CLI::LaTeX::Table のインストールと最初の使い方

連絡先: bin4tsv@gmail.com

2018 年 6 月 7 日 (木) – 第 1 版
2018 年 6 月 9 日 (土) – 第 2 版
2018 年 6 月 12 日 (火) – 第 3 版
2018 年 6 月 17 日 (日) – 第 4 版

この文書は、CPAN にアップロードされたモジュール CLI::LaTeX::Table について、インストール方法などを解説する。他の文書に書き切れなかったことで、本モジュールのインストールの方法と試す方法について、作成者の意図がきちんと伝わるようにこの文書で説明をする。ただし本モジュールの作成者のモジュール配布の経験は半年以下であり、間に合わせの知識でこの文書を作成しており、もっと良い方法はあり得る。実際の利用場面において、よく読むべき箇所は 1 ページ程度であると考えられ、残りは本書の太字の部分をさっと目で追うだけになることも多いかもしれないが、この文書の作成意図に従い、記すべきと思われることはできるだけ網羅したため 10 ページを超えている。

目次

第 I 部

CLI::LaTeX::Table のインストール

1　前提となること (インストール前に)

1.1　本文書の作成方針

本文書は、「全く知識が無くてもすぐ使えるマニュアル」にはなっていない。**「初心者であってもキーワードを元にインターネットで調べたり試行錯誤を何度か繰り返せば意味の分かる手順書」**かつ**「いくらか知識のある人がさっと読んで要領の理解しやすい手順書」**になっていることを意図して書かれている。

1.2　想定する計算機環境および利用者のスキル

Perl 5.14 以降[*1]が使える環境であればどんな環境でも使えるように、本モジュール CLI::LaTeX::Table は作られている。すなわち、大学、政府および国際機関、インフラ系企業、金融機関などで、システム更新が近年なされていない場合でも、計算機環境が先進的でもそうでなくても、個人の環境でも、問題なく動くように試みられている。そのため、Perl 5 の知識があれば、意図しない動作をしない場合に迅速に対処できる程度に、プログラムは簡素に作られている。[*2]

[*1] Perl のバージョンの確認方法は `perl -v` を実行することである。5.14 は 2011 年に公開。2018 年の最新版は 5.26 である。

[*2] Perl 5.14 でも新しすぎる場合は、手でプログラムをたとえば 5.1(Perl5.1 は 1994 年公開) と書き換えて、エラーを起こす部分の機能を制限または修正するなどの方法が考えられる。モジュールが提供するファイルで Perl 言語で書かれているプログラムは、何かエラーがある場合は、基本的には、そのエラーを起こしたファイルのみを手を加えれば修正が可能である (エラーを起こしているプログラム以外のファイルを修正する必要は、本モジュール CLI::LaTeX::Table の場合、原則的に無いはずである)。そして、最初の方に `use 5.014` とあるので、そこでエラーが起こる場合はそこから直していくことになる。

Unix でも Linux でも Windows でも、Mac OS X でも使えるが、本文書作成者は、Mac OS X 上の Unix 環境及びクラウドサービスの AWS (Amazon Web Service) の EC2 の Linux 端末で検証を行っている。bash と zsh のシェルで検証をしているが、それ以外の csh などのシェルを使う場合は、読者が下記の文書の意図を読み取って独自調査をしてインストール及び利用をする必要がある。

本モジュールの利用に際し、利用者が **Perl** 言語のプログラミングをする能力は必要とはしない。(それでも何か不具合があった場合に、意図した通りの動作をさせるためには、Perl 言語の知識は役に立つ。) むしろ Unix 環境もしくは Linux 環境で bash や zsh などのシェルについて基本的な使い方をマスターしていることは、利用のしやすさに大きく影響する。リダイレクションでファイルの読み書きが出来ること、様々な基本的な Unix コマンドを使いこなしていること、パイプで複数のコマンドを連結して使うこと、さらにはプロセス置換で余計な中間ファイルを作らずにいろいろな操作が出来ることは、本モジュールを使いこなす上ではやや大事である。

1.3 なぜこのソフトウェアを作ったか

まず、本モジュール CLI::LaTeX::Table で本質的なのは、約 5 個程度のそれぞれ単独で機能する Perl 言語で書かれたプログラムファイルである。[*3]それが提供するプログラムの 1 つ `latextable` は、タブ文字区切り形式 (TSV 形式) の表を LaTeX に使えるように変換するコマンドラインインターフェース (CLI) である。

表計算ソフトのエクセルや SQL(DB 管理システムのクエリ言語) の出力結果の多くは、クリップボードを経由したコピーとペーストの操作 (「コピペ」) により、他のソフトウェアに任意の領域を書き写すことができる。そして、書き写した先の、たとえば、ワードなどの文書作成ソフトによりいろいろな分析の結果をひとつのレポートにまとめるような作業は、データ分析などを業務とする場合は日常的に発生する。ところが、2018 年にいたってなお、多くの文書作成ソフトにおいて、多数の表をひとつの文書にまとめるとその文書の計算機 (パソコンなど) での操作が不安定となる[*4]。近年 (特に 2013 年に Big Data がブームになって以降)、企業等に蓄積されたデータの分析の需要が増しているが、ひとつのたとえば 10 列で行数は数千または数億またはそれ以上のテーブルひとつの意味を読み取るために、SQL で 10 個ぐらい何かを集計して、ひとまとまりの文章にしようとしたら、壁にぶつかることになって、それ以上先に進みにくくなることは容易に起こる[*5]。

LaTeX であれば多数の表をまとめても安定して動作するので、そのような問題は (容易に) 大幅に改善する。LaTeX の他の良い点は、(慣れてしまうと) 文章の体裁の変更が容易で、データ分析で得られた知見を後日必要

[*3] 本モジュール CLI::LaTeX::Table は、CPAN にアップロードできるモジュールの体裁を為すような目的のために、本質的なプログラムファイル以外にもいろいろなファイルが付属しているのであって、非常に制約のある環境である場合 (メモリやハードディスクが小さすぎて、標準的なインストール方法すらうまくいかない場合など) や、単にそれをしたい場合には `https://metacpan.org/source/TULAMILI/CLI-LaTeX-Table-0.56/scripts` などからのリンク先 や GITHUB から使うプログラムをひとつコピーしてペーストする使い方でも全然構わない。

[*4] 少し編集を加えようとしてもフリーズをしたり、意図しない文書の配置になったりし、その結果、スムーズな文書編集作業が妨害されて、意味のある分析が困難になる。また、起動に時間がかかったりして、可能であれば多数の文書ファイルの中身を閲覧したいのに、少数のファイルのみで我慢をしなければならなくなる。

