ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

«Κατάρτιση ωρολογίου προγράμματος με τεχνικές ακέραιου γραμμικού προγραμματισμού»

Του φοιτητή
Βαλέριϊ Κρισιούκ
Αρ. Μητρώου: 144186

Επιβλέπων
Ευστάθιος Αντωνίου
Καθηγητής

10 Σεπτεμβρίου 2024

Τίτλος Δ.Ε. Κατάρτιση ωρολογίου προγράμματος με τεχνικές ακέραιου γραμμικού προγραμματισμού

Κωδικός Δ.Ε. 23123

Ονοματεπώνυμο φοιτητή: Βαλέριϊ Κρισιούκ

Ονοματεπώνυμο εισηγητή Ευστάθιος Αντωνίου

Ημερομηνία ανάληψης Δ.Ε. 08-03-2023

Ημερομηνία περάτωσης Δ.Ε. 10-9-2024

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Η έγκριση της διπλωματικής εργασίας από το Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος, δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα, εκ μέρους του Τμήματος.

«Στους γονείς μου και τα αδέρφια μου»

Πρόλογος

Τα μαθηματικά αποτελούν αναπόσπαστο κομμάτι του προγραμματισμού. Μαθηματικές μέθοδοι και τεχνικές συχνά είναι απαραίτητοι για την επίλυση σύνθετων προβλημάτων. Η συγκεκριμένη Πτυχιακή Εργασία ασχολείται με τον Ακέραιο Γραμμικό προγραμματισμό και αποτέλεσε ερέθισμα για να διευρύνω τις γνώσεις μου και παράλληλα δοκιμάσω τις δυνατότητές μου πάνω σε όσα έχω διδαχθεί κατά τη διάρκεια των σπουδών.

Περίληψη

Στην παρούσα πτυχιακή εργασία παρουσιάζεται η εφαρμογή τεχνικής Ακέραιου Γραμμικού Προγραμματισμού για την επίλυση του προβλήματος ωρολόγιου προγράμματος (timetabling) Πανεπιστημίου. Η υλοποίηση πραγματοποιείται με την γλώσσα προγραμματισμού Python.

Στο πρώτο κεφάλαιο γίνεται μια βιβλιογραφική επισκόπηση του Γραμμικού και του Ακέραιου Γραμμικού Προγραμματισμού.

Στο δεύτερο κεφάλαιο παρουσιάζεται γενικότερα τα χαρακτηριστικά και το πρόβλημα ωρολογίου προγράμματος για εκπαιδευτικά ιδρύματα.

Στο τρίτο κεφάλαιο πραγματοποιείται η κατάστρωση του μαθηματικού μοντέλου του προβλήματος. Παρουσιάζονται αναλυτικά όλα τα δεδομένα, μεταβλητές, περιορισμοί και η συνάρτηση απόφασης.

Στο τέταρτο κεφάλαιο αναλύεται ο κώδικας που επιλύει το πρόβλημα με την χρήση της γλώσσας προγραμματισμού Python και της μαθηματικής βιβλιοθήκης Pulp. Επίσης παρουσιάζεται αναλυτικά η εισαγωγή δεδομένων στο πρόγραμμα από φύλλο Excel και η επεξεργασία του φύλλου Excel όπου θα αποθηκευτεί τελικά η βέλτιστη λύση που θα βρεθεί.

Στο τέλος ακολουθούν η ανάλυση αποτελεσμάτων, τα συμπεράσματα και η βιβλιογραφία.

«Timetable compilation with integer linear programming techniques»

Valery Krysiuk

Abstract

This thesis presents the application of the Integer Linear Programming technique to solve the timetabling problem of a university. The implementation is carried out using the Python programming language.

In the first chapter, a review of the literature on linear programming and integer linear programming is given.

The general characteristics and timetabling problems of educational institutions are presented in the second chapter.

The third chapter is the development of the mathematical model of the problem. All data, variables, constraints and the decision function are presented in detail.

The fourth chapter analyses the code that solves the problem using the Python programming language and the Pulp mathematical library. It also describes the data input to the program from an Excel spreadsheet and the processing of the Excel spreadsheet where the optimal solution found is finally stored.

Lastly, the conclusions and references are given.

Ευχαριστίες

Η παρούσα Πτυχιακή Εργασία εκπονήθηκε κατά το ακαδημαϊκό έτος 2023-2024 στα πλαίσια των σπουδών μου στη Σχολή Μηχανικών του Τμήματος Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνές Πανεπιστημίου Ελλάδος. Θα ήθελα να ευχαριστήσω όλους όσους συνέβαλαν στην ολοκλήρωση της εργασίας.

Επιβλέπων καθηγητής της εργασίας ήταν ο κ. Αντωνίου Ευστάθιος, Καθηγητής στο Διεθνές Πανεπιστημίου Ελλάδος, τον οποίο θα ήθελα να ευχαριστήσω για την ανάθεση του θέματος. Η συμβολή του στο ξεκίνημα και στην ολοκλήρωση της εργασίας ήταν καθοριστική.

Επιπλέων οφείλω ένα μεγάλο ευχαριστώ σε όλα τα μέλη της οικογένειας μου, οι οποίοι με στήριζαν καθ' όλη τη διάρκεια των σπουδών μου.

Τέλος, θα ήθελα να ευχαριστήσω την σύντροφό μου, Λυκοπούλου Χριστίνα για τη συνεχή υποστήριξη της όλα αυτά τα χρόνια, η εμπιστοσύνη και η αγάπη της μου έδωσε δύναμη να πιστέψω στον εαυτό μου.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Συντομογραφίες

Δ.Ε.	Διπλωματική Εργασία
ΔΙΠΑΕ	Διεθνές Πανεπιστήμιο Ελλάδος
Π.Ε.	Πτυχιακή Εργασία
ΓΠ	Γραμμικός Προγραμματισμός
ΑΓΠ	Ακέραιος Γραμμικός Προγραμματισμός
LP	Liner Programming
IP	Integer Programming
ILP	Integer Liner Programming

Κεφάλαιο 1ο: Γραμμικός Προγραμματισμός και Γραμμικός Ακέραιος Προγραμματισμός

1.1 Εισαγωγή

Η διαδικασία χρονοπρογραμματισμού της διδασκαλίας μαθημάτων σε μια εκπαιδευτική μονάδα, ιδιαίτερα σε ένα Πανεπιστημιακό Ίδρυμα με πολλά εξάμηνα και μαθήματα, μπορεί να αποδειχθεί ιδιαίτερα επίπονη και χρονοβόρα, με υψηλές πιθανότητες λαθών που οφείλονται στον ανθρώπινο παράγοντα. Από την άποψη της αλγοριθμικής πολυπλοκότητας το συγκεκριμένο πρόβλημα ανήκει στην κλάση των NP-Hard προβλημάτων, γεγονός που το καθιστά δυσεπίλυτο ιδιαίτερα όταν το πλήθος των πόρων που ζητείται να προγραμματιστούν είναι μεγάλο. Εξαιτίας αυτού του γεγονότος, το εν λόγω πρόβλημα αποτέλεσε και συνεχίζει να αποτελεί ενεργό αντικείμενο έρευνας, ενώ για την επίλυση του έχουν προταθεί λύσεις που βασίζονται σε μια πληθώρα διαφορετικών τεχνικών ανάλυσης και προγραμματισμού.

Στόχος της πτυχιακής εργασίας είναι η εφαρμογή τεχνικών ακέραιου γραμμικού προγραμματισμού (Integer Linear Programming - ILP) για την κατάρτιση ωρολογίου προγράμματος για εκπαιδευτικές μονάδες όπως πανεπιστημιακά τμήματα, σχολεία κ.λπ.

Σε αυτό το κεφάλαιο θα γίνει μια γενική αναφορά για τον γραμμικό και τον ακέραιο γραμμικό προγραμματισμό.

1.2 Ορισμοί

Γραμμικός προγραμματισμός: Ο γραμμικός προγραμματισμός, ασχολείται με τη λύση θεωρητικών και πρακτικών προβλημάτων βελτιστοποίησης όπου η αντικειμενική συνάρτηση είναι πρώτης τάξης, αλλά οι περιορισμοί της εκφράζονται επίσης μέσω ανισοτήτων πρώτης τάξης. Ο πρώτος άνθρωπος που έλυσε ένα τέτοιο πρόβλημα ήταν ο Σοβιετικός μαθηματικός L.V. Kantorovich και οι Koopmans και Hitchcock, νικητές του Νόμπελ Οικονομικών του 1975, θεωρούνται οι εφευρέτες του ονόματος «Γραμμικός Προγραμματισμός». Αλλά η μεγαλύτερη ανακάλυψη σε αυτόν τον τομέα έγινε πέρα από τον Ατλαντικό από τον George B. Dantzig, έναν διακεκριμένο μαθηματικό στο Πανεπιστήμιο του Σικάγο που ανακάλυψε τη μέθοδο simplex το 1947.

Γραμμικός προγραμματισμός: Ο γραμμικός προγραμματισμός ασχολείται με το πρόβλημα της κατανομής περιορισμένων πόρων μεταξύ διαφόρων ανταγωνιστικών δραστηριοτήτων με βέλτιστο τρόπο (Vanderbei, 2001).

Γραμμικός προγραμματισμός: Ο γραμμικός προγραμματισμός (Linear Programming, LP) είναι αναμφίβολα το πιο δημοφιλές μοντέλο στην επιχειρηματική έρευνα και την επιστήμη διοικητικής διαχείρισης (management science). Η εφαρμογή του σε προβλήματα λήψης αποφάσεων σε ιδιωτικές και δημόσιες επιχειρήσεις και οργανισμούς έχει μεγάλη επιτυχία, αφενός χάρη στα ερευνητικά αποτελέσματα μαθηματικών και οικονομολόγων σε θεωρητικό επίπεδο και από την άλλη στην επαναστατική ανάπτυξη της πληροφορικής επιστήμης και τεχνολογίας. Είναι πλέον γενικά αποδεκτό ότι τα τρία τέταρτα των εφαρμογών των μοντέλων επιχειρηματικής έρευνας σε πρακτικά προβλήματα διαχείρισης περιλαμβάνουν γραμμικό προγραμματισμό (LP) [1].

Οι ερευνητές επιχειρήσεων ή οι αναλυτές προβλημάτων αποφάσεων χρησιμοποιούν τον γραμμικό προγραμματισμό για να λύσουν το πρόβλημα της βέλτιστης κατανομής περιορισμένων πόρων ή μέσων σε εναλλακτικές και ανταγωνιστικές δραστηριότητες. Αυτό είναι το περίφημο πρόβλημα διανομής

«πίτας» (resource allocation problem). Τα προβλήματα αποφάσεων αυτής της μορφής περιλαμβάνουν, για παράδειγμα, την κατανομή εργατικού δυναμικού, τεχνικού εξοπλισμού και πρώτων υλών σε διάφορες παραγωγικές διαδικασίες, την κατανομή κεφαλαίου σε διάφορα επενδυτικά σχέδια, την κατανομή περιορισμένου προσωπικού σε διάφορες υπηρεσίες, την κατανομή καλλιεργήσιμης γης σε διάφορες διανομή υπηρεσιών. Τα αναμενόμενα αποτελέσματα (κριτήρια απόφασης) αυτών των αποφάσεων όπως οι αγροτικές δραστηριότητες μπορεί να είναι η μεγιστοποίηση των συνολικών κερδών από τις πωλήσεις, η ελαχιστοποίηση του συνολικού κόστους παραγωγής, η μεγιστοποίηση της απασχόλησης, η ελαχιστοποίηση των αρνητικών επιπτώσεων στο περιβάλλον κ.λπ. Ο όρος "γραμμικός" σημαίνει ότι τόσο η αντικειμενική συνάρτηση όσο και οι μαθηματικές συναρτήσεις που εμφανίζονται στους περιορισμούς πρέπει να είναι γραμμικές συναρτήσεις, ενώ ο όρος "προγραμματισμός" δεν αναφέρεται σε γλώσσα προγραμματισμού αλλά είναι συνώνυμο του σχεδιασμού [1].

Γραμμικός Ακέραιος Προγραμματισμός: Ο Γραμμικός Ακέραιος Προγραμματισμός είναι ουσιαστικά η μαθηματική μοντελοποίηση ενός προβλήματος βελτιστοποίησης. Ο όρος «γραμμικό» σημαίνει ότι η συνάρτηση του μοντέλου πρέπει να είναι γραμμική. Ο όρος "ακέραιος" σημαίνει ότι οι τιμές των μεταβλητών για ένα δεδομένο πρόβλημα είναι όλες ή εν μέρει ακέραιες. Ο όρος προγραμματισμός αναφέρεται σε θέματα προγραμματισμού [2]. Ένα γραμμικό μοντέλο προγραμματισμού στο οποίο όλες οι μεταβλητές απόφασης πρέπει να έχουν ακέραιες τιμές ονομάζεται ακέραιο πρόβλημα προγραμματισμού. Τα προβλήματα στα οποία μόνο ορισμένες από τις μεταβλητές απόφασης πρέπει να έχουν ακέραιες τιμές ονομάζονται προβλήματα προγραμματισμού μικτών ακεραίων. Μερικές φορές, ορισμένες (ή όλες) μεταβλητές απόφασης πρέπει να λαμβάνουν την τιμή 0 ή 1. Τέτοια προβλήματα ονομάζονται προβλήματα προγραμματισμού μεικτού ακέραιου αριθμού μηδέν-ένα ή προβλήματα προγραμματισμού δυαδικού (ακέραιου).

Όταν για ένα πρόβλημα Γραμμικού Προγραμματισμού δεν ισχύει για κάποιες μεταβλητές απόφασης η υπόθεση της διαιρετότητας και οι μεταβλητές αυτές πρέπει υποχρεωτικά να λαμβάνουν ακέραιες τιμές. Το πρόβλημα αυτό ονομάζεται πρόβλημα Ακέραιου Γραμμικού Προγραμματισμού (ILP=Integer Linear Programming). Όταν όλες οι μεταβλητές είναι ακέραιες τότε το πρόβλημα λέγεται αμιγώς ακέραιο (Pure Integer Program), ενώ αν μόνο κάποιες από τις μεταβλητές είναι ακέραιες τότε το πρόβλημα λέγεται μικτό ακέραιο πρόγραμμα (Mixed Integer Program).

1.3 Ιστορική αναδρομή Γραμμικού Προγραμματισμού και Ακέραιου Γραμμικού Προγραμματισμού

Ξεκινώντας από τα μέσα του 20ου αιώνα, ο γραμμικός προγραμματισμός αναπτύχθηκε γρήγορα και δικαίως έχει χαρακτηριστεί ως μία από τις σημαντικότερες εξελίξεις στην επιστήμη, αφού η επίδρασή του σε δεκάδες πεδία ήταν καταλυτική. Στην εποχή μας, είναι ένα εργαλείο που χρησιμοποιείται ευρέως από μεγάλες και μικρές εταιρείες σε όλο τον κόσμο και η χρήση του σε άλλους τομείς εξαπλώνεται ραγδαία. Υπάρχουν δεκάδες βιβλία για τη χρήση του γραμμικού προγραμματισμού, και εκατοντάδες αντίστοιχες δημοσιεύσεις [3].

Το πρόβλημα της επίλυσης συστημάτων γραμμικών ανισοτήτων ανακαλύφθηκε για πρώτη φορά πριν από πολύ καιρό. Το 1827, ο Jean Batist Joseph Fourier δημοσίευσε μια λύση σε ένα τέτοιο πρόβλημα [4]. Αυτή η μέθοδος ονομάζεται Fourier-Motzkin Elimination.

Ο Leonid Kantorovich μοντελοποίησε για πρώτη φορά ένα πρόβλημα γραμμικού προγραμματισμού ισοδύναμο με ένα γενικό πρόβλημα γραμμικού προγραμματισμού το 1939, και πρότεινε επίσης μια λύση [5]. Ο Kantorovich ουσιαστικά εισήγαγε για πρώτη φορά τεχνικές γραμμικού προγραμματισμού, χρησιμοποιώντας αρχικά κατά τη διάρκεια του Β' Παγκοσμίου Πολέμου για να ελαχιστοποιήσει το

κόστος διατήρησης της στρατιωτικής ισχύος μεγιστοποιώντας παράλληλα την αποτελεσματικότητα στο πεδίο της μάχης. Ωστόσο, το έργο του δεν διαδόθηκε ευρέως [3], [6].

Το 1941, ο Frank Lauren Hitchcock διατύπωσε επίσης το πρόβλημα μεταφοράς ως πρόβλημα γραμμικού προγραμματισμού, δίνοντας μια λύση πολύ κοντά σε αυτή που δόθηκε αργότερα με τη μέθοδο simplex [5]. Ο Hitchcock πέθανε το 1957 και τιμήθηκε μετά θάνατον με το βραβείο Νόμπελ.

Μεταξύ 1946 και 1947, ο George Bernard Dantzig ανέπτυξε τη δική του διατύπωση του προβλήματος γραμμικού προγραμματισμού για χρήση στην επίλυση προβλημάτων σχεδιασμού για την Πολεμική Αεροπορία των Ηνωμένων Πολιτειών. Το 1947, ο Dantzig ανακάλυψε επίσης τη μέθοδο Simplex, η οποία αντιμετώπιζε αποτελεσματικά προβλήματα γραμμικού προγραμματισμού σε μέγιστα σύνολα για πρώτη φορά. Ο George Dantzig πρότεινε ότι οι αλληλεπιδράσεις μεταξύ μεγάλων οργανωτικών δραστηριοτήτων μπορούν να θεωρηθούν ως μοντέλο ενός προβλήματος γραμμικού προγραμματισμού και ότι ένα βέλτιστο πρόγραμμα (λύση) μπορεί να επιτευχθεί ελαχιστοποιώντας μια (μοναδική) γραμμική αντικειμενική συνάρτηση. Η Πολεμική Αεροπορία ξεκίνησε το έργο SCOOP (Scientific Computing of Optimal Procedures). Το έργο SCOOP ξεκίνησε τον Ιούνιο του 1947 και μέχρι το τέλος εκείνου του καλοκαιριού ο Dantzig και οι συνεργάτες του είχαν αναπτύξει ένα αρχικό μαθηματικό μοντέλο ενός γενικού γραμμικού προγραμματισμού προβλήματος και μια γενική μέθοδο λύσης που ονομάζεται μέθοδος simplex. Όταν ο Danziger συναντήθηκε με τον John von Neumann για να συζητήσουν την προσέγγισή του, ο Neumann συνέλαβε τη θεωρία της δυαδικότητας, συνειδητοποιώντας ότι τα προβλήματα που μελετούσε στη θεωρία παιγνίων ήταν ισοδύναμα. Ο Dantzig το απέδειξε σε μια αναφορά στις 5 Ιανουαρίου 1948 για το «Θεώρημα Γραμμικής Ανισότητας» [6]. Μετά τον πόλεμο, πολλές βιομηχανίες το ενσωμάτωσαν στον καθημερινό τους σχεδιασμό. Σε αντίθεση με άλλους οικονομολόγους της γενιάς του, ο Dantzig έβλεπε τον γραμμικό προγραμματισμό όχι μόνο ως ένα ποιοτικό εργαλείο για την ανάλυση των οικονομικών φαινομένων, αλλά και ως μια μέθοδο που θα μπορούσε να δώσει ουσιαστικές λύσεις σε πρακτικά προβλήματα. Σήμερα, η μέθοδος simplex παραμένει ένα θεμελιώδες υπολογιστικό εργαλείο για γραμμικό και ακέραιο προγραμματισμό. Οι Kantorovich και Dantzig, μαζί με τον John von Neumann (1903-1957) που εισήγαγε τη θεωρία της δυαδικότητας, θεωρούνται οι ιδρυτές του γραμμικού προγραμματισμού [3].

Περίπου την ίδια εποχή, ο Ολλανδοαμερικανός οικονομολόγος T.C Koopmans μοντελοποίησε τα κλασικά οικονομικά προβλήματα ως προβλήματα γραμμικού προγραμματισμού. Το 1975, ο Kantorovich και ο Koopmans έλαβαν από κοινού το Νόμπελ Οικονομικών Επιστημών [4].

Ο Leonid Khachiyan έδειξε το 1979 ότι τα προβλήματα γραμμικού προγραμματισμού μπορούν να λυθούν σε πολυωνυμικό χρόνο [7]. Αλλά η μεγαλύτερη θεωρητική ανακάλυψη (και όχι απλώς μια θεωρητική ανακάλυψη) ήταν το 1984, όταν ο Narendra Karmarkar εισήγαγε τη μέθοδο εσωτερικού σημείου για την επίλυση προβλημάτων γραμμικού προγραμματισμού [4].

Στη δεκαετία του 1950 άρχισε η κατάλληλη μελέτη του ακέραιου γραμμικού προγραμματισμού. Αρχικά, ο Dantzig [8] έδειξε ότι τα προβλήματα μεταφορών αποτελούν LP-προβλήματα που έχουν αυτόματα ακέραιες βέλτιστες λύσεις, αναπτύσσοντας μια ειδική μορφή της μεθόδου simplex για προβλήματα μεταφορών.

Στη συνέχεια, αρκετά άλλα συνδυαστικά προβλήματα μελετήθηκαν ως ακέραια γραμμικά προγράμματα, όπως το πρόβλημα της σταθερής φόρτισης [9], [10] το πρόβλημα του πλανόδιου πωλητή [11], [12-16], [17], [18], ο οποίος αναζήτησε τις όψεις του πλανόδιου πωλητή, το πρόβλημα της κοπής (κοπή χαρτιού εφημερίδων) [35], το πρόβλημα της τροφοδοσίας [19], [20], [21], τα προβλήματα ροής

και μεταφοράς [22], [23], [24], [25], [26], και προβλήματα ανάθεσης [27], [28], [29-30], [31], [32], [33].

1.4 Χαρακτηρίστηκα Γραμμικού Προγραμματισμού και Γραμμικού Ακέραιου Προγραμματισμού

Τα ακέραια προβλήματα προγραμματισμού είναι ένας ειδικός τύπος προβλημάτων γραμμικού προγραμματισμού. Στον γραμμικό προγραμματισμό, ισχύει η υπόθεση διαιρετότητας, η οποία επιτρέπει στις μεταβλητές να λαμβάνουν κλασματικές τιμές. Στα προβλήματα ακέραιου προγραμματισμού αυτή η υπόθεση δεν ισχύει, επομένως εξετάζονται προβλήματα στα οποία επιτρέπεται σε όλες ή μερικές από τις μεταβλητές να λαμβάνουν ακέραιες τιμές.

Ο Γραμμικός Προγραμματισμός είναι ένας κλάδος του μαθηματικού προγραμματισμού που έχει σχεδιαστεί για την επίλυση προβλημάτων βελτιστοποίησης όπου όλοι οι περιορισμοί και οι στόχοι εκφράζονται ως γραμμικές διαδικασίες. Η πρώτη του εφαρμογή ήταν αφιερωμένη σε δραστηριότητες κατά τη διάρκεια του Β' Παγκοσμίου Πολέμου. Ωστόσο, σύντομα αναγνωρίστηκε η σημασία του και κατέλαβε εξέχουσα θέση στη βιομηχανία και το εμπόριο.

Χρησιμοποιείται για τη λήψη αποφάσεων υπό ντετερμινιστικές συνθήκες, δηλαδή όταν όλοι οι παράγοντες που επηρεάζουν το πρόβλημα είναι γνωστοί και οι επιχειρηματικοί στόχοι και οι περιορισμοί ποσοτικοποιούνται επίσης. Κατά τη διάρκεια αυτής της διαδικασίας, γίνεται επιλογή μίας από όλες τις πιθανές εναλλακτικές που θα παράγουν το καλύτερο αποτέλεσμα. Ο γραμμικός προγραμματισμός μπορεί επίσης να χρησιμοποιηθεί ως μηχανισμός επαλήθευσης και ελέγχου για τον προσδιορισμό της ακρίβειας και της αξιοπιστίας των αποφάσεων που βασίζονται αποκλειστικά στην εμπειρία του διαχειριστή χωρίς τη βοήθεια μαθηματικών μοντέλων.

