Exercise 1. *(2.3.2) Show that all binary connectives can be defined using either $\uparrow$ or $\downarrow$ alone.*

Solution: we know (p.15; cf. also the Ba LICS course), that negation and conjunction are functionally complete. Formally: for any function $f : \mathsf{Tr}^n \to \mathsf{Tr}$ with $\mathsf{Tr} = \{t, f\}$, there is an expression e only using negation and conjunction such that $f(\vec{b}) = v_{\vec{b}}(e)$ for any $\vec{b} = b_1, \ldots, b_n \in B^n$ and $v_{\vec{b}}$ the valuation mapping P_i to b_i for all $1 \le i \le n$.

Hence it suffices to show that we can define negation and conjunction using just $\uparrow$ (NAND). Defining $\neg(P_1) = \uparrow(P_1, P_1)$ works for negation since $v(\neg(P_1)) = v(\uparrow(P_1, P_1))$ is seen to hold for all valuations v. Using the definition of negation, defining $\wedge(P_1, P_2) = \neg(\uparrow(P_1, P_2))$ is seen to work.

For $\downarrow$ the result follows from that for $\uparrow$ using that these are dual operations.

Remark 1. *$\uparrow$ and $\downarrow$ are not* strictly *functionally complete, where* strict *functional completeness means that the expression e in the above is only allowed to contain the propositional letters $P_1, \ldots, P_n$ when defining an n-ary function f. In particular, to define $\bot$ and $\top$ (both 0-ary),* no *propositional letters would be allowed, but there are no such expressions built just from the (binary) connectives $\uparrow$ and $\downarrow$; any expression must contain* some *propositional letters to start with!*

Exercise 2. *(2.4.3) Which of the following are, and which are not tautologies.*

1. *yes. known as Peirce's law;*
2. *no. take $v(P) = v(R) = t$ and $v(Q) = f$;*
3. *no. both sides of $\not\equiv$ are equivalent to $\neg Q$;*
4. *yes. both sides of $\equiv$ are equivalent to $\neg(P \supset Q)$;*
5. *yes. to make the premiss true, Q must be false, hence P as well;*
6. *no. take $v(P) = f$, $v(Q) = t$;*
7. *yes. $\not\equiv$ is commutative since $\equiv$ is;*
8. *yes. $\not\equiv$ is associative since $\equiv$ is, using equivalence of $X \equiv \neg Y$ and $\neg(X \equiv Y)$.*

Note that for the non-tautologies it suffices to give one valuation such that the formula evaluates to false; i.e. it is not necessary to give the whole truth-table.

Exercise 3. *(2.4.12) Prove that if the propositional formula $X \supset Y$ is a tautology, so is $Y^d \supset X^d$.*

$Y^d \supset X^d$ is equivalent to $\neg\overline{Y} \supset \neg\overline{X}$ (Exercise 2.4.11) which is equivalent to $\overline{X} \supset \overline{Y}$ (for example by reasoning that $\neg P \supset \neg Q$ is equivalent first to $\neg\neg P \vee \neq Q$ then to $P \vee \neq Q$, then $\neg Q \vee P$ and finally $Q \supset P$). From that we conclude since tautologies are invariant under negating their propositional letters (Exercise 2.4.10).

The idea captured in symbols by exercise 2.4.11 is that to dualize an expression, we need to dualize the connectives in it, and negate both its input(s) and its output. This patterns generalizes the idea for single connectives, e.g. that the dual of $P \wedge Q$ is $\neg(\neg P \vee \neg Q)$.

Note that, in general, $X \supset Y$ is not equivalent to $Y^d \supset X^d$. For instance, the truth tables of $P \supset Q$ and $Q^d \supset P^d$, i.e. $Q \supset P$ are distinct.

Exercise 4. *(2.6.1) Intuitively, the degree d counts the number of occurrences of connectives of arity > 0 (i.e. all except for $\top$ and $\bot$). Intuitively, the rank r counts the number of conjunctions, disjunctions, double negations, and negations on top of tops and bottoms, after first eliminating other (possibly negated) binary connectives where double negations arising during elimination are elided on the fly. (For example, $\neg(X \supset Y)$ is essentially conjunctive, an α, but instead of simply using De Morgan and continuing the transformation with $\neg\neg X \wedge \neg Y$, the double negation on the X is elided, and we continue transforming from $X \wedge \neg Y$.*

- $d = 4$ *(two implications, a NOR, and a negation),* $r = 4$
- $d = 2$ *(two implications),* $r = 2$
- $d = 2$ *(two implications),* $r = 3$

