ΙΩΑΝΝΗΣ ΚΑΠΕΤΑΝΓΕΩΡΓΗΣ **Α.Μ. 1115201800061**

PROJECT 4
Τεχνητή Νοημοσύνη

Άσκηση 1

Ερώτημα α

Πρέπει να ορίσουμε μια ερμηνεία I έτσι ώστε να περιγράψουμε την εικόνα της εκφώνησης. Η ερμηνεία αυτή βασίζεται στις προτάσεις που μας δόθηκαν, δηλαδή δίνουμε μια ερμηνεία για τα σύμβολα του λεξιλογίου των προτάσεων που περιγράφουν τον κόσμο αυτό.

Αρχικά πρέπει να ορίσουμε το πεδίο της I. Το πεδίο αυτό πρέπει να περιέχει όλα τα αντικείμενα τα οποία βλέπουμε στην εικόνα. Θα παραστήσω τα αντικείμενα στην ελληνική γλώσσα έτσι ώστε να μην τα μπερδεύουμε με τα σύμβολα της λογικής (σταθερές, συναρτήσεις και κατηγορήματα). Τα σύμβολα της λογικής παριστάνονται στην αγγλική γλώσσα.
Στην εικόνα υπάρχουν πολλά αντικείμενα, ωστόσο στην ερμηνεία θα συμπεριλάβουμε μόνο όσα σχετίζονται με τις προτάσεις που μας δόθηκαν. Τα αντικείμενα αυτά είναι οι δύο χαρακτήρες Μακαρένα και Σαράι.
Επομένως το πεδίο της I που προκύπτει είναι:

|I| = { Μακαρένα , Σαράι }

Στην συνέχεια κάνουμε της αντιστοιχήσεις της I για τα σύμβολα σταθερών. Έχουμε:

$Makarena^I =$ **Μακαρένα** , $Saray^I =$ **Σαράι**

Στην συνέχεια κάνουμε της αντιστοιχήσεις της I για τα σύμβολα κατηγορήματος που υπάρχουν στις προτάσεις που μας δόθηκαν.

Η I αντιστοιχίζει στο μοναδιαίο σύμβολο κατηγορήματος **Blonde** την ακόλουθη μοναδιαία σχέση:

{ <Μακαρένα> }

(αφού μόνο η Μακαρένα είναι ξανθιά, όπως φαίνεται και στην εικόνα)

Η I αντιστοιχίζει στο μοναδιαίο σύμβολο κατηγορήματος **Woman** την ακόλουθη μοναδιαία σχέση:

{ <Μακαρένα> , <Σαράι> }

(αφού και οι δύο είναι γυναίκες)

Έτσι, έχουμε ορίσει κατάλληλα την ερμηνεία I έτσι ώστε να έχουν νόημα οι προτάσεις που μας δόθηκαν.

Ερώτημα β

- $\phi_1 \ : \ Blonde(Macarena)$

Για τον τύπο φ1, από τον ορισμό της ικανοποίησης έχουμε:

$$\models_I Blonde(Macarena) \quad \alpha\nu\nu \quad < \bar{s}(Macarena) > \in Blonde^I$$

Επίσης, από την ερμηνεία I έχουμε ότι:

$$\bar{s}(Macarena) = Macarena^I = \text{Μακαρένα}$$

και

$$Blonde^I = \ \{ <\text{Μακαρένα}> \}$$

Δηλαδή ισχύει ότι: $< \bar{s}(Macarena) > \in Blonde^I$
Άρα η φ1 ικανοποιείται από την I.

• $\phi_2 \ : \ Blonde(Saray)$

Ομοίως με προηγουμένως, για την φ2 έχουμε:

$$\vDash_I Blonde(Saray) \quad ανν \quad < \overline{s}(Saray) > \in Blonde^I$$

Δηλαδή για να ικανοποιείται η φ2 πρέπει να ισχύει: $< \overline{s}(Saray) > \in Blonde^I$
Από την ερμηνεία I έχουμε ότι:

$$\overline{s}(Saray) = Saray^I = \text{Σαράι}$$

Όμως: $Blonde^I =$ { <Μακαρένα> }

Δηλαδή, $< \overline{s}(Saray) > \notin Blonde^I$

Άρα η φ2 δεν ικανοποιείται από την I.

• $\phi_3 \ : \ (\exists x) Blonde(x)$

Ο τύπος αυτός ικανοποιείται αν υπάρχει $d_x \in |I|$ τέτοιο ώστε η πρόταση Blonde(x) να ικανοποιείται.
Το πεδίο της I είναι: |I| = { Μακαρένα , Σαράι }.
Αν στο x ανατεθεί η τιμή Μακαρένα προκύπτει ο τύπος: $Blonde(x) \ [s(x \,|\, Μακαρενα)]$
Ο οποίος ικανοποιείται (όπως αποδείχθηκε για την φ1) αφού $Μακαρενα = < \overline{s}(Macarena) > \in Blonde^I$

Δηλαδή έχουμε ότι: $\vDash_I (Blonde(x)) \ [s(x \,|\, Μακαρενα)]$
Άρα υπάρχει τουλάχιστον ένα $d_x \in |I|$ τέτοιο ώστε να ικανοποιείται ο τύπος Blonde(x).
Άρα η φ3 ικανοποιείται από την I.

• $\phi_4 \ : \ (\forall x)(Woman(x) \Rightarrow Blonde(x))$

Ο τύπος φ4 ικανοποιείται αν για κάθε $d_x \in |I|$ ισχύει:

$$\vDash_I (Woman(x) \Rightarrow Blonde(x)) \ [s(x \,|\, d_x)]$$

Το οποίο με την σειρά του ισχύει αν για κάθε $d_x \in |I|$ ισχύει:

$\nvDash_I (Woman(x)) \ [s(x \,|\, d_x)]$ ή $\vDash_I (Blonde(x)) \ [s(x \,|\, d_x)]$

- Αν στο x ανατεθεί η τιμή: Μακαρένα

Τότε ισχύει: $\vDash_I (Woman(x)) \ [s(x \,|\, Μακαρενα)]$

Και επίσης: $< \overline{s}(Macarena) > = < Macarena^I > =$ < Μακαρένα > $\in Blonde^I$
Επομένως η φ4 ικανοποιείται για την ανάθεση [s(x|Μακαρένα)]

- Αν στο x ανατεθεί η τιμή: Σαράι

Τότε ισχύει: $\vDash_I (Woman(x)) \ [s(x \,|\, Σαραι)]$

Όμως έχουμε ότι: $< \overline{s}(Saray) > = < Saray^I > =$ Σαράι $\notin Blonde^I$
Επομένως η φ4 δεν ικανοποιείται για την ανάθεση [s(x|Σαράι)]

Άρα ο τύπος φ4 δεν ικανοποιείται για κάθε ανάθεση $d_x \in |I|$ της μεταβλητής x.
Άρα ο τύπος φ4 δεν ικανοποιείται από την ερμηνεία I.

Άσκηση 2

Για να βρούμε τον γενικό ενοποιητή για κάθε έναν από τους τύπους, εφαρμόζουμε τον αλγόριθμο ενοποίησης UNIFY.

- $P(x,x)$ και $P(G(f(v)),G(u))$

UNIFY(P(x,x) , P(G(f(v)),G(u)))
Αρχικά έχουμε ότι γ={}
- Για i=0: έχουμε UNIFY(P,P). Ισχύει ότι P=P άρα η UNIFY επιστρέφει {}.
- Για i=1 έχουμε UNIFY(x , G(f(v))).
  Το πρώτο όρισμα (x) είναι μια μεταβλητή, επομένως γίνεται η κλήση UNIFY-VAR(x , G(f(v))).
  Η μεταβλητή x δεν υπάρχει στο G(f(v)), επομένως η UNIFY-VAR μας επιστρέφει την ανάθεση { x/G(f(v)) }.
  Προσθέτουμε την νέα ανάθεση στο σύνολο αναθέσεων γ μέσω της COMPOSE(γ , { x/G(f(v)) }).
  Άρα το γ που προκύπτει είναι γ = { x/G(f(v)) }.
  Αντικαθιστούμε την μεταβλητή x με την νέα ανάθεση της, όπου αυτή εμφανίζεται στους δύο τύπους.
  Τελικά οι δύο τύποι που προκύπτουν είναι: P(G(f(v)) , G(f(v))) και P(G(f(v)) , G(u))
- Για i=2 έχουμε UNIFY(G(f(v)) , G(u))
  Και τα 2 ορίσματα είναι σύμβολα συναρτήσεων, επομένως ακολουθούμε την ίδια διαδικασία αναδρομικά.
  γ2 = {}
  - Για i=0 έχουμε UNIFY(G,G). Ισχύει ότι G=G άρα η UNIFY επιστρέφει {}.
  - Για i=1 έχουμε UNIFY(f(v) , u).
    Παρατηρούμε ότι το δεύτερο όρισμα (u) είναι μεταβλητή, άρα γίνεται η κλήση UNIFY-VAR(u , f(v)).
    Η μεταβλητή u δεν υπάρχει στο f(v), επομένως επιστρέφεται η ανάθεση { u/f(v) }.
    Προσθέτουμε την νέα ανάθεση στο σύνολο αναθέσεων γ2 μέσω της COMPOSE(γ2 , { u/f(v) }).
    Άρα το γ που προκύπτει είναι γ2 = { u/f(v) }.
    Αντικαθιστούμε την u με την νέα ανάθεση της, όπου αυτή εμφανίζεται στους δύο τύπους.
    Τελικά οι δύο τύποι που προκύπτουν είναι: G(f(v)) και G(f(v))
  - Οι ατομικοί τύποι είναι μήκους 1, οπότε επιστρέφουμε το σύνολο αναθέσεων γ2.

  Από την κλήση της UNIFY(G(f(v)) , G(u)) μας επιστράφηκε το σύνολο αναθέσεων { u/f(v) }.
  Προσθέτουμε την νέα ανάθεση στο σύνολο αναθέσεων γ μέσω της COMPOSE(γ , { u/f(v) }).
  Άρα το γ που προκύπτει είναι γ = { x/G(f(v)) , u/f(v) }.
  Αντικαθιστούμε τις αναθέσεις των μεταβλητών στους ατομικούς τύπους.
  Και οι τύποι που προκύπτουν είναι: P(G(f(v)) , G(f(v))) και P(G(f(v)) , G(f(v))).
- Οι ατομικοί τύποι είναι μήκους 2, οπότε επιστρέφουμε το σύνολο αναθέσεων γ.

Έτσι, έχουμε καταφέρει να βρούμε τον γενικό ενοποιητή των δύο αυτών τύπων, ο οποίος είναι ο γ που μας επέστρεψε η συνάρτηση.
Δηλαδή ο πιο γενικός ενοποιητής είναι ο **{ x/G(f(v)) , u/f(v) }**
Και έπειτα από τις αναθέσεις αυτές οι δύο ατομικοί τύποι που προκύπτουν είναι:
P(G(f(v)) , G(f(v))) και **P(G(f(v)) , G(f(v)))**

- $P(x_1, G(x_2, x_3), x_2, B)$ και $P(G(H(A, x_5), x_2), x_1, H(A, x_4), x_4)$

$UNIFY(\ P(x_1, G(x_2, x_3), x_2, B)\ ,\ P(G(H(A, x_5), x_2), x_1, H(A, x_4), x_4)\ )$

Αρχικά έχουμε ότι γ={}
- Για i=0: έχουμε UNIFY(P,P). Ισχύει ότι P=P άρα η UNIFY επιστρέφει {}.

- Για i=1 έχουμε $UNIFY(x_1, G(H(A, x_5), x_2))$
  Το πρώτο όρισμα (x1) είναι μεταβλητή, άρα γίνεται η κλήση της $UNIFY-VAR(x_1, G(H(A, x_5), x_2))$
  Η UNIFY-VAR επιστρέφει την ανάθεση $\{\ x_1/G(H(A, x_5), x_2)\ \}$
  Προσθέτουμε την νέα ανάθεση στο γ και μέσω της $COMPOSE(\ \gamma, \{\ x_1/G(H(A, x_5), x_2)\ \})$
  Άρα το γ που προκύπτει είναι $\gamma = \{\ x_1/G(H(A, x_5), x_2)\ \}$
  Αντικαθιστούμε την μεταβλητή x1 με την νέα της ανάθεση και στους δύο ατομικούς τύπους.
  Και οι δύο τύποι που προκύπτουν είναι:
  $P(G(H(A, x_5), x_2), G(x_2, x_3), x_2, B)$ και $P(G(H(A, x_5), x_2), G(H(A, x_5), x_2), H(A, x_4), x_4)$

- Για i=2 έχουμε $UNIFY(G(x_2, x_3), G(H(A, x_5), x_2))$
  Και τα 2 ορίσματα είναι σύμβολα συναρτήσεων, επομένως ακολουθούμε την ίδια διαδικασία αναδρομικά.

γ2={}

- Για i=0 έχουμε UNIFY(G,G). Ισχύει ότι G=G άρα η UNIFY επιστρέφει {}.
- Για i=1 έχουμε $UNIFY(x_2, H(A, x_5))$

  Παρατηρούμε ότι το πρώτο όρισμα (x2) είναι μεταβλητή, άρα γίνεται η κλήση $UNIFY-VAR(x_2, H(A, x_5))$

  Η UNIFY-VAR επιστρέφει την ανάθεση $\{x_2/H(A, x_5)\}$

  Προσθέτουμε την νέα ανάθεση στο γ2 και μέσω της $COMPOSE(\ \gamma 2,\ \{x_2/H(A, x_5)\}\ )$

  Άρα το γ2 που προκύπτει είναι $\gamma 2 = \{\ x_2/H(A, x_5)\ \}$

  Αντικαθιστούμε την μεταβλητή x2 με την νέα της ανάθεση και στους δύο ατομικούς τύπους.

  Και οι δύο τύποι που προκύπτουν είναι: $G(H(A, x_5), x_3)$ και $G(H(A, x_5), H(A, x_5))$
- Για i=2 έχουμε $UNIFY(x_3, H(A, x_5))$

  Παρατηρούμε ότι το πρώτο όρισμα (x3) είναι μεταβλητή, άρα γίνεται η κλήση $UNIFY-VAR(x_3, H(A, x_5))$

  Η UNIFY-VAR επιστρέφει την ανάθεση $\{x_3/H(A, x_5)\}$

  Προσθέτουμε την νέα ανάθεση στο γ και μέσω της $COMPOSE(\ \gamma 2,\ \{x_3/H(A, x_5)\}\ )$

  Άρα το γ2 που προκύπτει είναι $\gamma 2 = \{\ x_2/H(A, x_5)\ ,\ x_3/H(A, x_5)\ \}$

  Αντικαθιστούμε την μεταβλητή x3 με την νέα της ανάθεση και στους δύο ατομικούς τύπους.

  Και οι δύο τύποι που προκύπτουν είναι: $G(H(A, x_5), H(A, x_5))$ και $G(H(A, x_5), H(A, x_5))$
- Οι ατομικοί τύποι είναι μήκους 2, οπότε επιστρέφουμε το γ2.

Από την κλήση της $UNIFY(G(x_2, x_3), G(H(A, x_5), x_2))$ μας επιστράφηκε το $\gamma 2 = \{\ x_2/H(A, x_5)\ ,\ x_3/H(A, x_5)\ \}$

Προσθέτουμε το σύνολο αυτό στο σύνολο αναθέσεων γ μέσω της $COMPOSE(\ \gamma, \{\ x_2/H(A, x_5)\ ,\ x_3/H(A, x_5)\ \}\ )$

Άρα το γ που προκύπτει είναι $\gamma = \{\ x_1/G(H(A, x_5), H(A, x_5))\ ,\ x_2/H(A, x_5)\ ,\ x_3/H(A, x_5)\ \}$

Αντικαθιστούμε τις αναθέσεις των μεταβλητών στους ατομικούς τύπους.

Και οι δύο τύποι που προκύπτουν είναι: $P(G(H(A, x_5), H(A, x_5)), G(H(A, x_5), H(A, x_5)), H(A, x_5), B)$

και $P(G(H(A, x_5), H(A, x_5)), G(H(A, x_5), H(A, x_5)), H(A, x_4), x_4)$

- Για i=3 έχουμε $UNIFY(H(A, x_5), H(A, x_4))$

  Και τα 2 ορίσματα είναι σύμβολα συναρτήσεων, επομένως ακολουθούμε την ίδια διαδικασία αναδρομικά.

