# Technologie programmas U-buigen : listing Ubuig

Laboratorium voor Omvormtechnologie
Technische Universiteit Eindhoven

Technologie programmas U–BUIGEN

Auteurs: ir. S.M. Hoogenboom
ir. A.C.E.C. Melis
ir. A.B. Perduijn
ing. D. Slagter

# LISTING UBuig

26 augustus 1989
IOP–Metalen WPA rapport 0792



In opdracht van TNO Metaalinstituut,
in het kader van IOP–Metalen Buigen van voorbeklede plaat

# Listing Programma Ubuig

(Technologieprogramma U-buigen)

```
(*****************************************************************
 *                                                               *
 *                                                               *
 *   Programma     : UBuig (Technologieprogramma U-buigen        *
 *                   deterministisch model volgens Hoogenboom)   *
 *                                                               *
 *   Overige modulen :  UBCalcul                                 *
 *                      UBVar                                    *
 *                                                               *
 *   Versie        : 1.00                                        *
 *                                                               *
 *   Datum         : 8 september 1989                            *
 *                                                               *
 *   Language      : Turbo Pascal versie 5.0 (Borland)           *
 *                                                               *
 *   Systeem       : IBM - AT/XT met EGA-kaart                   *
 *                                                               *
 *   Auteurs       : ir. S.M Hoogenboom                          *
 *                   ir. A.C.E.C. Melis                          *
 *                   ir. A.B.C. Perduijn                         *
 *                   ing. D. Slagter (Invoer en uitvoerroutines) *
 *                                                               *
 *   Informatie    : Laboratorium voor Omvormtechnologie         *
 *                   Technische Universiteit Eindhoven           *
 *                                                               *
 *   Telefoon      : 040 - 472630 (Secretariaat)                 *
 *                                                               *
 *   (Gemaakt door TUE in opdracht van TNO in het kader van het  *
 *    project IOP-metalen buigen van voorbeklede plaat)          *
 *                                                               *
 *****************************************************************)
```

```
PROGRAM UBuig;

Uses
  UBVar,       (* Globale variabelen       *)
  UBCalcul;    (* Rekenmodule voor U-buigen *)

CONST

     pi        = 3.1415927;


TYPE

     real_4         = REAL;              (* 32 bits precisie *)
     integer_2      = INTEGER;           (* 16 bits precisie *)
     confirm        = BOOLEAN;           (* waar, of niet waar *)
```

```
    l_string      = STRING[ 80  ];      (* conformant string *)
    ascii_ch      = CHAR;               (* een ascii character *)

    facts         = RECORD
                      val : real_4;     (* invoerwaarde *)
                      upp : real_4;     (* bovenlimiet *)
                      low : real_4;     (* onderlimiet *)
                      prm : l_string;   (* prompt regel *)
                      dim : l_string    (* dimensie aanduiding *)
                    END;

    validate      = ( on, off );

VAR

 (* U-buig input gegevens *)

    thick,            (* plaatdikte                          *)
    c_nad,            (* evenredigheidsconstante van Nadai   *)
    n_nad,            (* verstevigings exponent van Nadai    *)
    eps_0,            (* voordeformatie                      *)
    e_mod,            (* Young's modulus                     *)
    nu_po,            (* Poisson constante                   *)
    stroke,           (* procesweg                           *)
    rho_d,            (* matrijsradius                       *)
    rho_p,            (* stempelradius                       *)
