A COMPARISON OF THE PROGRAM PROVERS FPP, NPPV, SPARK

RESULT PROTOCOLS FOR SPARK-AUT 2002.JAN.18

Carsten Freining, Jürgen F H Winkler

Friedrich-Schiller University, Institute of Computer Science, D-07740 Jena, Germany
http://psc.informatik.uni-jena.de

Example 1

Ex_01vc.ada

```
  -- ex_01vc   Examiner: verification = vc

   --# main_program;
procedure ex_01vc (X: in out integer; Y: in out integer)
   --# derives X, Y from X, Y;
is
   Xi: integer := 0;
begin
   Xi := X;
      --# assert -127 <= X and X <= 127 and X=Xi;
   IF X<0
   THEN Y := -X;
   ELSE Y := X;
   END IF;
      --# assert Y <= 127 and Y = abs X and X=Xi;
end ex_01vc;
```

Ex_01vc.lst

```
          ******************************************************
                          Listing of SPARK Text
      SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                       DATE : 10-DEC-2001 16:20:11.86

Line
   1    -- ex_01vc   Examiner: verification = vc
   2
   3     --# main_program;
   4  procedure ex_01vc (X: in out integer; Y: in out integer)
   5     --# derives X, Y from X, Y;
   6  is
   7     Xi: integer := 0;

!!! (  1)  Flow Error        : 54: The initialization at declaration of Xi is
            ineffective.

   8  begin
```

```
    9     Xi := X;
          ^2
 !!! (  2)  Flow Error        : 10: Ineffective statement.

   10        --# assert -127 <= X and X <= 127 and X=Xi;
   11     IF X<0
   12     THEN Y := -X;
   13     ELSE Y := X;
   14     END IF;
   15        --# assert Y <= 127 and Y = abs X and X=Xi;
   16  end ex_01vc;

 !!! (  3)  Flow Error        : 31: The variable X is exported but not (inter-
nally)
            defined.
 !!! (  4)  Flow Error        : 35: Importation of the initial value of vari-
able Y
            is ineffective.
 !!! (  5)  Flow Error        : 33: The variable Xi is neither referenced nor
            exported.
 !!! (  6)  Flow Error        : 50: The imported value of Y is not used in the
            derivation of X.
 !!! (  7)  Flow Error        : 50: The imported value of Y is not used in the
            derivation of Y.

 --End of file------------------------------------------------
```

Ex_01vc.siv

```
          ******************************************************
                       Semantic Analysis of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                           Demonstration Version
          ******************************************************


       CREATED 10-DEC-2001, 16:20:11  SIMPLIFIED 10-DEC-2001, 16:20:29
                (Simplified by SPADE Simplifier, Version 1.4)

                                procedure ex_01vc



 For path(s) from start to assertion of line 10:

 procedure_ex_01vc_1.
 H1:    x >= integer__first .
 H2:    x <= integer__last .
 H3:    y >= integer__first .
 H4:    y <= integer__last .
        ->
 C1:    - 127 <= x .
 C2:    x <= 127 .

 For path(s) from assertion of line 10 to assertion of line 15:
```

```
procedure_ex_01vc_2.
H1:    - 127 <= xi .
H2:    xi < 0 .
       ->
C1:    - xi = abs(xi) .
```

true

```
procedure_ex_01vc_3.
H1:    xi <= 127 .
H2:    0 <= xi .
       ->
C1:    xi = abs(xi) .
```

true

```
For path(s) from assertion of line 15 to finish:

procedure_ex_01vc_4.
*** true .          /* all conclusions proved */
```

Ex_01:	4 VCs gen.	2 VCs rel.	0 of rel. VCs proved	checked 2001.12.14

Example 2

Ex_02vc.ada

```
  -- ex_02vc   Examiner: verification = vc

   --# main_program;
procedure ex_02vc (X: in out integer)
   --# derives X from X;
is
   SUBTYPE RA is Integer Range 1..10;
   TYPE TA is array(RA) of integer;
   A: TA;
begin
      --# assert  a(1) = 1 and a(5) = X ;
   a(a(1)) := 5;
      --# assert  a(a(1)) = 5;

end ex_02vc;
```

Ex_02vc.lst

```
          ******************************************************
                          Listing of SPARK Text
      SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                       DATE : 11-DEC-2001 08:09:51.56

Line
```

```
    1    -- ex_02vc   Examiner: verification = vc
    2
    3     --# main_program;
    4  procedure ex_02vc (X: in out integer)
    5     --# derives X from X;
    6  is
    7     SUBTYPE RA is Integer Range 1..10;
    8     TYPE TA is array(RA) of integer;
    9     A: TA;
   10  begin
   11        --# assert  a(1) = 1 and a(5) = X ;
   12     a(a(1)) := 5;
         ^1,2
 !!! (  1)  Flow Error         : 23: Statement contains reference(s) to unde-
fined
            variable A.
 !!! (  2)  Flow Error        : 10: Assignment to A is ineffective.

   13        --# assert  a(a(1)) = 5;
   14
   15  end ex_02vc;

 !!! (  3)  Flow Error         : 35: Importation of the initial value of vari-
able X
            is ineffective.
 !!! (  4)  Flow Error        : 31: The variable X is exported but not (inter-
nally)
            defined.
 !!! (  5)  Flow Error        : 50: The imported value of X is not used in the
            derivation of X.



 --End of file-----------------------------------------------------
```

Ex_02vc.siv

```
           ******************************************************
                       Semantic Analysis of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                           Demonstration Version
           ******************************************************


       CREATED 11-DEC-2001, 08:09:51  SIMPLIFIED 11-DEC-2001, 08:10:00
                (Simplified by SPADE Simplifier, Version 1.4)

                              procedure ex_02vc



 For path(s) from start to assertion of line 11:

 procedure_ex_02vc_1.
 H1:    x >= integer__first .
 H2:    x <= integer__last .
        ->
 C1:    element(a,[1]) = 1 .
```

```
C2:    element(a,[5]) = x .
```

```
For path(s) from assertion of line 11 to assertion of line 13:

procedure_ex_02vc_2.
H1:    element(a,[1]) = 1 .
       ->
C1:    element(a,[5]) = 5 .
```

partially correct:
x=5 or a(5)=x should be mentioned

```
For path(s) from assertion of line 13 to finish:

procedure_ex_02vc_3.
*** true .          /* all conclusions proved */
```

Ex_02:	3 VCs gen.	1 VC rel.	0 of rel. VCs correct	checked 2002.Jan.07

Example 3

Ex_03vc.ada

```
  -- ex_03vc   Examiner: verification = vc

   --# main_program;
procedure ex_03vc (n: in out integer; a: in out integer)
   --# derives n, a from n, a;
is

   x : integer := 0;
   y : integer := 0;

   function p(a: in integer) return boolean is
      Ret_Value: boolean;
   begin
      return Ret_Value;
   end p;

   function s(a: in integer; b: in integer) return integer is
      Ret_Value: integer;
   begin
      return Ret_Value;
   end s;

   function r(a: in integer) return integer is
      Ret_Value: integer;
   begin
      return Ret_Value;
   end r;

   function h(a: in integer) return integer is
      Ret_Value: integer;
   begin
      return Ret_Value;
   end h;
```

```
   --# function f(a: in integer) return integer;

begin
   --# assert true;

   x := a;
   y := n;

   while not P(y) loop

      x := s(x,y);
      y := r(y);

   --# assert s(x,f(y)) = f(n);

   end loop;

   x := s(x,h(y));

   --# assert x = f(n);
end ex_03vc;
```

Ex_03vc.lst

```
          ******************************************************
                          Listing of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                        DATE : 04-JAN-2002 09:10:23.62

Line
   1    -- ex_03vc   Examiner: verification = vc
   2
   3     --# main_program;
   4  procedure ex_03vc (n: in out integer; a: in out integer)
   5     --# derives n, a from n, a;
   6  is
   7
   8     x : integer := 0;

!!! (  1)  Flow Error        : 54: The initialization at declaration of x is
            ineffective.

   9     y : integer := 0;

!!! (  2)  Flow Error        : 54: The initialization at declaration of y is
            ineffective.

  10
  11     function p(a: in integer) return boolean is
  12        Ret_Value: boolean;
  13     begin
  14        return Ret_Value;
```

```
                    ^3
 !!! (  3)  Flow Error         : 20: Expression contains reference(s) to unde-
fined
            variable Ret_Value.

   15     end p;

 !!! (  4)  Flow Error         : 50: The imported value of a is not used in the
            derivation of the function value.
 ??? (  5)  Warning             :602: The undefined initial value of Ret_Value
may be
            used in the derivation of the function value.
 !!! (  6)  Flow Error         : 30: The variable a is imported but neither
            referenced nor exported.
 !!! (  7)  Flow Error         : 32: The variable Ret_Value is neither imported
nor
            defined.

   16
   17     function s(a: in integer; b: in integer) return integer is
   18        Ret_Value: integer;
   19     begin
   20        return Ret_Value;
                    ^8
 !!! (  8)  Flow Error         : 20: Expression contains reference(s) to unde-
fined
            variable Ret_Value.

   21     end s;

 !!! (  9)  Flow Error         : 32: The variable Ret_Value is neither imported
nor
            defined.
 !!! ( 10)  Flow Error         : 50: The imported value of a is not used in the
            derivation of the function value.
 !!! ( 11)  Flow Error         : 50: The imported value of b is not used in the
            derivation of the function value.
 ??? ( 12)  Warning             :602: The undefined initial value of Ret_Value
may be
            used in the derivation of the function value.
 !!! ( 13)  Flow Error         : 30: The variable a is imported but neither
            referenced nor exported.
 !!! ( 14)  Flow Error         : 30: The variable b is imported but neither
            referenced nor exported.

   22
   23     function r(a: in integer) return integer is
   24        Ret_Value: integer;
   25     begin
   26        return Ret_Value;
                    ^15
 !!! ( 15)  Flow Error         : 20: Expression contains reference(s) to unde-
fined
            variable Ret_Value.

   27     end r;

 !!! ( 16)  Flow Error         : 50: The imported value of a is not used in the
            derivation of the function value.
```

```
 ??? ( 17)  Warning            :602: The undefined initial value of Ret_Value
may be
            used in the derivation of the function value.
 !!! ( 18)  Flow Error         : 30: The variable a is imported but neither
            referenced nor exported.
 !!! ( 19)  Flow Error         : 32: The variable Ret_Value is neither imported
nor
            defined.

   28
   29     function h(a: in integer) return integer is
   30        Ret_Value: integer;
   31     begin
   32        return Ret_Value;
                    ^20
 !!! ( 20)  Flow Error         : 20: Expression contains reference(s) to unde-
fined
            variable Ret_Value.

   33     end h;

 !!! ( 21)  Flow Error         : 50: The imported value of a is not used in the
            derivation of the function value.
 ??? ( 22)  Warning            :602: The undefined initial value of Ret_Value
may be
            used in the derivation of the function value.
 !!! ( 23)  Flow Error         : 30: The variable a is imported but neither
            referenced nor exported.
 !!! ( 24)  Flow Error         : 32: The variable Ret_Value is neither imported
nor
            defined.

   34
   35
   36     --# function f(a: in integer) return integer;
   37
   38  begin
   39     --# assert true;
   40
   41     x := a;
          ^25
 !!! ( 25)  Flow Error         : 10: Ineffective statement.

   42     y := n;
          ^26
 !!! ( 26)  Flow Error         : 10: Ineffective statement.

   43
   44     while not P(y) loop
                ^27
 !!! ( 27)  Flow Error         : 10: Ineffective statement.

   45
   46        x := s(x,y);
             ^28
 !!! ( 28)  Flow Error         : 10: Ineffective statement.

   47        y := r(y);
             ^29
```

```
 !!! ( 29)  Flow Error        : 10: Ineffective statement.

   48
   49     --# assert s(x,f(y)) = f(n);
   50
   51     end loop;
   52
   53     x := s(x,h(y));
          ^30
 !!! ( 30)  Flow Error        : 10: Ineffective statement.

   54
   55     --# assert x = f(n);
   56  end ex_03vc;

 !!! ( 31)  Flow Error        : 31: The variable n is exported but not (inter-
nally)
            defined.
 !!! ( 32)  Flow Error        : 31: The variable a is exported but not (inter-
nally)
            defined.
 !!! ( 33)  Flow Error        : 50: The imported value of a is not used in the
            derivation of n.
 !!! ( 34)  Flow Error        : 50: The imported value of n is not used in the
            derivation of a.



 --End of file-----------------------------------------------------
```

Ex_03vc.siv

```
          ******************************************************
                       Semantic Analysis of SPARK Text
             SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                            Demonstration Version
          ******************************************************


             CREATED 04-JAN-2002, 09:10:23  SIMPLIFIED 04-JAN-2002, 09:10:31
                      (Simplified by SPADE Simplifier, Version 1.4)

                                 procedure ex_03vc



 For path(s) from start to assertion of line 39:

 procedure_ex_03vc_1.
 *** true .          /* all conclusions proved */


 For path(s) from assertion of line 39 to assertion of line 49:

 procedure_ex_03vc_2.
 H1:    not p(n) .
```

```
       ->
C1:    s(s(a,n),f(r(n))) = f(n) .


For path(s) from assertion of line 49 to assertion of line 49:

procedure_ex_03vc_3.
H1:    s(x,f(y)) = f(n) .
H2:    not p(y) .
       ->
C1:    s(s(x,y),f(r(y))) = f(n) .


For path(s) from assertion of line 39 to assertion of line 55:

procedure_ex_03vc_4.
H1:    p(n) .
       ->
C1:    s(a,h(n)) = f(n) .


For path(s) from assertion of line 49 to assertion of line 55:

procedure_ex_03vc_5.
H1:    s(x,f(y)) = f(n) .
H2:    p(y) .
       ->
C1:    s(x,h(y)) = f(n) .
```

```
For path(s) from assertion of line 55 to finish:

procedure_ex_03vc_6.
*** true .          /* all conclusions proved */
```

Ex_03:	6 VCs gen.	4 VC rel.	4 of rel. VCs correct	checked 2002.Jan.07

Example 4

Ex_04vc.ada

```
  -- ex_04vc   Examiner: verification = vc

   --# main_program;
procedure ex_04vc (n: in out integer; prod: in out integer)
   --# derives n, prod from n, prod;
is
   n_i: integer := 0;
   --# function fact(n: in integer) return natural;
begin
   n_i := n;
   prod := 1;
```

```
      --# assert prod = 1 and n >= 0 and n = n_i;
FOR i IN integer range 1..n LOOP
    prod := prod * i;
      --# assert prod = fact(i) and n>=0 and n = n_i;
END LOOP;
      --# assert prod = fact(n) and n>=0 and n = n_i;
end ex_04vc;
```

Ex_04vc.lst

```
          ****************************************************
                          Listing of SPARK Text
      SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ****************************************************


                      DATE : 18-DEC-2001 10:44:55.22

Line
   1    -- ex_04vc   Examiner: verification = vc
   2
   3     --# main_program;
   4  procedure ex_04vc (n: in out integer; prod: in out integer)
   5     --# derives n, prod from n, prod;
   6  is
   7     n_i: integer := 0;

!!! (  1)  Flow Error        : 54: The initialization at declaration of n_i is
           ineffective.

   8     --# function fact(n: in integer) return natural;
   9  begin
  10     n_i := n;
         ^2
!!! (  2)  Flow Error        : 10: Ineffective statement.

  11     prod := 1;
  12        --# assert prod = 1 and n >= 0 and n = n_i;
  13  FOR i IN integer range 1..n LOOP
  14      prod := prod * i;
  15        --# assert prod = fact(i) and n>=0 and n = n_i;
  16  END LOOP;
  17        --# assert prod = fact(n) and n>=0 and n = n_i;
  18  end ex_04vc;

!!! (  3)  Flow Error        : 31: The variable n is exported but not (internally)
           defined.
!!! (  4)  Flow Error        : 35: Importation of the initial value of variable
           prod is ineffective.
!!! (  5)  Flow Error        : 33: The variable n_i is neither referenced nor
           exported.
!!! (  6)  Flow Error        : 50: The imported value of prod is not used in the
           derivation of n.
!!! (  7)  Flow Error        : 50: The imported value of prod is not used in the
```

```
           derivation of prod.


--End of file-----------------------------------------------
```

Ex_04vc.rls

```
         /*********************************************************/
                        /*Proof Rule Declarations*/
      /*SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00*/
                         /*Demonstration Version*/
         /*********************************************************/


                      /*DATE : 18-DEC-2001 10:44:55.20*/

                           /*procedure ex_04vc*/
```

```
rule_family fact:
   fact(n) requires [n: i].
   fact(1): fact(0) may_be_replaced_by 1.
   fact(2): fact(a+1) may_be_replaced_by fact(a)*(a+1) if [a>=0].
```

definition of the proof function fact
but it is NOT used by the Simplifier
the rules for fact have been added manually

```
rule_family ex_04vc_rules:
     X      requires [X:any] &
     X <= Y requires [X:ire, Y:ire] &
     X >= Y requires [X:ire, Y:ire].

ex_04vc_rules(1): character__pos(X) may_be_replaced_by X.
ex_04vc_rules(2): character__val(X) may_be_replaced_by X.
ex_04vc_rules(3): integer__first <= integer__last may_be_deduced.
ex_04vc_rules(4): integer__base__first <= integer__base__last may_be_deduced.
ex_04vc_rules(5): integer__base__first <= integer__first may_be_deduced.
ex_04vc_rules(6): integer__base__last >= integer__last may_be_deduced.
```

Ex_04vc.siv

```
         ***************************************************
                     Semantic Analysis of SPARK Text
      SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                         Demonstration Version
         ***************************************************


      CREATED 18-DEC-2001, 10:44:55  SIMPLIFIED 18-DEC-2001, 10:57:34
               (Simplified by SPADE Simplifier, Version 1.4)

                           procedure ex_04vc
```

```
For path(s) from start to assertion of line 12:

procedure_ex_04vc_1.
H1:    n >= integer__first .
H2:    n <= integer__last .
H3:    prod >= integer__first .
H4:    prod <= integer__last .
       ->
C1:    n >= 0 .
```

```
For path(s) from assertion of line 12 to assertion of line 15:

procedure_ex_04vc_2.
H1:    1 <= n_i .
       ->
C1:    1 = fact(1) .
```

true

```
For path(s) from assertion of line 15 to assertion of line 15:

procedure_ex_04vc_3.
H1:    n_i >= 0 .
H2:    loop__1__i <> n_i .
       ->
C1:    fact(loop__1__i) * (loop__1__i + 1) = fact(loop__1__i + 1) .
```

true, in principle
H2 should read : 1 <= loop__1__i < n_i

```
For path(s) from assertion of line 12 to assertion of line 17:

procedure_ex_04vc_4.
H1:    n_i >= 0 .
H2:    n_i < 1 .
       ->
C1:    1 = fact(n_i) .
```

true

```
For path(s) from assertion of line 15 to assertion of line 17:

procedure_ex_04vc_5.
*** true .          /* all conclusions proved */
```

```
For path(s) from assertion of line 17 to finish:

procedure_ex_04vc_6.
*** true .          /* all conclusions proved */
```