Τα χαρακτηριστικά ή οι βασικές παραδοχές του γραμμικού προγραμματισμού:

- Μεταβλητές απόφασης και οι αλληλεξαρτήσεις τους
- Αντικειμενικότητα και ποσοτικοποίηση στόχων
- Περιορισμένοι παράγοντες
- Υπάρχουν διαφορετικές εναλλακτικές λύσεις
- Μη αρνητικοί παράγοντες
- Γραμμικό Πρότυπο
- Συμπληρωματικότητα
- Κριτήρια εξαίρεσης
- Διαιρετότητα
- Βεβαιότητα
- Υποθέσεις

1.5 Εφαρμογές Γραμμικού Προγραμματισμού και Γραμμικού Ακέραιου Προγραμματισμού

Πολλά προβλήματα του πραγματικού κόσμου επιλύονται χρησιμοποιώντας μοντέλα γραμμικού προγραμματισμού. Υπάρχουν γνωστά πεδία εφαρμογών στη βιομηχανία, το μάρκετινγκ, τις οικονομικές επενδύσεις, τη διαφήμιση και τη γεωργία. Έχουν αναπτυχθεί αποτελεσματικές τεχνικές για την επίλυση μοντέλων γραμμικού προγραμματισμού. Τα αποτελέσματα που παράγονται από το πακέτο γραμμικού προγραμματισμού παρέχουν χρήσιμες πληροφορίες για τη μελλοντική εξέλιξη της αντικειμενικής συνάρτησης του υπό εξέταση μοντέλου [2].

Οι εφαρμογές του Ακέραιου Γραμμικού Προγραμματισμού εμπίπτουν σε δύο κατηγορίες:

- **Άμεσες**: Η φύση του προβλήματος καθιστά αδύνατη την εκχώρηση κλασματικών τιμών στις μεταβλητές του μοντέλου (π.χ. εύρεση του βέλτιστου πλήθους των μηχανών που απαιτούνται για την πραγματοποίηση μιας εργασίας).
- **Μετασχηματισμένες**: Προκύπτει η ανάγκη χρήσης βοηθητικών ακέραιων μεταβλητών για τη μοντελοποίηση του προβλήματος (π.χ. κατά την αλληλουχία εκτέλεσης δύο εργασιών Α και Β σε μια απλή μηχανή η εργασία Α πρέπει να προηγείται της εργασίας Β).

Όπως ισχύει και στον γραμμικό προγραμματισμό, οι τομείς εφαρμογής του ακέραιου προγραμματισμού και η σημασία του είναι σημαντικότατη στη πράξη επειδή ένας μεγάλος αριθμός καθημερινών πρακτικών προβλημάτων επιχειρηματικής έρευνας είναι προβλήματα ακέραιου προγραμματισμού. Ένα κλασικό παράδειγμα που εμπίπτει σε αυτές τις κατηγορίες προβλημάτων είναι η μαζική παραγωγή προϊόντων με σκοπό τη μεγιστοποίηση των κερδών. Η παραγωγή μη ακέραιου αριθμού ειδών δεν έχει καμία φυσική σημασία, για παράδειγμα η παραγωγή αυτοκινήτων [1].

Ενδεικτικά, τομείς εφαρμογής του ακέραιου προγραμματισμού είναι [1].

- Αριθμός εργαζομένων στο εργοστάσιο παραγωγής,
- Χρονικός προγραμματισμός εργασιών με περιορισμένα διαθέσιμα μέσα,
- Προγραμματισμός πληρώματος αεροπλάνων (aircrew scheduling),
- Προγραμματισμός μεταφορικών μέσων (vehicle schedulingand truck dispatching),
- Επενδυτικός σχεδιασμός (capital budgeting) και διάφοροι άλλοι.

1.6 Προϋποθέσεις Γραμμικού Προγραμματισμού και Ακέραιου Γραμμικού Προγραμματισμού

Για μια δεδομένη κατάσταση ενός προβλήματος, υπάρχουν ορισμένες προϋποθέσεις (ή υποθέσεις/παραδοχές) που πρέπει να πληρούνται προκειμένου να λυθεί το πρόβλημα χρησιμοποιώντας τεχνικές γραμμικού προγραμματισμού [1], [3].

Βεβαιότητα (Certainty): Τα μοντέλα γραμμικού προγραμματισμού υποθέτουν ότι όλες οι παράμετροι του προβλήματος είναι σαφώς προσδιορισμένες.
Γραμμικότητα (Linearity), Αναλογικότητα (Proportionality) και Προσθετικότητα (Additivity): Όλες οι συναρτήσεις προβλημάτων, οι αντικειμενικές συναρτήσεις και οι περιορισμοί πρέπει να είναι γραμμικές ως προς τις άγνωστες μεταβλητές xj , j = 1,...,n. Αυτό σημαίνει ότι πρέπει να ισχύουν οι ιδιότητες της αναλογικότητας και της προσθετικότητας, δηλαδή πρέπει να ισχύει εάν το y είναι συνάρτηση n μεταβλητών και το aj, j = 1,...,n είναι σταθερές. Σε πολλές περιπτώσεις όπου οι γραμμικές συνθήκες δεν ισχύουν αυστηρά, μπορούν να χρησιμοποιηθούν γραμμικές συναρτήσεις για την παροχή αρκετά καλών προσεγγίσεων.
Αντικειμενικός στόχος: Αναφέρεται στον στόχο της βελτιστοποίησης (μεγιστοποίηση κέρδους ή ελαχιστοποίηση κόστους).
Διαιρετότητα (Divisibility): Τα μοντέλα γραμμικού προγραμματισμού υποθέτουν ότι κάθε δραστηριότητα (δηλαδή μεταβλητή) είναι συνεχής και ως εκ τούτου απεριόριστα διαιρούμενη. Αυτό σημαίνει ότι οι δεκαδικές ή ακέραιες τιμές επιτρέπονται για όλα τα επίπεδα δραστηριότητας και κάθε χρήση πόρων. Όταν η υπόθεση διαχωριστικότητάς δεν ισχύει, υπάρχουν δύο πιθανότητες:
Να αγνοηθεί η υπόθεση αυτή, να λυθεί το πρόβλημα χρησιμοποιώντας μεθόδους γραμμικού προγραμματισμού και στρογγυλοποιήστε τις τιμές των μεταβλητών στην πλησιέστερη ακέραια μονάδα. Αυτή η μέθοδος χρησιμοποιείται κυρίως όταν η τιμή της μεταβλητής είναι μεγάλη.
Όταν η τιμή της μεταβλητής είναι πολύ μικρή (όπως 0 ή 1) (όπως σε πολλά επενδυτικά προβλήματα), τότε πρέπει να χρησιμοποιούνται τεχνικές ακέραιου προγραμματισμού.
Εκτός από τις παραπάνω παραδοχές, υιοθετούνται υποθέσεις σε κάθε διαδικασία μοντελοποίησης (κατά περίπτωση) που απλοποιούν και επιτρέπουν την ευελιξία στο μοντέλο που προκύπτει.

Αν η τελευταία προϋπόθεση δεν ισχύει (δηλαδή οι μεταβλητές δεν παίρνουν συνεχείς τιμές αλλά ακέραιες) τότε το πρόβλημα ανάγεται σε πρόβλημα Ακέραιου Γραμμικού Προγραμματισμού (ΑΓΠ). Η μόνη διαφορά με το υπόδειγμα του Γραμμικού Προγραμματισμού είναι ότι προστίθεται περιορισμός για τις ακέραιες μεταβλητές.

1.7 Προβλήματα Ακέραιου Γραμμικού Προγραμματισμού

Τα προβλήματα του Ακέραιου Γραμμικού Προγραμματισμού ανήκουν γενικά σε 3 κατηγορίες:

1. Προβλήματα στα οποία οι μεταβλητές είναι ακέραιες. Λύνονται ως κλασσικά προβλήματα γραμμικού προγραμματισμού.
2. Προβλήματα στα οποία οι μεταβλητές δεν έχουν φυσικό νόημα όπως οι κλασσικές γραμμικές μεταβλητές(πχ ώρες εργασίας) αλλά λογικό νόημα (ναι ή όχι που συνήθως συμβολίζεται με 0 ή 1). Αυτά ονομάζονται Προβλήματα 0/1.
3. Μερικά προβλήματα 0/1 περιλαμβάνουν ταυτόχρονα τόσο κλασικές μεταβλητές όσο και μεταβλητές με λογικό νόημα (0 ή 1).

Προβλήματα που περιλαμβάνουν μεταβλητές οι οποίες είναι από τη φύση τους ποσοτικές και επιβάλλεται να λάβουν ακέραιες τιμές ονομάζονται άμεσα ακέραια μοντέλα. Μια άλλη κατηγόρια είναι τα κωδικοποιημένα (coded) ακέραια μοντέλα, στα οποία οι μεταβλητές απόφασης περιγράφουν ένα αυστηρά περιορισμένο αριθμό δυνατοτήτων. Οι μεταβλητές περιγράφουν αποφάσεις του τύπου «ναι - όχι» ή «αληθές - ψευδές». Σ' αυτά γίνεται χρήση δυαδικών (binary) μεταβλητών του τύπου 1 ή 0, όπου το 1 αντιπροσωπεύει την απόφαση τύπου «ναι» ενώ το 0 αντιπροσωπεύει τη μοναδική άλλη δυνατότητα «όχι».

Επιπρόσθετα, σε μοντέλα ακέραιου προγραμματισμού μπορούν να διαμορφωθούν και τα προβλήματα συνδυαστικής. Ένα πρόβλημα συνδυαστικής συνίσταται στην εύρεση, μεταξύ ενός συνόλου συνδυασμών, εκείνων που βελτιστοποιεί μία αντικειμενική συνάρτηση.

Παραδείγματα τέτοιων προβλημάτων είναι:

- το πρόβλημα του περιοδεύοντος πωλητή σε n πόλεις, αφού περιλαμβάνει (n-1) δυνατές διαδρομές, δηλαδή συνδυασμούς
- και το πρόβλημα της διάταξης n εργασιών σε k μηχανές, αφού περιλαμβάνει (n!)k δυνατές διατάξεις των εργασιών, δηλαδή δυνατούς συνδυασμούς.

1.7.1 Επίλυση προβλημάτων Ακέραιου Γραμμικού Προγραμματισμού

Αντίθετα με τον Γραμμικό Προγραμματισμό, δεν υπάρχει μια καθιερωμένη μέθοδος για την επίλυση προβλημάτων Ακέραιου Γραμμικού Προγραμματισμού. Τα προβλήματα είναι συνδυαστικά προβλήματα βελτιστοποίησης. Μπορούν να επιλυθούν απαριθμητικά, δηλαδή να εξεταστούν για κάθε πιθανό συνδυασμό ακέραιων μεταβλητών και να επιλεχθεί η καλύτερη λύση (η μέθοδος της απαρίθμησης όμως είναι υπολογιστικά χρονοβόρα ή ανέφικτη για τα περισσότερα πραγματικά προβλήματα).

Η πιο διαδεδομένες μέθοδοι είναι του τύπου Branch and Bound [34]:

- Προϋπόθεση να είναι φραγμένες οι ακέραιες μεταβλητές.
- Έμμεση απαρίθμηση όλων των δυνατών ακέραιων λύσεων που επιδέχεται το πρόβλημα.

Υπάρχουν και άλλες μέθοδοι επίλυσης προβλημάτων Ακέραιου Γραμμικού Προγραμματισμού όπως cutting planes method και άλλα πολλά.

Ένας φαινομενικά απλός τρόπος για τον υπολογισμό της βέλτιστης λύσης σε ένα πρόβλημα ακέραιου προγραμματισμού είναι η επίλυση του προβλήματος μέσω της γνωστής τεχνικής γραμμικού

προγραμματισμού και η περικοπή ή η στρογγυλοποίηση της μη ακέραιης λύσης για να ικανοποιηθούν οι συνθήκες πληρότητας. Στην περίπτωση που χρησιμοποιηθεί αποκοπή υπάρχει μεγάλη περίπτωση η λύση να είναι δυνατή αλλά όχι η βέλτιστη ενώ στην περίπτωση στρογγύλευσης η λύση να μην είναι δυνατή [1].

Στην πράξη, τα προβλήματα ακέραιου προγραμματισμού είναι πιο περίπλοκα από τα προβλήματα γραμμικού προγραμματισμού. Εάν και είναι γεγονός ότι οι ακέραιες λύσεις είναι πολύ λιγότερες από τις κλασματικές.

1.8 Το μαθηματικό μοντέλο του προβλήματος Γραμμικού Προγραμματισμού

Το μοντέλο του γραμμικού προγραμματισμού περιλαμβάνει:

- **Τις μεταβλητές απόφασης (decision variables)**. Η μεταβλητή απόφασης ορίζει ξεκάθαρα την ποσότητα που πρέπει να υπολογίσουμε και θα γίνει η λύση στο πρόβλημά μας. Ο ορισμός των μεταβλητών απόφασης θεωρείται ένα από τα πιο δύσκολα μέρη της διαδικασίας μοντελοποίησης των προβλημάτων, που οδηγεί σε ανεπαρκείς ή μη πρακτικές λύσεις. Συνήθως, αντιπροσωπεύουμε τις μεταβλητές με το γράμμα x και τον δείκτη j έτσι ώστε να είναι αρκετά διακριτές.
- **Την αντικειμενική συνάρτηση (objective function)**. Η αντικειμενική συνάρτηση επιχειρεί να εκφράσει τον στόχο - την επιδίωξη του προβλήματος - με μετρήσιμο τρόπο και τελικά να βελτιστοποιήσει (μεγιστοποιήσει ή ελαχιστοποιήσει) τα επιθυμητά κριτήρια απόδοσης (ελαχιστοποίηση κόστους, μεγιστοποίηση κέρδους κ.λπ).
- **Τους περιορισμούς (constraints)**. Οι περιορισμοί είναι εξισώσεις ή ανισότητες που αντιστοιχούν σε περιορισμούς σε ένα φυσικό πρόβλημα. Πρέπει να δούμε τους παράγοντες του προβλήματος που απαιτούν κάποιο περιορισμό στις τιμές τους και δεν πρέπει να ξεχνάμε τον περιορισμό στη μη αρνητικότητα των τιμών των μεταβλητών, ο οποίος προκύπτει από την φυσική ερμηνεία του προβλήματος.

Στη μαθηματική γλώσσα, ο γραμμικός προγραμματισμός είναι ένα μαθηματικό μοντέλο στο οποίο επιχειρείται μια βελτιστοποίηση (κριτήριο βελτιστοποίησης) για άγνωστες πραγματικές μεταβλητές των οποίων το εύρος τιμών οριοθετείτε έμμεσα από μια συνάρτηση γραμμικών περιορισμών (ανισότητα) αυτών των μεταβλητών. Οι άγνωστες μεταβλητές καθορίζουν τα αντικείμενα απόφασης του προβλήματος (μοντέλο) και ονομάζονται μεταβλητές απόφασης για το σκοπό αυτό (decision variables).

Η θεωρία του γραμμικού προγραμματισμού μέχρι τις αρχές της δεκαετίας του 1970 αναπτύχθηκε σε μια ενιαία μέθοδο βελτιστοποίησης με κριτήριο απόφασης που ονομάζεται αντικειμενική συνάρτηση (objective function). Όμως, η πολυπλοκότητα των συστημάτων απόφασης καθώς και οι συνθήκες ανταγωνιστικότητας κάτω από τις οποίες παίρνονται οι αποφάσεις καθιστούν τη προσέγγιση αυτή κάθε άλλο παρά ρεαλιστική. Γι' αυτό και αναπτύχθηκε η θεωρία του γραμμικού προγραμματισμού με πολλαπλά́ κριτήρια απόφασης (πολυκριτήριος γραμμικός προγραμματισμός) Από την άλλη, αυτή η γενίκευση δεν μπορεί να επιτευχθεί χωρίς τα αποτελέσματα που έφερε το κλασικό μονοκριτήριο γραμμικού προγραμματισμού [1].

1.8.1 Κανονική Μορφή Προβλημάτων Γραμμικού Προγραμματισμού

Η γενική μορφή ενός προβλήματος γραμμικού προγραμματισμού είναι ο προσδιορισμός των τιμών των μεταβλητών $\chi_1, \dots, \chi_n$ που μεγιστοποιούν ή ελαχιστοποιούν μια γραμμική συνάρτηση:

$$z = c_1 x_1 + c_2 x_2 + c_n x_n$$

και συγχρόνως πρέπει να ικανοποιούνται κάποιες συνθήκες της μορφής [1], [2]:

$$a_{11} x_1 + a_{12} + \cdots + a_{1n} x_n \leq, =, \geq b_1$$

$$a_{21}x_1 + a_{22} + \cdots + a_{2n}\,x_n \leq, =, \geq b_2$$
$$a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \leq, =, \geq b_m$$

$$\text{Επιπρόσθετα, ισχύει ότι : } \chi_1, \chi_2, \dots, \chi_n \geq 0$$

Συνοπτικά ένα LPP μπορεί να γραφεί ως εξής:

$$\max \text{ή } min\ z = \sum_{j=1}^{n} c_j x_j$$

υπό τους περιορισμούς:

$$\sum_{j=1}^{n} a_{ij}x_j \geq, =, \leq b_i\ \mu\varepsilon\ i = 1, \dots, m$$
$$x_j \geq 0\ \gamma\iota\alpha\ j = 1, \dots, n$$

Σημειώνουμε ότι εφόσον η ταυτότητα min f(z)=-max (-f(z)) ισχύει για κάθε γραμμική συνάρτηση f(z), οποιοδήποτε πρόβλημα ελαχιστοποίησης μπορεί να μετατραπεί σε ισοδύναμο πρόβλημα μεγιστοποίησης.

Η γραμμική συνάρτηση f(z) που θέλουμε να μεγιστοποιήσουμε ονομάζεται αντικειμενική συνάρτηση. Οι παραπάνω γραμμικές εξισώσεις ή ανισότητες ονομάζονται συνθήκες ή περιορισμοί του προβλήματος σχεδιασμού. Η περιοχή που ορίζεται από αυτούς τους περιορισμούς ονομάζεται εφικτή περιοχή (feasible area) και κάθε λύση που ικανοποιεί όλους τους περιορισμούς ονομάζεται εφικτή λύση (feasible solution). Η βέλτιστη λύση είναι μια εφικτή λύση που μεγιστοποιεί ή ελαχιστοποιεί την αντικειμενική συνάρτηση σε ολόκληρη την εφικτή περιοχή. Οι συντελεστές c_j αναφέρονται και ως συντελεστές κέρδους (ή κόστους αντίστοιχα εάν αναφερόμαστε σε πρόβλημα ελαχιστοποίησης) ενώ τα a_{ij}, b_i είναι οι παράμετροι (parameters) του προβλήματος.

Εάν όλοι οι περιορισμοί είναι εξισώσεις ή ανισότητες ταυτόχρονα, τότε το πρόβλημα μπορεί να γραφεί συνοπτικά σε μορφή πίνακα [1]:

$$w = \max(c^T x)$$
$$Ax \leq, =, \geq b$$
$$x \geq 0$$

όπου:

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ xn \end{pmatrix},\ c = \begin{pmatrix} c_1 \\ c_2 \\ \dots \\ cn \end{pmatrix},\ b = \begin{pmatrix} b_1 \\ bz \\ \dots \\ bn \end{pmatrix}$$

και:

$$A = \begin{pmatrix} a_{11}a_{12} \dots a_{1n} \\ a_{21}a_{22} \dots a_{2n} \\ \dots \\ a_{n1}a_{n2} \dots a_{nn} \end{pmatrix}$$

Γενικά, οποιοδήποτε πρόβλημα γραμμικού προγραμματισμού μπορεί να μετατραπεί σε κανονική μορφή χρησιμοποιώντας στοιχειώδεις μετασχηματισμούς. Ακολουθούν περιπτώσεις που απαιτούν τροποποίηση [2]:

1. Ελαχιστοποίηση: Όταν ζητείται ο προσδιορισμός του ελαχίστου της αντικειμενικής συνάρτησης $f(x) = c_1x_1 + c_2x_2 + \cdots + c_nx_n$ τότε θέτουμε $g(x) = -f(x)$ και μεγιστοποιούμε τη συνάρτηση $g(x)$. Το ελάχιστο της $f(x)$ δίνεται από τη σχέση: $\min f = -\max g$
2. Αρνητικοί σταθεροί όροι: Τους μετατρέπουμε σε θετικούς πολλαπλασιάζοντας και τα δύο μέλη των αντίστοιχων περιορισμών με (-1).
3. Ανισώσεις στους περιορισμούς: Αυτές μπορούν να αναχθούν σε εξισώσεις με την εισαγωγή νέων μη αρνητικών μεταβλητών που ονομάζονται περιθώριες μεταβλητές. Έτσι οι αρχικοί περιορισμοί:

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \leq b_i$$
$$a_{j1}x_1 + a_{j2}x_2 + \cdots + a_{jn}x_n \geq b_j$$

είναι ισοδύναμοι με τους περιορισμούς:

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n + x_r = b_i \,, x_r \geq 0$$
$$a_{j1}x_1 + a_{j2}x_2 + \cdots + a_{jn}x_n - x_s = b_j \,, x_s \geq 0$$

4. Περιορισμός πρόσημου: Αν η μεταβλητή x_i είναι μη αρνητική θέτουμε $x_i = -x_i'$ όπου $x_i' > 0$, ενώ αν η μεταβλητή x_j δεν υπόκειται σε περιορισμό για πρόσημο , τότε θέτουμε $x_j = x_j' - x_j''$ με $x_j' \,, x_j'' \geq 0$.

1.8.2 Επίλυση ενός μοντέλου Γραμμικού Προγραμματισμού

Η επίλυση ενός γραμμικού προγραμματισμού συνήθως απαιτεί υπολογιστή και κατάλληλο λογισμικό. Όταν όμως ένα πρόβλημα γραμμικού προγραμματισμού είναι δισδιάστατο, δηλαδή υπάρχουν 2 μεταβλητές απόφασης ή το πολύ 3D μπορεί να λυθεί γραφικά. Για το σκοπό αυτό, αρκεί να σχεδιάσουμε προσεκτικά το σύνολο των περιορισμών του προβλήματος ώστε να οριοθετήσουμε το σύνολο των πιθανών λύσεων Α και να ορίσουμε την έννοια της βελτιστοποίησης της αντικειμενικής συνάρτησης. Η εποπτεία διαδραματίζει βασικό ρόλο σε αυτή την προσέγγιση [1]:

- Ανεξάρτητα από τους περιορισμούς, ο σχεδιασμός εκτελείται έτσι ώστε το σύνολο Α των πιθανών λύσεων να ορίζεται πλήρως.
- Σχεδίαση αντικειμενικής συνάρτησης Z για μια συγκεκριμένη τιμή του Z και καθορισμός της έννοιας βελτιστοποίησης.
- Η παράλληλη μετατόπιση της αντικειμενικής συνάρτησης με την έννοια της βελτιστοποίησης, αυξάνοντας σταδιακά την τιμή z μέχρι να καθοριστούν μία ή περισσότερες βέλτιστες λύσεις.