    rho_t,            (* radius tegenhouder                  *)
    beta_t,           (* hoek tegenhouderafronding           *)
    breedte_p,        (* breedte van stempel                 *)
    zetspleet,        (* zetspleet                           *)
    friction : facts; (* Coulomb wrijvingsconstante          *)

 (* U-buig output gegevens *)

    angle,              (* hoek onbelast           *)
    force_normal,       (* normaalkracht           *)
    force_punch,        (* stempelkracht           *)
    force_tegenhouder,  (* normaalkracht           *)
    delta_vering,       (* totale terugveerhoek    *)
    qster : facts;      (* Vlak stempeldeel        *)


 (* redirection faciliteit *)

    redir  : validate;


 (* cursor positie *)

    line,             (* vertikaal    *)
    tab    : integer_2; (* horizontaal  *)
```

```
PROCEDURE set_cursor( line, tab : integer_2);
(* plaatst de cursor op de aangeven postitie *)
  BEGIN
    WRITE( output, CHR(27):1, '[', line:1, ';', tab:1, 'H' )
  END; (* set_cursor *)


{***************** leeg keyboardbuffer ********************************}

PROCEDURE leeg_keyboardbuffer;
{zie blz. 418 van 'TurboPascal Programmers Library'}
VAR
  buffer_head:integer absolute $0000:$041A;  {current head}
  buffer_tail:integer absolute $0000:$041C;  {current tail}
BEGIN
  buffer_tail:=buffer_head;
END; {of procedure leeg_keyboardbuffer}



PROCEDURE video( switch : l_string );
  (* video atribuut zetten *)
  BEGIN

    IF ( switch = 'inverse' )
      THEN WRITE( output, CHR( 27 ):1, '[7m' );

    IF ( switch = 'normal' )
      THEN WRITE( output, CHR( 27 ):1, '[0m' );

    IF ( switch = 'bold' )
      THEN WRITE( output, CHR( 27 ):1, '[1m' );

    IF ( switch = 'black' )
      THEN WRITE( output, CHR( 27 ):1, '[30m' );

    IF ( switch = 'red' )
      THEN WRITE( output, CHR( 27 ):1, '[31m' );

    IF ( switch = 'green' )
      THEN WRITE( output, CHR( 27 ):1, '[32m' );

    IF ( switch = 'yellow' )
      THEN WRITE( output, CHR( 27 ):1, '[33m' );

    IF ( switch = 'blue' )
      THEN WRITE( output, CHR( 27 ):1, '[34m' );

    IF ( switch = 'magenta' )
      THEN WRITE( output, CHR( 27 ):1, '[35m' );
```

```
    IF ( switch = 'cyan' )
      THEN WRITE( output, CHR( 27 ):1, '[36m' );

    IF ( switch = 'white' )
      THEN WRITE( output, CHR( 27 ):1, '[37m' );

    IF ( switch = 'cyan background' )
      THEN WRITE( output, CHR( 27 ):1, '[44m' );

    IF ( switch = 'clear' )
      THEN WRITE( output, CHR(27):1, '[2', 'J' );

    IF ( switch = 'bell' )
      THEN WRITE( output, CHR(7):1 );

  END; (* video *)



FUNCTION ask_conf( comment, response : l_string ) : confirm;
  (* bevestig de gestelde vraag *)
  VAR conf       : l_string;
      check      : confirm;
      line, tab : integer_2;
  BEGIN

    line := 22; tab := 3;
    conf := response;

    video( 'white' );
    video( 'inverse' );
    REPEAT
      set_cursor( line, tab );
      video( 'blue' );
      WRITE( output, comment, response );
      set_cursor( line, tab );
      WRITE( output, comment );
      {$I-} (* Disable automatic error checking *)
      RESET( input );
      IF NOT EOLN( input )
        THEN READ( input, conf );
      {$I+} (* Enable automatic error checking *)


      check := ( ( IOresult = 0 ) AND
                 ( conf[ 1 ] IN [ 'j', 'J', 'n', 'N' ] ) );

      IF NOT check THEN video( 'bell' );

    UNTIL check;
```

```
    IF ( conf[ 1 ] IN [ 'j', 'J' ] )
      THEN ask_conf := TRUE
      ELSE ask_conf := FALSE;

    set_cursor( line, tab );
    WRITE( output, '                                                    ' )

  END; (* ask_conf *)




PROCEDURE prolog;
  (* initialiseer de input gegevens *)
  BEGIN

    (* produkt gegevens *)

    thick.val  :=      1.0;
    thick.low  :=      0.1;
    thick.upp  :=     10.0;
    thick.prm  :=  ' Plaatdikte          (     0.1   tot     10.0  ): ';
    thick.dim  :=  '   mm          ';

    c_nad.val  :=    620.0;
    c_nad.low  :=    100.0;
    c_nad.upp  :=   2000.0;
    c_nad.prm  :=  ' Constante C   Nadai  (   100.0   tot   2000.0  ): ';