  γ2={}

  - Για i=0 έχουμε $UNIFY(H, H)$. Ισχύει ότι H=H άρα η UNIFY επιστρέφει {}.
  - Για i=1 έχουμε $UNIFY(A, A)$. Ισχύει ότι A=A άρα η UNIFY επιστρέφει {}.
  - Για i=2 έχουμε $UNIFY(x_5, x_4)$

    Παρατηρούμε ότι το πρώτο όρισμα (x5) είναι μεταβλητή, άρα γίνεται η κλήση $UNIFY-VAR(x_5, x_4)$

    Η UNIFY-VAR επιστρέφει την ανάθεση $\{x_5/x_4\}$

    Προσθέτουμε την νέα ανάθεση στο γ2 μέσω της $COMPOSE(\ \gamma 2,\ \{x_5/x_4\}\ )$

    Άρα το γ2 που προκύπτει είναι $\gamma 2 = \{\ x_5/x_4\ \}$

    Αντικαθιστούμε την μεταβλητή x5 με την νέα της ανάθεση και στους δύο ατομικούς τύπους.

    Και οι δύο τύποι που προκύπτουν είναι: $H(A, x_4)$ και $H(A, x_4)$
  - Οι ατομικοί τύποι είναι μήκους 2, οπότε επιστρέφουμε το γ2.

  Από την κλήση της $UNIFY(H(A, x_5), H(A, x_4))$ μας επιστράφηκε το $\gamma 2 = \{\ x_5/x_4\ \}$

  Προσθέτουμε το σύνολο αυτό στο σύνολο αναθέσεων γ μέσω της $COMPOSE(\ \gamma, \{\ x_5/x_4\ \}\ )$

  Η μεταβλητή x5 περιέχεται σε προηγούμενες αναθέσεις που υπάρχουν στο γ.

  Έτσι, πρέπει να γίνει η αντίστοιχη ανάθεση και σε αυτές. Επομένως το νέο σύνολο γ που προκύπτει είναι:

  $$\gamma = \{\ x_1/G(H(A, x_4), H(A, x_4))\ ,\ x_2/H(A, x_4)\ ,\ x_3/H(A, x_4)\ ,\ x_5/x_4\ \}$$

  Αντικαθιστούμε τις αναθέσεις των μεταβλητών στους ατομικούς τύπους. Και προκύπτουν οι τύποι:

  $P(G(H(A, x_4), H(A, x_4)), G(H(A, x_4), H(A, x_4)), H(A, x_4), B)$ και

  $P(G(H(A, x_4), H(A, x_4)), G(H(A, x_4), H(A, x_4)), H(A, x_4), x_4)$

- Για i=4 έχουμε $UNIFY(B, x_4)$

  Παρατηρούμε ότι το πρώτο όρισμα (x4) είναι μεταβλητή, άρα γίνεται η κλήση $UNIFY-VAR(x_4, B)$

  Η UNIFY-VAR επιστρέφει την ανάθεση $\{x_4/B\}$

  Προσθέτουμε την νέα ανάθεση στο γ μέσω της $COMPOSE(\ \gamma,\ \{x_4/B\}\ )$

  Η μεταβλητή x4 περιέχεται σε προηγούμενες αναθέσεις που υπάρχουν στο γ.

  Έτσι, πρέπει να γίνει η αντίστοιχη ανάθεση και σε αυτές. Επομένως το νέο σύνολο γ που προκύπτει είναι:

  $$\gamma = \{\ x_1/G(H(A, B), H(A, B))\ ,\ x_2/H(A, B)\ ,\ x_3/H(A, B)\ ,\ x_5/B\ ,\ x_4/B\ \}$$

  Αντικαθιστούμε τις αναθέσεις των μεταβλητών στους ατομικούς τύπους. Και προκύπτουν οι τύποι:

  $P(G(H(A, B), H(A, B)), G(H(A, B), H(A, B)), H(A, B), B)$ και

  $P(G(H(A, B), H(A, B)), G(H(A, B), H(A, B)), H(A, B), B)$

- Οι ατομικοί τύποι έχουν μήκος 4, επομένως επιστρέφουμε το σύνολο γ.

Έτσι, έχουμε καταφέρει να βρούμε τον γενικό ενοποιητή των δύο αυτών τύπων, ο οποίος είναι ο γ που μας επέστρεψε η συνάρτηση.

Δηλαδή ο πιο γενικός ενοποιητής είναι ο: $\{\ x_1/G(H(A, B), H(A, B))\ ,\ x_2/H(A, B)\ ,\ x_3/H(A, B)\ ,\ x_5/B\ ,\ x_4/B\ \}$

Και οι δύο τύποι μετατρέπονται στον: $P(G(H(A, B), H(A, B)), G(H(A, B), H(A, B)), H(A, B), B)$

- $P(x_1, x_2, \ldots, x_n, F(y_0, y_0), \ldots, F(y_{n-1}, y_{n-1}), y_n)$ και $P(F(x_0, x_0), F(x_1, x_1), \ldots, F(x_{n-1}, x_{n-1}), y_1, \ldots, y_n, x_n)$

Η εύρεση ενός γενικού ενοποιητή για τους δύο αυτούς τύπους είναι η χειρότερη περίπτωση όσον αφορά την πολυπλοκότητα για τον αλγόριθμο UNIFY.
Πιο συγκεκριμένα, αρχίζοντας την εφαρμογή του αλγορίθμου:

- Για i=0: έχουμε UNIFY(P,P). Ισχύει ότι P=P άρα η UNIFY επιστρέφει {}.
- Για i=1 έχουμε $UNIFY(x_1, F(x_0, x_0))$
  Το πρώτο όρισμα (x1) είναι μεταβλητή, άρα γίνεται η κλήση της $UNIFY-VAR(x_1, F(x_0, x_0))$
  Η UNIFY-VAR επιστρέφει την ανάθεση $\{\ x_1/F(x_0, x_0)\ \}$
  Προσθέτουμε την νέα ανάθεση στο γ και μέσω της $COMPOSE(\ \gamma, \{\ x_1/F(x_0, x_0)\ \})$
  Άρα το γ που προκύπτει είναι $\gamma = \{\ x_1/F(x_0, x_0)\ \}$
  Αντικαθιστούμε την μεταβλητή x1 με την νέα της ανάθεση και στους δύο ατομικούς τύπους.
  Και οι δύο τύποι που προκύπτουν είναι:
  $P(F(x_0, x_0), x_2, \ldots, x_n, F(y_0, y_0), \ldots, F(y_{n-1}, y_{n-1}), y_n)$ και
  $P(F(x_0, x_0), F(F(x_0, x_0), F(x_0, x_0)), \ldots, F(x_{n-1}, x_{n-1}), y_1, \ldots, y_n, x_n)$
- Για i=2 έχουμε $UNIFY(\ x_2\ ,\ F(F(x_0, x_0), F(x_0, x_0))\ )$
  Το πρώτο όρισμα (x2) είναι μεταβλητή, άρα γίνεται η κλήση της $UNIFY-VAR(\ x_2\ ,\ F(F(x_0, x_0), F(x_0, x_0))\ )$
  Η UNIFY-VAR επιστρέφει την ανάθεση $\{\ x_2/F(F(x_0, x_0), F(x_0, x_0))\ \}$
  Προσθέτουμε την νέα ανάθεση στο γ και μέσω της $COMPOSE(\ \gamma, \{\ x_2/F(F(x_0, x_0), F(x_0, x_0))\ \})$
  Άρα το γ που προκύπτει είναι $\gamma = \{\ x_1/F(x_0, x_0)\ ,\ x_2/F(F(x_0, x_0), F(x_0, x_0))\ \}$
  Αντικαθιστούμε την μεταβλητή x2 με την νέα της ανάθεση και στους δύο ατομικούς τύπους.
  Και οι δύο τύποι που προκύπτουν είναι:
  $P(F(x_0, x_0), F(F(x_0, x_0), F(x_0, x_0)), \ldots, x_n, F(y_0, y_0), \ldots, F(y_{n-1}, y_{n-1}), y_n)$ και
  $P(F(x_0, x_0), F(F(x_0, x_0), F(x_0, x_0)), F(F(F(x_0, x_0), F(x_0, x_0)), F(F(x_0, x_0), F(x_0, x_0))), \ldots, F(y_0, y_0), \ldots, y_n, \ldots)$

Ομοίως συνεχίζουμε και τις επόμενες επαναλήψεις.

- Για i=n έχουμε την κλήση $UNIFY(\ x_n\ ,\ F(x_{n-1}, x_{n-1})\ )$
  Όπου στο xn-1 στην προηγούμενη επανάληψη (για i=n-1) έχει γίνει μια ανάθεση της μορφής :
  $\{\ x_{n-1}/F(F(F\ldots, F\ldots), F(F\ldots, F\ldots))\ \}$
  Άρα τελικά, ομοίως με παραπάνω, για το xn έχουμε μια ανάθεση της μορφής:
  $\{\ x_n/F(F(F(F\ldots, F\ldots), F(F\ldots, F\ldots))\ ,\ F(F(F\ldots, F\ldots), F(F\ldots, F\ldots)))\ \}$

  Δηλαδή περιέχει δύο φορές την ανάθεση του xn-1. Η ανάθεση αυτή προστίθεται στο σύνολο αναθέσεων γ.

- Συνεχίζουμε με την επανάληψη i=n+1 όπου έχουμε την κλήση $UNIFY(F(y_0, y_0), y_1)$
  Το δεύτερο όρισμα (y1) είναι μεταβλητή, άρα γίνεται η κλήση της $UNIFY-VAR(y_1, F(y_0, y_0))$
  Άρα, ομοίως με παραπάνω προκύπτει η ανάθεση $\{\ y_1/F(y_0, y_0)\ \}$ την οποία και προσθέτουμε στο γ.

Για τις επαναλήψεις i=n+1 μέχρι και i=2*n που αντιστοιχούν στις αναθέσεις των μεταβλητών y1,…,yn ακολουθούμε ακριβώς αντίστοιχη διαδικασία που ακολουθούσαμε στις επαναλήψεις από i=1 μέχρι και i=n για τις μεταβλητές x1,…,xn.

- Τελικά καταλήγουμε στην τελευταία επανάληψη για i=2*n+1 οπού έχουμε μια κλήση της μορφής:

$$UNIFY(\ F(F(\ldots F(y_0, y_0), \ldots F(y_0, y_0)), F(\ldots F(y_0, y_0), \ldots F(y_0, y_0)))\ ,\ F(F(\ldots F(x_0, x_0), \ldots F(x_0, x_0)), F(\ldots F(x_0, x_0), \ldots F(x_0, x_0)))\ )$$

καθώς στα xn και yn έχουν γίνει οι αναθέσεις που περιγράφηκαν παραπάνω.
Έπειτα από αναδρομικές κλήσεις, η παραπάνω κλήση της UNIFY καταλήγει στην ανάθεση $\{\ y_0/x_0\ \}$
η οποία προστίθεται στο σύνολο αναθέσεων γ.

Έτσι τελικά έχουμε βρει τον πιο γενικό ενοποιητή ο οποίος είναι της μορφής:

$$MGU = \{\ x_1/F(x_0, x_0)\ ,\ x_2/F(F(x_0, x_0), F(x_0, x_0))\ , \ldots,\ x_n/F(F(F\ldots F_n(x_0, x_0)), F(\ldots F_n(x_0, x_0)))\ ,$$
$$y_1/F(x_0, x_0)\ ,\ y_2/F(F(x_0, x_0), F(x_0, x_0))\ , \ldots,\ y_n/F(F(F\ldots F_n(x_0, x_0)), F(\ldots F_n(x_0, x_0)))\ ,\ y_0/x_0\ \}$$

Παρατηρούμε ότι έχουμε μια ανάθεση για όλες τις μεταβλητές και έχουμε καταλήξει σε έναν ενοποιητή στον οποίον κάθε μεταβλητή x_i και y_i θα είναι δεσμευμένη σε έναν όρο με $2^{i+1} - 1$ σύμβολα.
Βέβαια, στον γενικό ενοποιητή που βρήκαμε για τους παραπάνω όρους περιέχονται οι ίδιοι υποόροι πολλές φορές.

Άσκηση 3

Ερώτημα α

Αρχικά θα ορίσω τα σύμβολα σταθερών. Τα σύμβολα σταθερών θα αναπαρίστανται στην ελληνική γλώσσα και θα αρχίζουν με κεφαλαίο γράμμα.
Έχουμε τα σύμβολα σταθερών για τους ανθρώπους που περιέχονται στις προτάσεις που μας δόθηκαν. Οι σταθερές αυτές είναι: **Κυριάκος** , **Αλέξης** , **Φώφη**
Επίσης έχουμε τις σταθερές σχετικά με τα κοινωνικά/οικονομικά συστήματα που μπορεί να αρέσουν στον κάθε έναν, δηλαδή: **Σοσιαλισμός** , **Καπιταλισμός**
Άρα έχουμε τα σύμβολα σταθερών: **Κυριάκος, Αλέξης, Φώφη, Σοσιαλισμός** , **Καπιταλισμός**

Στην συνέχεια θα ορίσω τα εξής σύμβολα κατηγορημάτων:

- **ΜέλοςΚΟΡΩΝΑ**(x): Το μοναδιαίο κατηγόρημα αυτό χρησιμοποιείται για δηλώσουμε ότι ένα άτομο ανήκει στο πολιτικό κόμμα ΚΟΡΩΝΑ.
- **Δεξιός**(x): Το μοναδιαίο κατηγόρημα αυτό χρησιμοποιείται για να δηλώσουμε αν ένα άτομο είναι δεξιός.
- **Φιλελεύθερος**(x): Το μοναδιαίο κατηγόρημα αυτό χρησιμοποιείται για να δηλώσουμε αν ένα άτομο είναι φιλελεύθερο.
- **Αρέσει**(x,y): Το δυαδικό κατηγόρημα αυτό χρησιμοποιείται για δηλώσουμε αν κάτι αρέσει σε ένα άτομο ή όχι.

Έχουμε ορίσει κατάλληλα όλα τα σύμβολα τα οποία χρειαζόμαστε. Οπότε στην συνέχεια θα μετατρέψουμε τις προτάσεις σε λογική πρώτης τάξης.

I. Ο Κυριάκος, ο Αλέξης και η Φώφη είναι μέλη του πολιτικού κόμματος ΚΟΡΩΝΑ.

Μπορούμε να “σπάσουμε” την πρόταση αυτήν σε τρεις ξεχωριστές εξάγοντας έτσι τους τύπους:
ΜέλοςΚΟΡΩΝΑ(Κυριάκος) , **ΜέλοςΚΟΡΩΝΑ(Αλέξης)** , **ΜέλοςΚΟΡΩΝΑ(Φώφη)**

II. Κάθε μέλος του κόμματος ΚΟΡΩΝΑ που δεν είναι δεξιός, είναι φιλελεύθερος.

Από την πρόταση αυτή μπορούμε να εξάγουμε τον τύπο:

$$(\forall x)\left(\text{ΜέλοςΚΟΡΩΝΑ}(x) \wedge (\neg \text{Δεξιός}(x)) \Longrightarrow \text{Φιλελεύθερος}(x)\right)$$

III. Στους δεξιούς δεν αρέσει ο σοσιαλισμός.
Από την πρόταση αυτή μπορούμε να εξάγουμε τον τύπο:

$$(\forall x)\left(\text{Δεξιός}(x) \Longrightarrow \neg\,\text{Αρέσει}(x, \text{Σοσιαλισμός})\right)$$

IV. Σ’ όποιον δεν αρέσει ο καπιταλισμός, δεν είναι φιλελεύθερος.
Από την πρόταση αυτή μπορούμε να εξάγουμε τον τύπο:

$$(\forall x)\left(\neg\,\text{Αρέσει}(x, \text{Καπιταλισμός}) \Longrightarrow \neg\,\text{Φιλελεύθερος}(x)\right)$$

V. Στον Κυριάκο δεν αρέσει ό,τι αρέσει στον Αλέξη, και του αρέσει ό,τι δεν αρέσει στον Αλέξη.
Από την πρόταση αυτή μπορούμε να εξάγουμε τον τύπο:

$$(\forall x)\left(\text{Αρέσει}(\text{Κυριάκος}, x) \Longleftrightarrow \neg\,\text{Αρέσει}(\text{Αλέξης}, x)\right)$$

VI. Στον Αλέξη αρέσει ο σοσιαλισμός και ο καπιταλισμός.
Αν “σπάσουμε” την πρόταση αυτή, μπορούμε να εξάγουμε τους εξής δύο τύπους:

Αρέσει(Αλέξης , Σοσιαλισμός) και **Αρέσει(Αλέξης , Καπιταλισμός)**

Επομένως η βάση γνώσης **KB** που προκύπτει από τις πρώτες έξι προτάσεις είναι η εξής:

$$\text{ΜέλοςΚΟΡΩΝΑ(Κυριάκος)} , \text{ΜέλοςΚΟΡΩΝΑ(Αλέξης)} , \text{ΜέλοςΚΟΡΩΝΑ(Φώφη)}$$

$$(\forall x)\,(\ \text{ΜέλοςΚΟΡΩΝΑ}(x) \wedge (\neg\,\text{Δεξιός}(x)) \Longrightarrow \text{Φιλελεύθερος}(x)\ )$$

$$(\forall x)\,(\ \text{Δεξιός}(x) \Longrightarrow \neg\,\text{Αρέσει}(x , \text{Σοσιαλισμός})\ )$$

$$(\forall x)\,(\ \neg\,\text{Αρέσει}(x , \text{Καπιταλισμός}) \Longrightarrow \neg\ \text{Φιλελεύθερος}(x)\ )$$

$$(\forall x)\,(\ \text{Αρέσει}(\text{Κυριάκος} , x) \Longleftrightarrow \neg\,\text{Αρέσει}(\text{Αλέξης} , x)\ )$$

$$\text{Αρέσει(Αλέξης , Καπιταλισμός)} \qquad \text{Αρέσει(Αλέξης , Σοσιαλισμός)}$$

Τέλος μετατρέπουμε και την πρόταση (vii) σε λογική πρώτης τάξης και έχουμε:
VII. Υπάρχει ένα μέλος του κόμματος ΚΟΡΩΝΑ που είναι φιλελεύθερος αλλά δεν είναι δεξιός
Από την πρόταση αυτή μπορούμε να εξάγουμε τον τύπο:

$$\phi = (\exists x)\,(\ \text{ΜέλοςΚΟΡΩΝΑ}(x) \wedge \text{Φιλελεύθερος}(x) \wedge \neg\ \text{Δεξιός}(x)\ )$$

<u>Ερώτημα β</u>

Για να εφαρμόσουμε ανάλυση, πρέπει πρώτα να μετατρέψουμε όλους τους τύπους της KB σε συζευκτική κανονική μορφή (CNF).

