EXERCISES IN MATHEMATICS

Series F, No. 1: Answers

Sets, Logic and Venn Diagrams

1. Prove De Morgan's rules using a Venn diagram.

2. Substitute the symbols + for addition and × for multiplication in place of ∪ and ∩ respectively in the statements (15)–(18) of the laws of Boolean algebra. Determine whether the resulting statements concerning the arithmetic operations are true or false. Attempt to give a complete statement of the rules of arithmetic.

Answer. On substituting × for ∩ and + for ∪ we derive the following statements:

(15)	Commutative law:	$A + B = B + A,$	*True*
		$A \times B = B \times A,$	*True*
(16)	Associative law:	$(A + B) + C = A + (B + C),$	*True*
		$(A \times B) \times C = A \times (B \times C),$	*True*
(17)	Distributive law:	$A \times (B + C) = (A \times B) + (A \times C),$	*True*
		$A + (B \times C) = (A + B) \times (A + C),$	*False*
(18)	Idempotency law:	$A + A = A,$	*False*
		$A \times A = A.$	*False*

The axioms of arithmetic may be stated as follows:

The set of real number is closed under the binary operations of addition + and multiplication × such that, for any numbers x, y, z, the following are true:

(i)	Commutativity :	$x + y = y + x,$
(ii)		$x \times y = y \times x,$
(iii)	Associativity :	$(x + y) + z = x + (y + z),$
(iv)		$(x \times y) \times z = x \times (y \times z),$
(v)	Distributivity :	$x \times (y + z) = (x \times y) + (x \times z),$

(*)		$x + (y \times z) \neq (x + y) \times (x + z)$,
(vi)	Unity:	There exists a number 1 such that $1 \times x = x$ for every x.
(vii)	Zero:	There exists a number 0 such that $x + 0 = x$ for every x.
(viii)	Negatives:	For every $x \neq 0$, there exist an unique negative $-x$ such that $x + (-x) = 0$.
(ix)	Inverses:	For every $x \neq 0$, there exist an unique inverse x^{-1} such that $x \times x^{-1} = 1$.

3. Evaluate the following expressions:

(a) $A \cup (B^c \cup A)^c$, (c) $A \cap (B^c \cap A)^c$,

(b) $A \cap (B \cup A^c)$, (d) $A \cup (A \cap B)^c$.

Answer.

(a)
$$\begin{aligned} A \cup (B^c \cup A)^c &= A \cup (B \cap A^c) && \textit{De Morgan} \\ &= (A \cup B) \cap (A \cup A^c) && \textit{Distributivity} \\ &= (A \cup B) \cap S = A \cup B && \textit{By } (20), \text{(i), (iv).} \end{aligned}$$

(b)
$$\begin{aligned} A \cap (B \cup A^c) &= (A \cap B) \cup (A \cap A^c) && \textit{Distributivity} \\ &= (A \cap B) \cup \emptyset && \textit{By } (20), \text{(ii),} \\ &= A \cap B && \textit{By } (20), \text{(v).} \end{aligned}$$

(c)
$$\begin{aligned} A \cap (B^c \cap A)^c &= A \cap (B \cup A^c) && \textit{De Morgan} \\ &= (A \cap B) && \textit{See above} \end{aligned}$$

(d)
$$\begin{aligned} A \cup (A \cap B)^c &= A \cup (A^c \cup B^c) && \textit{De Morgan} \\ &= (A \cup A^c) \cup B^c && \textit{Associativity} \\ &= S \cup B^c = S && \textit{By } (20), \text{(i) (iii).} \end{aligned}$$

4. A statement in Boolean algebra may be transformed into a statement in propositional logic. Consider the following:

(a) $(A \cup B)^c = A^c \cap B^c$ and (b) $(A \cap B)^c = A^c \cup B^c$.

Let A represent the circumstance of being a man and let B be the condition of having less than forty years of age. Let $\cap$ stand for the conjunction "and" and let $\cup$ stand for the logical "or". Also let A^c be rendered as not A. Using these elements, attempt to construct two sentences in simple English which correspond to the Boolean expressions above.

Answer. (a) A person who is not either male or under forty is neither male nor under forty. (The person is therefore a woman over forty). (b) A person who is not both male and under forty is either female or over forty.

Our use of language may be symbolised by writing

(a) not(either A or B) $\iff$ (neither A nor B),

(b) not(both A and B) $\iff$ either (not A) or (not B).

5. The Pascal computer language defines the operators **not**, **and** and **or**. Two of these operators are used in the following lines of code which instruct an algorithm for plotting a graph to draw line segments only within vertical limits:

if (**not**(y >= *upperBound*)) **and** (**not**(y <= *lowerBound*)) **then**
LineTo(x, y);

Rewrite the first line of code (a) using a **not** and an **or** just once and (b) using a single **and**.

Answer. The following three clauses are equivalent:

if (*not*(y >= *upperBound*)) **and** (*not*(y <= *lowerBound*)) **then**
LineTo(x, y);

if **not** ((y >= *upperBound*) **or** (y <= *lowerBound*)) **then**
LineTo(x, y);

if (y < *upperBound*) **and** (y > *lowerBound*) **then**
LineTo(x, y);