    c_nad.dim  :=  '   N / mm'+chr(253)+'    ';

    n_nad.val  :=     0.23;
    n_nad.low  :=     0.05;
    n_nad.upp  :=     0.50;
    n_nad.prm  :=  ' Exponent  n   Nadai  (     0.05  tot      0.50 ): ';
    n_nad.dim  :=  '   -          ';

    eps_0.val  :=     0.01;
    eps_0.low  :=     0.0001;
    eps_0.upp  :=     0.5;
    eps_0.prm  :=  ' Voordeformatie       (     0.0001 tot  n Nadai ): ';
    eps_0.dim  :=  '   -          ';

    e_mod.val  := 210000.0;
    e_mod.low  :=  10000.0;
    e_mod.upp  := 250000.0;
    e_mod.prm  :=  ' Young-s modulus      ( 10000.0  tot  250000.0  ): ';
    e_mod.dim  :=  '   N / mm'+chr(253)+'    ';
```

```
    nu_po.val  :=      0.28;
    nu_po.low  :=      0.2;
    nu_po.upp  :=      0.5;
    nu_po.prm  :=  ' Poisson constante    (     0.2  tot       0.5  ): ';
    nu_po.dim  :=  '   -          ';

    stroke.val :=      1.00;
    stroke.low :=      0.00;
    stroke.upp :=    100.00;
    stroke.prm :=  ' Procesweg            (     0.0  tot     100.0  ): ';
    stroke.dim :=  '   mm         ';


    (* gereedschap gegevens *)

    rho_d.val :=       1.0;
    rho_d.low :=       0.01;
    rho_d.upp :=     250.0;
    rho_d.prm :=   ' Matrijs radius       (     0.05 tot     250.0  ): ';
    rho_d.dim :=   '   mm         ';

    rho_p.val :=       1.0;
    rho_p.low :=       0.05;
    rho_p.upp :=     100.0;
    rho_p.prm :=   ' Stempel radius       (     0.05 tot     100.0  ): ';
    rho_p.dim :=   '   mm         ';

    rho_t.val :=       5.0;
    rho_t.low :=       0.01;
    rho_t.upp :=     250.0;
    rho_t.prm :=   ' Tegenhouder radius   (     0.05 tot     250.0  ): ';
    rho_t.dim :=   '   mm         ';

    beta_t.val :=      pi/8;
    beta_t.low :=      0.0;
    beta_t.upp :=     80.0;
    beta_t.prm :=     ' Tegenh. afr. hoek ' + chr(225) + 't (     0.0  tot
90.0  ): ';
    beta_t.dim :=  '   gr.min    ';

    breedte_p.val :=     10.0;
    breedte_p.low :=      0.1;    {thick.low}
    breedte_p.upp :=    250.0;    {1.5 * thick.upp}
    breedte_p.prm :=  ' stempelbreedte       (     0.1  tot     250.0  ): ';
    breedte_p.dim :=  '   mm         ';

    zetspleet.val :=      1.2;
    zetspleet.low :=      0.1;    {thick.low}
    zetspleet.upp :=     15.0;    {1.5 * thick.upp}
    zetspleet.prm :=  ' Zetspleet            (  1.0 tot 1.5*plaatdikte ): ';
    zetspleet.dim :=  '   mm         ';
```

```
    friction.val  :=      0.10;
    friction.low  :=      0.01;
    friction.upp  :=      0.50;
    friction.prm  :=  ' Wrijvingscoefficient (     0.01 tot       0.50 ): ';
    friction.dim  :=  '   -          ';

    (* uitkomsten *)

    angle.val             :=      1.00;
    angle.low             :=      0.00;
    angle.upp             :=      0.00;
    angle.prm             := ' Produkthoek (onbelast) : ';
    angle.dim             :=  '   gr.min  ';

    force_normal.val      :=      2.00;
    force_normal.low      :=      0.00;
    force_normal.upp      :=      0.00;
    force_normal.prm      := ' Maximum Normaalkracht  : ';
    force_normal.dim      :=  '   N / mm  ';

    force_punch.val       :=       2.00;
    force_punch.low       :=       0.00;
    force_punch.upp       :=       0.00;
    force_punch.prm       := ' Maximum Stempelkracht  : ';
    force_punch.dim       :=  '   N / mm  ';

    force_tegenhouder.val :=       2.00;
    force_tegenhouder.low :=       0.00;
    force_tegenhouder.upp :=       0.00;
    force_tegenhouder.prm := ' Tegenhouderkracht      : ';
    force_tegenhouder.dim :=  '   N / mm  ';

    qster.val             :=        2.00;
    qster.low             :=        0.00;
    qster.upp             :=        0.00;
    qster.prm             := ' Vlak Stempeldeel       : ';
    qster.dim             :=  '   mm      ';

    delta_vering.val      :=      1.00;
    delta_vering.low      :=      0.00;
    delta_vering.upp      :=      0.00;
    delta_vering.prm      := ' Totale terugveerhoek   : ';
    delta_vering.dim      :=  '   gr.min  ';

  END; (* prolog *)



PROCEDURE un_do;
