```
1  ---------- MODULE OT ----------
   Specification of OT (Operational Transformation) functions. It consists of the basic OT functions
   for two operations and more general ones involving operation sequences.
7  EXTENDS Op
8  ----------
   OT (Operational Transformation) functions. Naming convention: I for "Ins" and D for "Del".
13  The left "Ins" lins transformed against the right "Ins" rins.
14 XformII(lins, rins) ≜
15     IF lins.pos < rins.pos
16      THEN lins
17      ELSE IF lins.pos > rins.pos
18               THEN [lins EXCEPT !.pos = @ + 1]
19               ELSE IF lins.ch = rins.ch
20                        THEN Nop
21                        ELSE IF lins.pr > rins.pr
22                                 THEN [lins EXCEPT !.pos = @ + 1]
23                                 ELSE lins
24  The left "Ins" lins transformed against the right "Del" rdel.
25 XformID(ins, del) ≜
26     IF ins.pos < del.pos
27      THEN ins
28      ELSE [ins EXCEPT !.pos = @ − 1]
29  The left "Del" ldel transformed against the right "Ins" rins.
30 XformDI(del, ins) ≜
31     IF del.pos < ins.pos
32      THEN del
33      ELSE [del EXCEPT !.pos = @ + 1]
34  The left "Del" ldel transformed against the right "Del" rdel.
35 XformDD(ldel, rdel) ≜
36     IF ldel.pos < rdel.pos
37      THEN ldel
38      ELSE IF ldel.pos > rdel.pos
39               THEN [ldel EXCEPT !.pos = @ − 1]
40               ELSE Nop
41 ----------
   Transform the left operation lop against the right operation rop with appropriate OT function.
46 Xform(lop, rop) ≜
47     CASE lop = Nop ∨ rop = Nop → lop
48        □ lop.type = "Ins" ∧ rop.type = "Ins" → XformII(lop, rop)
49        □ lop.type = "Ins" ∧ rop.type = "Del" → XformID(lop, rop)
50        □ lop.type = "Del" ∧ rop.type = "Ins" → XformDI(lop, rop)
51        □ lop.type = "Del" ∧ rop.type = "Del" → XformDD(lop, rop)
52 ----------
   Iteratively/recursively transforms the operation op against an operation sequence ops.
```

```
RECURSIVE XformOpOps(_, _)
XformOpOps(op, ops) ≜
    IF ops = ⟨⟩
        THEN op
        ELSE XformOpOps(Xform(op, Head(ops)), Tail(ops))
```

Iteratively/recursively transforms the operation *op* against an operation sequence *ops*. Different from *XformOpOps*, *XformOpOpsX* maintains the intermediate transformed operation

```
RECURSIVE XformOpOpsX(_, _)
XformOpOpsX(op, ops) ≜
    IF ops = ⟨⟩
        THEN ⟨op⟩
        ELSE ⟨op⟩ ∘ XformOpOpsX(Xform(op, Head(ops)), Tail(ops))
```

Iteratively/recursively transforms the operation sequence *ops* against an operation *op*.

```
XformOpsOp(ops, op) ≜
    LET opX ≜ XformOpOpsX(op, ops)
    IN  [i ∈ 1 .. Len(ops) ↦ Xform(ops[i], opX[i])]
```

* Modification History
* Last modified Sun *Jun* 24 18:09:53 *CST* 2018 by *hengxin*
* Created Sun *Jun* 24 15:57:48 *CST* 2018 by *hengxin*