Exercise 5. *(Bonus) The relevant Haskell code snippet is*

```
t :: Form a -> Form a
t x = case (map t (usub x)) of
  [] -> x
  [y] -> Neg (Neg y)
  [y,z] -> if (fisa x) then (Bin And y z) else (Bin Or y z)
```

where `usub` *computes the subformula in uniform notation (Table 2.2), and* `fisa` *checks whether a formula is conjunctive, an α, or not. The properties then are that $r(X) = r(t(X))$ and that X is equivalent to $t(X)$, but only comprises conjunctions, disjunctions and negations. A structural property is that 'above' binary connectives there always only are even numbers of negations. This can be proven by structural induction (Theorem 2.6.3). The basis step trivially holds as there t is the identity. Of the induction steps, the double negation case follows directly by the IH. In case of an α, the properties hold for the α_i by the IH. From that they are seen to hold, by construction, for α as well: α is equivalent to $\alpha_1 \wedge \alpha_2$ (Proposition 2.6.1) so to $t(\alpha) = t(\alpha_1) \wedge t(\alpha_2)$ as well by the IH; From the latter equality and the IH it also follows that $t(\alpha)$ only comprises conjunctions, disjunctions and negations, and finally $t(\alpha)$ has an even (namely 0) number of negations above it.*

Exercise 6. *(2.6.2)*

- *The intuition is the same as for a binary tree of depth n having fewer than 2^n nodes. Formally, the proof is by the principal of structural induction (Theorem 2.6.3):*

  - *Basis step. For A a propositional letter $r(A) = r(\neg A) = r(\top) = r(\bot) = 0$, $h(A) = h(\neg A) = h(\top) = h(\bot) = 0$, and indeed $0 \leq 0 = 2^0 - 1$. $r(\neg\top) = r(\neg\bot) = 1$, $h(\neg\top) = h(\neg\bot) = 1$, and indeed $1 \leq 1 = 2^1 - 1$.*
  - *Induction steps. Suppose $r(X) \leq 2^{h(X)}$. Then $r(\neg\neg X) = r(X) + 1 \leq 2^{h(X)} - 1 + 2^{h(X)} = 2^{h(\neg\neg X)} - 1$ where the inequality uses the IH and the fact that $1 \leq 2^n$ for any natural number n.*

    Suppose $r(\alpha_i) \leq 2^{h(\alpha_i)}$ for $i \in \{1, 2\}$. Then $r(\alpha) = 1 + \sum_i r(\alpha_i) \leq 1 + 2 \cdot \max_i r(\alpha_i) \leq 1 + 2 \cdot \max_i (2^{h(\alpha_i)} - 1) = 2^{1+\max_i h(\alpha_i)} - 1 = 2^{h(\alpha)} - 1$ where the second inequality uses the IH.

    The case for β is analogous to that for α.

- *Inductively define $X_0 = P$ and $X_n = X_{n-1} \wedge X_{n-1}$ for $n > 0$. We show by induction on n that $r(X_n) = 2^{d(X_n)} - 1$. For the basis step $r(X_0) = 0 = 2^0 - 1 = 2^{d(X_0)} - 1$. In the induction step $r(X_n) = r(X_{n-1} \wedge X_{n-1}) = 1 + 2 \cdot r(X_{n-1}) = 2^{1+d(X_n)} - 1 = 2^{d(X_n)} - 1$, using the IH for the 3rd equality.*

Exercise 7. *(2.8.4. Bonus)*

1. $\langle [P, \neg P] \rangle$;
2. $\langle [P], [\neg P] \rangle$;
3. $\langle [P, Q, \neg P], [P, Q, \neg Q] \rangle$;
4. $\langle [\neg P, \neg Q, R], [\neg P, Q], [P], [\neg R] \rangle$;
5. $\langle [\neg P], [\neg Q, \neg R], [P] \rangle$.

obtained by the Haskell code

```
cnf :: Form a -> Form a
cnf = dcist . nnf . t
```

where `t` *is as above,* `nnf` *removes double negations and negations on top of top and bottom, and finally* `dcist` *distributes disjunctions over conjunctions.*

