1. **[2.5 marks]** Consider the following first-order logic formula:

$$\exists x\ (P(x) \rightarrow \forall y\ P(y))$$

If we consider the universe as the students in COL703, and interpret $P(x)$ as '*x likes pizza*', then the statement above means that *there is someone in the class who if (s)he likes pizza, then everyone in the class likes pizza.*

You are required to either give an interpretation that makes this formula false (you are free to use or adapt the interpretation given above), or prove that the formula is valid (and explain why the interpretation above is not contradictory).

Ans: We will prove that the formula is valid, by showing the unsatisfiability of the negation of the formula. The negated formula can be written as:

$$\begin{gathered}\neg\exists x\ (P(x) \rightarrow \forall y\ P(y)) \\ \forall x\ \neg(P(x) \rightarrow \forall y\ P(y)) \\ \forall x\ (P(x) \wedge \neg\forall y\ P(y)) \\ \forall x\ (P(x) \wedge \exists y\ \neg P(y)) \\ \forall x\ P(x) \wedge \exists y\ \neg P(y))\end{gathered}$$

We can skolemize the second conjunct to get $\neg P(a)$, for some constant symbol 'a'. Now, unsatisfiability can be proved using ground resolution, by producing the ground instance $P(a)$ from the first conjunct, and resolving it with $\neg P(a)$, the skolemized form of the second conjunct.

The interpretation given above is fine because the formula simply says that either everyone likes pizza or there is someone who doesn't like pizza, which is not contradictory at all.

2. **[3.5 marks]** We claim that ROBDDs can be used to count all solutions to a propositional satisfiability problem. Write down an algorithm that computes the number of satisfying truth assignments of a propositional formula, given its ROBDD.

Ans: Here is an algorithm to compute the number of satisfying truth assignments of a node u in an ROBDD.

function $count(u)$
  if $u = 0$ **then** $res \leftarrow 0$
  else if $u = 1$ **then** $res \leftarrow 1$
  else $res \leftarrow\ 2^{var(low(u)) - var(u) - 1} * count(low(u))$
    $+\ 2^{var(high(u)) - var(u) - 1} * count(high(u))$
  return res
end $count$

return $2^{var(u)-1} * count(u)$

Note that *var* of a node returns the variable index of the variable at that node in the ROBDD. We assume that the n variables are numbered from 1 to n, and $var(\boxed{0}) = var(\boxed{1}) = n + 1$.

3. **[2 marks]** Use the axioms (A1-A3) given below, and Modus Ponens, to derive $\alpha \to \alpha$.

- $\alpha \to (\beta \to \alpha)$ (A1)
- $(\neg\beta \to \neg\alpha) \to ((\neg\beta \to \alpha) \to \beta)$ (A2)
- $(\alpha \to (\beta \to \gamma)) \to ((\alpha \to \beta) \to (\alpha \to \gamma))$ (A3)

Ans:

1.	$(p \supset ((p \supset p) \supset p)) \supset ((p \supset (p \supset p)) \supset (p \supset p))$	Instance of (A3)
2.	$p \supset ((p \supset p) \supset p)$	Instance of (A1)
3.	$(p \supset (p \supset p)) \supset (p \supset p))$	From 1 and 2 by MP
4.	$p \supset (p \supset p)$	Instance of (A1)
5.	$p \supset p$	From 3 and 4 by MP

Note that I have been lazy here and simply copied this proof from the notes. This is fine, except for two things that are changed: p is written in place of α, and $\supset$ symbol has been used in place of $\to$.

4. Let F and G be propositional logic formulas such that $F \models G$.

(a) **[1.5 marks]** Show that if F and G have no variables in common, then either F is unsatisfiable, or G is valid.

Ans: We will show that if G is not valid, then F is not satisfiable. On the contrary, suppose that G is not valid but F is satisfiable. This means we have an assignment A_1 that satisfies $\neg G$, and another assignment A_2 that satisfies F. Since A_1 and A_2 do not have any variables in common, we can combine them into a single assignment A. For any variable v, A copies the assignment that v has in A_1 or A_2 (this is possible without a conflict because we know that v appears in only one of A_1, A_2).

Now the assignment A makes G false (because A and A_1 are the same as far as the variables in G are concerned), and it makes F true (once again, because A and A_2 are the same as far as the variables in F are concerned). This contradicts that $F \models G$, which essentially says that any assignment that makes F true must make G true.

(b) **[2.5 marks]** Now let F and G be arbitrary formulas. Show how you will construct a formula H, mentioning only propositional variables common to F and G, such that $F \models H$ and $H \models G$. [*Note that such a formula H always exists. In order to convince yourself, you may use induction on the number of propositional atoms in F that are not in G. This is not a part of the question, though.*]

Ans: We start with the formula F and consider all the variables that are in F but not in G. Let us denote them as $v_1, \ldots, v_k$. We start with $H_0 = F$, and obtain H_1 to H_k one by one, where H_i is $H_{i-1}[true/v_i] \vee H_{i-1}[false/v_i]$.