Εάν υπάρχουν περισσότερες από τρεις μεταβλητές, τότε η καταλληλότερη μέθοδος για την επίλυση του προβλήματος είναι η μέθοδος simplex, η οποία είναι μια γενική διαδικασία για την επίλυση προβλημάτων γραμμικού προγραμματισμού με περισσότερες από δύο μεταβλητές. Είναι μια συστηματική διαδικασία, ένας αλγόριθμος, που επαναλαμβάνεται μέχρι να βρεθεί το επιθυμητό αποτέλεσμα.

Για να εφαρμοστεί η μέθοδος simplex, το πρόβλημα γραμμικού προγραμματισμού πρέπει να γραφτεί σε επίσημη μορφή, δηλαδή της μορφής:

$$w = \max(c^\tau x)$$
$$Ax = b$$
$$x \geq 0$$

Το πρώτο βήμα είναι η μετατροπή της ανισότητας σε ισότητα. Αυτό επιτυγχάνεται με την εισαγωγή μεταβλητών χαλάρωσης ή βοηθητικών μεταβλητών (slack variables). Η εισαγωγή αυτών των μεταβλητών δεν αλλάζει τις ιδιότητες των περιορισμών ή της αντικειμενικής συνάρτησης. Αυτές οι μεταβλητές εισάγονται στην αντικειμενική συνάρτηση με μηδενικούς συντελεστές [1].

Κεφάλαιο 2ο: Σχεδιασμός ωρολόγιου προγράμματος Εκπαιδευτικών Ιδρυμάτων

2.1 Εισαγωγή

Ο χρονοπρογραμματισμός (scheduling) είναι ένα πρόβλημα που αντιμετωπίζουν συχνά τα εκπαιδευτικά ιδρύματα (σχολεία και πανεπιστήμια) και αναφέρεται στην εύρεση του εβδομαδιαίου προγράμματος μαθημάτων ή εξετάσεων. Το αναλυτικό σχολικό πρόγραμμα διατυπώνεται τουλάχιστον μία φορά το χρόνο ενώ το αναλυτικό πρόγραμμα και το πρόγραμμα εξετάσεων για τα πανεπιστημιακά τμήματα μαθημάτων συνήθως διατυπώνονται περισσότερες από φορές το χρόνο αντίστοιχα. Ακόμα και σήμερα, σε ορισμένες σχολές και πανεπιστημιακές σχολές, η δημιουργία των προγραμμάτων τους δεν γίνεται με υπολογιστές, αλλά με παραδοσιακές μεθόδους (από ανθρώπους). Ωστόσο, τα τελευταία χρόνια έχουν γίνει σημαντικές προσπάθειες για την αυτοματοποιημένη δημιουργία τους [36].

Το πρόβλημα προγραμματισμού χρονοδιαγράμματος (Timetabling) των εκπαιδευτικών ιδρυμάτων είναι ένα πρόβλημα προγραμματισμού χρόνου, το οποίο συνίσταται στον προγραμματισμό μιας σειράς διαλέξεων, σεμιναρίων ή εξετάσεων εντός καθορισμένου χρόνου ώστε οι καθηγητές και οι φοιτητές του ιδρύματος να ανταποκρίνονται σε πολυάριθμους περιορισμούς. Με απλά λόγια, αναφέρεται στον καθορισμό του εβδομαδιαίου προγράμματος διδασκαλίας και εξετάσεων των εκπαιδευτικών ιδρυμάτων. Ο ρόλος της εβδομαδιαίας διδασκαλίας ή του προγραμματισμού των εξετάσεων μπορεί να μην φαίνεται τόσο σημαντικός όσο είναι στην πραγματικότητα. Ωστόσο, αν αναλογιστεί κανείς το σύνολο των ανθρώπων πάνω στους οποίους καθορίζεται ο χρόνος του και τους πολύ συγκεκριμένους υλικούς πόρους που πρέπει να διαχειριστεί, μπορεί κάλλιστα να συμπεράνει ότι είναι πολύ σημαντικό για την ομαλή, παραγωγική και αποδοτική λειτουργία ενός επιχειρηματικού παράγοντα [37].

Το ακαδημαϊκό ωρολόγιο πρόγραμμα είναι ένα πολύ γνωστό πρόβλημα προγραμματισμού που συζητείται στην επιχειρηματική έρευνα. Τα μοντέλα ακέραιου προγραμματισμού (IP) ανήκουν στην οικογένεια των προβλημάτων συνδυαστικής βελτιστοποίησης NP-Hard προβλημάτων συνδυαστικής βελτιστοποίησης που είναι δύσκολο να επιλυθούν. Για τέτοια προβλήματα, ο χρόνος που απαιτείται για την εύρεση της βέλτιστης λύσης αυξάνεται με γοργούς ρυθμούς αναλόγως το μέγεθος του προβλήματος. Ωστόσο, λόγω των γρήγορων βελτιώσεων στην απόδοση του αλγορίθμου και στην υπολογιστική ισχύ, είναι πλέον διαθέσιμα αποτελεσματικά εργαλεία για την επίλυση ακόμη και προβλημάτων ωρολόγιού προγράμματος μεγάλης κλίμακας [38], [39] αλλά η εφαρμογή αυτής της μεθόδου βελτιστοποίησης δεν είναι ακόμα πολύ διαδεδομένη.

Το εκπαιδευτικό ωρολόγιο πρόγραμμα είναι ένα δύσκολο έργο που τα εκπαιδευτικά ιδρύματα πρέπει να εκτελούν τακτικά. Περιλαμβάνει την ανάθεση μαθημάτων μεταξύ φοιτητών και διδασκόντων σε ώρες και αίθουσες, αποφεύγοντας όσο το δυνατόν περισσότερο τις συγκρούσεις. Ανάλογα με το είδος των ιδρυμάτων (σχολείο ή πανεπιστήμιο) και το είδος των εκδηλώσεων (μάθημα ή εξέταση), αποτελείται κυρίως από τρεις κατηγορίες: ωρολόγιο πρόγραμμα σχολείου, ωρολόγιο πρόγραμμα πανεπιστημιακών εξετάσεων και ωρολόγιο πρόγραμμα πανεπιστημιακών μαθημάτων [40].

Συνήθως το έργο της ανάπτυξης χρονοδιαγραμμάτων εκπληρώνεται με την επαναχρησιμοποίηση των χρονοδιαγραμμάτων των προηγούμενων ετών και την δημιουργία ορισμένων τροποποιήσεων με το χέρι για την αντιμετώπιση των νέων απαιτήσεων. Ωστόσο, αυτό είναι χρονοβόρο και λιγότερο ευέλικτο σε σύγκριση με το αυτοματοποιημένο ωρολόγιο πρόγραμμα, ιδίως όταν υπάρχει μεγάλη αλλαγή στις

απαιτήσεις. Για το λόγο αυτό, έχουν διεξαχθεί πολλές έρευνες σχετικά με το αυτοματοποιημένο ωρολόγιο πρόγραμμα κατά τη διάρκεια του τις τελευταίες δεκαετίες [41].

2.2 Κύριοι Συντελεστές του Προβλήματος

Σύμφωνα με τη βιβλιογραφία, υπάρχουν πέντε βασικά στοιχεία του ρολογιού του προγράμματος σπουδών στα εκπαιδευτικά ιδρύματα [42]:

- το εκπαιδευτικό προσωπικό του ιδρύματος
- οι φοιτητές
- οι διαθέσιμες αίθουσες
- οι ώρες διδασκαλίας
- ο χρόνος

Οι ανθρώπινοι παράγοντες είναι ένας από τους σημαντικότερους παράγοντες που συμβάλλουν σε αυτό το πρόβλημα. Αν δούμε το θέμα των χρονοδιαγραμμάτων από την οπτική των εκπαιδευτικών, δηλαδή των δασκάλων, μπορούμε εύκολα να δούμε την επιθυμία τους για μαθήματα που καλύπτουν τις δικές τους ανάγκες, οι οποίες φυσικά είναι συχνά διαφορετικές για κάθε άτομο. Οι προσδοκίες τους για το πρόγραμμα μπορεί να περιλαμβάνουν λίγο ελεύθερο χρόνο μέσα στην εβδομάδα ώστε να μπορούν να ταξιδέψουν για επαγγελματικές υποχρεώσεις ή προσωπικούς λόγους εντός και εκτός Ελλάδας. Επιπλέον, μερικοί άνθρωποι τείνουν να εκφράζουν ιδιαίτερες προτιμήσεις για το χρόνο διδασκαλίας και ακόμη και τις αίθουσες διδασκαλίας λόγω της φύσης του μαθήματος και του εξοπλισμού που απαιτείται.

Από την πλευρά του μαθητή, οι απαιτήσεις είναι σίγουρα πολύ χαμηλότερες, αλλά σίγουρα θα εκφράσουν τις προτιμήσεις τους ως προς τις ημερομηνίες των μαθημάτων και κυρίως τις ημερομηνίες και τις ώρες των εξετάσεων, καθώς γενικά δεν θέλουν πρωινά ή βραδινά μαθήματα. Ωστόσο, μια πολύ κρίσιμη ανησυχία για τους φοιτητές είναι ότι η χρονική στιγμή των μαθημάτων ή των εξετάσεων πρέπει να λαμβάνει υπόψη τα επικαλυπτόμενα ζητήματα. Αυτή η ερώτηση σχετίζεται με μια κατάσταση όπου ένας μαθητής πρέπει να παρακολουθήσει δύο μαθήματα που πραγματοποιούνται την ίδια ώρα την ίδια μέρα. Οι επικαλύψεις πρέπει να εξαιρεθούν από το χρονοδιάγραμμα καθώς δημιουργούν ανισότητες μεταξύ των μαθητών.

Όσον αφορά τις αίθουσες διδασκαλίας και τα εργαστήρια, υπάρχουν συχνά ζητήματα διαθεσιμότητας. Η έλλειψη διαθεσιμότητας εκδηλώνεται κυρίως σε αίθουσες με ειδικό εξοπλισμό, αφού στην ελληνική πραγματικότητα δεν επαρκούν για να καλύψουν τις ανάγκες καθηγητών και μαθητών. Ανεξάρτητα από αυτό, τα ωράρια πρέπει να τοποθετούν σωστά τις τάξεις και τους δασκάλους στις κατάλληλες αίθουσες, ώστε να μπορούν να διδάσκουν όσο το δυνατόν πιο αποτελεσματικά.

Τέλος, το χρονοδιάγραμμα του εκπαιδευτικού ιδρύματος πρέπει να διασφαλίζει τη σωστή κατανομή του χρόνου διδασκαλίας των μαθημάτων εντός του εβδομαδιαίου προγράμματος σπουδών και να ακολουθεί πάντα το πρόγραμμα σπουδών για τον καθορισμό του απαιτούμενου εβδομαδιαίου χρόνου διάλεξης και σεμιναρίου [37].

2.3 Κατηγορίες Προβλημάτων

Πολλές διαφορετικές παραλλαγές του προβλήματος του προγραμματισμού έχουν αναφερθεί στη βιβλιογραφία έως σήμερα. Οι διαφορές που προκύπτουν έγκεινται ουσιαστικά στους διαφορετικούς τύπους ιδρυμάτων που εξετάζουμε (πανεπιστήμια ή σχολεία) και στους τύπους περιορισμών που τίθενται σε κάθε περίπτωση. Μια γενική ταξινόμηση δίνεται παρακάτω και προτείνονται τρεις κύριες κατηγορίες προβλημάτων [40]:

- **Σχολικό Ωρολόγιο Πρόγραμμα (School Timetabling)**: Ένα εβδομαδιαίο ωράριο για όλες τις τάξεις του σχολείου για να αποφευχθεί η ανάθεση ενός καθηγητή σε δύο τάξεις ταυτόχρονα και η ανάθεση δύο εκπαιδευτικών σε μία τάξη ταυτόχρονα.
- **Χρονοδιάγραμμα Πανεπιστημιακών Μαθημάτων (Course Timetabling)**: Ένα εβδομαδιαίο χρονοδιάγραμμα όλων των διαλέξεων και δραστηριοτήτων του μαθήματος, ελαχιστοποιώντας την επικάλυψη διαλέξεων και δραστηριοτήτων που μοιράζονται οι φοιτητές.
- **Χρονοδιάγραμμα Εξετάσεων (Examination Timetabling)**: Τακτοποίηση των εξετάσεων μαθημάτων για την αποφυγή αλληλεπικαλυπτόμενων εξετάσεων για μαθητές που μοιράζονται μαθήματα και για ομοιόμορφο προγραμματισμό των εξετάσεων εντός του διαθέσιμου χρονικού πλαισίου.

Βέβαια, η παραπάνω κατηγοριοποίηση δεν είναι απαραίτητα αυστηρή, σε ορισμένες περιπτώσεις μπορεί να βρίσκεται «ανάμεσα» στις δύο κατηγορίες. Για παράδειγμα, εάν ένα σχολείο παρέχει στους μαθητές μεγάλη ελευθερία κινήσεων στην επιλογή μαθημάτων, αυτό θα θεωρείται συχνά ως πρόβλημα οργάνωσης διαλέξεων και μαθημάτων παρά ως πρόβλημα του σχολείου.

2.3.1 Ωρολόγιο πρόγραμμα ενός Πανεπιστημίου

Τα ωρολόγια προγράμματα των πανεπιστημίων συνήθως αναπτύσσονται από τους διαχειριστές με βάση τα χρονοδιαγράμματα των προηγούμενων ετών και τροποποιούνται για να προσαρμόζονται στις νέες εξελίξεις. Μερικές φορές αυτά τα χρονοδιαγράμματα αναθεωρούνται πολλές φορές για να αποφευχθούν συγκρούσεις και να μην δημιουργηθούν δυσκολίες για το προσωπικό και τους μαθητές. Οι προτιμήσεις καθηγητών και μαθητών σπάνια λαμβάνονται υπόψη κατά την προετοιμασία των προγραμμάτων μαθημάτων. Μερικές φορές ο λόγος που οι μαθητές δεν συμμετέχουν σε αθλητικές και άλλες εξωσχολικές δραστηριότητες οφείλεται στις πολλές ώρες παρουσίας σε μαθήματα μέσα στην μέρα. Δεδομένου του μεγάλου αριθμού μαθημάτων, της ποικιλίας των συνδυασμών μαθημάτων και των περιορισμένων πόρων, είναι λογικό να αντιμετωπίζονται αυτά τα ζητήματα με ένα μη αυτόματο τρόπο διεξαγωγής ενός ωρολόγιού προγράμματος. Η έλλειψη κατανόησης των μεθόδων επιστημονικού σχεδιασμού, η θεσμική πολιτική και η κακή διαχείριση μπορούν να οδηγήσουν σε αναποτελεσματικό σχεδιασμό των χρονοδιαγραμμάτων των πανεπιστημιακών μαθημάτων [43]. Ismailova κ.ά.[44] εφάρμοσαν ένα μοντέλο πολλαπλών στόχων IP που λαμβάνει υπόψη τη διαχείριση και τις προτιμήσεις των εκπαιδευτικών για την επίλυση προβλημάτων προγραμματισμού μαθήματων των εκπαιδευτικών. Το μοντέλο λαμβάνει υπόψη την εργασία των εκπαιδευτικών-μαθητών. Οι Broek κ.ά. [45] πρότειναν ένα μοντέλο IP για την ανάπτυξη χρονοδιαγραμμάτων μαθημάτων με την ανάθεση των μαθητών σε μαθήματα με βάση τις προτιμήσεις τους. Οι Miranda κ.ά. [46] πρότειναν μια μέθοδο προγραμματισμού βάσει δικτύου (για λήψη δεδομένων από χρήστες στο διαδίκτυο) που χρησιμοποιεί IP για να δημιουργήσει ένα βέλτιστο χρονοδιάγραμμα. Οι Cacchiani κ.ά. [38] πρότειναν μια νέα μέθοδο για τον υπολογισμό ενός κατώτερου ορίου για ένα πρόβλημα προγραμματισμού μαθημάτων που βασίζεται σε μαθήματα με στόχο τη βελτιστοποίηση της κατανομής των διαλέξεων σε εβδομαδιαίες αίθουσες διδασκαλίας και χρονοθυρίδες.

Οι Vermuyten κ.ά. [47] πρότειναν ένα μοντέλο IP δύο σταδίων για την ελαχιστοποίηση της ροής των μαθητών μέσω των διαδρόμων και των κλιμακοστασίων, ενώ πληροί άλλες απαιτήσεις ωρολογίου προγράμματος.

Οι Daskalaki κ.ά. [42] πρότειναν μια ολοκληρωμένη διατύπωση IP για τον προγραμματισμό πανεπιστημιακών μαθημάτων, η οποία περιλαμβάνει ένα ευρύ φάσμα εφαρμοστέων περιορισμών όπως συγκρούσεις, πληρότητα, διαδοχικά μαθήματα, επαναλαμβανόμενα μαθήματα. Οι Bakir και Akshop

[48] χρησιμοποίησαν επίσης ένα μοντέλο IP παρόμοιο με τους [42]. Επίλυση σύνθετων προβλημάτων προγραμματισμού.

Ο Bardadym χωρίζει τα προβλήματα προγραμματισμού του κολλεγίου σε πέντε κατηγορίες:

- **Ωρολόγιο πρόγραμμα καθηγητών**: Αναθέτει στους κατάλληλους καθηγητές τα μαθήματα.
- **Ωρολόγιο πρόγραμμα τμημάτων-καθηγητών**: Τα μαθήματα καθορίζονται με βάση το ωρολόγιο πρόγραμμα του φοιτητή.
- **Ωρολόγιο πρόγραμμα μαθημάτων**: Τα μαθήματα κατανέμονται σύμφωνα με το πρόγραμμα του κάθε μαθητή.
- **Ωρολόγιο πρόγραμμα εξετάσεων**: Ανάθεση εξετάσεων σε μαθητές για να μην δίνουν δυο εξετάσεις ταυτόχρονα.
- **Ανάθεση αιθουσών**: Αφού οριστούν τμήματα-καθηγητές, αντιστοιχίζονται σε συγκεκριμένες αίθουσες.

Φυσικά και η ταξινόμηση σύμφωνα με τον Bardadym δεν είναι κανόνας, αφού κάθε εκπαιδευτικό ίδρυμα έχει τις δικές του ανάγκες και ιδιαιτερότητες.

2.4 Ανάλυση Περιορισμών (Σκληροί – Μαλακοί)

Για καλύτερη κατανόηση, περιγράφεται πρώτα η διαδικασία επίλυσης του προβλήματος του χρονοδιαγράμματος του πανεπιστημίου μέσω προγραμματισμού ακέραιων αριθμών. Αρχικά, απαριθμούνται οι κανόνες και οι πολιτικές του πανεπιστημίου, καθώς και οι ανάγκες των ενδιαφερομένων που εμπλέκονται στο χρονοδιάγραμμα (όπως καθηγητές και φοιτητές). Αυτές οι απαιτήσεις χωρίζονται σε δύο κατηγορίες: σκληρούς περιορισμούς και μαλακούς περιορισμούς [49]. Οι σκληροί περιορισμοί δεν μπορούν να παραβιαστούν για να δοθεί μια εφικτή λύση. Για παράδειγμα, οι εκπαιδευτές δεν μπορούν να δώσουν περισσότερες από μία διαλέξεις σε μια δεδομένη χρονική περίοδο. Οι μαλακοί περιορισμοί θα πρέπει να ικανοποιούνται όσο το δυνατόν περισσότερο για να διασφαλιστεί η ποιότητα των ωρολογίων του προγράμματος, όπως οι προτιμήσεις των εκπαιδευτικών [50]. Οι σκληροί περιορισμοί του προβλήματος του ωρολόγιού προγράμματος στη συνέχεια μετατρέπονται σε περιορισμούς σε ένα μοντέλο μαθηματικού ακέραιου προγραμματισμού, ενώ οι μαλακοί περιορισμοί συχνά ενσωματώνονται στην αντικειμενική συνάρτηση σε αυτό το μοντέλο [50].

Οι περιορισμοί που επιβάλλονται στο πρόβλημα βελτιστοποίησης χωρίζονται σε «σκληρούς» (hard) και «μαλακούς» (soft) περιορισμούς όπως αναφέρθηκε και παραπάνω. Η κατηγοριοποίηση αυτή έγκειται στο κατά πόσο ένας περιορισμός επιτρέπεται ή όχι να παραβιαστεί στη λύση που προτείνεται.

Πιο συγκεκριμένα υπάρχουν οι «σκληροί» περιορισμοί, οι οποίοι αποτελούν αυστηρές και συγκεκριμένες απαιτήσεις, που δεν πρέπει σε καμία περίπτωση να παραβιάζονται κατά την επίλυση ενός προβλήματος. Οι ειδικοί περιορισμοί στα ωράρια για τα εκπαιδευτικά ιδρύματα εκφράζουν κυρίως δύο συγκεκριμένες απαιτήσεις [37]:

- Κανείς δεν μπορεί να βρίσκεται σε δύο μέρη ταυτόχρονα. Οπότε, κανένας φοιτητής δεν μπορεί να παρακολουθήσει δύο διαλέξεις ταυτόχρονα, και κανένας καθηγητής δεν μπορεί να διδάξει σε δύο διαφορετικές αίθουσες ταυτόχρονα.
- Κάθε διάλεξη πραγματοποιείται σε συγκεκριμένη αίθουσα. Οι αίθουσες πρέπει να είναι διαθέσιμες για να καλύπτουν τις ανάγκες των μαθητών σε όλα τα μαθήματα που έχουν προγραμματιστεί ταυτόχρονα στο χρονοδιάγραμμα.

Υπάρχουν όμως και οι «μαλακοί» περιορισμοί οι οποίοι εκφράζουν ιδανικά και όχι απόλυτα πρότυπα, πράγμα που σημαίνει ότι είναι δυνατή η παραβίασή τους, σίγουρα στον μικρότερο δυνατό βαθμό. Κάθε εκπαιδευτικό ίδρυμα περιγράφει διαφορετικά τους περιορισμούς αυτούς. Οι «μαλακοί» περιορισμοί μπορούν επίσης να ονομαστούν ποιοτικοί κανόνες, αφού ο βαθμός παραβίασής τους αντιπροσωπεύεται

από τον βαθμό στον οποίο ελαχιστοποιείται/μεγιστοποιείται η τιμή της αντικειμενικής συνάρτησης του προβλήματος. Κάποιοι από τους «μαλακούς» περιορισμούς είναι οι εξής [37]:

- Οι προτιμήσεις των καθηγητών
- Εξάλειψη του χρόνου αδράνειας στα προγράμματα εργασίας καθηγητών ή φοιτητών
- Πολλαπλές ώρες αποστάσεις στο ίδιο τμήμα
- Ελάχιστος χρόνος μεταφοράς καθηγητών
- Ελάχιστος χρόνος μεταφοράς τμημάτων του ίδιου εξαμήνου

Αν και οι περιορισμοί διαφέρουν από άρθρο σε άρθρο, υπάρχουν ορισμένοι σημαντικοί τύποι σκληρών περιορισμών που δεν πρέπει να παραβιάζονται, διαφορετικά το χρονοδιάγραμμα δεν θα επιτευχθεί [42].Τέτοιοι περιορισμοί αναφέρονται σε:

- **Αποφυγή συγκρούσεων σε ένα χρονοδιάγραμμα**. Οι συγκρούσεις μπορεί να προκύψουν, για παράδειγμα, όταν έχουν προγραμματιστεί περισσότερες από μία διαλέξεις στην ίδια αίθουσα την ίδια στιγμή ή όταν περισσότερες από μία διαλέξεις ανατίθενται ταυτόχρονα σε έναν δάσκαλο ή μαθητή.
- **Πληρότητα χρονοδιαγράμματος.** Όλα τα μαθήματα πρέπει να προγραμματιστούν τη σωστή ώρα και στην κατάλληλη αίθουσα.

