

```
01 pop :: (DiceChoice, DiceVals)
02     -> Maybe ((Bool, Integer), (DiceChoice, DiceVals))
```

```
03 pop ([], []) = Nothing
04 pop (chosen:choices, v:vs) = Just ((chosen, v),
05     (choices, vs))
06 pop (_:_, []) = error "Invariant violated: missing val
07 pop (Tl._:_ ) = error "Invariant violated: missing
08     choice"
```

```
10 allRolls :: (DiceChoice, DiceVals)
11           -> Integer
12           -> [ (DiceVals, Integer) ]
13 allRolls t n = [ (vals, n-1) | vals <- allRollsNoN t ]
14
15 allRollsNoN :: (DiceChoice, DiceVals) -> [ DiceVals ]
16
```

```
01 type DiceTurn = (DiceChoice, DiceVals)
02
03 pop :: DiceTurn
04     -> Maybe ((Bool, Integer), DiceTurn)
05
06 pop ([], []) = Nothing
07 pop (chosen:choices, v:vs) = Just ((chosen, v),
08     (choices, vs))
09 pop (_:_, []) = error "Invariant violated: missing val
10 pop (Tl._:_ ) = error "Invariant violated: missing
11     choice"
12
13 allRolls :: DiceTurn
14           -> Integer
15           -> [ (DiceVals, Integer) ]
16 allRolls t n = [ (vals, n-1) | vals <- allRollsNoN t ]
17
18 allRollsNoN :: DiceTurn -> [ DiceVals ]
```