Practice Exam 1

Tables Provided on Exam 1

Table 1.5.1: Laws of propositional logic.

Idempotent laws:	$p \lor p \equiv p$	$p \land p \equiv p$
Associative laws:	$(p \lor q) \lor r \equiv p \lor (q \lor r)$	$(p \land q) \land r \equiv p \land (q \land r)$
Commutative laws:	$p \lor q \equiv q \lor p$	$p \land q \equiv q \land p$
Distributive laws:	$p \lor (q \land r) \equiv (p \lor q) \land (p \lor r)$	$p \land (q \lor r) \equiv (p \land q) \lor (p \land r)$
Identity laws:	$p \lor F \equiv p$	$p \land T \equiv p$
Domination laws:	$p \land F \equiv F$	$p \lor T \equiv T$
Double negation law:	$\neg\neg p \equiv p$	
Complement laws:	$p \land \neg p \equiv F$ $\neg T \equiv F$	$p \lor \neg p \equiv T$ $\neg F \equiv T$
De Morgan's laws:	$\neg(p \lor q) \equiv \neg p \land \neg q$	$\neg(p \land q) \equiv \neg p \lor \neg q$
Absorption laws:	$p \lor (p \land q) \equiv p$	$p \land (p \lor q) \equiv p$
Conditional identities:	$p \to q \equiv \neg p \lor q$	$p \leftrightarrow q \equiv (p \to q) \land (q \to p)$

Table 1.12.1: Rules of inference known to be valid arguments.

Rule of inference	Name
p $p \to q$ $\therefore q$	Modus ponens
$\neg q$ $p \to q$ $\therefore \neg p$	Modus tollens
p $\therefore p \lor q$	Addition
$p \land q$ $\therefore p$	Simplification

Rule of inference	Name
p q $\therefore p \land q$	Conjunction
$p \to q$ $q \to r$ $\therefore p \to r$	Hypothetical syllogism
$p \lor q$ $\neg p$ $\therefore q$	Disjunctive syllogism
$p \lor q$ $\neg p \lor r$ $\therefore q \lor r$	Resolution

Table 1.13.1: Rules of inference for quantified stateme

Rule of Inference	Name
c is an element (arbitrary or particular) ∀x P(x) ∴ P(c)	Universal instantiation
c is an arbitrary element P(c) ∴ ∀x P(x)	Universal generalization
∃x P(x) ∴ (c is a particular element) ∧ P(c)	Existential instantiation*
c is an element (arbitrary or particular) P(c) ∴ ∃x P(x)	Existential generalization

Table 3.6.1: Set identities.

Name	Identities	
Idempotent laws	A ∪ A = A	A ∩ A = A
Associative laws	(A ∪ B) ∪ C = A ∪ (B ∪ C)	(A ∩ B) ∩ C = A ∩ (B ∩ C)
Commutative laws	A ∪ B = B ∪ A	A ∩ B = B ∩ A
Distributive laws	A ∪ (B ∩ C) = (A ∪ B) ∩ (A ∪ C)	A ∩ (B ∪ C) = (A ∩ B) ∪ (A ∩ C)
Identity laws	A ∪ ∅ = A	A ∩ U = A
Domination laws	A ∩ ∅ = ∅	A ∪ U = U
Double Complement law	$\overline{\overline{A}} = A$	
Complement laws	$A \cap \overline{A} = \varnothing$ $\overline{U} = \varnothing$	$A \cup \overline{A} = U$ $\overline{\varnothing} = U$
De Morgan's laws	$\overline{A \cup B} = \overline{A} \cap \overline{B}$	$\overline{A \cap B} = \overline{A} \cup \overline{B}$
Absorption laws	A ∪ (A ∩ B) = A	A ∩ (A ∪ B) = A

Set Theory

Given:

A = { 1, {2}, {{3, 4}} }
For each of the following statements, state whether they are **true** or **false.**

a. 1 ∈ A
b. 1 ⊆ A
c. {2} ∈ A
d. {2} ⊆ A
e. {3, 4} ∈ A
f. {3, 4} ⊆ A
g. {{3,4}} ∈ A
h. {{3,4}} ⊆ A
i. ∅ ∈ A
j. ∅ ⊆ A