- Για την: $(\forall x)\,(\ \text{ΜέλοςΚΟΡΩΝΑ}(x) \wedge (\neg\,\text{Δεξιός}(x)) \Longrightarrow \text{Φιλελεύθερος}(x)\ )$

  Αρχικά απαλείφουμε την συνεπαγωγή και έχουμε:

$$(\forall x)\,(\ \neg\ \text{ΜέλοςΚΟΡΩΝΑ}(x) \vee \text{Δεξιός}(x) \vee \text{Φιλελεύθερος}(x)\ )$$

  Απαλοιφή του καθολικού ποσοδείκτη:

$$\neg\text{ΜέλοςΚΟΡΩΝΑ}(x) \vee \text{Δεξιός}(x) \vee \text{Φιλελεύθερος}(x)$$

- Για την $(\forall x)\,(\ \text{Δεξιός}(x) \Longrightarrow \neg\,\text{Αρέσει}(x , \text{Σοσιαλισμός})\ )$

  Αρχικά απαλείφουμε την συνεπαγωγή και έχουμε:

$$(\forall x)(\ \neg\,\text{Δεξιός}(x) \vee \neg\,\text{Αρέσει}(x , \text{Σοσιαλισμός})\ )$$

  Απαλοιφή του καθολικού ποσοδείκτη και τελικά έχουμε:

$$\neg\,\text{Δεξιός}(x) \vee \neg\,\text{Αρέσει}(x , \text{Σοσιαλισμός})$$

- Για την: $(\forall x)$ (¬ **Αρέσει(x , Καπιταλισμός)** $\Longrightarrow$ ¬ **Φιλελεύθερος(x)**)

  Αρχικά απαλείφουμε την συνεπαγωγή και έχουμε:

  $(\forall x)$ (**Αρέσει(x , Καπιταλισμός)** $\vee$ ¬ **Φιλελεύθερος(x)**)

  Απαλοιφή του καθολικού ποσοδείκτη:

  Αρέσει(x , Καπιταλισμός) $\vee$ ¬ **Φιλελεύθερος(x)**

- Για την: $(\forall x)$ (**Αρέσει(Κυριάκος , x)** $\Longleftrightarrow$ ¬ **Αρέσει(Αλέξης , x)**)

  Αρχικά απαλείφουμε την διπλή συνεπαγωγή και έχουμε:

$(\forall x)$(**Αρέσει(Κυριάκος , x)** $\Rightarrow$ ¬ **Αρέσει(Αλέξης , x)** $\wedge$ ¬**Αρέσει(Αλέξης , x)** $\Rightarrow$ **Αρέσει(Κυριάκος , x)**)

  Απαλείφουμε τις απλές συνεπαγωγές και έχουμε:

$(\forall x)$((¬**Αρέσει(Κυριάκος , x)** $\vee$ ¬ **Αρέσει(Αλέξης , x)**) $\wedge$ (**Αρέσει(Αλέξης , x)** $\vee$ **Αρέσει(Κυριάκος , x)**))

  Απαλοιφή του καθολικού ποσοδείκτη:

(¬ **Αρέσει(Κυριάκος , x)** $\vee$ ¬ **Αρέσει(Αλέξης , x)**) $\wedge$ (**Αρέσει(Αλέξης , x)** $\vee$ **Αρέσει(Κυριάκος , x)**)

  Τελική μορφή:

  ¬ **Αρέσει(Κυριάκος , x)** $\vee$ ¬ **Αρέσει(Αλέξης , x)** και

  Αρέσει(Αλέξης , x) $\vee$ **Αρέσει(Κυριάκος , x)**

Τέλος, για να δείξουμε ότι $KB \models \phi$ προσθέτουμε την άρνηση της φ στην KB, δηλαδή την:

¬ ($(\exists x)$ (**ΜέλοςΚΟΡΩΝΑ(x)** $\wedge$ **Φιλελεύθερος(x)** $\wedge$ ¬ **Δεξιός(x)**))

Μετακινούμε την άρνηση προς τα μέσα:

$(\forall x)$ ¬ (**ΜέλοςΚΟΡΩΝΑ(x)** $\wedge$ **Φιλελεύθερος(x)** $\wedge$ ¬ **Δεξιός(x)**)

Το οποίο είναι ισοδύναμο με το:

$(\forall x)$ (¬ **ΜέλοςΚΟΡΩΝΑ(x)** $\vee$ ¬ **Φιλελεύθερος(x)** $\vee$ **Δεξιός(x)**)

Απαλείφουμε τον καθολικό ποσοδείκτη και τελικά έχουμε

¬ **ΜέλοςΚΟΡΩΝΑ(x)** $\vee$ ¬ **Φιλελεύθερος(x)** $\vee$ **Δεξιός(x)**

Άρα τελικά η **KB** που προκύπτει είναι η:

ΜέλοςΚΟΡΩΝΑ(Κυριάκος) , **ΜέλοςΚΟΡΩΝΑ(Αλέξης)** , **ΜέλοςΚΟΡΩΝΑ(Φώφη)**

¬**ΜέλοςΚΟΡΩΝΑ(x)** $\vee$ **Δεξιός(x)** $\vee$ **Φιλελεύθερος(x)**

¬ **Δεξιός(x)** $\vee$ ¬ **Αρέσει(x , Σοσιαλισμός)**

Αρέσει(x , Καπιταλισμός) $\vee$ ¬ **Φιλελεύθερος(x)**

¬ **Αρέσει(Κυριάκος , x)** $\vee$ ¬ **Αρέσει(Αλέξης , x)** **Αρέσει(Αλέξης , x)** $\vee$ **Αρέσει(Κυριάκος , x)**

Αρέσει(Αλέξης , Καπιταλισμός) **Αρέσει(Αλέξης , Σοσιαλισμός)**

¬ **ΜέλοςΚΟΡΩΝΑ(x)** $\vee$ ¬ **Φιλελεύθερος(x)** $\vee$ **Δεξιός(x)**

Αφού έχουμε προσθέσει την άρνηση της φ στην ΚΒ, για να δείξουμε ότι $KB \vDash \phi$ πρέπει να εφαρμόσουμε τον κανόνα τις ανάλυσης (όσες φορές χρειάζεται) έως ότου καταλήξουμε την κενή φράση.

Απο τους τύπους:

- **Αρέσει(Αλέξης , Σοσιαλισμός)**
- **¬ Δεξιός(x) ∨ ¬ Αρέσει(x , Σοσιαλισμός)**

και **MGU { x/Αλέξης }** παίρνουμε:

¬ Δεξιός(Αλέξης)

Απο τους τύπους:

- **¬ Δεξιός(Αλέξης)** (ο οποίος προέκυψε από την παραπάνω εφαρμογή της ανάλυσης)
- **¬ ΜέλοςΚΟΡΩΝΑ(x) ∨ ¬ Φιλελεύθερος(x) ∨ Δεξιός(x)**

και **MGU { x/Αλέξης }** παίρνουμε:

¬ ΜέλοςΚΟΡΩΝΑ(Αλέξης) ∨ ¬ Φιλελεύθερος(Αλέξης)

Απο τους τύπους:

- **¬ ΜέλοςΚΟΡΩΝΑ(Αλέξης) ∨ ¬ Φιλελεύθερος(Αλέξης)** (ο οποίος προέκυψε προηγουμένως)
- **ΜέλοςΚΟΡΩΝΑ(Αλέξης)**

και **MGU { }** παίρνουμε: **¬ Φιλελεύθερος(Αλέξης)** **(1)**

Απο τους τύπους:

- **¬ΜέλοςΚΟΡΩΝΑ(x) ∨ Δεξιός(x) ∨ Φιλελεύθερος(x)**
- **ΜέλοςΚΟΡΩΝΑ(Αλέξης)**

και **MGU { x/Αλέξης }** παίρνουμε:

Δεξιός(Αλέξης) ∨ Φιλελεύθερος(Αλέξης)

Απο τους τύπους:

- **Δεξιός(Αλέξης) ∨ Φιλελεύθερος(Αλέξης)** (ο οποίος προέκυψε από την αμέσως παραπάνω εφαρμογή)
- **¬ Δεξιός(Αλέξης)** (ο οποίος προέκυψε προηγουμένως)

και **MGU { }** παίρνουμε: **Φιλελεύθερος(Αλέξης)** **(2)**

Εφαρμόζουμε ανάλυση στους τύπους **(1)** και **(2)** που έχουν προκύψει παραπάνω, δηλαδή στους τύπους:

- **¬ Φιλελεύθερος(Αλέξης)**
- **Φιλελεύθερος(Αλέξης)**

Καταλήγουμε στην κενή φράση, δηλαδή σε μία αντίφαση.
Άρα αφού το σύνολο φράσεων που χρησιμοποιήσαμε είναι το $KB \cup \{\neg\phi\}$ και αφού αποδείξαμε ότι είναι μη ικανοποιήσιμο, μπορούμε να συμπεράνουμε ότι $KB \vDash \phi$.

Ερώτημα γ

Για να βρούμε το μέλος του ΚΟΡΩΝΑ που έχει την ιδιότητα που παριστάνει η φ, θα ακολουθήσουμε παρόμοια διαδικασία με το ερώτημα β με την διαφορά ότι αντί να προσθέσουμε την άρνηση της φ στην ΚΒ θα προσθέσουμε τον τύπο: $Ans(x) \vee \neg\phi$, όπου x η ελεύθερη μεταβλητή του φ.

Για να βρούμε την απάντηση που θέλουμε, πρέπει να καταλήξουμε σε μία φράση η οποία περιέχει μόνο το λεκτικό απάντησης Ans(x). Η σταθερά που θα είναι δεσμευμένη στην μεταβλητή x στην φράση αυτή θα είναι και το μέλος του ΚΟΡΩΝΑ που ψάχνουμε.

Ο τύπος του $Ans(x) \vee \neg\phi$ σε CNF μορφή είναι:

Ans(x) ∨ ¬ ΜέλοςΚΟΡΩΝΑ(x) ∨ ¬ Φιλελεύθερος(x) ∨ Δεξιός(x)

Άρα η **ΚΒ** που προκύπτει είναι:

ΜέλοςΚΟΡΩΝΑ(Κυριάκος) , ΜέλοςΚΟΡΩΝΑ(Αλέξης) , ΜέλοςΚΟΡΩΝΑ(Φώφη)

¬ΜέλοςΚΟΡΩΝΑ(x) ∨ Δεξιός(x) ∨ Φιλελεύθερος(x)

¬ Δεξιός(x) ∨ ¬ Αρέσει(x , Σοσιαλισμός)

Αρέσει(x , Καπιταλισμός) ∨ ¬ Φιλελεύθερος(x)

¬ Αρέσει(Κυριάκος , x) ∨ ¬ Αρέσει(Αλέξης , x) **Αρέσει(Αλέξης , x) ∨ Αρέσει(Κυριάκος , x)**

Αρέσει(Αλέξης , Καπιταλισμός) **Αρέσει(Αλέξης , Σοσιαλισμός)**

Ans(x) ∨ ¬ ΜέλοςΚΟΡΩΝΑ(x) ∨ ¬ Φιλελεύθερος(x) ∨ Δεξιός(x)

Ακολουθούμε την ίδια διαδικασία με το ερώτημα β.

Απο τους τύπους:

- **Αρέσει(Αλέξης , Σοσιαλισμός)**
- **¬ Δεξιός(x) ∨ ¬ Αρέσει(x , Σοσιαλισμός)**

και **MGU { x/Αλέξης }** παίρνουμε:

¬ Δεξιός(Αλέξης)

Απο τους τύπους:

- **¬ Δεξιός(Αλέξης)** (ο οποίος προέκυψε από την παραπάνω εφαρμογή της ανάλυσης)
- **Ans(x) ∨ ¬ ΜέλοςΚΟΡΩΝΑ(x) ∨ ¬ Φιλελεύθερος(x) ∨ Δεξιός(x)**

και **MGU { x/Αλέξης }** παίρνουμε:

Ans(Αλέξης) ∨ ¬ ΜέλοςΚΟΡΩΝΑ(Αλέξης) ∨ ¬ Φιλελεύθερος(Αλέξης)

Απο τους τύπους:

- **Ans(Αλέξης) ∨ ¬ ΜέλοςΚΟΡΩΝΑ(Αλέξης) ∨ ¬ Φιλελεύθερος(Αλέξης)** (ο οποίος προέκυψε προηγουμένως)
- **ΜέλοςΚΟΡΩΝΑ(Αλέξης)**

και **MGU { }** παίρνουμε: **Ans(Αλέξης) ∨ ¬Φιλελεύθερος(Αλέξης)** **(1)**

Απο τους τύπους:

- **¬ΜέλοςΚΟΡΩΝΑ(x) ∨ Δεξιός(x) ∨ Φιλελεύθερος(x)**
- **ΜέλοςΚΟΡΩΝΑ(Αλέξης)**

και **MGU { x/Αλέξης }** παίρνουμε:

Δεξιός(Αλέξης) ∨ Φιλελεύθερος(Αλέξης)

Απο τους τύπους:

- **Δεξιός(Αλέξης) ∨ Φιλελεύθερος(Αλέξης)** (ο οποίος προέκυψε από την αμέσως παραπάνω εφαρμογή)
- **¬ Δεξιός(Αλέξης)** (ο οποίος προέκυψε προηγουμένως)

και **MGU { }** παίρνουμε: **Φιλελεύθερος(Αλέξης)** **(2)**

Εφαρμόζουμε ανάλυση στους τύπους **(1)** και **(2)** που έχουν προκύψει παραπάνω, δηλαδή στους τύπους:

- **Ans(Αλέξης) ∨ ¬Φιλελεύθερος(Αλέξης)**
- **Φιλελεύθερος(Αλέξης)**

Καταλήγουμε στην φράση: **Ans(Αλέξης)**

Δηλαδή καταλήξαμε σε μια φράση που περιέχει μόνο το λεκτικό απάντησης. Το λεκτικό αυτό, περιέχει την σταθερά Αλέξης, η οποία είναι η απάντηση στο ερώτημα μας.

Επομένως το μέλος του ΚΟΡΩΝΑ που έχει την ιδιότητα που παριστάνει η φ είναι ο Αλέξης.