Οι μεν σκληροί (hard) επιβάλλονται αυστηρά και απαιτούν άμεση ικανοποίηση, οι δε μαλακοί (soft) είναι επιθυμητοί αλλά όχι απολύτως απαραίτητα. Για παράδειγμα, ένας σκληρός περιορισμός είναι ότι κανένας καθηγητής δεν μπορεί να δώσει δύο διαλέξεις ταυτόχρονα, ενώ ένας μαλακός περιορισμός είναι ότι ο καθηγητής θέλει να ξεκινήσει τη διάλεξή του μετά τις 11:00 π.μ.

Εκτός από τις παραδοχές και τις προϋποθέσεις που αναφέρθηκαν μέχρι τώρα, θα πρέπει να εφαρμοστούν ορισμένοι περιορισμοί, η εκπλήρωση των οποίων βοηθά στην εξεύρεση λύσης στο πρόβλημα του προγραμματισμού. Για τα μαθήματα οι περιορισμοί που πρέπει να ικανοποιούνται είναι οι παρακάτω [36]:

- Τα μαθήματα διδάσκονται πέντε ημέρες την εβδομάδα, από Δευτέρα έως Παρασκευή, εκτός Σαββάτου και Κυριακής.
- Το θεωρητικό μέρος του μαθήματος πρέπει να διδάσκεται στην κατάλληλη αίθουσα και το εργαστηριακό στο αντίστοιχο εργαστήριο.
- Δύο μαθήματα σε ένα εξάμηνο δεν επιτρέπεται να αλληλοκαλύπτονται, δηλαδή πρέπει να διδάσκονται την ίδια ημέρα, την ίδια ώρα, στην ίδια αίθουσα ή εργαστήριο.
- Για κάθε μάθημα, ο υπεύθυνος καθηγητής (ή καθηγητές) πρέπει να είναι παρών κατά τις ώρες διδασκαλίας του μαθήματος.
- Για τα μαθήματα, η αίθουσα ή το κατάλληλο εργαστήριο πρέπει να είναι διαθέσιμο κατά τις ώρες διδασκαλίας του μαθήματος, δηλαδή να μην έχει προγραμματιστεί σε άλλα μαθήματα στα ίδια ή διαφορετικά εξάμηνα.

2.5 Στόχοι βελτιστοποίησης του προβλήματος του ωρολόγιου προγράμματος

Οι στόχοι επικεντρώνονται στην ομαλή και αποτελεσματική λειτουργία του προγράμματος, στην ικανοποίηση των καθηγητών από το συγκεκριμένο πρόγραμμα, στην ορθολογική χρήση των υλικών πόρων του ιδρύματος και τελικά στη θετική αποδοχή του προγράμματος από τους φοιτητές. Ορισμένοι στόχοι βελτιστοποίησης παρατίθενται παρακάτω [37].

- Ελαχιστοποίηση κενών ωρών των φοιτητών
- Ο χρόνος διδασκαλίας κάθε εκπαιδευτικού κατανέμεται ισότιμα
- Ελαχιστοποίηση του χρόνου μεταφοράς εκπαιδευτικών/καθηγητών
- Ελαχιστοποίηση του χρόνου μεταφοράς μεταξύ τμημάτων του ίδιου εξαμήνου
- Πολλές ώρες του ίδιου τμήματος να πραγματοποιούνται στην ίδια αίθουσα
- Πολλές ώρες του ίδιου τμήματος να ανατίθενται στον ίδιο καθηγητή

Κεφάλαιο 3ο: Επίλυση του προβλήματος timetabling

3.1 Περιγραφή του προβλήματος

Σαν μελέτη εφαρμογής των όσων περιγράφονται στα προηγούμενα κεφάλαια, θα επιλυθεί στη συνέχεια, το πρόβλημα ωρολογίου προγράμματος για την σχολή Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου Ελλάδος.

Η επίλυση αφορά στον χρονοπρογραμματισμό των μαθημάτων σύμφωνα με το πρόγραμμα του εαρινού εξαμήνου 2023-2024. Ως βάση χρησιμοποιήθηκαν αφενός το πρόγραμμα σπουδών, και αφετέρου πληροφορίες για τους καθηγητές και τις αίθουσες που αντλήθηκαν από το υπάρχον ωρολόγιο πρόγραμμα.

Για την επίλυση του δεδομένου προβλήματος χρονοπρογραμματισμού, πρέπει να κατασκευαστεί ένα μοντέλο. Όλες οι μεταβλητές θα είναι δυαδικές, καθώς ουσιαστικά θα μοντελοποιούν τις αποφάσεις ναι ή όχι, δηλαδή εάν ένα μάθημα πρέπει να προγραμματιστεί σε μια συγκεκριμένη ώρα μιας συγκεκριμένης ημέρας σε μια αίθουσα με έναν καθηγητή ή όχι. Το μοντέλο αυτό έχει τη μορφή ενός προβλήματος ελαχιστοποίησης ακέραιου προγραμματισμού.

3.1.1 Μαθήματα

Η συγκεκριμένη σχολή προσφέρει μαθήματα κοινά και υποχρεωτικά στα εξάμηνα 2 και 4, ενώ από το 6ο εξάμηνο, οι φοιτητές χωρίζονται σε 2 κατευθύνσεις. Οι 2 κατευθύνσεις έχουν κοινά υποχρεωτικά μαθήματα, μαθήματα κατεύθυνσης υποχρεωτικά και μαθήματα επιλογής κοινά ή κατεύθυνσης.

3.1.1.1 Τύποι μαθημάτων

Ένα μάθημα μπορεί να περιλαμβάνει απλώς διαλέξεις (Th = Theory), ή διαλέξεις και εργαστήρια (Lab = Laboratory).

Οι διαλέξεις παραδίδονται από έναν ή περιστασιακά από 2 καθηγητές. **Στη συγκεκριμένη μελέτη περίπτωσης επιλέχθηκε τα μαθήματα να ανατίθενται σε έναν καθηγητή.**

Το εργαστήριο είναι μέρος ενός δεδομένου μαθήματος και στις περισσότερες περιπτώσεις, οι φοιτητές χωρίζονται σε ομάδες. Σε κάθε δηλαδή θεωρητικό μάθημα, μπορεί να αντιστοιχούν πολλά εργαστηριακά μαθήματα (ανάλογα με τις ομάδες φοιτητών που δημιουργούνται). Τα εργαστηριακά μαθήματα γίνονται σε ειδικά εξοπλισμένες αίθουσες διδασκαλίας με υπολογιστές και ανατίθενται είτε στον καθηγητή που διδάσκει τη θεωρία ή σε άλλους καθηγητές χαμηλότερης βαθμίδας ή/και σε βοηθούς εργαστηρίου.

Ένα μάθημα χαρακτηρίζεται από τον τύπο του (TYPE), που μπορεί να είναι:

- Th : Διάλεξη υποχρεωτικού κοινού ή κατεύθυνσης
- Ep: Διάλεξη μαθήματος επιλογής
- Lab: Εργαστήριο μαθήματος είτε υποχρεωτικού είτε επιλογής

3.1.1.2 Αριθμός Μαθημάτων

Για την ολοκλήρωση κάθε εξαμήνου, οι φοιτητές οφείλουν να παρακολουθήσουν 5 μαθήματα ως εξής:

- Στο 2ο εξάμηνο: 5 κοινά υποχρεωτικά μαθήματα που το καθένα αποτελείται από θεωρητικές διαλέξεις και εργαστήρια.

- Στο 4ο εξάμηνο: 5 κοινά μαθήματα, εκ των οποίων τα 2 αποτελούνται μόνο από διαλέξεις και τα υπόλοιπα 3 από διαλέξεις και εργαστήρια.
- Στο 6ο εξάμηνο: α) 2 κοινά υποχρεωτικά μαθήματα , εκ των οποίων το ένα περιλαμβάνει και εργαστήρια, β) 1 υποχρεωτικό μάθημα ανά κατεύθυνση (Ηλεκτρονικών / Πληροφορικής) και γ) 2 μαθήματα επιλογής ανά κατεύθυνση από μια γκάμα 7 μαθημάτων .
  -Τα επιλογής μαθήματα μπορεί να προσφέρονται και στις 2 κατευθύνσεις (κοινά) ή μόνο σε μία.
  -Κάθε μάθημα μπορεί να περιλαμβάνει διαλέξεις και εργαστήρια ή μόνο διαλέξεις.
- Στο 8ο εξάμηνο: α) 3 κοινά υποχρεωτικά μαθήματα, εκ των οποίων το ένα περιλαμβάνει και εργαστήρια και β) 2 μαθήματα επιλογής ανά κατεύθυνση από μια γκάμα 10 μαθημάτων.

Συνολικά, για τα 4 εξάμηνα, τα μαθήματα που πρέπει να χρονοπρογραμματιστούν είναι 138 (διαλέξεις και εργαστήρια), όπως φαίνεται αναλυτικά στο Παράρτημα Δεδομένα.

3.1.1.3 Κωδικοποίηση

Κάθε μάθημα προσδιορίζεται από έναν 4-ψήφιο κωδικό μαθήματος (SUBJECT_ID), πχ 1201. Η διάλεξη (Θεωρίας (Th) ή Επιλογής (Ep)) διατηρεί τον 4-ψήφιο κωδικό, ενώ τα τυχόν εργαστήρια του μαθήματος προσδιορίζονται ως 1201L1, 1201L2, 1201L3 κ.ο.κ.

3.1.1.4 Διάρκεια των μαθημάτων

Οι διαλέξεις έχουν διάρκεια 2 ή 4 ωρών, ενώ τα εργαστήρια έχουν όλα διάρκεια 2 ωρών.

Σημειώνεται ότι σε κάποιες σπάνιες περιπτώσεις όπου υπήρχαν μονόωρα εργαστήρια, αυτά προγραμματίστηκαν σε 2 ώρες ως εξής: Ενώ προβλεπόταν στο ωρολόγιο πρόγραμμα 10 ομάδες εργαστηρίου διάρκειας μιας ώρας, προγραμματίστηκαν 5 ομάδες των 2 ωρών.

Ανάλυση των στηλών του πίνακα μαθημάτων:

Πίνακας 3.1: Στήλες του πίνακα μαθημάτων

- SEMESTER	Το εξάμηνο στο οποίο διδάσκεται το μάθημα
- SUBJECT_ID	Η ταυτότητα (κωδικός) του μαθήματος
- SUBJECT	Ο τίτλος του μαθήματος
- Κατεύθυνση	Η κατεύθυνση στην οποία ανήκει το μάθημα (για τα εξάμηνα S6 και S8). Παίρνει τιμές: ΗΛΕΣ, ΠΛΗΡ ή ΚΟΙΝΟ
- CHARACTER	Ο χαρακτηρισμός του μαθήματος. Παίρνει τιμές: ΥΠΟΧ (= υποχρεωτικό), ΥΠΟΧ-ΕΠ (= Υποχρεωτικό για τη μια κατεύθυνση σπουδών και Επιλογής για την άλλη), ΕΠ (= Επιλογής)
- TYPE	Ο τύπος του μαθήματος. Παίρνει τιμές: Th (= Theory) για τις κοινές υποχρεωτικές διαλέξεις, Ep για τις διαλέξεις επιλογής και Lab (= laboratory) για τα εργαστήρια οποιουδήποτε τύπου
- DURATION	Η διάρκεια του μαθήματος σε ζώνες ωρών, δηλαδή σε 2-ωρα.
- THEORY	Η διάρκεια σε ώρες μιας διάλεξης
- LAB	Η διάρκεια σε ώρες ενός εργαστηρίου

- ΤΜΗΜΑ	Το τμήμα του εξαμήνου. Παίρνει τις τιμές Α ή Β, μόνο για το 2ο εξάμηνο
- ROOM_CATEGORY	Η ομάδα των αιθουσών στις οποίες μπορεί να διεξαχθεί το μάθημα
- PROFESSOR_N	Το όνομα του καθηγητή
- PROFESSOR_ID	Ο κωδικός αριθμός του καθηγητή
- SEMESTER_ID	Η ταυτότητα του εξαμήνου

3.1.2 Φοιτητές

Στη συγκεκριμένη σχολή εισάγονται ετησίως 200 φοιτητές. Όμως, για διάφορους λόγους, ο αριθμός αυτός δεν αντιπροσωπεύει τον αριθμό των φοιτητών που παρακολουθούνε τα μαθήματα κάθε εξαμήνου, όπως συμβαίνει και σε όλα τα πανεπιστήμια.

Σε γενικές γραμμές, οι παρακολουθούντες είναι περισσότεροι στα μικρά εξάμηνα και λιγότεροι στα μεγαλύτερα. Το γεγονός αυτό δημιουργεί πολλές δυσκολίες στον έλεγχο των προγραμμάτων σπουδών. Το πρόβλημα επιτείνεται από τα χαρακτηριστικά των πανεπιστημιακών μαθημάτων που περιεγράφηκαν παραπάνω (μαθήματα επιλογής), σε συνδυασμό με το γεγονός ότι διαφορετικοί φοιτητές μπορεί να ανήκουν σε διαφορετικές ομάδες εργαστηρίου.

Στις περισσότερες περιπτώσεις που συναντώνται στη σχετική αρθρογραφία, είτε γίνεται μια εκτίμηση του αριθμού των αναμενόμενων φοιτητών ανά μάθημα από τα στατιστικά δεδομένα προηγούμενων ετών, είτε το ωρολόγιο πρόγραμμα να οριστικοποιείται μετά από τις δηλώσεις των μαθημάτων. Με τον τρόπο αυτό, οι φοιτητές χωρίζονται σε ομάδες (GROUPS) συγκεκριμένου αριθμού μελών και ανάλογα με τις ομάδες και τη χωρητικότητα των αιθουσών διαμορφώνεται πχ ο αριθμός των διαφορετικών εργαστηρίων σε ένα μάθημα ή επιλέγεται αίθουσα που μπορεί να τους φιλοξενήσει.

Στη συγκεκριμένη μελέτη, προκειμένου να αποφευχθεί το πρόβλημα έλλειψης σχετικών στοιχείων, ελήφθησαν υπόψη ο ήδη προβλεπόμενος στο ισχύον ωρολόγιο πρόγραμμα αριθμός των ομάδων εργαστηρίων ανά μάθημα και ο τύπος των αιθουσών που προβλέπονται σ' αυτό.

3.1.3 Αίθουσες

Η σχολή διαθέτει 26 χώρους (ROOMS) για τη διεξαγωγή των μαθημάτων. Εξ' αυτών:

- Οι 2 είναι αμφιθέατρα
- Οι 9 είναι αίθουσες διδασκαλίας θεωρητικών μαθημάτων
- Οι 5 είναι εργαστήρια της κατεύθυνσης Πληροφορικής
- Οι 10 είναι εργαστήρια της κατεύθυνσης Ηλεκτρονικών

Κάθε χώρος χαρακτηρίζεται από έναν κωδικό (ROOM_ID) και τη χωρητικότητά του (CAPACITY).

Κάθε χώρος ανήκει σε μια κατηγορία (ROOM_CATEGORY). Οι κατηγορίες είναι οι εξής:

- LP – Εργαστήρια Πληροφορικής
- LH – Εργαστήρια Ηλεκτρονικής
- CK – Αίθουσες θεωριών
- CA – Αμφιθέατρα

Οι χώροι μιας Κατηγορίας μπορούν να χρησιμοποιηθούν εναλλακτικά για τη διεξαγωγή αντίστοιχων μαθημάτων. Πχ τα 18 εργαστήρια που προβλέπονται για το μάθημα 1304 μπορούν να διεξαχθούν σε εργαστήρια της κατηγορίας LP (δηλ. τα LP_201 ή LP_202 ή LP_208 ή LP_210 ή LP_211).

Αναλυτικά, τα στοιχεία των αιθουσών παρουσιάζονται στο Παράρτημα Δεδομένα.

Ανάλυση των στηλών του πίνακα των αιθουσών:

Πίνακας 3.2: Ταυτότητα της αίθουσας

- ROOM_ID	Η ταυτότητα της αίθουσας
- ROOM_NAME	Η ονομασία της αίθουσας
- ROOM_TYPE	Ο τύπος της αίθουσας (Lab = Εργαστήριο, Th = Αίθουσα διαλέξεων, Amphitheater = Αμφιθέατρο διαλέξεων)
- CAPACITY	Η χωρητικότητα της αίθουσας
- DIRECTION	Η κατεύθυνση σπουδών που κυρίως χρησιμοποιεί την αίθουσα (ΠΛΗΡΟΦΟΡΙΚΗΣ / ΗΛΕΚΤΡΟΝΙΚΩΝ / ΚΟΙΝΕΣ)
- ROOM_CATEGORY	Η ομάδα αιθουσών στην οποία ανήκει η συγκεκριμένη αίθουσα και μπορεί να χρησιμοποιηθεί εναλλακτικά με τις υπόλοιπες της ίδιας ομάδας για συγκεκριμένα μαθήματα

3.1.4 Ημέρες και Ώρες

Οι ημέρες παραδόσεων είναι 5, από Δευτέρα ως Παρασκευή.

Οι ώρες παραδόσεων ξεκινούν από τις 9.00π.μ. και τελειώνουν στις 20.00μ.μ, με ένα διάλειμμα 1 ώρας ανάμεσα στις 13.00μμ και 14.00μμ για μεσημεριανό φαγητό.

Δεδομένου ότι τα μαθήματα είναι 2-ωρα ή 4-ωρα, οι ώρες χωρίστηκαν σε ζώνες των 2 ωρών η καθεμιά, δηλ. 9-11, 11-13, 14-16, 16-18 και 18-20.

Ένας από τους στόχους και σημείο βελτιστοποίησης του προβλήματος είναι να αποφεύγονται τα απογευματινά και περισσότερο τα βραδινά μαθήματα. Έτσι, ακολουθώντας το παράδειγμα σχετικής αρθρογραφίας, σε κάθε ζώνη αποδόθηκε ένα 'βάρος' που ξεκινάει από 1 (για τις πρωινές ζώνες,) μέχρι 4 (για τη ζώνη 18-20). Όσο μεγαλύτερο βάρος παίρνει η κάθε ζώνη, τόσο πιο ανεπιθύμητη γίνεται αυτή.

Η κωδικοποίηση και οι πληροφορίες για τις ημέρες και τις ώρες εμφανίζονται στο Παράρτημα Δεδομένων.

Ανάλυση των στηλών του πίνακα των ωρών:

Πίνακας 3.3: Στήλες του πίνακα ωρών

- DAY_NAME	Το όνομα της ημέρας
- DAY_ID	Ο κωδικός της ημέρας (D1, D2, D3, D4 & D5)
- START	Η ώρα έναρξης της χρονικής ζώνης
- END	Η ώρα λήξης της χρονικής ζώνης
- HOUR_ID	Ο κωδικός της ζώνης που χαρακτηρίζεται από τις ώρες έναρξης και λήξης (πχ H_09_11)
- HR_WEIGHT	Το βάρος της συγκεκριμένης χρονικής ζώνης

3.1.5 Καθηγητές

Στη σχολή διδάσκουν 42 καθηγητές διαφόρων βαθμίδων και ειδικοτήτων (P1, P2, ….. P42). Οι καθηγητές που παραδίδουν κάθε διάλεξη ή επιβλέπουν ένα εργαστήριο έχουν αντληθεί από το ισχύον ωρολόγιο πρόγραμμα. Για λόγους χαμηλής πολυπλοκότητας του προβλήματος επιλέχθηκε για το κάθε μάθημα να αντιστοιχεί μόνο ένας καθηγητής. Ένα τυχαίο παράδειγμα το μάθημα 1404 να διδάσκεται μόνο από τον κ. Παπακώστα και όχι κ. Παπακώστα και κ. Χατζόπουλο μαζί.

3.1.6 Εξάμηνα

Όπως προαναφέρθηκε, το σύνολο των εξαμήνων που θα χρονοπρογραμματιστούν είναι 4 (2ο, 4ο, 6ο και 8ο) και χαρακτηρίζονται με μια ταυτότητα (SEMESTER_ID), πχ. S2, S4, S6 και S8.

Ειδικά, για το 2ο εξάμηνο επισημαίνεται ότι, λόγω του μεγάλου αριθμού των φοιτητών που παρακολουθούν και της ανεπάρκειας των αιθουσών να καλύψουν τις σχετικές ανάγκες, έχει χωριστεί σε 2 τμήματα, που κάνουν ακριβώς τα ίδια μαθήματα σε διαφορετικές όμως ημέρες και ώρες, ενδεχομένως και σε διαφορετικές αίθουσες.

Με βάση τα παραπάνω, πρακτικά το 2ο εξάμηνο χωρίζεται σε 2: S2A και S2B, ενώ τα μαθήματα (θεωρία και εργαστήρια) που πρέπει να χρονοπρογραμματιστούν είναι διπλά, διαφοροποιούμενα ως προς την ταυτότητά τους σε xxxxA και xxxxB ή (αν είναι εργαστήρια) σε xxxxLyA και xxxxLyB. Πρακτικά, αυτό σημαίνει ότι πρέπει να χρονοπρογραμματιστούν 5 εξάμηνα, αντί για 4.

Ο πίνακας των εξαμήνων παρουσιάζεται επίσης στο Παράρτημα Δεδομένα.

3.1.7 Περιορισμοί

Η δημιουργία του εβδομαδιαίου ωρολογίου προγράμματος για οποιαδήποτε πανεπιστημιακή σχολή πρέπει να ακολουθεί αρκετούς κανόνες, μερικοί από τους οποίους είναι τόσο σημαντικοί που δεν πρέπει να παραβιαστούν ποτέ (σκληροί περιορισμοί) και άλλοι που δεν είναι τόσο σημαντικοί και συνήθως τηρούνται μόνο εάν πληρούνται όλοι οι σκληροί περιορισμοί και υπάρχει ακόμα περιθώριο για καλύτερες λύσεις βάσει ορισμένων στόχων για βελτιωμένη ποιότητα (μαλακοί περιορισμοί).

Ο αριθμός των περιορισμών εξαρτάται από τα χαρακτηριστικά της κάθε πανεπιστημιακής σχολής, τους εμπλεκόμενους και τις απαιτήσεις τους (φοιτητές, καθηγητές), καθώς και την υπάρχουσα υλικοτεχνική υποδομή (αίθουσες, εργαστήρια, χωρητικότητα κλπ.)

Στη συγκεκριμένη μελέτη έχουν εισαχθεί 6 σκληροί περιορισμοί και είναι οι εξής:

ΑΠΑΙΤΗΣΗ ΠΛΗΡΟΤΗΤΑΣ

- Το πρόγραμμα πρέπει να είναι πλήρες. Κάθε μάθημα πρέπει να εμφανίζεται μέσα στην εβδομάδα, τόσες φορές (ώρες), όσες είναι η διάρκειά του (duration)

ΑΠΑΙΤΗΣΗ ΑΠΟΦΥΓΗΣ ΣΥΓΚΡΟΥΣΕΩΝ

- Κάθε αίθουσα μπορεί να ανατεθεί μόνο σε ένα μάθημα την κάθε ωριαία ζώνη μιας ημέρας.
- Κάθε καθηγητής μπορεί να διδάξει μόνο ένα μάθημα σε κάθε ωριαία ζώνη μιας ημέρας.
- Σε κάθε εξάμηνο, μόνο ένα υποχρεωτικό μάθημα (διάλεξη) μπορεί να διδάσκεται σε μια ωριαία ζώνη μιας ημέρας.

(Οι διαλέξεις των υποχρεωτικών μαθημάτων (τύπου 'Th') που αφορούν το σύνολο των φοιτητών ενός εξαμήνου θα πρέπει να δίνονται σε ώρα που δεν συμπίπτει με άλλο υποχρεωτικό θεωρητικό μάθημα του ιδίου εξαμήνου)

- Σε κάθε εξάμηνο, την ώρα διεξαγωγής ενός υποχρεωτικού μαθήματος (διάλεξης), δεν πρέπει να διεξάγονται εργαστήρια ή μαθήματα επιλογής.

