Problem Set 4
Safe and Secure Software (Winter Term 2016/17)
Example Solution Tasks 2 and 3

Bauhaus-Universität Weimar, Chair of Media Security
Prof. Dr. Stefan Lucks, Eik List
URL: `http://www.uni-weimar.de/de/medien/professuren/mediensicherheit/teaching/`

Mini Project – Data Flow and Hoare Logic (1+1+2+2)

Given the following function `F`:

a) Add the correct data-flow annotations (`def`, `p-use`, `c-use`).

b) Visualize the control-flow graph.

c) Derive appropriate pre- and post-conditions, loop variant, and invariant.

d) Use Hoare logic to show its *total* correctness. Denote the known statements at every step, and denote your used rules (assignment, condition, Implication, etc.) for deriving and reforming statements.

e) **Bonus +2:** Prove the total correctness (with statements, rules) and give appropriate pre-/post-conditions, variant, invariant also for function `G`.

```
function F(N: Natural) return Natural is
    I: Natural := 0;   -- def(I)
    X: Natural := 1;   -- def(X)
begin                  -- def(N)
    while I < N loop   -- p-use(I, N)
        I := I + 1;    -- c-use(I), def(I)
        X := X * I;    -- c-use(X, I), def(X)
    end loop;

    return X;          -- c-use(X)
end F;
```

```
function F(N: Natural) return Natural is
    {Pre-Condition := $N \geq 0$}

    I: Natural := 0;
    {$I = 0$, $N \geq 0$} (Assignment)

    X: Natural := 1;
    {$X = 1$, $I = 0$, $N \geq 0$} (Assignment)
begin
    {$I \leq N$} (Implication)
    {$X = I!$} (Invariant)
    while I < N loop
        {$I < N$, $X = I!$} (Condition)

        I := I + 1;
        {$I = I'Old + 1$, $I'Old < N$, $X = I'Old!$} (Assignment)

        X := X * I;
        {$X = X'Old * I$, $I = I'Old + 1$, $I'Old < N$, $X = I'Old!$} (Assignment)
        {$X = (I-1)! * I = I!$, $I = I'Old + 1$, $I <= N$} (Implication)
        {Variant   := $N - I$}
        {Invariant := $X = I!$}
    end loop;
    {$I >= N$} (Inverse Condition)
    {$X = I!$} (Invariant)
    {$I <= N$} (From loop)
    {$X = I!$, $I = N$} (Strengthening)

    return X;
    {Post-Condition := $F'Result = N!$} (Implication)
end F;
```

```
function G(N: Natural; K: Natural) return Natural is
    {Pre-Condition := $N \geq 0$, $K \geq 0$, $N \geq K$}
    X: Natural;
    Y: Natural;
    Z: Natural;
begin
    X := F(K);
    {$X = K!$} (Assignment)

    Y := F(N-K);
    {$Y = (N-K)!$, $X = K!$} (Assignment)

    Z := F(N);
    {$Z = N!$, $Y = (N-K)!$, $X = K!$} (Assignment)

    return Z / (X * Y);
    {Post-Condition := $G'Result = \binom{N}{K}$} (Implication)
end G;
```

Mini-Project – Hoare Logic (4)

Given the package below.

a) Add the correct data-flow annotations (`def`, `p-use`, `c-use`).

b) Visualize the control-flow graph.

c) Derive appropriate pre- and post-conditions, loop variant, and invariant.