Solutions:

a. T. The number 1 is an element of set A. { 1, {2}, {{3, 4}} }
b. F. The subset relation ⊆ checks if every element of the set on the left is also an element of the set on the right. Here though, 1 is not a set (it would have to be {1}), and because of this the statement is false.
c. T. The set containing 2 is an element of A. { 1, {2}, {{3, 4}} }
d. F. The relation ⊆ is only checking if the elements within the left hand set are in A; the element in the set is just 2, while A contains the set {2}.
e. F. The set {{3,4}} is an element of A, but {3,4} is not.
f. F. As before, the relation ⊆ checks whether the elements within the left hand set are in A. Since 3 and 4 are not standalone elements of A (they are instead nested within 2 sets), this is false.
g. T. The set {{3,4}} is an element of A. { 1, {2}, {{3, 4}} }
h. F. To satisfy this, the set {3,4} would have to be a direct element of A. However, A only contains {{3,4}}.
i. F. An element is a distinct object belonging to a set, and because of this we cannot automatically classify the empty set as being an element of another set (like we can do with it being a subset). In this case A does not explicitly include {} (although it can), which makes this false.
j. T. The empty set has no elements, meaning that every element it contains is technically also present in every other set. Because of this, the empty set is always a subset of any set (including itself).

1.2 Let A = {1, 2, 3, 4}. Select the statement that is **false**.

a. ∅ ∈ P(A) The empty set is an included element of a power set

b. $\emptyset \subseteq P(A)$ The empty set is always a subset of another set
c. $\{2, 3\} \in A$ 2 and 3 are included individually, but not together as a set {2,3}
d. $\{2, 3\} \subseteq A$ Individually 2 and 3 are both in A, making this true

Solution:

C

Functions

Choose the property for which the function satisfies if well defined.

a. Neither one-to-one, nor onto

b. One-to-one, but not onto

c. Onto, but not one-to-one

d. one-to-one and onto

e. not well defined

Given a function whose domain is the set of all integers and whose target is the set of all positive integers:

a)

$f(x) \ = \ 2x + 1$

Solution: e. Example: if x=-1, then the output is not within the set of all positive integers.

b)

$f(x) \ = \ |x| \ + 1$

Solution: c.

Not one-to-one: f(x)=f(-x)

c)

$f(x) \ = \ x\text{^}2 \ + 1$

Solution: a.

Not one-to-one: f(x)=f(-x)

Not onto: cannot map to f(x)=9 with integer inputs

d)

$f(x) \ = \{x > 0 : 2x + 1 \ \wedge \ x <= 0 : \ - 2x\}$

Solution: e. x = 0 maps to 0, which is not a positive integer.

e)

$f(x) \ = \ x >= 0 \colon 2x + 1 \ \wedge \ x < 0 : \ - 2x + 2$

Solution: b. Can’t map to 2, so it’s not onto.

Proofs

4.1 Direct Proof

Prove that the product of two odd integers is an odd integer.

Proof.

Assume that integers m and n are odd integers. We will show that m*n is also an odd integer.

Integers m and n are odd integers

⇒ m = 2k+1 for some integer k and n = 2j + 1 for some integer j

⇒ m*n = (2k+1)(2j+1) for some integers k and j

⇒ = 4kj + 2k + 2j + 1

⇒ = 2(kj + k + j) + 1

Since j and k are both integers, then 2(kj + k + j) is also an integer. Since m*n can be expressed as 2 times an integer plus 1, m*n is an odd integer (2 times any integer yields an even result, and adding 1 will ensure it is odd).

4.2 Proof by Contrapositive

Prove that if n^2 is even, then n is even.

Consider the contraposition of the proposition which is ~q→~p. Show that if n is odd, then n^2 is odd.

Proof.

Assume the hypothesis of the implication to be true.

Assume that *n* is an odd integer.

