Sample Questions

Discrete Structures

(Midterm scheduled for Wednesday, February 24, 2021)

You must justify all your answers to recieve full credit

---

1 Boolean Expressions

Truth tables, logical equivalences, set operations, Venn diagrams.

1.(a). Given a statement in English and atomic propositions, write its Boolean expression.

1.(b). Given a Boolean expression, fill in missing values in its truth table.

1.(c). Given a Boolean expression equivalently transform it using Boolean identities.

1.(d). Given a Boolean expression, prove or disprove a tautology.

1.(e). Given a truth table, create a DNF or a CNF for it (and vice versa).

1.(f). Given a set expression, shade the regions in a Venn diagram that belong to it.

1.(g). Given two set expressions prove or disprove set identity or subset relation.

1.(a). Rewrite both English sentences (P_1 and P_2) as Boolean expressions using the given propositional variables.
P_1 := "If it is not snowing nor raining, then Gilbert has to tend his garden."
Propositional variables in P_1:
S := "It is snowing"
R := "It is raining"
G := "Gilbert has to tend his garden"
P_2 := "Jane will be a candidate in the elections regardless of whether she has a chance to be elected or not."
Propositional variables in P_2:
J := "Jane will be a candidate in the elections."
E := "She has a chance to be elected."

1.(b). Consider the following Boolean expression $E = p \to q \to r$. Fill in the missing 4 slots in its truth table.

p	q	r	E
True	True	True	...
True	True	False	...
True	False	True	...
True	False	False	...
False	True	True	True
False	True	False	True
False	False	True	True
False	False	False	True

1.(c). Transform the Boolean expression $p \vee q \wedge r$ into a logically equivalent one, using the same propositional letters p, q, r and two connectors: implication $\to$ and negaton $\neg$.

1.(d). Prove or disprove that the following is a tautology:

$$(p \to q \to r) \leftrightarrow \neg(p \wedge q \wedge \neg r).$$

1.(e). Build the truth table for the following CNF:

$$f(A,B,C) = (A \vee B \vee C) \wedge (A \vee B \vee \neg C) \wedge (A \vee \neg B \vee C).$$

1.(f). Let A, B, C be subsets in the same universe U. Draw a Venn diagram for these sets and shade the region corresponding to the set $S = A \oplus (B \oplus C)$.

1.(g). Let A and B be two arbitrary subsets of the same universe U. Prove or disprove the following set identity:

$$A \oplus (A \cap B) = A - B.$$

2 Quantifiers

Predicates, quantifiers, precedence, simple proofs.

2.(a). Given an English sentence and predicates, write its predicate expression.

2.(b). Given a predicate expression, restore parentheses, identify free/bound variables.

2.(c). Given a predicate expression, write its negation (De Morgan laws etc.).

2.(d). Given truth tables for predicates, evaluate nested quantifier expressions.

2.(e). Given a description of a set, define it in a set-builder notation.

2.(f). Given a pseudocode, write the predicate expression that it computes.

2.(a). Rewrite the statement as predicate expression.
Statement: "No judges are crooks; but there are judges who are elderly yet sharp-witted."
Domain: H is the set of all humans.
Predicates $J(x)$, $C(x)$, $E(x)$, $S(x)$ from H to $\{True, False\}$:
Predicate $J(x)$ is true iff human x is a judge.
Predicate $C(x)$ is true iff human x is a crook.
Predicate $E(x)$ is true iff human x is elderly.
Predicate $S(x)$ is true iff human x is sharp-witted.

2.(b). Consider the following predicate expression:

$$\neg\exists y\ (\neg Q(x,z) \vee P(x,y) \wedge P(y,x)) \wedge \forall x\ (Q(y,z) \to \neg P(x,y) \to P(y,x)). \qquad (1)$$

Rewrite the equation (2); insert all the parentheses so that **every** subexpression serving as an argument for Boolean operations and quantifiers is in parentheses. Use the rules for precedence for Boolean operators:
Rule1: Decreasing order of precedence: $\neg$, $\wedge$, $\vee$, $\to$, $\leftrightarrow$
Rule2: Both quantifiers ($\forall$, $\exists$) have the same (highest) precedence as $\neg$. **Rule3:** Implication and equivalence are right-associative; conjunction and disjunction are left-associative.

Also underline those variables which are bound. Leave all the unbound variables without underlining.

2.(c). Simplify the expression with negation so that negation is only applied to individual predicates or propositional variables (rather than larger subexpressions or quantifiers):

$$\neg\left(\exists x \forall y\, (P(x,y) \rightarrow Q(x,y)) \vee \exists y \forall x (\neg P(x,y) \wedge Q(x,y))\right)$$

2.(d). Check the following nested predicate statement on the predicates defined below:

$$\forall x \in \mathbf{N}\ \exists y \in \mathbf{N}\ ((y > x) \wedge \neg(P(y) \rightarrow Q(y))). \qquad (2)$$

$P(x)$ defined on $\mathbf{N} = \{0, 1, 2, \ldots\}$; it is `True` iff x is full square.
$Q(x)$ defined on $\mathbf{N}$; it is true iff the last digit of x is not 1 or 2.

Determine whether the predicate expression (3) is `True` or `False`; explain your answer.

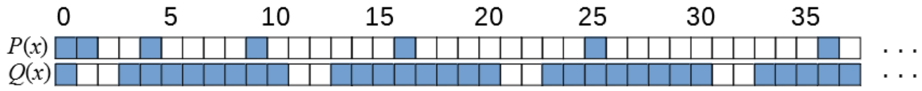

Figure 1: Predicate values that equal to `True` are shaded; `False` are white.

Both predicates are defined on the infinite set $\mathbf{N} = \{0, 1, 2, 3, \ldots\}$ of natural numbers. Some initial values are shown in Figure 2.

2.(e). The following expression:

$$\{x \in U \mid P(x)$$

is the regular set-builder notation: It denotes a subset of the universe U consisting of all those x that make the predicate $P(x)$ true.

Use this set-builder notation to describe the set of all full squares that have the last digit equal to 6 (namely, the set $\{16, 36, 196, 256, \ldots\}$). You can use the set of all integers $\mathbf{Z}$ as the universe, the arithmetic operations (including $(a \bmod b)$, the remainder dividing a by b), Boolean operations and quantifiers.

2.(f). Assume that there are two lists $A[1..100]$ and $B[1..100]$ containing 100 integer numbers each. They are passed to the function computing MyPredicate(A, B) on these lists (see pseudocode below). This function returns value `True` or `False` depending on the numbers in lists A and B.

```
MyPredicate(A : IntegerList, B : IntegerList)
val := True
for i := 1 to 100
    if A[i] > 0
        if (not B[i] > 0)
            val := False
return val
```

Write the predicate expression that is computed by this function. The expression can use quantifiers $\forall i \in \{1, \ldots, 100\}$ and $\exists i \in \{1, \ldots, 100\}$; all Boolean connectors, references to array elements $A[i]$ and $B[i]$ as well as equality and inequality predicates.