MODULE *BlockDag*

In this specification we define notions on *DAGs* useful for DAG-based consensus protocols (which build *DAGs* of blocks)

EXTENDS *FiniteSets*, *Sequences*, *Integers*, *Utils*, *Digraph*, *TLC*

CONSTANTS
N The set of all network nodes (not *DAG* nodes)
, R The set of rounds
, $Leader(\_)$ operator mapping each round to its leader

For our purpose of checking safety and liveness, *DAG* vertices just consist of a node and a round.

$V \triangleq N \times R$ the set of possible *DAG* vertices
$Node(v) \triangleq v[1]$
$Round(v) \triangleq$ IF $v = \langle\rangle$ THEN 0 ELSE $v[2]$ accomodates $\langle\rangle$ as default value

Next we define leader vertices:

$LeaderVertex(r) \triangleq$ IF $r > 0$ THEN $\langle Leader(r), r\rangle$ ELSE $\langle\rangle$
$IsLeader(v) \triangleq LeaderVertex(Round(v)) = v$
$Genesis \triangleq \langle\rangle$
ASSUME $IsLeader(Genesis)$ this should hold

OrderSet(S) arbitrarily order the members of the set S. Note that, in TLA+, CHOOSE is deterministic but arbitrary choice, *i.e.* CHOOSE $e \in S$: TRUE is always the same e if S is the same

RECURSIVE $OrderSet(\_)$
$OrderSet(S) \triangleq$ IF $S = \{\}$ THEN $\langle\rangle$ ELSE
    LET $e \triangleq$ CHOOSE $e \in S$: TRUE
    IN $Append(OrderSet(S \setminus \{e\}), e)$

PreviousLeader(*dag*, r) returns the leader vertex in *dag* whose round is the largest but smaller than r. We assume that *dag* contains at least the genesis vertex.

$PreviousLeader(dag, r) \triangleq$ CHOOSE $l \in Vertices(dag)$:
    $\land\ IsLeader(l)$
    $\land\ Round(l) = Max(\{Round(l2) : l2 \in$
        $\{l2 \in Vertices(dag) : IsLeader(l2) \land Round(l2) < r\}\})$

Linearize a *DAG* by repeatedly linearizing the causal past of each successive leader. In a real blockchain we should use a topological ordering, but, for the purpose of ensuring agreement, an arbitrary ordering (as provided by *OrderSet*) is fine. This assume a *DAG* where all paths end with the *Genesis* vertex.

RECURSIVE $Linearize(\_, \_)$
$Linearize(dag, l) \triangleq$ IF $Vertices(dag) = \{\langle\rangle\}$ THEN $\langle\rangle$ ELSE
    LET $dagOfL \triangleq SubDag(dag, \{l\})$
        $prevL \triangleq PreviousLeader(dagOfL, Round(l))$
        $dagOfPrev \triangleq SubDag(dag, \{prevL\})$

$$
\begin{array}{ll}
 & remaining \triangleq Vertices(dagOfL) \setminus Vertices(dagOfPrev) \\
\text{IN} & Linearize(dagOfPrev,\ prevL) \circ OrderSet(remaining \setminus \{l\}) \circ \langle l \rangle
\end{array}
$$

$Compatible(s1,\ s2) \triangleq$ whether the sequence $s1$ is a prefix of the sequence $s2$, or vice versa

$$\forall\, i \in 1 \,..\, Min(\{Len(s1),\ Len(s2)\}) : s1[i] = s2[i]$$