Show the implication.

n is an odd integer

⇒ n = 2q+1 for some integer q

⇒ $n^2 = (2q + 1)^2$ for some integer q

⇒ $n^2 = 4q^2 + 4q + 1$ for some integer q

⇒ $n^2 = 2(2q^2 + 2q) + 1$ for some integer q and because q is an integer, then $(2q^2 + 2q)$ is also an integer,

⇒ $n^2 = 2j + 1$ for some integer j

Therefore n^2 is an odd integer that can be represented as 2j+1.

Having used a direct proof of the contrapositive statement, we can conclude that if n^2 is even, then n is even.

4.2 Proof by Contradiction

Prove by contradiction that if 3n+5 is odd, then n is even.

Proof.

Begin by assuming that 3n+5 is odd and n is *odd*.

Integer n is odd therefore n=2k+1 for some integer k.

substituting n=2k+1 for n in 3n+5,

3n+5 = 3(2k+1) + 5 for some integer k

= 6k +8 for some integer k, algebra

= 2(3k+4)

Let k' = (3k+4) for some integer k,

and since k is an integer, then k' = (3k+4) is also an integer.

= 2(k')

Further, since k'= (3k+4) is an integer, and 2 multiplied by an integer is an even integer, by definition 3(2k+1) + 5 is even.

Thus, 3n+5 is even and because this **contradicts** the assumption that 3n+5 is odd then n is odd; we have proven that if 3n+5 is odd, then n is even.

Number Systems Conversion

5.1 Decimal to 8-bit Two's Complement

$(-43)_{10}$

43 in binary = 00101011

Invert bits 11010100

11010100 + 1 = 11010101

$(11010101)_{\text{8-bit Two's Complement}}$

5.2 Binary to Hexadecimal

$(110011100)_2$

0001 = 1 decimal = 1 hexadecimal

1001 = 9 decimal = 9 hexadecimal

1100 = 12 decimal = C hexadecimal

Combine them: 19C

$(19C)_{16}$

Coding

```
    a
   b b
  c   c
 d     d
e       e
 d     d
  c   c
   b b
    a
```

Make a hollowed-out diamond made up of in-order alphabet letters. An example is below when `n=5`.

Solution:

```
#include <iostream>
using namespace std;

int main(){
   int userInput = 7;
   char letter = 'a';
   char space = ' ';

   int outsideSpaceLimit = userInput - 1;
   int insideSpaceLimit = 0;

   for (int i = 1; i < (userInput * 2); i++) {
       // outside spaces
       for (int j = 0; j < outsideSpaceLimit; j++) {
           cout << space;
       }
       // first letter

       cout << letter;

       //inside spaces
       for (int j = 1; j < insideSpaceLimit; j++) {
           cout << space;
       }

       //second letter
       if (i != 1 && i != ((userInput * 2) -1)){
           cout << letter;
       }

       if(i < userInput){
           letter++;
           insideSpaceLimit += 2;
           outsideSpaceLimit--;
       }
```

```
        else{
            letter--;
            insideSpaceLimit -= 2;
            outsideSpaceLimit++;
        }

        cout << endl;
    }


    return 0;
}
```

Challenge Question

For this question, you have to create the following Barn Door shape with `n=10`:

```
##########
#$      $#
# $    $ #
#  $  $  #
#   $$   #
#   $$   #
#  $  $  #
# $    $ #
#$      $#
##########
```

ONE answer (again, there are **many** ways to do this!):

```
#include <iostream>
using namespace std;

int main(){
    int input;

    char frame = '#';
    char planks = '$';
    char space = ' ';

    cout << "Enter positive integer: ";
    cin >> input;

    for (int i = 0; i < input; i++) { //rows
        for (int j = 0; j < input; j++) {
            // left or right or top or bottom
            if (j == 0 || j == input - 1 || i == 0 || i == input - 1){
                cout << frame;
            }
            // left diagonal going right || right diagonal going left
            else if(i == j || i + j == input - 1){
                cout << planks;
            }
            else{
                cout << space;
            }

        }
        cout << endl;
    }
```

```
    return 0;
}
```