Ex_04:	6 VCs gen.	4 VC rel.	1 of rel. VCs proved	checked 2001.12.18

Example 5

Ex_05vc.ada

```
   -- ex_05vc2   Examiner: verification = vc

    --# main_program;
 procedure ex_05vc2 (n: in out integer; prod: in out integer)
    --# derives n, prod from n, prod;
 is
    subtype IndexTy is integer range 0..7;
    Type FactTy is array(IndexTy) of natural;
    fact: FactTy;
    n_i: integer := 0;
 begin
    prod := 1;
    fact(0) := prod;
    n_i := n;
       --# assert n >= 0 and n <= 7 and n = n_i and prod=1 and fact(0)=1;
 FOR i IN integer range 1..n LOOP
     prod := prod * i;
     fact(i) := prod;
       --# assert n>=0 and n <= 7 and n = n_i and prod<=32767 and prod=fact(i)
and
       --#        fact(0)=1 and ( for all K in indexty range 1..i =>
       --#                        (fact(K)=fact(K-1)*K) );
 END LOOP;
       --# assert n>=0 and n<=7 and n = n_i and prod<=32767 and prod=fact(n)
and
       --#        fact(0)=1 and ( for all K in indexty range 1..n =>
       --#                        (fact(K)=fact(K-1)*K) );
 end ex_05vc2;
```

Ex_05vc.lst

```
           ******************************************************
                          Listing of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
           ******************************************************


                        DATE : 19-DEC-2001 15:58:20.78

 Line
    1    -- ex_05vc2   Examiner: verification = vc
    2
    3     --# main_program;
    4  procedure ex_05vc2 (n: in out integer; prod: in out integer)
    5     --# derives n, prod from n, prod;
    6  is
    7     subtype IndexTy is integer range 0..7;
    8     Type FactTy is array(IndexTy) of natural;
    9     fact: FactTy;
   10     n_i: integer := 0;
```

```
 !!! (  1)  Flow Error         : 54: The initialization at declaration of n_i
is
            ineffective.

   11  begin
   12     prod := 1;
   13     fact(0) := prod;
          ^2,3
 !!! (  2)  Flow Error         : 23: Statement contains reference(s) to unde-
fined
            variable fact.
 !!! (  3)  Flow Error        : 10: Assignment to fact is ineffective.

   14     n_i := n;
          ^4
 !!! (  4)  Flow Error        : 10: Ineffective statement.

    15           --# assert  n >= 0 and n <= 7 and n = n_i and prod=1 and
fact(0)=1;
   16  FOR i IN integer range 1..n LOOP
   17      prod := prod * i;
   18      fact(i) := prod;
           ^5
 !!! (  5)  Flow Error        : 10: Assignment to fact is ineffective.

    19          --# assert n>=0 and n <= 7 and n = n_i and prod<=32767 and
prod=fact(i) and
   20        --#        fact(0)=1 and ( for all K in indexty range 1..i =>
   21        --#                        (fact(K)=fact(K-1)*K) );
   22  END LOOP;
    23           --# assert  n>=0 and  n<=7 and  n = n_i and prod<=32767 and
prod=fact(n) and
   24        --#        fact(0)=1 and ( for all K in indexty range 1..n =>
   25        --#                        (fact(K)=fact(K-1)*K) );
   26  end ex_05vc2;

 !!! (  6)  Flow Error        : 31: The variable n is exported but not (inter-
nally)
            defined.
 !!! (  7)  Flow Error         : 35: Importation of the initial value of vari-
able
            prod is ineffective.
 !!! (  8)  Flow Error        : 33: The variable n_i is neither referenced nor
            exported.
 !!! (  9)  Flow Error         : 50: The imported value of prod is not used in
the
            derivation of n.
 !!! ( 10)  Flow Error         : 50: The imported value of prod is not used in
the
            derivation of prod.


 --End of file-----------------------------------------------------
```

Ex_05vc.siv

```
           ******************************************************
```

```
                    Semantic Analysis of SPARK Text
        SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                           Demonstration Version
          ******************************************************


        CREATED 19-DEC-2001, 15:58:20  SIMPLIFIED 19-DEC-2001, 15:58:52
                 (Simplified by SPADE Simplifier, Version 1.4)

                              procedure ex_05vc2



 For path(s) from start to assertion of line 15:

 procedure_ex_05vc2_1.
 H1:    n >= integer__first .
 H2:    n <= integer__last .
 H3:    prod >= integer__first .
 H4:    prod <= integer__last .
        ->
 C1:    n >= 0 .
 C2:    n <= 7 .
```

```
 For path(s) from assertion of line 15 to assertion of line 19:

 procedure_ex_05vc2_2.
 H1:    n_i <= 7 .                          true
 H2:    element(fact,[0]) = 1 .
 H3:    1 <= n_i .
        ->
 C1:    (for_all)(k_ : integer,k_ >= 1 and k_ <= 1 ->
       element(update(fact,[1],1),[k_]) = element(update(fact,[1],1),[k_ - 1])
* k_) .


 For path(s) from assertion of line 19 to assertion of line 19:

 procedure_ex_05vc2_3.                      ( true )
 H1:    n_i >= 0 .                          H6 is not sufficient
 H2:    n_i <= 7 .                          H7:  0 <= loop__l__1  required
 H3:    element(fact,[loop__1__i]) <= 32767 .
 H4:    element(fact,[0]) = 1 .
 H5:    (for_all)(k_ : integer,k_ >= 1 and k_ <= loop__1__i ->
                           element(fact,[k_]) = element(fact,[k_ - 1]) * k_) .
 H6:    loop__1__i <> n_i .
        ->
 C1:    element(fact,[loop__1__i]) * (loop__1__i + 1) <= 32767 .
 C2:       element(update(fact,[loop__1__i  +  1],element(fact,[loop__1__i])  *
(loop__1__i + 1)),[0]) = 1 .
 C3:      (for_all)(k_ : integer,k_ >= 1 and k_ <= loop__1__i + 1  -> ele-
ment(update(fact,[loop__1__i + 1],element(fact,[loop__1__i]) * (loop__1__i +
1)),[k_]) = element(update(fact,[loop__1__i + 1],element(fact,[loop__1__i]) *
(loop__1__i + 1)),[k_ - 1]) * k_) .


 For path(s) from assertion of line 15 to assertion of line 23:
```

```
 procedure_ex_05vc2_4.
 H1:    n_i >= 0 .
 H2:    element(fact,[0]) = 1 .
 H3:    n_i < 1 .
        ->
 C1:    1 = element(fact,[n_i]) .
 C2:    (for_all)(k_ : integer,k_ >= 1 and k_ <= n_i -> element(fact,[k_]) =
                                                        element(fact,[k_ - 1])
* k_) .
```

true

```
 For path(s) from assertion of line 19 to assertion of line 23:

 procedure_ex_05vc2_5.
 *** true .          /* all conclusions proved */
```

```
 For path(s) from assertion of line 23 to finish:

 procedure_ex_05vc2_6.
 *** true .          /* all conclusions proved */
```

Ex_05:	6 VCs gen.	4 VC rel.	1 of rel. VCs proved	checked 2001.Dec.20

Example 6

Ex_06vc.ada

```
  -- ex_06vc   Examiner: verification = vc

   --# main_program;
procedure ex_06vc (x, y: in out integer; s: in out integer)
   --# derives x, y, s from x, y, s;
is
   x_i: integer := 0;
   y_i: integer := 0;
begin
   x_i := x;  y_i := y; s:= 0;
   --# assert  x_i>=0 and x=x_i and y=y_i and s=0;
   WHILE x /= 0 LOOP
      --# assert  x*y + s = x_i * y_i and x>0;
      WHILE x mod 2 = 0 LOOP
         y := 2*y;
         x := x / 2;
         --# assert  x*y + s = x_i * y_i and x>0;
      END LOOP;
      --# assert  x*y + s = x_i * y_i and x>0 and x mod 2 /= 0;
      s := s + y;
      x := x - 1;
      --# assert  x*y + s = x_i * y_i and x>=0;
   END LOOP;
   --# assert  s = x_i * y_i;
```

```
end ex_06vc;
```

Ex_06vc.lst

```
          ******************************************************
                          Listing of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                       DATE : 19-DEC-2001 10:34:23.32

Line
   1    -- ex_06vc   Examiner: verification = vc
   2
   3     --# main_program;
   4  procedure ex_06vc (x, y: in out integer; s: in out integer)
   5     --# derives x, y, s from x, y, s;
   6  is
   7     x_i: integer := 0;

!!! (  1)  Flow Error        : 54: The initialization at declaration of x_i is
           ineffective.

   8     y_i: integer := 0;

!!! (  2)  Flow Error        : 54: The initialization at declaration of y_i is
           ineffective.

   9  begin
  10     x_i := x;  y_i := y; s:= 0;
         ^3         ^4
!!! (  3)  Flow Error        : 10: Ineffective statement.
!!! (  4)  Flow Error        : 10: Ineffective statement.

  11     --# assert  x_i>=0 and x=x_i and y=y_i and s=0;
  12     WHILE x /= 0 LOOP
  13        --# assert  x*y + s = x_i * y_i and x>0;
  14        WHILE x mod 2 = 0 LOOP
  15           y := 2*y;
  16           x := x / 2;
  17           --# assert  x*y + s = x_i * y_i and x>0;
  18        END LOOP;
  19        --# assert  x*y + s = x_i * y_i and x>0 and x mod 2 /= 0;
  20        s := s + y;
  21        x := x - 1;
  22        --# assert  x*y + s = x_i * y_i and x>=0;
  23     END LOOP;
  24     --# assert  s = x_i * y_i;
  25
  26  end ex_06vc;

!!! (  5)  Flow Error        : 35: Importation of the initial value of vari-
able s
           is ineffective.
!!! (  6)  Flow Error        : 33: The variable x_i is neither referenced nor
           exported.
```

```
!!! (  7)  Flow Error        : 33: The variable y_i is neither referenced nor
           exported.
!!! (  8)  Flow Error        : 50: The imported value of y is not used in the
           derivation of x.
!!! (  9)  Flow Error        : 50: The imported value of s is not used in the
           derivation of x.
!!! ( 10)  Flow Error        : 50: The imported value of s is not used in the
           derivation of y.
!!! ( 11)  Flow Error        : 50: The imported value of s is not used in the
           derivation of s.


--End of file-----------------------------------------------------
```

Ex_06vc.siv

```
          ******************************************************
                      Semantic Analysis of SPARK Text
      SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                           Demonstration Version
          ******************************************************


      CREATED 19-DEC-2001, 10:34:23  SIMPLIFIED 19-DEC-2001, 10:34:31
               (Simplified by SPADE Simplifier, Version 1.4)

                              procedure ex_06vc



For path(s) from start to assertion of line 11:

procedure_ex_06vc_1.
H1:    x >= integer__first .
H2:    x <= integer__last .
H3:    y >= integer__first .
H4:    y <= integer__last .
H5:    s >= integer__first .
H6:    s <= integer__last .
       ->
C1:    x >= 0 .


For path(s) from assertion of line 11 to assertion of line 13:

procedure_ex_06vc_2.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 22 to assertion of line 13:

procedure_ex_06vc_3.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 13 to assertion of line 17:
```

```
procedure_ex_06vc_4.
H1:    x * y + s = x_i * y_i .
H2:    x > 0 .
H3:    x mod 2 = 0 .
       ->
C1:    x div 2 * (2 * y) + s = x_i * y_i .
C2:    x div 2 > 0 .
```

true

```
For path(s) from assertion of line 17 to assertion of line 17:

procedure_ex_06vc_5.
H1:    x * y + s = x_i * y_i .
H2:    x > 0 .
H3:    x mod 2 = 0 .
       ->
C1:    x div 2 * (2 * y) + s = x_i * y_i .
C2:    x div 2 > 0 .
```

true

```
For path(s) from assertion of line 13 to assertion of line 19:

procedure_ex_06vc_6.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 17 to assertion of line 19:

procedure_ex_06vc_7.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 19 to assertion of line 22:

procedure_ex_06vc_8.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 11 to assertion of line 24:
```

```
procedure_ex_06vc_9.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 22 to assertion of line 24:

procedure_ex_06vc_10.
*** true .          /* all conclusions proved */
```

```
For path(s) from assertion of line 24 to finish:

procedure_ex_06vc_11.
*** true .          /* all conclusions proved */
```

Ex_06:	11 VCs gen.	9 VC rel.	7 of rel. VCs proved	checked 2001.Dec.21

Example 7

Ex_07vc.ada

```
  -- ex_07vc   Examiner: verification = vc

   --# main_program;
procedure ex_07vc (x, y: in out integer; s: in out integer)
   --# derives x, y, s from x, y, s;
is
   x_i: integer := 0;
   y_i: integer := 0;
   termO: integer := 0;
   termI: integer := 0;
begin
   x_i := x;  y_i := y; s:= 0; termO := x;
   --# assert  x_i>=0 and x=x_i and y=y_i and s=0   and
   --#         termO>=0 and termO=x;
WHILE x /= 0 LOOP
   termO := x;
   termI := x;
      --# assert  x*y + s = x_i * y_i and x>0   and
      --#         termO>0 and termO=x and termI>=0 and termI=x;
   WHILE x mod 2 = 0 LOOP
      termI := x;
      y := 2*y;
      x := x / 2;
      --# assert  x*y + s = x_i * y_i and x>0   and
      --#         termO>0 and termO>=x and termI>0 and x<termI;
   END LOOP;
      --# assert  x*y + s = x_i * y_i and x>0   and
      --#         termO>0 and termO>=x;
   s := s + y;
   x := x - 1;
      --# assert  x*y + s = x_i * y_i and x>=0   and x<termO;
END LOOP;
   --# assert  s = x_i * y_i;

end ex_07vc;
```

Ex_07vc.lst

```
          ******************************************************
                          Listing of SPARK Text
      SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                       DATE : 19-DEC-2001 16:57:17.03

Line
```

```
   1    -- ex_07vc   Examiner: verification = vc
   2
   3     --# main_program;
   4  procedure ex_07vc (x, y: in out integer; s: in out integer)
   5     --# derives x, y, s from x, y, s;
   6  is
   7     x_i: integer := 0;

!!! (  1)  Flow Error        : 54: The initialization at declaration of x_i is
           ineffective.

   8     y_i: integer := 0;

!!! (  2)  Flow Error        : 54: The initialization at declaration of y_i is
           ineffective.

   9     termO: integer := 0;

!!! (  3)  Flow Error        : 54: The initialization at declaration of termO
is
           ineffective.

  10     termI: integer := 0;

!!! (  4)  Flow Error        : 54: The initialization at declaration of termI
is
           ineffective.

  11  begin
  12     x_i := x;  y_i := y; s:= 0; termO := x;
         ^5         ^6               ^7
!!! (  5)  Flow Error        : 10: Ineffective statement.
!!! (  6)  Flow Error        : 10: Ineffective statement.
!!! (  7)  Flow Error        : 10: Ineffective statement.

  13     --# assert  x_i>=0 and x=x_i and y=y_i and s=0   and
  14     --#         termO>=0 and termO=x;
  15  WHILE x /= 0 LOOP
  16     termO := x;
         ^8
!!! (  8)  Flow Error        : 10: Ineffective statement.

  17     termI := x;
         ^9
!!! (  9)  Flow Error        : 10: Ineffective statement.

  18        --# assert  x*y + s = x_i * y_i and x>0   and
  19        --#         termO>0 and termO=x and termI>=0 and termI=x;
  20     WHILE x mod 2 = 0 LOOP
  21        termI := x;
            ^10
!!! ( 10)  Flow Error        : 10: Ineffective statement.

  22        y := 2*y;
  23        x := x / 2;
  24        --# assert  x*y + s = x_i * y_i and x>0   and
  25        --#         termO>0 and termO>=x and termI>0 and x<termI;
  26     END LOOP;
  27        --# assert  x*y + s = x_i * y_i and x>0   and
```

```
  28        --#         termO>0 and termO>=x;
  29     s := s + y;
  30     x := x - 1;
  31        --# assert  x*y + s = x_i * y_i and x>=0   and x<termO;
  32  END LOOP;
  33     --# assert  s = x_i * y_i;
  34
  35  end ex_07vc;

!!! ( 11)  Flow Error        : 35: Importation of the initial value of vari-
able s
           is ineffective.
!!! ( 12)  Flow Error        : 33: The variable x_i is neither referenced nor
           exported.
!!! ( 13)  Flow Error        : 33: The variable y_i is neither referenced nor
           exported.
!!! ( 14)  Flow Error        : 33: The variable termO is neither referenced
nor
           exported.
!!! ( 15)  Flow Error        : 33: The variable termI is neither referenced
nor
           exported.
!!! ( 16)  Flow Error        : 50: The imported value of y is not used in the
           derivation of x.
!!! ( 17)  Flow Error        : 50: The imported value of s is not used in the
           derivation of x.
!!! ( 18)  Flow Error        : 50: The imported value of s is not used in the
           derivation of y.
!!! ( 19)  Flow Error        : 50: The imported value of s is not used in the
           derivation of s.



--End of file-----------------------------------------------------
```