[*5] とりあえず、ひとつのテーブルを与えられたら、たとえば次の様な中間分析をした表が生成されるであろうし、後の分析のために分かりやすくひとまとめにする必要がある。各列が異なる値をいくつ持つか/最小の値と最大の値/最頻値などをまとめた表、各列の null/特別値/テスト値/異常値/破損値を探索してその結果をまとめた表、着目した列が重複した値を持つ様子を表す表、着目した複数の列に対して 2 元分割表にして値の組みあわせの様子をまとめた表、具体的な値をみるために、数十行をランダムサンプリングした表、金額などの数値に対して合計または分位点をまとめた表、どの列がキー列となるか考察するために、W 列の中から 1 つずつ列を除去して重複が何通りあるかをまとめた表など。他の表があれば、どう結合するかを検討する為に表を用意すると、さらに増えることになる。

に応じて参照しやすい形の作成がしやすいことである。従って、そのような表を、各行は改行文字区切りで各列はさらにタブ文字区切りの文字列に変換したものとして扱うことができる。

したがって、コピペでクリップボードに書き写した表形式のデータをすぐに LaTeX の表 (\begin{table} で始まり、\end{table} で終わる LaTeX のコマンド) に変換出来るコマンドが便利であるので、2018 年 2 月ころより作り始めたのが、このソフトウェアを作った経緯である。

そして、`latextable` を CLI::LaTeX::Table で提供するにあたり、それに入力するテストデータを生成する `saikoro` (一様分布の乱数行列を生成)、行列の縦と横を交換する便利な機能を提供する `transpose`、テーブルの列を AWK 言語や cut コマンドよりも自在に (正確には容易な指定方法により) 扱える `csel`、CSV 形式 (RFC4180 で策定のダブルクォーテーションで囲んだコンマ文字で列区切り) を TSV 形式に変換するユーティリティー `csv2tsv` を同梱する。これらの各コマンドは `latextable` と同様のインターフェースの考え方が反映されている。それは、便利な機能をコマンドと、そのスイッチオプションと、必要に応じてそのパラメータを指定することで、簡単な指示は現状複雑になりがちなプログラミング無しに迅速に実行できるようにすることである。

2　インストールの方法

この 2 節では、CLI::LaTeX::Table のインストール法及び関連するソフトウェアのインストール法を記載している。下記のインストールの方法のどれかが完了したら、次の 4 節の「各コマンド使い始める」を参照のこと。

2.1　インストール前の注意点

インストール方法の選択:

本モジュールのインストールの方法は、以下のどれか 1 つであり、利用者の状況に応じて選択する。[*6]

4. cpan を使う方法
3. cpanm を使う方法
2. Module::Starter の Build を使う方法
1. ディレクトリ scripts 下のプログラムにパスを通す方法

上記 4 通りのインストールの方法を決めるには、表 1 による比較が参考になるであろう。**cpan** がインストールされていない場合それをインストールするには通常 root 権限 (管理者権限) を必要とするし、そのインストールには数分間画面を注視するような 作業を要する。近年、**cpanm** の利用も多いと思われるが、それが未インストールの場合は cpanm のインストール作業が数分間必要である。またこれら適切に使いこなすには[*7]、様々な知識の習得があらかじめ必要と考えられる。Module::Starter の **Build** を使う方法と**パス**を通す方法は、利用者が metacpan のサイトから本モジュールをダウンロードして、`tar xzvf` または `tar -xzvf`

[*6] どれを採用するのが最適であるかについては、利用者の計算機環境での権限、本モジュールを迅速にインストールしたいかどうか、計算機環境の変更を最小限にしたいかどうか、本モジュールを試しに使うかもしくは恒久的に使うか、利用者の外部ライブラリの管理ポリシーなどに依存する。

[*7] たとえば次の方法を **cpan** 及び **cpanm** について知ることが望ましい: 計算機環境に不具合を起こさないかテストする方法、ユーザー権限のみあれば実行可能なローカル環境にインストールする方法、アンインストールの方法。

もしくは `tar zxvf` などで解凍する作業を伴う。

インストール直後に記録/記憶すべきこと:

今後のインストール更新やアンインストールに備えて、どの方法でインストールをしたか、どの権限 (root かその他のユーザーか) でインストールしたか、どこのディレクトリにプログラムをインストールしたか (手動設定をした場合) は、必要な場合に思い出せるようにする必要はある。また、インストール時に見つけた不具合、もしくは誤操作をしそうに思われた箇所は、対処のために記録することが望ましい。

2.2 インストールの手順 (4 通りから選ぶ)

この節で下記は 主に Unix/Linux 環境を仮定する。しかし他の OS であっても十分に参考になるであろう。

本モジュール CLI::LaTeX::Table をインストールする方法 4 個の比較

本 module の install 法→	4. cpan コマンドの方法	3. cpanm コマンドの方法	2. Build の方法	1. 手作業展開とパス設定
作業前に必要な環境:	cpan install 済みの環境	cpanm intall 済みの環境	`Module::Starter` 事前 install	Perl のみ :-)
その環境の準備の手間:	cpan の初期設定に数分間	cpanm の install	おそらく cpan や cpanm に頼る	皆無 :-)
管理者 (root) 権限の必要性:	必要がある場合が多い	install 先 directory による	install 先 directory による	不要 :-)
本 module の install の手間:	`cpan` module 名 で一発 :-)	`cpanm` module 名 で一発 :-)	**download, 解凍, コマンド数回**	左に加え、**PATH 設定作業**
ありがちな追加の手間:	cpan 設定の理解と右の (#)	cpanm の install 法の選択と (#)	`Module::Starter` の install(#)	**コマンド格納 directory の検討**
install 先 directory の指定法	cpan 起動して o conf など	`cpanm -l` foo	`Build install --intall_base` foo	手作業; mkdir など
アンインストールの方法:	難度高 (別環境で変更箇所を特定)	cpanm -U (実験的)	書換/書き加えのファイルの追跡	PATH 設定の書き加えの除去など
計算機環境の破損の危険:	root 権限で目で追いきれない操作があるのでありうる。ただし、この 8 年間実例を聞いたことがない。			install 作業時にまず起きない
既存のコマンド上書きの危険:	あり得る	あり得る	あり得る	手作業で確認するので起きにくい
既存コマンドと名前衝突:	あり得る	あり得る	あり得る	手で名前を書き換えて回避可能

表 1　この表の情報は確実とは限らない。最後の 3 行は誇張の可能性もある。文字幅を減らすため、インストール、ディレクトリは英単語で表記した。CPAN モジュールインストールの方法として「標準的」な方法は上記 4 通りの方法の内左側の 3 個であるが、環境設定をしたことがない人が試しに使うには、一見手間がかかっても、最も右の方法が結局は容易で確実であるように思われる。