d) Use Hoare logic to show its *total* correctness. Of course, you are allowed to simplify the type casts.

```
Procedure S(X: in out Natural; Y: in out Natural) is
begin -- def(X), def(Y)
    X := Natural(Unsigned(X) xor Unsigned(Y)); -- c-use(X, Y), def(X)
    Y := Natural(Unsigned(X) xor Unsigned(Y)); -- c-use(X, Y), def(Y)
    X := Natural(Unsigned(X) xor Unsigned(Y)); -- c-use(X, Y), def(X)
end S;

Procedure T(X: in out Natural; Y: in out Natural; Z: in out Natural) is
begin             -- def(X, Y, Z)
    if X > Y then -- p-use(X, Y)
        S(X, Y);  -- c-use(X, Y), def(X, Y)
    end if;

    if Y > Z then -- p-use(Y, Z)
        S(Y, Z);  -- c-use(Y, Z), def(Y, Z)
    end if;

    if X > Y then -- p-use(X, Y)
        S(X, Y);  -- c-use(X, Y), def(X, Y)
    end if;
end T;
```

```
Procedure S(X: in out Natural; Y: in out Natural) is
begin
    X := Natural(Unsigned(X) xor Unsigned(Y));
    {$X = X^O \oplus Y$} (Assignment)

    Y := Natural(Unsigned(X) xor Unsigned(Y));
    {$Y = Y^O \oplus X, X = X^O \oplus Y^O$} (Assignment)
    {$Y = X^O, X = X^O \oplus Y^O$} (Implication)

    X := Natural(Unsigned(X) xor Unsigned(Y));
    {$X = X'Old \oplus Y$, $Y = X^O$, $X'Old = X^O \oplus Y^O$} (Assignment)
    {$X = Y^O, Y = X^O$} (Implication)
    {Post-Condition := $X = Y^O, Y = X^O$}
end S;

Procedure T(X: in out Natural; Y: in out Natural; Z: in out Natural) is
begin
    {$X = X^O$, $Y = Y^O$, $Z = Z^O$}
    if X > Y then
        {$X = X^O$, $Y = Y^O$, $Z = Z^O$, $X > Y$} (Condition)

        S(X, Y);
        {$X = Y^O$, $Y = X^O$, $Z = Z^O$, $X < Y$} (Swap)
    else
        {$X = X^O$, $Y = Y^O$, $Z = Z^O$, $X \leq Y$} (Inverse Condition)
        Null;
    end if;
    {($X = Y^O$, $Y = X^O$, $Z = Z^O$, $X < Y$) or
     ($X = X^O$, $Y = Y^O$, $Z = Z^O$, $X \leq Y$)} (Implication)
    {$X \leq Y$} (Strengthening)

    if Y > Z then
        {($X = Y^O$, $Y = X^O$, $Z = Z^O$, $X < Y$, $Y > Z$) or
         ($X = X^O$, $Y = Y^O$, $Z = Z^O$, $X \leq Y$, $Y > Z$)}
      (Condition)

        S(Y, Z);
        {($X = Y^O$, $Y = Z^O$, $Z = X^O$, $X < Z$, $Y < Z$) or
         ($X = X^O$, $Y = Z^O$, $Z = Y^O$, $X \leq Z$, $Y < Z$)} (Swap)
    else
        {($X = Y^O$, $Y = X^O$, $Z = Z^O$, $X < Y$, $Y \leq Z$) or
         ($X = X^O$, $Y = Y^O$, $Z = Z^O$, $X \leq Y$, $Y \leq Z$)}
        (Inverse Condition)
        Null;
    end if;
    {($X = Y^O$, $Y = Z^O$, $Z = X^O$, $X < Z$, $Y < Z$) or
     ($X = X^O$, $Y = Z^O$, $Z = Y^O$, $X \leq Z$, $Y < Z$) or
     ($X = Y^O$, $Y = X^O$, $Z = Z^O$, $X < Y$, $Y \leq Z$) or
     ($X = X^O$, $Y = Y^O$, $Z = Z^O$, $X \leq Y$, $Y \leq Z$)} (Implication)
    {$(X \leq Z, Y < Z) \vee (X \leq Y, Y \leq Z)$} (Strengthening)

    if X > Y then
      {($X = Y^O$, $Y = Z^O$, $Z = X^O$, $X < Z$, $Y < Z$, $X > Y$) or
       ($X = X^O$, $Y = Z^O$, $Z = Y^O$, $X \leq Z$, $Y < Z$, $X > Y$) or
       ($X = Y^O$, $Y = X^O$, $Z = Z^O$, $X < Y$, $Y \leq Z$, $X > Y$) or
       ($X = X^O$, $Y = Y^O$, $Z = Z^O$, $X \leq Y$, $Y \leq Z$, $X > Y$)}
      (Condition)

      {($X = Y^O$, $Y = Z^O$, $Z = X^O$, $X < Z$, $Y < Z$, $X > Y$) or
       ($X = X^O$, $Y = Z^O$, $Z = Y^O$, $X \leq Z$, $Y < Z$, $X > Y$)}
        (Contradiction)

        S(X, Y);
      {($X = Z^O$, $Y = Y^O$, $Z = X^O$, $Y < Z$, $X < Z$, $X < Y$) or
```

```
       ($X = Z^O$, $Y = X^O$, $Z = Y^O$, $Y \le Z$, $X < Z$, $X < Y$)}
        (Swap)
    else
      {($X = Y^O$, $Y = Z^O$, $Z = X^O$, $X < Z$, $Y < Z$, $X \le Y$) or
       ($X = X^O$, $Y = Z^O$, $Z = Y^O$, $X \le Z$, $Y < Z$, $X \le Y$) or
       ($X = Y^O$, $Y = X^O$, $Z = Z^O$, $X < Y$, $Y \le Z$, $X \le Y$) or
       ($X = X^O$, $Y = Y^O$, $Z = Z^O$, $X \le Y$, $Y \le Z$, $X \le Y$)}
      (Inverse Condition)
        Null;
    end if;
    {($X = Z^O$, $Y = Y^O$, $Z = X^O$, $Y < Z$, $X < Z$, $X < Y$) or
     ($X = Z^O$, $Y = X^O$, $Z = Y^O$, $Y \le Z$, $X < Z$, $X < Y$) or
     ($X = Y^O$, $Y = Z^O$, $Z = X^O$, $X < Z$, $Y < Z$, $X \le Y$) or
     ($X = X^O$, $Y = Z^O$, $Z = Y^O$, $X \le Z$, $Y < Z$, $X \le Y$) or
     ($X = Y^O$, $Y = X^O$, $Z = Z^O$, $X < Y$, $Y \le Z$, $X \le Y$) or
     ($X = X^O$, $Y = Y^O$, $Z = Z^O$, $X \le Y$, $Y \le Z$, $X \le Y$)}
    (Implication)

    {($X = Z^O$, $Y = Y^O$, $Z = X^O$, $X < Y$, $Y < Z$) or
     ($X = Z^O$, $Y = X^O$, $Z = Y^O$, $X < Y$, $Y \le Z$) or
     ($X = Y^O$, $Y = Z^O$, $Z = X^O$, $X \le Y$, $Y < Z$) or
     ($X = X^O$, $Y = Z^O$, $Z = Y^O$, $X \le Y$, $Y < Z$) or
     ($X = Y^O$, $Y = X^O$, $Z = Z^O$, $X < Y$, $Y \le Z$) or
     ($X = X^O$, $Y = Y^O$, $Z = Z^O$, $X \le Y$, $Y \le Z$)}
    (Strengthening)

    {$X \le Y$, $Y \le Z$} (Strengthening)
    {Post-Condition := $X \le Y$, $Y \le Z$}
end T;
```