Dec 2005　（日本語バージョン：2010/1/20）

A Mizar demo

PC Mizar 7.5.01　Original Article: http://www.cs.ualberta.ca/~piotr/Mizar/MiniTut/

by Piotr Rudnicki　translated by l'Hospitalier for noncommercial use(2007.11.11)

Mizar の新ユーザーにはこのミニチュートリアルが有用であると思います。

フィボナッチ数列は前もって **REF_FF** というミザールのアーティクルの中で以下のように定義されています。

```
definition let n;
  func Fib (n) -> Nat means
:: PRE_FF:def 1
 ex fib being Function of NAT, [:NAT, NAT:] st
   it = (fib.n)`1 & fib.0 = [0,1] &
   for n being Nat, x being Element of [: NAT, NAT :] st x =
fib.n
             holds fib.(n+1) = [ x`2, x`1 + x`2 ];
end;
```

Mizar はファンクタの再帰的な定義を許さないので、上の定義は再帰的に定義された集合の理論的機能（set theoretic function）に依存しています（いくつかスキームが関数の再帰的定義のために使用可能です）。　そして、ある定理はファンクタ Fib の再帰的特性を作り直します。　結局、次の定理がしばしば参照されます（通常フィボナッチ数列の定義として理解されています）。

```
theorem :: PRE_FF:1
         Fib(0) = 0 &
         Fib(1) = 1 &
         (for n being Nat holds Fib((n+1)+1) = Fib(n) +
Fib(n+1));
```

n 次のフィボナッチ数が n より小さくないことを証明してみましょう。

```
environ  :: mt1
   begin
   for n being Nat holds Fib n >= n;
```

（上の記載は**.miz** ファイルに書いてあります）。　残念ながら、Mizar は空の環境ではこの文を理解できないので次のエラーが出ます。

```
   environ  :: mt1

   begin

   for n being Nat holds Fib n >= n;
::>               *151     *143,321
::>
::> 143: No implicit qualification
::> 151: Unknown mode format
::> 321: Predicate symbol or "is" expected
```

Natの下の***151** のエラーはこの文脈での**Nat**はモード（型）（mode (a type)）を意味することが期待されていて、現在の（空の）環境はその意味を提供しないことを意味します。　—自然数型はアーティクル **NAT_1** で導入されます—

この **Nat** の意味を導入するため（意味解析のために）**NAT_1** からコンストラクタを、（構文解析のフォーマットのために）**NAT_1 notation** をインポートします。　この修正で、残る構文エラーは２個です。

```
   environ  :: mt2
    notations NAT_1;
    constructors NAT_1;
   begin

   for n being Nat holds Fib n >= n;
::>                          *143,321
::>
::> 143: No implicit qualification
::> 321: Predicate symbol or "is" expected
```

Mizar は入力されたテキストを処理し、そのファイルの中に::>で始まるエラーメッセージ行を挿入します。　これらの行はコメントとして扱われ、次回テキストが（Mizar により）処理されたときに取り除かれます。　通常、新しい別のエラーメッセージが挿入されてしまいますが・・・。　結局、経過を通じて単一のテキストファイルを修正しながら仕事をすることになります。

***143** のエラーは **Fib** が自由変数として（実際は、デフォルトで自由変数に量化されたのですが）扱われ、その種の変数型は事前に宣言されていることが必要です。　ここでは、**Fib** はフィボナッチ数列のファンクタを表すことを意図しました。　**Fib** がファンクタシンボルであることは **PRE_FF** ボキャブラリで宣言されています；これは **findvoc** ユーティリティを使うことでわかります。ボキャブラリ **PRE_FF** を環境に加えると：

```
    environ  :: mt3
     vocabularies PRE_FF;
     notations NAT_1;
     constructors NAT_1;
    begin

    for n being Nat holds Fib n >= n;
::>                         *165 *153,395
::> 153: Unknown predicate format
::> 165: Unknown functor format
::> 395: Justification expected
```