Exercise 8. *(2.8.6. Bonus)*

1. $[\langle P \rangle, \langle \neg P \rangle]$;
2. $[\langle P, \neg P \rangle]$;
3. $[\langle P \rangle, \langle Q \rangle, \langle \neg P, \neg Q \rangle]$;
4. $[\langle \neg P, \neg P, P, \neg R \rangle, \langle \neg P, Q, P, \neg R \rangle, \langle \neg Q, \neg P, P, \neg R \rangle, \langle \neg Q, Q, P, \neg R \rangle, \langle R, \neg P, P, \neg R \rangle, \langle R, Q, P, \neg R \rangle]$;
5. $[\langle \neg P, \neg Q, P \rangle, \langle \neg P, \neg R, P \rangle]$.

obtained by the Haskell code

```
dnf :: Form a -> Form a
dnf = cdist . nnf . t
```

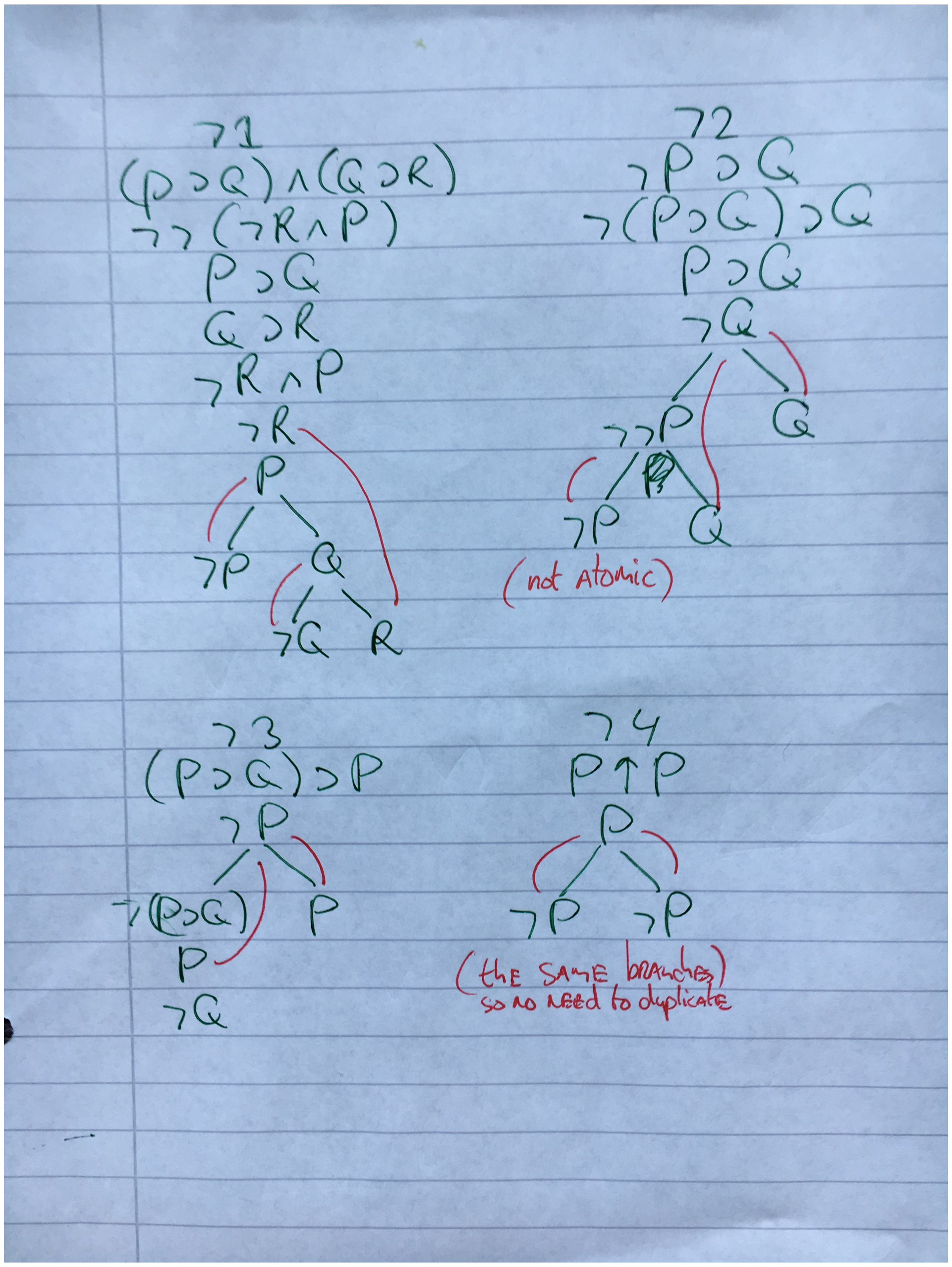
where `dcist` *is dual to* `cdist` *above.*

