```
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 OpOperators, AdditionalSetOperators

9  ----------

   OT (Operational Transformation) functions.

   Naming convention: I for "Ins" and D for "Del".

   The left "Ins" lins transformed against the right "Ins" rins.
19 XformII(lins, rins) ≜
20     IF lins.pos < rins.pos
21      THEN lins
22      ELSE IF lins.pos > rins.pos
23               THEN [lins EXCEPT !.pos = @ + 1]
24               ELSE IF lins.ch = rins.ch
25                        THEN Nop
26                        ELSE IF lins.pr > rins.pr
27                                 THEN [lins EXCEPT !.pos = @ + 1]
28                                 ELSE lins

   The left "Ins" ins transformed against the right "Del" del.
33 XformID(ins, del) ≜
34     IF ins.pos ≤ del.pos
35      THEN ins
36      ELSE [ins EXCEPT !.pos = @ − 1]

   The left "Del" del transformed against the right "Ins" ins.
41 XformDI(del, ins) ≜
42     IF del.pos < ins.pos
43      THEN del
44      ELSE [del EXCEPT !.pos = @ + 1]

   The left "Del" ldel transformed against the right "Del" rdel.
49 XformDD(ldel, rdel) ≜
50     IF ldel.pos < rdel.pos
51      THEN ldel
52      ELSE IF ldel.pos > rdel.pos
53               THEN [ldel EXCEPT !.pos = @ − 1]
54               ELSE Nop
55 ----------

   Transform the left operation lop against the right operation rop with appropriate OT function.
60 Xform(lop, rop) ≜
61     CASE lop = Nop ∨ rop = Nop → lop
62       □ lop.type = "Ins" ∧ rop.type = "Ins" → XformII(lop, rop)
```

```
63          □ lop.type = "Ins" ∧ rop.type = "Del" → XformID(lop, rop)
64          □ lop.type = "Del" ∧ rop.type = "Ins" → XformDI(lop, rop)
65          □ lop.type = "Del" ∧ rop.type = "Del" → XformDD(lop, rop)
66  ├──────────────────────────────────────────────────────────────────┤
```

Generalized OT functions on operation sequences.

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

```
75  RECURSIVE XformOpOps(_, _)
76  XformOpOps(op, ops) ≜
77      IF ops = ⟨⟩
78          THEN op
79          ELSE XformOpOps(Xform(op, Head(ops)), Tail(ops))
```

Iteratively/recursively transforms the operation op against an operation sequence ops. Being different from $XformOpOps$, $XformOpOpsX$ maintains the intermediate transformed operation

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

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

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

Iteratively/recursively transforms an operation sequence $ops1$ against another operation sequence $ops2$.

See also Definition 2.13 of the paper "Imine @ $TCS06$".

```
107 RECURSIVE XformOpsOps(_, _)
108 XformOpsOps(ops1, ops2) ≜
109     IF ops2 = ⟨⟩
110      THEN ops1
111      ELSE XformOpsOps(XformOpsOp(ops1, Head(ops2)), Tail(ops2))
112 └──────────────────────────────────────────────────────────────────┘
```

* Modification History
* Last modified *Tue Aug* 28 15:52:04 *CST* 2018 by *hengxin*
* Created Sun *Jun* 24 15:57:48 *CST* 2018 by *hengxin*