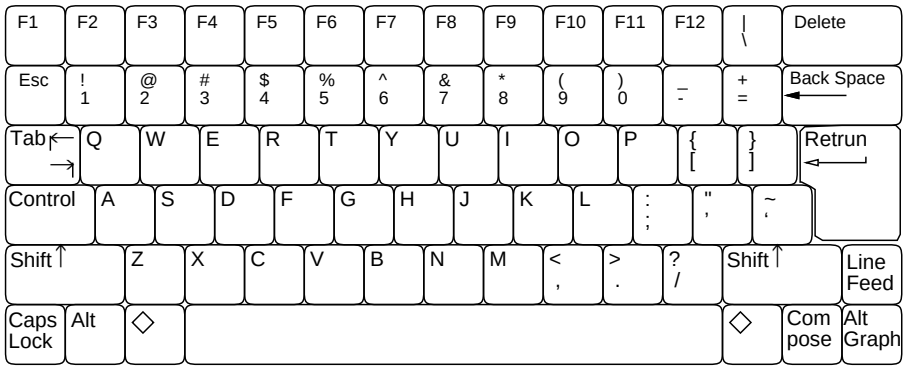

図 2: キーボードの配置図

まず左手で「asdfg」右手で「hjkl;」を何回か打つ。「g」と「h」の時は左手と右手の人さし指を横へ延ばして打つ。自信ができるまでは、キーを手探りして確認するとよい。あと、右手小指をうんと横へ延ばして [RET]

キーを打つ練習もしよう。このキーはばかでかいから触ってみればすぐわかる。

では、今度は次の単語を数回ずつ打つ。「sad」「fad」「gas」(以上左手のみ)「has」「jaf」「kasa」「hal」「jal」(右手も含む)。語の間には `[SP]`(スペースバー) を親指で打って間隔をあけ、また適宜改行する。位置が分からなくなったらプリントの図を見る。間違えた場合は取り消さず全体を打ち直す。

タッチタイプという感じが分かりますか。では右手の上段と下段を加えよう。指を上に伸ばすという感じをつかむ。「jus」「mad」「fuss」「mass」「kid」「as, is」「loss」「hold off.」「;pad」「gap」「up/off」どうですか。では左手。「qasi」「zap」「was」「sax」「desk」「cap」「frog」「have」「save」あと残っているのは、左手と右手を伸ばした位置の上と下で、これが実は一番大変。「hear」「year」「near」「gear」「tear」「bear」。時々これを復習して、あとは自分で工夫してみてください。

演習☆ 少し時間を差し上げますので、上の説明にならってできるところまでタッチタイプの練習をしてください。最初に`^X^F` でファイル名を指定し (`touch1.txt` とかいう名前にしてみよ)、終わったら`^X^S` で保存すること。

演習△ すでにタッチタイプできる人は「trr」というタイプ練習システムがあるので試してみてください。

- Mule を動かす。
- `[ESC] x trr [RET]` と打つ。
- trr が起動し、画面が 3 つに分かれる。ここで一番上の部分の例文と同じものをこれから打ち込む。
- `Are you ready?`と表示されているので、「`y`」と打ってからひたすら例文と同じものを打ち込む (間違いがあると打ち直しになる)。`^M` のところは `[RET]` で改行する。
- 1 画面打ち終ると打鍵速度が表示され、`Continue?`と表示されるので、やめたければ「`n`」もっとやりたければ「`y`」を打つ。

4 ファイル関係のコマンド/☆

さて、このように Mule でファイルを作ったり修正できるようになったわけだが、自分がどんな名前のファイルを持っているかなどすぐに忘れてしまうのが普通である。また、ファイルの中身をプリンタに出したいこともある。そのような場合は、Kterm の窓で各種のコマンドを使うことに

なる。くどいようだが繰り返すと、Mule ので窓はファイルを打ち込んだり直したりでき、Kterm の窓では様々な指令を実行できる (図 1)。この区分に早く慣れていただきたい。なお「Console」という窓も Kterm の窓の一種である。