(Ο περιορισμός αυτός είναι -πρακτικά- επέκταση του προηγούμενου, όσον αφορά τις ώρες διεξαγωγής των υποχρεωτικών διαλέξεων και των εργαστηρίων ή των μαθημάτων επιλογής)

ΛΟΙΠΟΙ ΠΕΡΙΟΡΙΣΜΟΙ

- Κάθε μάθημα διάρκειας 2 χρονικών ζωνών (δηλ. 4 ώρες), θα πρέπει να χωρίζεται και να διδάσκεται σε 2 διαφορετικές ημέρες (από ένα 2-ωρο τη φορά).

(Αφορά μόνο διαλέξεις για τις οποίες – ως επί το πλείστον – προβλέπεται στο πρόγραμμα σπουδών διάρκεια 4 ωρών)

3.2 Μαθηματική μοντελοποίηση του προβλήματος

3.2.1 Σύνολα

- SU: Το σύνολο των μαθημάτων
- SU_{SID}: Το σύνολο των μαθημάτων ενός εξαμήνου SID
- SU_{SIDTH}: Το σύνολο των υποχρεωτικών θεωρητικών διαλέξεων ενός εξαμήνου SID
- SU_{SIDLAB}: Το σύνολο των εργαστηριακών μαθημάτων ενός εξαμήνου SID
- SU_{SIDEP}: Το σύνολο των θεωρητικών διαλέξεων επιλογής ενός εξαμήνου SID
- $D = \{D1, D2, D3, D4, D5\}$: Το σύνολο των ημερών από Δευτέρα ως Παρασκευή
- H = {H_{09_11}, H_{11_13}, H_{14_16}, H_{16_18}, H_{18_20}}: Το σύνολο των ημερήσιων ωρών διδασκαλίας
- R: Το σύνολο των αιθουσών διδασκαλίας:
- R_cat= {CA, CK, LH, LP}: Οι διαφορετικές ομάδες αιθουσών που είναι κατάλληλες για τη διεξαγωγή ενός μαθήματος. Ισχύει: $\mathrm{CA} \cup \mathrm{CK} \cup \mathrm{LH} \cup \mathrm{LP} = R$ και η τομή των ομάδων του R_cat είναι το κενό σύνολο.
- R_cat_s: Η υπό-ομάδα αιθουσών $r \in R$ στην οποία μπορεί να διεξαχθεί ένα μάθημα $s \in SU$
- $SE = \{S2A, S2B, S4, S6, S8\}$: Το σύνολο των εξαμήνων SID που χρονοπρογραμματίζονται
- $P = \{P1, P2, \dots, P42\}$: Το σύνολο των καθηγητών

3.2.2 Παράμετροι

- sid_s: Το εξάμηνο ενός μαθήματος $s \in SU$
- $duration_s$: Οι ώρες διδασκαλίας ενός μαθήματος $s \in SU$
- p_s: Ο καθηγητής στον οποίο έχει ανατεθεί το μάθημα $s \in SU$
- w_h: Το βάρος μιας ώρας $h \in H$ στην οποία διεξάγονται μαθήματα
- M: ένας μεγάλος αριθμός (μεγαλύτερος από τον αριθμό του αθροίσματος των εργαστηρίων και των διαλέξεων επιλογής ενός εξαμήνου, για οποιοδήποτε εξάμηνο)

3.2.3 Μεταβλητές

Ορίζονται οι παρακάτω μεταβλητές:

- x_{s,d,h,r,p_s,sid_s}: Δυαδική μεταβλητή, όπου $s \in SU, d \in D, h \in H, r \in R\_cat_s, p_s \in P$ και $sid_s \in SE$. Η μεταβλητή παίρνει την τιμή 1 εάν ένα μάθημα s του εξαμήνου sid γίνεται την ημέρα d και ώρα h σε μια αίθουσα r από τον καθηγητή p. Αλλιώς παίρνει την τιμή 0.

- $y_{r,d,h}$: Δυαδική μεταβλητή, όπου $r \in R,\ d \in D\ και\ h \in H$. Η μεταβλητή παίρνει την τιμή 1 εάν στην αίθουσα r διεξάγεται κάποιο μάθημα την ημέρα d και την ώρα h. Αλλιώς παίρνει την τιμή 0.
- $a_{p,d,h}$: Δυαδική μεταβλητή, όπου $p \in P,\ d \in D\ και\ h \in H$. Η μεταβλητή παίρνει την τιμή 1 εάν ο καθηγητής p παραδίδει κάποιο μάθημα την ημέρα d και την ώρα h. Αλλιώς παίρνει την τιμή 0.
- $b1_{s,d,h}$: Δυαδική μεταβλητή, όπου $s \in SU_{SIDTH},\ d \in D\ και\ h \in H,\ \forall sid \in SE$. Η μεταβλητή παίρνει την τιμή 1 εάν μία θεωρητική διάλεξη του εξαμήνου sid παραδίδεται την ημέρα d και την ώρα h. Αλλιώς παίρνει την τιμή 0.
- $b2_{s,d,h}$: Δυαδική μεταβλητή, όπου $s \in SU_{SIDLAB},\ d \in D\ και\ h \in H,\ \forall sid \in SE$. Η μεταβλητή παίρνει την τιμή 1 εάν ένα εργαστήριο του εξαμήνου sid παραδίδεται την ημέρα d και την ώρα h. Αλλιώς παίρνει την τιμή 0.
- $b3_{s,d,h}$: Δυαδική μεταβλητή, όπου $s \in SU_{SIDEP},\ d \in D\ και\ h \in H,\ \forall sid \in SE$. Η μεταβλητή παίρνει την τιμή 1 εάν μία θεωρητική διάλεξη επιλογής του εξαμήνου sid παραδίδεται την ημέρα d και την ώρα h. Αλλιώς παίρνει την τιμή 0.

3.2.4 Περιορισμοί

Το πρόγραμμα πρέπει να είναι πλήρες. Κάθε μάθημα πρέπει να εμφανίζεται μέσα στην εβδομάδα, τόσες φορές (ώρες), όσες είναι η διάρκειά του (duration):

$$\forall s \in SU, \quad \sum_{d \in D} \sum_{h \in H} \sum_{r \in R_{cat_s}} x_{s,d,h,r,p_s,sid_s} = duration_s \qquad (3.1)$$

Κάθε αίθουσα μπορεί να ανατεθεί μόνο σε ένα μάθημα την κάθε ωριαία ζώνη μιας ημέρας:

$$\forall r \in R, \quad \sum_{d \in D} \sum_{h \in H} y_{r,d,h} \leq 1 \qquad (3.2)$$

Κάθε καθηγητής μπορεί να διδάξει μόνο ένα μάθημα σε κάθε ωριαία ζώνη μιας ημέρας:

$$\forall p \in P, \quad \sum_{d \in D} \sum_{h \in H} a_{p,d,h} \leq 1 \qquad (3.3)$$

Σε κάθε εξάμηνο, μόνο ένα υποχρεωτικό μάθημα (διάλεξη) μπορεί να διδάσκεται σε μια ωριαία ζώνη μιας ημέρας:

$$\forall sid \in SE, \quad \sum_{d \in D} \sum_{h \in H} \sum_{s \in SU_{SIDTH}} b1_{s,d,h} \leq 1 \qquad (3.4)$$

Σε κάθε εξάμηνο, την ώρα διεξαγωγής ενός υποχρεωτικού μαθήματος (διάλεξης), δεν πρέπει να διεξάγονται εργαστήρια ή μαθήματα επιλογής:

$$\forall sid \in SE, \quad \sum_{d \in D} \sum_{h \in H} \sum_{s \in SU_{SIDLAB}} b2_{s,d,h} + \sum_{d \in D} \sum_{h \in H} \sum_{s \in SU_{SIDEP}} b3_{s,d,h} \leq \left(1 - \sum_{d \in D} \sum_{h \in H} \sum_{s \in SU_{SIDTH}} b1_{s,d,h}\right) * M \qquad (3.5)$$

Με βάση την ανισότητα (3.5), εάν προγραμματιστεί οποιαδήποτε θεωρητική διάλεξη ($\sum_{d \in D} \sum_{h \in H} \sum_{s \in SU_{SIDTH}} b1_{s,d,h} = 1$), τότε το άθροισμα των προγραμματιζόμενων εργαστηρίων και

επιλογής πρέπει να είναι 0. Στην αντίθετη περίπτωση ($\sum_{d\in D}\sum_{h\in H}\sum_{s\in SU_{SIDTH}} b1_{s,d,h} = 0$), μπορούν να προγραμματιστούν μέχρι **M** εργαστήρια ή διαλέξεις επιλογής.

Κάθε μάθημα διάρκειας 2 χρονικών ζωνών (δηλ. 4 ώρες), θα πρέπει να χωρίζεται και να διδάσκεται σε 2 διαφορετικές ημέρες (από ένα 2-ωρο τη φορά):

$$\forall s \in SU, \varepsilon\acute{\alpha}\nu\ \ duration_s = 2, \tau\acute{o}\tau\varepsilon \sum_{d\in D} x_{s,d,h,r,p_s,sid_s} \leq 1 \qquad (3.6)$$

3.2.5 Αντικειμενική Συνάρτηση

Οι περιορισμοί που μόλις παρουσιάστηκαν, όταν τοποθετηθούν σε ένα μοντέλο ακέραιου προγραμματισμού, διασφαλίζουν πάντα ότι το εβδομαδιαίο χρονοδιάγραμμα που προκύπτει είναι έγκυρο, δηλαδή θα επιστρέψουν πάντα εφικτές λύσεις.

Η αντικειμενική συνάρτηση θα πρέπει να επιλεγεί ώστε να αντικατοπτρίζει τις όποιες προτιμήσεις έχουν αυτοί που αποφασίζουν. Αυτό γίνεται με την ανάθεση βαρών στις μεταβλητές που συμμετέχουν σ' αυτήν. Καθώς πρόκειται για πρόβλημα ελαχιστοποίησης, ένα μεγάλο βάρος θα αυξήσει την αντικειμενική συνάρτηση και επομένως δεν θα είναι επωφελές για τη λύση. Επομένως, τα βάρη ελέγχουν το αποτέλεσμα και ο τρόπος που επιλέγονται είναι συζητήσιμος.

Στην προκειμένη περίπτωση, δεδομένου ότι είναι θεμιτό τα μαθήματα να διεξάγονται κυρίως πρωινές ώρες, η αντικειμενική συνάρτηση που επιλέχθηκε είναι η:

$$Minimize \left\{ \sum_{s\in SU}\sum_{d\in D}\sum_{h\in H}\sum_{r\in R_{cat_S}} w_h * x_{s,d,h,r,p_s,sid_s} \right\} \qquad (3.7)$$

Όπως φαίνεται στην εξίσωση (10), ο στόχος είναι να ελαχιστοποιηθεί η συνάρτηση κόστους διεξαγωγής του μαθήματος s στην h χρονική περίοδο της ημέρας d.

3.2.6 Καθορισμός των βαρών

Βάρος w_h

Το βάρος της κάθε χρονικής περιόδου καθορίστηκε με βάση προηγούμενες παρόμοιες βιβλιογραφικές έρευνες (βλ. [42]), ως εξής:

Πίνακας 3.4: Ανάθεση βαρών

START	END	HOUR_ID	W_h
9	11	H_09_11	1
11	13	H_11_13	1
14	16	H_14_16	2
16	18	H_16_18	3
18	20	H_18_20	4

Αυτό σημαίνει ότι:

- το κόστος των ζωνών 9-11 πμ και 11πμ-13μμ είναι το μικρότερο δυνατόν
- το κόστος τα ζώνης 14-16μμ είναι διπλάσιο του κόστους των προηγούμενων ζωνών

- το κόστος της ζώνης 16-18μμ, τριπλάσιο και της ζώνης 18-20μμ τετραπλάσιο του κόστους των ζωνών 9-11 πμ ή 11πμ-13μμ.

Κεφάλαιο 4ο: Προγραμματισμός και γλώσσα Python

Το μοντέλο ακέραιου (δυαδικού) προγραμματισμού για την επίλυση του προβλήματος timetabling για την Σχολή Μηχανικών του Τμήματος Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙΠΑΕ, όπως αυτό αναλύθηκε στο προηγούμενο κεφάλαιο, προγραμματίστηκε σε γλώσσα Python για την επιβεβαίωση και ανάλυση των αποτελεσμάτων.

Η έκδοση της γλώσσας που χρησιμοποιήθηκε είναι η Python 3.9.18 και το πρόγραμμα έτρεξε σε laptop με επεξεργαστή Intel core i5-1235U 3.30 GHz, εγκατεστημένη RAM 16,00 GB και λειτουργικό Windows 11 Pro-64-bit.

4.1 Είσοδος – Έξοδος. Κλάση auxiliary

Για την πληρέστερη είσοδο των δεδομένων και έξοδο των αποτελεσμάτων δημιουργήθηκε μια κλάση της python με όνομα auxiliary, στην οποία περιλήφθηκαν σχετικές μέθοδοι διευκόλυνσης. Επιπλέων η κλάση χρησιμοποιεί της βιβλιοθήκες PANDAS και OPENPYXL για την είσοδο και την έξοδο των δεδομένων στο πρόγραμμα.

4.1.1 Βοηθητικές βιβλιοθήκες PANDAS και OPENPYXL

PANDAS[57]**:**

Τύπος: Ανοικτού κώδικα

Χαρακτηριστικά: Κατάλληλη για την εισαγωγή, επεξεργασία και ανάλυση μεγάλου όγκου δεδομένων καθώς παρέχει δομές δεδομένων όπως το DataFrame (δισδιάστατος πίνακας παρόμοιος με υπολογιστικά φύλλα excel)

Υποστήριξη αρχείων: CSV, Excel, SQL βάσεις δεδομένων κ.α.

Άδεια χρήσης: Δεν απαιτείται

OPENPYXL [58]**:**

Τύπος: Ανοικτού κώδικα

Χαρακτηριστικά: Κατάλληλη για την δημιουργία αρχείων Excel, την επεξεργασία και την μορφοποίηση των περιεχομένων. Καθίσταται ιδιαίτερα χρήσιμη διότι δεν απαιτείται η εγκατάσταση του ίδιου Microsoft Excel.

Υποστήριξη αρχείων: Excel

Άδεια χρήσης: Δεν απαιτείται

4.1.2 Είσοδος

Τα δεδομένα προετοιμάστηκαν με τη μορφή αρχείου Excel, το οποίο αποτελούνταν από 6 φύλλα με τις ονομασίες:

- ROOMS
- SUBJECTS
- SEMESTERS

- PROFESSORS
- DAYS
- HOURS

Οι πίνακες κάθε φύλλου είχαν τη δομή των πινάκων που παρουσιάζονται στο Παράρτημα Δεδομένων και φορτώθηκαν στο πρόγραμμα με την auxiliary μέθοδο **upload_data(excel_file_path)**

```
def upload_data(excel_file_path):
    """
    Anevazei ta dedomena pou uparxoun sto excel
    param: h diadromi tou arxeiou
    return: ta fulla excel me tin morfi dataframes opou key -->
    """

    # anoigma arxeiou excel me ti vivliothiki Pandas
    excel_file = pd.ExcelFile(excel_file_path)
    # Dimiourgia adeiou pinaka pou tha periexei ta dedomena apo
    df_dict = {}
    # prospelasi sta fulla tou excel kai metafora ton dedomenon
    for sheet_name in excel_file.sheet_names:
        df = pd.read_excel(excel_file, sheet_name=sheet_name)
        df_dict[sheet_name] = df
    # kleisimo tou arxeioy excel
    excel_file.close()

    return df_dict
```

Σχήμα 4.1: Μέθοδος upload_data()

Με τη μέθοδο αυτή διαβάστηκε το αρχείο excel από το πρόγραμμα και δημιουργήθηκε ένα λεξικό df_dict από 6 αντίστοιχα dataframes. Οι ενέργειες αυτές πραγματοποιούνται με την χρήση της βιβλιοθήκης PANDAS.

Στη συνέχεια χρησιμοποιήθηκαν 2 προπαρασκευαστικές μέθοδοι: **add_potential_rooms(df_subjects, df_rooms)** και **add_professor_hours(df_subjects, df_professors)**:

```python
def add_potential_rooms(df_subjects, df_rooms):
    """
    Prosthetei mia stili sta SUBJECTS
    pou periexei mia lista me ROOMS simfona me to room_category gia to sigkekrimeno mat
    :param: 2 dataframes
    :return: SUBJECTS me mia prostheti stili 'POTENTIAL_ROOMS'.
    """
    # prosthiki neas stilis kai arxikopoihsh kenhs listas gia tin apothikeusi pithanwn
    df_subjects['POTENTIAL_ROOMS'] = [[] for _ in range(len(df_subjects))]

    # Gia kathe grammi tou DataFrame twn Subjects
    for index_s, subject_row in df_subjects.iterrows():

        # Gia kathe grammi tou Dataframe twn Rooms
        for _, room_row in df_rooms.iterrows():
            # Check an i katigoria tou Room tairiazei me tin antistixi katigoria gia to
            if (room_row['ROOM_CATEGORY'] == subject_row['ROOM_CATEGORY']):
                df_subjects.at[index_s, 'POTENTIAL_ROOMS'].append(room_row['ROOM_ID'])

    return df_subjects
```

Σχήμα 4.2: Μέθοδος add_potential_rooms()

```python
def add_professor_hours(df_subjects, df_professors):
    """
    Prosthetei mia stili sto PROFESSORS pou periexei upologismenes xronothirides (1= 2wres) ana ka
    simfona me to programma
    :param: 2 dataframes
    :return: PROFESSORS me mia prostheti stili 'HOURS_PER_WEEK'.
    """
    # Arxikopoihsh stilis me 0
    df_professors['HOURS_PER_WEEK']= 0

    # Gia kathe mathima vriskei ton kathigiti kai tis ores pou apaitountai
    for _, subject_row in df_subjects.iterrows():
        prof_id = subject_row['PROFESSOR_ID']
        duration = subject_row['DURATION']
        # Prosthetei ston kathe kathigiti tis ores pou didaskei stin nea stili
        df_professors.loc[df_professors['PROFESSOR_ID'] == prof_id, 'HOURS_PER_WEEK'] += duration
        # .loc Access group of values using labels.

    return df_professors
```

Σχήμα 4.3: Μέθοδος add_professor_hours()

Η μέθοδος **add_potential_rooms(df_subjects, df_rooms)** δημιουργεί στο dataframe ‘SUBJECTS’ μια στήλη ‘POTENTIAL_ROOMS’ , με στοιχεία μιας λίστας αιθουσών όπου μπορεί να διεξαχθεί το κάθε μάθημα, ανάλογα με τα δεδομένα της στήλης ‘ROOM_CATEGORY’ και των αντίστοιχων στοιχείων του dataframe ‘ROOMS’. Με τον τρόπο αυτό περιορίζονται οι αίθουσες στις οποίες μπορεί να διεξαχθεί ένα μάθημα στις προβλεπόμενες και άρα μειώνεται και ο τελικός αριθμός των μεταβλητών που συμμετέχουν στο πρόβλημα.

Η μέθοδος **add_professor_hours(df_subjects, df_professors)** δημιουργεί στο dataframe ‘PROFESSORS’ μια στήλη ‘HOURS_PER_WEEK’ , με στοιχεία το σύνολο των προβλεπόμενων ωρών κάθε καθηγητή ανά εβδομάδα, όπως αντλούνται από τις αναθέσεις (dataframe ‘SUBJECTS’, στήλες ‘PROFESSOR_ID’ και ‘DURATION’. Η πληροφορία αυτή είναι χρήσιμη για την περίπτωση που οι

αναθέσεις ενός καθηγητή είναι περισσότερες από τον προβλεπόμενο μέγιστο αριθμό ωρών διδασκαλίας. Στην συγκεκριμένη περίπτωση του προβλήματός οι 2-ωρες ζώνες των καθηγητών δεν λαμβάνονται υπόψιν, αλλά η πληροφορία μπορεί να χρησιμοποιηθεί μελλοντικά.