Άσκηση 4

Ερώτημα α

- **A:** $(\forall x)(\forall s)(\forall t)(In(x,s) \land In(x,t) \iff In(x, Intersection(s,t)))$

Αρχικά απαλείφουμε την διπλή συνεπαγωγή και έχουμε:

$$(\forall x)(\forall s)(\forall t)(\ (\ In(x,s) \land In(x,t) \implies In(x, Intersection(s,t)\ )\ \land\ (\ In(x, Intersection(s,t)) \implies In(x,s) \land In(x,t)\ )\ )$$

Στην συνέχεια απαλείφουμε τις απλές συνεπαγωγές και έχουμε:

$$(\forall x)(\forall s)(\forall t)(\ (\neg(In(x,s) \land In(x,t)) \lor In(x, Intersection(s,t)\ )\ \land\ (\ \neg In(x, Intersection(s,t)) \lor (In(x,s) \land In(x,t))\ )\ )$$

Μετακινούμε την άρνηση προς τα μέσα και έχουμε:

$$(\forall x)(\forall s)(\forall t)(\ ((\neg In(x,s) \lor \neg In(x,t)) \lor In(x, Intersection(s,t)\ )\ \land\ (\ \neg In(x, Intersection(s,t)) \lor (In(x,s) \land In(x,t))\ )\ )$$

Απαλείφουμε τους τρεις καθολικούς ποσοδείκτες και έχουμε:

$$((\neg In(x,s) \lor \neg In(x,t)) \lor In(x, Intersection(s,t)\ )\ \land\ (\ \neg In(x, Intersection(s,t)) \lor (In(x,s) \land In(x,t))\ )$$

Εφαρμόζουμε την επιμεριστική ιδιότητα του V ως προς το Λ και έχουμε

$$((\neg In(x,s) \lor \neg In(x,t)) \lor In(x, Intersection(s,t))\ )\ \land\ (\ (\neg In(x, Intersection(s,t)) \lor In(x,s)) \land (\neg In(x, Intersection(s,t)) \lor In(x,t))\ )$$

Αφαιρούμε τις παρενθέσεις που δεν χρειάζονται:

$$(\neg In(x,s) \lor \neg In(x,t) \lor In(x, Intersection(s,t))\ )\ \land\ (\neg In(x, Intersection(s,t)) \lor In(x,s)) \land (\neg In(x, Intersection(s,t)) \lor In(x,t))$$

Έχουν προκύψει τρεις φράσεις οι οποίες συνδέονται με τον λογικό σύνδεσμο Λ.
Έτσι, μπορούμε να τις "σπάσουμε" στις τρεις εξής φράσεις. **Η CNF της πρότασης Α είναι:**

- $\neg In(x,s) \lor \neg In(x,t) \lor In(x, Intersection(s,t))$
- $\neg In(x, Intersection(s,t)) \lor In(x,s)$
- $\neg In(x, Intersection(s,t)) \lor In(x,t)$

- **B:** $(\forall s)(\forall t)(\ (\forall x)(In(x,s) \implies In(x,t)) \implies SubsetOf(s,t)\ )$

Αρχικά απαλείφουμε τις απλές συνεπαγωγές και έχουμε:

$$(\forall s)(\forall t)(\ \neg(\forall x)(\neg In(x,s) \lor In(x,t)) \lor SubsetOf(s,t)\ )$$

Μετακινούμε την άρνηση προς τα μέσα και έχουμε:

$$(\forall s)(\forall t)(\ (\exists x)(In(x,s) \land \neg In(x,t)) \lor SubsetOf(s,t)\ )$$

Αντικαθιστούμε τον υπαρξιακό ποσοδείκτη (μετατροπή κατά Skolem). Επειδή ο υπαρξιακός ποσοδείκτης βρίσκεται στην εμβέλεια των καθολικών ποσοδεικτών $\forall s$, $\forall t$, απαλείφουμε την τον υπαρξιακό ποσοδείκτη $\exists x$ και αντικαθιστούμε όλες τις εμφανίσεις της μεταβλητής x με τον όρο $F(s,t)$ όπου F είναι το σύμβολο συνάρτησης Skolem. Τελικά έχουμε

$$(\forall s)(\forall t)(\ (In(F(s,t),s) \land \neg In(F(s,t),t)) \lor SubsetOf(s,t)\ )$$

Στην συνέχεια απαλείφουμε τους δύο καθολικούς ποσοδείκτες και έχουμε:

$$(In(F(s,t),s) \land \neg In(F(s,t),t)) \lor SubsetOf(s,t)$$

Εφαρμόζουμε την επιμεριστική ιδιότητα του V ως προς το Λ:

$$(In(F(s,t),s) \lor SubsetOf(s,t)) \land (\neg In(F(s,t),t) \lor SubsetOf(s,t))$$

Τέλος, "σπάμε" τις δύο φράσεις που χωρίζονται με το (Λ) , αφαιρούμε τις παρενθέσεις και προκύπτουν **οι εξής δύο φράσεις οι οποίες αποτελούν την CNF μορφή της πρότασης Β**:

- $In(F(s,t),s) \lor SubsetOf(s,t)$
- $\neg In(F(s,t),t) \lor SubsetOf(s,t)$

- **¬ C :** $\neg(\ (\forall s)(\forall t) SubsetOf(Intersection(s,t),s)\ )$

  Αρχικά μετακινούμε την άρνηση προς τα μέσα και έχουμε:

  $$(\exists s)(\exists t)\neg SubsetOf(Intersection(s,t),s)$$

  Στην συνέχεια αντικαθιστούμε τον υπαρξιακό ποσοδείκτη (μετατροπή κατά Skolem).
  Επειδή οι δύο υπαρξιακοί ποσοδείκτες δεν βρίσκονται στην εμβέλεια κάποιου καθολικού ποσοδείκτη, απλά απαλείφουμε τους ποσοδείκτες και αντικαθιστούμε τις εμφανίσεις των μεταβλητών τους με τις αντίστοιχες σταθερές Skolem. Έστω S η σταθερά Skolem που αντιστοιχεί στο $\exists s$ και T η σταθερά Skolem που αντιστοιχεί στο $\exists t$. **Έτσι προκύπτει η CNF μορφή της πρότασης ¬ C η οποία είναι:**

  $$\neg SubsetOf(Intersection(S,T),S)$$

<u>Ερώτημα β</u>

Θέλουμε να χρησιμοποιήσουμε ανάλυση έτσι ώστε να δείξουμε ότι η πρόταση C είναι συνέπεια των προτάσεων A και B. Έτσι αρχικά δημιουργούμε την βάση γνώσης KB από τις προτάσεις A και B. Θέλουμε να δείξουμε ότι η πρόταση C έπεται λογικά από την KB που περιέχει τις προτάσεις A και B. Έτσι προσθέτουμε και την άρνηση της πρότασης C στην βάση γνώσης και μέσω εφαρμογών των τύπων της ανάλυσης προσπαθούμε να καταλήξουμε σε μια κενή φράση.
Έχουμε ήδη βρει στο ερώτημα α τις CNF μορφές των προτάσεων αυτών οπότε η βάση γνώσης **KB** που προκύπτει είναι η:

$$\neg In(x,s) \lor \neg In(x,t) \lor In(x, Intersection(s,t))$$

$$\neg In(x, Intersection(s,t)) \lor In(x,s) \qquad \neg In(x, Intersection(s,t)) \lor In(x,t)$$

$$In(F(s,t),s) \lor SubsetOf(s,t) \qquad \neg In(F(s,t),t) \lor SubsetOf(s,t)$$

$$\neg SubsetOf(Intersection(S,T),S)$$

* <u>Παρατήρηση</u>: Τα S και T είναι σταθερές Skolem (όπως αναφέρθηκε παραπάνω)

Απο τους τύπους της KB:

- $In(F(s,t),s) \lor SubsetOf(s,t)$
- $\neg SubsetOf(Intersection(S,T),S)$

και $MGU = \{\ s/Intersection(S,T)\ ,\ t/S\ \}$ παίρνουμε:

$$In(\ F(Intersection(S,T),S)\ ,\ Intersection(S,T)\ ) \quad \textbf{(1)}$$

Απο τους τύπους:

- $In(\ F(Intersection(S,T),S)\ ,\ Intersection(S,T)\ )$ (που προέκυψε στην (1))
- $\neg In(x, Intersection(s,t)) \lor In(x,s)$ (Από την KB)

και $MGU = \{\ x/F(Intersection(S,T),S)\ ,\ s/S\ ,\ t/T\ \}$ παίρνουμε:

$$In(\ F(Intersection(S,T),S)\ ,\ S\ ) \quad \textbf{(2)}$$

Απο τους τύπους:

- $In(\ F(Intersection(S,T),S)\ ,\ S\ )$ (που προέκυψε στην (2))
- $\neg In(F(s,t),t) \vee SubsetOf(s,t)$ (Από την KB)

και $MGU\ =\ \{\ s/Intersection(S,T)\ ,\ t/S\ \}$ παίρνουμε:

$$SubsetOf(Intersection(S,T),S) \qquad \textbf{(3)}$$

Απο τους τύπους:

- $SubsetOf(Intersection(S,T),S)$ (που προέκυψε στην (3))
- $\neg SubsetOf(Intersection(S,T),S)$ (Από την KB)

και $MGU\ =\ \{\ \ \}$ παίρνουμε την κενή φράση.

Άρα τελικά καταλήξαμε σε μια αντίφαση. Αφού η KB περιέχει τις προτάσεις A και B καθώς και την άρνηση της πρότασης C και καταλήξαμε στην κενή φράση, συμπεραίνουμε ότι η πρόταση C έπεται λογικά από τις προτάσεις A και B.
Επομένως χρησιμοποιώντας ανάλυση αποδείξαμε το ζητούμενο, δηλαδή ότι Η πρόταση C είναι συνέπεια των προτάσεων A και B.

Άσκηση 5

Αρχικά πρέπει να μετατρέψουμε όλες τις προτάσεις σε φράσεις Horn.
Θα χρησιμοποιήσω τα εξής σύμβολα κατηγορημάτων:
- **Pretty(x)** : Δηλώνει το αν κάποιος είναι όμορφος.
- **Rich(x)** : Δηλώνει το αν κάποιος είναι πλούσιος.
- **Muscly(x)** : Δηλώνει το αν κάποιος είναι μυώδης.
- **Kind(x)** : Δηλώνει το αν κάποιος είναι ευγενικός.
- **Likes(x,y)** : Δηλώνει το αν κάποιος στον/στην x αρέσει ο/η y.
- **Man(x)** : Δηλώνει το αν κάποιος είναι άνδρας.
- **Woman(x)** : Δηλώνει το αν κάποιος είναι γυναίκα.
- **Happy(x)** : Δηλώνει το αν κάποιος είναι ευτυχισμένος.

Επίσης αντιστοιχίζω τα ονόματα ατόμων που υπάρχουν στις προτάσεις σε σύμβολα σταθερών ως εξής:
- Ελένη = **Helen**
- Γιάννης = **John**
- Πέτρος = **Peter**
- Τίμος = **Tim**
- Κατερίνα = **Catherine**

Χρησιμοποιώντας τα σύμβολα αυτά μετατρέπουμε τις προτάσεις και έχουμε:

- Η Ελένη είναι όμορφη
    - $Pretty(Helen)$
- Ο Γιάννης είναι όμορφος και πλούσιος.
    - $Pretty(John)$
    - $Rich(John)$
- Ο Πέτρος είναι μυώδης και πλούσιος
    - $Muscly(Peter)$
    - $Rich(Peter)$

- Ο Τίμος είναι μυώδης και ευγενικός.
  - $Muscly(Tim)$
  - $Kind(Tim)$
- Σε όλους τους άνδρες αρέσουν οι όμορφες γυναίκες.
  - $Man(x) \wedge Woman(y) \wedge Pretty(y) \implies Likes(x, y)$

    (το οποίο μπορεί να γραφτεί και ως: $\neg Man(x) \vee \neg Woman(y) \vee \neg Pretty(y) \vee Likes(x, y)$)
- Όλοι οι πλούσιοι είναι ευτυχισμένοι.
  - $Rich(x) \implies Happy(x)$
- Όλοι οι άνδρες που τους αρέσει μια γυναίκα, στην οποία αρέσουν, είναι ευτυχισμένοι
  - $Man(x) \wedge Woman(y) \wedge Likes(x, y) \wedge Likes(y, x) \implies Happy(x)$
- Όλες οι γυναίκες που τους αρέσει ένας άνδρας, στον οποίο αρέσουν, είναι ευτυχισμένες.
  - $Man(x) \wedge Woman(y) \wedge Likes(x, y) \wedge Likes(y, x) \implies Happy(y)$
- Στην Κατερίνα αρέσουν όλοι οι άνδρες, στους οποίους η ίδια αρέσει.
  - $Man(x) \wedge Likes(x, Catherine) \implies Likes(Catherine, x)$
- Στην Ελένη αρέσουν όλοι οι άνδρες που είναι ευγενικοί και πλούσιοι ή μυώδεις και όμορφοι
  - $Man(x) \wedge Kind(x) \wedge Rich(x) \implies Likes(Helen, x)$
  - $Man(x) \wedge Muscly(x) \wedge Pretty(x) \implies Likes(Helen, x)$

Τέλος προσθέτουμε για κάθε άτομο την πληροφορία για το αν είναι άντρας ή γυναίκα. Η πληροφορία αυτή δεν αναφέρεται στις προτάσεις αλλά είναι απαραίτητη για την ερμηνεία τους και αποτελεί προηγούμενη γνώση (background knowledge).
Άρα η **KB** που προκύπτει είναι:

$Man(Tim)$ $Man(Peter)$ $Man(John)$ $Woman(Helen)$ $Woman(Catherine)$

$Muscly(Tim)$ $Muscly(Peter)$ $Pretty(John)$ $Pretty(Helen)$

$Kind(Tim)$ $Rich(Peter)$ $Rich(John)$

$Man(x) \wedge Woman(y) \wedge Pretty(y) \implies Likes(x, y)$ $Rich(x) \implies Happy(x)$

$$Man(x) \wedge Woman(y) \wedge Likes(x, y) \wedge Likes(y, x) \implies Happy(x)$$

$$Man(x) \wedge Woman(y) \wedge Likes(x, y) \wedge Likes(y, x) \implies Happy(y)$$

$$Man(x) \wedge Likes(x, Catherine) \implies Likes(Catherine, x)$$

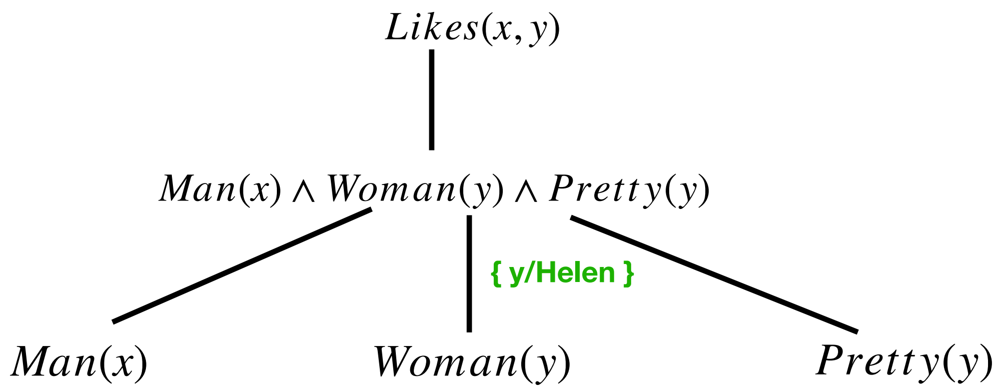
$$Man(x) \wedge Kind(x) \wedge Rich(x) \implies Likes(Helen, x)$$

$$Man(x) \wedge Muscly(x) \wedge Pretty(x) \implies Likes(Helen, x)$$

Ερώτημα α

- Ποιός αρέσει σε ποιόν

Θα χρησιμοποιήσω Backward chaining.
Για την πρόταση $Man(x) \wedge Woman(y) \wedge Pretty(y) \implies Likes(x,y)$ έχουμε:



Για την ανάθεση { y/Helen } παρατηρούμε ότι ικανοποιούνται οι τύπου Woman(y) και Pretty(y).
Επίσης, ο τύπος Man(x) ικανοποιείται για οποιαδήποτε από τις αναθέσεις:

{ x/Peter } , { x/Tim } , { x/John }

Άρα εφαρμόζοντας κάθε μία από τις 3 παραπάνω αναθέσεις της μεταβλητής x σε συνδυασμό με την ανάθεση { y/Helen } καταλήγουμε στα εξής συμπεράσματα:

- $Likes(Peter, Helen)$
- $Likes(Tim, Helen)$
- $Likes(John, Helen)$

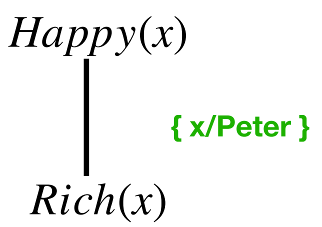
Επίσης, το αν ένα άτομο αρέσει σε κάποιο άλλο μπορεί να προκύψει και από τις 3 ακόλουθες προτάσεις:
- $Man(x) \wedge Likes(x, Catherine) \implies Likes(Catherine, x)$
- $Man(x) \wedge Kind(x) \wedge Rich(x) \implies Likes(Helen, x)$
- $Man(x) \wedge Muscly(x) \wedge Pretty(x) \implies Likes(Helen, x)$

Εφαρμόζοντας όμως backward chaining, δεν καταλήγουμε σε κάποιο συμπέρασμα για καμία ανάθεση.
Άρα τελικά, η "απάντηση" στο ερώτημα ποιος αρέσει σε ποιόν είναι:
- Η Ελένη αρέσει στον Πέτρο.
- Η Ελένη αρέσει στον Τίμο.
- Η Ελένη αρέσει στον Γιάννη.

