ΚΡΥΩΝΑΣ ΠΑΡΑΣΚΕΥΑΣ
it114/14
it14114@uom.edu.gr

ΑΣΚΗΣΗ 1

```
%       EXERCISE 1


%       exams/2
%       For this specific problem, this predicate returns the list of start times for the five
%       exams (L) as well as the total amount of time needed for them all to be done (Duration).

exams(L, Duration):-
        Starts = [T1,T2,T3,T4,T5],
        Ends = [E1,E2,E3,E4,E5],
        Starts #:: 0..inf,
        Ends #:: 0..inf,
        E1 #= T1+15,
        E2 #= T2+18,
        E3 #= T3+23,
        E4 #= T4+13,
        E5 #= T5+21,
        disjunctive([T1,T3],[15,23]),
        cumulative(Starts,[15,18,23,13,21],[12,20,19,25,7],40),
        ic_global:maxlist(Ends,MakeSpan),
        bb_min(labeling(Starts),MakeSpan,bb_options{strategy:restart}),
        T11 #= 10 * T1,
        T12 #= 10 * T2,
        T13 #= 10 * T3,
        T14 #= 10 * T4,
        T15 #= 10 * T5,
        L = [T11,T12,T13,T14,T15],
        Duration is 10 * MakeSpan.
```

ΣΧΟΛΙΑ

Ακολοθούθηκε το tip της άσκησης καθώς χωρίς αυτό οι υπολογισμοί ήταν πολύ πιο απαιτητικοί. Για λόγους αναπαράστασης, οι τιμές που βρέθηκαν από την λύση του προβλήματος πολλαπλασιάστηκαν με το 10 στο τέλος της άσκησης.

Παρουσιάστηκε πρόβλημα με τα αρχικά νούμερα του προβλήματος το οποίο λύθηκε όταν και οι φοιτητές μειώθηκαν κατά παράγοντα του 10. Συγκεκριμένα η γραμμή
cumulative(Starts,[15,18,23,13,21],[120,200,190,250,70],400)
ήταν υπεύθυνη για αρνητική απάντηση από την ECLiPSe, σε σημείο που το 400 έπρεπε να αλλάξει σε 430 για να δουλέψει το πρόγραμμα σωστά.

Δεν συμπεριλήφθηκε κάποιο παράδειγμα εκτέλεσης καθώς το αποτέλεσμα είναι ακριβώς ίδιο με αυτό που δίνεται στο Coursework2.

ΑΣΚΗΣΗ 2

```
%       EXERCISE 2


%       list/2
%       list(N,ListOfNumbers)

list(1,[2,3,4,10,22,11,17]).
list(2,[1,2,3,4,10,22,11,11,10,24]).
list(3,[2,3,4,5,10,23,10,22,11,17]).

%       allign/3
%       Uses collect_trios and collect_sums to solve the specified problem. Afterwards it makes
%       extensive use of element and element_aux in order to properly calculate Triple and Pos.

allign(Pos,Triple,Sum):-
        list(1,L1),
        list(2,L2),
        list(3,L3),
        collect_trios(L1,L2,L3,L4),
        L4 \= [],
        collect_sums(L4,Sums),
        ic_global:maxlist(Sums,Sum),
        element(I,Sums,Sum),
        I7 is I*3-2,
        element(I7,L4,FirstValue),
        I11 is I7 + 1,
        I12 is I7 + 2,
        element(I11,L4,SecondValue),
        element(I12,L4,ThirdValue),
        Triple = [FirstValue,SecondValue,ThirdValue],
        element_aux(I8,L1,FirstValue,SecondValue,ThirdValue),
        element_aux(I9,L2,FirstValue,SecondValue,ThirdValue),
        element_aux(I10,L3,FirstValue,SecondValue,ThirdValue),
        Pos = [I8,I9,I10].

%       collect_trios/3
%       Looks up every trio that can be formed by the elements of the first list (L1) in the
%       other two (L2 and L3). All valid combinations are stored in L4.

collect_trios([_,_],_,_,[]).
collect_trios([H1,H2,H3|T],L2,L3,L4):-
        element(I1,L2,H1),
        element(I2,L2,H2),
        element(I3,L2,H3),
        I2 #= I1 + 1,
        I3 #= I2 + 1,
        element(I4,L3,H1),
        element(I5,L3,H2),
        element(I6,L3,H3),
        I5 #= I4 + 1,
```

```
        I6 #= I5 + 1,
        !,
        L4 = [H1,H2,H3|Temp],
        collect_trios([H2,H3|T],L2,L3,Temp).
collect_trios([_,H2,H3|T],L2,L3,L4):-
        collect_trios([H2,H3|T],L2,L3,L4).

%       collect_sums/2
%       Sums each trio found in the list given and saves each result in the second list.

collect_sums([],[]).
collect_sums([H1,H2,H3|T1],[H4|T2]):-
        H4 is H1+H2+H3,
        collect_sums(T1,T2).

%       element_aux/3
%       ECLiPSe Prolog's element/3 was found inadequate for part of this exercise since its
%       results were messed up by the existence of duplicate elements. element_aux actually
%       looks ahead to the two elements that follow the one whose spot it's looking for, simply
%       to get around this specific issue.

element_aux(1,[First,Second,Third|_],First,Second,Third).
element_aux(Sol,[_|Rest],F,S,T):-
        element_aux(RecursiveSol,Rest,F,S,T),
        Sol is RecursiveSol+1.
```