を得ます。　最初の ***165** のエラーだけに注目しましょう。　**Fib** はファンクタとして（訳注：システムに）知られていますが、構文解析のフォーマットと項（term）を構築するのに、どのように使用するかの情報は今の環境には依然として含まれていません。　**Fib** はアーティクル **PRE_FF** で定義されているので、コンストラクタとノーテーションをそこからインポートします。

```
    environ  :: mt4
     vocabularies PRE_FF;
     notations NAT_1, PRE_FF;
     constructors NAT_1, PRE_FF;
    begin

    for n being Nat holds Fib n >= n;
```

```
::>                                     *153,395
::> 153: Unknown predicate format
::> 395: Justification expected
```

さて**Fib n**という項はMizarに理解されたようですが、別の２つのエラーが残りました。　もう一度、最初の***153** エラーだけに注目しましょう、二番目のエラーは（訳注：***153** エラーの）影響によるものです。　このエラーは、**>=** は述語論理のシンボルとして認知されているのですが（それはボキャブラリ**HIDDEN**で導入されています）、その使用法の実際のフォーマットは現行の環境を通じてはアクセスできないと言っています。　使用したいgreater than あるいはequal という関係はアーティクル **XREAL_0** で定義されています。　それで**XREAL_0** からノーテーションとコンストラクタをインポートせねばなりません。　このインポートを成功させるのには、このページまででは、まだ議論してないクラスタを使ったいくつかの追加をインポートする必要があります。

```
     environ  :: mt5
      vocabularies PRE_FF;
      notations XREAL_0, NAT_1, PRE_FF;
      constructors ARYTM_2, XREAL_0, NAT_1, PRE_FF;
      registrations XREAL_0, ARYTM_3;
     begin

     for n being Nat holds Fib n >= n;
::>                                        *4
::> 4: This inference is not accepted
```

チェッカーはエラーを１つ残しています、やっと今の時点で証明に集中できる、正しい考え方の環境を構築したようです。　その証明はアーティクル **NAT_1** で定義されているスキーム **Ind** を採用するため：

```
scheme :: NAT_1:sch 1
 Ind { P[Nat] } :
 for k holds P[k]
   provided
   P[0] and
   for k st P[k] holds P[k + 1];
```

の導入 (induction) で終了する予定です。　このスキームにアクセスするため、環境で schemes 指令を使います。 スキームを適用するため仮定を定式化して:

```
    environ  :: mt6
     vocabularies PRE_FF;
     notations XREAL_0, NAT_1, PRE_FF;
     constructors ARYTM_2, XREAL_0, NAT_1, PRE_FF;
     registrations XREAL_0, ARYTM_3;
     schemes NAT_1;
    begin
    P1: Fib 0 >= 0;
::>       *103

    P2: for n being Nat
         st Fib n >= n
          holds Fib (n+1) >= n+1;
::>                  *103    *103

    for n being Nat holds Fib n >= n from NAT_1:sch 1(P1, P2);

::> 103: Unknown functor
```

を得ます。　なぜ **Fib n** が OK なのに **Fib 0** は **Fib** の「未知のファンクタ」と示されるのでしょう？

0 や **1** のような小さな定数はプロセッサに組み込まれていますが、デフォルトでは集合（set）と型付けされています。　これらを自然数として使うために環境で **requirements** 指令を使用せねばなりません。　われわれの場合、**ARYTM** と **SUBSET** という名前の **requirements** が必要です。（**requirements** 指令はデフォルトである種のオブジェクトの処理を可能にします；現時点ではこれについてのこれ以上のコメントはしません）

```
    environ  :: mt7
     vocabularies PRE_FF;
     notations XREAL_0, NAT_1, PRE_FF;
     constructors ARYTM_2, XREAL_0, NAT_1, PRE_FF;
     registrations XREAL_0, ARYTM_3;
     requirements NUMERALS, SUBSET;
```

```
     schemes NAT_1;
    begin

    P1: Fib 0 >= 0;
::>               *4

    P2: for n being Nat
        st Fib n >= n
         holds Fib (n+1) >= n+1;
::>                         *4

    for n being Nat holds Fib n >= n from NAT_1:sch 1(P1, P2);
::>                                *23
::> 4: This inference is not accepted
::> 23: Invalid order of arguments in the instantiated predicate
```

P1 と **P2** とラベルされた命題（proposition）は、Mizar チェッカーに理解されてはいますが、チェッカーは、なぜジャスティフィケーション無しで真になるのかを了解しないので、受け入れられていないことに注意してください。