4.1.3 Έξοδος

Τα αποτελέσματα της επίλυσης εγγράφηκαν σε αρχείο Excel με όνομα TIMETABLE.xlsx και 2 φύλλα, με αντίστοιχα ονόματα:

- SUBJECTS: Το εβδομαδιαίο πρόγραμμα όπου αναγράφονται ανά ημέρα, ώρα και αίθουσα τα μαθήματα που διεξάγονται, σε μορφή πίνακα διπλής εισόδου
- PROFESSORS: Το αντίστοιχο εβδομαδιαίο πρόγραμμα των καθηγητών, στην ίδια μορφή

Γιαν να επιτευχθεί αυτό χρησιμοποιήθηκαν συγκεκριμένες μέθοδοι στην κλάση auxiliary, όπως περιγράφονται παρακάτω:

α) **set_up_timetable(df_rooms, df_days, df_hours):** Δημιουργεί το εβδομαδιαίο πρόγραμμα σε έναν πίνακα, με γραμμές τις Ημέρες και τις Ώρες και στήλες τις Αίθουσες. Τον Πίνακα αυτό γεμίζει καταρχήν με 'None' σε κάθε κελί. Παράλληλα, δημιουργεί και μια βοηθητική λίστα free, η οποία χρησιμοποιείται παρακάτω στο πρόβλημα για την αποφυγή συγκρούσεων ανάμεσα στα αποτελέσματα για κάθε μάθημα, αίθουσα και καθηγητή (Αποφυγή διπλο-εγγραφών).

```
def set_up_timetable(df_rooms, df_days, df_hours):
    """
    Orizei ton pinaka orologiou programmatos kai to leksiko pou apothikeuei ta eleuth
        -matrix: lista me stixeia xronothiridon days_times_hours. Ta stoixia einai li
        me ton arithmo aithouswn
        - free: mia lista apo pedia (x,y), opou x metritis apo 0 - (d_times_hours -1)
        kai y metritis apo 0 - (numbers of rooms -1)

    :param num_of_rooms, arithmos twn classrooms
    :      days_times_hours,  days multiplied with hours/day
    :return: matrix, free
    """
    num_of_rooms = len(df_rooms)
    days_times_hours = len(df_days) * len(df_hours)
    matrix = [[None for x in range(num_of_rooms)] for y in range(days_times_hours)]
    free = []

    # arxokopoiisi ths free listas ws ola ta pedia apo to matrix
    for i in range(len(matrix)):
        for j in range(len(matrix[i])):
            free.append((i, j))

    return matrix, free
```

Σχήμα 4.4: Μέθοδος set_up_timetable()

β) **def populate_subjects_timetable (matrix, free, df_subjects, df_rooms, df_days, df_hours, variable_list)**: Χρησιμοποιεί την λίστα των αποτελεσμάτων της επίλυσης και ιδιαίτερα αυτών που έχουν τιμή = 1 (variable_list) για να τοποθετήσει τα μαθήματα στο εβδομαδιαίο πρόγραμμα στην κατάλληλη ημέρα, ώρα και αίθουσα. Σε περίπτωση διπλο-εγγραφής, δηλαδή προσπάθεια τοποθέτησης ενός μαθήματος σε μη άδειο κελί, τυπώνει στην κονσόλα σχετικό μήνυμα λάθους: *'Potential conflict:'*,

*matrix [index_d*len(df_days['DAY_ID'])+ index_h][index_r], ' vs ', s, ' on day ', d, ', hour ', h, ', room ', r, '. **********').* Η μέθοδος παρουσιάζεται αναλυτικά στο Παράρτημα Β: Κώδικας

γ) **def populate_professors_timetable (matrix, free, df_subjects, df_rooms, df_days, df_hours, variable_list)**: Κάνει την ίδια δουλειά με την προηγούμενη μέθοδο, τοποθετώντας αυτή τη φορά τους καθηγητές στο εβδομαδιαίο πρόγραμμα στην κατάλληλη ημέρα, ώρα και αίθουσα. Και αυτή η μέθοδος παρουσιάζεται αναλυτικά στο Παράρτημα Β: Κώδικας

δ) **def show_timetable(matrix, df_days, df_hours, df_rooms)**: Εμφανίζει το εβδομαδιαίο πρόγραμμα στην κονσόλα. Το πρόγραμμα που εμφανίζεται είναι αυτό για το οποίο χρησιμοποιήθηκε η αντίστοιχη μέθοδος populate την τελευταία φορά, και καταχώρησε εγγραφή στο matrix.

```
def show_timetable(matrix, df_days, df_hours, df_rooms):
    """
    Ektiponei ton pinaka me to orologio programma
    Dimiourgei ektiposi tou pinaka se morfi pinaka opou oi stiles einai to room_id
    kai oi grammes me vasi a) day_name kai b) hour_id
    """
    days = []
    hours = []
    rooms = []
    for day in df_days['DAY_NAME']:
        days.append(day)
    for hour in df_hours['HOUR_ID']:
        hours.append(hour)
    for room in df_rooms['ROOM_ID']:
        rooms.append(room)

    # Diamorfosi kefalidas
    headlineD = '|' + ' ' * 9
    headlineH = '|' + ' ' * 7
    headlineA = '|' + ' ' * 5
    headline = headlineD + headlineH + headlineA # Diamorfosi kenou xwrou prin tin ektiposi
    # Prosthese se kathe aithousa stin eksodo me aristeri stixisi se apostasi 13 spaces
    for room in rooms:
        headline += '{:<13}'.format(room)
    print (headline)
    print()

    # Simplirosi tou orologiou programmatos
    day_counter = 0
    hour_counter = 0
    for i in range(len(matrix)):
        day = days[day_counter]
        hour = hours[hour_counter]
        # ektiposi meres, ores kai ena velos stin arxi tis grammis
        print('{:<10} {:<8} ->  '.format(day, hour), end='')
        for j in range(len(matrix[i])):
            print('{:<12} '.format(str(matrix[i][j])), end='')
        print()
        hour_counter += 1
        if hour_counter == len(hours):
            hour_counter = 0
            day_counter += 1
            print()
```

Σχήμα 4.5: Μέθοδος show_timetable()

Για την εγγραφή των αποτελεσμάτων σε αρχείο excel, χρησιμοποιούνται οι παρακάτω μέθοδοι:

ε) **def write_timetable_subjects(matrix, df_days, df_hours, df_rooms)**: Δημιουργεί το αρχείο TIMETABLE.xlsx με την χρήση της της μεθόδου Workbook() της βιβλιοθήκης OPENPYXL και γράφει το εβδομαδιαίο πρόγραμμα των μαθημάτων στο φύλλο 'SUBJECTS'.

```
def write_timetable_subjects(matrix, df_days, df_hours, df_rooms):
    """
    Apothikeush orologiou programmatos se arxeio excel, me onoma TIMETABLE sto fullo SUBJECTS
    """
    days = df_days['DAY_NAME'].tolist()
    hours = df_hours['HOUR_ID'].tolist()
    rooms = df_rooms['ROOM_ID'].tolist()

    # Dimiourgeia arxeiou
    wb = Workbook()
    ws = wb.active
    ws.title = 'SUBJECTS'

    # Dimiourgia kefalidas
    blank = ' '
    headline = ['{:<20} '.format(blank)]
    headline.append('{:<20} '.format(blank))
    headline.append('{:<12} '.format(blank))

    for room in rooms:
        headline.append('{:<13}'.format(room))
    ws.append(headline)
    # prosthiki kenis grammis
    ws.append([''] * (len(headline) + 1))

    # eggrafi tou programmatos
    day_counter = 0
    hour_counter = 0
    for i in range(len(matrix)):
        day = days[day_counter]
        hour = hours[hour_counter]
        row = ['{:<20} '.format(day)]
        row.append('{:<20} '.format(hour))
        row.append('  -->  ')
        for j in range(len(matrix[i])):
            row.append('{:<12} '.format(str(matrix[i][j])))
        ws.append(row)
        hour_counter += 1
        if hour_counter == len(hours):
            hour_counter = 0
            day_counter += 1
            ws.append([''] * (len(headline) + 1))

    # apothikeusi arxeiou
    wb.save('TIMETABLE.xlsx')
```

Σχήμα 4.6: Μέθοδος write_timetable_subjects()

στ) **def write_timetable_professors(matrix, df_days, df_hours, df_rooms)**: Ανοίγει το αρχείο TIMETABLE.xlsx με την χρήση της μεθόδου Load_workbook() της βιβλιοθήκης OPENPYXL και γράφει το εβδομαδιαίο πρόγραμμα των καθηγητών στο φύλλο 'PROFESSORS'.

```
def write_timetable_professors(matrix, df_days, df_hours, df_rooms):
    """
    Apothikeush orologiou programmatos kathigitwn se arxeio excel, me onoma TIMETABLE sto fullo PROFESSORS
    """
    days = df_days['DAY_NAME'].tolist()
    hours = df_hours['HOUR_ID'].tolist()
    rooms = df_rooms['ROOM_ID'].tolist()

    # Fortosi tou uparxontos arxeiou
    wb = load_workbook('TIMETABLE.xlsx')

    # Dimiourgia neou fillou PROFESSOR sto arxeio
    ws = wb.create_sheet('PROFESSORS')

    # Dimiourgia kefalidas
    blank = ' '
    headline = ['{:<20} '.format(blank)]
    headline.append('{:<20} '.format(blank))
    headline.append('{:<12} '.format(blank))

    for room in rooms:
        headline.append('{:<13}'.format(room))
    ws.append(headline)
    ws.append([''] * (len(headline) + 1))

    # Eggrafi tou programmatos
    day_counter = 0
    hour_counter = 0
    for i in range(len(matrix)):
        day = days[day_counter]
        hour = hours[hour_counter]
        row = ['{:<20} '.format(day)]
        row.append('{:<20} '.format(hour))
        row.append('  -->  ')
        for j in range(len(matrix[i])):
            row.append('{:<12} '.format(str(matrix[i][j])))
        ws.append(row)
        hour_counter += 1
        if hour_counter == len(hours):
            hour_counter = 0
            day_counter += 1
            ws.append([''] * (len(headline) + 1))
    # Apothikeusi sto arxeio
    wb.save('TIMETABLE.xlsx')
```

Σχήμα 4.7: Μέθοδος write_timetable_professors()

Τα αποτελέσματα έχουν ενδεικτικά την παρακάτω μορφή:

			LP_201	LP_202	LP_208	LP_210	LP_211	LH_A1	------	CK_102	CK_109	AK_AMF1	AK_AMF2	CK_B1	CK_B2	CK_B3	CK_B4	CK_B5	CK_B6

Monday	H_09_11	-->	1304L5	1203L2A	1403L8	1304L11	1401L3	None	------	None	1801	1201B	None	None	None	1602	None	None	None
Monday	H_11_13	-->	1304L9	1304L18	1401L6	1403L7	1304L13	None	------	None	None	1203B	1204A	1802	None	None	1641	None	None
Monday	H_14_16	-->	None	None	None	None	None	1611L4	------	None	1404	None	1205A	None	None	None	None	None	None
Monday	H_16_18	-->	None	None	None	None	None	None	------	None	None	None	None	None	None	None	None	None	None
Monday	H_18_20	-->	None	None	None	None	None	None	------	1403	None	None	None	None	None	None	None	None	None
									------										
Tuesday	H_09_11	-->	1205L2A	1203L5A	None	1203L4A	1205L3A	1204L2A	------	1402	1803	1204B	None	1673	1642	None	1613	1612	1643
Tuesday	H_11_13	-->	1403L2	1304L3	1304L8	1403L5	1304L10	None	------	1602	None	1205B	1203A	None	None	None	1802	None	None
Tuesday	H_14_16	-->	None	1205L1A	1205L4A	1203L1A	None	None	------	None	1601	1401	1201B	None	None	None	None	None	None
Tuesday	H_16_18	-->	None	None	None	None	None	None	------	None	None	None	None	None	None	None	None	None	None
Tuesday	H_18_20	-->	None	None	None	None	None	None	------	None	None	None	None	None	None	None	None	None	None
									------										
Wednesd	H_09_11	-->	1205L2B	1672L	1203L5B	1203L2B	1205L3B	1602L3	------	None	None	1898	1205A	None	1403	None	1972	1873	None
Wednesd	H_11_13	-->	1403L6	1401L4	1401L7	1203L4B	1304L2	1611L2	------	1612	1672	None	1201A	1642	None	1671	None	1613	1643
Wednesd	H_14_16	-->	None	None	None	None	None	None	------	None	None	1204B	1401	None	None	1202A	None	None	None
Wednesd	H_16_18	-->	None	None	None	None	None	None	------	None	None	1304	None	None	None	None	None	None	None
Wednesd	H_18_20	-->	None	None	None	None	None	None	------	None	None	None	None	None	None	None	None	None	None
									------										
Thursday	H_09_11	-->	1205L1B	None	1205L4B	1203L6B	1203L1B	None	------	1874	None	1898	None	1202A	1842	1948	1972	1871	1641
Thursday	H_11_13	-->	1403L3	1304L15	1304L17	1304L7	1401L5	None	------	1873	None	1205B	1203A	None	None	1611	None	None	1974
Thursday	H_14_16	-->	None	None	1203L6A	None	1203L3A	1204L4B	------	None	None	None	1304	None	None	None	1601	None	None
Thursday	H_16_18	-->	None	None	None	None	None	None	------	None	None	None	None	None	None	None	None	None	None
Thursday	H_18_20	-->	None	None	None	None	None	None	------	None	None	None	None	None	None	None	None	None	None
									------										
Friday	H_09_11	-->	1304L16	1304L12	1403L1	1304L14	1401L1	1202L6A	------	1974	1671	1203B	None	1812	None	1874	1948	1871	1842
Friday	H_11_13	-->	1304L4	1304L6	1401L2	1403L4	1304L1	None	------	None	1801	1201A	None	None	None	None	None	1202B	1611
Friday	H_14_16	-->	None	1203L3B	None	None	None	None	------	None	None	1204A	None	None	None	None	None	None	1402
Friday	H_16_18	-->	None	None	None	None	None	None	------	None	None	None	None	None	None	None	None	None	None
Friday	H_18_20	-->	None	None	None	None	None	None	------	None	None	None	None	None	None	None	None	None	None

Σχήμα 4.8: Μορφή αποτελεσμάτων 1

			LP_201	LP_202	LP_208	LP_210	LP_211	LH_A1	------	CK_102	CK_109	AK_AMF1	AK_AMF2	CK_B1	CK_B2	CK_B3	CK_B4	CK_B5	CK_B6

Monday	H_09_11	-->	P11	P2	P1	P20	P17	None	------	None	P12	P35	None	None	None	P29	None	None	None
Monday	H_11_13	-->	P16	P38	P17	P1	P39	None	------	None	None	P41	P26	P10	None	None	P5	None	None
Monday	H_14_16	-->	None	None	None	None	None	P40	------	None	P30	None	P6	None	None	None	None	None	None
Monday	H_16_18	-->	None	None	None	None	None	None	------	None	None	None	None	None	None	None	None	None	None
Monday	H_18_20	-->	None	None	None	None	None	None	------	P32	None	None	None	None	None	None	None	None	None
									------										
Tuesday	H_09_11	-->	P34	P37	None	P2	P33	P7	------	P14	P41	P26	None	P22	P18	None	P13	P23	P21
Tuesday	H_11_13	-->	P32	P3	P16	P1	P20	None	------	P29	None	P6	P41	None	None	None	P10	None	None
Tuesday	H_14_16	-->	None	P34	P33	P2	None	None	------	None	P27	P28	P35	None	None	None	None	None	None
Tuesday	H_16_18	-->	None	None	None	None	None	None	------	None	None	None	None	None	None	None	None	None	None
Tuesday	H_18_20	-->	None	None	None	None	None	None	------	None	None	None	None	None	None	None	None	None	None
									------										
Wednesd	H_09_11	-->	P34	P29	P37	P2	P33	P9	------	None	None	P22	P6	None	P32	None	P24	P8	None
Wednesd	H_11_13	-->	P1	P17	P28	P2	P3	P11	------	P23	P29	None	P35	P18	None	P14	None	P13	P21
Wednesd	H_14_16	-->	None	None	None	None	None	None	------	None	None	P26	P28	None	None	P19	None	None	None
Wednesd	H_16_18	-->	None	None	None	None	None	None	------	None	None	P20	None	None	None	None	None	None	None
Wednesd	H_18_20	-->	None	None	None	None	None	None	------	None	None	None	None	None	None	None	None	None	None
									------										
Thursday	H_09_11	-->	P34	None	P33	P37	P2	None	------	P15	None	P22	None	P19	P6	P31	P24	P8	P5
Thursday	H_11_13	-->	P1	P39	P20	P16	P17	None	------	P8	None	P6	P41	None	None	P42	None	None	P24
Thursday	H_14_16	-->	None	None	P37	None	P2	P25	------	None	None	None	P20	None	None	None	P27	None	None
Thursday	H_16_18	-->	None	None	None	None	None	None	------	None	None	None	None	None	None	None	None	None	None
Thursday	H_18_20	-->	None	None	None	None	None	None	------	None	None	None	None	None	None	None	None	None	None
									------										
Friday	H_09_11	-->	P16	P20	P32	P39	P17	P36	------	P24	P14	P41	None	P19	None	P15	P31	P8	P6
Friday	H_11_13	-->	P11	P16	P17	P1	P3	None	------	None	P12	P35	None	None	None	None	None	P19	P42
Friday	H_14_16	-->	None	P2	None	None	None	None	------	None	None	P26	None	None	None	None	None	None	P14
Friday	H_16_18	-->	None	None	None	None	None	None	------	None	None	None	None	None	None	None	None	None	None
Friday	H_18_20	-->	None	None	None	None	None	None	------	None	None	None	None	None	None	None	None	None	None

Σχήμα 4.9: Μορφή αποτελεσμάτων 2

Τα χρώματα στα παραπάνω σχήματα με την μορφή αποτελεσμάτων έχουν μπει χειροκίνητα ώστε να είναι πιο ευδιάκριτα, επίσης έχουν αγνοηθεί κάποιες αίθουσες για λόγους χωρητικότητας στο σχήμα.

4.2 Βιβλιοθήκη για την επίλυση προβλημάτων Γραμμικού Προγραμματισμού

Υπάρχουν αρκετές βιβλιοθήκες Python που είναι κατάλληλες για την επίλυση προβλημάτων γραμμικού προγραμματισμού. Μερικές από τις πιο δημοφιλείς είναι:

PuLP [51]**:**

Τύπος: Ανοικτού κώδικα

Χαρακτηριστικά: Εύκολη σύνταξη. Επίλυση προβλημάτων γραμμικού, ακέραιου και τετραγωνικού προγραμματισμού. Κατάλληλη για επίλυση γραμμικών προβλημάτων με χιλιάδες μεταβλητές και περιορισμούς.

Υποστήριξη αλγορίθμων: Δυνατότητα διασύνδεσης με πολλούς αλγόριθμους ανοικτού κώδικα ή αδειοδοτούμενους, όπως οι CBC, GLPK και CPLEX.

Άδεια χρήσης: Δεν απαιτείται

Pyomo [52]**:**

Τύπος: Ανοικτού κώδικα

Χαρακτηριστικά: Εύκολη σύνταξη. Επίλυση προβλημάτων γραμμικού (LP), μεικτού ακέραιου (MILP), τετραγωνικού (QP) και μη-γραμμικού (NLP) προγραμματισμού

Υποστήριξη αλγορίθμων: Δυνατότητα διασύνδεσης με πολλούς αλγόριθμους ανοικτού κώδικα ή αδειοδοτούμενους, όπως οι GLPK, Gurobi και CPLEX.

Άδεια χρήσης: Δεν απαιτείται

SciPy.optimize.linprog [53]**:**

Τύπος: Η βιβλιοθήκη SciPy είναι Ανοικτού κώδικα και χρησιμοποιείται για επιστημονικούς υπολογισμούς

Χαρακτηριστικά: Η συνάρτηση linprog στο module optimize μπορεί να λύσει προβλήματα γραμμικού προγραμματισμού χρησιμοποιώντας τη μέθοδο Simplex.

Υποστήριξη αλγορίθμων: Παρέχει μια βασική υλοποίηση του αλγορίθμου Simplex για γραμμικό προγραμματισμό.

Άδεια χρήσης: Δεν απαιτείται

CVXPY [54]**:**

Τύπος: Ανοικτού κώδικα

Χαρακτηριστικά: Επιτρέπει στους χρήστες να επιλύουν κυρτά προβλήματα βελτιστοποίησης χρησιμοποιώντας τη σύνταξη Python. Υποστηρίζει διάφορους τύπους προβλημάτων, συμπεριλαμβανομένου του γραμμικού προγραμματισμού και του τετραγωνικού προγραμματισμού

Υποστήριξη αλγορίθμων: Δυνατότητα διασύνδεσης με πολλούς αλγόριθμους ανοικτού κώδικα ή αδειοδοτούμενους, όπως οι SCS, Gurobi και Mosek

Άδεια χρήσης: Διανέμεται μέσω του Apache και δεν απαιτείται άδεια

Gurobi [55] **and CPLEX** [56]**:**

Τύπος: Το Gurobi και το CPLEX είναι πολύ ισχυρά εμπορικά πακέτα βελτιστοποίησης που χρησιμοποιούνται ευρέως για πολύπλοκα προβλήματα.

Χαρακτηριστικά: Και οι δύο λύτες προσφέρουν αλγόριθμους βελτιστοποίησης υψηλής απόδοσης και επιλύουν διάφορους τύπους προβλημάτων, συμπεριλαμβανομένου του γραμμικού προγραμματισμού, του προγραμματισμού μεικτού ακέραιου και του τετραγωνικού προγραμματισμού.

Υποστήριξη αλγορίθμων: Αν και είναι αυτόνομα προγράμματα, τόσο το Gurobi όσο και το CPLEX παρέχουν Python API για ενσωμάτωση με βιβλιοθήκες μοντελοποίησης που βασίζονται σε Python, όπως οι PuLP, CVXPY και Pyomo.

Άδεια χρήσης: Το Gurobi και το CPLEX είναι εμπορικά πακέτα λογισμικού που απαιτούν άδειες χρήσης για χρήση σε περιβάλλοντα παραγωγής. Προσφέρουν ακαδημαϊκές άδειες για εκπαιδευτικούς και ερευνητικούς σκοπούς, εφόσον εγκατασταθούν σε υπολογιστή με πανεπιστημιακή IP.

Για τις ανάγκες της συγκεκριμένης περίπτωσης χρησιμοποιήθηκε η βιβλιοθήκη **Pulp** έκδοση 2.8. Ο αλγόριθμος που εγκαθίσταται default με την pulp είναι ο PULP_CBC_CMD. Συνδέεται με τον λύτη CBC (Coin-or branch and cut), ο οποίος είναι ένας ανοικτού κώδικα λύτης LP και MIP προβλημάτων που αναπτύχθηκε από το έργο Coin-OR. Αποτελεί μια αξιόπιστη επιλογή, ιδιαίτερα για προβλήματα ακέραιου προγραμματισμού. Χρησιμοποιεί προηγμένες μεθόδους επίλυσης, όπως τη Branch and Bound, την Branch and Cut, αλλά και ευρετικές μεθόδους. Παράλληλα, εφαρμόζει τεχνικές απλοποίησης και μείωσης των διαστάσεων του προβλήματος, πριν την επίλυση, γεγονός που συντομεύει τον χρόνο ολοκλήρωσης του.

4.3 Σύντομος οδηγός επίλυσης με την Pulp. Η κλάση schedule

Ο τρόπος σύνταξης και επίλυσης των προβλημάτων με την pulp, παρουσιάζεται αναλυτικά στο on-line manual [51]. Παρακάτω θα παρουσιαστεί ένας σύντομος οδηγός που θα βοηθήσει στην επίλυση του προβλήματος.

4.3.1 Διαθέσιμοι solvers

Η βιβλιοθήκη pulp στην python δίνει τη δυνατότητα εμφάνισης μέσω της κονσόλας, των ήδη εγκατεστημένων επιλυτών (solvers) και των διαθέσιμων προς εγκατάσταση (available). Η εμφάνιση τους πραγματοποιείται με τον παρακάτω τρόπο:

```
from pulp import *

from pulp import *

solver_list_available = listSolvers(onlyAvailable=True)
print(solver_list_available)
print("*****************************************************************")
solver_list_all = listSolvers()
print(solver_list_all)
print("*****************************************************************")

CONSLOLE:
['PULP_CBC_CMD']
*****************************************************************
['GLPK_CMD', 'PYGLPK', 'CPLEX_CMD', 'CPLEX_PY', 'GUROBI', 'GUROBI_CMD',
'MOSEK', 'XPRESS', 'XPRESS', 'XPRESS_PY', 'PULP_CBC_CMD', 'COIN_CMD',
'COINMP_DLL', 'CHOCO_CMD', 'MIPCL_CMD', 'SCIP_CMD', 'FSCIP_CMD', 'SCIP_PY',
'HiGHS', 'HiGHS_CMD', 'COPT', 'COPT_DLL', 'COPT_CMD']
*****************************************************************
```

Σχήμα 4.10: Διαθέσιμοι Solvers

Όπως προαναφέρθηκε, ο επιλυτής που χρησιμοποιείται είναι ο 'PULP_CBC_CMD', που όπως φαίνεται στο Σχήμα 4.10 είναι ήδη εγκατεστημένος.

4.3.2 Κλάσεις της Pulp

Οι κλάσεις που δέχεται και εφαρμόζει η βιβλιοθήκη pulp είναι οι εξής :

Πίνακας 4.1: Κλάσεις της βιβλιοθήκης Pulp

- **LpProblem ([name, sense])**	Ένα πρόβλημα γραμμικού προγραμματισμού. name – Το όνομα του προβλήματος sense – Ο στόχος του προβλήματος: (LpMinimize (default) ή LpMaximize.
- **LpVariable(name[, lowBound, upBound, cat])**	Μεταβλητή γραμμικού προγραμματισμού name – Το όνομα της μεταβλητής lowBound – Το κάτω όριο στο εύρος αυτής της μεταβλητής. Η προεπιλογή είναι αρνητικό άπειρο upBound – Το άνω όριο στο εύρος αυτής της μεταβλητής. Η προεπιλογή είναι θετικό άπειρο cat – Η κατηγορία στην οποία ανήκει αυτή η μεταβλητή:, Integer, Binary ή Continuous(default)
- LpAffineExpression([e, constant, name])	Γραμμικός συνδυασμός των LpVariables. Πχ c = LpAffineExpression([(x[0],1), (x[1],-3), (x[2],4)]) => c = 1*x_0 + -3*x_1 + 4*x_2 + 0
- LpConstraint([e, sense, name, rhs])	Ένας περιορισμός (σκληρός). e – μια LpAffineExpression sense – ένα από τα LpConstraintEQ, LpConstraintGE, LpConstraintLE (0, 1, -1 αντίστοιχα) name – το όνομα του περιορισμού rhs – αριθμητική τιμή στόχου του περιορισμού
- LpConstraint.makeElasticSubProblem(constraint, penalty=None, proportionFreeBound=None, proportionFreeBoundList=None)	Μετατρέπει τον περιορισμό σε ελαστικό (μαλακό), εισάγοντας τα χαρακτηριστικά του constraint – Ο LpConstraint στον οποίο βασίζεται ο ελαστικός περιορισμός penalty – ποινή που εφαρμόζεται για παραβίαση (+ve ή -ve) των περιορισμών proportionFreeBound – το αναλογικό όριο (+ve και -ve) στην παραβίαση περιορισμών που είναι απαλλαγμένο από ποινή proportionFreeBoundList – το αναλογικό όριο για την παραβίαση περιορισμών που είναι απαλλαγμένο από ποινή, εκφρασμένο ως λίστα όπου [-ve, +ve]

- FixedElasticSubProblem(constraint, penalty=None, proportionFreeBound=None, proportionFreeBoundList=None)	Μετατρέπει έναν σταθερό περιορισμό της μορφής $\sum_i a_i * x_i = b$ σε ελαστικό. Οι παράμετροι είναι ίδιοι με την προηγούμενη κλάση

Η Αναλυτική περιγραφή των παραμέτρων κάθε κλάσης είναι διαθέσιμη στο documentation της Pulp [51].