Ex_07vc.siv

```
          ******************************************************
                       Semantic Analysis of SPARK Text
      SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                            Demonstration Version
          ******************************************************


      CREATED 19-DEC-2001, 16:57:16  SIMPLIFIED 19-DEC-2001, 16:58:52
               (Simplified by SPADE Simplifier, Version 1.4)

                              procedure ex_07vc



For path(s) from start to assertion of line 13:

procedure_ex_07vc_1.
H1:    x >= integer__first .
H2:    x <= integer__last .
H3:    y >= integer__first .
H4:    y <= integer__last .
```

```
H5:    s >= integer__first .
H6:    s <= integer__last .
       ->
C1:    x >= 0 .
```

```
For path(s) from assertion of line 13 to assertion of line 18:

procedure_ex_07vc_2.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 31 to assertion of line 18:

procedure_ex_07vc_3.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 18 to assertion of line 24:

procedure_ex_07vc_4.
H1:    x * y + s = x_i * y_i .
H2:    x > 0 .
H3:    x > 0 .
H4:    x mod 2 = 0 .
       ->
C1:    x div 2 * (2 * y) + s = x_i * y_i .
C2:    x div 2 > 0 .
C3:    x >= x div 2 .
C4:    x div 2 < x .


For path(s) from assertion of line 24 to assertion of line 24:

procedure_ex_07vc_5.
H1:    x * y + s = x_i * y_i .
H2:    x > 0 .
H3:    termo >= x .
H4:    x < termi .
H5:    x mod 2 = 0 .
       ->
C1:    x div 2 * (2 * y) + s = x_i * y_i .
C2:    x div 2 > 0 .
C3:    termo >= x div 2 .
C4:    x div 2 < x .


For path(s) from assertion of line 18 to assertion of line 27:

procedure_ex_07vc_6.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 24 to assertion of line 27:

procedure_ex_07vc_7.
*** true .          /* all conclusions proved */
```

```
For path(s) from assertion of line 27 to assertion of line 31:

procedure_ex_07vc_8.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 13 to assertion of line 33:

procedure_ex_07vc_9.
*** true .          /* all conclusions proved */
```

```
For path(s) from assertion of line 31 to assertion of line 33:

procedure_ex_07vc_10.
*** true .          /* all conclusions proved */
```

```
For path(s) from assertion of line 33 to finish:

procedure_ex_07vc_11.
*** true .          /* all conclusions proved */
```

Ex_07:	11 VCs gen.	9 VC rel.	7 of rel. VCs proved	checked 2001.12.21

Example 8

Ex_08vc.ada

```
  -- ex_08vc   Examiner: verification = vc

   --# main_program;
procedure ex_08vc (x, y: in out integer; s: in out integer)
   --# derives x, y, s from x, y, s;
is
   x_i: integer := 0;
   y_i: integer := 0;
   termO: integer := 0;
   termI: integer := 0;
begin
   x_i := x;  y_i := y; s:= 0; termO := x;
   --# assert  x_i>=0 and x=x_i and y=y_i and s=0 and
   --#         -32767 <= Y_i and Y_i <= 32767 and
   --#         -32767 <= X_i * Y_i and X_i * Y_i <= 32767   and
   --#         termO>=0 and termO=x;
WHILE x /= 0 LOOP
   termO := x;
   termI := x;
      --# assert  x*y + s = x_i * y_i and x>0   and
      --#         -32767<=x_i*y_i and x_i*y_i<=32767    and
      --#         termO>0 and termO=x and termI>=0 and termI=x;
```

```
   WHILE x mod 2 = 0 LOOP
      termI := x;
         --# assert x*y + s = x_i * y_i and x>=2 and x mod 2 = 0 and
         --#        -32767<=X_i*Y_i and X_i*Y_i<=32767    and
         --#        termO>0 and termO>=x and termI>0 and termI=x;
      y := 2*y;
      x := x / 2;
      --# assert  x*y + s = x_i * y_i and x>0   and
      --#         -32767<=X_i*Y_i and X_i*Y_i<=32767 and
      --#         termO>0 and termO>=x and termI>0 and x<termI;
   END LOOP;
      --# assert  x*y + s = x_i * y_i and x>0 and
      --#         -32767<=Y_i*X_i and X_i*Y_i<=32767 and
      --#         X mod 2 /= 0    and
      --#         termO>0 and termO>=x;
   s := s + y;
   x := x - 1;
      --# assert  x*y + s = x_i * y_i and x>=0 and
      --#         -32767 <= X_i * Y_i and X_i * Y_i <= 32767   and
      --#         x < termO;
END LOOP;
   --# assert  s = x_i * y_i and
   --#         -32767 <=s and s <= 32767;
```

```
end ex_08vc;
```

Ex_08vc.lst

```
          ******************************************************
                          Listing of SPARK Text
      SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                      DATE : 19-DEC-2001 10:55:45.99

Line
   1    -- ex_08vc   Examiner: verification = vc
   2
   3     --# main_program;
   4  procedure ex_08vc (x, y: in out integer; s: in out integer)
   5     --# derives x, y, s from x, y, s;
   6  is
   7     x_i: integer := 0;

!!! (  1)  Flow Error        : 54: The initialization at declaration of x_i is
           ineffective.

   8     y_i: integer := 0;

!!! (  2)  Flow Error        : 54: The initialization at declaration of y_i is
           ineffective.

   9     termO: integer := 0;

!!! (  3)  Flow Error        : 54: The initialization at declaration of termO
is
```

```
           ineffective.

  10     termI: integer := 0;

!!! (  4)  Flow Error        : 54: The initialization at declaration of termI
is
           ineffective.

  11  begin
  12     x_i := x;  y_i := y; s:= 0; termO := x;
         ^5         ^6               ^7
!!! (  5)  Flow Error        : 10: Ineffective statement.
!!! (  6)  Flow Error        : 10: Ineffective statement.
!!! (  7)  Flow Error        : 10: Ineffective statement.
```

```
  13     --# assert  x_i>=0 and x=x_i and y=y_i and s=0 and
  14     --#         -32767 <= Y_i and Y_i <= 32767 and
  15     --#         -32767 <= X_i * Y_i and X_i * Y_i <= 32767   and
  16     --#         termO>=0 and termO=x;
  17  WHILE x /= 0 LOOP
  18     termO := x;
         ^8
!!! (  8)  Flow Error        : 10: Ineffective statement.

  19     termI := x;
         ^9
!!! (  9)  Flow Error        : 10: Ineffective statement.

  20        --# assert  x*y + s = x_i * y_i and x>0   and
  21        --#         -32767<=x_i*y_i and x_i*y_i<=32767    and
  22        --#         termO>0 and termO=x and termI>=0 and termI=x;
  23     WHILE x mod 2 = 0 LOOP
  24        termI := x;
            ^10
!!! ( 10)  Flow Error        : 10: Ineffective statement.

  25           --# assert x*y + s = x_i * y_i and x>=2 and x mod 2 = 0 and
  26           --#        -32767<=X_i*Y_i and X_i*Y_i<=32767    and
  27           --#        termO>0 and termO>=x and termI>0 and termI=x;
  28        y := 2*y;
  29        x := x / 2;
  30        --# assert  x*y + s = x_i * y_i and x>0   and
  31        --#         -32767<=X_i*Y_i and X_i*Y_i<=32767 and
  32        --#         termO>0 and termO>=x and termI>0 and x<termI;
  33     END LOOP;
  34        --# assert  x*y + s = x_i * y_i and x>0 and
  35        --#         -32767<=Y_i*X_i and X_i*Y_i<=32767 and
  36        --#         X mod 2 /= 0    and
  37        --#         termO>0 and termO>=x;
  38     s := s + y;
  39     x := x - 1;
  40        --# assert  x*y + s = x_i * y_i and x>=0 and
  41        --#         -32767 <= X_i * Y_i and X_i * Y_i <= 32767   and
  42        --#         x < termO;
  43  END LOOP;
  44     --# assert  s = x_i * y_i and
  45     --#         -32767 <=s and s <= 32767;
```

```
  46
  47  end ex_08vc;

!!! ( 11)  Flow Error        : 35: Importation of the initial value of vari-
able s
           is ineffective.
!!! ( 12)  Flow Error        : 33: The variable x_i is neither referenced nor
           exported.
!!! ( 13)  Flow Error        : 33: The variable y_i is neither referenced nor
           exported.
!!! ( 14)  Flow Error        : 33: The variable termO is neither referenced
nor
           exported.
!!! ( 15)  Flow Error        : 33: The variable termI is neither referenced
nor
           exported.
!!! ( 16)  Flow Error        : 50: The imported value of y is not used in the
           derivation of x.
!!! ( 17)  Flow Error        : 50: The imported value of s is not used in the
           derivation of x.
!!! ( 18)  Flow Error        : 50: The imported value of s is not used in the
           derivation of y.
!!! ( 19)  Flow Error        : 50: The imported value of s is not used in the
           derivation of s.



--End of file-----------------------------------------------------
```

Ex_08vc.siv

```
          ******************************************************
                      Semantic Analysis of SPARK Text
      SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                           Demonstration Version
          ******************************************************


      CREATED 19-DEC-2001, 10:55:45  SIMPLIFIED 19-DEC-2001, 10:56:00
               (Simplified by SPADE Simplifier, Version 1.4)

                              procedure ex_08vc



For path(s) from start to assertion of line 13:

procedure_ex_08vc_1.
H1:    x >= integer__first .
H2:    x <= integer__last .
H3:    y >= integer__first .
H4:    y <= integer__last .
H5:    s >= integer__first .
H6:    s <= integer__last .
       ->
C1:    x >= 0 .
C2:    - 32767 <= y .
C3:    y <= 32767 .
```

```
C4:    - 32767 <= x * y .
C5:    x * y <= 32767 .
```

```
For path(s) from assertion of line 13 to assertion of line 20:

procedure_ex_08vc_2.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 40 to assertion of line 20:

procedure_ex_08vc_3.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 20 to assertion of line 25:

procedure_ex_08vc_4.
H1:    x * y + s = x_i * y_i .
H2:    x > 0 .
H3:    - 32767 <= x_i * y_i .
H4:    x_i * y_i <= 32767 .
H5:    x > 0 .
H6:    x mod 2 = 0 .
       ->
C1:    x >= 2 .
```

true

```
For path(s) from assertion of line 30 to assertion of line 25:

procedure_ex_08vc_5.
H1:    x * y + s = x_i * y_i .
H2:    x > 0 .
H3:    - 32767 <= x_i * y_i .
H4:    x_i * y_i <= 32767 .
 H5:    termo >= x .
 H6:    x < termi .
 H7:    x mod 2 = 0 .
        ->
 C1:    x >= 2 .
```

true

```
For path(s) from assertion of line 25 to assertion of line 30:

procedure_ex_08vc_6.
H1:    x * y + s = x_i * y_i .
H2:    x >= 2 .
H3:    x mod 2 = 0 .
H4:    - 32767 <= x_i * y_i .
H5:    x_i * y_i <= 32767 .
H6:    termo >= x .
       ->
C1:    x div 2 * (2 * y) + s = x_i * y_i .
C2:    x div 2 > 0 .
C3:    termo >= x div 2 .
C4:    x div 2 < x .
```

true

```
For path(s) from assertion of line 20 to assertion of line 34:

procedure_ex_08vc_7.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 30 to assertion of line 34:

procedure_ex_08vc_8.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 34 to assertion of line 40:

procedure_ex_08vc_9.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 13 to assertion of line 44:

procedure_ex_08vc_10.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 40 to assertion of line 44:

procedure_ex_08vc_11.
*** true .          /* all conclusions proved */
```

```
For path(s) from assertion of line 44 to finish:

procedure_ex_08vc_12.
*** true .          /* all conclusions proved */
```

Ex_08:	12 VCs gen.	10 VC rel.	7 of rel. VCs proved	checked 2001.Dec.21

Example 9

Ex_09vc.ada

```
  -- ex_09vc   Examiner: verification = vc

   --# main_program;
procedure ex_09vc (N: in out integer; current: in out integer)
   --# derives current, N from N, current;
is

   previous: integer := 0;
   count   : integer := 1;
   x       : integer := 0;
   n_i     : integer := 0;

   --# function fib(i: in integer) return integer;
```

```
begin
   current := 1;
   n_i := n;

   --# assert count = 1 and count <= N and n = n_i   and
   --#        current = fib(1) and previous = fib(0);
   WHILE count < N  LOOP

      x := current;
      current := current + previous;
      previous := x;
      count := count+1;

      --# assert count >= 1 and count <= N and n = n_i   and
      --#        current = fib(count) and previous = fib(count-1);
   END LOOP;
   --# assert current=fib(N);

end ex_09vc;
```

Ex_09vc.lst

```
          ******************************************************
                          Listing of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                       DATE : 02-JAN-2002 07:23:35.29

Line
   1    -- ex_09vc   Examiner: verification = vc
   2
   3     --# main_program;
   4  procedure ex_09vc (N: in out integer; current: in out integer)
   5     --# derives current, N from N, current;
   6  is
   7
   8     previous: integer := 0;
   9     count   : integer := 1;
  10     x       : integer := 0;

!!! (  1)  Flow Error        : 54: The initialization at declaration of x is
           ineffective.

  11     n_i     : integer := 0;

!!! (  2)  Flow Error        : 54: The initialization at declaration of n_i is
           ineffective.

  12
  13     --# function fib(i: in integer) return integer;
  14
  15  begin
  16     current := 1;
  17     n_i := n;
         ^3
```

```
!!! (  3)  Flow Error        : 10: Ineffective statement.

  18
  19     --# assert count = 1 and count <= N and n = n_i   and
  20     --#        current = fib(1) and previous = fib(0);
  21     WHILE count < N  LOOP
  22
  23        x := current;
  24        current := current + previous;
  25        previous := x;
  26        count := count+1;
  27
  28        --# assert count >= 1 and count <= N and n = n_i   and
  29        --#        current = fib(count) and previous = fib(count-1);
  30     END LOOP;
  31     --# assert current=fib(N);
  32
  33  end ex_09vc;

!!! (  4)  Flow Error        : 31: The variable N is exported but not (inter-
nally)
            defined.
!!! (  5)  Flow Error        : 35: Importation of the initial value of vari-
able
            current is ineffective.
!!! (  6)  Flow Error        : 33: The variable n_i is neither referenced nor
            exported.
!!! (  7)  Flow Error        : 50: The imported value of current is not used
in the
            derivation of N.
!!! (  8)  Flow Error        : 50: The imported value of current is not used
in the
            derivation of current.

--End of file------------------------------------------------
```

Ex_09vc.siv

```
          ******************************************************
                       Semantic Analysis of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                           Demonstration Version
          ******************************************************


       CREATED 02-JAN-2002, 07:23:35  SIMPLIFIED 02-JAN-2002, 07:23:51
                (Simplified by SPADE Simplifier, Version 1.4)

                               procedure ex_09vc



For path(s) from start to assertion of line 19:

procedure_ex_09vc_1.
H1:    n >= integer__first .
H2:    n <= integer__last .
H3:    current >= integer__first .
H4:    current <= integer__last .
```

```
       ->
C1:    1 <= n .
C2:    1 = fib(1) .
C3:    0 = fib(0) .
```

```
For path(s) from assertion of line 19 to assertion of line 28:

procedure_ex_09vc_2.
H1:    1 < n_i .                          true
       ->
C1:    fib(1) + fib(0) = fib(2) .


For path(s) from assertion of line 28 to assertion of line 28:

procedure_ex_09vc_3.
H1:    count >= 1 .
H2:    count < n_i .                      true
       ->
C1:    fib(count) + fib(count - 1) = fib(count + 1) .


For path(s) from assertion of line 19 to assertion of line 31:

procedure_ex_09vc_4.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 28 to assertion of line 31:

procedure_ex_09vc_5.
*** true .          /* all conclusions proved */
```

```
For path(s) from assertion of line 31 to finish:

procedure_ex_09vc_6.
*** true .          /* all conclusions proved */
```

Ex_09:	6 VCs gen.	4 VC rel.	2 of rel. VCs proved	checked 2002.Jan.03

Example 10

Ex_10vc.ada

```
  -- ex_10vc   Examiner: verification = vc

   --# main_program;
procedure ex_10vc (n: in out integer; current: in out integer)
   --# derives current, n from n, current;
is

   previous: integer := 0;
   count   : integer := 1;
```

```
   x       : integer := 0;
   n_i     : integer := 0;
   Term    : integer := 0;
   -- Terminierungsfunktion: Term = n-count

   --# function fib(i: in integer) return integer;

begin
   n_i := n;
   --# assert 1 <= n and n = n_i and fib(0) = 0 and fib(1) = 1;
   previous := 0;
   current := 1;
   count := 1;
   Term := n-count;

   --# assert count = 1 and count <= n and n = n_i and
   --#        previous = 0 and current = 1 and
   --#        fib(0) = 0 and fib(1) = 1 and
   --#        Term>=0;
   WHILE count < N  LOOP

      Term := n - count;

      x := current;
      current := current + previous;
      previous := x;
      count := count+1;

      --# assert count >= 1 and count <= N and previous = fib(count-1) and
      --#        current = fib(count) and n = n_i   and
      --#        Term > n - count and n - count >= 0;
   END LOOP;
   --# assert current=fib(n) and n=n_i;

end ex_10vc;
```

Ex_10vc.lst

```
          ******************************************************
                          Listing of SPARK Text
      SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                       DATE : 08-JAN-2002 14:59:15.02

Line
   1    -- ex_10vc   Examiner: verification = vc
   2
   3     --# main_program;
   4  procedure ex_10vc (n: in out integer; current: in out integer)
   5     --# derives current, n from n, current;
   6  is
   7
   8     previous: integer := 0;
```

```
!!! (  1)  Flow Error        : 54: The initialization at declaration of previ-
ous is
           ineffective.

   9     count   : integer := 1;

!!! (  2)  Flow Error        : 54: The initialization at declaration of count
is
           ineffective.

  10     x       : integer := 0;

!!! (  3)  Flow Error        : 54: The initialization at declaration of x is
           ineffective.

  11     n_i     : integer := 0;

!!! (  4)  Flow Error        : 54: The initialization at declaration of n_i is
           ineffective.

  12     Term    : integer := 0;

!!! (  5)  Flow Error        : 54: The initialization at declaration of Term
is
           ineffective.

  13     -- Terminierungsfunktion: Term = n-count
  14
  15     --# function fib(i: in integer) return integer;
  16
  17   begin
  18     n_i := n;
         ^6
!!! (  6)  Flow Error        : 10: Ineffective statement.

  19     --# assert 1 <= n and n = n_i and fib(0) = 0 and fib(1) = 1;
  20     previous := 0;
  21     current := 1;
  22     count := 1;
  23     Term := n-count;
         ^7
!!! (  7)  Flow Error        : 10: Ineffective statement.

  24
  25     --# assert count = 1 and count <= n and n = n_i and
  26     --#        previous = 0 and current = 1 and
  27     --#        fib(0) = 0 and fib(1) = 1 and
  28     --#        Term>=0;
  29     WHILE count < N  LOOP
  30
  31        Term := n - count;
            ^8
!!! (  8)  Flow Error        : 10: Ineffective statement.

  32
  33        x := current;
  34        current := current + previous;
  35        previous := x;
  36        count := count+1;
```

```
  37
  38        --# assert count >= 1 and count <= N and previous = fib(count-1)
and
  39        --#        current = fib(count) and n = n_i   and
  40        --#        Term > n - count and n - count >= 0;
  41     END LOOP;
  42     --# assert current=fib(n) and n=n_i;
  43
  44  end ex_10vc;

!!! (  9)  Flow Error        : 31: The variable n is exported but not (inter-
nally)
            defined.
!!! ( 10)  Flow Error        : 35: Importation of the initial value of vari-
able
            current is ineffective.
!!! ( 11)  Flow Error        : 33: The variable n_i is neither referenced nor
            exported.
!!! ( 12)  Flow Error        : 33: The variable Term is neither referenced nor
            exported.
!!! ( 13)  Flow Error        : 50: The imported value of current is not used
in the
            derivation of n.
!!! ( 14)  Flow Error        : 50: The imported value of current is not used
in the
            derivation of current.

--End of file-----------------------------------------------------
```