エラー***23** は導入のパラメータ： **defpred** を使用して明示的に述べられねばならない公式についての述語（formal predicate）に関連したものです。

```
    environ  :: mt7a
     vocabularies PRE_FF;
     notations XREAL_0, NAT_1, PRE_FF;
     constructors ARYTM_2, XREAL_0, NAT_1, PRE_FF;
     registrations XREAL_0, ARYTM_3;
     requirements NUMERALS, SUBSET;
     schemes NAT_1;
    begin

    defpred P[Nat] means Fib $1 >= $1;

        P1: P[0];
::>         *4
```

```
    P2: for n being Nat st P[n] holds P[n+1];
::>                                       *4

    for n being Nat holds P[n] from NAT_1:sch 1(P1, P2);

::> 4: This inference is not accepted
```

論理的エラーだけになりましたが、前に述べたように、帰納ステップは証明が難しいので先へ進むことができません。

すなわち、**Fib** はその再帰的特性で先行する２つの値への参照をおこないます、一方帰納仮説は直接先行するものについてのみ、ある種の情報を提供します。この場合は、証明しようとしている命題（proposition）を強化し（induction loading）、そして、フィボナッチ数列の後に続く２つの項についての不等式の論理積（conjunction）にするという単純な救済法があります。　適切な調整の後：

```
   environ  :: mt8
     vocabularies PRE_FF;
     notations XREAL_0, NAT_1, PRE_FF;
     constructors ARYTM_2, XREAL_0, NAT_1, PRE_FF;
     registrations XREAL_0, ARYTM_3;
     requirements NUMERALS, SUBSET;
     schemes NAT_1;
    begin

    defpred P[Nat] means Fib $1 >= $1 & Fib ($1+1) >= $1+1;

    P1: P[0];
::>       *4,4
    P2: for n being Nat st P[n] holds P[n+1];
::>                                       *4
    for n being Nat holds P[n] from NAT_1:sch 1(P1, P2);

::> 4: This inference is not accepted
```

というテキストに出会います。　帰納のための最初の仮定はフィボナッチ数列についての定理 **PRE_FF:1** から簡単に理解（follow）できるでしょう。　しかし：

```
    environ  :: mt9
     vocabularies PRE_FF;
     notations XREAL_0, NAT_1, PRE_FF;
     constructors ARYTM_2, XREAL_0, NAT_1, PRE_FF;
     registrations XREAL_0, ARYTM_3;
     requirements NUMERALS, SUBSET;
     schemes NAT_1;
    begin

    defpred P[Nat] means Fib $1 >= $1 & Fib ($1+1) >= $1+1;

    P1: P[0] by PRE_FF:1;
::>                  *144,330
    P2: for n being Nat st P[n] holds P[n+1];
::>                                        *4
    for n being Nat holds P[n] from NAT_1:sch 1(P1, P2);

::> 4: This inference is not accepted
::> 144: Unknown label
::> 330: Unexpected end of an item (perhaps ";" missing)
```

となります。　ライブラリ参照を実行するときはいつも、この場合は **PRE_FF:1** に対してですが、アクセスしたい定理を含むアーティクルの名前を **thorems** というライブラリ指令に記述する必要があります。　この簡単な変更のあとは：

```
    environ  :: mt10
     vocabularies PRE_FF;
     notations XREAL_0, NAT_1, PRE_FF;
     constructors ARYTM_2, XREAL_0, NAT_1, PRE_FF;
     registrations XREAL_0, ARYTM_3;
     requirements NUMERALS, SUBSET;
     theorems PRE_FF;
     schemes NAT_1;
    begin
```

```
    defpred P[Nat] means Fib $1 >= $1 & Fib ($1+1) >= $1+1;

    P1: P[0] by PRE_FF:1;
::>            *4
    P2: for n being Nat st P[n] holds P[n+1];
::>                                            *4
    for n being Nat holds P[n] from NAT_1:sch 1(P1, P2);

::> 4: This inference is not accepted
```