4.3.3 Εκκίνηση και ονοματοδοσία του προβλήματος

Για την επίλυση του ILP προβλήματος της συγκεκριμένης μελέτης χρησιμοποιήθηκε μια κλάση της python που ονομάστηκε schedule.

Το πρώτο βήμα είναι η δημιουργία του προβλήματος.

```
# Dimiourgia LP provlimatos elaxistopoiisis***
prob = LpProblem("University_Timetabling", LpMinimize)
```

Σχήμα 4.11: Δημιουργία LpProblem()

Η μέθοδος για την δημιουργία γραμμικού προγραμματισμού είναι η LpProblem, η διαφορά με τον ακέραιο γραμμικό προγραμματισμό έγκειται στο γεγονός πως οι μεταβλητές μας πρέπει να είναι ακέραιοι αριθμοί. Το όνομα του προβλήματος είναι prob και η ετικέτα του "University_Timetabling". Επίσης χρησιμοποιείται η παράμετρος sense για τον καθορισμό του προβλήματος αν είναι ελαχιστοποίησης (LpMinimize) ή μεγιστοποίησης (LpMaximize). Η επιλογή της παραμέτρου sense επηρεάζει αναλόγως το αποτέλεσμα.

4.3.4 Εισαγωγή των μεταβλητών

Από τη στιγμή που δημιουργήθηκε το μοντέλο, ορίζονται οι μεταβλητές απόφασης ως στιγμιότυπα της κλάσης LpVariable. Εάν οι μεταβλητές είναι συνεχείς, θα πρέπει να οριστεί ένα χαμηλότερο όριο με lowBound=0 επειδή η προεπιλεγμένη τιμή είναι αρνητικό άπειρο. Αντίστοιχα, η παράμετρος upBound ορίζει το ανώτερο όριο, εφόσον υπάρχει τέτοιο στο πρόβλημα. Στην συγκεκριμένη περίπτωση δε χρησιμοποιείται κάποιο όριο.

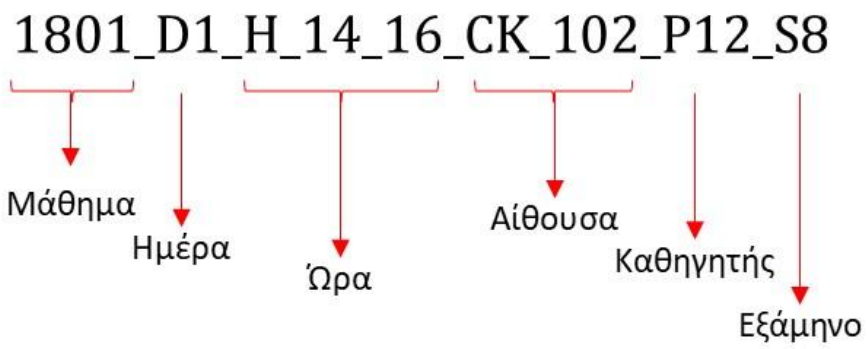
Η προαιρετική παράμετρος cat ορίζει την κατηγορία μιας μεταβλητής απόφασης. Η default τιμή είναι continuous, όμως στην συγκεκριμένη περίπτωση, θα πρέπει να οριστούν οι μεταβλητές ως: cat=binary.

Η δημιουργία της δυαδικής μεταβλητής x[(s,d,h,r,p,sid)] (βλ. παράγραφο 3.2.3) γίνεται με την μορφή λεξικού. Τα κλειδιά είναι το όνομα της μεταβλητής και οι τιμές είναι το 1 ή το 0, κατά περίπτωση.

```
# Dimiourgia diadikis meta gia kathe subject, day, and hour***
x = {}  # x einai ena pinakas opou ta klidia einai upopinakes (s, d, h, r, p, sid) kai oi tim
for _, subject_row in df_dict['SUBJECTS'].iterrows(): #gia kathe grammi sto subjects dld gia
    s = subject_row['SUBJECT_ID'] # apothikeusi anagnoristikou mathimatos sto s
    sid = subject_row['SEMESTER_ID']  # apothikeusi anagnoristikou eksaminou
    for d in df_dict['DAYS']['DAY_ID']: # gia oles tis meres (D1 - D5)
        for h in df_dict['HOURS']['HOUR_ID']: # gia oles tis ores #
            for r_list in [subject_row['POTENTIAL_ROOMS']]: # gia kathe pithani aithousa
                for r in r_list:
                    for p in [subject_row['PROFESSOR_ID']]: # gia kathe kathigiti
                        variable_name_x = "{}_{}_{}_{}_{}_{}".format(s, d, h, r, p, sid)  # d
                        x[(s, d, h, r, p, sid)] = LpVariable(variable_name_x, cat='Binary')
```

Σχήμα 4.12: Δημιουργία της μεταβλητής x

Το όνομα κάθε μεταβλητής που δημιουργείται, παίρνει το όνομα από τις παραμέτρους που είναι χωρισμένες με '_' underscore. Για παράδειγμα μια μεταβλητή x έχει (ενδεικτικό) όνομα: 1801_D1_H_14_16_CK_102_P12_S8 (βλ. Σχήμα 4.13)



Σχήμα 4.13: Δομή της μεταβλητής

4.3.5 Ορισμός της αντικειμενικής συνάρτησης

Η αντικειμενική συνάρτηση (objective_function) ορίζεται σύμφωνα με την εξίσωση (3.7):

```
objective_function = 0
for _, subject_row in df_dict['SUBJECTS'].iterrows(): # gia kathe mathima
    s = subject_row['SUBJECT_ID']    # anagnoristiko mathimatos
    r_list = subject_row['POTENTIAL_ROOMS'] # aithouses pou mporoun na xrisimopoiithoun gia ena mathima
    p = subject_row['PROFESSOR_ID'] # anagnoristiko kathigiti
    sid = subject_row['SEMESTER_ID'] # anagnoristiko eksaminou
    for _, day_row in df_dict['DAYS'].iterrows(): #gia oles tis meres
        d = day_row['DAY_ID']  #anagnoristiko imeras
        for _, hour_row in df_dict['HOURS'].iterrows(): # gia oles tis ores
            h = hour_row['HOUR_ID'] #anagnoristiko wrwn
            hr_weight = hour_row['HR_WEIGHT'] # sintelestis varous gia sigkekrimenes wres
            for r in r_list: # gia kathe pirhani aithousa
                objective_function += lpSum(hr_weight * x[(s, d, h, r, p, sid)])
```

Σχήμα 4.14: Ορισμός της αντικειμενικής συνάρτησης

4.3.6 Εισαγωγή περιορισμών στο πρόγραμμα

Οι σκληροί περιορισμοί δημιουργούνται σύμφωνα με τις εξισώσεις (3.1) έως (3.6) και προστίθενται στο πρόβλημα με την εντολή prob + = .

Σημειώνεται ότι, δεδομένου των πολλών πληροφοριών που υπάρχουν στις μεταβλητές x[(s,d,h,r,p,sid)] και των ιδιοτήτων των dataframes, δεν απαιτείται η εκ νέου δημιουργία των παρακάτω μεταβλητών:

y[(r,d,h)],α[(p,d,h)],b1[(s,b,h)],b2[(s,b,h)] και b3[(s,b,h)].

Οι μεταβλητές αυτές αντικαταστάθηκαν από υποσύνολα των μεταβλητών x[(s,d,h,r,p,sid)], δίνοντας συγκεκριμένες τιμές στα κλειδιά s ή d ή h ή r ή p ή sid. (βλ. περιορισμούς (3.2), (3.3), (3.4) και (3.5)).

```
# Constraint C1): kathe mathima prepei na emfanizetai sto orologio programma toses fores oses einai i diarkeia tou
for _, subject_row in df_dict['SUBJECTS'].iterrows(): # gia kethe mathima
    s = subject_row['SUBJECT_ID']
    duration = subject_row['DURATION']
    variables_for_subject_s = [x[key] for key in x.keys() if key[0] == s] #format(s, d, h, r, p, sid) key[0] einai
    prob += lpSum(variables_for_subject_s) == duration # prosthiki periorismou sto provlima
    # to mathima s tha topothetithei sto programma akrivos gia ton arithmo oron pou xreiazetai
```

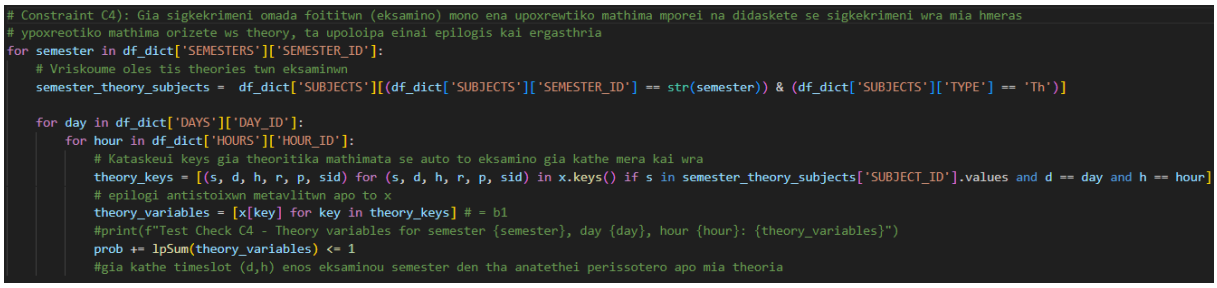
Σχήμα 4.15: Περιορισμός 3.1

```
# Constraint C2): kathe aithousa mporei na anatethei mono se ena mathima se mia sigkekrimeni wra mias hmeras
for _, room_row in df_dict['ROOMS'].iterrows():
    r = room_row['ROOM_ID']
    for _, day_row in df_dict['DAYS'].iterrows():
        d= day_row['DAY_ID']
        for _, hour_row in df_dict['HOURS'].iterrows():
            h = hour_row['HOUR_ID']
            variables_for_rooms = [x[key] for key in x.keys() if key[3] == r and key[1]== d and key[2]== h]
            prob += lpSum(variables_for_rooms) <= 1
            # i sigkekrimeni aithousa r de mporei na anatethei se perissotero apo ena mathima s tautoxrona
```

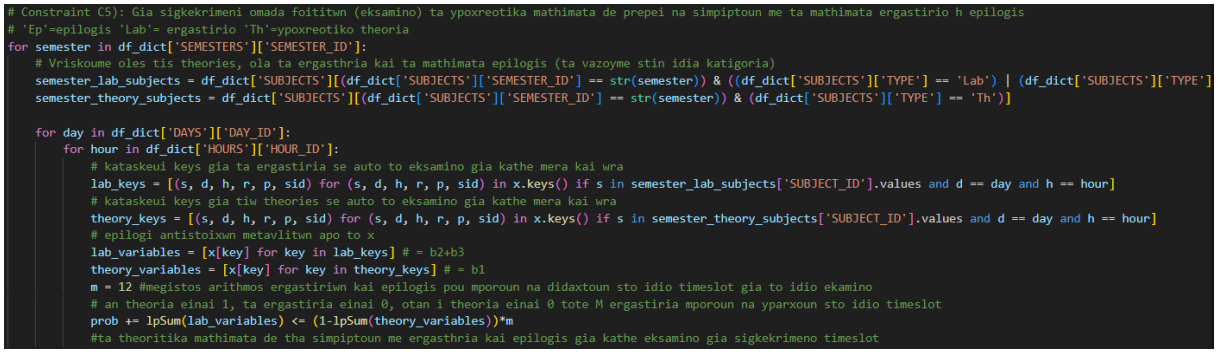
Σχήμα 4.16: Περιορισμός 3.2

```
# Constraint C3): Kathe kathigitis mporei na didaksei mono ena mathima se mia sigkekrimeni wra mias hmeras
for _, day_row in df_dict['DAYS'].iterrows():
    d= day_row['DAY_ID']
    for _, hour_row in df_dict['HOURS'].iterrows():
        h = hour_row['HOUR_ID']
        for _, professor_row in df_dict['PROFESSORS'].iterrows():
            p = professor_row['PROFESSOR_ID']
            variables_for_professors = [x[key] for key in x.keys() if key[4] == p and key[1]== d and key[2]== h]
            prob += lpSum(variables_for_professors) <= 1
            # o kathigitis de tha anatethei se prissotero apo ena mathima gia sigkekrimeni xroniki stigmi
```

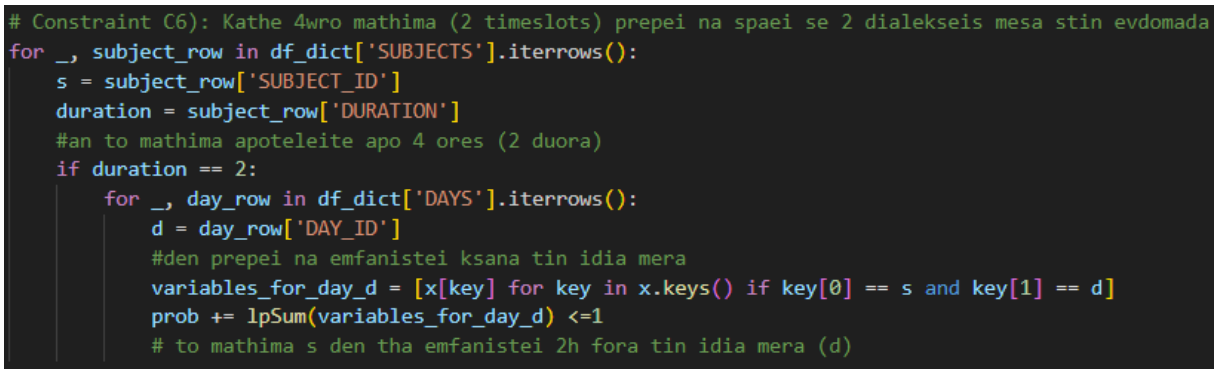
Σχήμα 4.17: Περιορισμός 3.3



Σχήμα 4.18: Περιορισμός 3.4



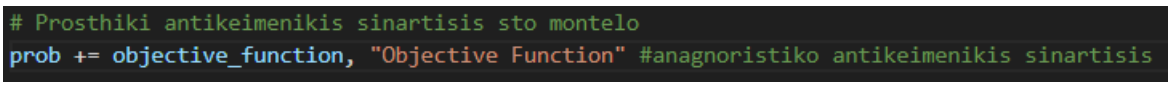
Σχήμα 4.19: Περιορισμός 3.5



Σχήμα 4.20: Περιορισμός 3.6

4.3.7 Προσθήκη την αντικειμενικής συνάρτησης στο πρόβλημα και επίλυση

Στο τελευταίο βήμα πριν την επίλυση, γίνεται η πρόσθεση της αντικειμενικής συνάρτησης στο πρόβλημα.



Σχήμα 4.21: Προσθήκη της αντικειμενικής συνάρτησης

Επίσης πραγματοποιείται αποθήκευση των στοιχείων του προβλήματος σε εξωτερικό αρχείο *.lp το οποίο είναι επεξεργάσιμο από οποιονδήποτε editor. Τα στοιχεία αυτά είναι: η αντικειμενική συνάρτηση,

περιορισμοί και μεταβλητές. Για την εγγραφή και αποθήκευση καλείται η μέθοδος writeLP(“filename”) της κλάσης LpProblem της βιβλιοθήκης Pulp.

Στην συνέχεια για την επίλυση, καλείται η μέθοδος solve() της κλάσης LpProblem. Η solve() χρησιμοποιεί το default αλγόριθμο επίλυσης και ολοκληρώνει όταν θα βρεθεί η optimal λύση. Ο αλγόριθμος μπορεί να οριστεί μέσα στις παραμέτρους της μεθόδου, πχ prob.solve(pulp.PULP_CBC_CMD).

Σημειώνεται ότι, συχνά όταν υπάρχουν μεγάλα προβλήματα (με πολλές μεταβλητές) και μικρή υπολογιστική ισχύ, ο χρόνος επίλυσης ξεπερνάει τον εύλογο χρόνο αναμονής. Θέτοντας όριο στον χρόνο εκτέλεσης, μπορεί να βρεθεί μια λύση που είναι εφικτή (δηλ. καλύπτει όλους τους περιορισμούς), η οποία όμως δεν είναι κατ’ ανάγκη η βέλτιστη. Στις περιπτώσεις αυτές χρησιμοποιείται στην solve() το όριο χρόνου, πχ η prob.solve(pulp.PULP_CBC_CMD(timeLimit=300)) περιορίζει τον χρόνο εκτέλεσης σε 300sec (5min.).

```
#CBC (Coin-or branch and cut) methodos tis pulp, timeLimit xroniko orio se seconskds gia tin evresi kaliteris lisis
#Ektelesi tis epilisis tou montelou
prob.solve(pulp.PULP_CBC_CMD(timeLimit=300))
```

Σχήμα 4.22: Εκτέλεση του επιλυτή

4.3.8 Προετοιμασία αποτελεσμάτων

Μετά την επίλυση, συνέχεια έχει η προετοιμασία των αποτελεσμάτων:

1. Με την print(f"status: {prob.status}, {LpStatus[prob.status]}") τυπώνεται το status της λύσης που μπορεί να είναι optimal (βέλτιστο), infeasible (αδύνατο), indefinite (αόριστο)
2. Δημιουργείται λίστα των μεταβλητών x[(s,d,h,r,p,sid)] που έχουν τιμή 1, για να εισαχθεί στο timetable matrix
3. Τυπώνεται την τιμή της αντικειμενικής συνάρτησης
4. Δημιουργείται το αρχείο TIMETABLE.xlsx με το εβδομαδιαίο πρόγραμμα μαθημάτων και το εβδομαδιαίο πρόγραμμα των καθηγητών

```
print('\n********** STATUS ********************************')
# Elegxos katastasis tis lisis
print(f"status: {prob.status}, {LpStatus[prob.status]}")


print('\n********** VARIABLES ****************************\n')
# Ektiposi twn metavlitwn apofasis xwris tis imeres twn eksaminwn
print ('********** DECISION VARIABLES CHECK WITH VALUE 1 **********')
print("Decision Variables:")
for var in prob.variables():
    if var.varValue == 1 and not var.name.startswith('S'):
        print(f"{var.name}: {var.value()}")

print('\n********** VARIABLES INTO variable_list ***************************')
#metafora twn parapanw metavlitwn stin variable_list
variable_list = [(var.name, var.varValue) for var in prob.variables() if var.varValue == 1 and not var.name.startswith('S')]
print (variable_list)


print('\n********** OBJECTIVE FUNCTION ***************************')
# Timi synartisis stoxou meta tin epilush to provlimatos
print("Objective Function Value:", value(prob.objective),'\n')


print('\n********** TIMETABLE ********************************')
# Simplirosi, emfanisi kai apothikeusi tou orologiou programmatos twn mathimatwn sto TIMETABLE.xlsx/SUBJECTS
timetable, free = populate_subjects_timetable (timetable, free, df_dict['SUBJECTS'], df_dict['ROOMS'], df_dict['DAYS'], df_dict['HOURS'], variable_list)
print('SUBJECTS TIMETABLE')
show_timetable(timetable, df_dict['DAYS'], df_dict['HOURS'], df_dict['ROOMS'])
write_timetable_subjects(timetable, df_dict['DAYS'], df_dict['HOURS'], df_dict['ROOMS'])

# Simplirosi, emfanisi kai apothikeusi tou orologiou programmatos twn kathigitwn in TIMETABLE.xlsx/PROFESSORS
timetable, free = set_up_timetable(df_dict['ROOMS'], df_dict['DAYS'], df_dict['HOURS'])
timetable, free = populate_professors_timetable (timetable, free, df_dict['SUBJECTS'], df_dict['ROOMS'], df_dict['DAYS'], df_dict['HOURS'], variable_list)
print('PROFESSORS TIMETABLE')
show_timetable(timetable, df_dict['DAYS'], df_dict['HOURS'], df_dict['ROOMS'])
write_timetable_professors(timetable, df_dict['DAYS'], df_dict['HOURS'], df_dict['ROOMS'])
```

Σχήμα 4.23: Έξοδος αποτελεσμάτων

Κεφάλαιο 5ο: Ανάλυση Αποτελεσμάτων, συμπεράσματα

5.1 Αποτελέσματα

Για την επίλυση χρησιμοποιήθηκε όριο χρόνου 5 λεπτών στον επιλυτή. Στο χρόνο αυτό έδωσε μια λύση που έχει τιμή στην αντικειμενική συνάρτηση ίση με 210.

Στην κονσόλα παρουσιάζεται αναλυτικά η επίλυση και τα αποτελέσματα:

```
********************** STATUS **********************************
status: 1, Optimal


Result - Stopped on time limit

Objective value:                210.00000000
Lower bound:                    184.333
Gap:                            0.14
Enumerated nodes:               16815
Total iterations:               760707
Time (CPU seconds):             298.13
Time (Wallclock seconds):       298.13

Option for printingOptions changed from normal to all
Total time (CPU seconds):       298.32   (Wallclock seconds):       298.32
```

Σχήμα 5.1: Έξοδος κονσόλας

Το εβδομαδιαίο πρόγραμμα σώζεται στο αρχείο TIMETABLE.xlsx το οποίο τοποθετεί τις 173 συνεδρίες μαθημάτων και τους αντίστοιχους καθηγητές στα φύλλα SUBJECTS και PROFESSORS.

Αυτά παρουσιάζονται στους επόμενους πίνακες (οι χρωματισμοί μπήκαν εκ των υστέρων για την καλύτερη οπτικοποίηση των αποτελεσμάτων).