Ex_10vc.siv

```
          ******************************************************
                      Semantic Analysis of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                           Demonstration Version
          ******************************************************


       CREATED 08-JAN-2002, 14:59:14  SIMPLIFIED 08-JAN-2002, 14:59:48
                (Simplified by SPADE Simplifier, Version 1.4)

                               procedure ex_10vc



For path(s) from start to assertion of line 19:

procedure_ex_10vc_1.
H1:    n >= integer__first .
H2:    n <= integer__last .
H3:    current >= integer__first .
H4:    current <= integer__last .
       ->
C1:    1 <= n .
C2:    fib(0) = 0 .
C3:    fib(1) = 1 .
```

```
For path(s) from assertion of line 19 to assertion of line 25:

procedure_ex_10vc_2.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 25 to assertion of line 38:

procedure_ex_10vc_3.
H1:    fib(0) = 0 .
H2:    fib(1) = 1 .
H3:    term >= 0 .
H4:    1 < n_i .
       ->
C1:    1 = fib(2) .
```

true

```
For path(s) from assertion of line 38 to assertion of line 38:

procedure_ex_10vc_4.
H1:    count >= 1 .
H2:    term > n_i - count .
H3:    count < n_i .
       ->
C1:    fib(count) + fib(count - 1) = fib(count + 1) .
```

true

```
For path(s) from assertion of line 25 to assertion of line 42:

procedure_ex_10vc_5.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 38 to assertion of line 42:

procedure_ex_10vc_6.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 42 to finish:

procedure_ex_10vc_7.
*** true .          /* all conclusions proved */
```

Ex_10:	7 VCs gen.	5 VC rel.	3 of rel. VCs proved	checked 2002.Jan.08

Example 11

Ex_11vc.ada

```
  -- ex_11vc   Examiner: verification = vc

   --# main_program;
procedure ex_11vc (n: in out integer; current: in out integer)
```

```
   --# derives current, n from n, current;
is

   previous: integer := 0;
   count   : integer := 1;
   x       : integer := 0;
   n_i     : integer := 0;
   Term    : integer := 0;
   -- Terminierungsfunktion: Term = n-count

   SubType IndexTy is integer range 0..23;
   Type FibTy is array(indexTy) of natural;
   fib: FibTy;



begin
   n_i := n;
   --# assert n >= 1 and n <= 23 and n = n_i;

   previous := 0;
   current  := 1;
   count    := 1;
   fib(0)   := 0;
   fib(1)   := 1;
   Term     := n-count;

   --# assert n >= 1 and n <= 23 and n = n_i and
   --#        previous = 0 and current = 1 and
   --#        count = 1 and
   --#        1 = fib(1) and 0 = fib(0) and
   --#        Term >= 0;
   WHILE count < n  LOOP

      Term := n - count;
      fib(count+1) := fib(count) + fib(count-1);

      x := current;
      current := current + previous;
      previous := x;
      count := count+1;

      --# assert count >= 1 and count <= n and n<=23 and
      --#        previous = fib(count-1) and
      --#        current = fib(count) and n = n_i and
      --#        current <= 32767 and

      --#        fib(0) = 0 and fib(1) = 1 and
      --#        (for all K in indexTy range 2..count =>
      --#        (fib(K) = fib(k-2) + fib(k-1)) )   and

      --#        Term > n-count and n-count >= 0;
   END LOOP;

   --# assert current=fib(n) and current <= 32767 and n=n_i and n<=23 and
   --#        fib(0) = 0 and fib(1) = 1 and
   --#        (for all K in indexTy range 2..n =>
   --#        (fib(K) = fib(k-2) + fib(k-1)) );
```

```
 end ex_11vc;
```

Ex_11vc.lst

```
          ******************************************************
                          Listing of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                       DATE : 08-JAN-2002 16:07:40.85

Line
   1    -- ex_11vc   Examiner: verification = vc
   2
   3     --# main_program;
   4  procedure ex_11vc (n: in out integer; current: in out integer)
   5     --# derives current, n from n, current;
   6  is
   7
   8     previous: integer := 0;

!!! (  1)  Flow Error        : 54: The initialization at declaration of previ-
ous is
           ineffective.

   9     count   : integer := 1;

!!! (  2)  Flow Error        : 54: The initialization at declaration of count
is
           ineffective.

  10     x       : integer := 0;

!!! (  3)  Flow Error        : 54: The initialization at declaration of x is
           ineffective.

  11     n_i     : integer := 0;

!!! (  4)  Flow Error        : 54: The initialization at declaration of n_i is
           ineffective.

  12     Term    : integer := 0;

!!! (  5)  Flow Error        : 54: The initialization at declaration of Term
is
           ineffective.

  13     -- Terminierungsfunktion: Term = n-count
  14
  15     SubType IndexTy is integer range 0..23;
  16     Type FibTy is array(indexTy) of natural;
  17     fib: FibTy;
  18
  19
```

```
  20
  21  begin
  22     n_i := n;
         ^6
!!! (  6)  Flow Error        : 10: Ineffective statement.
```

```
  23     --# assert n >= 1 and n <= 23 and n = n_i;
  24
  25     previous := 0;
  26     current  := 1;
  27     count    := 1;
  28     fib(0)   := 0;
         ^7,8
!!! (  7)  Flow Error        : 23: Statement contains reference(s) to unde-
fined
           variable fib.
!!! (  8)  Flow Error        : 10: Assignment to fib is ineffective.

  29     fib(1)   := 1;
         ^9
!!! (  9)  Flow Error        : 10: Assignment to fib is ineffective.

  30     Term     := n-count;
         ^10
!!! ( 10)  Flow Error        : 10: Ineffective statement.

  31
  32     --# assert n >= 1 and n <= 23 and n = n_i and
  33     --#        previous = 0 and current = 1 and
  34     --#        count = 1 and
  35     --#        1 = fib(1) and 0 = fib(0) and
  36     --#        Term >= 0;
  37     WHILE count < n  LOOP
  38
  39        Term := n - count;
            ^11
!!! ( 11)  Flow Error        : 10: Ineffective statement.

  40        fib(count+1) := fib(count) + fib(count-1);
            ^12
!!! ( 12)  Flow Error        : 10: Assignment to fib is ineffective.

  41
  42        x := current;
  43        current := current + previous;
  44        previous := x;
  45        count := count+1;
  46
  47        --# assert count >= 1 and count <= n and n<=23 and
  48        --#        previous = fib(count-1) and
  49        --#        current = fib(count) and n = n_i and
  50        --#        current <= 32767 and
  51
  52        --#        fib(0) = 0 and fib(1) = 1 and
  53        --#        (for all K in indexTy range 2..count =>
  54        --#        (fib(K) = fib(k-2) + fib(k-1)) )   and
  55
  56        --#        Term > n-count and n-count >= 0;
  57     END LOOP;
```

```
  58
  59     --# assert current=fib(n) and current <= 32767 and n=n_i and n<=23
and
  60     --#        fib(0) = 0 and fib(1) = 1 and
  61     --#        (for all K in indexTy range 2..n =>
  62     --#        (fib(K) = fib(k-2) + fib(k-1)) );
  63
  64  end ex_11vc;

!!! ( 13)  Flow Error        : 31: The variable n is exported but not (inter-
nally)
           defined.
!!! ( 14)  Flow Error        : 35: Importation of the initial value of vari-
able
           current is ineffective.
!!! ( 15)  Flow Error        : 33: The variable n_i is neither referenced nor
           exported.
!!! ( 16)  Flow Error        : 33: The variable Term is neither referenced nor
           exported.
!!! ( 17)  Flow Error        : 50: The imported value of current is not used
in the
           derivation of n.
!!! ( 18)  Flow Error        : 50: The imported value of current is not used
in the
           derivation of current.

--End of file------------------------------------------------
```

Ex_11vc.siv

```
          ******************************************************
                      Semantic Analysis of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                           Demonstration Version
          ******************************************************


       CREATED 08-JAN-2002, 16:07:40  SIMPLIFIED 08-JAN-2002, 16:07:49
                (Simplified by SPADE Simplifier, Version 1.4)

                               procedure ex_11vc



For path(s) from start to assertion of line 23:

procedure_ex_11vc_1.
H1:    n >= integer__first .
H2:    n <= integer__last .
H3:    current >= integer__first .
H4:    current <= integer__last .
       ->
C1:    n >= 1 .
C2:    n <= 23 .
```

For path(s) from assertion of line 23 to assertion of line 32:

```
procedure_ex_11vc_2.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 32 to assertion of line 47:

procedure_ex_11vc_3.
H1:    n_i <= 23 .
H2:    1 = element(fib,[1]) .
H3:    0 = element(fib,[0]) .
H4:    term >= 0 .
H5:    1 < n_i .
       ->
C1:    1 = element(fib,[1]) + element(fib,[0]) .
C2:    (for_all)(k_ : integer,k_ >= 2 and k_ <= 2 -> ele-
ment(update(fib,[2],element(fib,[1]) + element(fib,[0])),[k_]) =
                     element(update(fib,[2],element(fib,[1]) + ele-
ment(fib,[0])),[k_ - 2]) +
                     element(update(fib,[2],element(fib,[1]) + ele-
ment(fib,[0])),[k_ - 1])) .
```

true

```
For path(s) from assertion of line 47 to assertion of line 47:

procedure_ex_11vc_4.
H1:    count >= 1 .
H2:    n_i <= 23 .
H3:    element(fib,[count]) <= 32767 .
H4:    element(fib,[0]) = 0 .
H5:    element(fib,[1]) = 1 .
H6:    (for_all)(k_ : integer,k_ >= 2 and k_ <= count -> element(fib,[k_]) =
element(fib,[k_ - 2]) + element(fib,[k_ - 1])) .
H7:    term > n_i - count .
H8:    count < n_i .
       ->
C1:    element(fib,[count]) + element(fib,[count - 1]) <= 32767 .
C2:    element(update(fib,[count + 1],element(fib,[count]) + ele-
ment(fib,[count - 1])),[0]) = 0 .
C3:    element(update(fib,[count + 1],element(fib,[count]) + ele-
ment(fib,[count - 1])),[1]) = 1 .
C4:    (for_all)(k_ : integer,k_ >= 2 and k_ <= count + 1 -> ele-
ment(update(fib,[count + 1],element(fib,[count]) + element(fib,[count -
1])),[k_]) =
                     element(update(fib,[count + 1],element(fib,[count]) +
element(fib,[count - 1])),[k_ - 2]) +
                     element(update(fib,[count + 1],element(fib,[count]) +
element(fib,[count - 1])),[k_ - 1])) .
```

true

```
For path(s) from assertion of line 32 to assertion of line 59:

procedure_ex_11vc_5.
H1:    n_i >= 1 .
H2:    1 = element(fib,[1]) .
H3:    0 = element(fib,[0]) .
H4:    term >= 0 .
H5:    n_i <= 1 .
       ->
C1:    1 = element(fib,[n_i]) .
```

```
C2:    (for_all)(k_ : integer,k_ >= 2 and k_ <= n_i -> element(fib,[k_]) =
element(fib,[k_ - 2]) + element(fib,[k_ - 1])) .


For path(s) from assertion of line 47 to assertion of line 59:

procedure_ex_11vc_6.
*** true .          /* all conclusions proved */
```

```
For path(s) from assertion of line 59 to finish:

procedure_ex_11vc_7.
*** true .          /* all conclusions proved */
```

Ex_11:	7 VCs gen.	5 VC rel.	2 of rel. VCs proved	checked 2002.Jan.08

Example 12

Ex_12vc.ada

```
  -- ex_1vc2   Examiner: verification = vc

   --# main_program;
procedure ex_12vc (n: in out integer; sum: in out integer)
   --# derives sum, n from n, sum;
is

   x : integer :=0;

begin
   --# assert n>0;
   x := 0;
   sum := 0;

   WHILE x < n LOOP
      x := x+1 ;
      sum := sum + x;
      --# assert sum = x*(x+1)/2 and x <= n;
   END LOOP;

   --# assert sum = n*(n+1)/2;
end ex_12vc;
```

Ex_12vc.lst

```
          ******************************************************
                          Listing of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                        DATE : 04-JAN-2002 08:49:18.40
```

```
Line
   1    -- ex_1vc2   Examiner: verification = vc
   2
   3     --# main_program;
   4  procedure ex_12vc (n: in out integer; sum: in out integer)
   5     --# derives sum, n from n, sum;
   6  is
   7
   8     x : integer :=0;

!!! (  1)  Flow Error        : 54: The initialization at declaration of x is
           ineffective.

   9
  10  begin
  11     --# assert n>0;
  12     x := 0;
  13     sum := 0;
  14
  15     WHILE x < n LOOP
  16        x := x+1 ;
  17        sum := sum + x;
  18        --# assert sum = x*(x+1)/2 and x <= n;
  19     END LOOP;
  20
  21     --# assert sum = n*(n+1)/2;
  22  end ex_12vc;

!!! (  2)  Flow Error        : 31: The variable n is exported but not (inter-
nally)
           defined.
!!! (  3)  Flow Error        : 35: Importation of the initial value of vari-
able sum
           is ineffective.
!!! (  4)  Flow Error        : 50: The imported value of sum is not used in
the
           derivation of n.
!!! (  5)  Flow Error        : 50: The imported value of sum is not used in
the
           derivation of sum.

--End of file------------------------------------------------
```

Ex_12vc.siv

```
          ******************************************************
                      Semantic Analysis of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                           Demonstration Version
          ******************************************************


       CREATED 04-JAN-2002, 08:49:18  SIMPLIFIED 04-JAN-2002, 08:49:25
                (Simplified by SPADE Simplifier, Version 1.4)

                               procedure ex_12vc
```

```
For path(s) from start to assertion of line 11:

procedure_ex_12vc_1.
H1:    n >= integer__first .
H2:    n <= integer__last .
H3:    sum >= integer__first .
H4:    sum <= integer__last .
       ->
C1:    n > 0 .
```

```
For path(s) from assertion of line 11 to assertion of line 18:

procedure_ex_12vc_2.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 18 to assertion of line 18:

procedure_ex_12vc_3.
H1:    x < n .
       ->
C1:    x * (x + 1) div 2 + (x + 1) = (x + 1) * (x + 1 + 1) div 2 .


For path(s) from assertion of line 11 to assertion of line 21:

procedure_ex_12vc_4.
*** true .   /* contradiction within hypotheses. */



For path(s) from assertion of line 18 to assertion of line 21:

procedure_ex_12vc_5.
*** true .          /* all conclusions proved */
```

true

```
For path(s) from assertion of line 21 to finish:

procedure_ex_12vc_6.
*** true .          /* all conclusions proved */
```

Ex_12:	6 VCs gen.	4 VC rel.	3 of rel. VCs proved	checked 2002.Jan.04

Example 13

Ex_13vc.ada

```
  -- ex_13vc   Examiner: verification = vc

   --# main_program;
```

```
procedure ex_13vc (N: in out integer; summe: in out integer)
   --# derives N, summe from N, summe;
is

   n_i : integer := 0;
   x : integer := 0;
   term_old : integer := 0;
   -- Terminierungsfunktion = n-x

begin
   n_i := n;
   --# assert 0 < n and n = n_i;
   x := 0;
   summe := 0;
   term_old := n-x;

   --# assert n > 0 and x = 0 and summe = 0 and n = n_i   and term_old >= 0;
   WHILE x < n LOOP
      term_old := n-x; -- alter Wert der Terminierungsfunktion

      x := x+1;
      summe := summe + x;

      --# assert summe = x*(x+1)/2 and x <= n and 0 < n and n=n_i and
      --#        term_old > n - x and n - x >= 0;
   END LOOP;

   --# assert summe = n*(n+1)/2 and n = n_i;

end ex_13vc;
```

Ex_13vc.lst

```
          ******************************************************
                          Listing of SPARK Text
      SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                       DATE : 02-JAN-2002 08:23:42.89

Line
   1    -- ex_13vc   Examiner: verification = vc
   2
   3     --# main_program;
   4  procedure ex_13vc (N: in out integer; summe: in out integer)
   5     --# derives N, summe from N, summe;
   6  is
   7
   8     n_i : integer := 0;

!!! (  1)  Flow Error        : 54: The initialization at declaration of n_i is
           ineffective.

   9     x : integer := 0;

!!! (  2)  Flow Error        : 54: The initialization at declaration of x is
```

```
           ineffective.

  10     term_old : integer := 0;

!!! (  3)  Flow Error        : 54: The initialization at declaration of
term_old is
           ineffective.

  11     -- Terminierungsfunktion = n-x
  12
  13  begin
  14     n_i := n;
         ^4
!!! (  4)  Flow Error        : 10: Ineffective statement.

  15     --# assert 0 < n and n = n_i;
  16     x := 0;
  17     summe := 0;
  18     term_old := n-x;
         ^5
!!! (  5)  Flow Error        : 10: Ineffective statement.

  19
  20     --# assert n > 0 and x = 0 and summe = 0 and n = n_i   and term_old
>= 0;
  21     WHILE x < n LOOP
  22        term_old := n-x; -- alter Wert der Terminierungsfunktion
            ^6
!!! (  6)  Flow Error        : 10: Ineffective statement.

  23
  24        x := x+1;
  25        summe := summe + x;
  26
  27        --# assert summe = x*(x+1)/2 and x <= n and 0 < n and n=n_i and
  28        --#        term_old > n - x and n - x >= 0;
  29     END LOOP;
  30
  31     --# assert summe = n*(n+1)/2 and n = n_i;
  32
  33  end ex_13vc;

!!! (  7)  Flow Error        : 31: The variable N is exported but not (inter-
nally)
           defined.
!!! (  8)  Flow Error        : 35: Importation of the initial value of vari-
able
           summe is ineffective.
!!! (  9)  Flow Error        : 33: The variable n_i is neither referenced nor
           exported.
!!! ( 10)  Flow Error        : 33: The variable term_old is neither referenced
nor
           exported.
!!! ( 11)  Flow Error        : 50: The imported value of summe is not used in
the
           derivation of N.
!!! ( 12)  Flow Error        : 50: The imported value of summe is not used in
the
           derivation of summe.
```

```
--End of file------------------------------------------------
```