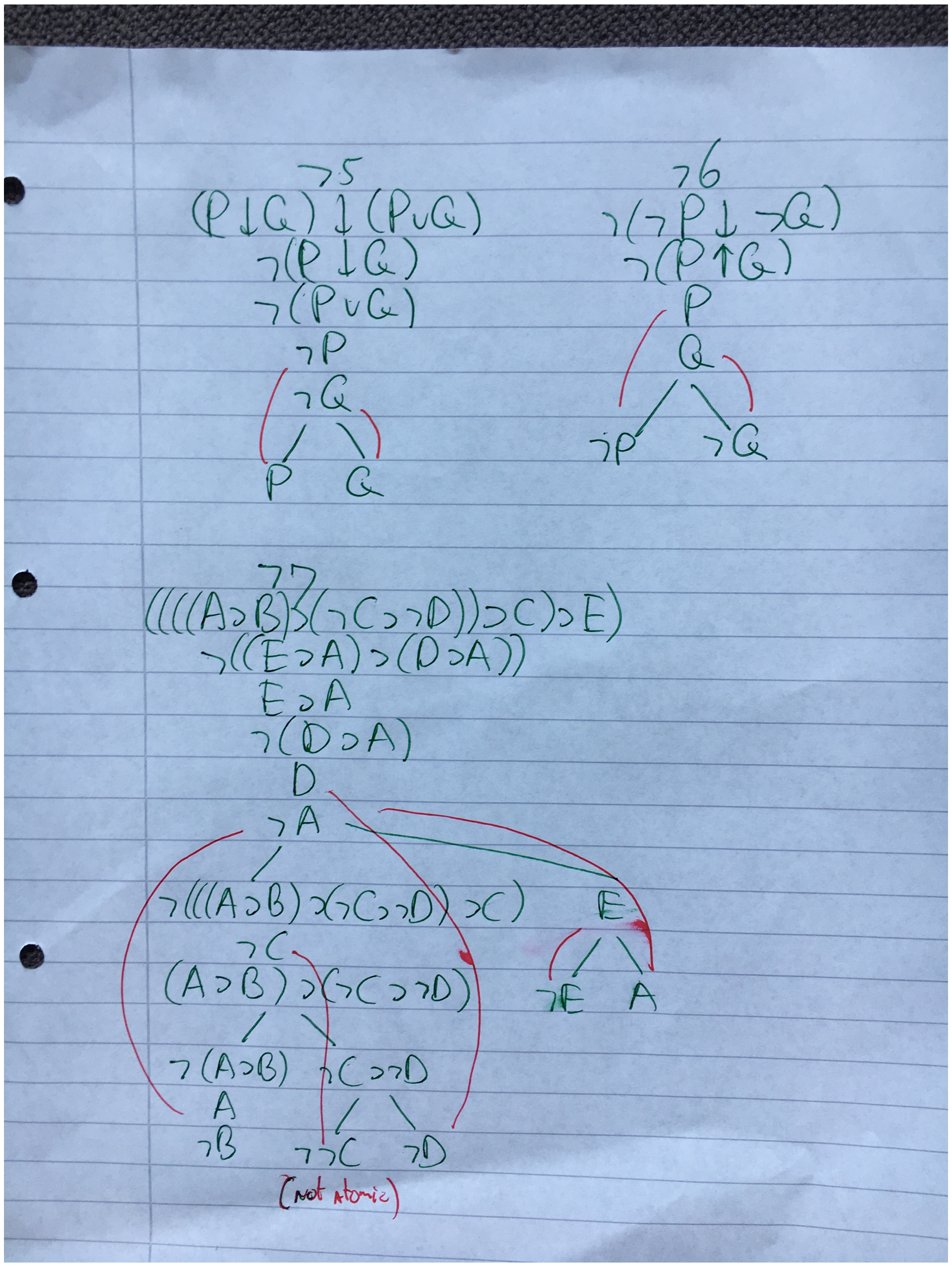
Remark 2. *The algorithms only use distributivity and De Morgan. This may lead to repetitions of literals and (dual) clauses, which could be easily removed by adjoining a rule to simplify $X \vee X$ into X and its dual.*

Exercise 9. *(2.8.7) By duality from the same properties of the Clause Form Algorithm. Formally, it follows by first replacing all connectives by their dual, then executing the Clause Form Algorithm, and then replacing all connectives by their dual again, noting that all transformation rules dualize.*