  (* logo *)
  BEGIN

    (* toon het logo *)
```

```
    video( 'normal' );
    video( 'white' );
    video( 'inverse' );
    video( 'clear' );
    video( 'blue' );
    line :=1; tab := 1;
    set_cursor( line, tab );
    WRITELN;
    WRITELN( output, '  (****** U-buigen (model Hoogenboom) *****) ' );

    line :=19;
    set_cursor( line, tab );
    WRITELN;
    WRITELN( output, '  (** TNO/TUE ( c ) 1989 **) ' );

  END; (* un_do *)


PROCEDURE get_test_values;
(* leest input waarden van file met IO redirection *)

VAR
  i,j,k : integer_2;

  BEGIN
    READLN( input, thick.val, c_nad.val, n_nad.val, eps_0.val,
                   e_mod.val, nu_po.val, stroke.val, rho_d.val,
                   rho_p.val, rho_t.val, beta_t.val, breedte_p.val,
                   zetspleet.val, friction.val );

  END;

PROCEDURE get_values;
  (* lees de invoerwaarden *)
  TYPE format = ( rea_val, ang_val );

    PROCEDURE get_real( VAR data : facts );
      (* leest en controleert een real waarde *)
      VAR val   : real_4;
          check : confirm;
      BEGIN

        REPEAT

          val := data.val;
          set_cursor( line, tab );
          video( 'blue' );
          WRITE( output, data.val:10:2 );
          set_cursor( line, tab );
          {$I-} (* Disable automatic error checking *)
          RESET( input );
```

```
      IF NOT EOLN( input )
        THEN READ( input, val );
      {$I+} (* Enable automatic error checking *)

      check := ( ( val >= data.low ) AND
                 ( val <= data.upp ) AND
                 ( IOresult = 0 ) );

      IF NOT check
        THEN
          BEGIN
            video( 'bell ');
            set_cursor( line, tab );
            video( 'yellow' );
            WRITE( output, '          ' );
          END;

    UNTIL check;

    data.val := val;
    set_cursor( line, tab );
    WRITE( output, data.val:10:2 )

  END; (* get_real *)


PROCEDURE get_ang( VAR data : facts );
  (* hoeken worden afgebeeld en ingegeven  in sexagesimale notatie,
     in de variabele wordt de waarde in radialen opgenomen *)
  VAR i, j, k, l : integer_2;
  BEGIN

    data.val := data.val * ( 360.0 / ( 2.0 * pi ) );
    IF ( data.val < 0.0 )
      THEN l := -1
      ELSE l :=  1;

    data.val := ABS( data.val*100.0 );
    k        := ROUND( data.val );
    i        := k DIV 100;
    j        := k MOD 100;

    data.val := ( i + ( j / 100.0 ) * 0.6 ) * l;

    get_real( data );

    IF ( data.val < 0.0 )
      THEN l := -1
      ELSE l :=  1;
```

```
    data.val := ABS( data.val * 100.0 );
    k        := ROUND( data.val );
    i        := k DIV 100;
    j        := k MOD 100;

    data.val := ( i + ( j / 60.0 ) ) * 1;
    data.val := data.val * ( (2.0 * pi) / 360.0 )

  END; (* get_ang *)


PROCEDURE one_value( VAR data : facts; field : format );
  (* lees een invoerwaarde *)
  BEGIN

    line := line +1;
    tab := 8;
    set_cursor( line, tab );
    video( 'inverse' );
    video( 'bold' );
    video( 'yellow' );
    WRITE( output, data.prm );
    video( 'yellow' );
    WRITE( output, data.val:10:2 );
    video( 'yellow' );
    WRITE( output, data.dim );

    tab := 59;
    set_cursor( line, tab );
    IF ( field = rea_val )
      THEN
        BEGIN
          get_real( data );
        END
      ELSE
        BEGIN
          get_ang( data );
        END;


  END; (* one_value *)


BEGIN

  line := 4;

  one_value( thick     , rea_val   );
  one_value( c_nad     , rea_val   );
  one_value( n_nad     , rea_val   );
  one_value( eps_0     , rea_val   );
  one_value( e_mod     , rea_val   );
  one_value( nu_po     , rea_val   );
  one_value( stroke    , rea_val   );
```

```
    one_value( rho_d     , rea_val   );
    one_value( rho_p     , rea_val   );
    one_value( rho_t     , rea_val   );
    one_value( beta_t    , ang_val   );
    one_value( breedte_p , rea_val   );
    one_value( zetspleet , rea_val   );
    one_value( friction  , rea_val   )

  END; (* get_values *)



PROCEDURE calculate;
  (* bevat het model van het U-buig mechanisme *)


  BEGIN

    {normalisatie input}