Ex_13vc.siv

```
          ******************************************************
                       Semantic Analysis of SPARK Text
             SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                            Demonstration Version
          ******************************************************


             CREATED 02-JAN-2002, 08:23:42  SIMPLIFIED 02-JAN-2002, 08:23:49
                      (Simplified by SPADE Simplifier, Version 1.4)

                                 procedure ex_13vc



For path(s) from start to assertion of line 15:

procedure_ex_13vc_1.
H1:    n >= integer__first .
H2:    n <= integer__last .
H3:    summe >= integer__first .
H4:    summe <= integer__last .
       ->
C1:    0 < n .
```

```
For path(s) from assertion of line 15 to assertion of line 20:

procedure_ex_13vc_2.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 20 to assertion of line 27:

procedure_ex_13vc_3.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 27 to assertion of line 27:

procedure_ex_13vc_4.
H1:    0 < n_i .
H2:    term_old > n_i - x .
H3:    x < n_i .
       ->
C1:    x * (x + 1) div 2 + (x + 1) = (x + 1) * (x + 1 + 1) div 2 .
```

true

```
For path(s) from assertion of line 20 to assertion of line 31:

procedure_ex_13vc_5.
*** true .   /* contradiction within hypotheses. */
```

```
For path(s) from assertion of line 27 to assertion of line 31:

procedure_ex_13vc_6.
*** true .          /* all conclusions proved */
```

```
For path(s) from assertion of line 31 to finish:

procedure_ex_13vc_7.
*** true .          /* all conclusions proved */
```

Ex_13:	7 VCs gen.	5 VC rel.	4 of rel. VCs proved	checked 2002.Jan.03

Example 14

Ex_14vc.ada

```
  -- ex_14vc   Examiner: verification = vc

  -- Fehler bei den für FPP generierten Ada sourcen.
  -- Es wird n_i als boolsche variable eingesetzt und nicht zur verifikation
von n konstant.
  -- Bsp: summe = --!pre:  n <= 10 and n_i;

   --# main_program;
procedure ex_14vc (n: in out integer; summe: in out integer)
   --# derives n, summe from n, summe;
is

   x: integer := 0;
   n_i : integer := 0;

   term_old : integer := 0; -- alter Wert Terminierungsfunktion

begin

  n_i := n;
```

```
   --# assert 0 < n and n <= 10 and n = n_i;

   x := 0;
   summe := 0;


   --# assert  n > 0 and n <= 10 and x = 0 and summe = 0 and n_i = n;

   WHILE x < n LOOP

      term_old := n-x;

      x := x+1;
      summe := summe + x;
```

```
      --# assert summe = x*(x+1)/2 and x <= n  and 0 < n and n <= 10 and n_i =
n   and
      --#        n - x < term_old and n - x >= 0;  -- Terminierungsfkt
   END LOOP;

   --# assert summe = n*(n+1)/2 and summe <= 60 and n_i = n;
end ex_14vc;
```

Ex_14vc.lst

```
          ******************************************************
                          Listing of SPARK Text
      SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                       DATE : 02-JAN-2002 08:31:01.97

Line
   1    -- ex_14vc   Examiner: verification = vc
   2
   3    -- Fehler bei den für FPP generierten Ada sourcen.
   4    -- Es wird n_i als boolsche variable eingesetzt und nicht zur verifi-
kation von n konstant.
   5    -- Bsp: summe = --!pre:  n <= 10 and n_i;
   6
   7     --# main_program;
   8  procedure ex_14vc (n: in out integer; summe: in out integer)
   9     --# derives n, summe from n, summe;
  10  is
  11
  12     x: integer := 0;

!!! (  1)  Flow Error        : 54: The initialization at declaration of x is
           ineffective.

  13     n_i : integer := 0;

!!! (  2)  Flow Error        : 54: The initialization at declaration of n_i is
           ineffective.

  14
  15     term_old : integer := 0; -- alter Wert Terminierungsfunktion

!!! (  3)  Flow Error        : 54: The initialization at declaration of
term_old is
           ineffective.

  16
  17  begin
  18
  19    n_i := n;
        ^4
!!! (  4)  Flow Error        : 10: Ineffective statement.

  20
  21     --# assert 0 < n and n <= 10 and n = n_i;
  22
```

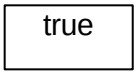
```
  23     x := 0;
  24     summe := 0;
  25
  26
  27     --# assert  n > 0 and n <= 10 and x = 0 and summe = 0 and n_i = n;
  28
  29     WHILE x < n LOOP
  30
  31        term_old := n-x;
            ^5
!!! (  5)  Flow Error        : 10: Ineffective statement.

  32
  33        x := x+1;
  34        summe := summe + x;
  35
  36        --# assert summe = x*(x+1)/2 and x <= n  and 0 < n and n <= 10 and
n_i = n   and
  37        --#        n - x < term_old and n - x >= 0;  -- Terminierungsfkt
  38     END LOOP;
  39
  40     --# assert summe = n*(n+1)/2 and summe <= 60 and n_i = n;
  41  end ex_14vc;

!!! (  6)  Flow Error        : 31: The variable n is exported but not (inter-
nally)
            defined.
!!! (  7)  Flow Error        : 35: Importation of the initial value of vari-
able
            summe is ineffective.
!!! (  8)  Flow Error        : 33: The variable n_i is neither referenced nor
            exported.
!!! (  9)  Flow Error        : 33: The variable term_old is neither referenced
nor
            exported.
!!! ( 10)  Flow Error        : 50: The imported value of summe is not used in
the
            derivation of n.
!!! ( 11)  Flow Error        : 50: The imported value of summe is not used in
the
            derivation of summe.

--End of file------------------------------------------------
```

Ex_14vc.siv

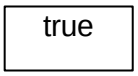
```
          ******************************************************
                      Semantic Analysis of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                           Demonstration Version
          ******************************************************


       CREATED 02-JAN-2002, 08:31:01  SIMPLIFIED 02-JAN-2002, 08:31:18
                (Simplified by SPADE Simplifier, Version 1.4)

                               procedure ex_14vc
```

```
For path(s) from start to assertion of line 21:

procedure_ex_14vc_1.
H1:    n >= integer__first .
H2:    n <= integer__last .
H3:    summe >= integer__first .
H4:    summe <= integer__last .
       ->
C1:    0 < n .
C2:    n <= 10 .
```

```
For path(s) from assertion of line 21 to assertion of line 27:

procedure_ex_14vc_2.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 27 to assertion of line 36:

procedure_ex_14vc_3.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 36 to assertion of line 36:

procedure_ex_14vc_4.
H1:    0 < n .
H2:    n <= 10 .
H3:    n - x < term_old .
H4:    x < n .
       ->
C1:    x * (x + 1) div 2 + (x + 1) = (x + 1) * (x + 1 + 1) div 2 .
```




```
For path(s) from assertion of line 27 to assertion of line 40:

procedure_ex_14vc_5.
*** true .   /* contradiction within hypotheses. */


For path(s) from assertion of line 36 to assertion of line 40:

procedure_ex_14vc_6.
H1:    0 < n .
H2:    n <= 10 .
H3:    n - n < term_old .
       ->
C1:    n * (n + 1) div 2 <= 60 .
```




```
For path(s) from assertion of line 40 to finish:

procedure_ex_14vc_7.
*** true .          /* all conclusions proved */
```

Ex_14: 2002.Jan.03	7 VCs gen.	5 VC rel.	3 of rel. VCs proved	checked

Example 15

Ex_15vc.ada

```
   -- ex_15vc   Examiner: verification = vc

    --# main_program;
procedure ex_15vc (x: in out integer;
                   A: in out integer; z: in out integer;
                   s: in out integer; Empty: in out integer)
    --# derives x, A, z, s, Empty from x, A, z, s, Empty;
is

   A_i: integer := 0;
   Empty_i: integer := 0;

   function P(a: in integer) return boolean is
      Ret_Value: boolean;
   begin
      return Ret_Value;
   end P;

   function r(a: in integer) return integer is
      ret_Value: integer;
   begin
      return Ret_Value;
   end r;

   function g(a: in integer) return integer is
      Ret_Value: integer;
   begin
      return Ret_Value;
   end g;

   function h(a: in integer; b: in integer) return integer is
      ret_Value: integer;
   begin
      return ret_Value;
   end h;

   function push(a: in integer; b: in integer) return integer is
      Ret_value: integer;
   begin
      return ret_Value;
   end push;

   function pop(a: in integer) return integer is
      Ret_value: integer;
   begin
      return Ret_Value;
   end pop;

   function top(a: in integer) return integer is
```

```
      Ret_Value: integer;
   begin
      return ret_Value;
   end top;

   --# function P_virt(a: in integer; b: in integer) return integer;
   --# function f(a: in integer) return integer;

begin

   A_i := A;
   Empty_i := Empty;
   --# assert x = A and A = A_i and Empty = Empty_i;
   s := Empty ;
   WHILE not(P(x)) LOOP
      s := push(x,s) ;
      x := r(x);
      --# assert P_virt(x,s) = f(A) and A = A_i and Empty = Empty_i;
   END loop;

   z := g(x) ;

   WHILE s /= Empty LOOP
      z := h(z,top(s)) ;
      s := pop(s);
      --# assert p_virt(z,s) = f(A) and A = A_i and Empty = Empty_i;
   END loop;

   --# assert z = f(A) and A = A_i and Empty = Empty_i;
end ex_15vc;
```

Ex_15vc.lst

```
            ******************************************************
                           Listing of SPARK Text
      SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                           Demonstration Version
          ******************************************************


                        DATE : 12-JAN-2002 09:08:17.37

Line
   1    -- ex_15vc   Examiner: verification = vc
   2
   3     --# main_program;
   4  procedure ex_15vc (x: in out integer;
   5                     A: in out integer; z: in out integer;
   6                     s: in out integer; Empty: in out integer)
   7     --# derives x, A, z, s, Empty from x, A, z, s, Empty;
   8  is
   9
  10     A_i: integer := 0;

 !!! (  1)  Flow Error         : 54: The initialization at declaration of A_i
is
            ineffective.
```

```
   11     Empty_i: integer := 0;

 !!! (   2)   Flow Error          : 54: The initialization at declaration of
Empty_i is
            ineffective.

   12
   13     function P(a: in integer) return boolean is
   14        Ret_Value: boolean;
   15     begin
   16        return Ret_Value;
                    ^3
 !!! (  3)  Flow Error        : 20: Expression contains reference(s) to unde-
fined
            variable Ret_Value.

   17     end P;

 !!! (  4)  Flow Error        : 50: The imported value of A is not used in the
            derivation of the function value.
 ??? (  5)  Warning            :602: The undefined initial value of Ret_Value
may be
            used in the derivation of the function value.
 !!! (  6)  Flow Error        : 30: The variable A is imported but neither
            referenced nor exported.
 !!! (  7)  Flow Error        : 32: The variable Ret_Value is neither imported
nor
            defined.

   18
   19     function r(a: in integer) return integer is
   20        ret_Value: integer;
   21     begin
   22        return Ret_Value;
                    ^8
 !!! (  8)  Flow Error        : 20: Expression contains reference(s) to unde-
fined
            variable Ret_Value.

   23     end r;

 !!! (  9)  Flow Error        : 50: The imported value of A is not used in the
            derivation of the function value.
 ??? ( 10)  Warning            :602: The undefined initial value of Ret_Value
may be
            used in the derivation of the function value.
 !!! ( 11)  Flow Error        : 30: The variable A is imported but neither
            referenced nor exported.
 !!! ( 12)  Flow Error        : 32: The variable Ret_Value is neither imported
nor
            defined.

   24
   25     function g(a: in integer) return integer is
   26        Ret_Value: integer;
   27     begin
   28        return Ret_Value;
                    ^13
```

```
 !!! ( 13)  Flow Error        : 20: Expression contains reference(s) to unde-
fined
            variable Ret_Value.

   29     end g;

 !!! ( 14)  Flow Error        : 50: The imported value of A is not used in the
            derivation of the function value.
 ??? ( 15)  Warning           :602: The undefined initial value of Ret_Value
may be
            used in the derivation of the function value.
 !!! ( 16)  Flow Error        : 30: The variable A is imported but neither
            referenced nor exported.
 !!! ( 17)  Flow Error        : 32: The variable Ret_Value is neither imported
nor
            defined.

   30
   31     function h(a: in integer; b: in integer) return integer is
   32        ret_Value: integer;
   33     begin
   34        return ret_Value;
                    ^18
 !!! ( 18)  Flow Error        : 20: Expression contains reference(s) to unde-
fined
            variable Ret_Value.

   35     end h;

 !!! ( 19)  Flow Error        : 32: The variable Ret_Value is neither imported
nor
            defined.
 !!! ( 20)  Flow Error        : 50: The imported value of A is not used in the
            derivation of the function value.
 !!! ( 21)  Flow Error        : 50: The imported value of b is not used in the
            derivation of the function value.
 ??? ( 22)  Warning           :602: The undefined initial value of Ret_Value
may be
            used in the derivation of the function value.
 !!! ( 23)  Flow Error        : 30: The variable A is imported but neither
            referenced nor exported.
 !!! ( 24)  Flow Error        : 30: The variable b is imported but neither
            referenced nor exported.

   36
   37     function push(a: in integer; b: in integer) return integer is
   38        Ret_value: integer;
   39     begin
   40        return ret_Value;
                    ^25
 !!! ( 25)  Flow Error        : 20: Expression contains reference(s) to unde-
fined
            variable Ret_Value.

   41     end push;

 !!! ( 26)  Flow Error        : 32: The variable Ret_Value is neither imported
nor
            defined.
```

```
 !!! ( 27)  Flow Error        : 50: The imported value of A is not used in the
            derivation of the function value.
 !!! ( 28)  Flow Error        : 50: The imported value of b is not used in the
            derivation of the function value.
 ??? ( 29)  Warning            :602: The undefined initial value of Ret_Value
may be
            used in the derivation of the function value.
 !!! ( 30)  Flow Error        : 30: The variable A is imported but neither
            referenced nor exported.
 !!! ( 31)  Flow Error        : 30: The variable b is imported but neither
            referenced nor exported.

   42
   43     function pop(a: in integer) return integer is
   44        Ret_value: integer;
   45     begin
   46        return Ret_Value;
                    ^32
 !!! ( 32)  Flow Error         : 20: Expression contains reference(s) to unde-
fined
            variable Ret_Value.

   47     end pop;

 !!! ( 33)  Flow Error        : 50: The imported value of A is not used in the
            derivation of the function value.
 ??? ( 34)  Warning            :602: The undefined initial value of Ret_Value
may be
            used in the derivation of the function value.
 !!! ( 35)  Flow Error        : 30: The variable A is imported but neither
            referenced nor exported.
 !!! ( 36)  Flow Error        : 32: The variable Ret_Value is neither imported
nor
            defined.

   48
   49     function top(a: in integer) return integer is
   50        Ret_Value: integer;
   51     begin
   52        return ret_Value;
                    ^37
 !!! ( 37)  Flow Error         : 20: Expression contains reference(s) to unde-
fined
            variable Ret_Value.

   53     end top;

 !!! ( 38)  Flow Error        : 50: The imported value of A is not used in the
            derivation of the function value.
 ??? ( 39)  Warning            :602: The undefined initial value of Ret_Value
may be
            used in the derivation of the function value.
 !!! ( 40)  Flow Error        : 30: The variable A is imported but neither
            referenced nor exported.
 !!! ( 41)  Flow Error        : 32: The variable Ret_Value is neither imported
nor
            defined.

   54
```

```
   55     --# function P_virt(a: in integer; b: in integer) return integer;
   56     --# function f(a: in integer) return integer;
   57
   58  begin
   59
   60     A_i := A;
          ^42
 !!! ( 42)  Flow Error        : 10: Ineffective statement.

   61     Empty_i := Empty;
          ^43
 !!! ( 43)  Flow Error        : 10: Ineffective statement.

   62     --# assert x = A and A = A_i and Empty = Empty_i;
   63     s := Empty ;
   64     WHILE not(P(x)) LOOP
   65        s := push(x,s) ;
   66        x := r(x);
   67        --# assert P_virt(x,s) = f(A) and A = A_i and Empty = Empty_i;
   68     END loop;
   69
   70     z := g(x) ;
   71
   72     WHILE s /= Empty LOOP
   73        z := h(z,top(s)) ;
   74        s := pop(s);
   75        --# assert p_virt(z,s) = f(A) and A = A_i and Empty = Empty_i;
   76     END loop;
   77
   78     --# assert z = f(A) and A = A_i and Empty = Empty_i;
   79  end ex_15vc;

 !!! ( 44)  Flow Error        : 31: The variable A is exported but not (inter-
nally)
            defined.
 !!! ( 45)  Flow Error        : 35: Importation of the initial value of vari-
able z
            is ineffective.
 !!! ( 46)  Flow Error        : 35: Importation of the initial value of vari-
able s
            is ineffective.
 !!! ( 47)  Flow Error        : 31: The variable Empty is exported but not
            (internally) defined.
 !!! ( 48)  Flow Error        : 33: The variable A_i is neither referenced nor
            exported.
 !!! ( 49)  Flow Error        : 33: The variable Empty_i is neither referenced
nor
            exported.
 !!! ( 50)  Flow Error        : 50: The imported value of A is not used in the
            derivation of x.
 !!! ( 51)  Flow Error        : 50: The imported value of z is not used in the
            derivation of x.
 !!! ( 52)  Flow Error        : 50: The imported value of s is not used in the
            derivation of x.
 !!! ( 53)  Flow Error        : 50: The imported value of Empty is not used in
the
            derivation of x.
 !!! ( 54)  Flow Error        : 50: The imported value of x is not used in the
            derivation of A.
```

```
 !!! ( 55)  Flow Error        : 50: The imported value of z is not used in the
            derivation of A.
 !!! ( 56)  Flow Error        : 50: The imported value of s is not used in the
            derivation of A.
 !!! ( 57)  Flow Error        : 50: The imported value of Empty is not used in
the
            derivation of A.
 !!! ( 58)  Flow Error        : 50: The imported value of A is not used in the
            derivation of z.
 !!! ( 59)  Flow Error        : 50: The imported value of z is not used in the
            derivation of z.
 !!! ( 60)  Flow Error        : 50: The imported value of s is not used in the
            derivation of z.
 !!! ( 61)  Flow Error        : 50: The imported value of A is not used in the
            derivation of s.
 !!! ( 62)  Flow Error        : 50: The imported value of z is not used in the
            derivation of s.
 !!! ( 63)  Flow Error        : 50: The imported value of s is not used in the
            derivation of s.
 !!! ( 64)  Flow Error        : 50: The imported value of x is not used in the
            derivation of Empty.
 !!! ( 65)  Flow Error        : 50: The imported value of A is not used in the
            derivation of Empty.
 !!! ( 66)  Flow Error        : 50: The imported value of z is not used in the
            derivation of Empty.
 !!! ( 67)  Flow Error        : 50: The imported value of s is not used in the
            derivation of Empty.


 --End of file------------------------------------------------
```