となります。　最初のエラーはMizarが **0+1=1** を理解しないという事実のせいです。　Mizarは小さな自然数の定数を含むある種の算術演算表現を自動的に処理しますが依然として適切なリクワイアメンツ、この場合は**ARITHM**ですが、を特定してそうするようにとMizarに言ってやる必要があります。(インポートした特性のリストはファイル **ARITHM** を見てください)　こうして：

```
    environ  :: mt10a
     vocabularies PRE_FF;
     notations XREAL_0, NAT_1, PRE_FF;
     constructors ARYTM_2, XREAL_0, NAT_1, PRE_FF;
     registrations XREAL_0, ARYTM_3;
     requirements ARITHM, NUMERALS, SUBSET;
     theorems PRE_FF;
     schemes NAT_1;
    begin

    defpred P[Nat] means Fib $1 >= $1 & Fib ($1+1) >= $1+1;

    P1: P[0] by PRE_FF:1;
    P2: for n being Nat st P[n] holds P[n+1];
::>                                             *4
    for n being Nat holds P[n] from NAT_1:sch 1(P1, P2);

::> 4: This inference is not accepted
```

を得ます。　帰納スキームに２つ目の仮定をジャスティファイするには、その構造がすでに証明中の命題により記述してある証明を書きます。

```
    P2: for n being Nat st P[n] holds P[n+1]
        proof
          let n be Nat; assume
        IH: Fib n >= n & Fib (n+1) >= n+1;
         thus Fib (n+1) >= n+1 by IH;
         thus Fib (n+1+1) >= n+1+1;
::>                                   *4
        end;
```

まだジャスティファイされていない結論は２つの場合を考えることを要求しています：　**n** が **0** の場合と、そうでない場合です。　これに対し **per cases** という Mizar 構造を用います。

```
    P2: for n being Nat st P[n] holds P[n+1]
        proof
          let n be Nat; assume
        IH: Fib n >= n & Fib (n+1) >= n+1;
         thus Fib (n+1) >= n+1 by IH;
         per cases;
::>              *4
         suppose S1: n = 0;
          thus Fib (n+1+1) >= n+1+1;
::>                                    *4
         end;
         suppose S1: n > 0;
          thus Fib (n+1+1) >= n+1+1;
::>                                    *4
         end;
        end;
```

per cases 構造は **suppose** 条件のすべての論理積（disjunction）のジャスティフィケーションを必要とします。

```
         per cases by NAT_1:19;
         suppose S1: n = 0;
          thus Fib (n+1+1) >= n+1+1;
::>                                    *4
         end;
```

```
        suppose S1: n > 0;
         thus Fib (n+1+1) >= n+1+1;
::>                               *4
        end;
```

定理 **NAT_1:19** は：

```
theorem :: NAT_1:19
  0 <> k implies 0 < k;
```

です。　**Theorem** 指令にさらに **NAT_1** を付け加えます。

残念ながら、最初の場合（**n=0**）の結論は **Fib 2** が **1** なので偽です。　証明しようとしている定理は簡単に偽だとわかりますが、こんなに遅くなってからではちょっと困りましたね（訳注：//??embarassing-> embarrassing)。　でも、機械的証明検証システムは成功の見込みがあるのであきらめません。

このケースの単純な救済策は、証明しようとしている命題をアップグレードして：　**(n+1)**番目のフィボナッチ数は **n** より小さくないとすることです。　必要な変更をすべて行うと：

```
    environ  :: mt14
     vocabularies PRE_FF;
     notations XREAL_0, NAT_1, PRE_FF;
     constructors ARYTM_2, XREAL_0, NAT_1, PRE_FF;
     registrations XREAL_0, ARYTM_3;
     requirements REAL, ARITHM, NUMERALS, SUBSET;
     theorems PRE_FF, NAT_1;
     schemes NAT_1;
    begin

    defpred P[Nat] means Fib ($1+1) >= $1 & Fib ($1+1+1) >=
$1+1;

    P1: P[0] by PRE_FF:1;
    P2: for n being Nat st P[n] holds P[n+1]
       proof
         let n be Nat; assume
```

```
        IH: Fib (n+1) >= n & Fib (n+1+1) >= n+1;
         thus Fib (n+1+1) >= n+1 by IH;
         per cases by NAT_1:19;
         suppose S1: n = 0;
          thus Fib (n+1+1+1) >= n+1+1;
::>                                 *4
         end;
         suppose S1: n > 0;
          thus Fib (n+1+1+1) >= n+1+1;
::>                                 *4
         end;
        end;
    for n being Nat holds P[n] from NAT_1:sch 1(P1, P2);

::> 4: This inference is not accepted
```