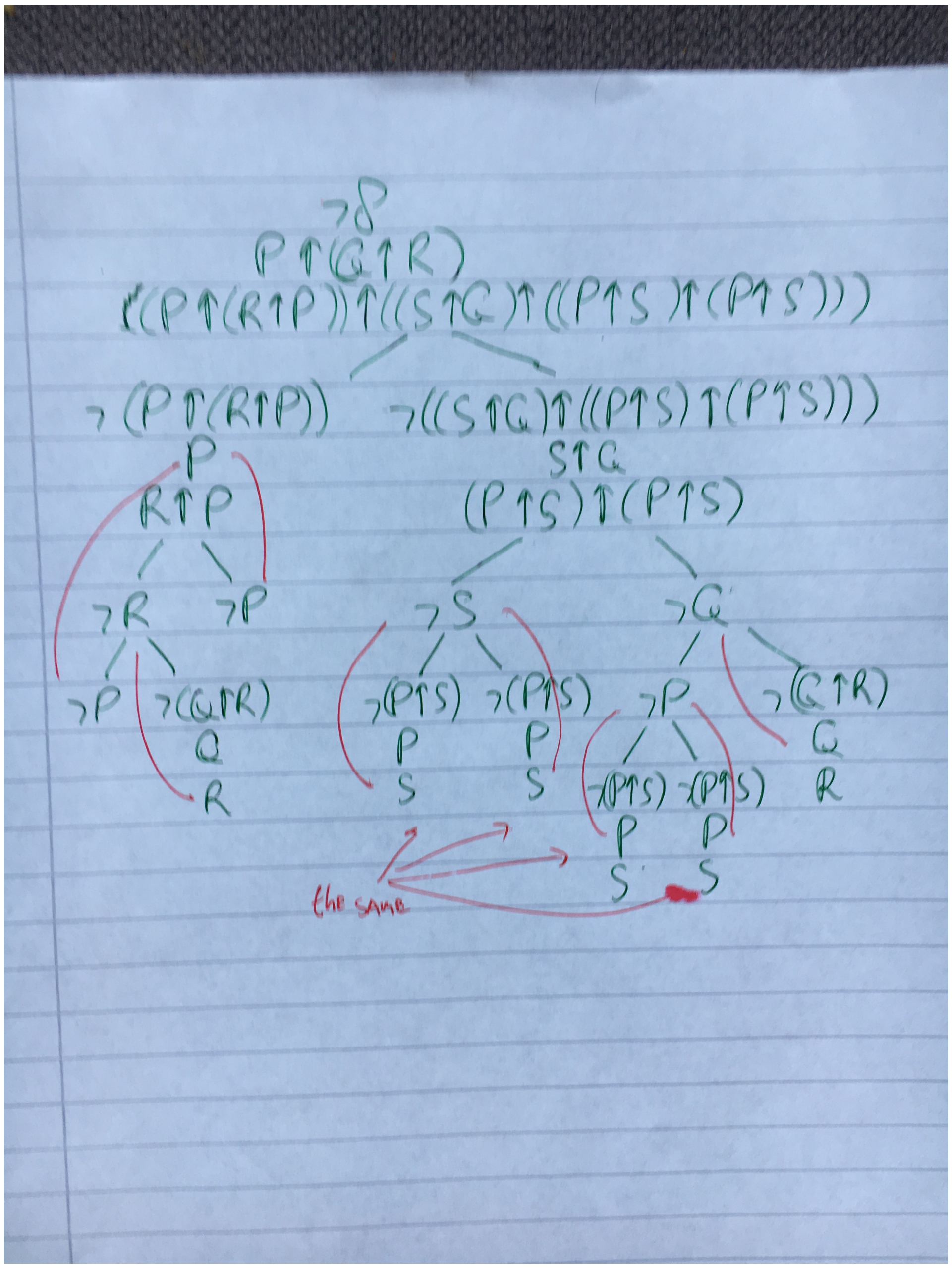
Note that the problem with showing termination of these algorithms is that simple measures, such as counting the number of connectives in an expression, do not suffice. In particular, distributing conjunction over disjunction (or vice versa) may lead to an increase *in the number of connectives: distribution applied to $P \wedge (Q \vee R)$ yields the formula $(P \wedge Q) \vee (P \wedge R)$, which is* larger. *(That this terminates nonetheless can be intuitively understood by noting that distribution sorts connectives top–down with, in this example, disjunctions being moved to the top/root and conjunctions moved to the bottom/leaves.)*