We take H to be H_k. [*The correctness argument was not asked for, but we can see that the construction works. H_k only refers to variables that are common to both F and G, because we have eliminated the ones that were in F but not in G. Further, any assignment that makes H_i true, also makes H_{i+1} true. Because the assignment must have a value for the variable that was* *eliminated, and no matter what the value was, it will make one of the disjuncts true. Lastly, every assignment that makes H_k true, can be extended into an assignment that makes F true. So the extended assignment must make G true. But note that the extensions weren't really required for G because those variables do not appear in G. So, the original assignment that made H_k true must also make G true.*]

5. **[3 marks]** Prove that a set of Horn clauses is unsatisfiable if and only if it has a positive unit resolution refutation. A *positive unit resolution refutation* is a resolution refutation containing only positive unit inferences. A *positive unit inference* is a resolution inference in which one of the hypotheses is a unit clause containing a positive literal only.

   Ans: Let Γ be an unsatisfiable set of Horn clauses. Γ must contain at least one positive unit clause $\{p\}$, since otherwise the truth assignment that assigned *false* to all variables would satisfy Γ. By resolving $\{p\}$ against all clauses containing $\neg p$, and then discarding all clauses which contain p or $\neg p$, we get a smaller unsatisfiable set of Horn clauses. Iterating this yields the desired positive unit refutation.

6. **[2 marks]** Show that $\Box(\phi \wedge \psi) \leftrightarrow (\Box\phi \wedge \Box\psi)$ is a valid modal logic formula.

   Ans: We will show that if any world w of any model M makes $\Box(\phi \wedge \psi)$ true, then it also makes $(\Box\phi \wedge \Box\psi)$ true, and vice-versa.

   Suppose $M, w \vDash \Box(\phi \wedge \psi)$. Then, for every world w' such that wRw', $M, w' \vDash (\phi \wedge \psi)$. This means that $M, w' \vDash \phi$ and $M, w' \vDash \psi$. However, since the choice of w' was arbitrary, $M, w' \vDash \phi$ and $M, w' \vDash \psi$ hold for all w' where wRw'. This implies that $M, w \vDash \Box\phi$ and $M, w \vDash \Box\psi$. Therefore, $M, w \vDash (\Box\phi \wedge \Box\psi)$.

   Conversely, suppose $M, w \vDash (\Box\phi \wedge \Box\psi)$. Clearly, $M, w \vDash \Box\phi$ and $M, w \vDash \Box\psi$. This means that, for every w' such that wRw', $M, w' \vDash \phi$ and $M, w' \vDash \psi$. Therefore, $M, w' \vDash (\phi \wedge \psi)$. Once again, since the choice of w' was arbitrary, this must hold for all w' where wRw'. So, $M, w \vDash \Box(\phi \wedge \psi)$.

7. **[3.5 marks]** Let us consider the sentences that follow. Everyone who loves all animals is loved by someone. Anyone who kills an animal is loved by no one. Ramesh loves all animals. Either Ramesh or Curiosity killed the cat, who is named Molly. Did Curiosity kill Molly? Use resolution to answer this.

   Ans: Let us convert these sentences into first-order logic formulas.

   Everyone who loves all animals is loved by someone.
   $\forall x\ ((\forall y\ (\mathit{Animal}(y) \rightarrow \mathit{Loves}(x, y))) \rightarrow \exists y\ \mathit{Loves}(y, x))$
   $\forall x\ (\neg(\forall y\ (\mathit{Animal}(y) \rightarrow \mathit{Loves}(x, y))) \vee \exists y\ \mathit{Loves}(y, x))$ (rewriting outermost $\rightarrow$ using $\neg$ and $\vee$)
   $\forall x\ (\exists y\ \neg(\mathit{Animal}(y) \rightarrow \mathit{Loves}(x, y)) \vee \exists y\ \mathit{Loves}(y, x))$ (changing $\forall$ to $\exists$)
   $\forall x\ (\exists y\ (\mathit{Animal}(y) \wedge \neg\mathit{Loves}(x, y)) \vee \exists y\ \mathit{Loves}(y, x))$ (rewriting $\rightarrow$, taking $\neg$ inside)
   $\forall x\ ((\mathit{Animal}(F(x)) \wedge \neg\mathit{Loves}(x, F(x))) \vee \mathit{Loves}(G(x), x))$ (skolemizing)

   Converting to CNF form (dropping universal quantifiers and distributing $\vee$ over $\wedge$), we get two clauses:
   $\{\mathit{Animal}(F(x)), \mathit{Loves}(G(x), x)\}$, and $\{\neg\mathit{Loves}(x, F(x)), \mathit{Loves}(G(x), x)\}$ (call them 1a, 1b)