ファイルを管理するコマンドの一番の基本は、ls(list files) である。このコマンドを単に「ls[RET]」のように実行すると、自分の持っているファイルの一覧が表示される。

```
% ls
Mail/          bin/           test1.txt
Wnn4/          lib/           test2.txt
%
```

なお、上で「%」と出ているところは、皆様の初期設定では「ユーザ名@マシン名>」という形のものになっている。これは「プロンプト (コマンド入力促進記号)」といって、「前のコマンドが終わったから、次のコマンドを入れてください」という意味を持っている。

ところで、ファイル名一覧の中には Wnn4/のように自分に覚えのないものも見えるが…これらは管理者が初期設定として用意してくれたものなので、とりあえず触らないでおく。実は、これ以外にも初期設定用ファイルは一杯あるが、それらはすべて「.」で始まる名前になっている。ls は通常は (初期設定ファイルの山が見えるとうるさいので) これらを表示しないが、「ls -a[RET]」とした場合にはこれらも表示する。[1]

```
% ls -a
.              .emacs         .mnews_setup  Mail/          test1.txt
..             .history       .newsrc       Wnn4/          test2.txt
.Xauthority    .login         .twmrc        bin/
.cshrc         .mh_profile    .xsession*    lib/
%
```

次に、ファイルの中身を確認するのに、いちいち Mule を動かさないでも Kterm の窓だけでやることができる。それには「cat ファイル名 [RET]」による。[2]

```
% cat test1.txt
....(ファイルの内容)...
%
```

[1] なお、末尾の「/」と「*」はある目印であって、ファイル名ではないので注意。つまり「.xsession*」ではなく「.xsession」というファイルがある。

[2] cat はファイル名を入れ忘れると一見何も反応しない状態になってしまう。そうなった場合には Control-C を打てば中止できる。

cat は簡便でいいのだが、ファイルがちょっと長いと画面が「流れて行って」読めなくなってしまう。そのような場合は代わりに「less ファイル名 [RET]」を使う。less を使うと、1 画面が一杯になるとそこで止まってくれる。先をみたい場合は [SP](スペースバー) を打つ。また、[BS] を打つと 1 画面戻ることができる。いつでも「q」を打つとそこでやめることができる。あと、要らないファイルは

```
% rm ファイル名
```

で消せるので適宜整理してください。ファイル名に「*」などの記号が入っている場合には

```
% rm 'test1.txt*scratch*'
```

のように全体を「'」(シングルクォート) で囲む必要があります。

演習☆ ls で自分の持っているファイルの名前を確認し、cat で内容を表示してみよ。less ではどうか。

演習☆ ls -a で初期設定ファイル群の名前も調べ、いくつか cat や less で内容を表示してみよ。

演習△ less /usr/X11R6/lib/X11/rgb.txt[RET] により、X-Window の色名一覧ファイルを眺めてみよ。

演習△ Mule で.twmrc、.xsession のいずれかを^X^F で読み込む。その中に色の名前があるから、それを自分の好きな色で適宜置き換えて ^X^S で保存する。その後 login し直して結果を観察する。

ヒント: 名前だけでどんな色か分からないようなら、「colors &」というコマンドを実行してみよ。

5 プログラムとアルゴリズム/*

ここまででは、ある加工に対して、それを行うためのコマンドが既に存在していて、それを示す情報を計算機に与えればいい、という前提で話をしてきた。しかし、現実には自分のやりたいことすべてが予めコマンドとして用意されている、などということはもちろん不可能である。

ではどうするか? それには、「どのような加工をしたいか」という情報を、(またまたビット列として) 計算機に与えればよい。この情報のことを (散々聞いたことがあると思うが)「プログラム」という。実は「既にあるコマンド」というのは、「あらかじめ計算機に入力されて蓄えられているプログラム」に他ならない。