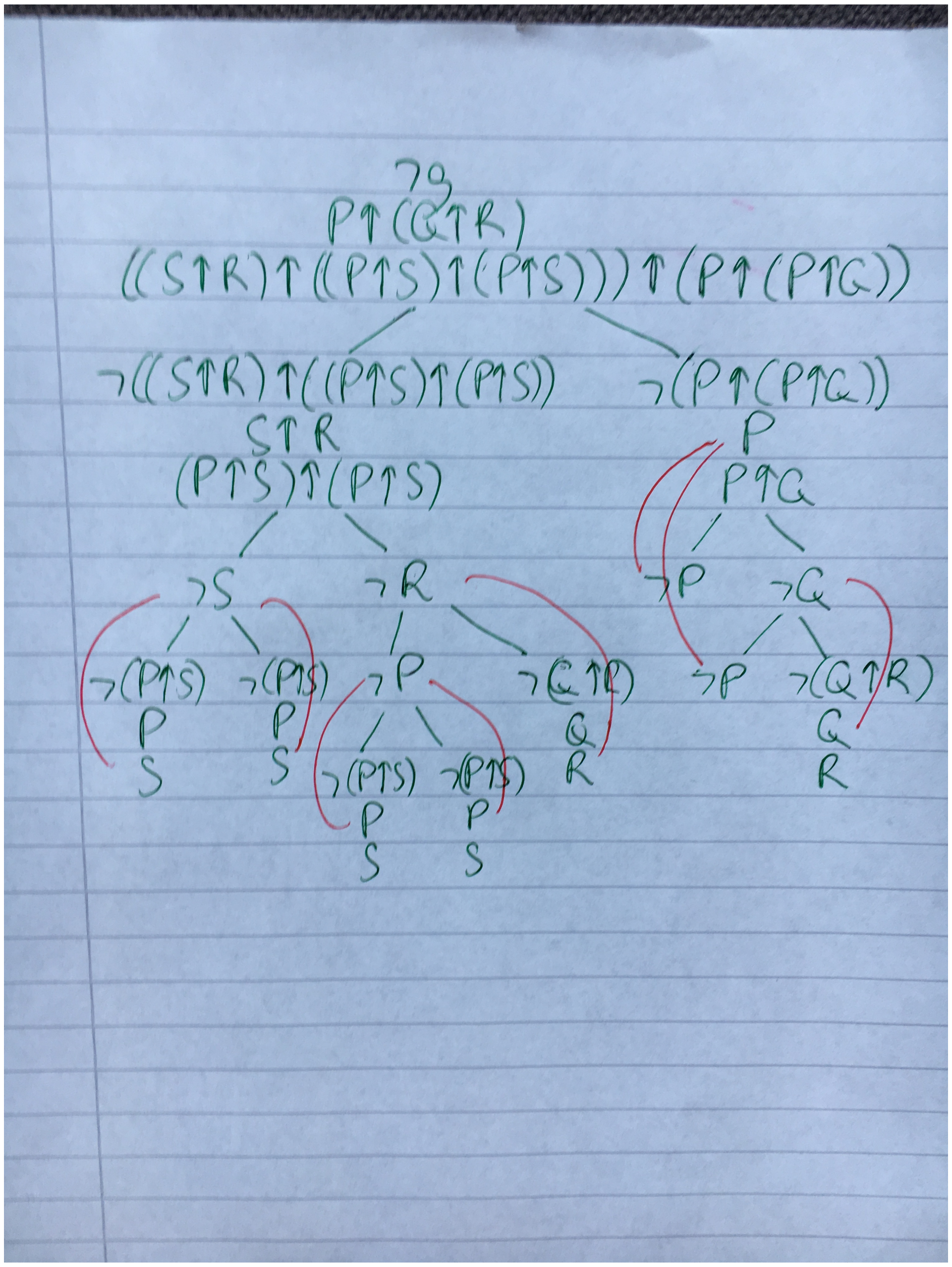
Exercise 10. *(3.1.1) Below some solutions (hope I didn't make any mistakes). It is obvious that several trees contain repeated subtrees, so one could think about how to avoid these, For instance, by some sharing mechanism (working with dags instead of trees). A simple case of this arises for a β-formula such that $\beta_1 = \beta_2$; is it ok to share these and to have 'only one branch' with that subformula on it? Moreover, one can think about what is the shortest possible tree in each item, and how that could be found (is brute-forcing it, say by breadth-first search, the only way)?*


