表記	
C-x	Ctrl キーを押しながら x キーを押す
$\langle CR \rangle$	Enter キーを押す

1 Prolog プログラムの作成と実行

1. ソースプログラム (例：`foo.pl`) の作成 (ファイル名は小文字でスタートし，ハイフンなどは使用しない．拡張子は pl)
2. Prolog の起動およびソースプログラムのロード (`foo.pl` をダブルクリック)
3. インタプリタによるソースプログラムのロード
   File → Consult
4. (特に必要はないが) 現在ロードされている述語の表示 (`?- listing(述語名).<CR>`).
5. ゴールの実行 (`?- mygoal.<CR>`)
6. Prolog の終了 (`?- halt.<CR>`)

インタプリタとコンパイラ

Prolog では一般に，プログラム開発時はインタプリタによってプログラムを実行する．完成したらコンパイルすることにより，プログラムの最適化によって高速実行が可能になる．演習は基本的にインタプリタで行うが，時間があればコンパイルも行う予定である．

デバッグについて

実行のステップごとのトレース
`?- trace,goal.<CR>`
割り込み – 実行中に停止しなくなった場合に数回実行するとよい
`C-c`
その他よく使う命令
`a` (abort) デバッグの中断
`w` (write) 省略された部分を完全に表示
`s` (skip) それ以下の実行の表示をスキップ

2 Prolog プログラムの構造と意味

プログラムの構造

プログラムは有限個のホーン節 (Horn clause) から成る

- ホーン節の形

$$\underbrace{H}_{\text{頭部 (\textbf{head})}} \text{ :- } \underbrace{B_1, \ldots, B_n.}_{\text{本体 (\textbf{body})}}$$

$H, B_1, \ldots, B_n$ をそれぞれ**ゴール (goal)** と呼ぶ.

- ホーン節の論理的意味

$$B_1 \wedge \ldots \wedge B_n \rightarrow H$$

- ホーン節の種類
  - **単位節 (unit clause)** – 事実 (fact) を表す
    記述形式　$goal$.
    例　`parent(tom,liz).`
  - **確定節 (definite clause)** – ルール (rule) を表す
    記述形式　$goal$:- $goal_1, \ldots, goal_n$.
    例　`father(N,M) :- parent(N,M), male(N).`
  - **ゴール節 (goal cluase)** – 質問 (query) を表す
    記述形式　:- $goal_1, \ldots, goal_n$.
    例　`:- parent(X,liz).`

プログラムの実行

上から下，左から右へと行われる
Prolog 処理系は大文字小文字を区別して扱う

- 変数は大文字または _ から始まる文字列
- 定数，関数記号，述語記号は小文字から始まる文字列

再帰プログラミング

ある対象を定義するのにその対象自身を使用することを再帰 (recursion) といい，そのような定義を再帰的定義という．Prolog の基本的なプログラミングスタイルであり，計算機科学における基本的かつ重要な考え方の 1 つである．

例 1　述語 ancestor(X,Y) 「X は Y の祖先である」の再帰的定義
ancestor(X,Y) ← parent(X,Y).
　X が Y の親ならば X は Y の祖先である
ancestor(X,Y) ← parent(X,Z) ∧ ancestor(Z,Y).
　X が Z の親でかつ Z が Y の祖先ならば X は Y の祖先である
例 2　述語 sum(X,Y) 「0 から X までの和は Y である」の再帰的定義
sum(0,0).
　0 から 0 までの和は 0 である
sum(N,M) ← sum(N-1,M1) ∧ M=M1+N.
　0 から N までの和 M は 0 から N-1 までの和 M1 に N を加えたものである
注意：これらの例は文法を一部変更しているのでプログラムそのものではない.

3 レポートの提出について

- アップロードするファイルはソースプログラムと動きに対する説明の 2 種類ある.
- ソースプログラム (r*_*.pl) については，動作確認しエラーがなく答えが出力されることを確かめた上でアップロードする．(ファイル名は自由).
- レポートファイル (r*_*.txt) については，(1) 指定されたプログラムの論理的意味，(2) 解答例を実際に動かした動作について説明し，(3) 自分が正しいプログラムができなかった場合，どこが間違ったか，なぜ間違ったかについての考察を書く．正しいプログラムができていた場合は,「正しくできた」と書き，もし新たな知見や疑問があればそれを書く．(特になければ「正しくできた」だけでよい．)
- プログラムごとにコメントがある場合は各プログラムの冒頭に記述してもよい．コメントは % から始まる行に記述する．複数行にまたがる場合は，/* と */ ではさんで記述してもよい．どうしても完成できなかった場合は，その由をここに書いて未完成プログラムとして送付してもよい．**この場合，どこがわからないのか必ず記述すること．**
- 提出にはレポート提出システムを使用する (プロ I, プロ III 等と同じ).
  本実習のホームページは `http://ist.ksc.kwansei.ac.jp/~kitamura/lecture/PROLOG/EonKP.html` であり，ここからレポート提出システムにリンクをはっている．ユーザ ID およびパスワードは情報科学科教務関係の URI で使用しているものである.

- **〆切は 1 週間後の月曜日 8:30AM. 〆切後は受け付けない．**
- 必ず自分で理解して書くこと (単純コピーが発覚した場合，コピー先，コピー元の両者ともその回の点を 0 点とする).
- 毎回レポート提出しかつオプション課題（* で示す）を除いてすべての問題を解いている (完全な解でなくてもある程度のチャレンジが行われている) もののみ成績評価の対象とする.
- これらの規則に反しているものは TA から再提出を求めるメールを送ることがある.
- 理解度テストに合格することが単位取得の必要条件だが，合格しても，出席状況，課題の提出状況や内容によって単位取得できない場合もあるので注意すること.
- 遅刻者は入室時に TA に報告すること．遅刻/欠席の扱いは以下のとおり.
  1. 2 回の遅刻で 1 回の欠席とみなす.
  2. 欠席を 3 回以上すれば不合格とする.
  3. 1,2 に関し正当な理由がある場合はすみやかに申し出ること.
- レポート未提出で正当な事由のあるものは速やかに担当教員に届け出ること.
- 質問先は北村 (ykitamura@kwansei.ac.jp) とする.

4 自習時の環境

- 学校での自習：4 号館 4 階のすべての演習室のコンピュータには Prolog がインストールされている.
- 家での自習：SWI-prolog は `http://www.swi-prolog.org` からフリーで入手可能.

教科書・参考文献

- 「Prolog への入門」 I.Bratko 著，安部憲広 訳，近代科学社，1990 年 (教科書).
- 「AI プログラミング」I.Bratko 著，安部憲広 訳，近代科学社，1996 年.
- 「PROLOG 入門」 後藤 滋樹 著，サイエンス社，1984 年.
- 「知識処理論」 萩野 達也 著，産業図書，1996 年.
- 「Prolog の技芸」 L.Sterling and E.Shapiro 著, 松田利夫訳，共立出版，1988 年 (絶版).
- 「論理による問題の解法 - Prolog 入門 -」 R.Kowalski 著，浦 昭二 監修， 山田 眞一 他 訳，培風館，1987 年.

参考 URL

`http://www.sakalab.org/swi-prolog/swi-prolog.html`
`http://home.soka.ac.jp/~unemi/LispProlog/`
`http://bach.istc.kobe-u.ac.jp/prolog/intro/database.html`