言い替えれば、計算機というのは実はプログラムに従って情報を処理 (= ビット列を加工) する装置なわけである。そして、どのような処理であっても (たとえその計算機が製造された時には夢想だにされなかったようなものでも)、その処理方法をプログラムとして与えさえすれば処理できるようになる、というところに計算機の特徴があるわけである。というわけで、この科目の目標の 1 つとして皆様にプログラミングとアルゴリズムについて学んでいただくことになっているわけである。

なお、ここで「プログラム」と「アルゴリズム」ないし「手順」の違いを整理しておく。「アルゴリズム」ないし「手順」といった場合、それは加工の方法自体をいう。例えば駒場地区正門から渋谷駅まで歩いて行く行き方、といったものである。一方、「プログラム」といった場合はアルゴリズムを計算機に与えられるような形に表現したものをいう。例えば道筋を他人に教えるためには、行き方を記した地図やメモといった具体的な形に表現する必要がある。

課題 1 ☆ 「アルゴリズム」と「プログラム」、「行き方」と「行き方を示した地図」のような関係にあるものの例を他にも挙げてみよ。

課題 2 △ 「行き方」または課題 1 で挙げた相当物について、それを具体的に表現する方法を 3 つ以上挙げ、それらの方法の優劣について論じよ。

6 計算機のためのアルゴリズム/∗

計算機で使う「アルゴリズム」ないし「手順」は、何かを求めるための具体的な計算 (ないし情報の加工) 方法でなければならない。たとえば、海外のニュースなどを見ていると気温が華氏で表示されているので、それは摂氏では何度かな、と知りたくなる。それを求めるには、華氏の温度を f として、

$$c = \frac{5}{9}(f - 32)$$

により値 c を求めればよい。これも立派なアルゴリズムである。

ところで、何をもって「具体的」というかは実は簡単ではない。たとえば、n 個のボールを (すき間があってもいいから) 正方形の箱に平らに入れたければ、その箱の 1 辺の長さはボールの直径の

$$l = \lfloor \sqrt{n} \rfloor$$

倍であればいい。ただしこれは、「切り上げ」とか「平方根」とかの計算手順が具体的にわかっていれば、である。もし加減乗除だけしか使えない

のなら、例えば

$$l^2 \geq n$$

なる最初の l が見つかるまで $l = 0, 1, 2, 3, 4, \ldots$ を順に試していく、という手順を使うことになろう。

実は、四則演算も「具体的」かどうかは議論があってよい。たとえば小学生はどうやって四則演算をやるかの手順を一生懸命憶えさせられるわけである。しかし幸いなことに、計算機には四則演算のための機能がもともと備わっているから、それにお願いすることにして、まずは四則演算は「具体的」だということにする。

課題 3 ☆ 仮に、かけ算と割り算は「具体的でない」(つまり使えない) ものとする。足し算と引き算と数の大小比較は使ってよいとして、次のことをするアルゴリズムを考えよ (ただし x、y は正の整数とする)。

a. ☆ 数 x が奇数か偶数かを判定する。

b. △ 数 x と y の積を求める。

c. △ 数 x と y の最大公約数を求める。

7 アルゴリズムの表記方法と PAD/☆

アルゴリズムを計算機での処理に使おうと思うと、「具体的」の他にも色々問題がある。まず、上の温度の例では「華氏の温度を f として」などと書いてあったが、人間が手で計算するならそれでいいとしても、計算機で処理する場合にはそこで「華氏の温度を計算機に読み込ませる (入力する)」という動作が必要である。その次に式にしたがって四則演算を行ない、最後に結果を「人間に見えるように表示する (出力する)」動作が必要になる。これらのことを順番におこなって始めて、計算機で仕事がこなせるのである。