2.2.1 方法 1: ディレクトリ scripts 下のプログラムに手動でパスを通す場合

”手作業”で本モジュールをインストールする方法: (全行程の概要)

1. metacpan などのサイトから **ダウンロード**。(Download と書かれた項目をブラウザ上で探してクリックする。)
2. **解凍する**。ファイル名 *.tar.gz を対象に tar xzvf または tar -xzvf を実行することで解凍は可能。
3. 解凍先のディレクトリに入って (cd コマンド)、scripts 直下のファイルを確認 (ls コマンド) :
   - これからコマンドとして使うファイル名なので、後の必要に応じてすぐ把握可能とするため。
   - (必要に応じ) init.sh 以外の各ファイルのパーミッションが実行可能に設定されていることを確認。[*8]
   - 他の実行可能ファイルと名前が**衝突**することがないかどうかを後で必要に応じ確認可能とするため。[*9]
4. scripts 直下の全ファイルの新たな**格納先のディレクトリ**を決める (例: ～/bin や /home/you/bin4tsv)。
   - そのコピー先のディレクトリがまだ無い場合は、 mkdir コマンドで作成。
   - そのコピー先に既に他のファイルが存在する場合は、scripts 下のファイルと名前が衝突しないかを確認する。衝突する場合は、scripts 直下のファイル名を適宜変更する (mv -i コマンド)。
5. **パス PATH の設定:** (格納されたディレクトリにあるファイルが今後必要なときにすぐ実行出来るようにする。)
   - ここで前記で決めた格納先のディレクトリを以下、 somewhere とする。
   - Windows の場合は、**システム環境変数** の Path の値を編集し、somewhere を書き足す。[*10]
   - Unix/Linux の OS の場合: 使っているシェルに応じて、Bash ならば ～/.bash_profile に、Zsh ならば ～/.zshrc に、次の行を (ログイン時に毎回実行されるように) 書き加える : [*11]

     ```
     source somewhere/init.sh   # ← 直接実行ではなく、あるファイルに挿入。
     # init.sh はそれを格納するディレクトリを PATH に追加する働きがある。
     ```
   - 一時的に PATH を設定する場合: Bash または Zsh の場合、次を実行。[*12]: `PATH=somewhere:$PATH`
     もしくは、init.sh を source コマンドにより実行する (source init.sh など)。

格納先のディレクトリの決め方に関して

- ～/bin4tsv と決めてしまう方法がある。(～ はホームディレクトリを意味する。)
- `echo $PATH | tr : "\n"` で検討する方法もある。自分で以前独自に作ったディレクトリがあればそこに追加することもあり。
- `man hier` のコマンドを実行して、Unix の**ファイルシステムレイアウト**を知っておくと、上記のコマンドの実行結果で現れたディレクトリ名の意味が分かるであろう。なお、Build の方法で root 権限でインストールすると、本モジュールの提供するプログラムファイルは /usr/local/bin に配置される。

[*3] ls -l コマンドの出力において、空白区切りの 1 列目で r,w,x のなどの文字の配列を確認する。r は読み取り可能、w は書込可能、x は実行可能を意味する。rwxrwxrwx や r-xr-xr-x のような文字列において、これら 9 文字は 3 文字ごとに左から、ユーザーの権限、グループの権限、その他の人の権限を意味する。この 9 文字の前後に、ディレクトリであるか否か、リンクファイルであるか否かを表す文字が付加される場合がある。

[*4] which -a プログラム名 で検証可能。後の作業で既存ファイルへの上書き回避と、パス上の実行優先順位の検討が可能。

[*5] セミコロン (;) で somewhere を後ろに追加する。システム環境変数の設定ができるようにするには、Windows 7 の場合は、「**システムのプロパティ**」を起動して「**詳細設定**」のメニューを選び「**環境変数**」のボタンを押す。(なお、システム環境変数の設定の方法は、インターネットで検索すれば容易に発見できる。)

[*6] 使っているシェルを確かめる場合は `echo $0` (ドル・ゼロ) を実行する。

[*7] = の前後に空白文字を含めてはいけない。

ある環境で$PATH に記載されたディレクトリが持つファイルの個数

451	/usr/local/bin
976	/usr/bin
36	/bin
246	/usr/sbin
61	/sbin
424	/Library/TeX/texbin
126	/opt/X11/bin

表 2　for i in ‘echo $PATH | tr ”:” ”\n” | tail -7‘; do echo ‘ls $i | wc -l; echo $i ‘ ; done | column -t | awk ’{print $1 ”\t” $2 } ’ | latextable -3 1

- なお、ある環境にてパスで指定されたディレクトリの中に含まれているファイルの個数の例は表 2 のようになった。
- プログラムの**優先順位**の問題を考えるには、下記を実行。

  解凍したファイルの scripts ディレクトリ下で `which -a $( ls )` または `which -a ‘ls‘` このコマンドは、次の上書きを防ぐ問題の簡易な検査になっている。
- 既存のディレクトリに書き込む場合に**上書き**を防ぐためには、 下記のコマンドを解凍したファイル直下でを実行して、scripts 下にあるファイルと、パスが既に通っているディレクトリにあるファイルと名前が一致しないか、確認する。

  `grep -f <( ls scripts ) <(find -L $(echo $PATH | tr : ”\n” ) )` または
  `grep -f <( ls scripts | perl -pe ’chomp;$_=”^$_\$\n”’ ) <(find -L $(echo $PATH | tr : ”\n” ) )`

PATH 上の優先順位に関して

- 本モジュール CLI::LaTeX::Table をインストールする目的は、これが提供するコマンドが動くようにすることであるから、そのコマンド (プログラムファイル) を格納したディレクトリ somewhere は既存のパスに記載されたディレクトリよりも優先順位を低くする理由は無い。従って、$PATH=$PATH:somewhere とするより$PATH=somewhere:$PATH とするのが妥当のように思われるので、init.sh ではそのようにしてある。
- しかし、$PATH の中の順序がきちんと管理されているように見えることは、管理上重要なことでもありうるので、実際の動作上に差し支えなければ、$PATH=$PATH:somewhere と書き換えても良い。

2.2.2　方法 2: Module::Starter の Build でインストールする場合

1. metacpan などのサイトから CLI::LaTeX::Table を **ダウンロード**。(Download と書かれた項目をブラウザ上で探してクリック)。
2. **解凍する**。ファイル名 *.tar.gz を対象に tar xzvf または tar -xzvf を実行することで解凍は可能。
3. 解凍先のディレクトリに入って (cd コマンドを使う)、下記を順に実行。