Ex_15vc.siv

```
          ******************************************************
                      Semantic Analysis of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


       CREATED 12-JAN-2002, 09:08:16  SIMPLIFIED 12-JAN-2002, 09:08:36
                (Simplified by SPADE Simplifier, Version 1.4)

                               procedure ex_15vc



 For path(s) from start to assertion of line 62:

 procedure_ex_15vc_1.
 H1:    x >= integer__first .
 H2:    x <= integer__last .
 H3:    a >= integer__first .
 H4:    a <= integer__last .
 H5:    z >= integer__first .
 H6:    z <= integer__last .
 H7:    s >= integer__first .
```

```
H8:    s <= integer__last .
H9:    empty >= integer__first .
H10:   empty <= integer__last .
       ->
C1:    x = a .
```

```
For path(s) from assertion of line 62 to assertion of line 67:

procedure_ex_15vc_2.
H1:    not p(a_i) .
       ->
C1:    p_virt(r(a_i),push(a_i,empty_i)) = f(a_i) .


For path(s) from assertion of line 67 to assertion of line 67:

procedure_ex_15vc_3.
H1:    p_virt(x,s) = f(a_i) .
H2:    not p(x) .
       ->
C1:    p_virt(r(x),push(x,s)) = f(a_i) .


For path(s) from assertion of line 62 to assertion of line 75:

procedure_ex_15vc_4.
*** true .   /* contradiction within hypotheses. */



For path(s) from assertion of line 67 to assertion of line 75:

procedure_ex_15vc_5.
H1:    p_virt(x,s) = f(a_i) .
H2:    p(x) .
H3:    s <> empty_i .
       ->
C1:    p_virt(h(g(x),top(s)),pop(s)) = f(a_i) .


For path(s) from assertion of line 75 to assertion of line 75:

procedure_ex_15vc_6.
H1:    p_virt(z,s) = f(a_i) .
H2:    s <> empty_i .
       ->
C1:    p_virt(h(z,top(s)),pop(s)) = f(a_i) .


For path(s) from assertion of line 62 to assertion of line 78:

procedure_ex_15vc_7.
H1:    p(a_i) .
       ->
C1:    g(a_i) = f(a_i) .


For path(s) from assertion of line 67 to assertion of line 78:
```

```
procedure_ex_15vc_8.
H1:    p_virt(x,empty_i) = f(a_i) .
H2:    p(x) .
       ->
C1:    g(x) = f(a_i) .


For path(s) from assertion of line 75 to assertion of line 78:

procedure_ex_15vc_9.
H1:    p_virt(z,empty_i) = f(a_i) .
       ->
C1:    z = f(a_i) .
```

```
For path(s) from assertion of line 78 to finish:

procedure_ex_15vc_10.
*** true .          /* all conclusions proved */
```

Ex_15:	10 VCs gen.	8 VC rel.	8 of rel. VCs correct	JW 2002.01.12

Example 16

Ex_16vc.ada

```
  -- ex_16vc   Examiner: verification = vc

with IntArray;

   --# main_program;
procedure ex_16vc (b: in out IntArray.IntArrayType;
                   X: in out integer; Ind: in out integer)
   --# derives X, B, ind from X, B, ind;
is
   term: integer := 0;
   len : integer := 0;
   x_i : integer := 0;
   len_i : integer :=0;

begin

   ind  := 1;
   len  := b'last;
   len_i := len;
   x_i  := x;
   term := len+1-ind;
   --# assert ind=1 and len>=1 and
   --#        (for some j in natural range 1..len => (b(j) = x))
```

```
   --#        and x = x_i and len = len_i;
   WHILE b(ind) /= x LOOP
      term := len+1-ind;
      ind := ind+1;

   --# assert 1<=ind and ind <=len and
   --#        not((for some j in natural range 1..ind-1 => (b(j) = x)))
   --#        and (for some j in natural range 1..len => (b(j) = x))
   --#        and x = x_i and len = len_i
   --#        and term > len+1-ind and len+1-ind>=0; -- Terminierungsfkt

   END LOOP;

   --# assert 1<=ind and ind<=len and b(ind) = x
   --#        and not((for some j in natural range 1..ind-1 => (b(j)=x)))
   --#        and x = x_i and len = len_i;
end ex_16vc;
```

IntArray.ads

```
package IntArray is

   SubType IndexType is Natural range 1..100;
   Type IntArrayType is Array (IndexType) of Integer;

end IntArray;
```

Ex_16vc.lst

```
          ******************************************************
                           Listing of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                           Demonstration Version
          ******************************************************


                        DATE : 03-JAN-2002 10:38:21.21

 Line
    1    -- ex_16vc   Examiner: verification = vc
    2
    3  with IntArray;
           ^1
 --- (  1)  Warning            :  1: The identifier IntArray is either unde-
clared or
           not visible at this point.

    4
    5     --# main_program;
    6  procedure ex_16vc (b: in out IntArray.IntArrayType;
    7                     X: in out integer; Ind: in out integer)
    8     --# derives X, B, ind from X, B, ind;
```

```
    9  is
   10     term: integer := 0;

 !!! (  2)  Flow Error        : 54: The initialization at declaration of term
is
            ineffective.

   11     len : integer := 0;

 !!! (  3)  Flow Error        : 54: The initialization at declaration of len
is
            ineffective.

   12     x_i : integer := 0;

 !!! (  4)  Flow Error        : 54: The initialization at declaration of x_i
is
            ineffective.

   13     len_i : integer :=0;

 !!! (  5)  Flow Error        : 54: The initialization at declaration of len_i
is
            ineffective.

   14
   15  begin
   16
   17     ind  := 1;
   18     len  := b'last;
          ^6
 !!! (  6)  Flow Error        : 10: Ineffective statement.

   19     len_i := len;
          ^7
 !!! (  7)  Flow Error        : 10: Ineffective statement.

   20     x_i  := x;
          ^8
 !!! (  8)  Flow Error        : 10: Ineffective statement.

   21     term := len+1-ind;
          ^9
 !!! (  9)  Flow Error        : 10: Ineffective statement.
```

```
   22     --# assert ind=1 and len>=1 and
   23     --#        (for some j in natural range 1..len => (b(j) = x))
   24     --#        and x = x_i and len = len_i;
   25     WHILE b(ind) /= x LOOP
   26        term := len+1-ind;
```

```
             ^10
 !!! ( 10)  Flow Error        : 10: Ineffective statement.

   27        ind := ind+1;
   28
   29     --# assert 1<=ind and ind <=len and
   30     --#        not((for some j in natural range 1..ind-1 => (b(j) = x)))
   31     --#        and (for some j in natural range 1..len => (b(j) = x))
   32     --#        and x = x_i and len = len_i
   33        --#            and  term  >  len+1-ind  and  len+1-ind>=0;  --  Ter-
minierungsfkt
   34
   35     END LOOP;
   36
   37     --# assert 1<=ind and ind<=len and b(ind) = x
    38       --#           and  not((for  some  j  in  natural  range  1..ind-1 =>
(b(j)=x)))
   39     --#        and x = x_i and len = len_i;
   40  end ex_16vc;

 !!! ( 11)  Flow Error        : 31: The variable b is exported but not (inter-
nally)
            defined.
 !!! ( 12)  Flow Error        : 31: The variable X is exported but not (inter-
nally)
            defined.
 !!! ( 13)  Flow Error         : 35: Importation of the initial value of vari-
able Ind
            is ineffective.
 !!! ( 14)   Flow Error         : 33: The variable term is neither referenced
nor
            exported.
 !!! ( 15)  Flow Error        : 33: The variable x_i is neither referenced nor
            exported.
 !!! ( 16)   Flow Error         : 33: The variable len_i is neither referenced
nor
            exported.
 !!! ( 17)  Flow Error        : 50: The imported value of X is not used in the
            derivation of b.
 !!! ( 18)   Flow Error         : 50: The imported value of Ind is not used in
the
            derivation of b.
 !!! ( 19)  Flow Error        : 50: The imported value of b is not used in the
            derivation of X.
 !!! ( 20)   Flow Error         : 50: The imported value of Ind is not used in
the
            derivation of X.
 !!! ( 21)   Flow Error         : 50: The imported value of Ind is not used in
the
            derivation of Ind.
```

```
 --End of file-----------------------------------------------------
```

IntArray.lst

```
           ******************************************************
                            Listing of SPARK Text
        SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                            Demonstration Version
           ******************************************************


                         DATE : 03-JAN-2002 10:38:21.10

 Line
    1  package IntArray is
    2
    3     SubType IndexType is Natural range 1..100;
    4     Type IntArrayType is Array (IndexType) of Integer;
    5
    6  end IntArray;


 --End of file-----------------------------------------------------
```

Ex_16vc.siv

```
           ******************************************************
                        Semantic Analysis of SPARK Text
        SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                            Demonstration Version
           ******************************************************


        CREATED 03-JAN-2002, 10:38:21  SIMPLIFIED 03-JAN-2002, 10:38:28
                  (Simplified by SPADE Simplifier, Version 1.4)

                               procedure ex_16vc




 For path(s) from start to assertion of line 22:

 procedure_ex_16vc_1.
 H1:       (for_all)(i___1 : integer,i___1 >= 1 and i___1 <= 100 -> ele-
ment(b,[i___1]) >= integer__first and element(b,[i___1]) <= integer__last) .
 H2:    x >= integer__first .
 H3:    x <= integer__last .
 H4:    ind >= integer__first .
 H5:    ind <= integer__last .
        ->
 C1:    (for_some)(j_ : integer,j_ >= 1 and j_ <= 100 and element(b,[j_]) = x)
.
```

```
 For path(s) from assertion of line 22 to assertion of line 29:

 procedure_ex_16vc_2.
 H1:    len_i >= 1 .
 H2:    (for_some)(j_ : integer,j_ >= 1 and j_ <= len_i and element(b,[j_]) =
x_i) .
 H3:    element(b,[1]) <> x_i .                          true
        ->
 C1:    2 <= len_i .
 C2:    not (for_some)(j_ : integer,j_ >= 1 and j_ <= 1 and element(b,[j_]) =
x_i) .


 For path(s) from assertion of line 29 to assertion of line 29:

 procedure_ex_16vc_3.
 H1:    1 <= ind .
 H2:    ind <= len_i .
 H3:    (for_all)(j_ : integer,j_ < 1 or ind - 1 < j_ or element(b,[j_]) <>
x_i) .
 H4:    (for_some)(j_ : integer,j_ >= 1 and j_ <= len_i and element(b,[j_]) =
x_i) .
 H5:    term > len_i + 1 - ind .
 H6:    element(b,[ind]) <> x_i .                        true
        ->
 C1:    ind + 1 <= len_i .
 C2:    not (for_some)(j_ : integer,j_ >= 1 and j_ <= ind and element(b,[j_])
= x_i) .


 For path(s) from assertion of line 22 to assertion of line 37:

 procedure_ex_16vc_4.
 H1:    len_i >= 1 .
 H2:    (for_some)(j_ : integer,j_ >= 1 and j_ <= len_i and element(b,[j_]) =
element(b,[1])) .                                       true
        ->
 C1:    not (for_some)(j_ : integer,j_ >= 1 and j_ <= 0 and element(b,[j_]) =
element(b,[1])) .


 For path(s) from assertion of line 29 to assertion of line 37:

 procedure_ex_16vc_5.
 *** true .          /* all conclusions proved */


 For path(s) from assertion of line 37 to finish:
```

```
procedure_ex_16vc_6.
*** true .          /* all conclusions proved */
```

Ex_16:	6 VCs gen.	4 VC rel.	1 of rel. VCs proved	checked 2002.Jan.04

Example 17

Ex_17vc.ada

```
   -- ex_17vc   Examiner: verification = vc

    --# main_program;
procedure ex_17vc (A: in out integer; B: in out integer; P: in out integer)
    --# derives A, B, P from A, B, P;
is

    A_I: integer := 0;
    B_I: integer := 0;

begin

    --# assert a >= 0 and b >= 0 and a = a_i and b = b_i;
    p := 0;

    --# assert p = 0 and a >= 0 and b >= 0 and a = a_i and b = b_i;
    FOR i IN integer RANGE 1..b LOOP

       --# assert p = (i-1)*a and a >= 0 and b > 0 and a = a_i and b = b_i;
       FOR j IN integer RANGE 1..a LOOP

          p := p+1;
          --# assert p = (i-1)*a+j and a > 0 and b > 0 and a = a_i and b =
b_i;

       END LOOP;
       --# assert p=i*a and a >= 0 and b > 0 and a = a_i and b = b_i;

    END LOOP;

    --# assert p=b*a and a >= 0 and b >= 0 and a = a_i and b = b_i;

end ex_17vc;
```

Ex_17vc.lst

```
          ******************************************************
                              Listing of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
```

```
                          Demonstration Version
          ***************************************************


                       DATE : 02-JAN-2002 08:41:29.32

 Line
    1    -- ex_17vc   Examiner: verification = vc
    2
    3     --# main_program;
    4  procedure ex_17vc (A: in out integer; B: in out integer; P: in out in-
teger)
    5     --# derives A, B, P from A, B, P;
    6  is
    7
    8     A_I: integer := 0;

 !!! (  1)  Flow Error         : 54: The initialization at declaration of A_I
is
            ineffective.

    9     B_I: integer := 0;

 !!! (  2)  Flow Error         : 54: The initialization at declaration of B_I
is
            ineffective.

   10
   11  begin
   12
   13     --# assert a >= 0 and b >= 0 and a = a_i and b = b_i;
   14     p := 0;
   15
   16     --# assert p = 0 and a >= 0 and b >= 0 and a = a_i and b = b_i;
   17     FOR i IN integer RANGE 1..b LOOP
   18
   19         --# assert p = (i-1)*a and a >= 0 and b > 0 and a = a_i and b =
b_i;
   20        FOR j IN integer RANGE 1..a LOOP
   21
   22           p := p+1;
   23           --# assert p = (i-1)*a+j and a > 0 and b > 0 and a = a_i and b
= b_i;
   24
   25        END LOOP;
   26        --# assert p=i*a and a >= 0 and b > 0 and a = a_i and b = b_i;
   27
   28     END LOOP;
   29
   30     --# assert p=b*a and a >= 0 and b >= 0 and a = a_i and b = b_i;
```

```
   31
   32  end ex_17vc;

 !!! (  3)  Flow Error        : 31: The variable A is exported but not (inter-
nally)
            defined.
 !!! (  4)  Flow Error        : 31: The variable B is exported but not (inter-
nally)
            defined.
 !!! (  5)  Flow Error         : 35: Importation of the initial value of vari-
able P
            is ineffective.
 !!! (  6)  Flow Error        : 33: The variable A_I is neither referenced nor
            exported.
 !!! (  7)  Flow Error        : 33: The variable B_I is neither referenced nor
            exported.
 !!! (  8)  Flow Error        : 50: The imported value of B is not used in the
            derivation of A.
 !!! (  9)  Flow Error        : 50: The imported value of P is not used in the
            derivation of A.
 !!! ( 10)  Flow Error        : 50: The imported value of A is not used in the
            derivation of B.
 !!! ( 11)  Flow Error        : 50: The imported value of P is not used in the
            derivation of B.
 !!! ( 12)  Flow Error        : 50: The imported value of P is not used in the
            derivation of P.
 --- ( 13)  Warning           :403: A_I is declared as a variable but used as
a
            constant.
 --- ( 14)  Warning           :403: B_I is declared as a variable but used as
a
            constant.

 --End of file------------------------------------------------
```

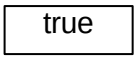
Ex_17vc.siv

```
          ******************************************************
                       Semantic Analysis of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                           Demonstration Version
          ******************************************************


       CREATED 02-JAN-2002, 08:41:29  SIMPLIFIED 02-JAN-2002, 08:41:37
                (Simplified by SPADE Simplifier, Version 1.4)

                              procedure ex_17vc
```

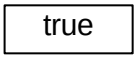
```
For path(s) from start to assertion of line 13:

procedure_ex_17vc_1.
H1:    a >= integer__first .
H2:    a <= integer__last .
H3:    b >= integer__first .
H4:    b <= integer__last .
H5:    p >= integer__first .
H6:    p <= integer__last .
       ->
C1:    a >= 0 .
C2:    b >= 0 .
C3:    a = 0 .
C4:    b = 0 .
```

```
For path(s) from assertion of line 13 to assertion of line 16:

procedure_ex_17vc_2.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 16 to assertion of line 19:

procedure_ex_17vc_3.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 26 to assertion of line 19:

procedure_ex_17vc_4.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 19 to assertion of line 23:

procedure_ex_17vc_5.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 23 to assertion of line 23:

procedure_ex_17vc_6.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 19 to assertion of line 26:
```

```
procedure_ex_17vc_7.
H1:    a_i >= 0 .
H2:    b_i > 0 .
H3:    a_i < 1 .
       ->
C1:    a_i = 0 .
```




```
For path(s) from assertion of line 23 to assertion of line 26:

procedure_ex_17vc_8.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 16 to assertion of line 30:

procedure_ex_17vc_9.


H1:    a_i >= 0 .


H2:    b_i >= 0 .
H3:    b_i < 1 .
       ->
C1:    0 = b_i * a_i .
```




```
For path(s) from assertion of line 26 to assertion of line 30:

procedure_ex_17vc_10.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 30 to finish:

procedure_ex_17vc_11.
*** true .          /* all conclusions proved */
```