Ερώτημα β

- Ποιός είναι ευτυχισμένος.

Θα χρησιμοποιήσω Backward chaining.
Από τον τύπο $Rich(x) \implies Happy(x)$ έχουμε:



Αφού από την ΚΒ γνωρίζουμε ότι Rich(Peter).

Ομοίως, από την ΚΒ γνωρίζουμε ότι Rich(John), άρα ο παραπάνω τύπος ικανοποιείται και για την ανάθεση { x/John }.

Άρα συμπεραίνουμε ότι:

- $Happy(Peter)$
- $Happy(John)$

Επίσης, το αν κάποιος είναι ευτυχισμένος να προκύψει και από τις 2 ακόλουθες προτάσεις:
- $Man(x) \wedge Woman(y) \wedge Likes(x, y) \wedge Likes(y, x) \implies Happy(x)$
- $Man(x) \wedge Woman(y) \wedge Likes(x, y) \wedge Likes(y, x) \implies Happy(y)$

Εφαρμόζοντας όμως backward chaining, δεν καταλήγουμε σε κάποιο συμπέρασμα για καμία ανάθεση.
Άρα τελικά, η “απάντηση” στο ερώτημα ποιος είναι ευτυχισμένος είναι:
- Ο Πέτρος
- Ο Γιάννης

<u>Επαλήθευση των αποτελεσμάτων με Prolog</u>

Έτσι ώστε να επαληθεύσω τα αποτελέσματα που προέκυψαν παραπάνω θα χρησιμοποιήσω την Prolog.

Αρχικά πρέπει να μεταφράσουμε την ΚΒ σε γλώσσα prolog.
Η ΚΒ (δηλαδή το πρόγραμμα prolog) βρίσκεται στο αρχείο **askisi5.pl** το οποίο περιέχεται στον παραδοτέο φάκελο.
Έτσι, “τρέχουμε” το πρόγραμμα και εκτελούμε τις αντίστοιχες εντολές έτσι ώστε να πάρουμε απάντηση για τα δύο ερωτήματα της άσκησης.

Το ερώτημα ποιος αγαπάει ποιόν, αντιστοιχεί στον παρακάτω τύπο:
- bagof([A,B],likes(A,B),List).

Το αποτέλεσμα είναι μια λίστα από “σετ” δύο ατόμων. Κάθε σετ της μορφής [Άτομο1,Άτομο2] αντιστοιχεί στην πληροφορία ότι στο Άτομο1 αρέσει το Άτομο2.

Το ερώτημα ποιος είναι ευτυχισμένος, αντιστοιχεί στον παρακάτω τύπο:
- bagof(A,happy(A),List).

Το αποτέλεσμα είναι μία λίστα από άτομα τα οποία είναι ευτυχισμένα.

Εκτελώντας τα παραπάνω, τα αποτελέσματα που παίρνουμε είναι:

```
?- bagof([A,B],likes(A,B),List).
List = [[tim, helen], [peter, helen], [john, helen]].

?- bagof(A,happy(A),List).
List = [peter, john].
```

Δηλαδή για το πρώτο ερώτημα παίρνουμε ότι:
- Στον Τίμο αρέσει η Ελένη
- Στον Πέτρο αρέσει η Ελένη
- Στον Γιάννη αρέσει η Ελένη

Και για το δεύτερο ερώτημα παίρνουμε ότι:
- Ο Πέτρος είναι ευτυχισμένος
- Ο Γιάννης είναι ευτυχισμένος

Άρα, τα αποτελέσματα ταυτίζονται με αυτά που βρήκαμε παραπάνω με την εφαρμογή του backward chaining.

Άσκηση 6

- **Για την άσκηση 3:**

Αρχικά πρέπει να προσθέσουμε όλους τους τύπους της KB στα Assumptions του Prover9 έτσι ώστε να δημιουργήσουμε την αντίστοιχη βάση γνώσης. Βάζω τους τύπους σε CNF μορφή καθώς τους έχουμε ήδη μετατρέψει στην άσκηση 3.
Έτσι τα Assumptions που προκύπτουν είναι:

```
MemberCorona(Kuriakos).
MemberCorona(Alexis).
MemberCorona(Fofi).
(-MemberCorona(x)) | Right(x) | Liberal(x) .
(-Right(x)) | (-Likes(x,Socialism)).
Likes(x,Socialism) | (-Liberal(x)).
(-Likes(Kuriakos,x)) | (-Likes(Alexis,x)).
Likes(Alexis,x) | Likes(Kuriakos,x).
Likes(Alexis,Capitalism).
Likes(Alexis,Socialism).
```

- **Για το ερώτημα (b)** / $KB \vDash \phi$

Προσθέτουμε την άρνηση της πρότασης φ στα Assumptions, και ομοίως με την άσκηση 3 θέλουμε να καταλήξουμε σε μία αντίφαση. Δηλαδή προσθέτουμε στα Assumptions τον τύπο:

```
(-MemberCorona(y)) | (-Liberal(y)) | (Right(y))
```

Διαφορετικά μπορούμε να προσθέσουμε την φ στα Goals. Και στις δύο περιπτώσεις το αποτέλεσμα είναι ακριβώς το ίδιο.
Το αποτέλεσμα που παίρνουμε είναι:

```
============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.00 (+ 0.01) seconds.
% Length of proof is 11.
% Level of proof is 4.
% Maximum clause weight is 3.
% Given clauses 0.

1 -MemberCorona(x) | Right(x) | Liberal(x).  [assumption].
3 MemberCorona(Alexis).  [assumption].
5 -MemberCorona(x) | -Liberal(x) | Right(x).  [assumption].
7 -Right(x) | -Likes(x,Socialism).  [assumption].
8 Right(Alexis) | Liberal(Alexis).  [resolve(1,a,3,a)].
11 -Liberal(Alexis) | Right(Alexis).  [resolve(5,a,3,a)].
15 Liberal(Alexis) | -Likes(Alexis,Socialism).  [resolve(8,a,7,a)].
18 -Liberal(Alexis) | -Likes(Alexis,Socialism).  [resolve(11,b,7,a)].
23 Likes(Alexis,Socialism).  [assumption].
26 -Likes(Alexis,Socialism) | -Likes(Alexis,Socialism).  [resolve(18,a,15,a)].
27 $F.  [copy(26),merge(b),unit_del(a,23)].

============================== end of proof ==========================
```

Δηλαδή, όπως ήταν αναμενόμενο καταλήξαμε σε μια αντίφαση. Το οποίο σημαίνει ότι η φ ικανοποιείται από την βάση γνώσης.
Παρατηρούμε ότι το αποτέλεσμα του prover9 συμπίπτει με αυτό της άσκησης 3.

- **Για το ερώτημα (c)**

Στην άρνηση της πρότασης φ που προσθέσαμε στα assumptions στην παραπάνω εκτέλεση, προσθέτουμε και το λεκτικό απάντησης έτσι ώστε να βρούμε ποιο μέλος του ΚΟΡΩΝΑ έχει την ιδιότητα της πρότασης γ.
Δηλαδή αντί της άρνησης της φ, προσθέτουμε στα assumptions την πρόταση:

```
(-MemberCorona(y)) | (-Liberal(y)) | (Right(y)) # answer(y).
```

Ομοίως, μπορούμε θα μπορούσαμε αντί αυτού να προσθέσουμε την φ μαζί με το λεκτικό απάντησης στα Goals.
Το αποτέλεσμα που παίρνουμε είναι:

```
============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.00 (+ 0.00) seconds: Alexis.
% Length of proof is 11.
% Level of proof is 4.
% Maximum clause weight is 3.
% Given clauses 0.

1 -MemberCorona(x) | Right(x) | Liberal(x).  [assumption].
3 MemberCorona(Alexis).  [assumption].
5 -MemberCorona(x) | -Liberal(x) | Right(x) # answer(x).  [assumption].
7 -Right(x) | -Likes(x,Socialism).  [assumption].
8 Right(Alexis) | Liberal(Alexis).  [resolve(1,a,3,a)].
11 -Liberal(Alexis) | Right(Alexis) # answer(Alexis).  [resolve(5,a,3,a)].
15 Liberal(Alexis) | -Likes(Alexis,Socialism).  [resolve(8,a,7,a)].
18 -Liberal(Alexis) | -Likes(Alexis,Socialism) # answer(Alexis).
[resolve(11,b,7,a)].
23 Likes(Alexis,Socialism).  [assumption].
26 -Likes(Alexis,Socialism) | -Likes(Alexis,Socialism) # answer(Alexis).
[resolve(18,a,15,a)].
27 $F # answer(Alexis).  [copy(26),merge(b),unit_del(a,23)].

============================== end of proof ==========================
```

Δηλαδή συμπεραίνουμε ότι ο Αλέξης έχει την ιδιότητα της πρότασης φ.
Επίσης, το αποτέλεσμα συμπίπτει με αυτό της άσκησης 3.

- **Για την άσκηση 4:**

Ομοίως με πριν, πρέπει να προσθέσουμε όλους τους τύπους της ΚΒ στα Assumptions του Prover9 έτσι ώστε να δημιουργήσουμε την αντίστοιχη βάση γνώσης.
Βάζω τους τύπους σε CNF μορφή καθώς τους έχουμε ήδη μετατρέψει στην άσκηση 4.
Διαφορετικά θα μπορούσαμε να προσθέσουμε της προτάσεις σε λογική πρώτης τάξης (δηλαδή όπως μας δίνονται από την εκφώνηση) και να τις μετατρέψει ο prover9 σε CNF μορφή.

Επέλεξα να προσθέσω τις προτάσεις που έχω μετατρέψει σε CNF μορφή στην άσκηση 4, έτσι ώστε να επαληθεύσω ότι η μετατροπή έχει γίνει σωστά. Έτσι, εφόσον ο prover βγάζει το σωστό αποτέλεσμα, σημαίνει ότι η μετατροπή στην άσκηση 4 έχει γίνει σωστά.

Ομοίως με προηγουμένως προσθέτω στα assumptions και την άρνηση της πρότασης C (επίσης στην CNF μορφή που υπολόγισα στην άσκηση 4). Διαφορετικά θα μπορούσα να βάλω την πρόταση C στα Goals.

Έτσι τα Assumptions που προκύπτουν είναι:

```
-In(x,y) | -In(x,z) | In(x,Intersection(y,z)).  % protasi A
-In(x,Intersection(y,z)) | In(x,y).           % protasi A
-In(x,Intersection(y,z)) | In(x,z).           % protasi A
In(f1(x,y),x) | SubsetOf(x,y).                % protasi B
-In(f1(x,y),y) | SubsetOf(x,y).               % protasi B

-SubsetOf(Intersection(c1,c2),c1).            % arnisi protasis C
```

<u>Παρατηρήσεις:</u>
Η f1 αντιστοιχεί στην συνάρτηση Skolem η οποία προέκυψε κατά την μετατροπή σε CNF στην άσκηση 4.
Τα ονόματα των μεταβλητών s και t έχουν αλλάξει σε y και z αντίστοιχα σε σχέση με την άσκηση 4.
Οι c1 και c2 είναι αντιστοιχούν στις σταθερές Skolem οι οποίες είχαν προκύψει και στην άσκηση 4.
Κατά τα άλλα οι προτάσεις είναι ακριβώς ίδιες.

Το αποτέλεσμα που προκύπτει είναι:

```
============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.00 (+ 0.01) seconds.
% Length of proof is 8.
% Level of proof is 3.
% Maximum clause weight is 9.
% Given clauses 4.

1 -SubsetOf(Intersection(c1,c2),c1).  [assumption].
2 In(f1(x,y),x) | SubsetOf(x,y).  [assumption].
3 -In(f1(x,y),y) | SubsetOf(x,y).  [assumption].
5 -In(x,Intersection(y,z)) | In(x,y).  [assumption].
7 In(f1(Intersection(c1,c2),c1),Intersection(c1,c2)).  [resolve(1,a,2,b)].
8 -In(f1(Intersection(c1,c2),c1),c1).  [resolve(1,a,3,b)].
10 In(f1(Intersection(c1,c2),c1),c1).  [hyper(5,a,7,a)].
11 $F.  [resolve(10,a,8,a)].

============================== end of proof ==========================
```

Πάλι καταλήξαμε σε κενή φράση, το οποίο σημαίνει ότι η πρόταση C έπεται λογικά από την KB (δηλαδή από τις προτάσεις A και B).

<u>Σημαντική παρατήρηση:</u>

Όλες οι προτάσεις που προσθέτουμε στα Assumptions σε κάθε ένα από τα ερωτήματα υπάρχουν στο αρχείο **askisi6.txt** το οποίο περιέχεται στο παραδοτέο directory, έτσι ώστε να μπορούν εύκολα να γίνουν copy/paste και να δοκιμαστούν στον prover9.
Για την άσκηση 3, προσθέτουμε μία από τις 2 προτάσεις/goals στα assumptions ανάλογα με το αν θέλουμε να εκτελέσουμε το ερώτημα 3β ή το ερώτημα 3γ.

Άσκηση 7

Ερώτημα α

Για να μοντελοποιήσουμε κατάλληλα την οντολογία του σχήματος, αρχικά θα χρησιμοποιήσουμε τα δύο δυαδικά σύμβολα κατηγορήματος subClassOf και belongsTo , όπου:

- subClassOf(x,y) : δηλώνει ότι το αντικείμενο/κλάση x είναι υποκλάση της y. Δηλαδή αναφέρεται σε όλες τις πλειάδες της μορφής <x,y> όπου x είναι υποκλάση της y.
- belongsTo(x,y) : δηλώνει ότι μια διοικητική υποδιαίρεση x ανήκει σε μία y. Δηλαδή αναφέρεται σε όλες τις πλειάδες της μορφής <x,y> όπου x ανήκει στην y.

Αρχικά χρησιμοποιούμε τα σύμβολα αυτά έτσι ώστε να αναπαραστήσουμε τις πληροφορίες που “παίρνουμε” από το σχήμα της εκφώνησης και χρησιμοποιούμε σύμβολα σταθερών για να αναπαραστήσουμε την κάθε κλάση (π.χ. AdministrativeUnit).
Από τα μπλε βελάκια του σχήματος έχουμε:
- subClassOf(Country,AdministrativeUnit).
- subClassOf(DecentralizedAdministration,AdministrativeUnit).
- subClassOf(Region,AdministrativeUnit).
- subClassOf(RegionalUnit,AdministrativeUnit).
- subClassOf(Municipality,AdministrativeUnit).
- subClassOf(MunicipalityUnit,AdministrativeUnit).
- subClassOf(MunicipalCommunity,AdministrativeUnit).
- subClassOf(LocalCommunity,AdministrativeUnit).

Από τα κόκκινα βελάκια του σχήματος έχουμε:
- belongsTo(DecentralizedAdministration,Country).
- belongsTo(Region,DecentralizedAdministration).
- belongsTo(RegionalUnit,Region).
- belongsTo(Municipality,RegionalUnit).
- belongsTo(MunicipalityUnit,Municipality).
- belongsTo(MunicipalCommunity,MunicipalityUnit).
- belongsTo(LocalCommunity,MunicipalityUnit).