   (1) `perl Build.PL`　# `Module::Starter` が未インストールの場合はエラーとなる。

(2) `./Build`
(3) `./Build test`
(4) `./Build install`

2.2.3 方法 3: cpanm でインストールする場合

`cpanm CLI::LaTeX::Table` で完了する。cpanm の設定によってはローカル (~/perl5/bin/など) にインストールされる。もしくは root の権限を使って `cpanm -S CLI::LaTeX::Table` もしくは `sudo cpanm CLI::LaTeX::Table` でインストール。`Module::Starter` のインストールが必要である場合は、それをあらかじめインストール。

2.2.4 方法 4: cpan でインストールする場合

`cpan CLI::LaTeX::Table` で完了する。もしくは `sudo cpan CLI::LaTeX::Table` で完了する。`Module::Starter` のインストールが必要である場合は、それをあらかじめインストール。

2.3 標準的なインストーラーである cpan または cpanm をインストールする方法

利用している環境 (特に OS) により方法は異なるので、自分で調べることが望まれる。AWS(Amazon Web Service) の EC2 で Linux を使っている場合は、`yum install cpan` または `sum install perl-CPAN` をまず実行する。root 権限が必要な場合は、このコマンドの先頭に `sudo` を付加する。そして cpan コマンドの実行をして、適宜質問に答えながら進める。Mac の場合は brew がインストールされていれば `brew install cpanminus` で cpanm コマンドが使えるようにする。もしも、cpan がインストールされていて使える状態の場合に、cpanm をインストールする場合は `cpan App::cpanminus` を実行する。

2.4 提供プログラムの 1 つが実行中に依存する別モジュールのインストール

本モジュール CLI::LaTeX::Table は `csv2tsv` というプログラムがあり、このプログラムはコアモジュールではない CPAN モジュール Text::CSV_XS に依存する。これがインストールされていない場合は、cpanm または cpan でインストールする。(Text::CSV_XS は既に広く使われているので、この方法で問題が起きることは考えにくい。) もしくはそのモジュールの他のインストール方法が良ければ、その方法を調べてインストールする。

2.5 CLI::LaTeX::Table のアンインストールの方法

cpan, cpanm もしくは Build の方法を使った場合には、それぞれの方法のアンインストールの方法が利用できる。(本モジュールの作成者は現状、このやり方についてここでは触れない。) 手作業でプログラムにパスを通す方法でインストールをした場合は、上記の作業で行ったことを元に戻す。

3 インストール上の補足事項

3.1 注意点となりうること (Perl に多少の経験がある人が気にするようなこと)

- **本モジュールが通常の CPAN にあるモジュールと異なる点:** 本モジュールは、単数または複数の Perl

言語で書かれたプログラムファイルが単独で働くように作られている。各プログラムの中に、オンラインマニュアルヘルプも含まれているが、それらの文書は必ずしも POD 形式に従って書かれている訳ではない。通常のモジュールと異なり、ライブラリとなることを意図して作られていないので、本モジュールに含まれる *.pm ファイルを必要としているプログラムファイルは存在しない。

- **当該モジュールが別のモジュールを必要とするか:**
  - **本モジュールのインストール時** — Build の方法を使う場合は、Module::Starter を必要とするかも知れない。(本モジュールをダウンロードした場合に含まれていると考えられるが、場合による。) cpan がインストールされている状態で新たに cpanm をインストールする場合は、様々な方法があるが、cpan App::cpanminus を実行する方法もある。(この場合はつまり App::cpanminus に依存する。)
  - **プログラムの実行時** — 各プログラムファイルは、内部では use 文によりいくつかのモジュールに依存している。できるだけコアモジュールに依存するので、CPAN から新たなインストールを必要としないようにしてある。ただし List::MoreUtils と Text::CSV_XS と PerlIO::gzip などを必要により使うかもしれない。[*13]
- **利用者の計算機環境に影響を与えないか:**
  1. **インストール時に既存の実行可能なプログラムとのファイル名の衝突** — 本モジュールのプログラムファイルの名前と別のプログラムのファイルが衝突[*14]した場合に、実行時にどちらが実行されるかの問題、および、インストール時に上書きされる問題がある。cpan または cpanm の設定などに依存するが、/usr/local/bin/ に同名のファイルがあっても上書きがされないように、インストールするディレクトリを別の場所 (ローカルディレクトリなど) に変更することはできる。[*15] 本モジュールのインストール方法に記載した cpan, cpanm, Module::Starter を使わない方法でインストールする場合には、パスの設定 (環境変数$PATH への書き込み) が通常は (特にその計算機に再びログインしてから本モジュールをすぐ使える状態にする場合は) 必要であるが、その場合でもそのパスに記載された複数のディレクトリの優先順位を検討する必要が発生しうる。[*16]
  2. **実行時の汚染チェック:** 一般に、Perl インタプリタの起動時に -T または -t のスイッチを与えることで汚染チェックが可能であるが、本モジュールにおいてそのような設定はまだ一部しか与えていない。
  3. **本モジュールにテストモジュールは付属しているか:** 未実装である。[*17]　実行結果が正しいかどうかの検証には、下記の方法により確かめることが望ましい。
     (a) 単純な例を入力して、動作を理解し、意図した通りに動作するか確認すること。
     (b) 別のプログラムをうまく利用して異なる方法で結果をチェックをすること。

[*13] **任意のモジュールがどの Perl のバージョンからコアモジュールとされているか知る方法:** たとえば List::Util が Perl 5.7 からコアモジュールとなっていることを知るには、次を実行する。

```
perl -MModule::CoreList -e 'print Module::CoreList->first_release(q[List::Util]);'
```

もしくは端末上で `corelist List::Util` を実行。

[*14] ファイル名の衝突とは「一致」と同じ意味であるが、この一致は望ましいものではないので、あえて衝突と表記する。なお、ファイル名とは、パス名と混同されがちだがここでは異なる。パス名とは、ディレクトリ名が階層的にファイル名の前に/を区切り文字にして連結した文字列である。

[*15] ただし、その場合は cpan, cpanm, Module::Starter についての単なる利用者であること以上の知識が必要とされる。

[*16] echo $PATH | tr : "\n" や which -a コマンド名 のようなコマンドを使って検討することになるであろう。

[*17] 通常 Perl のテストはライブラリ (*.pm 形式のファイル) に対して行う。Test::More などのモジュールはそのように設計されているように思われる (異なるかも知れないが、少なくともチュートリアルではそのようだ)。Perl のスクリプトに対して、汎用性の高いテストの方法を現在検討中である。

(c) Perl で書かれている本モジュールのプログラムファイルを閲覧して、意図しない動作が起こりにくいことを確認すること。(閲覧して、プログラムが十分にシンプルに書かれていることに疑問がある場合は、連絡して欲しい。たとえば、同じような動作を 2 箇所以上で実装していたりすると、バグの原因になりやすい。)