    IF zetspleet.val=thick.val THEN zetspleet.val:=zetspleet.val+1E-12;

    WITH Materiaal DO BEGIN
      Emodulus:= e_mod.val;
      nu      := nu_po.val;
      C       := c_nad.val;
      n       := n_nad.val;
      Epsilon0:= eps_0.val;
      END;{with}

    WITH Gereedschap DO BEGIN
      w0                    := rho_d.val+rho_p.val+zetspleet.val;    (* mm *)
      Zetspleet_ster        := zetspleet.val/w0;
      RhoDie_ster           := rho_d.val/w0;
      RhoPunch_ster         := rho_p.val/w0;
      RhoTegenhouder_ster   := rho_t.val/w0;
      Breedte_p_ster        := breedte_p.val/w0;
      Beta_t_rad            := Beta_t.val;
      END;{with}

    WITH Produkt DO BEGIN
      s0_ster      := thick.val/Gereedschap.w0;
      END;{with}

    WITH Proces DO BEGIN
      mu           := friction.val;
      Weg_ster     := stroke.val/Gereedschap.w0;
      END;{with}
```

```
    {*** berekenen van u-buigproces ***}
       {  input  = Materiaal, Produkt, Proces, Gereedschap
          output = Materiaal, Produkt, Proces, Gereedschap }

    BerekenUBuigen;


    {normalisatie output}

    angle.val              := Produkt.Alpha_onbelast;
    force_normal.val       := Proces.Fnormaal_ster_max*Materiaal.C
                              * SQR(Produkt.s0_ster) * Gereedschap.w0;
    force_punch.val        := Proces.Fpunch_ster_max*Materiaal.C
                              * SQR(Produkt.s0_ster) * Gereedschap.w0;
    force_tegenhouder.val  := Proces.Ftegenhouder_ster_max*Materiaal.C
                              * SQR(Produkt.s0_ster) * Gereedschap.w0;
    qster.val              := Gereedschap.q_ster * Gereedschap.w0;
    delta_vering.val       := Proces.DeltaGamma;


  END;


PROCEDURE give_test_values;
(* schrijft output waarden naar file met IO redirection *)

VAR
  produkthoek_onbelast : REAL;
  k,l,i,j     : integer_2;

  FUNCTION give_ang( hoek : facts ):real;    (* hoek.val in radialen *)
    (* terug naar sexagesimale notatie *)

    BEGIN

      {transformatie van radialen naar graden (deg) }
      hoek.val:=hoek.val*180/Pi;


      IF ( hoek.val < 0.0 )
        THEN l := -1
        ELSE l :=  1;

      hoek.val   := ABS( hoek.val * 100.0 );
      k          := ROUND( hoek.val );
      i          := k DIV 100;
      j          := k MOD 100;
      hoek.val   := ( i + ( j / 100.0 ) * 0.6 ) * l;

      give_ang:=hoek.val;

  END;{give_ang}
```

```
  BEGIN

    produkthoek_onbelast:=give_ang(angle);
    delta_vering.val:=give_ang(delta_vering);

    WRITELN( output, produkthoek_onbelast:10:2, force_normal.val:10:2,
                     force_punch.val:10:2, force_tegenhouder.val:10:2,
                     qster.val:10:2, delta_vering.val:10:2 );

  END;



PROCEDURE give_values;
  (* toon uitkomsten op scherm *)
  TYPE format = ( rea_val, ang_val );
  VAR
      i, j, k, l : integer_2;
      conf       : confirm;


    PROCEDURE give_real( data : facts );
      BEGIN
        WRITE( output, data.val:9:2 )
      END;


    PROCEDURE give_ang( hoek : facts );     (* hoek.val in radialen *)
      (* terug naar sexagesimale notatie *)

      BEGIN

        {transformatie van radialen naar graden (deg) }
        hoek.val:=hoek.val*180/Pi;


        IF ( hoek.val < 0.0 )
          THEN l := -1
          ELSE l :=  1;

        hoek.val   := ABS( hoek.val * 100.0 );
        k          := ROUND( hoek.val );
        i          := k DIV 100;
        j          := k MOD 100;
        hoek.val   := ( i + ( j / 100.0 ) * 0.6 ) * l;

      set_cursor( line, tab );

      give_real( hoek )

    END;
```

```
PROCEDURE one_value( VAR data : facts; field : format );
  (* schrijft een invoerwaarde *)
  BEGIN

    line := line + 1;
    tab := 13;
    set_cursor( line, tab );
    video( 'cyan background' );
    video( 'cyan' );
    WRITE( output, ' ',data.prm );

    tab := 39;
    set_cursor( line, tab );
    video( 'yellow' );
    IF ( field = rea_val )
      THEN
        BEGIN
          give_real( data );
        END
      ELSE
        BEGIN
          give_ang( data );
        END;