¬1
(P ⊃ Q) ∧ (Q ⊃ R)
¬¬(¬R ∧ P)
P ⊃ Q
Q ⊃ R
¬R ∧ P
¬R
P
¬P
Q
¬Q
R
¬2
¬P ⊃ Q
¬(P ⊃ Q) ⊃ Q
P ⊃ Q
¬Q
¬¬P
Q
P
¬P
Q
(not atomic)
¬3
(P ⊃ Q) ⊃ P
¬P
¬(P ⊃ Q)
P
P
¬Q
¬4
P ↑ P
P
¬P
¬P
(the same branches)
so no need to duplicate


```
        ¬5                                  ¬6
  (P↓Q)↓(P∨Q)                         ¬(¬P↓¬Q)
     ¬(P↓Q)                            ¬(P↑Q)
     ¬(P∨Q)                               P
       ¬P                                 Q
       ¬Q                               /   \
      /   \                           ¬P     ¬Q
     P     Q

       ¬7
((((A⊃B)⊃(¬C⊃¬D))⊃C)⊃E)
   ¬((E⊃A)⊃(D⊃A))
       E⊃A
      ¬(D⊃A)
        D
        ¬A
       /                              \
¬(((A⊃B)⊃(¬C⊃¬D))⊃C)                   E
       ¬C                            /   \
 (A⊃B)⊃(¬C⊃¬D)                     ¬E     A
    /        \
¬(A⊃B)      ¬C⊃¬D
   A         /   \
  ¬B      ¬¬C     ¬D
       (not atomic)
```


¬δ
P↑(Q↑R)
¬((P↑(R↑P))↑((S↑Q)↑((P↑S)↑(P↑S))))
¬(P↑(R↑P))
¬((S↑Q)↑((P↑S)↑(P↑S)))
P
R↑P
¬R
¬P
¬P
¬(Q↑R)
Q
R
S↑Q
(P↑S)↑(P↑S)
¬S
¬Q
¬(P↑S)
¬(P↑S)
P
S
P
S
¬P
¬(Q↑R)
Q
R
¬(P↑S)
¬(P↑S)
P
S
P
S
the same


¬Q
P↑(Q↑R)
((S↑R)↑((P↑S)↑(P↑S)))↑(P↑(P↑Q))
¬((S↑R)↑((P↑S)↑(P↑S))
S↑R
(P↑S)↑(P↑S)
¬(P↑(P↑Q))
P
P↑Q
¬S
¬R
¬P
¬Q
¬(P↑S)
P
S
¬(P↑S)
P
S
¬P
¬(Q↑R)
Q
R
¬P
¬(Q↑R)
Q
R
¬(P↑S)
P
S
¬(P↑S)
P
S

Exercise 11. *(3.6.3) To show that a Propositional Consistency Property that is subset closed can be extended to one of finite character. That is, given a Propositional Consistency Property $\mathcal{C}$ such that if $T \subseteq S \in \mathcal{C}$ then $T \in \mathcal{C}$, we must show there exists a Propositional Consistency Property $\mathcal{C}^+$ such that $\mathcal{C} \subseteq \mathcal{C}^+$ and $S \in \mathcal{C}^+$ iff for every finite $T \subseteq S$, $T \in \mathcal{C}^+$.*

Following the hint, we define $\mathcal{C}^+$ to consist of those sets S all of whose finite subsets are in $\mathcal{C}$, and show that it meets the above three conditions.

- *We first show that $\mathcal{C}^+$ extends $\mathcal{C}$, i.e. that $\mathcal{C} \subseteq \mathcal{C}^+$. If $S \in \mathcal{C}$ then by $\mathcal{C}$ being subset closed we have for every $T \subseteq S$, that $T \in \mathcal{C}$, so certainly for finite such T. Thus, $S \in \mathcal{C}^+$.*

- *Next, we show that $\mathcal{C}^+$ is of finite character, i.e. that $S \in \mathcal{C}^+$ iff for every finite $T \subseteq S$, $T \in \mathcal{C}^+$.*

  For the only–if-direction, note that if $S \in \mathcal{C}^+$, then by definition of $\mathcal{C}^+$ for every finite $T \subseteq S$ we have $T \in \mathcal{C}$, hence $T \in \mathcal{C}^+$ by the previous item.

  For the if-direction, suppose for every finite $T \subseteq S$, $T \in \mathcal{C}^+$. Note that for such T, we have $T \in \mathcal{C}$ by definition of $\mathcal{C}^+$ using that T is a finite subset of itself, hence $S \in \mathcal{C}^+$.