Ex_17: 11 VCs gen. 9 VC rel. 7 of rel. VCs proved checked 2002.Jan.14

Example 18

Ex_18vc.ada

```
  -- ex_18vc   Examiner: verification = vc

   --# main_program;
procedure ex_18vc (k: in out integer)
   --# derives k from k;
is

   --# function I(k: in integer) return boolean;

   function P(k: in integer) return boolean is
      RetValue : boolean;
   begin
      return RetValue;
   end P;

begin

   --# assert true;
   k := 0;
   WHILE P(k) LOOP
      k := k+1;
      --# assert I(k);
   END LOOP;

   --# assert false;

end ex_18vc;
```

Ex_18vc.lst

```
          ******************************************************
                          Listing of SPARK Text
      SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                       DATE : 08-JAN-2002 10:40:40.85

Line
   1    -- ex_18vc   Examiner: verification = vc
   2
   3     --# main_program;
   4  procedure ex_18vc (k: in out integer)
   5     --# derives k from k;
   6  is
   7
   8     --# function I(k: in integer) return boolean;
   9
  10     function P(k: in integer) return boolean is
  11        RetValue : boolean;
  12     begin
  13        return RetValue;
                  ^1
```

```
 !!! (  1)  Flow Error        : 20: Expression contains reference(s) to unde-
fined
            variable RetValue.

   14     end P;

 !!! (  2)  Flow Error        : 30: The variable k is imported but neither
            referenced nor exported.
 !!! (  3)  Flow Error        : 32: The variable RetValue is neither imported
nor
            defined.
 !!! (  4)  Flow Error        : 50: The imported value of k is not used in the
            derivation of the function value.
 ??? (  5)  Warning             :602: The undefined initial value of RetValue
may be
            used in the derivation of the function value.

   15
   16  begin
   17
   18     --# assert true;
   19     k := 0;
   20     WHILE P(k) LOOP
   21        k := k+1;
   22        --# assert I(k);
   23     END LOOP;
   24
   25     --# assert false;
   26
   27  end ex_18vc;

 !!! (  6)  Flow Error        : 35: Importation of the initial value of vari-
able k
            is ineffective.
 !!! (  7)  Flow Error        : 50: The imported value of k is not used in the
            derivation of k.



 --End of file-----------------------------------------------------
```

Ex_18vc.siv

```
          ******************************************************
                       Semantic Analysis of SPARK Text
             SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                            Demonstration Version
          ******************************************************


             CREATED 08-JAN-2002, 10:40:40  SIMPLIFIED 08-JAN-2002, 10:40:48
                      (Simplified by SPADE Simplifier, Version 1.4)

                                 procedure ex_18vc
```

```
For path(s) from start to assertion of line 18:

procedure_ex_18vc_1.
*** true .          /* all conclusions proved */
```

```
For path(s) from assertion of line 18 to assertion of line 22:

procedure_ex_18vc_2.
H1:    p(0) .
       ->
C1:    i(1) .
```

OK

```
For path(s) from assertion of line 22 to assertion of line 22:

procedure_ex_18vc_3.
H1:    i(k) .
H2:    p(k) .
       ->
C1:    i(k + 1) .
```

OK

```
For path(s) from assertion of line 18 to assertion of line 25:

procedure_ex_18vc_4.
H1:    not p(0) .
       ->
C1:    false .
```

OK

```
For path(s) from assertion of line 22 to assertion of line 25:

procedure_ex_18vc_5.
H1:    i(k) .
H2:    not p(k) .
       ->
C1:    false .
```

OK

```
For path(s) from assertion of line 25 to finish:

procedure_ex_18vc_6.
*** true .          /* all conclusions proved */
```

Ex_18:	6 VCs gen.	4 VC rel.	4 of rel. VCs OK	checked 2002.Jan.14

Example 19

Ex_19vc.ada

```
   -- ex_19vc   Examiner: verification = vc
```

```
   --# main_program;
procedure ex_19vc (N: in out integer; s: in out integer)
   --# derives S, N from N, S;
is

   n_i: integer := 0;
   j  : integer := 0;

begin
   n_i := n;
   --# assert n >= 0 and n = n_i;
   s := 0;
   --# assert n >= 0 and s = 0 and n = n_i;
   FOR i IN integer RANGE 0..n-1 LOOP
      j := 2*i+1;
      s := s+j;
      --# assert s = (i+1)**2 and n = n_i;
   END LOOP;

   --# assert s = n**2 and n = n_i;
end ex_19vc;
```

Ex_19vc.lst

```
          ******************************************************
                          Listing of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                        DATE : 02-JAN-2002 09:05:33.89

Line
   1    -- ex_19vc   Examiner: verification = vc
   2
   3     --# main_program;
   4  procedure ex_19vc (N: in out integer; s: in out integer)
   5     --# derives S, N from N, S;
   6  is
   7
   8     n_i: integer := 0;

 !!! (  1)  Flow Error         : 54: The initialization at declaration of n_i
is
            ineffective.

   9     j  : integer := 0;
```

```
 !!! (  2)  Flow Error        : 54: The initialization at declaration of j is
            ineffective.

   10
   11  begin
   12     n_i := n;
          ^3
 !!! (  3)  Flow Error        : 10: Ineffective statement.
```

```
   13     --# assert n >= 0 and n = n_i;
   14     s := 0;
   15     --# assert n >= 0 and s = 0 and n = n_i;
   16     FOR i IN integer RANGE 0..n-1 LOOP
   17        j := 2*i+1;
   18        s := s+j;
   19        --# assert s = (i+1)**2 and n = n_i;
   20     END LOOP;
   21
   22     --# assert s = n**2 and n = n_i;
```

```
   23  end ex_19vc;

 !!! (  4)  Flow Error        : 31: The variable N is exported but not (inter-
nally)
            defined.
 !!! (  5)  Flow Error        : 35: Importation of the initial value of vari-
able s
            is ineffective.
 !!! (  6)  Flow Error        : 33: The variable n_i is neither referenced nor
            exported.
 !!! (  7)  Flow Error        : 50: The imported value of s is not used in the
            derivation of N.
 !!! (  8)  Flow Error        : 50: The imported value of s is not used in the
            derivation of s.

 --End of file------------------------------------------------
```

Ex_19vc.siv

```
          ******************************************************
                       Semantic Analysis of SPARK Text
              SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                              Demonstration Version
          ******************************************************


              CREATED 02-JAN-2002, 09:05:33  SIMPLIFIED 02-JAN-2002, 09:05:41
                       (Simplified by SPADE Simplifier, Version 1.4)

                                  procedure ex_19vc
```

```
For path(s) from start to assertion of line 13:

procedure_ex_19vc_1.
H1:    n >= integer__first .
H2:    n <= integer__last .
H3:    s >= integer__first .
H4:    s <= integer__last .
       ->
C1:    n >= 0 .
```

```
For path(s) from assertion of line 13 to assertion of line 15:

procedure_ex_19vc_2.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 15 to assertion of line 19:

procedure_ex_19vc_3.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 19 to assertion of line 19:

procedure_ex_19vc_4.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 15 to assertion of line 22:

procedure_ex_19vc_5.
H1:    n_i >= 0 .
H2:    n_i < 1 .
       ->
C1:    0 = n_i * n_i .


For path(s) from assertion of line 19 to assertion of line 22:

procedure_ex_19vc_6.
*** true .          /* all conclusions proved */
```

true

```
For path(s) from assertion of line 22 to finish:

procedure_ex_19vc_7.
```

```
*** true .          /* all conclusions proved */
```

Ex_19: 7 VCs gen. 5 VC rel. 4 of rel. VCs proved checked 2002.Jan.17

Example 20

Ex_20vc.ada

```
   -- ex_20vc   Examiner: verification = vc

    --# main_program;
procedure ex_20vc (n: in out integer; a: in out integer)
    --# derives n, a from n, a;
is

    n_i: integer := 0;
    term_old: integer := 0; -- Terminierungsfunktion alt

begin
    N_i := n;
    term_old := n - a;
    --# assert  a=0 and n>=0 and n = n_i and
    --# term_old = n - a;
    WHILE (a+1)**2 <= n LOOP
       term_old := n - a;

       a := a+1;

       --# assert a**2<=n and n = n_i and
       --# n - a < term_old and n - a >=0; -- Terminierungsfunktion
    END LOOP;

    --# assert (a+1)**2>n and n>=a**2 and n = n_i;
end ex_20vc;
```

Ex_20vc.lst

```
          ******************************************************
                          Listing of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                        DATE : 02-JAN-2002 09:39:48.84

Line
```

```
    1    -- ex_20vc   Examiner: verification = vc
    2
    3     --# main_program;
    4  procedure ex_20vc (n: in out integer; a: in out integer)
    5     --# derives n, a from n, a;
    6  is
    7
    8     n_i: integer := 0;

 !!! (  1)  Flow Error         : 54: The initialization at declaration of n_i
is
            ineffective.

    9     term_old: integer := 0; -- Terminierungsfunktion alt

 !!! (  2)  Flow Error          : 54: The initialization at declaration of
term_old is
            ineffective.

   10
   11  begin
   12     N_i := n;
          ^3
 !!! (  3)  Flow Error        : 10: Ineffective statement.

   13     term_old := n - a;
          ^4
 !!! (  4)  Flow Error        : 10: Ineffective statement.

   14     --# assert  a=0 and n>=0 and n = n_i and
   15     --# term_old = n - a;
   16     WHILE (a+1)**2 <= n LOOP
   17        term_old := n - a;
             ^5
 !!! (  5)  Flow Error        : 10: Ineffective statement.

   18
   19        a := a+1;
   20
   21        --# assert a**2<=n and n = n_i and
   22        --# n - a < term_old and n - a >=0; -- Terminierungsfunktion
   23     END LOOP;
   24
   25     --# assert (a+1)**2>n and n>=a**2 and n = n_i;
   26  end ex_20vc;

 !!! (  6)  Flow Error        : 31: The variable n is exported but not (inter-
nally)
            defined.
 !!! (  7)  Flow Error        : 33: The variable n_i is neither referenced nor
```

```
           exported.
 !!! (   8)  Flow Error         : 33: The variable term_old is neither refer-
enced nor
           exported.
 !!! (   9)  Flow Error        : 50: The imported value of a is not used in the
           derivation of n.

 --End of file------------------------------------------------
```

Ex_20vc.siv

```
          ******************************************************
                      Semantic Analysis of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


       CREATED 02-JAN-2002, 09:39:48  SIMPLIFIED 02-JAN-2002, 09:39:56
               (Simplified by SPADE Simplifier, Version 1.4)

                              procedure ex_20vc


 For path(s) from start to assertion of line 14:

 procedure_ex_20vc_1.
 H1:    n >= integer__first .
 H2:    n <= integer__last .
 H3:    a >= integer__first .
 H4:    a <= integer__last .
        ->
 C1:    a = 0 .
 C2:    n >= 0 .


 For path(s) from assertion of line 14 to assertion of line 21:

 procedure_ex_20vc_2.
 *** true .          /* all conclusions proved */


 For path(s) from assertion of line 21 to assertion of line 21:

 procedure_ex_20vc_3.
 H1:    a * a <= n_i .
 H2:    n_i - a < term_old .
 H3:    n_i - a >= 0 .
 H4:    (a + 1) * (a + 1) <= n_i .
        ->
```

true: $(a+1)(a+1) \leq n\_i \equiv 0 \leq a(a+1) \leq n\_i - (a+1)$

```
C1:    n_i - (a + 1) >= 0 .


For path(s) from assertion of line 14 to assertion of line 25:

procedure_ex_20vc_4.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 21 to assertion of line 25:

procedure_ex_20vc_5.
*** true .          /* all conclusions proved */
```

```
For path(s) from assertion of line 25 to finish:

procedure_ex_20vc_6.
*** true .          /* all conclusions proved */
```

Ex_20: 6 VCs gen. 4 VC rel. 3 of rel. VCs proved checked 2002.Jan.17

Example 21

Ex_21vc.ada

```
  -- ex_21vc   Examiner: verification = vc

   --# main_program;
procedure ex_21vc (X: in out integer; Y: in out integer)
   --# derives X, Y from X, Y;
is

   temp: integer := 0;
   x_i : integer := 0;
   y_i : integer := 0;

begin
   x_i := x;
   y_i := y;

   --# assert x = x_i and y = y_i;
   temp := x ;
   x    := y ;
   y    := temp;
   --# assert x = y_i and y = x_i;

end ex_21vc;
```

Ex_21vc.lst

```
          ******************************************************
                              Listing of SPARK Text
                SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                              Demonstration Version
          ******************************************************


                             DATE : 02-JAN-2002 09:40:21.22

 Line
    1    -- ex_21vc   Examiner: verification = vc
    2
    3     --# main_program;
    4  procedure ex_21vc (X: in out integer; Y: in out integer)
    5     --# derives X, Y from X, Y;
    6  is
    7
    8     temp: integer := 0;

 !!! (  1)  Flow Error        : 54: The initialization at declaration of temp
is
            ineffective.

    9     x_i : integer := 0;

 !!! (  2)  Flow Error        : 54: The initialization at declaration of x_i
is
            ineffective.

   10     y_i : integer := 0;

 !!! (  3)  Flow Error        : 54: The initialization at declaration of y_i
is
            ineffective.

   11
   12  begin
   13     x_i := x;
          ^4
 !!! (  4)  Flow Error        : 10: Ineffective statement.

   14     y_i := y;
          ^5
 !!! (  5)  Flow Error        : 10: Ineffective statement.

   15
   16     --# assert x = x_i and y = y_i;
   17     temp := x ;
```

```
  18     x    := y ;
  19     y    := temp;
  20     --# assert x = y_i and y = x_i;
  21
  22  end ex_21vc;

!!! (  6)  Flow Error        : 33: The variable x_i is neither referenced nor
           exported.
!!! (  7)  Flow Error        : 33: The variable y_i is neither referenced nor
           exported.
!!! (  8)  Flow Error        : 50: The imported value of X is not used in the
           derivation of X.
!!! (  9)  Flow Error        : 50: The imported value of Y is not used in the
           derivation of Y.

--End of file------------------------------------------------
```

Ex_21vc.siv

```
          ******************************************************
                       Semantic Analysis of SPARK Text
             SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                              Demonstration Version
          ******************************************************


             CREATED 02-JAN-2002, 09:40:21  SIMPLIFIED 02-JAN-2002, 09:40:29
                      (Simplified by SPADE Simplifier, Version 1.4)

                                  procedure ex_21vc




For path(s) from start to assertion of line 16:

procedure_ex_21vc_1.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 16 to assertion of line 20:

procedure_ex_21vc_2.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 20 to finish:

procedure_ex_21vc_3.
```

```
*** true .          /* all conclusions proved */
```

Ex_21: 3 VCs gen. 1 VC rel. 1 of rel. VCs proved checked 2002.Jan.17

Example 22

Ex_22vc.ada

```
   -- ex_22vc   Examiner: verification = vc

    --# main_program;
procedure ex_22vc (X: in out integer; Y: in out integer)
    --# derives X, Y from X, Y;
is

    x_i: integer := 0;
    y_i: integer := 0;

begin
    x_i := x;
    y_i := y;

    --# assert x = x_i and y = y_i;
    x := x - y;
    y := x + y;
    x := y - x;
    --# assert x = y_i and y = x_i;

end ex_22vc;
```

Ex_22vc.lst

```
          ******************************************************
                          Listing of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                        DATE : 02-JAN-2002 09:44:01.55

Line
   1    -- ex_22vc   Examiner: verification = vc
   2
   3     --# main_program;
   4  procedure ex_22vc (X: in out integer; Y: in out integer)
   5     --# derives X, Y from X, Y;
   6  is
```

```
    7
    8     x_i: integer := 0;

 !!! (  1)  Flow Error         : 54: The initialization at declaration of x_i
is
            ineffective.

    9     y_i: integer := 0;

 !!! (  2)  Flow Error         : 54: The initialization at declaration of y_i
is
            ineffective.

   10
   11  begin
   12     x_i := x;
          ^3
 !!! (  3)  Flow Error        : 10: Ineffective statement.

   13     y_i := y;
          ^4
 !!! (  4)  Flow Error        : 10: Ineffective statement.

   14
   15     --# assert x = x_i and y = y_i;
   16     x := x - y;
   17     y := x + y;
   18     x := y - x;
   19     --# assert x = y_i and y = x_i;
   20
   21  end ex_22vc;

 !!! (  5)  Flow Error        : 33: The variable x_i is neither referenced nor
            exported.
 !!! (  6)  Flow Error        : 33: The variable y_i is neither referenced nor
            exported.

 --End of file-----------------------------------------------------
```

Ex_22vc.siv

```
            ******************************************************
                        Semantic Analysis of SPARK Text
          SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                             Demonstration Version
            ******************************************************


          CREATED 02-JAN-2002, 09:44:01  SIMPLIFIED 02-JAN-2002, 09:44:09
                   (Simplified by SPADE Simplifier, Version 1.4)
```

```
                                procedure ex_22vc


For path(s) from start to assertion of line 15:

procedure_ex_22vc_1.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 15 to assertion of line 19:

procedure_ex_22vc_2.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 19 to finish:

procedure_ex_22vc_3.
*** true .          /* all conclusions proved */
```

Ex_22: 3 VCs gen. 1 VC rel. 1 of rel. VCs proved checked 2002.Jan.17

Example 23

Ex_23vc.ada

```
   -- ex_23vc   Examiner: verification = vc

    --# main_program;
procedure ex_23vc (X: in out integer; Y: in out integer)
    --# derives X, Y from X, Y;
is

    x_i : integer := 0;
    y_i : integer := 0;

begin

    x_i := x;
    y_i := y;

    --# assert x = x_i and -100 <= x and x <= +100 and
    --#        y = y_i and -100 <= y and y <= +100;
    x := x - y;
```

```
   --# assert  -100 <= x and x <= +100 and -100 <= y and y <= +100;
   y := x + y;

   --# assert  -100 <= x and x <= +100 and -100 <= y and y <= +100;
   x := y - x;

   --# assert  x = y_i and -100 <= x and x <= +100 and
   --#         y = x_i and -100 <= y and y <= +100;
```

```
end ex_23vc;
```