そこで、計算機の分野ではもっとこの「入力」「出力」「順番の動作」がはっきりするような書き方を工夫することが古くから行なわれている。その古典かつ代表が「フローチャート (流れ図)」だが、現在ではこれは様々な弱点があってよくないとされている。ここでは PAD 図と呼ばれるものを使用する。上の温度変換の手順を PAD で記したものを図 3 に示す。

ここで、箱は何らかの計算機による処理を表している。特にそれが入力、出力の場合には、それぞれ右上、右下が斜めに欠けた箱を使う。箱の連なりが縦線で結んである場合には、それらを上から順にやっていくことを示す。そして、ある箱から右に引き出して記述がある場合には、その箱に書いてあるのは概略で、より具体的には右側の記述によること (詳細化) を示す。

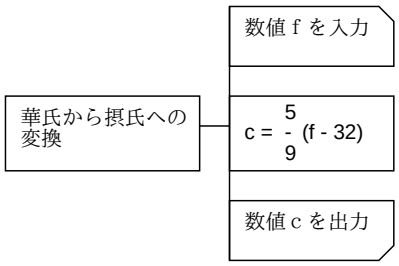

図 3: 華氏摂氏変換の PAD

ここにあるように、どんなプログラムでもまずそのプログラムが全体として何をするかを書き、その詳細化として手順を書いていくのがよい習慣である。

課題 4 ☆ 次のような計算の手順を PAD で書いてみよ。今度は四則演算は使ってよい。

a. ☆ 二つの数 a、b を入力し、その合計を出力する。

b. ☆ あるものの長さが X メートル Y センチ Z ミリの形で表されていたとして、それをセンチ単位に換算する。(例えば 1 メートル 10 センチ 5 ミリ→ 110.5 のように。)

c. △ 同じ日の 2 つの時刻 A 時 B 分 C 秒と X 時 Y 分 Z 秒を読み込み、その間の時間を秒単位で出力する。

8 プログラミング言語 Pascal/☆

「プログラム」とは、アルゴリズムを実際に計算機与えられる形で表現したものだった。さっきやったように、アルゴリズムのようなものを書き表す方法はいろいろある。これまでは日本語とか数式とか PAD 図を使って書き表していた。それに対して、計算機で計算させるためには日本語や数式ではやりにくいし、PAD 図では読み込ませにくいから、もっと杓子定規な書き方の規則を設けて、それに従って書く。これがプログラムである。

その「書き方の規則」も実は様々な流儀がある。つまり各種のプログラミング言語に相当する。どの流儀も、それなりの長所と弱点がある。ここでは Pascal と呼ばれる言語を用いる。Pascal はチューリヒ工科大学の Writh 教授が設計した言語であり、教育用によく使われている。また、強い型検査機能のためプログラムの間違いが発見しやすい。反面、大きなプ

ログラムや、システムプログラミング (各種システム機能を駆使するようなもの) にはあまり向いていないとされている。

では、先の華氏と摂氏の変換プログラムを Pascal で記述してみる。

```
program sam1(input, output);
var c, f: real;
begin
  write('degree F> ');
  readln(f);
  c := (5.0 * (f - 32.0)) / 9.0;
  writeln('degree C = ', c:5:2)
end.
```

先の PAD と比べてみると、いろいろ細かい記述が増えているなあと思う。

1. プログラムは「`program` プログラム名 `(input, output);`」という書き出しで始まらなければならない。(しかも名前には英小文字と数字しか使えず、数字で始まってはいけない。)
2. 値を表す名前 (変数) は「`var` 変数名, `...: real;`」によって予め使用を予告しておかなければならない。
3. `begin` から `end.` までの間に手順を順次; で区切って書く。なお、この区切られた各 1 手順を Pascal では「文」と呼ぶ。
4. 入力は `readln`、出力は `writeln` で表す。なお `write` は改行なしになること以外は `writeln` と同じ。
5. 加減乗除は`+-*/`でそれぞれ表す。1 行に書かないといけないので、適宜 `()` でくくる必要がある。
6. 変数に値を設定するには「代入」と呼ばれる記号`:=` を使う。代入は「等しい」とは違う意味 (cf. `x := x + 1.0`)。