- *Finally, we show that $\mathcal{C}^+$ is a Propositional Consistency Property by verifying that the five defining conditions of the latter (Definition 3.6.1) hold. To that end, suppose $S \in \mathcal{C}^+$, so all its finite subsets are in the Propositional Consistency Property $\mathcal{C}$.*

  1. *If both $A \in S$ and $\neg A \in S$ for some propositional letter A, then by definition of $\mathcal{C}^+$ the finite subset $\{A, \neg A\}$ of S is in $\mathcal{C}$, which would contradict (item 1 of) $\mathcal{C}$ being a Propositional Consistency Property;*
  2. *If $\bot \in S$ or $\neg\top \in S$, then by definition of $\mathcal{C}^+$ the finite subset $\{\bot\}$ or $\{\neg\top\}$ of S is in $\mathcal{C}$, which would contradict (item 2 of) $\mathcal{C}$ being a Propositional Consistency Property;*
  3. *Suppose $\neg\neg Z \in S$ such that every finite subset of S is in $\mathcal{C}$. We must show that every finite $T \subseteq S \cup \{Z\}$ is in $\mathcal{C}$. Note that $U = (T - \{Z\}) \cup \{\neg\neg Z\}$ is a finite subset of S hence is in $\mathcal{C}$. By (item 3 of) $\mathcal{C}$ being a Propositional Consistency Property then $T \cup \{\neg\neg Z, Z\} \in \mathcal{C}$, from which we conclude to $T \in \mathcal{C}$ by $\mathcal{C}$ being subset closed;*
  4. *Suppose $\alpha \in S$ such that every finite subset of S is in $\mathcal{C}$. We must show that every finite $T \subseteq S \cup \{\alpha_1, \alpha_2\}$ is in $\mathcal{C}$. Note that $U = (T - \{\alpha_1, \alpha_2\}) \cup \{\alpha\}$ is a finite subset of S hence is in $\mathcal{C}$. By (item 4 of) $\mathcal{C}$ being a Propositional Consistency Property then $T \cup \{\alpha, \alpha_1, \alpha_2\} \in \mathcal{C}$, from which we conclude to $T \in \mathcal{C}$ by $\mathcal{C}$ being subset closed;*
  5. *Suppose $\beta \in S$ such that every finite subset of S is in $\mathcal{C}$. We have to show that $S \cup \{\beta_1\} \in \mathcal{C}^+$ or $S \cup \{\beta_2\} \in \mathcal{C}^+$. For a proof by contradiction suppose that neither $S \cup \{\beta_1\}$ nor $S \cup \{\beta_2\}$ is in $\mathcal{C}^+$. By definition of $\mathcal{C}^+$, then there exist finite $T_1 \subseteq S \cup \{\beta_1\}$ and $T_2 \subseteq S \cup \{\beta_2\}$ such that*

$T_1, T_2 \notin \mathcal{C}$. Let $T = T_1 \cup T_2$. Note that $U = (T \cap S) \cup \{\beta\}$ is a finite subset of S, hence is in $\mathcal{C}$. By (item 5 of) $\mathcal{C}$ being a Propositional Consistency Property then $(T \cap S) \cup \{\beta, \beta_1\} \in \mathcal{C}$ or $(T \cap S) \cup \{\beta, \beta_2\} \in \mathcal{C}$. Since $T_1 \subseteq (T \cap S) \cup \{\beta, \beta_1\}$ and $T_2 \subseteq (T \cap S) \cup \{\beta, \beta_2\}$ and $\mathcal{C}$ is subset closed, we have $T_1 \in \mathcal{C}$ or $T_2 \in \mathcal{C}$, providing the desired contradiction.

Remark 3. *Although the solution to this last exercise follows the general pattern when 'closing under an operation', namely showing that the 'existing' properties are preserved (in this case: being a Propositional Consistency Property) and that the 'desired' properties are created (in this case: being of finite character and extending the original collection), the solution is longish as I decided to spell out the details, because:*

- *it's good to see all the proof steps; in particular many solutions missed either showing that $\mathcal{C}^+$ extends $\mathcal{C}$, or showing that $\mathcal{C}^+$ is of finite character. Neither is difficult, but both essentially depend on the construction of $\mathcal{C}^+$ from $\mathcal{C}$, and the above solution pinpoints where and how.*
- *the reasoning in the last item (for β) is more subtle than one might expect; we need to show an 'exists for all' where one seems to only have a 'for all exists'.*

(For people wanting to see what could go wrong in such 'closing under' arguments, a simple example is provided by relations, where the symmetric closure of a transitive relation need not be transitive; but note that the transitive closure of a symmetric relation is symmetric again!)