Ex_23vc.lst

```
          ******************************************************
                          Listing of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                       DATE : 02-JAN-2002 09:45:55.33

 Line
    1    -- ex_23vc   Examiner: verification = vc
    2
    3     --# main_program;
    4  procedure ex_23vc (X: in out integer; Y: in out integer)
    5     --# derives X, Y from X, Y;
    6  is
    7
    8     x_i : integer := 0;

 !!! (  1)  Flow Error         : 54: The initialization at declaration of x_i
is
            ineffective.

    9     y_i : integer := 0;

 !!! (  2)  Flow Error         : 54: The initialization at declaration of y_i
is
            ineffective.

   10
   11  begin
   12
   13     x_i := x;
          ^3
 !!! (  3)  Flow Error        : 10: Ineffective statement.

   14     y_i := y;
          ^4
```

```
!!! (  4)  Flow Error        : 10: Ineffective statement.

  15
  16     --# assert x = x_i and -100 <= x and x <= +100 and
  17     --#        y = y_i and -100 <= y and y <= +100;
  18     x := x - y;
  19
  20     --# assert  -100 <= x and x <= +100 and -100 <= y and y <= +100;
  21     y := x + y;
  22
  23     --# assert  -100 <= x and x <= +100 and -100 <= y and y <= +100;
  24     x := y - x;
  25
  26     --# assert  x = y_i and -100 <= x and x <= +100 and
  27     --#         y = x_i and -100 <= y and y <= +100;
  28
  29  end ex_23vc;

!!! (  5)  Flow Error        : 33: The variable x_i is neither referenced nor
           exported.
!!! (  6)  Flow Error        : 33: The variable y_i is neither referenced nor
           exported.

--End of file-----------------------------------------------------
```

Ex_23vc.siv

```
          ******************************************************
                      Semantic Analysis of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                           Demonstration Version
          ******************************************************


       CREATED 02-JAN-2002, 09:45:55  SIMPLIFIED 02-JAN-2002, 09:46:03
                (Simplified by SPADE Simplifier, Version 1.4)

                               procedure ex_23vc




For path(s) from start to assertion of line 16:

procedure_ex_23vc_1.
H1:    x >= integer__first .
H2:    x <= integer__last .
H3:    y >= integer__first .
H4:    y <= integer__last .
       ->
C1:    - 100 <= x .
```

```
C2:    x <= 100 .
C3:    - 100 <= y .
C4:    y <= 100 .
```

```
For path(s) from assertion of line 16 to assertion of line 20:

procedure_ex_23vc_2.
H1:    - 100 <= x_i .
H2:    x_i <= 100 .
H3:    - 100 <= y_i .
H4:    y_i <= 100 .
       ->                    false
C1:    - 100 <= x_i - y_i .
C2:    x_i - y_i <= 100 .


For path(s) from assertion of line 20 to assertion of line 23:

procedure_ex_23vc_3.
H1:    - 100 <= x .
H2:    x <= 100 .
H3:    - 100 <= y .
H4:    y <= 100 .
       ->                    false
C1:    - 100 <= x + y .
C2:    x + y <= 100 .


For path(s) from assertion of line 23 to assertion of line 26:

procedure_ex_23vc_4.
H1:    - 100 <= x .
H2:    x <= 100 .
H3:    - 100 <= y .
H4:    y <= 100 .
       ->                    false
C1:    y - x = y_i .
C2:    - 100 <= y - x .
C3:    y - x <= 100 .
C4:    y = x_i .
```

```
For path(s) from assertion of line 26 to finish:

procedure_ex_23vc_5.
*** true .          /* all conclusions proved */
```

Ex_23:	5 VCs gen.	3 VC rel.	0 of rel. VCs disproved	checked 2002.Jan.17

Example 24

Ex_24vc.ada

```
   -- ex_24vc   Examiner: verification = vc

    --# main_program;
procedure ex_24vc (X: in out integer; Y: in out integer)
    --# derives X, Y from X, Y;
is

    x_i: integer := 0;
    y_i: integer := 0;

begin
    x_i := x;
    y_i := y;

    --# assert  x = x_i and y = y_i and -100 <= x-y and x-y <= +100 and
    --#          -100 <= x and x <= +100 and -100 <= y and y <= +100;
    x := x - y;

    --# assert x = x_i-y_i and y = y_i and -100 <= x+y and x+y <= +100 and
    --#        -100 <= x and x <= +100 and -100 <= y and y <= +100;
    y := x + y;

    --# assert x = x_i-y_i and y = x_i and -100 <= y-x and y-x <= +100 and
    --#        -100 <= x and x <= +100 and -100 <= y and y <= +100;
    x := y - x;

    --# assert x = y_i and -100 <= x and x <= +100 and
    --#        y = x_i and -100 <= y and y <= +100;

end ex_24vc;
```

Ex_24vc.lst

```
          ******************************************************
                          Listing of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                        DATE : 02-JAN-2002 09:50:27.04

Line
   1    -- ex_24vc   Examiner: verification = vc
```

```
    2
    3     --# main_program;
    4  procedure ex_24vc (X: in out integer; Y: in out integer)
    5     --# derives X, Y from X, Y;
    6  is
    7
    8     x_i: integer := 0;

 !!! (  1)  Flow Error         : 54: The initialization at declaration of x_i
is
            ineffective.

    9     y_i: integer := 0;

 !!! (  2)  Flow Error         : 54: The initialization at declaration of y_i
is
            ineffective.

   10
   11  begin
   12     x_i := x;
          ^3
 !!! (  3)  Flow Error        : 10: Ineffective statement.

   13     y_i := y;
          ^4
 !!! (  4)  Flow Error        : 10: Ineffective statement.

   14
   15     --# assert  x = x_i and y = y_i and -100 <= x-y and x-y <= +100 and
   16     --#         -100 <= x and x <= +100 and -100 <= y and y <= +100;
   17     x := x - y;
   18
   19     --# assert x = x_i-y_i and y = y_i and -100 <= x+y and x+y <= +100
and
   20     --#        -100 <= x and x <= +100 and -100 <= y and y <= +100;
   21     y := x + y;
   22
   23     --# assert x = x_i-y_i and y = x_i and -100 <= y-x and y-x <= +100
and
   24     --#        -100 <= x and x <= +100 and -100 <= y and y <= +100;
   25     x := y - x;
   26
   27     --# assert x = y_i and -100 <= x and x <= +100 and
   28     --#        y = x_i and -100 <= y and y <= +100;
   29
   30  end ex_24vc;

 !!! (  5)  Flow Error        : 33: The variable x_i is neither referenced nor
            exported.
```

```
!!! (  6)  Flow Error        : 33: The variable y_i is neither referenced nor
           exported.

--End of file------------------------------------------------
```

Ex_24vc.siv

```
          ******************************************************
                        Semantic Analysis of SPARK Text
              SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                              Demonstration Version
          ******************************************************


              CREATED 02-JAN-2002, 09:50:27  SIMPLIFIED 02-JAN-2002, 09:50:34
                       (Simplified by SPADE Simplifier, Version 1.4)

                                  procedure ex_24vc



For path(s) from start to assertion of line 15:

procedure_ex_24vc_1.
H1:    x >= integer__first .
H2:    x <= integer__last .
H3:    y >= integer__first .
H4:    y <= integer__last .
       ->
C1:    - 100 <= x - y .
C2:    x - y <= 100 .
C3:    - 100 <= x .
C4:    x <= 100 .
C5:    - 100 <= y .
C6:    y <= 100 .


For path(s) from assertion of line 15 to assertion of line 19:

procedure_ex_24vc_2.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 19 to assertion of line 23:

procedure_ex_24vc_3.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 23 to assertion of line 27:
```

```
procedure_ex_24vc_4.
*** true .          /* all conclusions proved */


For path(s) from assertion of line 27 to finish:

procedure_ex_24vc_5.
*** true .          /* all conclusions proved */
```

Ex_24:	5 VCs gen.	3 VC rel.	3 of rel. VCs proved	checked 2002.01.18

Example 25

Ex_25vc.ada

```
   -- ex_25vc   Examiner: verification = vc

    --# main_program;
procedure ex_25vc (X: in out integer; Y: in out integer)
    --# derives X, Y from X, Y;
is

    x_i: integer := 0;
    y_i: integer := 0;

    function f(a: in integer; b: in integer) return integer is
       Ret_Value: integer;
    begin
       return ret_Value;
    end f;

    function g(a: in integer; b: in integer) return integer is
       Ret_Value: integer;
    begin
       return Ret_Value;
    end g;

    function h(a: in integer; b: in integer) return integer is
       Ret_Value: integer;
    begin
       return Ret_Value;
    end h;

begin
    x_i := x;
    y_i := y;
```

```
    --# assert x = x_i and y = y_i;

    x := f(x,y);
    y := g(x,y);
    x := h(x,y);

    --# assert x = y_i and y = x_i;
 end ex_25vc;
```

Ex_25vc.lst

```
          ******************************************************
                          Listing of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                          Demonstration Version
          ******************************************************


                       DATE : 03-JAN-2002 10:25:23.04

 Line
    1    -- ex_25vc   Examiner: verification = vc
    2
    3     --# main_program;
    4  procedure ex_25vc (X: in out integer; Y: in out integer)
    5     --# derives X, Y from X, Y;
    6  is
    7
    8     x_i: integer := 0;

 !!! (  1)  Flow Error         : 54: The initialization at declaration of x_i
is
            ineffective.

    9     y_i: integer := 0;

 !!! (  2)  Flow Error         : 54: The initialization at declaration of y_i
is
            ineffective.

   10
   11     function f(a: in integer; b: in integer) return integer is
   12        Ret_Value: integer;
   13     begin
   14        return ret_Value;
                    ^3
 !!! (  3)  Flow Error         : 20: Expression contains reference(s) to unde-
fined
            variable Ret_Value.

   15     end f;
```

```
 !!! (  4)  Flow Error        : 32: The variable Ret_Value is neither imported
nor
            defined.
 !!! (  5)  Flow Error        : 50: The imported value of a is not used in the
            derivation of the function value.
 !!! (  6)  Flow Error        : 50: The imported value of b is not used in the
            derivation of the function value.
 ??? (  7)  Warning            :602: The undefined initial value of Ret_Value
may be
            used in the derivation of the function value.
 !!! (  8)  Flow Error        : 30: The variable a is imported but neither
            referenced nor exported.
 !!! (  9)  Flow Error        : 30: The variable b is imported but neither
            referenced nor exported.

   16
   17     function g(a: in integer; b: in integer) return integer is
   18        Ret_Value: integer;
   19     begin
   20        return Ret_Value;
                    ^10
 !!! ( 10)  Flow Error         : 20: Expression contains reference(s) to unde-
fined
            variable Ret_Value.

   21     end g;

 !!! ( 11)  Flow Error        : 32: The variable Ret_Value is neither imported
nor
            defined.
 !!! ( 12)  Flow Error        : 50: The imported value of a is not used in the
            derivation of the function value.
 !!! ( 13)  Flow Error        : 50: The imported value of b is not used in the
            derivation of the function value.
 ??? ( 14)  Warning            :602: The undefined initial value of Ret_Value
may be
            used in the derivation of the function value.
 !!! ( 15)  Flow Error        : 30: The variable a is imported but neither
            referenced nor exported.
 !!! ( 16)  Flow Error        : 30: The variable b is imported but neither
            referenced nor exported.

   22
   23     function h(a: in integer; b: in integer) return integer is
   24        Ret_Value: integer;
   25     begin
   26        return Ret_Value;
                    ^17
```

```
 !!! ( 17)  Flow Error        : 20: Expression contains reference(s) to unde-
fined
            variable Ret_Value.

   27     end h;

 !!! ( 18)  Flow Error        : 32: The variable Ret_Value is neither imported
nor
            defined.
 !!! ( 19)  Flow Error        : 50: The imported value of a is not used in the
            derivation of the function value.
 !!! ( 20)  Flow Error        : 50: The imported value of b is not used in the
            derivation of the function value.
 ??? ( 21)  Warning           :602: The undefined initial value of Ret_Value
may be
            used in the derivation of the function value.
 !!! ( 22)  Flow Error        : 30: The variable a is imported but neither
            referenced nor exported.
 !!! ( 23)  Flow Error        : 30: The variable b is imported but neither
            referenced nor exported.

   28
   29  begin
   30     x_i := x;
          ^24
 !!! ( 24)  Flow Error        : 10: Ineffective statement.

   31     y_i := y;
          ^25
 !!! ( 25)  Flow Error        : 10: Ineffective statement.

   32
   33     --# assert x = x_i and y = y_i;
   34
   35     x := f(x,y);
   36     y := g(x,y);
   37     x := h(x,y);
   38
   39     --# assert x = y_i and y = x_i;
   40  end ex_25vc;

 !!! ( 26)  Flow Error        : 33: The variable x_i is neither referenced nor
            exported.
 !!! ( 27)  Flow Error        : 33: The variable y_i is neither referenced nor
            exported.

 --End of file-------------------------------------------------
```

Ex_25vc.siv

```
           ******************************************************
```

```
                    Semantic Analysis of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                         Demonstration Version
            ******************************************************


       CREATED 03-JAN-2002, 10:25:23  SIMPLIFIED 03-JAN-2002, 10:25:30
                (Simplified by SPADE Simplifier, Version 1.4)

                              procedure ex_25vc



For path(s) from start to assertion of line 33:

procedure_ex_25vc_1.
*** true .          /* all conclusions proved */
```

```
For path(s) from assertion of line 33 to assertion of line 39:

procedure_ex_25vc_2.
H1:    true .                    OK
       ->
C1:    h(f(x_i,y_i),g(f(x_i,y_i),y_i)) = y_i .
C2:    g(f(x_i,y_i),y_i) = x_i .
```

```
For path(s) from assertion of line 39 to finish:

procedure_ex_25vc_3.
*** true .          /* all conclusions proved */
```

Ex_25:	3 VCs gen.	1 VC rel.	1 of rel. VCs correct	checked 2002.Jan.18

Example 26

Ex_26vc.ada

```
   -- ex_26vc   Examiner: verification = rtc

    --# main_program;
procedure ex_26vc (n: in out integer; x: in out integer)
    --# derives X, n from n, x;
is
    n_i: integer := 0;
    y  : integer := 0;
```

```
   z  : integer := 0;


begin
   n_i := n;
   --# assert n >= 0 and n = n_i;
   x := 0;
   y := 1;
   z := 6;
   --# assert n >= 0 and n = n_i and
   --#        x=0 and y=1 and z=6;
   FOR i IN integer RANGE 1 .. n LOOP
      x := x+y;
      y := y+z;
      z := z+6;
      --# assert x = i**3 and y = 3*i**2 + 3*i + 1 and
      --#        z=6*i+6 and n = n_i;

   END LOOP;

   --# assert x=n**3 and n = n_i;
end ex_26vc;
```

Ex_26vc.lst

```
           ******************************************************
                              Listing of SPARK Text
          SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
                              Demonstration Version
           ******************************************************


                           DATE : 02-JAN-2002 09:53:23.84

Line
   1    -- ex_26vc   Examiner: verification = rtc
   2
   3     --# main_program;
   4  procedure ex_26vc (n: in out integer; x: in out integer)
   5     --# derives X, n from n, x;
   6  is
   7     n_i: integer := 0;

 !!! (  1)  Flow Error         : 54: The initialization at declaration of n_i
is
            ineffective.

   8     y  : integer := 0;

 !!! (  2)  Flow Error        : 54: The initialization at declaration of y is
            ineffective.
```

```
    9     z  : integer := 0;

 !!! (  3)  Flow Error        : 54: The initialization at declaration of z is
            ineffective.

   10
   11
   12  begin
   13     n_i := n;
          ^4
 !!! (  4)  Flow Error        : 10: Ineffective statement.

   14     --# assert n >= 0 and n = n_i;
   15     x := 0;
   16     y := 1;
   17     z := 6;
   18     --# assert n >= 0 and n = n_i and
   19     --#        x=0 and y=1 and z=6;
   20     FOR i IN integer RANGE 1 .. n LOOP
   21        x := x+y;
   22        y := y+z;
   23        z := z+6;
   24        --# assert x = i**3 and y = 3*i**2 + 3*i + 1 and
   25        --#        z=6*i+6 and n = n_i;
   26
   27     END LOOP;
   28
   29     --# assert x=n**3 and n = n_i;
   30  end ex_26vc;

 !!! (  5)  Flow Error        : 31: The variable n is exported but not (inter-
nally)
            defined.
 !!! (  6)  Flow Error        : 35: Importation of the initial value of vari-
able x
            is ineffective.
 !!! (  7)  Flow Error        : 33: The variable n_i is neither referenced nor
            exported.
 !!! (  8)  Flow Error        : 50: The imported value of x is not used in the
            derivation of n.
 !!! (  9)  Flow Error        : 50: The imported value of x is not used in the
            derivation of x.

 --End of file------------------------------------------------
```

Ex_26vc.siv

```
          ******************************************************
                      Semantic Analysis of SPARK Text
       SPARK95 Examiner with VC and RTC Generator Release 5.01 / 08.00
```

```
                        Demonstration Version
          ******************************************************


       CREATED 02-JAN-2002, 09:53:23  SIMPLIFIED 02-JAN-2002, 09:53:31
                (Simplified by SPADE Simplifier, Version 1.4)

                              procedure ex_26vc


 For path(s) from start to assertion of line 14:

 procedure_ex_26vc_1.
 H1:    n >= integer__first .
 H2:    n <= integer__last .
 H3:    x >= integer__first .
 H4:    x <= integer__last .
        ->
 C1:    n >= 0 .


 For path(s) from assertion of line 14 to assertion of line 18:

 procedure_ex_26vc_2.
 *** true .          /* all conclusions proved */


 For path(s) from assertion of line 18 to assertion of line 24:

 procedure_ex_26vc_3.
 *** true .          /* all conclusions proved */


 For path(s) from assertion of line 24 to assertion of line 24:

 procedure_ex_26vc_4.
 H1:    loop__1__i <> n_i .
        ->
 C1:    loop__1__i ** 3 + (3 * (loop__1__i * loop__1__i) + 3 * loop__1__i + 1)
= (loop__1__i + 1) ** 3 .


 For path(s) from assertion of line 18 to assertion of line 29:

 procedure_ex_26vc_5.
 H1:    n_i >= 0 .
 H2:    n_i < 1 .
        ->
```

true

true

```
C1:    0 = n_i ** 3 .


For path(s) from assertion of line 24 to assertion of line 29:

procedure_ex_26vc_6.
*** true .          /* all conclusions proved */
```

```
For path(s) from assertion of line 29 to finish:

procedure_ex_26vc_7.
*** true .          /* all conclusions proved */
```

Ex_26:	7 VCs gen.	5 VC rel.	3 of rel. VCs proved	checked 2002.Jan.18