さらに `writeln` では文字列を書く時には'' で囲む、数値を書く時には「値:幅:桁数」の指定を行なうと、全体として指定した幅以上の文字数を用いて、少数点以下を指定した桁数ぶん表示する、といったこともおぼえる必要がある (ああ面倒だ!) [3]

要は、プログラミング言語というのは計算機に対して実際にアルゴリズムを実行する際のありとあらゆる細かい所まで指示できるように決めた形式であり、だからプログラムのどこか少しでも変更すると計算機の動作もそれに相応して変わるか、(もっとよくあることだが) そういう風には変えられないよ、と怒られることになっている。いくら怒られても偉いのは人

[3]桁数を指定しない場合には、指数形式での表示になる。

間であって計算機ではないのだから、そういうものだと思って許してやって頂きたい。

では実際にこれを計算機で動かそう。まず、Mule で上と同じ内容を「sam1.p」というファイルに書いてもらう。Pascal プログラムを入れるファイル名の最後は必ず「.p」になっている必要がある。次に Kterm の窓で「pc ファイル名」というコマンドを実行して、このファイルを実行できる形式に変換する。変換されたプログラムは「a.out」という名前のファイルにできる。そして、そのファイルの名前を言うとプログラムが実行される。

```
% ls
sam1.p              ←プログラムを入れたファイルがある
% pc sam1.p         ←変換する
% ls
a.out*  sam1.p      ← a.out というファイルができている
% a.out             ←その名前だけを入力→実行開始
degree F> 60        ←データの入力
degree C = 15.56    ←結果の出力
%
```

苦労のわりにはあんまり大したことはない感じだが、まあ初心者の第 1 歩ということで、そうがっかりしないで戴きたい。この先は来週のお楽しみである。

課題 5 ☆ 例題の華氏摂氏変換プログラムをそのまま、打ち込んで実行させてみよ。動いたらその窓のハードコピー出力を取れ。[4]

課題 6 △ 例題の writeln のところにある幅指定や桁指定の値を変えたり、桁指定をなくしたりしてプログラムを動かし、出力の表示がどう変化するか調べよ。気に入った出力例ができたらその窓をハードコピー出力せよ。

課題 7 △ 課題 4 で描いた各 PAD を Pascal のプログラムに変換し、打ち込んで動かせ。

[4]それには、別の Kterm の窓を用意してそこで hardcopy -Pプリンタ [RET] と打ち込む。するとマウスカーソル十字形になるので、ハードコピーしたい窓の上でクリックすればよい。

9 World Wide Web/*

つらく苦しい :-) プログラミングだけではつまらないと思うので、もっと楽をして楽しめる話…つまり、他人が作った情報を気軽に見るだけ、というシステムを体験して頂こう。