    video( 'cyan' );
    WRITE( output, data.dim );

  END; (* one_value *)


BEGIN

  line := 7; tab := 13;
  set_cursor( line, tab );
    video( 'cyan background' );
    video( 'cyan' );
  WRITE( output, '                                                    ' );
  one_value( angle,              ang_val );
  one_value( force_normal,       rea_val );
  one_value( force_punch,        rea_val );
  one_value( force_tegenhouder,  rea_val );
  one_value( qster,              rea_val );
  one_value( delta_vering,       ang_val );

  line := line + 1;
  tab := 13;
  set_cursor( line, tab );
  WRITE( output, '                                                    ' );

END; (* store *)
```

```
FUNCTION get_exit : confirm;
  (* naar einde of herhaal het programma *)
  BEGIN
    IF ( redir = off )
    THEN BEGIN
      leeg_keyboardbuffer;
      get_exit := ask_conf( 'laatste berekening ? J/N : ', 'N' );
    END{then}
    ELSE
      get_exit := EOLN( input )
  END; (* get_exit *)



PROCEDURE epilog;
  (* bye *)
  VAR line, tab : integer_2;

  BEGIN
    video( 'normal' );
    video( 'white' );
    video( 'clear' );
  END; (* epilog *)




BEGIN (*********** program.body **************)

  (* input output redirection *)
  redir := off; (* select tussen on en off ) *)

  prolog;

  REPEAT

    CASE redir of
      off : BEGIN
              un_do;
              get_values;
              calculate;
              give_values;
            END;
      on  : BEGIN
              get_test_values;
              calculate;
              give_test_values;
            END;
    END;{case}
```

```
  UNTIL get_exit;
  IF ( redir = off ) THEN epilog;

END.(********** end program.body ***********)
```

Listing Module UBCalcul

(Module Technologieprogramma U-buigen)

```
(*****************************************************************
 *                                                               *
 *                                                               *
 *   Unit          : UBCalcul                                    *
 *                   (Module van Technologieprogramma U-buigen)  *
 *                    deterministisch model volgens Hoogenboom   *
 *                                                               *
 *   Overige modulen :  UBUIG (hoofdprogramma)                   *
 *                      UBVar                                    *
 *                                                               *
 *   Versie        : 1.00                                        *
 *                                                               *
 *   Datum         : 8 september 1989                            *
 *                                                               *
 *   Language      : Turbo Pascal versie 5.0 (Borland)           *
 *                                                               *
 *   Systeem       : IBM - AT/XT met EGA-kaart                   *
 *                                                               *
 *   Auteurs       : ir. S.M. Hoogenboom                         *
 *                 : ir. A.C.E.C. Melis                          *
 *                 : ir. A.B. Perduijn                           *
 *                                                               *
 *   Informatie    : Laboratorium voor Omvormtechnologie         *
 *                   Technische Universiteit Eindhoven           *
 *                                                               *
 *   Telefoon      : 040 - 472630 (Secretariaat)                 *
 *                                                               *
 *   (Gemaakt door TUE in opdracht van TNO in het kader van het  *
 *    project IOP-metalen buigen van voorbeklede plaat)          *
 *                                                               *
 *****************************************************************)
```

```
UNIT UBCalcul;

{berekent krachten en hoeken bij U-buigen.}

INTERFACE


USES
  UBVar;

PROCEDURE BerekenUBuigen;
```

```
IMPLEMENTATION


{************* Macht ************************************************}