という結果が得られます。 **P1** とラベルした宣言をアクセプトさせるためにもう１つリクワイアメントを付け加えたことに注意してください。

最初のケースの証明は **PRE_FF:1** から組み込み計算機能ですぐに可能で：

```
    suppose S1: n = 0;
      Fib (0+1+1+1) = Fib (0+1) + Fib (0+1+1) by PRE_FF:1
                 .= 1+1 by PRE_FF:1;
      hence Fib (n+1+1+1) >= n+1+1 by S1;
    end;
```

という風になるでしょう。 ２つ目のケースの証明では、まず **REF_FF:1** を使います。 それから２つの不等式の両辺を **IH** とラベルされた帰納仮定からつけ加えます。 これはつぎの定理 **XREAL_1:9**：

```
theorem :: XREAL_1:9
 a <= b & c <= d implies a+c <= b+d;
```

の使用が必要で､次に **theorem** 指令に **XREAL_1** を加えることが必要です。 まず、**n** が少なくとも **1** であることを示すことからはじめますが、それには **n+(n+1)**がすくなくとも **n+1+1** であることを示す必要があります。 これで：

```
         suppose S1: n > 0;
```

```
          A: Fib (n+1+1+1) = Fib (n+1) + Fib (n+1+1) by PRE_FF:1;
          B: Fib (n+1) + Fib (n+1+1) >= n+(n+1) by IH, XREAL_1:9;
             0+1 < n+1 by S1, XREAL_1:10; then
             n >= 1 by NAT_1:38; then
             n+(n+1) >= n+1+1 by XREAL_1:9;
           thus Fib (n+1+1+1) >= n+1+1;
::>                                      *4
          end;
```

を得ます。　参照する定理は：

```
theorem :: NAT_1:38
  k < n + 1 iff k <= n;

theorem :: XREAL_1:9
 a <= b & c <= d implies a+c <= b+d;

theorem :: XREAL_1:10
 a < b & c <= d implies a+c < b+d;
```

となります。　この時点で、greater than あるいは equal の推移性をつかえば証明を完成することができます。　順序付けの関係（ordering relation）は less than あるいは equal の同義語としてアーティクル **XREAL_0** のなかで：

```
definition let x,y be real number;
  pred x <= y means
:: XREAL_0:def 2
                  ... definiens omitted ...
  reflexivity;
  connectedness;
  synonym y >= x;
  antonym y < x;
  antonym x > y;
end;
```

と定義されています。　less than あるいは equal の推移性はアーティクル **XREAL_1** のなかに：

```
theorem :: XREAL_1:2  :: AXIOMS:22
```

```
a <= b & b <= c implies a <= c;
```

とあります。　２つ目のケースは：

```
suppose S1: n > 0;
A: Fib (n+1+1+1) = Fib (n+1) + Fib (n+1+1) by PRE_FF:1;
B: Fib (n+1) + Fib (n+1+1) >= n+(n+1) by IH, XREAL_1:9;
   0+1 < n+1 by S1, XREAL_1:10; then
   n >= 1 by NAT_1:38; then
   n+(n+1) >= n+1+1 by XREAL_1:9;
 hence Fib (n+1+1+1) >= n+1+1 by A, B, XREAL_1:2;
end;
```