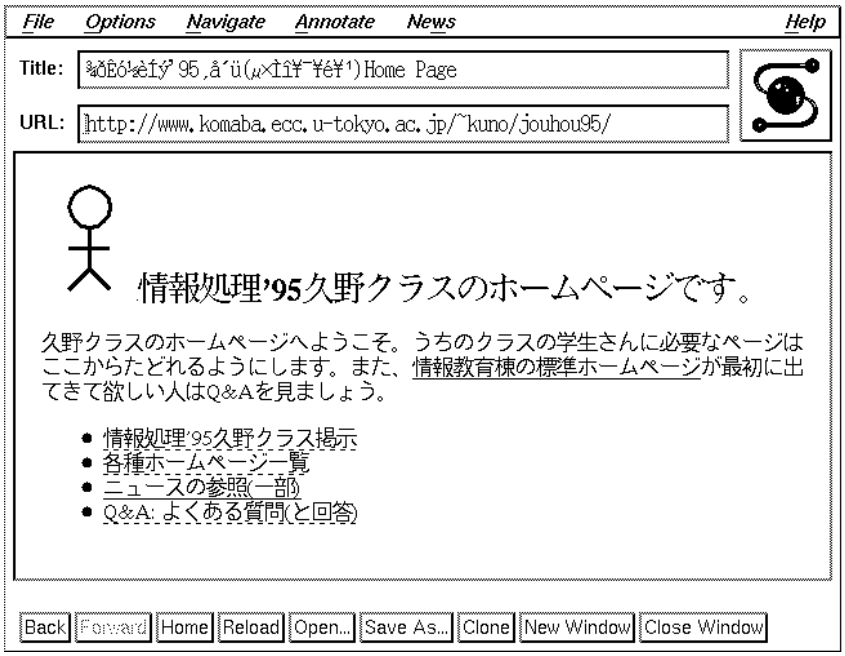

図 4: Mosaic による WWW の検索

さて、ここで説明するシステムは World Wide Web(WWW) といい、世界中のネットワーク情報がくもの巣のようにつながっていてどんどんたぐれる、というスグレモノである。「テキストエディタ」の一例として「Mule」があるように、「WWW のビュア (見るためのプログラム)」の一例として「Mosaic」があり、ここでもそれを使って頂く。その動かし方は簡単で、背景左ボタンメニューの「Mosaic」を選ぶかまたは Kterm の窓で「`xmosaic &`」を実行するだけである。すると、新しい窓が開いて図 4 のようなものが見えるはずである。

ここで、窓の中の「下線」が引いてあるところにマウスカーソルを持って行って左ボタンでクリックするとその情報ポインタがたどられて別の画面に切り替わる。ただこれだけで、どこまででもネットワークの中を進んで行けるのでまずは体験してみて頂きたい。なお、行きすぎて戻りたいときは画面下にある「Back」の箱、満足したので終わりたいときは同じく「Close Window」の箱をクリックすればよい。

重要! 今後、授業に関する掲示を WWW で行う。この科目のホームページ (標準設定で最初に出て来るページ) から「科目の掲示」をたどった所は 2 日 1 回程度はチェックするように!

課題 8 △ WWW で目新しい情報を検索してみよ。東大とか手近なところでなく、なるたけ地の果てで、理系ぽくない情報が好ましい。

課題 9 △ 「ニュースの参照 (一部)」という項目の下には、`komaba.*`という電子ニュースのグループに属する記事が見えるようにしてある。電子ニュースとはどんなものかなんとなく分るまでこの中を探検してみよ。

A 本日の課題 2A

今日は「課題 1」「課題 3a」(以上手書き)、「課題 5」「できた人は課題 6」(以上プリンタ出力) を提出してください。レポート番号は 2A です。

以下にレポートの守るべき要件を再掲しておきますので、気を付けてください。守られていない場合には採点の公平を保証しません。

- 提出する紙が 2 枚以上にわたる場合にはホチキスで綴じること。
- 少なくとも一番最後の紙は裏が白紙 (印刷されてない) であること。
- その白紙部分に次のものを上 (綴じた側) からこの順で記入すること。
  - 提出日付、四角く囲ったレポート番号、学籍番号、氏名 (以上 1 行に)
  - アンケートの回答
  - それ以下の余白部分は好きに利用してよい

  前回の提出で無記名のがありました。注意してください。

アンケートは次の通り (そんなに大量に書く必要はありません)。

Q1. 今日の内容は難しかったですか? 難しいとすれば、どの辺ですか? またどこをもう少し工夫したらよくなると思いますか?

Q2. プログラムについてどう思いましたか? 紙で考えていた時と、動かした時で考えは変わりましたか?