3.2 本モジュール CLI::LaTeX::Table の仕組み

- **どのバージョンの Perl に依存するか:**
  現状 5.14 以前に合わせている。(2018 年 4 月現在の Perl の最新バージョンは 5.26 である。) 世界のさまざまな機関で利用可能となるよう、2011 年以降の Perl であれば十分かもしれないという理由がある。技術的な要請として、乱数シードを設定する srand の動作が 5.14 以降で変更となったためでもある。[*18]それでも、出来るだけ古い計算機環境でも利用可能とすべく、5.1 でも動くように、case, say, state などの利用は出来るだけ避けるか避けようとしている。同様の理由で splice でのリファレンスの利用もしていない。

- **本モジュールに含まれるプログラムが どのモジュールをよくインポートしているか:**
  - `strict`
  - `warnings` → Perl 5.6 からコアモジュール。プラグマである。
  - **Getopt::Std** → オプションスイッチのオプションとそのパラメータを取得するため、および、--help でオンラインマニュアルの出力をするため。Perl 5 の最初からコアモジュールである。
  - `FindBin` → 実行中のプログラムの名前を $Script で参照するため。Perl 5.3.7 からコアモジュール。
  - **Term::ANSIColor** → 出力に ANSI シーケンスカラーで色を付けるため。Perl 5.6 からコアモジュール。
  - `List::Util` → min や max 、sum 、sum0 関数を使う為。Perl 5.7.3 からコアモジュール。

- **本モジュールのインストールに必要なモジュール:**
  `Module::Starter` の CPAN モジュールの事前インストールが、CLI::LaTeX::Table を cpan, cpanm もしくは Build の方法でインストールする際に、必要である。

第 II 部
CLI::LaTeX::Table を使い始める

4 各コマンドを使い始める

提供されている全てのプログラムについて、第一行目は #!/usr/bin/perl で始まる文字列で書かれている。これは、シェバングもしくはシバングと呼ばれ、たとえば `latextable` というコマンドについて

[*18] perldoc -f srand で参照が可能。

perl latextable のように perl を付加して実行しなくても、単に latextable と書くだけで同じ動作をする。ただし、下記の条件を必要とする。

- そのプログラムが実行可能であること。
  (そうでないならパーミッションを変更。chmod +x filename を実行。)
- そのプログラムを格納しているディレクトリにパスが通っていること。
- 実行可能な perl が /usr/bin/perl にあること。
  (他の場所にあるなら、シェバングをその場所に書き換える必要がある。)
- Unix/Linux などの OS であること。シェバングを認識しない OS であれば、コマンドを実行したい場合は perl filename と律儀に書く必要がある。

4.1 saikoro 一様乱数

saikoro は一様乱数を出力する。コマンドラインで下記を順次実行する。

saikoro → 乱数が表示される。サイコロのように 1 から 6 までの整数が出力される。
saikoro > /dev/null → 標準エラー出力のみが表示される。
saikoro -g 5x3 → 5 行 3 列の乱数が表示される。
saikoro -g 10x5 -y 91..100 → 91 から 100 の範囲の整数の一様乱数が、10 行 5 列の乱数で出力。
saikoro -s 123 → 乱数シードの設定。再現性が確保される (繰り返し実行しても同じ結果になる)。
saikoro -. 3 → 小数点以下 3 桁まで表示する。連続一様乱数になる。
saikoro --help → ヘルプマニュアルの表示
saikoro --help opt → ヘルプの内、オプションスイッチのみを表示。
saikoro --help en → 日本語では無く、英語のヘルプを表示。

saikoro -/ , -g 5x5 → 横方向の区切り文字の変更。CSV 形式として使える。
saikoro -g12x12 -y 1.5e5 → 科学的表記法 (1.5e5) も使える。
saikoro -g12x12 -y 1.5e5 | less -x7 → タブ間隔は less で表示中も -x N enter で変更可能。
tabs -6 → タブの表示間隔を 6 文字に変更する。元に戻す場合は、単に tabs を実行。

4.2 transpose 転置行列

transpose は標準入力から TSV 形式 (タブ文字区切り形式) で行列を読取り、その転置行列 (縦方向と横方向を逆転した行列) を標準出力に出力する。入力は数値で無くても良い。

transpose <(saikoro -s123) → <(..) はプロセス置換である。
saikoro -s123 -g3x5 | transpose → saikoro -s123 -g3x5 と比較せよ。
saikoro -s56 -g 7x8 | transpose
saikoro -/, -g3x5 | transpose -/ ";" → トリッキーな使い方なので、汎用性はあまり無い。

4.3 csel 列選択

csel は簡単な列処理を AWK 言語や cut コマンドよりも簡単に実行出来るように考えて作られた。標準出力に出力する。入力は数値で無くても良い。

```
saikoro -q -y23 -g4x5 -s67 > tmp01  ← 一時的なファイル tmp01 にデータを書き込む。
csel -p4,2 tmp01 ← 4 列目と 2 列目のみを出力。上記のと比較せよ。
csel -p4,2 < tmp01 ←リダイレクション ( < )
< tmp01 csel -p4,2  ← 似た様なコマンドの末尾のみを変更したい場合、便利。
csel -d 2,4 tmp01  ← 2 列目と 4 列目の出力を抑制。
csel -d 2..4 tmp01  ← 2 列目から 4 列目までの出力を抑制。
csel -d -1 tmp01 ← 最右列の出力を抑制
csel -h -1 tmp01 ← 最右列を最も左 (先頭 head) に出力。
csel -t -2,3 tmp01 ← 2,3 列目を最も右 (末尾 tail) に移動。
csel -t -2,3 tmp01 | csel -~ -t 2,3 ← 列の並びを元に戻す。何か処理を挟む場合に便利。
```

latextable コマンドによりこのような表が容易に作成出来る。罫線の有無は選択可能。
Excel でも Google のワークシートでもコピペして LaTeX 用の表が作成出来る。