FUNCTION macht(mantisse,exponent : real) : real;
VAR a,b,c:real;
BEGIN
  a:=ln(mantisse);
  b:=a*exponent;
  c:=exp(b);
  macht:=c;
END;{of function macht}




{************* Bereken U-buigen *********************************}

PROCEDURE BerekenUBuigen;


CONST
  w3        = 1.732050808;   (* wortel 3 *)
  TweeDerde = 0.6666666666667;

VAR
  i:integer;
  weg_ster_Stap,
  weg_ster_actual,
  w1_ster,
  RhoSterP,
  RhoSterT,
  at_ster,
  q_ster,
  ReciRhoVloei_ster,
  MomentElastisch,
  Cs_phiaan,
  Tn_phiaan,
  Phiaan,
  nuKwadraat,
  wortelnu,
  E_ster,
  HulpConstPlast1,
  HulpConstPlast2,
  Buigmodelhulp1,
  Buigmodelhulp2,
  BuigMomentP,
  BuigMomentT,
  delta_beta_t_EA,
```

```
  delta_beta_t_BH,
  delta_phiaan,
  delta_alpha_e : real;


  {************* Buigmodel ********************************************}

  FUNCTION BuigMomentSter(Rho_ster : real) : real;

  BEGIN
    WITH Materiaal DO BEGIN
      E_ster:=Emodulus / C;
      wortelnu   := sqrt(nu*nu - nu + 1.0);
      nuKwadraat := (1.0 - nu*nu);
      HulpConstPlast1:=tweeDerde * macht(epsilon0,3.0*n)
                       * nuKwadraat * nuKwadraat
                       / (E_ster*E_ster*macht(wortelnu,3.0));
      ReciRhoVloei_ster:=(2.0 * nuKwadraat * macht(epsilon0,n))
                     /(E_ster * wortelnu) ;
      HulpConstPlast2:=macht(ReciRhoVloei_ster/W3 + epsilon0 , n + 1.0)
                        * ( ReciRhoVloei_ster*(n + 1.0)/W3 - epsilon0 );
      MomentElastisch := HulpConstPlast1/SQR(1/Rho_ster);
      Buigmodelhulp1:= macht(1/(Rho_ster * w3) + epsilon0,1.0+n)
                         * ( (n+1.0)/(Rho_ster * w3) - epsilon0 )
                             - HulpConstPlast2;
      Buigmodelhulp2:= (n+1.0) * (n+2.0)* SQR(1/Rho_ster);

      IF Rho_ster > 1/ReciRhoVloei_ster
        THEN BuigMomentSter:=MomentElastisch
        ELSE BuigMomentSter:=MomentElastisch+w3*buigmodelhulp1/buigmodelhulp2;
    END;{with}
  END;{of function BuigMomentSter}



{********** BerekenUBuigen.body ***************************************}

BEGIN



  Proces.Fnormaal_ster_max     :=0;
  Proces.Fpunch_ster_max       :=0;
  Proces.Ftegenhouder_ster_max :=0;

  weg_ster_Stap:=Proces.weg_ster/20;
```

```
FOR i:=1 to 20 DO BEGIN

  weg_ster_actual:=i*weg_ster_Stap;

  WITH Gereedschap DO BEGIN
    w1_ster:=RhoDie_ster + RhoPunch_ster + Produkt.s0_ster;
  END;{with}


  WITH Proces, Produkt DO BEGIN
    at_ster:=SQRT(1 + SQR(weg_ster_actual) - 2 * w1_ster * weg_ster_actual);

    RhoSterP:=Gereedschap.RhoPunch_ster/s0_ster + 0.5;
    RhoSterT:=Gereedschap.RhoTegenhouder_ster/s0_ster + 0.5;

    BuigMomentP:=BuigMomentSter(RhoSterP);
    BuigMomentT:=BuigMomentSter(RhoSterT);

    Fnormaal_ster:=BuigMomentP/(at_ster + 0.5*mu*s0_ster);

    Cs_phiaan:=(at_ster+w1_ster*(w1_ster-weg_ster_actual))
                /(1+SQR(w1_ster-weg_ster_actual));
    Tn_phiaan:=SQRT(1-SQR(Cs_phiaan))/Cs_phiaan;

    Phiaan:=arctan(Tn_phiaan);



    delta_beta_t_EA := 12*(nuKwadraat/E_ster)*Gereedschap.Beta_t_rad
                       *BuigMomentT*RhoSterT;
    delta_beta_t_BH := 12*(nuKwadraat/E_ster)*Gereedschap.Beta_t_rad
                       *BuigMomentP*RhoSterP;
    delta_phiaan    := 12*(nuKwadraat/E_ster)*BuigMomentP*RhoSterP
                       *Phiaan;
    delta_alpha_e   := 6*Fnormaal_ster*SQR(at_ster)/(E_ster*s0_ster);

    DeltaGamma:=delta_beta_t_BH - delta_beta_t_EA
                + delta_phiaan + delta_alpha_e;

    Alpha_onbelast:=2*(pi/2-phiaan+DeltaGamma);

    WITH Gereedschap DO BEGIN
      q_ster:=(Breedte_p_ster-RhoPunch_ster)/COS(Beta_t_rad)
              - (RhoTegenhouder_ster + RhoPunch_ster + s0_ster)
                * SQRT(1-SQR(COS(Beta_t_rad)))/COS(Beta_t_rad);


      Ftegenhouder_ster:=2*COS(Beta_t_rad)*BuigMomentP/q_ster;
      Fpunch_ster:=2*((1-mu*at_ster/w1_ster)*Cs_phiaan+mu/w1_ster)
                    *Fnormaal_ster+FTegenhouder_ster;
    END;{with}
```

```
      IF Fnormaal_ster>Fnormaal_ster_max
        THEN Fnormaal_ster_max:=Fnormaal_ster;
      IF Fpunch_ster>Fpunch_ster_max