ΣΧΟΛΙΑ

Παρά την αχρείαστη πολυπλοκότητα του κώδικα, δεν βρέθηκε κάποιο πρόβλημα ή bug. Η λύση που δίνεται είναι μακριά από την βέλτιστη και πιο συγκεκριμένα μακριά από το “C” μέρος του “CLP” λόγω κυρίως ελιππούς κατανόησης, αλλά κατά τ' άλλα λειτουργική. Χρησιμοποιούνται τρία βοηθητικά ορίσματα: το πρώτο δημιουργεί μία λίστα με όλες τις υποψήφιες τριάδες (δηλαδή τις συνεχόμενες που εμφανίζονται και στις τρεις λίστες), το δεύτερο δημιουργεί μία άλλη λίστα την οποία γεμίζει με τα αθροίσματα των προαναφερθεισών τριάδων (αλλά δεν τις αθροίζει και μεταξύ τους), ενώ το τρίτο αποτελεί μία ειδική υλοποίηση του element/3 (ένας “αθροιστής” μεγαλώνει μέχρι να βρεθεί το σημείο εκκίνησης της τριάδας).

Η εύρεση του Sum επιτυγχάνεται μέσω αρχικά χρήσης generate and test (collect_trios), και έπειτα απλής αναδρομής (collect_sums). Το χάος που ακολουθεί έπειτα εξηγείται ως εξής: αρχικά βρίσκεται η θέση του μεγαλύτερου αθροίσματος (Sum) στην λίστα αθροισμάτων (Sums). Έπειτα η θέση αυτή (I) χρησιμοποιείται για να βρεθεί η θέση (I7) του πρώτου στοιχείου της τριάδας (FirstValue) στην οποία αντίστοιχει το άθροισμα αυτό. Αφού έχει βρεθεί η θέση του πρώτου στοιχείου τα επόμενα δύο (SecondValue και ThirdValue) βρίσκονται μέσω των θέσεών τους στην L4. Τέλος, αφού το Triple ενοποιηθεί με τις κατάλληλες τιμές χρησιμοποιείται και το τελευταίο βοηθητικό όρισμα για την εύρεση του Pos.

Παρατίθεται το παρακάτω παράδειγμα εκτέλεσης που προκύπτει όταν η δεύτερη λίστα αλλάζει ως εξής:

list(2,[1,2,3,4,10,22,11,**17**,11,10,24]).

?- allign(Pos,Triple,Sum)

Pos = [5, 6, 8]
Triple = [22, 11, 17]
Sum = 50

Επίσης δοκιμάστηκε η εκτέλεση σε σύνολο λιστών χωρίς κοινή συνεχόμενη τριάδα, στο οποίο σημείο συμπεριλήφθηκε η δικλείδα ασφαλείας
L4 \= []
έτσι ώστε να αποφευχθεί το “Abort” και η απάντηση να είναι απλά “No”.

ΑΣΚΗΣΗ 3

```
%       EXERCISE 3

%       Scheduling Application
%       The Newspaper Story

students(WakeA,WakeV,WakeG):-
        Wakes = [WakeA,WakeV,WakeG],
        Wakes #:: 0..90.
```

ΣΧΟΛΙΑ

Η άσκηση αφέθηκε στην αρχή λόγω χρόνου.