Assignment 1 (*Marks: 7, Due date: Sept. 10*) **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]** Let us introduce a new connective $\phi \leftrightarrow \psi$ which should abbreviate $(\phi \to \psi) \wedge (\psi \to \phi)$. Design introduction and elimination rules (like the ones we had in natural deduction) for $\leftrightarrow$ and show that they are derived rules if $\phi \leftrightarrow \psi$ is interpreted as $(\phi \to \psi) \wedge (\psi \to \phi)$.

2. Prove the validity of the following sequents using natural deduction proof rules:

   (a) **[0.25 marks]** $(p \to r) \wedge (q \to r) \vdash p \wedge q \to r$

   (b) **[0.25 marks]** $p \to q \wedge r \vdash (p \to q) \wedge (p \to r)$

3. **[1 marks]** An adequate set of connectives for propositional logic is a set such that for every formula of propositional logic there is an equivalent formula with only connectives from that set. For example, $\{\neg, \vee\}$ is adequate. Is $\{\leftrightarrow, \neg\}$ adequate. Justify your answer. Recall that $\phi \leftrightarrow \psi$ is interpreted as $(\phi \to \psi) \wedge (\psi \to \phi)$.

4. Show that the following sequents are not valid by finding a valuation in which the truth values of the formulas to the left of $\vdash$ are `T` and the truth value of the formula to the right of $\vdash$ is `F`.

   (a) **[0.25 marks]** $\neg r \to (p \vee q), r \wedge \neg q \vdash r \to q$

   (b) **[0.25 marks]** $p \to (q \to r) \vdash p \to (r \to q)$

5. Let X be a set of propositional logic formulas. X is said to be a *finitely satisfiable set (FSS)* if every $Y \subseteq_{fin} X$ is satisfiable.

   Equivalently, X is an FSS if there is no finite subset $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ of X such that $\neg(\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n)$ is valid.

   (Note that if X is an FSS we are not promised a single valuation v which satisfies every finite subset of X. Each finite subset could be satisfied by a *different* valuation.)

   Show that:

   (a) **[0.25 marks]** Every FSS can be extended to a maximal FSS.

   (b) **[0.25 marks]** If X is a maximal FSS then for every formula α, $\alpha \in X$ *iff* $\neg\alpha \notin X$.

   (c) **[0.25 marks]** If X is a maximal FSS then for all formulas α, β, $(\alpha \vee \beta) \in X$ *iff* ($\alpha \in X$ *or* $\beta \in X$).

   (d) **[0.25 marks]** Every maximal FSS X generates a valuation v_X such that for every formula α, $v_X \vDash \alpha$ *iff* $\alpha \in X$.

   From these facts, conclude that:

   (e) **[0.5 marks]** Any FSS X is simultaneously satisfiable (that is, for any FSS X, there exists v_X such that $v_X \vDash X$).

   (f) **[0.5 marks]** For all X and all α, $X \vDash \alpha$ *iff* there exists $Y \subseteq_{fin} X$ such that $Y \vDash \alpha$.

6. [**2 marks**] Your task is to write a Python program that can take a natural deduction proof (as a text file), and output *correct* or *incorrect* depending on whether the proof is correct or not.

   The first line of the input proof file will always be the sequent whose validity is being proved. The second link will be an empty line, and the proof will begin from the third line. Each line of the proof will begin have an explanation preceeding the proof statement (this is unlike the proofs we did in the class, where we wrote the explanations *after* the statements). The explanations can only be of the following forms (where *i, j, k, l, m, n* denote line numbers in the proof file, as expected): `[premise]`, `[assumption]`, `[copy i]`, `[mp i, j]`, `[mt i, j]`, `[and-in i, j]`, `[and-e1 i]`, `[and-e2 i]`, `[or-in1 i]`, `[or-in2 i]`, `[or-el i, j-k, l-m]`, `[impl-in i-j]`, `[neg-in i-j]`, `[neg-el i, j]`, `[bot-el i]`, `[dneg-in i]`, `[dneg-el i]`, `[pbc i-j]`, `[lem]`.

   The keywords are self-explanatory, and the precise interpretation of these rules can be found in Chapter 1 of the book by Huth and Ryan. Please also note that we will denote the logical symbols $\wedge$ and $\vee$ using backward and forward slashes, $\neg$ using the `!` symbol, $\rightarrow$ using `->`, and $\vdash$ using `|-` in the text file.

   Here are two sample input files and their corresponding output:

```
q -> r |- p \/ q -> p \/ r

[premise]               q -> r
[assumption]            p \/ q
[assumption]            p
[or-in1 5]              p \/ r
[assumption]            q
[mp 7, 3]               r
[or-in2 8]              p \/ r
[or-el 4, 5-6, 7-9]     p \/ r
[impl-in 4-10]          p \/ q -> p \/ r
```

   Output: `correct`

```
p -> q |- !p \/ q

[premise]               p -> q
[lem]                   q \/ !q
[assumption]            q
[or-in2 5]              !p \/ q
[assumption]            !q
[mp 7, 3]               !p
[or-in1 8]              !p \/ q
[or-el 4, 5-6, 7-9]     !p \/ q
```

   Output: `incorrect`