        THEN Fpunch_ster_max:=Fpunch_ster;
      IF Ftegenhouder_ster>Ftegenhouder_ster_max
        THEN Ftegenhouder_ster_max:=Ftegenhouder_ster;

    END;{with}


  END;{next}

END;{of procedure BerekenUBuigen}


END.{of unit UBCalcul}
```

Listing Module UBVar

(Module Technologieprogramma U-buigen)

```
(*****************************************************************
 *                                                               *
 *                                                               *
 *   Unit          : UBVar                                       *
 *                   (Module van Technologieprogramma U-buigen   *
 *                    deterministisch model volgens Hoogenboom  ) *
 *                                                               *
 *   Overige modulen :  UBUIG (hoofdprogramma)                   *
 *                      UBCalcul                                 *
 *                                                               *
 *   Versie        : 1.00                                        *
 *                                                               *
 *   Datum         : 8 september 1989                            *
 *                                                               *
 *   Language      : Turbo Pascal versie 5.0 (Borland)           *
 *                                                               *
 *   Systeem       : IBM - AT/XT met EGA-kaart                   *
 *                                                               *
 *   Auteurs       : ir. S.M. Hoogenboom                         *
 *                   ir. A.C.E.C. Melis                          *
 *                   ir. A.B. Perduijn                           *
 *                                                               *
 *   Informatie    : Laboratorium voor Omvormtechnologie         *
 *                   Technische Universiteit Eindhoven           *
 *                                                               *
 *   Telefoon      : 040 - 472630 (Secretariaat)                 *
 *                                                               *
 *   (Gemaakt door TUE in opdracht van TNO in het kader van het  *
 *    project IOP-metalen buigen van voorbeklede plaat)          *
 *                                                               *
 *****************************************************************)


UNIT UBVar;
{globale variabelen behorende bij het programma UBuig}


INTERFACE


TYPE
  MateriaalType = RECORD
    Emodulus,nu,                    (* Elastische grootheden *)
    C,n,Epsilon0 : real;            (* Plastische grootheden *)
    END;
```

```
  GereedschapGegType = RECORD
    w0,                                  (* mm           *)
    Zetspleet_ster,                      (* dimensieloos *)
    RhoDie_ster,                         (* dimensieloos *)
    RhoPunch_ster,                       (* dimensieloos *)
    RhoTegenhouder_ster,                 (* dimensieloos *)
    Beta_t_rad,                          (* radialen     *)
    Breedte_p_ster,                      (* dimensieloos *)
    q_ster          : real;              (* dimensieloos *)
    END;{record}

  ProcesGegType = RECORD
    mu,
    Fnormaal_ster,                       (* dimensieloos *)
    Fnormaal_ster_max,                   (* dimensieloos *)
    Fpunch_ster,                         (* dimensieloos *)
    Fpunch_ster_max,                     (* dimensieloos *)
    Ftegenhouder_ster,                   (* dimensieloos *)
    Ftegenhouder_ster_max,               (* dimensieloos *)
    DeltaGamma,                          (* radialen     *)
    Weg_ster : real;                     (* dimensieloos *)
    END;{record}

  ProduktGegType = RECORD
    s0_ster,                             (* dimensieloos *)
    Alpha_onbelast : real;               (* radialen     *)
    END;{record}


VAR
  Materiaal    : MateriaalType;
  Gereedschap : GereedschapGegType;
  Proces       : ProcesGegType;
  Produkt      : ProduktGegType;


IMPLEMENTATION


END. {of unit UBVar}
```