$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	R_1	R_2	R_3	R_4	R_5	KB
false	false	false	false	false	false	false	true	true	true	true	false	false
false	false	false	false	false	false	true	true	true	false	true	false	false
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
false	true	false	false	false	false	false	true	true	false	true	true	false
false	true	false	false	false	false	true	true	true	true	true	true	*true*
false	true	false	false	false	true	false	true	true	true	true	true	*true*
false	true	false	false	false	true	true	true	true	true	true	true	*true*
false	true	false	false	true	false	false	true	false	false	true	true	false
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
true	true	true	true	true	true	true	false	true	true	false	true	false

Figure 7.9 A truth table constructed for the knowledge base given in the text. *KB* is true if R_1 through R_5 are true, which occurs in just 3 of the 128 rows (the ones underlined in the right-hand column). In all 3 rows, $P_{1,2}$ is false, so there is no pit in [1,2]. On the other hand, there might (or might not) be a pit in [2,2].

---

function TT-ENTAILS?(*KB*, α) **returns** *true* or *false*
  inputs: *KB*, the knowledge base, a sentence in propositional logic
        α, the query, a sentence in propositional logic

  symbols ← a list of the proposition symbols in *KB* and α
  return TT-CHECK-ALL(*KB*, α, *symbols*, { })

function TT-CHECK-ALL(*KB*, α, *symbols*, *model*) **returns** *true* or *false*
  if EMPTY?(*symbols*) **then**
      if PL-TRUE?(*KB*, *model*) **then return** PL-TRUE?(α, *model*)
      else return *true*   // when KB is false, always return true
  else
      P ← FIRST(*symbols*)
      rest ← REST(*symbols*)
      return (TT-CHECK-ALL(*KB*, α, *rest*, *model* ∪ {*P* = *true*})
            and
            TT-CHECK-ALL(*KB*, α, *rest*, *model* ∪ {*P* = *false*}))

Figure 7.10 A truth-table enumeration algorithm for deciding propositional entailment. (TT stands for truth table.) PL-TRUE? returns *true* if a sentence holds within a model. The variable *model* represents a partial model—an assignment to some of the symbols. The keyword **and** here is an infix operator in the pseudocode programming language, not an operator in propositional logic; it returns *true* if either argument is true.