Q3. 本日の全体的な感想と今後の要望をお書きください。

B　次回までの課題 2B

次回授業開始までに「課題 4 の a〜c のうち 1 つ以上 (できれば全部)」、「課題 7(課題 4 でやったものでいい)」、「課題 8」を提出してください。課題 7 と 8 はそれぞれファイルのプリントアウトと実行例のハードコピーで出してください。

なお、課題 7 や 8 は他人のを写すのは容易ですが、まず自分なりに (資料を参考に) 努力して見ることをすすめます。どうしても判らない場合には丸写しではなく要点を聞くように。ここで写してしまうと、自分で書けるようになりませんし、後の方のレポートではメールでの提出になるので、プログラムの丸写しは計算機によって検出することができてしまいます (当然、発覚した場合にはそれなりに採点します)。

提出先は 1F 自習室内のレポートボックス、レポート番号は 2B です。アンケートは次の通り。

Q1. プログラミングの課題はどれくらい大変でしたか? PAD 図を描くのと、Pascal に直すのと、打ち込んで動かすのとで掛かった手間の比率はどうですか?

Q2. WWW についてどう思いましたか? どんな (自分にとって) 興味深い情報が得られましたか? (または得られると思ったが結局得られませんでしたか?)

Q3. 課題に対する感想と今後の要望をお書きください。

課題は、次回授業開始時刻までに、レポートボックスに提出してください。

C　今後の資料の入手とプリンタ関連事項について

資料を当日より前に入手したいと思われるのは当然だと思います。そこで、前の週の後半 (木曜くらい? にできるよう努力します) 以降に

```
~kuno/handsout -Pprinter 資料番号 [RET]
```

を (Kterm の窓で打ち込んで) 実行すれば資料がプリンタに打ち出されるように設定するつもりです。準備ができてない時や、授業当日の正午以降はプリントできない旨表示されます。lpq でプリンタの混雑を確認してから実行するようにしてください。

なお、各端末にはプリンタ名が貼ってありますが、それに関係なく「混んでいない」プリンタを選んでくださって結構です。`lwp`、`hardcopy`、`handsout`、`lpq`、`lprm` のどれでも、「`-P`プリンタ名」という指定を追加することにより特定のプリンタを指定できます。例えば

```
% lwp -Plw01 test1.txt
% lpq -Plw02
% lprm -Plw03 12    ←※
% hardcopy -Plw04
% ~kuno/handsout -Plw05 2
```

のような具合です。活用してください。また、間違って余計なものを打ってしまったり、混雑してるプリンタをあきらめて別のプリンタに出し直す場合には必ずいらない出力を※のように lprm でキャンセルしてください。ここで指定する番号は、lpq の表示で「Job」という欄に書かれた番号です。

さて、プリンタの紙はすぐなくなるので、なくなったら自分で補充してください。紙はプリンタ専用紙を使うこと! 使用済の紙の裏とか、自分のレポート用紙などは間違っても入れない! (簡単にプリンタが壊れます。) まっさらの専用紙が見当たらない場合は 1F 事務室に行ってもらうこと。補充方法は

1. プリンタ前面の引き出し (上) をひっこ抜く。
2. 紙をきれいに揃えて入れ (ガイドに記した横線以上に入れないこと!)、その手前両すみを金属のつめに引っかける。
3. 引きだしをもと通りに戻す。

これでプリンタについてる表示が「01」にならない場合は紙づまりをチェックしてください。

1. 再度引き出しを抜いて抜いた中をのぞき、引っかかった紙がないかチェックする。
2. 上のレバーを引いて本体を開け、引っかかった紙がないか見る。
3. 引き出しの紙のセットが正しいことを確認してすべて戻す。

引っかかった紙を取るときはやぶかないように。破れた紙はただの詰まった紙よりずっと厄介です。取るのが難しそうだったり、上の通りやっても直らない場合は 1 階事務室に相談に行くこと。また、経費節約のため、トナーランプがついても本当に印刷がうすくなるまではカートリッジを交換しません。本当にうすくなったらうすくなった紙を持って事務室に行き「どのプリンタのカートリッジが本当にない」と申し出るように。