という具合に完成します。　完成したテキストをここに書きます。

```
environ  :: mt18
 vocabularies PRE_FF;
 notations XREAL_0, NAT_1, PRE_FF;
 constructors ARYTM_2, XREAL_0, NAT_1, PRE_FF;
 registrations XREAL_0, ARYTM_3;
 requirements REAL, ARITHM, NUMERALS, SUBSET;
 theorems PRE_FF, NAT_1, XREAL_1;
 schemes NAT_1;
begin

defpred P[Nat] means Fib ($1+1) >= $1 & Fib ($1+1+1) >= $1+1;

P1: P[0] by PRE_FF:1;
P2: for n being Nat st P[n] holds P[n+1]
    proof
      let n be Nat; assume
    IH: Fib (n+1) >= n & Fib (n+1+1) >= n+1;
     thus Fib (n+1+1) >= n+1 by IH;
     per cases by NAT_1:19;
     suppose S1: n = 0;
       Fib (0+1+1+1) = Fib (0+1) + Fib (0+1+1) by PRE_FF:1
                    .= 1+1 by PRE_FF:1;
      hence Fib (n+1+1+1) >= n+1+1 by S1;
```

```
      end;
      suppose S1: n > 0;
      A: Fib (n+1+1+1) = Fib (n+1) + Fib (n+1+1) by PRE_FF:1;
      B: Fib (n+1) + Fib (n+1+1) >= n+(n+1) by IH, XREAL_1:9;
         0+1 < n+1 by S1, XREAL_1:10; then
         n >= 1 by NAT_1:38; then
         n+(n+1) >= n+1+1 by XREAL_1:9;
       hence Fib (n+1+1+1) >= n+1+1 by A, B, XREAL_1:2;
      end;
     end;
 for n being Nat holds P[n] from NAT_1:sch 1(P1, P2);

then for n being Nat holds Fib (n+1) >= n;
```

最後の命題から、２番目の等位項をスキップした、より簡単なステートメントを推論することができます。

```
   then for n being Nat holds Fib (n+1) >= n;
```

完全な帰納（course of values）を使った同じ定理の証明を見てください。

```
     environ  :: mt_cov
      vocabularies PRE_FF;
      notations XREAL_0, NAT_1, PRE_FF;
      constructors ARYTM_2, XREAL_0, NAT_1, PRE_FF;
      registrations XREAL_0, ARYTM_3;
      requirements REAL, ARITHM, NUMERALS, SUBSET;
      theorems PRE_FF, NAT_1, XREAL_1, CQC_THE1, XCMPLX_1;
      schemes NAT_1;
     begin

     defpred P[Nat] means Fib ($1+1) >= $1;

P: for k being Nat
     st for n being Nat st n < k holds P[n]
      holds P[k]
proof let k be Nat; assume
IH: for n being Nat st n < k holds Fib (n+1) >= n;
```

```
 per cases;
  suppose k <= 1; then k = 0 or k = 0+1 by CQC_THE1:2;
   hence Fib (k+1) >= k by PRE_FF:1;
  end;
  suppose 1 < k; then
     1+1 <= k by NAT_1:38; then
     consider m being Nat such that
  A: k = 1+1+m by NAT_1:28;
   thus Fib (k+1) >= k proof
    per cases by NAT_1:19;
    suppose S1: m = 0;
      Fib (0+1+1+1) = Fib(0+1) + Fib(0+1+1) by PRE_FF:1
                 .= 1 + 1 by PRE_FF:1;
     hence Fib (k+1) >= k by A, S1;
    end;
    suppose m > 0; then
      m+1 > 0+1 by XREAL_1:10; then
      m >= 1 by NAT_1:38; then
    B: m+(m+1) >= m+1+1 by XREAL_1:9;
      m < m+1 & m+1 < m+1+1 by XREAL_1:31; then
      m < k & m+1 < k by A, XREAL_1:2; then
    C: Fib (m+1) >= m & Fib (m+1+1) >= m+1 by IH;
      Fib (m+1+1+1) = Fib (m+1) + Fib (m+1+1) by PRE_FF:1; then
      Fib (m+1+1+1) >= m+(m+1) by C, XREAL_1:9;
    hence Fib (k+1) >= k by A, B, XREAL_1:2;
   end;
  end;
  end;
end;

for n being Nat holds P[n] from NAT_1:sch 4(P);

then for n being Nat holds Fib(n+1) >= n;
```

On Wed, Nov 14, 2007 at 09:01:26AM +0900,
Tamiya-sama:

Thank you for your interest in the mini tutorial. Please feel free to
to use this tutorial in whichever way you like.

Best regards,
Piotr Rudnicki http://web.cs.ualberta.ca/~piotr

url: http://www18.ocn.ne.jp/~tamiya/A_MizarDemo_JP.pdf