Επίσης, έχουμε τις εξής ιδιότητες:
Αν μια κλάση x είναι υποκλάση της y και η κλάση y είναι υποκλάση της z τότε η x είναι υποκλάση της z.
Το οποίο “μεταφράζεται” σε λογική πρώτης τάξης ως εξής:

$$(\forall x)(\forall y)(\forall z)(subClassOf(x,y) \land subClassOf(y,z) \implies subClassOf(x,z))$$

Ομοίως αν μια διοικητική υποδιαίρεση x είναι ανήκει σε μία y και η y ανήκει σε μία z τότε η x ανήκει και στην z.
Το οποίο “μεταφράζεται” σε λογική πρώτης τάξης ως εξής:

$$(\forall x)(\forall y)(\forall z)(belongsTo(x,y) \land belongsTo(y,z) \implies belongsTo(x,z))$$

Ανάλογα με τις προτάσεις που θα θελήσουμε να αποδείξουμε ενδεχομένως να χρειαστεί να προστεθούν και προτάσεις όπως:
Μια κλάση δεν είναι υποκλάση του εαυτού της. Δηλαδή: $(\forall x)(\neg subClassOf(x,x))$
Μια διοικητική υποδιαίρεση δεν είναι ανήκει στον εαυτό της. Δηλαδή: $(\forall x)(\neg belongsTo(x,x))$
Στην περίπτωση μας, δεν μας χρειάζονται οι προτάσεις αυτές επομένως μπορούμε να τις παραλείψουμε. Είτε τις προσθέσουμε είτε όχι το αποτέλεσμα θα είναι ακριβώς το ίδιο

Ερώτημα β

Θα προσθέσουμε στην οντολογία την κλάση Class. Η κλάση αυτή θα έχει ως στοιχεία όλες τις άλλες κλάσεις. Επομένως θα χρειαστούμε ένα δυαδικό σύμβολο κατηγορήματος το οποίο να αναπαριστά τα στοιχεία μιας κλάσης και το σύμβολο σταθεράς Class. Το σύμβολο αυτό θα το ονομάσω elementOf. Πιο συγκεκριμένα:
- elementOf(x,y) : δηλώνει ότι το αντικείμενο x είναι στοιχείο της κλάσης y. Δηλαδή αναφέρεται σε όλες τις πλειάδες της μορφής <x,y> όπου x είναι αντικείμενο της y.

Επομένως προσθέτουμε στην οντολογία τους παρακάτω τύπους:

- elementOf(AdministrativeUnit,Class).
- elementOf(Country,Class).
- elementOf(DecentralizedAdministration,Class).
- elementOf(Region,Class).
- elementOf(RegionalUnit,Class).
- elementOf(Municipality,Class).
- elementOf(MunicipalityUnit,Class).
- elementOf(MunicipalCommunity,Class).
- elementOf(LocalCommunity,Class).

Επίσης, χρειάζεται να προσθέσουμε στην οντολογία την εξής ιδιότητα:
Αν μια κλάση x είναι υποκλάση μιας κλάσης y και ένα αντικείμενο z είναι στοιχείο της κλάσης x τότε το αντικείμενο z είναι στοιχείο και της κλάσης y. Το οποίο μεταφράζεται σε λογική πρώτης τάξης ως εξής:

$$(\forall x)(\forall y)(\forall z)(elementOf(x,Class) \wedge elementOf(y,Class) \wedge subClassOf(x,y) \wedge elementOf(z,x) \implies elementOf(z,y))$$

Ερώτημα γ

Θα προσθέσουμε στην οντολογία το αντικείμενο MunicipalityOfAthens το οποίο είναι στοιχείο της κλάσης Municipality. Για να αναπαραστήσουμε την πληροφορία αυτή θα χρησιμοποιήσουμε το σύμβολο κατηγορήματος elementOf το οποίο ορίστηκε στο προηγούμενο ερώτημα και το σύμβολο σταθεράς MunicipalityOfAthens για να αναπαραστήσουμε το αντικείμενο. Δηλαδή προκύπτει ο τύπος:

- elementOf(MunicipalityOfAthens,Municipality).

Άρα τελικά η βάση γνώσης που προκύπτει είναι (την οποία βάζουμε στα Assumptions του Prover9):

```
subClassOf(Country,AdministrativeUnit).
subClassOf(DecentralizedAdministration,AdministrativeUnit).
subClassOf(Region,AdministrativeUnit).
subClassOf(RegionalUnit,AdministrativeUnit).
subClassOf(Municipality,AdministrativeUnit).
subClassOf(MunicipalityUnit,AdministrativeUnit).
subClassOf(MunicipalCommunity,AdministrativeUnit).
subClassOf(LocalCommunity,AdministrativeUnit).
belongsTo(DecentralizedAdministration,Country).
belongsTo(Region,DecentralizedAdministration).
belongsTo(RegionalUnit,Region).
belongsTo(Municipality,RegionalUnit).
belongsTo(MunicipalityUnit,Municipality).
belongsTo(MunicipalCommunity,MunicipalityUnit).
belongsTo(LocalCommunity,MunicipalityUnit).
elementOf(AdministrativeUnit,Class).
elementOf(Country,Class).
elementOf(DecentralizedAdministration,Class).
elementOf(Region,Class).
elementOf(RegionalUnit,Class).
elementOf(Municipality,Class).
elementOf(MunicipalityUnit,Class).
elementOf(MunicipalCommunity,Class).
elementOf(LocalCommunity,Class).
all x all y all z (subClassOf(x,y) & subClassOf(y,z)-> subClassOf(x,z)).
all x all y all z (belongsTo(x,y) & belongsTo(y,z)-> belongsTo(x,z)).
all x all y all z (elementOf(x,Class) & elementOf(y,Class) & elementOf(z,x) & subClassOf(x,y) -> elementOf(z,y)).
elementOf(MunicipalityOfAthens,Municipality).
```

Για να αποδείξουμε ότι το αντικείμενο MunicipalityOfAthens είναι επίσης στοιχείο της κλάσης AdministrativeUnit προσθέτουμε στο Goal του Prover9 τον τύπο:

```
elementOf(MunicipalityOfAthens,AdministrativeUnit).
```

Το αποτέλεσμα που προκύπτει είναι:

```
============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.01 (+ 0.01) seconds.
% Length of proof is 10.
% Level of proof is 3.
% Maximum clause weight is 15.
% Given clauses 28.

3 (all x all y all z (elementOf(x,Class) & elementOf(y,Class) & elementOf(z,x) &
subClassOf(x,y) -> elementOf(z,y))) # label(non_clause).  [assumption].
4 elementOf(MunicipalityOfAthens,AdministrativeUnit) # label(non_clause) # label(goal).
[goal].
9 subClassOf(Municipality,AdministrativeUnit).  [assumption].
20 elementOf(AdministrativeUnit,Class).  [assumption].
25 elementOf(Municipality,Class).  [assumption].
29 elementOf(MunicipalityOfAthens,Municipality).  [assumption].
32 -elementOf(x,Class) | -elementOf(y,Class) | -elementOf(z,x) | -subClassOf(x,y) |
elementOf(z,y).  [clausify(3)].
33 -elementOf(MunicipalityOfAthens,AdministrativeUnit).  [deny(4)].
40 elementOf(MunicipalityOfAthens,AdministrativeUnit).  [ur(32,a,25,a,b,20,a,c,29,a,d,9,a)].
41 $F.  [resolve(40,a,33,a)].

============================== end of proof ==========================
```

Επομένως η πρόταση ικανοποιείται.

<u>Σημαντική παρατήρηση:</u>
Όλες οι προτάσεις που προσθέτουμε στα Assumptions (δηλαδή η βάση γνώσης) υπάρχει στο αρχείο **askisi7.txt** το οποίο περιέχεται στο παραδοτέο directory, έτσι ώστε να μπορούν εύκολα να γίνουν copy/paste και να δοκιμαστούν στον prover9.
Προσθέτοντας τα Assumptions και το Goal από το αρχείο txt στον prover9 παίρνουμε το παραπάνω αποτέλεσμα.

Άσκηση 8

Ερώτημα β

Για να μοντελοποιήσουμε σε λογική πρώτης τάξης την βάση δεδομένων που μας δόθηκε χρειαζόμαστε δύο σύμβολα κατηγορήματος.
Το <u>μοναδιαίο σύμβολο κατηγορήματος</u> Person(x) το οποίο αντιστοιχεί στον πρώτο πίνακα της βάσης και αναφέρεται στα αντικείμενα τα οποία είναι άτομα και περιέχονται σε αυτόν.
Το <u>δυαδικό σύμβολο κατηγορήματος</u> Loves(x,y) το οποίο αντιστοιχεί στον δεύτερο πίνακα της βάσης και αναφέρεται σε όλες τις πλειάδες της μορφής <x,y> όπου το αντικείμενο x αγαπά το αντικείμενο y.

Έτσι η βάση γνώσης που προκύπτει από την βάση δεδομένων χρησιμοποιώντας τα δύο παραπάνω σύμβολα κατηγορήματος περιέχει τους εξής τύπους:

```
Person(Donald).
Person(Melania).
Person(Ivanka).
Person(Barron).
Loves(Donald,Donald).
Loves(Donald,Ivanka).
Loves(Ivanka,Donald).
Loves(Melania,Barron).
Loves(Barron,Melania).
```

Ερώτημα γ

Για να μοντελοποιήσουμε κατάλληλα την βάση γνώσης και να χρησιμοποιήσουμε την prover, χρειάζεται πρώτα να προστεθούν στην βάση γνώσης κάποιοι επιπλέον τύποι (non-monotonic reasoning).
Αρχικά προσθέτουμε το domain closure axiom. Δηλαδή προσθέτουμε την βάση γνώσης τον τύπο:

```
all x (x=Donald | x=Melania | x=Ivanka | x=Barron ).
```

Επίσης εφαρμόζουμε unique names assumption (UNA). Δηλαδή προσθέτουμε την βάση γνώσης τους τύπους:

```
Donald!=Ivanka.
Donald!=Melania.
Donald!=Barron.
Melania!=Ivanka.
Melania!=Barron.
Barron!=Ivanka.
```

Τέλος, πρέπει να εφαρμόσουμε το predicate completion. Για να εφαρμοστεί, μπορούμε να αντικαταστήσουμε τους τύπους που περιέχουν το κατηγόρημα Loves που προσθέσαμε στην βάση γνώσης στο προηγούμενο ερώτημα με τους εξής τύπους (ο οποίοι αναφέρονται στην πληροφορία για το ποιος αγαπάει ποιόν αλλά και στο ποιος δεν αγαπάει ποιον):

```
all x ( (x=Donald | x=Ivanka) <->  Loves(x,Donald)).
all x ( (x=Donald) <->  Loves(x,Ivanka)).
all x ( (x=Melania) <->  Loves(x,Barron)).
all x ( (x=Barron) <->  Loves(x,Melania)).
```

Όμως, αντί αυτού θα κρατήσω της προτάσεις του κατηγορήματος Love που πρόσθεσα στην βάση γνώσης προηγουμένως και θα προσθέσω τις εξής επιπλέον προτάσεις οι οποίες αναπαριστούν την πληροφορία για το ποιος δεν αγαπάει ποιον:

```
all x ( (x!=Donald & x!=Ivanka) ->  -Loves(x,Donald)).
all x ( (x!=Donald) ->  -Loves(x,Ivanka)).
all x ( (x!=Melania) ->  -Loves(x,Barron)).
all x ( (x!=Barron) ->  -Loves(x,Melania)).
```

Έτσι συνολικά έχουμε πληροφορία και για το ποιος αγαπάει ποιον αλλά και για το ποιος δεν αγαπάει ποιον.

Παρατήρηση: Η επιλογή αυτή αφορά το ερώτημα δ και εξηγείται αναλυτικά παρακάτω. Όσον αφορά το ερώτημα γ μπορούμε να επιλέξουμε οποιονδήποτε από τους δύο τρόπους για να εφαρμόσουμε το predicate completion και τα αποτελέσματα θα είναι ακριβώς τα ίδια σε όλες τις προτάσεις.

Παρατήρηση: τα equality axioms δεν χρειάζεται να προστεθούν στον prover9.

Άρα τελικά η βάση γνώσης που προκύπτει είναι η εξής:

```
Person(Donald).
Person(Melania).
Person(Ivanka).
Person(Barron).
Loves(Donald,Donald).
Loves(Donald,Ivanka).
Loves(Ivanka,Donald).
Loves(Melania,Barron).
Loves(Barron,Melania).
Donald!=Ivanka.
Donald!=Melania.
Donald!=Barron.
Melania!=Ivanka.
Melania!=Barron.
Barron!=Ivanka.
```

```
all x (x=Donald | x=Melania | x=Ivanka | x=Barron ).
all x ( (x!=Donald & x!=Ivanka) ->  -Loves(x,Donald)).
all x ( (x!=Donald) ->  -Loves(x,Ivanka)).
all x ( (x!=Melania) ->  -Loves(x,Barron)).
all x ( (x!=Barron) ->  -Loves(x,Melania)).
```

Τέλος, το μόνο που μένει είναι να μετατρέψουμε τις προτάσεις προς απόδειξη σε λογική πρώτης τάξης.

i. Υπάρχουν δύο άνθρωποι που ο ένας αγαπάει τον άλλον

```
exists x exists y (Person(x) & Person(y) & Loves(x,y) & Loves(y,x)).
```

ii. Υπάρχουν δύο άνθρωποι διαφορετικοί μεταξύ τους που ο ένας αγαπάει τον άλλον

```
exists x exists y (x!=y & Person(x) & Person(y) & Loves(x,y) & Loves(y,x)).
```

iii. Η Melania δεν αγαπάει τον Donald.

```
-Loves(Melania,Donald).
```

iv. Υπάρχει κάποιος άνθρωπος που δεν αγαπάει τον Donald.

```
exists x (Person(x) & -Loves(x,Donald)).
```

v. Για κάθε άνθρωπο, υπάρχει κάποιος άνθρωπος διαφορετικός από αυτόν που τον αγαπάει.

```
all x (Person(x) & exists y (Person(y) & x!=y & Loves(y,x))).
```

vi. Για κάθε άνθρωπο, υπάρχει κάποιος άνθρωπος διαφορετικός από αυτόν που δεν τον αγαπάει.

```
all x (Person(x) & exists y (Person(y) & x!=y & -Loves(y,x))).
```

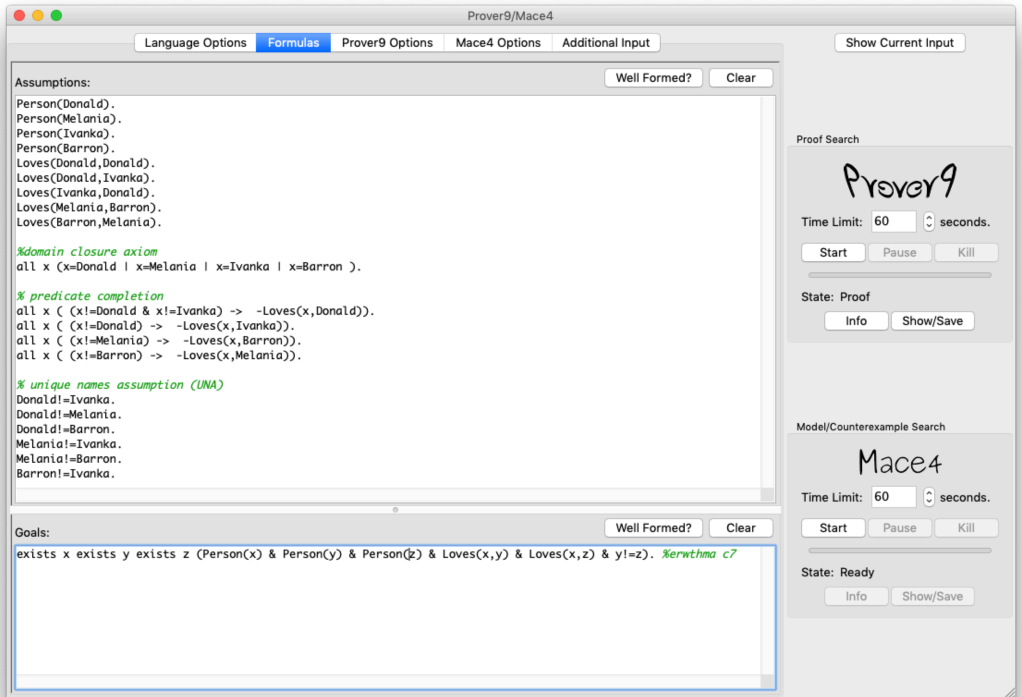
vii. Υπάρχει κάποιος άνθρωπος που αγαπάει δύο ανθρώπους διαφορετικούς μεταξύ τους.

```
exists x exists y exists z (Person(x) & Person(y) & Person(z) & Loves(x,y) & Loves(x,z) & y!=z).
```

Για να αποδείξουμε μια από αυτές τις προτάσεις, προσθέτουμε στα Assumptions της βάση γνώσης που αναφέρθηκε σε αυτό το ερώτημα καθώς και τον αντίστοιχο τύπο της πρότασης που θέλουμε να αποδείξουμε στα Goals.