2	4	9	1	1	3	6	8	6	1	7	3	9	1	8	1	8	3	9	1	9	5	2	8	3	8	7	1	7	8
7	9	0	9	6	9	2	5	8	6	8	6	1	5	9	4	1	9	3	1	6	3	5	2	1	6	2	3	1	6
9	5	2	4	3	3	9	4	7	9	3	6	8	6	3	0	1	5	2	5	0	3	9	9	1	5	5	8	2	6
2	4	1	3	7	2	7	5	8	3	2	1	2	9	8	0	2	0	0	5	3	5	6	2	9	8	7	3	5	0
8	7	4	9	4	1	4	9	4	7	9	0	9	1	7	7	7	2	5	1	0	9	1	9	4	5	9	4	0	2
6	3	2	2	7	2	9	8	1	0	2	9	7	3	6	9	7	7	8	8	5	9	6	5	9	9	2	7	4	0
2	0	8	4	8	9	7	3	4	0	1	8	3	0	2	8	9	1	3	3	3	7	3	5	5	9	7	8	2	2
5	0	7	6	8	5	6	0	5	0	6	7	8	1	1	1	7	4	3	9	6	4	6	9	7	4	9	7	4	2
6	4	6	2	9	2	6	5	7	0	5	7	5	4	4	8	0	8	1	7	7	2	6	9	1	6	3	7	9	4
5	8	3	9	1	3	4	1	3	0	7	7	4	2	6	4	1	0	8	8	6	1	8	9	9	0	6	8	4	7
5	1	3	7	8	7	8	7	7	3	2	1	5	9	0	9	7	2	7	7	5	4	6	0	0	2	5	9	5	5
4	8	2	5	2	8	4	6	6	9	9	2	4	9	0	8	8	2	0	3	5	7	6	7	0	6	0	7	3	9
4	7	3	8	1	5	2	4	9	5	8	6	9	8	2	5	5	2	9	6	8	3	0	6	8	1	8	0	6	0
0	5	8	8	3	9	1	4	4	2	9	8	1	9	3	5	1	9	6	4	9	7	9	5	1	3	8	7	2	0
7	5	8	1	7	1	7	4	8	2	7	1	4	5	3	4	4	2	2	1	5	4	2	6	7	3	1	9	9	1
7	1	0	2	0	5	8	0	8	3	4	8	8	9	9	6	3	8	8	1	6	8	4	3	7	4	0	1	0	8
3	1	4	5	6	1	8	4	4	3	0	1	2	7	2	6	5	4	3	2	0	0	0	8	3	7	6	9	2	3
9	6	5	8	7	2	2	0	5	2	7	8	5	2	6	3	4	2	7	0	1	7	8	9	4	5	9	9	4	2
2	0	7	6	2	6	3	9	8	5	1	8	2	5	4	1	3	2	4	9	6	4	3	4	5	7	6	4	6	1
2	2	6	5	7	1	6	4	7	1	8	5	9	5	1	9	7	2	2	9	8	9	7	7	0	8	6	5	4	8
5	9	8	9	9	4	6	9	0	7	8	6	3	8	0	2	0	8	2	6	0	6	1	1	7	0	4	4	3	9
5	1	3	7	5	0	3	7	8	2	5	7	8	0	2	8	9	1	0	0	2	7	3	3	8	8	1	8	0	8
4	4	5	4	6	5	6	5	3	2	8	0	6	9	6	6	3	1	5	4	0	3	2	5	5	3	4	4	4	8
7	0	0	7	2	3	9	0	5	5	8	1	1	8	8	0	2	2	3	5	1	2	6	2	9	0	0	1	2	8
0	1	5	0	1	7	1	2	7	1	2	7	5	9	7	9	1	6	9	5	3	0	6	4	7	6	0	7	6	6
7	1	2	9	7	6	8	8	6	2	4	0	8	9	8	4	6	1	1	9	7	5	4	5	3	5	2	3	1	5
7	1	6	1	1	0	0	5	4	1	5	2	0	9	1	0	9	3	4	0	2	6	3	6	2	1	5	9	4	3
3	3	1	7	4	0	9	6	0	7	3	0	0	4	7	9	0	8	6	3	5	6	8	1	4	3	0	5	8	1
8	5	4	4	4	9	1	2	3	8	0	5	1	6	4	9	6	7	6	8	9	4	3	7	6	9	0	4	6	6
1	0	8	9	4	1	1	0	6	3	6	5	6	0	3	6	7	2	1	7	9	5	8	7	4	1	7	0	1	4

表 3 saikoro -s123 -g 30x30 -y0..9 | latextable -m0.6 -\# . -1 -=

4.4 latextable LaTeX 用製表機

latextable は、エクセルや各種 SQL ソフトの出力結果から LaTeX の表 (table 環境) への使い易く高機能な変換器として作られた。様々な記号を含む文字列を LaTeX に使えるようにすぐ変換する目的に使うこともできる。本モジュール CLI::LaTeX::Table の中心となる機能である。

```
saikoro -g12x12 | latextable ← LaTeX を知っているなら何が起きたか分かるはず。
latextable ← この 1 つの単語を端末で入力して (Enter を押下する)。
             そしてキーボードで何かを入力し最後に Ctrl+D を押下。
             すると \begin{table} と \end{table} で囲まれた文字列が生成される。
             要領がつかめたら、LaTeX の .tex 形式のファイルに記入して動作を確かめる。
             エクセルや各種表計算ソフトからコピーをしてペーストをして、いろいろ試すと良い。
latextable --help opt ← どのようなスイッチオプションが使えるか確かめる。
latextable -s ← 入力の余計な空白を取り去る。(BigQuery をブラウザ経由で使う場合に便利。)
latextable -_ ← 半角と全角の空白、全角ハイフンの存在を分かり安く表示するようにする。
latextable -3 コンマ区切り 1 始まりの列番号 ← 数値を 3 桁毎に区切って右寄せする。
latextable -j ← 日本語の半角カナを半角の幅で表示するようにする。
latextable -m0.7 -x40mm ← 大きな表を 0.7 倍の大きさにし、センタリングをする時に使う。
latextable -C1 ← 任意の (UTF8 の) 文字列を、LaTeX 用に表とは関係無く簡潔に出力。
latextable -Cmn ← table 環境の caption で複数行にまたがる SQL 文を挿入する時便利。
```

4.5 csv2tsv CSV 形式→ TSV 形式

csv2tsv は CSV 形式 (RFC4180) を TSV 形式に変換する。このプログラムは コアモジュールでは無い CPAN モジュール Text::CSV_XS に依存している。(似た様な実装は容易に他の人も思いつくので、プログラム名が衝突する可能性は、本モジュールの他の 4 個のプログラムに比べ、最も高いと考えられる。)

```
csv2tsv foo.csv > foo.tsv  ← CSV 形式の foo.csv を読取り TSV 形式に変換。
csv2tsv  もしくは cat | csv2tsv  ← 手で CSV 形式で入力する。
csv2tsv -t '(TAB)'  ← CSV 形式のあるレコードがタブ文字を含む場合、何に変換するか指定。
csv2tsv -n '(ENTER)'  ← C あるレコードが改行文字を含む場合、何に変換するか指定。
csv2tsv -2  ← 出力の区切りが 2 行の改行文字になる。レコード中のの改行の様子を簡易に調べる。
csv2tsv -~  ← 逆変換、つまり、TSV 形式のデータを CSV 形式に変換する。
csv2tsv --help  ← ヘルプ
csv2tsv --help opt ← オプションのヘルプのみ見る。opt は opti, optio, option, options でも良い。
```

