Assignment 2 (*Marks: 7, Due date: Nov. 02*) **COL703, Jul-Nov 2023**

Please note that there will be zero tolerance for dishonest means like copying solutions from others, and even letting others copy your solution, in any assignment/quiz/exam. If you are found indulging in such an activity, your answer-paper/code will not be evaluated and your participation/submission will not be counted. Second-time offenders will be summarily awarded an F grade. The onus will be on the supposed offender to prove his or her innocence.

---

1. **[1 marks]** Formalise the following as sentences of first order logic. Use $B(x)$ for "x is a barber", and $S(x, y)$ for "x shaves y".

   (a) Every barber shaves all persons who do not shave themselves.

   (b) No barber shaves any person who shaves himself.

   Convert your answers to Skolem form and use ground resolution to show that (c), given below, is a consequence of (a) and (b).

   (c) There are no barbers.

2. **[1 marks]** Fix a signature σ. Consider a relation $\sim$ on σ-assignments that satisfies the following two properties.

   - If $\mathcal{A} \sim \mathcal{B}$ then for every atomic formula F we have $\mathcal{A} \models F$ iff $\mathcal{B} \models F$.
   - If $\mathcal{A} \sim \mathcal{B}$ then for each variable x we have (i) for each $a \in U_{\mathcal{A}}$ there exists $b \in U_{\mathcal{B}}$ such that $\mathcal{A}_{[x \mapsto a]} \sim \mathcal{B}_{[x \mapsto b]}$, and (ii) for all $b \in U_{\mathcal{B}}$ there exists $a \in U_{\mathcal{A}}$ such that $\mathcal{A}_{[x \mapsto a]} \sim \mathcal{B}_{[x \mapsto b]}$.

   Prove that if $\mathcal{A} \sim \mathcal{B}$ then for any formula F, $\mathcal{A} \models F$ iff $\mathcal{B} \models F$. You may assume that F is built from atomic formulas using the connectives $\wedge$ and $\neg$, and the quantifier $\exists$.

3. **[1 marks]** Express the following by formulas of first-order logic, using predicates $H(x)$ for "x is happy", $R(x)$ for "x is rich", $G(x)$ for "x is a graduate", and $C(x, y)$ for "y is a child of x".

   (a) Any person is happy if all their children are rich.

   (b) All graduates are rich.

   (c) Someone is a graduate if they are a child of a graduate.

   (d) All graduates are happy.

   Use first-order resolution to show that (d) is entailed by (a), (b), and (c). Indicate the substitutions in each resolution step.

4. **[1 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.

5. **[1 marks]** Let $A(x_1, \ldots, x_n)$ be a formula with no quantifiers and no function symbols. Prove that $\forall x_1 \ldots \forall x_n A(x_1, \ldots, x_n)$ is satisfiable if and only if it is satisfiable in an interpretation with there being just one element in the universe.

6. In this question, we work with first-order logic without equality.

(a) **[1 marks]** Consider a signature σ containing only a binary relation R. For each positive integer n show that there is a satisfiable σ-formula F_n such that every model $\mathcal{A}$ of F_n has at least n elements.

(b) **[1 marks]** Consider a signature σ containing only unary predicate symbols $P_1, \ldots, P_k$. Using the question 2 (above), or otherwise, show that any satisfiable σ-formula has a model where the universe has at most 2^k elements.

7. **This is a practice exercise. You are not supposed to submit the solution to this.** A *renamable Horn formula* is a CNF formula that can be turned into a Horn formula by negating (all occurrences of) some of its variables. For example,

$$(p_1 \vee \neg p_2 \vee \neg p_3) \wedge (p_2 \vee p_3) \wedge (\neg p_1)$$

can be turned into a Horn formula by negating p_1 and p_2.

Given a CNF formula F, it is in fact possible to derive a 2-CNF formula G such that G is satisfiable if and only if F is a renamable Horn formula. Moreover, you can derive a renaming (that turns F into a Horn formula) from a satisfying assignment for G.

Your task is to write a Python program that takes a CNF formula F, in DIMACS format, and also take a positive integer n ($1 \leq n \leq 4$) as a command-line argument, and does the following (for different values of n).

Let us take the following formula as our *sample input*:

```
c CNF formula (p1 ∨ !p2 ∨ !p3) ∧ (p2 ∨ p3) ∧ (!p1)
p cnf 3 3
1 -2 -3 0
2 3 0
-1 0
```

(a) When $n = 1$, your program should check whether the given formula is already a Horn formula or not, and output `already horn` and `not horn` accordingly.

Sample output: `not horn`

(b) When $n = 2$, your program should output a 2-CNF formula G, also in DIMACS format, such that G is satisfiable if and only if F is a renamable Horn formula.

Sample output:

```
c 2-CNF formula which is sat iff input is renamable Horn
p cnf 3 4
-2 -3 0
2 3 0
1 -3 0
1 -2 0
```

(c) When $n = 3$, your program should check whether F is a renamable Horn formula or not (in other words, whether the G that you would have output in the previous case is satisfiable or not[1]). If F is already a Horn formula, the program should output `already horn`. Otherwise, it should output `renamable` or `not renamable` accordingly.

Sample output: `renamable`

(d) When $n = 4$, your program should output a way to do the renaming of F. Essentially, you just need to output all the variables (space separated, sorted in ascending order, all in one line) that need to be negated. If the formula is not a renamable Horn formula or if it is already a Horn formula to begin with, you should output `not renamable` or `already horn` accordingly.

Sample output: `1 2`

[1]It might be interesting to do this by resolution.