Παρατήρηση: Όλα τα Assumptions και τα Goals υπάρχουν στο αρχείο με όνομα **askisi8.txt** έτσι ώστε να μπορούν εύκολα να δοκιμαστούν στον prover. Από το αρχείο αυτό χρειάζεται να αντιγραφούν όλα τα Assumption στην αντίστοιχη "θέση" του prover9 και κάθε φορά η πρόταση (από τα Goals του αρχείου txt) που θέλουμε να αποδείξουμε αυτούσια στην "θέση" Goals του prover9.
ΠΡΟΣΟΧΗ: στα Assumptions του αρχείου υπάρχει και η πρόταση που αφορά το επόμενο ερώτημα. Για την απόδειξη των ερωτημάτων του ερωτήματος γ **ΔΕΝ** πρέπει να προστεθεί η πρόταση αυτή στα Assumptions

Παράδειγμα εκτέλεσης (για την πρόταση vii):



Αποτέλεσμα:

```
============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.01 (+ 0.01) seconds.
% Length of proof is 16.
% Level of proof is 4.
% Maximum clause weight is 15.
% Given clauses 29.

2 (all x (x = Donald | x = Ivanka <-> Loves(x,Donald))) # label(non_clause).  [assumption].
3 (all x (x = Donald <-> Loves(x,Ivanka))) # label(non_clause).  [assumption].
6 (exists x exists y exists z (Person(x) & Person(y) & Person(z) & Loves(x,y) & Loves(x,z) & y != z)) # label(non_clause) # label(goal)
[goal].
7 Person(Donald).  [assumption].
9 Person(Ivanka).  [assumption].
13 x != Donald | Loves(x,Donald).  [clausify(2)].
14 Donald != x | Loves(x,Donald).  [copy(13),flip(a)].
19 x != Donald | Loves(x,Ivanka).  [clausify(3)].
20 Donald != x | Loves(x,Ivanka).  [copy(19),flip(a)].
31 Donald != Ivanka.  [assumption].
32 Ivanka != Donald.  [copy(31),flip(a)].
41 -Person(x) | -Person(y) | -Person(z) | -Loves(x,y) | -Loves(x,z) | z = y.  [deny(6)].
43 -Person(x) | -Person(y) | -Loves(x,y) | -Loves(x,x) | x = y.  [factor(41,a,c)].
46 Loves(Donald,Donald).  [xx_res(14,a)].
48 Loves(Donald,Ivanka).  [xx_res(20,a)].
68 $F.  [ur(43,a,7,a,b,9,a,d,46,a,e,32,a(flip)),unit_del(a,48)].

============================== end of proof ==========================
```

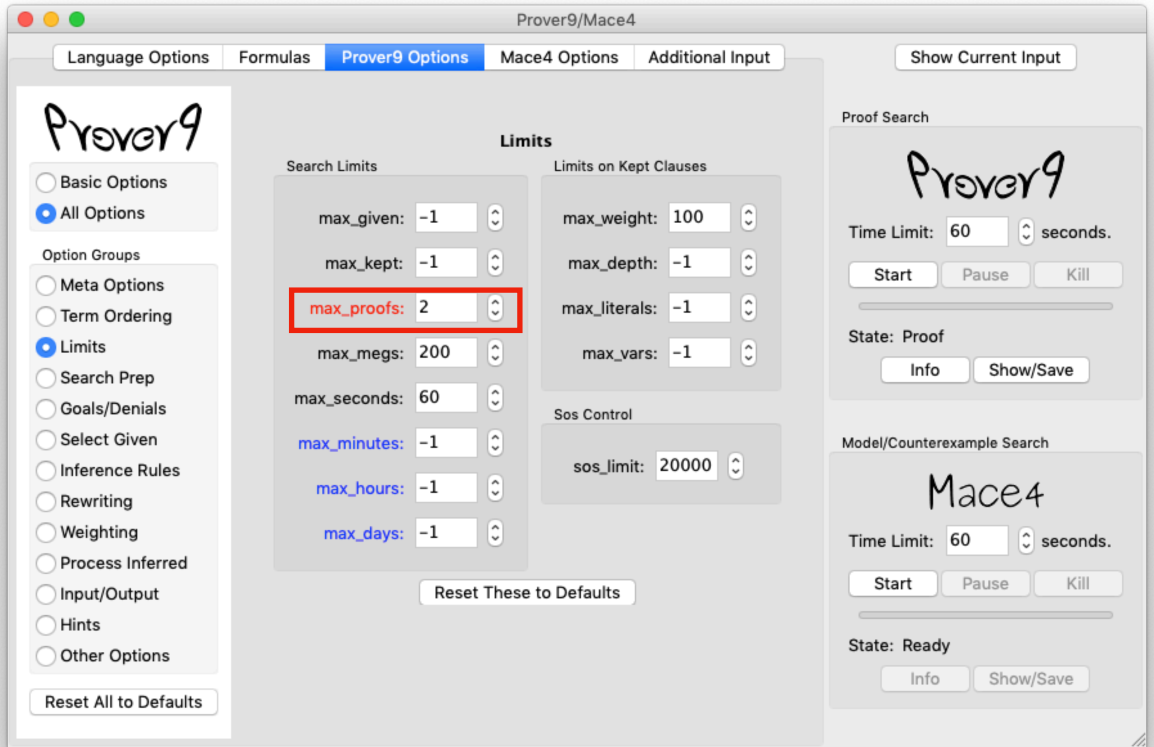
Ομοίως μπορούν να εκτελεστούν και οι υπόλοιπες προτάσεις αλλάζοντας τον τύπο στο Goal με τον αντίστοιχο τύπο του ερωτήματος που θέλουμε να αποδείξουμε (και υπάρχει στο αρχείο txt) και να πάρουμε τις αντίστοιχες αποδείξεις.

Ερώτημα δ

Για την απάντηση στην ερώτηση Ποιους αγαπάει ο Donald πρέπει να χρησιμοποιήσουμε λεκτικό απάντησης. Έτσι θα προσθέσω στα Assumptions την άρνηση της πρότασης αυτής σε CNF μορφή προσθέτοντας το λεκτικό απάντησης.
Η πρόταση που προκύπτει είναι:

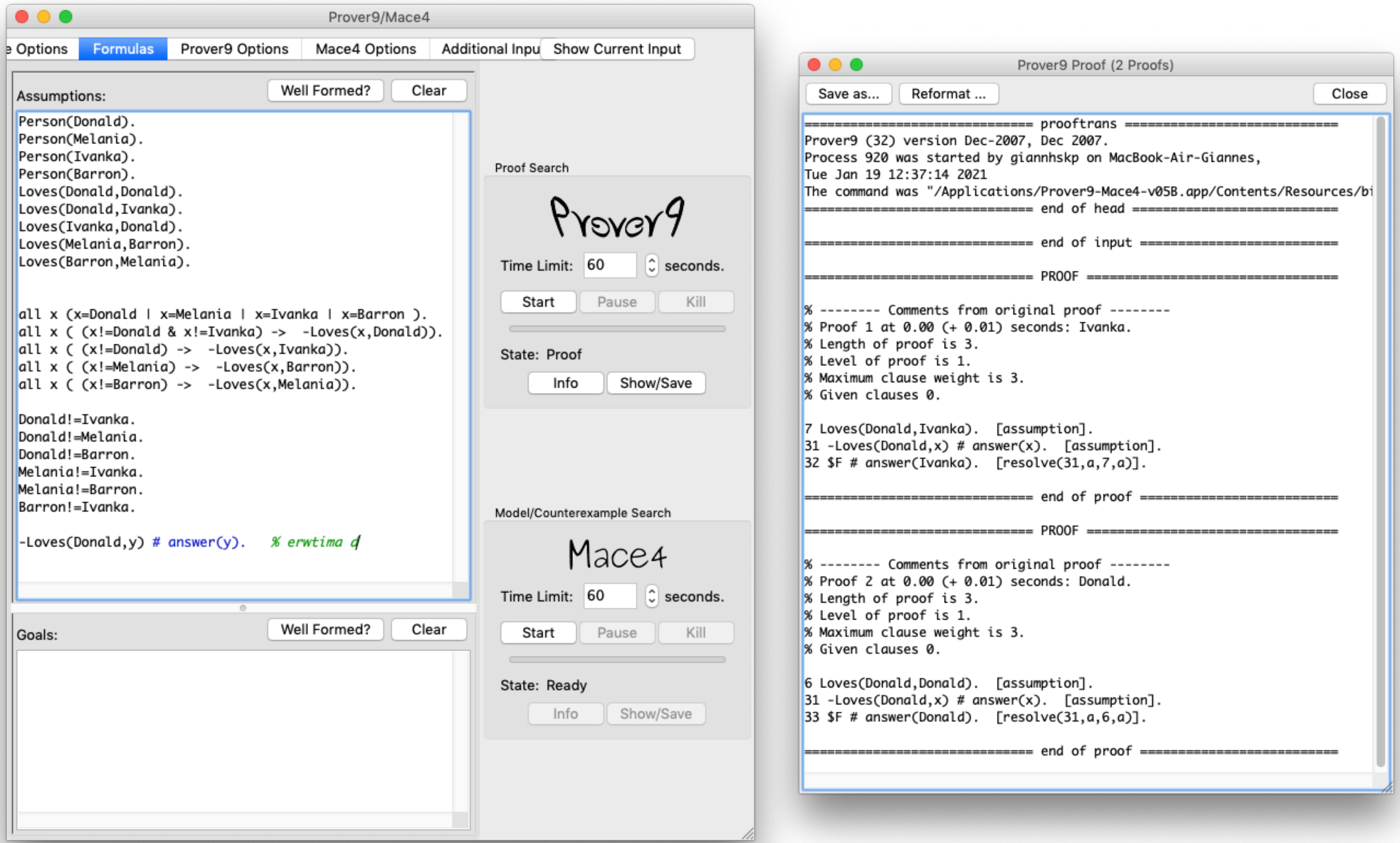
-Loves(Donald,y) # answer(y).

Επίσης, καθώς θέλουμε παραπάνω από ένα αποτελέσματα, πρέπει να αλλάξουμε τα setting του prover9. Έτσι, κάνουμε την εξής αλλαγή:



Δηλαδή, αλλάζουμε το **max_proofs** από 1 σε 2.
Θεωρητικά αν δεν γνωρίζουμε το πόσες λύσεις υπάρχουν αυξάνουμε το max_proofs ακόμα παραπάνω έτσι ώστε να βρεθούν όλες οι πιθανές λύσεις.
Στην περίπτωση μας, μπορούμε "με το μάτι" να δούμε από τα δεδομένα μας ότι υπάρχουν μόνο 2 απαντήσεις επομένως μπορούμε να θέσουμε το max_proofs σε 2 έτσι ώστε να μην ψάχνει "άδικα" για τρίτη διαφορετική λύση η οποία δεν υπάρχει.
Βέβαια το αποτέλεσμα θα είναι ακριβώς το ίδιο όσο και αν αυξήσουμε το max_proofs με την διαφορά ότι θα εμφανιστεί μήνυμα λάθους για το γεγονός ότι δεν βρέθηκαν όλες οι λύσεις.
Για παράδειγμα αν θέσουμε max_proofs=3 στο παράδειγμα μας θα βρεθούν οι δύο λύσεις και θα εκτυπωθεί μήνυμα λάθους ότι δεν βρέθηκε η 3η λύση.

Τελικά, προσθέτοντας την παραπάνω πρόταση στα Assumptions και αλλάζοντας το max_proofs έχουμε:



Δηλαδή, οι 2 απαντήσεις είναι **Ivanka** και **Donald**.

<u>Παρατήρηση</u>: Η πρόταση που προσθέσαμε στα Assumptions υπάρχει επίσης στο αρχείο **askisi8.txt** μαζί με τις προτάσεις του προηγούμενου ερωτήματος. Για την εκτέλεση του ερωτήματος αυτού πρέπει να προσθέσουμε μόνο τις προτάσεις που αφορούν το ερώτημα αυτό και όχι προτάσεις/goals του προηγούμενου ερωτήματος που επίσης υπάρχουν στο txt αρχείο. Δηλαδή τα Assumptions πρέπει να είναι όπως το παραπάνω screenshot.

<u>Παρατήρηση 2</u>: Στο ερώτημα γ αναφέρθηκε ένας διαφορετικός τρόπος για να εφαρμόσουμε το predicate completion από αυτόν που χρησιμοποίησα στην πράξη. Ο τρόπος αυτός δουλεύει κανονικά και ενδεχομένως να είναι και κομψότερος από αυτόν που τελικά χρησιμοποίησα. Όμως συνάντησα το εξής πρόβλημα με τον prover9 και την εφαρμογή του ερωτήματος δ. Ο prover έβρισκε κανονικά τις δύο διαφορετικές λύσεις για το ερώτημα αυτό χρησιμοποιώντας ανάλυση. Ωστόσο, το λεκτικό απάντησης δεν περιείχε την πραγματική απάντηση αλλά περιείχε μια άγνωστη σε εμένα "πληροφορία" το: `v100`. Δηλαδή η απάντηση ήταν: # `answer(v100)`.
Από ότι κατάλαβα πρέπει να είναι κάποιου είδους bug το οποίο αφορά την τεχνική που χρησιμοποιούσε για την τελευταία "απλοποίηση" έτσι ώστε να καταλήξει στο κενό σύνολο, καθώς δεν χρησιμοποιούσε τον κλασικό κανόνα του resolution αλλά κάποια "επέκταση" του κανόνα η οποία ενδεχομένως να απέτρεπε τον prover από το να εντοπίσει το λεκτικό απάντησης.

<u>Έτσι, κατέληξα να χρησιμοποιώ τις προτάσεις που περιγράφηκαν στο ερώτημα γ έτσι ώστε τα λεκτικά απάντησης να επιστρέφουν τις σωστές απαντήσεις, αλλά και η βάση γνώσης (Assumptions) να είναι κοινή και για τα δύο ερωτήματα γ και δ.</u>

Συνολικά, τα proofs των ερωτήσεων της άσκησης 8 είναι:

Ερώτημα c (i)

```
============================== prooftrans ============================
Prover9 (32) version Dec-2007, Dec 2007.
Process 2179 was started by giannhskp on MacBook-Air-Giannes,
Wed Jan 27 23:47:05 2021
The command was "/Applications/Prover9-Mace4-v05B.app/Contents/Resources/bin-mac-intel/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.01 (+ 0.00) seconds.
% Length of proof is 7.
% Level of proof is 2.
% Maximum clause weight is 10.
% Given clauses 21.

6 (exists x exists y (Person(x) & Person(y) & Loves(x,y) & Loves(y,x))) # label(non_clause) # label(goal).  [goal].
8 Person(Melania).  [assumption].
10 Person(Barron).  [assumption].
14 Loves(Melania,Barron).  [assumption].
15 Loves(Barron,Melania).  [assumption].
36 -Person(x) | -Person(y) | -Loves(x,y) | -Loves(y,x).  [deny(6)].
55 $F.  [resolve(36,c,15,a),unit_del(a,10),unit_del(b,8),unit_del(c,14)].

============================== end of proof ==========================
```

Ερώτημα c (ii)

```
============================== prooftrans ============================
Prover9 (32) version Dec-2007, Dec 2007.
Process 2188 was started by giannhskp on MacBook-Air-Giannes,
Wed Jan 27 23:47:39 2021
The command was "/Applications/Prover9-Mace4-v05B.app/Contents/Resources/bin-mac-intel/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.01 (+ 0.00) seconds.
% Length of proof is 8.
% Level of proof is 2.
% Maximum clause weight is 13.
% Given clauses 21.

6 (exists x exists y (x != y & Person(x) & Person(y) & Loves(x,y) & Loves(y,x))) # label(non_clause) # label(goal).  [goal].
8 Person(Melania).  [assumption].
10 Person(Barron).  [assumption].
14 Loves(Melania,Barron).  [assumption].
15 Loves(Barron,Melania).  [assumption].
33 Melania != Barron.  [assumption].
36 x = y | -Person(y) | -Person(x) | -Loves(y,x) | -Loves(x,y).  [deny(6)].
54 $F.  [resolve(36,d,15,a),unit_del(a,33),unit_del(b,10),unit_del(c,8),unit_del(d,14)].

============================== end of proof ==========================
```