5 提供コマンドの実行 (latextable)

下記で latextable を実行した例を示す。

```
latextable -s   # -s はデータの余分な空白セルを取り除くために（やむなく）付加した
      ↑↑この 1 行がコマンドラインに入力したコマンド↑↑  ↓↓この下はエクセルなどの表からのコピペ。↓↓
75 2007-02-22 12:30:17.000 UTC 34;Allen Morgan said tha 278
76 2007-02-22 13:13:39.000 UTC rhaps you have heard of 454
77 2007-02-22 13:28:06.000 UTC tp://www.miketaber.net/a 461
78 2007-02-22 14:48:18.000 UTC 's too bad. Javascript-i 257
79 2007-02-22 16:52:25.000 UTC y omarish - i'm in on th 493
80 2007-02-22 17:19:04.000 UTC ry good idea. We can act 501
81 2007-02-22 17:29:42.000 UTC is is a cool service--I'504
82 2007-02-22 19:37:53.000 UTC cellent point; YC might 452
83 2007-02-22 21:07:47.000 UTC at a strange mix. About 530
84 2007-02-22 21:20:44.000 UTC won't clutter it up. Lo 388
85 2007-02-22 21:22:59.000 UTC that case, I _am_ excit 540
86 2007-02-22 21:49:17.000 UTC e simplest repro case is 531
87 2007-02-22 21:50:03.000 UTC r the question:<p>Why wo 531
      ↑↑ データの入力はここで終わり。↑↑  ↓↓下記は出力。これをさらにコピペして LaTeX で使う。↓↓
13 lines are read and accepted.
\begin{table}
    \center
    %\hspace*{-0mm}
    %\rotatebox{0}{
    %\scalebox{1.}{
        \begin{tabular}{ | llll | }
            \hline
            75 & 2007-02-22 12:30:17.000 UTC & 34;Allen Morgan said tha & 278\\
            76 & 2007-02-22 13:13:39.000 UTC & rhaps you have heard of & 454\\
            77 & 2007-02-22 13:28:06.000 UTC & tp://www.miketaber.net/a & 461\\
            78 & 2007-02-22 14:48:18.000 UTC & 's too bad. Javascript-i & 257\\
            79 & 2007-02-22 16:52:25.000 UTC & y omarish - i'm in on th & 493\\
            80 & 2007-02-22 17:19:04.000 UTC & ry good idea. We can act & 501\\
            81 & 2007-02-22 17:29:42.000 UTC & is is a cool service-{}-I' & 504\\
            82 & 2007-02-22 19:37:53.000 UTC & cellent point; YC might & 452\\
            83 & 2007-02-22 21:07:47.000 UTC & at a strange mix. About & 530\\
            84 & 2007-02-22 21:20:44.000 UTC & won't clutter it up. Lo & 388\\
            85 & 2007-02-22 21:22:59.000 UTC & that case, I \_am\_ excit & 540\\
            86 & 2007-02-22 21:49:17.000 UTC & e simplest repro case is & 531\\
            87 & 2007-02-22 21:50:03.000 UTC & r the question:$<$p$>$Why wo & 531\\  \hline
        \end{tabular}
    %}% closing scalebox
    %}% closing rotatebox
    %\hspace*{-0mm}
    \caption{}
    \label{tbl:547441}
\end{table}
```

上記の\begin{table}から\end{table}までを LaTeX を使う為に *.tex ファイルに記入してコンパイルすると、表 4 が現れる。手作業でいろいろな編集をすることと比較すると、非常に楽である。

75	2007-02-22 12:30:17.000 UTC	34;Allen Morgan said tha	278
76	2007-02-22 13:13:39.000 UTC	rhaps you have heard of	454
77	2007-02-22 13:28:06.000 UTC	tp://www.miketaber.net/a	461
78	2007-02-22 14:48:18.000 UTC	's too bad. Javascript-i	257
79	2007-02-22 16:52:25.000 UTC	y omarish - i'm in on th	493
80	2007-02-22 17:19:04.000 UTC	ry good idea. We can act	501
81	2007-02-22 17:29:42.000 UTC	is is a cool service--I'	504
82	2007-02-22 19:37:53.000 UTC	cellent point; YC might	452
83	2007-02-22 21:07:47.000 UTC	at a strange mix. About	530
84	2007-02-22 21:20:44.000 UTC	won't clutter it up. Lo	388
85	2007-02-22 21:22:59.000 UTC	that case, I _am_ excit	540
86	2007-02-22 21:49:17.000 UTC	e simplest repro case is	531
87	2007-02-22 21:50:03.000 UTC	r the question:<p>Why wo	531

表 4　Google BigQuery で次のコマンドを実行した結果を `latextable -s` のコピペで入力した結果を LaTeX のファイルにコピペした結果が上の表である。英数字以外の記号が含まれていて、正しく LaTeX のコマンドにする手間が掛かる場合でも、すぐに `latextable` に変換出来る。次のコマンドも `latextable` で作成している。→ **with T as (select time_ts , substr(text ,3,24) text , parent, id from 'bigquery-public-data.hacker_news.comments' where text like '% € %' or not regexp_contains(substr(text,1,30), '^[0-9a-zA-Z ,?) !\']*$')) select * except (id) from T where length (text) < 40 and length(text) >8 and text not like '%.' order by id**

6　不具合 (バグ等) を発見した場合/機能を追加したい場合

全てのソフトウェアで言われるように、バグの無いプログラムは無い。現状においても、本モジュールの作成者は、バグを見つけているし、それを仕様とごまかすつもりはない。本モジュール CLI::LaTeX::Table についてのバグについては以下の傾向があるであろう。

1. 各プログラムファイルについて、最近、もしくは、最後に追加した機能は、不具合は発生しよいであろう。十分なテストをしていないためである。
2. 各プログラムファイルについて、最初の方に設計され実装がされた機能は、何度も使われているので、バグはあまりないだろう。しかし、プログラムに編集を別の機能のために編集を加えたために、巻き添えでバグを誘発することは、ありそうである。
3. 各プログラムにスイッチ `--help` を追加して表示されるオンラインヘルプマニュアルは、書き漏らしや書き間違いがあるかもしれない。
4. 別のモジュール、特に本モジュールを作成した私が作った別のモジュールをインストールすると、プログラムファイルを上書きをすると思われるので、古いプログラムに戻してしまうかもしれない。
5. CLI::LaTeX::Table が提供するコマンドに不具合があった場合、それはほぼ全てそのコマンドの機能を実現している 1 つのプログラムファイルが引き起こしている。`saikoro` のバグは `saikoro` というファイル内に存在し、`latextable` のバグは `latextable` というファイル内に存在し、それ以外の場所のバ