   Anyone who kills an animal is loved by no one.
   $\forall x\ ((\exists z\ (\mathit{Animal}(z) \wedge \mathit{Kills}(x, z))) \rightarrow \forall y\ \neg\mathit{Loves}(y, x))$
   $\forall x\ (\neg(\exists z\ (\mathit{Animal}(z) \wedge \mathit{Kills}(x, z))) \vee \forall y\ \neg\mathit{Loves}(y, x))$ (rewriting $\rightarrow$)
   $\forall x\ ((\forall z\ (\neg\mathit{Animal}(z) \vee \neg\mathit{Kills}(x, z))) \vee \forall y\ \neg\mathit{Loves}(y, x))$ (rewriting $\exists$ as $\forall$, taking $\neg$ inside)
   $\forall x \forall z \forall y\ (\neg\mathit{Animal}(z) \vee \neg\mathit{Kills}(x, z) \vee \neg\mathit{Loves}(y, x))$ ($\forall y$ can be moved out; no y in first part)

To convert to CNF form, we simply drop the universal quantifiers and get this:
$\{\neg Animal(z), \neg Kills(x,z), \neg Loves(y,x)\}$ (let's call this 2)

Ramesh loves all animals.
$\forall x\ Animal(x) \rightarrow Loves(Ramesh, x)$
In CNF form, this is $\{\neg Animal(x), Loves(Ramesh, x)\}$ (3)

Either Ramesh or Curiosity killed the cat, who is named Molly.
$Kills(Ramesh, Molly) \lor Kills(Curiosity, Molly)$ (4)

We also take that $Animal(Molly)$. It is alright to infer or obtain this directly, or even indirectly by taking $Cat(Molly)$ and then assuming a more general statement that $\forall x\ Cat(x) \rightarrow Animal(x)$. The latter can be written in CNF form as $\{\neg Cat(x), Animal(x)\}$, and it can be resolved with $Cat(Molly)$ to obtain $Animal(Molly)$ under the substitution $x \mapsto Molly$. (5)

We assume that it wasn't *Curiosity* who killed the cat, to see if we can derive an empty clause. So, we have that $\neg Kills(Curiosity, Molly)$. Resolving this with $Kills(Ramesh, Molly) \lor Kills(Curiosity, Molly)$, we get $Kills(Ramesh, Molly)$. (6)

Here is the complete resolution proof:

(a)	$Animal(Molly)$	(5)
(b)	$Kills(Ramesh, Molly)$	(6)
(c)	$\neg Animal(z), \neg Kills(x,z), \neg Loves(y,x)$	(2)
(d)	$\neg Animal(x), Loves(Ramesh, x)$	(3)
(e)	$Animal(F(x)), Loves(G(x), x)$	(1a)
(f)	$\neg Loves(w, F(w)), Loves(G(w), w)$	(1b, with w in place of x)
(g)	$\neg Kills(x, Molly), \neg Loves(y,x)$	a & c, $[z \mapsto Molly]$
(h)	$\neg Loves(y, Ramesh)$	b & g, $[x \mapsto Ramesh]$
(i)	$\neg Animal(F(Ramesh)), Loves(G(Ramesh), Ramesh)$	d & f, $[x \mapsto F(w); w \mapsto Ramesh]$
(j)	$Loves(G(Ramesh), Ramesh)$	e & i, $[x \mapsto Ramesh]$
(k)	$\Box$	h & j, $[y \mapsto G(Ramesh)]$

8. **[4.5 marks]** Prove that if Γ is an unsatisfiable set of clauses, then Γ has a refutation containing only positive resolution inferences. A *positive resolution inference* is one in which one of the hypothesis is a positive clause. A clause is said to be *positive* if it contains only positive literals.

   Ans: We will prove that it is impossible for an unsatisfiable set of clauses Γ to be closed under positive resolution inference and not contain the empty clause.

   Let A be the set of positive clauses in Γ. Clearly, A is non-empty. [*Because if there are no positive clauses, then each clause has at least one negative literal. And setting every variable to "false" should satisfy* Γ.] Pick a truth assignment τ that satisfies all the clauses in A and assigns the minimum possible number of "true" values. Pick a clause L in $\Gamma \setminus A$ which is falsified by τ and has the minimum number of negative literals, and let $\neg p$ be one of the negative literals in L. [*Note that such an L must exist, because Γ is unsatisfiable.*] τ must have assigned "true" to p because L is falsified by τ. Pick a clause $J \in A$ that contains p and has the rest of its members assigned "false" by τ. [*Such a clause must exist, because of the choice of τ.*] But then J and L can be resolved, which leads to a contradiction.