Ερώτημα c (iii)

```
============================== prooftrans ============================
Prover9 (32) version Dec-2007, Dec 2007.
Process 2198 was started by giannhskp on MacBook-Air-Giannes,
Wed Jan 27 23:49:17 2021
The command was "/Applications/Prover9-Mace4-v05B.app/Contents/Resources/bin-mac-intel/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.01 (+ 0.00) seconds.
% Length of proof is 10.
% Level of proof is 3.
% Maximum clause weight is 9.
% Given clauses 14.

2 (all x (x != Donald & x != Ivanka -> -Loves(x,Donald))) # label(non_clause).  [assumption].
6 -Loves(Melania,Donald) # label(non_clause) # label(goal).  [goal].
14 x = Donald | x = Ivanka | -Loves(x,Donald).  [clausify(2)].
15 Donald = x | Ivanka = x | -Loves(x,Donald).  [copy(14),flip(a),flip(b)].
24 Donald != Melania.  [assumption].
25 Melania != Donald.  [copy(24),flip(a)].
28 Melania != Ivanka.  [assumption].
29 Ivanka != Melania.  [copy(28),flip(a)].
34 Loves(Melania,Donald).  [deny(6)].
45 $F.  [ur(15,a,25,a(flip),b,29,a),unit_del(a,34)].

============================== end of proof ==========================
```

Ερώτημα c (iv)

```
============================== prooftrans ============================
Prover9 (32) version Dec-2007, Dec 2007.
Process 2209 was started by giannhskp on MacBook-Air-Giannes,
Wed Jan 27 23:50:48 2021
The command was "/Applications/Prover9-Mace4-v05B.app/Contents/Resources/bin-mac-intel/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.01 (+ 0.00) seconds.
% Length of proof is 11.
% Level of proof is 3.
% Maximum clause weight is 9.
% Given clauses 14.

2 (all x (x != Donald & x != Ivanka -> -Loves(x,Donald))) # label(non_clause).  [assumption].
6 (exists x (Person(x) & -Loves(x,Donald))) # label(non_clause) # label(goal).  [goal].
7 -Person(x) | Loves(x,Donald).  [deny(6)].
9 Person(Melania).  [assumption].
19 x = Donald | x = Ivanka | -Loves(x,Donald).  [clausify(2)].
20 Donald = x | Ivanka = x | -Loves(x,Donald).  [copy(19),flip(a),flip(b)].
29 Donald != Melania.  [assumption].
30 Melania != Donald.  [copy(29),flip(a)].
33 Melania != Ivanka.  [assumption].
37 Loves(Melania,Donald).  [resolve(7,a,9,a)].
50 $F.  [ur(20,a,30,a(flip),b,33,a(flip)),unit_del(a,37)].

============================== end of proof ==========================
```

Ερώτημα c (v)

```
============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.02 (+ 0.01) seconds.
% Length of proof is 30.
% Level of proof is 14.
% Maximum clause weight is 19.
% Given clauses 47.

1 (all x (x = Donald | x = Melania | x = Ivanka | x = Barron)) # label(non_clause).  [assumption].
6 (all x (Person(x) & (exists y (Person(y) & x != y & Loves(y,x))))) # label(non_clause) # label(goal).  [goal].
7 Person(Donald).  [assumption].
8 Person(Melania).  [assumption].
9 Person(Ivanka).  [assumption].
10 Person(Barron).  [assumption].
12 Loves(Donald,Ivanka).  [assumption].
13 Loves(Ivanka,Donald).  [assumption].
14 Loves(Melania,Barron).  [assumption].
15 Loves(Barron,Melania).  [assumption].
16 x = Donald | x = Melania | x = Ivanka | x = Barron.  [clausify(1)].
17 Donald = x | Melania = x | Ivanka = x | Barron = x.  [copy(16),flip(a),flip(b),flip(c),flip(d)].
26 Donald != Ivanka.  [assumption].
27 Ivanka != Donald.  [copy(26),flip(a)].
33 Melania != Barron.  [assumption].
36 -Person(c1) | -Person(x) | x = c1 | -Loves(x,c1).  [deny(6)].
37 -Person(c1) | -Person(x) | c1 = x | -Loves(x,c1).  [copy(36),flip(c)].
38 Donald = x | Ivanka = x | Barron = x | Person(x).  [para(17(b,1),8(a,1))].
55 c1 = Donald | c1 = Ivanka | c1 = Barron | -Person(c1) | -Person(x) | c1 = x | -Loves(x,Melania).  [para(17(b,2),37(d,2)),flip(a),flip(b),flip(c)].
56 c1 = Donald | c1 = Ivanka | c1 = Barron | -Person(c1).  [factor(55,c,f),unit_del(e,10),unit_del(f,15)].
58 c1 = Donald | c1 = Ivanka | c1 = Barron.  [resolve(56,d,38,d),flip(d),flip(e),flip(f),merge(d),merge(e),merge(f)].
59 c1 = Donald | c1 = Barron | -Person(c1) | -Person(x) | c1 = x | -Loves(x,Ivanka).  [para(58(b,1),37(d,2))].
60 c1 = Donald | c1 = Barron | -Person(c1).  [factor(59,a,e),unit_del(d,7),unit_del(e,12)].
61 c1 = Donald | c1 = Barron.  [para(58(b,1),60(c,1)),merge(c),merge(d),unit_del(c,9)].
62 c1 = Donald | -Person(c1) | -Person(x) | c1 = x | -Loves(x,Barron).  [para(61(b,1),37(d,2))].
64 c1 = Donald | -Person(c1) | c1 = Melania.  [resolve(62,e,14,a),unit_del(c,8)].
97 c1 = Donald | c1 = Melania.  [para(61(b,1),64(b,1)),merge(b),unit_del(b,10)].
100 c1 = Donald.  [para(97(b,1),61(b,1)),merge(b),unit_del(b,33)].
101 -Person(x) | Donald = x | -Loves(x,Donald).  [back_rewrite(37),rewrite([100(1),100(4),100(6)]),unit_del(a,7)].
102 $F.  [resolve(101,c,13,a),flip(b),unit_del(a,9),unit_del(b,27)].

============================== end of proof ==========================
```

Ερώτημα c (vi)

```
============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.02 (+ 0.01) seconds.
% Length of proof is 42.
% Level of proof is 10.
% Maximum clause weight is 17.
% Given clauses 46.

1 (all x (x = Donald | x = Melania | x = Ivanka | x = Barron)) # label(non_clause).  [assumption].
2 (all x (x != Donald & x != Ivanka -> -Loves(x,Donald))) # label(non_clause).  [assumption].
3 (all x (x != Donald -> -Loves(x,Ivanka))) # label(non_clause).  [assumption].
4 (all x (x != Melania -> -Loves(x,Barron))) # label(non_clause).  [assumption].
5 (all x (x != Barron -> -Loves(x,Melania))) # label(non_clause).  [assumption].
6 (all x (Person(x) & (exists y (Person(y) & x != y & -Loves(y,x))))) # label(non_clause) # label(goal).  [goal].
7 Person(Donald).  [assumption].
8 Person(Melania).  [assumption].
9 Person(Ivanka).  [assumption].
10 Person(Barron).  [assumption].
16 x = Donald | x = Melania | x = Ivanka | x = Barron.  [clausify(1)].
17 Donald = x | Melania = x | Ivanka = x | Barron = x.  [copy(16),flip(a),flip(b),flip(c),flip(d)].
18 x = Donald | x = Ivanka | -Loves(x,Donald).  [clausify(2)].
19 Donald = x | Ivanka = x | -Loves(x,Donald).  [copy(18),flip(a),flip(b)].
20 x = Donald | -Loves(x,Ivanka).  [clausify(3)].
21 Donald = x | -Loves(x,Ivanka).  [copy(20),flip(a)].
22 x = Melania | -Loves(x,Barron).  [clausify(4)].
23 Melania = x | -Loves(x,Barron).  [copy(22),flip(a)].
24 x = Barron | -Loves(x,Melania).  [clausify(5)].
25 Barron = x | -Loves(x,Melania).  [copy(24),flip(a)].
28 Donald != Melania.  [assumption].
29 Melania != Donald.  [copy(28),flip(a)].
30 Donald != Barron.  [assumption].
31 Barron != Donald.  [copy(30),flip(a)].
32 Melania != Ivanka.  [assumption].
36 -Person(c1) | -Person(x) | x = c1 | Loves(x,c1).  [deny(6)].
37 -Person(c1) | -Person(x) | c1 = x | Loves(x,c1).  [copy(36),flip(c)].
45 -Loves(Donald,Barron).  [ur(23,a,29,a)].
47 -Loves(Donald,Melania).  [ur(25,a,31,a)].
48 -Loves(Barron,Ivanka).  [ur(21,a,31,a(flip))].
50 -Loves(Melania,Donald).  [ur(19,a,29,a(flip),b,32,a(flip))].
55 -Person(c1) | c1 = Barron | Loves(Barron,c1).  [resolve(37,b,10,a)].
57 -Person(c1) | c1 = Melania | Loves(Melania,c1).  [resolve(37,b,8,a)].
58 -Person(c1) | c1 = Donald | Loves(Donald,c1).  [resolve(37,b,7,a)].
59 c1 = Donald | c1 = Ivanka | c1 = Barron | -Person(x) | c1 = x | Loves(x,c1).  [para(17(b,2),37(a,1)),flip(a),flip(b),flip(c),unit_del(d,8)].
60 c1 = Donald | c1 = Ivanka | c1 = Barron | Loves(Donald,c1).  [factor(59,a,e),unit_del(d,7)].
66 c1 = Donald | c1 = Ivanka | c1 = Barron.  [para(17(b,2),60(d,2)),flip(a),flip(b),flip(c),merge(d),merge(e),merge(f),unit_del(d,47)].
68 c1 = Donald | c1 = Barron | Loves(Barron,c1).  [para(66(b,1),55(a,1)),merge(d),unit_del(c,9)].
105 c1 = Donald | c1 = Barron.  [para(66(b,1),68(c,2)),merge(c),merge(d),unit_del(c,48)].
109 c1 = Donald | Loves(Donald,c1).  [para(105(b,1),58(a,1)),merge(c),unit_del(b,10)].
110 c1 = Donald.  [para(105(b,1),109(b,2)),merge(b),unit_del(b,45)].
112 $F.  [back_rewrite(57),rewrite([110(1),110(3),110(7)]),flip(b),unit_del(a,7),unit_del(b,29),unit_del(c,50)].

============================== end of proof ==========================
```

Ερώτημα c (vii)

```
============================== prooftrans ============================
Prover9 (32) version Dec-2007, Dec 2007.
Process 2130 was started by giannhskp on MacBook-Air-Giannes,
Wed Jan 27 23:40:04 2021
The command was "/Applications/Prover9-Mace4-v05B.app/Contents/Resources/bin-mac-intel/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.01 (+ 0.00) seconds.
% Length of proof is 9.
% Level of proof is 2.
% Maximum clause weight is 15.
% Given clauses 21.

6 (exists x exists y exists z (Person(x) & Person(y) & Person(z) & Loves(x,y) & Loves(x,z) & y != z)) # label(non_clause) # label(goal).  [goal].
7 Person(Donald).  [assumption].
9 Person(Ivanka).  [assumption].
11 Loves(Donald,Donald).  [assumption].
12 Loves(Donald,Ivanka).  [assumption].
26 Donald != Ivanka.  [assumption].
27 Ivanka != Donald.  [copy(26),flip(a)].
36 -Person(x) | -Person(y) | -Person(z) | -Loves(x,y) | -Loves(x,z) | z = y.  [deny(6)].
61 $F.  [ur(36,b,9,a,c,7,a,d,12,a,e,11,a,f,27,a(flip)),unit_del(a,7)].

============================== end of proof ==========================
```

Ερώτημα d

```
============================== prooftrans ============================
Prover9 (32) version Dec-2007, Dec 2007.
Process 2143 was started by giannhskp on MacBook-Air-Giannes,
Wed Jan 27 23:41:09 2021
The command was "/Applications/Prover9-Mace4-v05B.app/Contents/Resources/bin-mac-intel/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.00 (+ 0.00) seconds: Ivanka.
% Length of proof is 3.
% Level of proof is 1.
% Maximum clause weight is 3.
% Given clauses 0.

7 Loves(Donald,Ivanka).  [assumption].
31 -Loves(Donald,x) # answer(x).  [assumption].
32 $F # answer(Ivanka).  [resolve(31,a,7,a)].

============================== end of proof ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 2 at 0.00 (+ 0.00) seconds: Donald.
% Length of proof is 3.
% Level of proof is 1.
% Maximum clause weight is 3.
% Given clauses 0.

6 Loves(Donald,Donald).  [assumption].
31 -Loves(Donald,x) # answer(x).  [assumption].
33 $F # answer(Donald).  [resolve(31,a,6,a)].

============================== end of proof ==========================
```

Άσκηση 10 (Bonus)

- $(\exists x)(P(x) \land Q(x)) \implies (\exists x)P(x) \land (\exists x)Q(x)$

Θέλουμε να αποδείξουμε αν ο παραπάνω τύπος είναι έγκυρος ή όχι.

Έστω : $\phi = (\exists x)(P(x) \land Q(x))$ και $\psi = (\exists x)P(x) \land (\exists x)Q(x)$

Από βασικό θεώρημα γνωρίζουμε ότι $\phi \vDash \psi$ ανν ο τύπος $\phi \implies \psi$ (δηλαδή ο αρχικός μας τύπος) είναι έγκυρος. Άρα μας αρκεί να αποδείξουμε ότι $\phi \vDash \psi$.

Έστω μια τυχαία ερμηνεία I και μια τυχαία ανάθεση μεταβλητών s για τις οποίες ισχύει:

$$\vDash_I (\exists x)(P(x) \land Q(x))[s]$$

Σύμφωνα με τον ορισμό της ικανοποίησης, για κάθε τύπο ο οποίος περιέχει έναν υπαρξιακό ποσοδείκτη, υπάρχει ένα $d \in |I|$ τέτοιο ώστε:

$$\vDash_I (P(x) \land Q(x))[s(x/d)]$$

Σύμφωνα με τον ορισμό της ικανοποίησης για τους τύπους που περιέχουν σύζευξη, ισχύει:

$$\vDash_I P(x)[s(x/d)] \quad \text{και} \quad \vDash_I Q(x)[s(x/d)]$$

Εφαρμόζοντας πάλι τον ορισμό την ικανοποίησης για τους τύπους με υπαρξιακό ποσοδείκτη, έχουμε:

$$\vDash_I (\exists x)P(x)[s] \quad \text{και} \quad \vDash_I (\exists x)Q(x)[s]$$

Τέλος, από τον ορισμό της ικανοποίησης για τους τύπους με σύζευξη, έχουμε:

$$\vDash_I (\exists x)P(x) \land (\exists x)Q(x)[s]$$

Δηλαδή αποδείξαμε ότι $\phi \vDash \psi$, επομένως ο τύπος $\phi \implies \psi$ είναι έγκυρος.

Άρα ο τύπος $(\exists x)(P(x) \land Q(x)) \implies (\exists x)P(x) \land (\exists x)Q(x)$ **είναι έγκυρος**.

- $(\exists x)P(x) \land (\exists x)Q(x) \implies (\exists x)(P(x) \land Q(x))$

Έστω μια ερμηνεία για τους ακέραιους αριθμούς.
Υποθέτουμε ότι η P(x) αναφέρεται στο ότι ένας αριθμός x είναι θετικός
και η Q(x) αναφέρεται στο ότι ένας αριθμός x είναι αρνητικός.
Το πεδίο τιμών είναι οι ακέραιοι αριθμοί.

Εύκολα συμπεραίνουμε ότι υπάρχει ένας αριθμός ο οποίος είναι θετικός έτσι ώστε να ικανοποιείται η $(\exists x)P(x)$ και υπάρχει ένας διαφορετικός αριθμός x τέτοιος ώστε να ικανοποιείται η $(\exists x)Q(x)$.

<u>Παρατήρηση:</u> η εμβέλεια των υπαρξιακών ποσοδεικτών στον τύπο $(\exists x)P(x) \land (\exists x)Q(x)$ είναι μόνο το αντίστοιχο σύμβολο κατηγορήματος το οποίο "βρίσκεται δίπλα" στον ποσοδείκτη.

Επομένως, ο τύπος $(\exists x)P(x) \land (\exists x)Q(x)$ ικανοποιείται από την ερμηνεία των ακεραίων.

Όμως, ο τύπος $(\exists x)(P(x) \land Q(x))$ δεν ικανοποιείται καθώς δεν υπάρχει κάποιος ακέραιος ο οποίος να είναι ταυτόχρονα θετικός και αρνητικός.

Δηλαδή έχουμε ότι: $(\exists x)P(x) \land (\exists x)Q(x) \nvDash (\exists x)(P(x) \land Q(x))$

Επομένως από το βασικό θεώρημα που αναφέρθηκε στο πρώτο ερώτημα, έχουμε ότι ο τύπος:
$(\exists x)P(x) \land (\exists x)Q(x) \implies (\exists x)(P(x) \land Q(x))$ **δεν είναι έγκυρος.**