グで発生することは極めて可能性が低い[*19]。

もしも利用者が、そのような不具合を修正したい場合は、Perl 言語で書かれた、プログラムファイルを修正する必要がある。(幸いなことに) 現状 CLI::LaTeX::Table の各コマンドは複雑な構造をしていない (機能は 1 つのプログラム内に集約されている) ので、Perl 言語の知識があれば、特別な知識や経験が無くとも、やや容易に修正されるであろう。Perl 言語を少しだけ使ったことがある場合、もしくは未経験だがプログラミングをしたい場合は、A.2 節を参照。

なお、不具合を減らすための方策としては、下記が考えられる。ただし、最後の方は、あまり標準的なプログラミングでは使わない方法ではある。

- テスト/診断機能:
  - 自動テストのプログラムを作る。`Test::More` などを利用して。
  - 不具合を見つかりやすいように、該当するプログラム内に検証機能を追加する。
  - 提供機能の各機能に対して、ダブルチェックが可能なようにする。
- プログラムの保守性を高める:
  - プログラムの構造を、工夫して分かり安く保つ。
  - 最低限、各変数については、その役割をコメントに書く。
  - プログラム内において、(1 行で収まらないような) 同じような機能は 2 回以上書かないようにする。
  - 関数内関数を多用する。[*20] 関数が多数あると最初に見た時に訳が分かりにくいので。
  - 演算子の優先順位を利用して、括弧 () を出来るだけ使わないようにする。修正時のタイピングの数が減るし、優先順位を熟知している場合は可読性が向上する。

付録 A　コマンド/プログラミングの必要が生じた場合

インターネットの接続が遅い、プログラミングの出来る人が周囲にいない、国際協力で同僚に最低限のプログラミングを教えながらデータ分析をするような状況を想定し，下記を記す。

A.1　Unix または Linux でシェルの bash などを使う場合に

各コマンドの機能の調べ方:

- `man command` のように調べたい `command` (例、`cat` など) の前に `man` をつけて実行する。
- `help for` のように、ある種の基本的なコマンドは (この場合は `for`) は man でなくて `help` を使う。
- `info iconv | cat` または `info iconv | less` のように info を使うと、詳細な説明が読める場合があるので注意 (この場合は iconv のコマンドを例に説明した)。

  単に `info iconv` だと 習得を必要とする info のキーボード操作を必要とする (例, h キーでヘルプ, q で終了)。したがって、パイプ (`|` 下記で説明) を使った説明をここでは示した。

[*19] それでも Perl 言語または OS の制御をしているファイルが一部書き換わっていたという事象は考えられるし、Perl もソフトウェアであるから多数の人のチェックを受けていてもバグは残っている可能性はあって、それが CLI::LaTeX::Table の提供コマンドの実行結果に影響することはありうる。可能性は低いが、それが疑われる場合の最終手段は、再インストールである。

[*20] my ではなくて our を多用することになるかもしれない。

パイプとリダイレクション、プロセス置換について:

- ２つのコマンド文の間にパイプ| を挟むことで、前のコマンドの標準出力を次のコマンドの標準入力に渡すことができる。
- リダイレクションとは < または > などのことで、この２つはそれぞれ、ファイルの中身をコマンドの標準入力に渡すことと、コマンド文の標準出力をファイルに書き込んだりする。

  リダイレクションには様々な変種があるので、必要に応じて、覚えていくと良い。>>で追加書込み、2>で標準エラー出力の書込みなどがある。これらはシェルによって違う場合がある。Windows のように OS が異なる場合は、そのような繊細な機能は備わっていないので、同じような機能が必要な場合は工夫を要する。
- プロセス置換 (process substitution) は `cat <( echo 123 )` のように、<(と) に別のコマンド文を挟んで使う。そのコマンド文の出力結果が、あたかもひとつのファイルの中身に書かれたものであるかのように、振る舞う。無駄に中間ファイルを残す必要が無くなるので、便利である。

  Ctrl+C などによって発生するシグナルに対する挙動が、`echo 123 | cat` の場合は echo に cat よりも早く働くが、プロセス置換の場合はその逆のことが多い。つまり、同じ機能をしていても、シグナルに対する挙動が違うので、注意 (場合により有用に利用可能)。

TSV 形式 (タブ文字区切りの形式) のデータファイルを使う上で習熟すべきと思われるコマンド:

`less`、`wc`、`gawk`、`cut`、`head`、`tail`、`grep`、`sed`、`od`、`iconv`、`nkf`、`lv`、`file`、`diff`、`sdiff`、`sort`、`uniq`、`paste`、`column`、`fold`、`time`、`unbuffer`

A.2 Perl 言語を使いたい場合

- Perl 言語の関数のマニュアルは `perldoc -f 関数名` ですぐ参照できる。例、`perldoc -f srand`
- Perl 言語の演算子については `perldoc perlop` ですぐ参照できる。演算子の優先順位を確認するときに使うであろう。
- Perl 言語の「チートシート」(カンニングペーパーのように狭い文面で Perl のプログラミングの重要事項を記載したメモ) は `perldoc perlcheat`。
- `perl` コマンドのさまざまなオプションを知りたい場合は `perldoc perlrun`。
- その他全てのドキュメントの一覧を見たい場合は `perldoc perltoc`。
- 本を購入して、やや高レベルの Perl 言語運用スキルを身に付けたい場合は、本文書の最後に記載した参考文献を参照のこと。

A.3 本文書で現状未解決の事項

* 本文書に索引を付ける。
* `latextable` コマンドのスイッチオプションの一覧を掲載する。
* 定まってない用語を調査して統一し、用語一覧のページを作る。(例、オプション、スイッチオプション、フラグ; マニュアル、オンラインマニュアル、ヘルプマニュアル; 端末、コマンドライン etc.)

参考文献

[1] ”A Hacking Toolset for Big Tabular Files”, in Proceedings of 2016 IEEE International Conference on Big Data, pp. 2902 – 2910, T. Shimono.

[2] 続・初めての Perl, Randal L.Schwartz, biran d foy, Tom Phoenix 著 (12 章 Perl ディストリビューションを作るには, 21 章 CPAN への投稿)

[3] PERL HACKS プロが教えるテクニック& ツール 101 選, chromatic, Damian Conway, Cutris ”Ovid” Poe 著 (4 章 モジュールの操作)

[4] UNIX の 1/4 世紀, A Quarter Century of UNIX, Peter H. Salus 著, QUIPU LLC 訳

[5] LaTeX2ϵ 美文書作成入門, 改訂第 6 販, 奥村晴彦, 黒木裕介著