			LP_201	LP_202	LP_208	LP_210	LP_211	LH_A1	LH_A2	LH_A3	LH_A5	LH_Γ1	LH_Γ2	LH_Γ4	LH_Δ1	LH_Δ2	LH_Δ4	CK_101	CK_102	CK_109	AK_AMF1	AK_AMF2	CK_B1	CK_B2	CK_B3	CK_B4	CK_B5	CK_B6
Monday	H_09_11	-->	1403L5	1304L15	1304L7	1203L2B	1401L1	1204L4B	None	None	None	None	None	None	None	None	None	None	1802	1602	1205A	None	None	None	None	None	None	None
Monday	H_11_13	-->	1403L4	1401L2	1403L2	1304L8	1304L17	None	None	None	None	None	None	None	None	None	None	1801	1202A	None	None	1204B	None	1611	None	None	None	None
Monday	H_14_16	-->	1304L3	1401L4	1304L4	1403L8	1304L10	None	1602L3	1611L4	None	None	None	None	None	None	None	None	None	None	1201B	1203A	None	None	None	None	None	None
Monday	H_16_18	-->	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None
Monday	H_18_20	-->	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None
Tuesday	H_09_11	-->	1205L4B	1203L4B	1205L2B	1203L5B	None	1202L5B	None	None	1204L5B	1202L7B	None	1204L1B	1202L3B	1202L1B	None	None	None	1803	1204A	1401	1602	None	None	None	None	None
Tuesday	H_11_13	-->	1304L12	1403L1	1304L18	1304L5	1304L9	None	None	None	None	None	None	None	None	None	None	None	None	None	1204B	1201A	None	None	None	1601	None	1802
Tuesday	H_14_16	-->	None	None	None	None	None	None	None	None	None	None	None	None	None	1611L3	None	None	None	None	None	1205B	1404	None	None	None	None	None
Tuesday	H_16_18	-->	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	1403
Tuesday	H_18_20	-->	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	1402
Wednesd	H_09_11	-->	1203L1B	None	1205L1B	1205L3B	1203L6B	None	None	1202L4B	None	1202L2B	1204L2B	None	1202L6B	None	1204L6B	1874	1871	None	1203A	1304	1842	1972	1641	None	1948	1812
Wednesd	H_11_13	-->	1304L1	1403L3	1401L7	1304L14	1401L5	1602L7	1602L1	None	None	None	None	1812L2	None	1673L	1602L5	1613	1873	1672	None	1201A	1974	1671	1612	1642	1202B	1643
Wednesd	H_14_16	-->	1203L4A	1403L7	1304L16	1401L3	1304L13	None	None	1611L2	None	None	None	None	None	None	1611L5	None	None	None	1205B	1898	None	None	None	None	None	None
Wednesd	H_16_18	-->	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None
Wednesd	H_18_20	-->	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None
Thursday	H_09_11	-->	1304L2	1304L11	1401L6	1304L6	1403L6	None	None	None	None	None	None	None	None	None	None	1801	None	None	1204A	1203B	None	None	None	None	1601	None
Thursday	H_11_13	-->	1203L1A	None	1205L3A	1672L	1203L6A	1602L2	1202L2A	1204L4A	1202L3A	1204L2A	1602L8	1611L1	None	1202L5A	None	1643	1612	None	1201B	None	1402	1803	1613	1673	None	1642
Thursday	H_14_16	-->	None	1203L3B	None	None	None	1602L6	1611L6	None	None	None	None	None	None	None	None	None	None	None	1401	1205A	None	None	None	None	None	None
Thursday	H_16_18	-->	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None
Thursday	H_18_20	-->	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None
Friday	H_09_11	-->	1205L4A	1203L5A	None	1205L1A	1203L2A	None	1204L6A	None	None	None	1202L4A	None	None	None	1204L1A	1404	1916	1641	1898	None	1974	1873	1874	1948	1842	1202B
Friday	H_11_13	-->	1205L2A	None	1203L3A	None	None	1204L5A	1202L1A	1204L3A	1202L7A	None	1916L	1202L6A	None	1204L3B	1812L1	None	1972	None	None	None	None	1403	1871	None	None	1611
Friday	H_14_16	-->	None	None	None	None	None	1602L9	1602L4	None	None	None	None	None	None	None	None	None	1202A	1671	1304	1203B	None	None	None	None	None	None
Friday	H_16_18	-->	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None
Friday	H_18_20	-->	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None

Σχήμα 5.2: Αρχείο Timetable, Πίνακας Subjects

Σχήμα 5.3: Αρχείο Timetable, Πίνακας Professors

Από την ανάλυση των αποτελεσμάτων προκύπτει ότι:

- Τοποθετούνται όλα τα 138 μαθήματα (173 timeslots, συμπεριλαμβανομένων αυτών που έχουν διάρκεια 2 ωρών)
- Δεν υπάρχουν συγκρούσεις και ισχύουν όλοι οι περιορισμοί που τέθηκαν στο κεφ. 3.2.4.
- Για όλα τα εξάμηνα, τις ημέρες παρουσίας τους στο Πανεπιστήμιο χρησιμοποιούνται από 3 timeslots (από τα συνολικά 5), με εξαίρεση του 2ου εξαμήνου όπου την Τρίτη χρησιμοποιεί 5 timeslots.

Τα πρωινά timeslots χρησιμοποιούνται τις περισσότερες φορές συγκριτικά με τα απογευματινά για συγκεκριμένες τιμές βαρών:

Πίνακας 5.1: Αποτελέσματα χρήσης βαρών

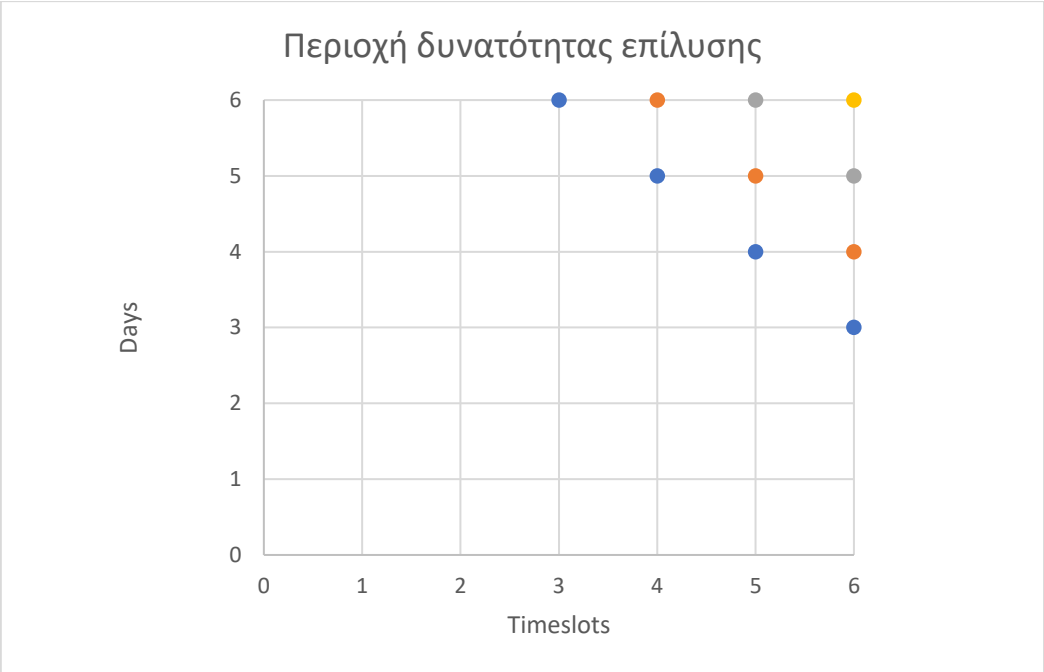
HOUR_ID	Βάρη Wh	Αριθμός εμφανίσεων ανά εβδομάδα
H_09_11	1	67
H_11_13	1	72
H_14_16	2	32
H_16_18	3	1
H_18_20	4	1

Τα παραπάνω αποτελέσματα αποδεικνύουν την λειτουργικότητα των βαρών στο πρόγραμμα.

5.2 Δοκιμές λειτουργίας του προγράμματος

Ώστε να υπάρχει μια καλύτερη εικόνα για την λειτουργείας του προγράμματος πραγματοποιήθηκαν επιμέρους δοκιμές, τροποποιώντας τον χρόνο εκτέλεσης και τον αριθμό ημερών. Στις δοκιμές αυτές τα βάρη στον πίνακα ωρών παρέμειναν σταθερά όπως και ο αριθμός των timeslots ανά ημέρα. Ο αριθμός των μαθημάτων που πρέπει να προγραμματιστούν ορίστηκε όπως αναγράφεται στο κεφάλαιο 3.1.1.2.

Αρχικά οι δοκιμές έδειξαν πως ο επιλυτής βρίσκει λύσει για τις περιπτώσεις όπου ο αριθμός των προγραμματιζόμενων ημέρων θα πρέπει να είναι τουλάχιστον 4 με 5 χρονοθυρίδες/ημέρα, ή για 5 ημέρες ο αριθμός των χρονοθυρίδων/ημέρα θα πρέπει να είναι τουλάχιστον 4. Σε κάθε διαφορετική περίπτωση φαίνεται πως τουλάχιστον ένα μάθημα “δεν χωράει” να μπει στο ωρολόγιο πρόγραμμα με συνέπεια ο επιλυτής να μην μπορεί να βρει κάποια λύση. Ο Συνδυασμός αριθμού μερών και χρονοθυρίδων/ημέρα που μπορεί ο επιλυτής να βρει λύση παρουσιάζεται αναλυτικά παρακάτω στο σχήμα 5.4, με κουκίδες.



Σχήμα 5.4: Τιμές για τις οποίες ο επιλυτής βρίσκει λύση

Σε αυτό το σημείο φαίνεται ότι το πρόγραμμα λειτουργεί σωστά, καθώς με βάση τον πίνακα 5.1, υπάρχει τουλάχιστον μια εμφάνιση προγραμματισμένων μαθημάτων στα Hour_ID 4 και 5, το οποίο συμφωνεί με το σχήμα 5.4.

Παρακάτω παρουσιάζονται δύο πίνακες με την καταγραφή των αποτελεσμάτων του επιλυτή. Οι δοκιμές αυτές αφορούν τις επιδόσεις του επιλυτή προς τον χρόνο εκτέλεσης, με λίγα λόγια θα αναδειχθεί το πόσο χρόνο χρειάζεται ο επιλυτής ώστε να βρει την βέλτιστη λύση. Οι πίνακες που θα ακολουθήσουν (5.2, 5.3) αναγράφουν τις τιμές των δοκιμών για την επίδοση του επιλυτή. Οι δοκιμές πραγματοποιήθηκαν για συγκεκριμένα χρονικά περιθώρια, ξεκινώντας από τα 15 δευτερόλεπτα έως και 2 ώρες. Ο πίνακας 5.2 περιέχει αποτελέσματα για τιμές 4 ημερών και 5 ζώνες-ωρών, ο πίνακας 5.3 για 5 ημερών και 5 χρονοθυρίδων αντίστοιχα, τα βάρη είναι ίδια και στις δύο περιπτώσεις ώστε να είναι συγκρίσιμα τα αποτελέσματα. Ο κάθε πίνακας αποτελείται από 6 στήλες. Αναλυτικά:

- **Time (sec)**: ο μέγιστος χρόνος σε δευτερόλεπτα που ορίστηκε στην συνάρτηση prob.solve() για τον χρόνο της εκτέλεσης του επιλυτή
- **Obj Value**: Η βέλτιστη τιμή της αντικειμενικής συνάρτησης που βρέθηκε
- **Lower Bound**: Το κάτω φράγμα για την τιμή της αντικειμενικής συνάρτησης
- **GAP**: Η διαφορά μεταξύ της βέλτιστης τιμής και του κάτω φράγματος, εκφρασμένη ως ποσοστό
- **Enum. Nodes (Enumerated nodes)**: Ο αριθμός των κόμβων που εξετάστηκαν
- **Iterations**: Ο αριθμός των επαναλήψεων που έγιναν κατά την επίλυση

Πίνακας 5.2: Ανάλυση αποτελεσμάτων για 4 ημέρες και 5 χρονοθυρίδες

Time (sec)	Obj Value	Lower Bound	GAP	Enum. nodes	Iterations
15	**No Solution**	201,833	-	5	6005
20	**No Solution**	201,833	-	68	13018

25	260	201,833	0,29	270	28260
30	260	201,833	0,29	708	51140
45	260	201,833	0,29	1401	69277
60	242	201,833	0,2	2979	130195
120	242	201,833	0,2	9038	267467
300	242	201,833	0,2	18683	913141
600	242	201,833	0,2	35321	1808584
3600	234	201,833	0,16	262683	6043292
7200	234	201,833	0,16	316383	6929087

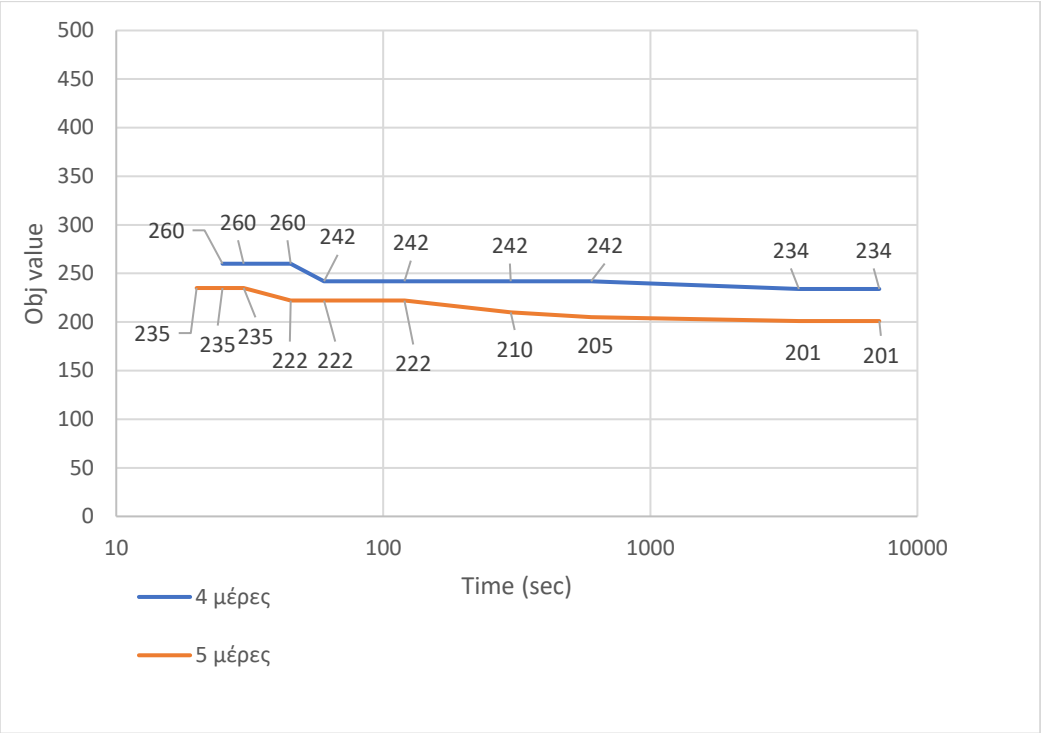
Πίνακας 5.3: Ανάλυση αποτελεσμάτων για 5 ημέρες και 5 χρονοθυρίδες

Time (sec)	Obj Value	Lower Bound	GAP	Enum. nodes	Iterations
15	**No Solution**	184,333		54	6558
20	235	184,333	0,27	420	18968
25	235	184,333	0,27	691	26870
30	235	184,333	0,27	1030	38625
45	222	184,333	0,2	1842	51228
60	222	184,333	0,2	2095	62944
120	222	184,333	0,2	6711	221533
300	210	184,333	0,14	14946	601667
600	205	184,333	0,11	34764	1563753
3600	201	184,333	0,09	322150	7552773
7200	201	184,333	0,09	638124	14073766

Από τα παραπάνω αποτελέσματα προκύπτει ότι:

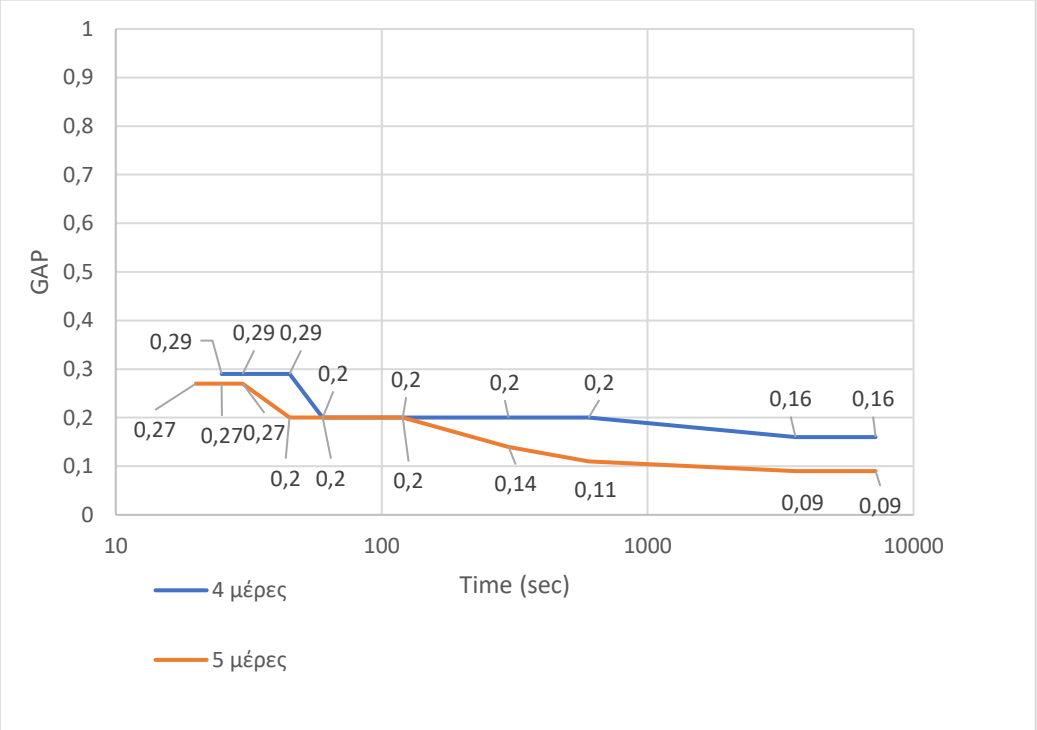
- Ο επιλυτής χρειάζεται έναν αρχικό χρόνο ώστε να δημιουργηθούν οι κόμβοι και να τρέξουν έναν αριθμό επαναλήψεων όπου ο αριθμός αυτός διαφέρει ανάλογα των περιπτώσεων. Και στις δύο περιπτώσεις χρειάζεται τουλάχιστον 15-25 δευτερόλεπτα ώστε να μπορέσει να βρει κάποια λύση.
- Στο 1ο λεπτό μετά τα 15 δευτερόλεπτα ο επιλυτής βρίσκει ήδη λύσεις για το πρόβλημα. Στα 2-5 λεπτά οι λύσεις φαίνονται είναι ικανοποιητικές η τιμή που δίνει ο επιλυτής είναι πιο κοντά στην βέλτιστη λύση συγκρίνοντας με το 1ο λεπτό της εκτέλεσης. Αφήνοντας το πρόγραμμα να τρέξει για έως και 2 ώρες, η τιμή του επιλυτή φτάνει ακόμα πιο κοντά στη βέλτιστη λύση, αλλά ο ρυθμός της σύγκλησης είναι μειωμένος σε σχέση με τα πρώτα 5 λεπτά.
- Όσο το πρόβλημα έχει μικρότερα περιθώρια (στην περίπτωσή μας 4 από 5 ημέρες), φαίνεται πως η τιμή της αντικειμενικής συνάρτησης και το κάτω όριο αυξάνεται, κάτι που είναι αναμενόμενο διότι αναγκαστικά πρέπει να χρησιμοποιηθούν χρονοθυρίδες με μεγαλύτερο βάρος. Ο ρυθμός μείωσης του GAP είναι χαμηλότερος (βλ. Σχήμα 3) που σημαίνει ότι στον ίδιο χρόνο δυσκολεύεται να πλησιάσει το κάτω όριο.

Παρακάτω παρουσιάζονται τα αποτελέσματα σε γράφημα. Το Σχήμα 5.5 συγκρίνει την τιμή της αντικειμενικής συνάρτησης του επιλυτή κατά τον χρόνο εκτέλεσης.



Σχήμα 5.5: Βέλτιστη τιμή της αντικειμενικής συνάρτησης στον χρόνο εκτέλεσης

Το Σχήμα 5.6 συγκρίνει την σύγκλιση του επιλυτή προς το κάτω φράγμα στη διάρκεια του χρόνου εκτέλεσης.



Σχήμα 5.6: Διαφορά από το κάτω φράγμα σε ποσοστό

Το πλήθος των διάφορων δοκιμών που μπορεί να πραγματοποιηθεί πάνω στο πρόβλημα ποικίλλει ανάλογα την φύση των περιπτώσεων που θέλει κανείς να εξετάσει. Οι παραπάνω δοκιμές επιλέχθηκαν ώστε να δοκιμαστεί η λειτουργία του προγράμματος και η απόδοση του επιλυτή προς τον χρόνο εκτέλεσης. Για να μπορέσουν τα αποτελέσματα να είναι συγκρίσιμα και κατανοητά η τροποποίηση των δεδομένων εισόδου αφορούσε αλλαγές μόνο προς τον αριθμό εξεταζόμενων ημερών.

Από τα παραπάνω φαίνεται πως ο χρόνος που απαιτείται για την εύρεση των βέλτιστων λύσεων εξαρτάται πάντα από τα δεδομένα εισόδου που θα δοθούν. Γενικότερα όμως όσο περισσότερος χρόνος δοθεί για την εκτέλεση του επιλυτή, η λύση που θα δώσει θα είναι πιο κοντά στην βέλτιστη.

Επιπλέον αν τροποποιηθούν τα δεδομένα των ωρών και μπουν ίδιες τιμές σε όλα τα βάρη, ο επιλυτής δεν θα έχει την δυνατότητα να βελτιστοποιήσει την συνάρτηση κόστους και θα σταματήσει στην 1η λύση που θα βρει χωρίς να εξαντλήσει όλο το χρονικό περιθώριο που θα του δοθεί. Σε αυτήν την περίπτωση το πρόβλημα μετατρέπεται από βελτιστοποίησης σε επιλογής, όλες οι λύσεις που πληρούν τους περιορισμούς, δίνουν πάντα την ίδια τιμή στην συνάρτηση κόστους (αντικειμενική συνάρτηση).

Στην περίπτωση που ο επιλυτής δε θα μπορέσει να βρει κάποια λύση, όπως αναφέρθηκε και στο Σχήμα 5.4, ή στον Πίνακα 5.3 για (15sec) ο επιλυτής θα εξαντλήσει όλο το χρονικό περιθώριο που αρχικά είχε δοθεί και θα σταματήσει με σφάλμα ότι δεν βρέθηκε κάποια λύση.

5.3 Σύνοψη και συμπεράσματα

Στην παρούσα εργασία παρουσιάστηκε μια διατύπωση ενός προβλήματος χρονοπρογραμματισμού όπως αυτό υπάρχει σε πολλές σχολές της χώρας. Για να παραμείνει χαμηλή η πολυπλοκότητα του προβλήματος, χρησιμοποιήθηκαν πολύ γενικοί περιορισμοί, το οποίο καθιστά σχετικά εύκολη προσαρμογή και για άλλα εκπαιδευτικά ιδρύματα. Ωστόσο απαιτεί την κατάλληλη επεξεργασία δεδομένων εισόδου στο πρόγραμμα. Είναι κατάλληλο για να λύσει συγκεκριμένα προβλήματα όπου είναι ήδη γνωστός ο αριθμός των μαθημάτων και εργαστηρίων, δηλαδή εξαρχής θα πρέπει να είναι γνωστό ποιος καθηγητής θα διδάξει ποιο μάθημα ή και ποιο εργαστήριο (το οποίο θεωρείται ξεχωριστό μάθημα) με βάση των αριθμό φοιτητών που θα παρακολουθήσουν το μάθημα, την διαθεσιμότητα των καθηγητών με βάση τις ώρες που πρέπει να διδάξει ο καθένας. Όλα τα παραπάνω καθιστούν δύσκολη την διαδικασία προ επεξεργασίας. Παρ' όλα αυτά είναι πλήρως λειτουργικό με δυνατότητες παραμετροποίησης στα παρακάτω σημεία:

- Η αντικειμενική συνάρτηση με την χρήση των βαρών δίνει την δυνατότητα να οριστούν επιθυμητές ώρες Διδασκαλίας.
- Δίνεται κατηγοριοποίησης των αιθουσών.
- Δίνεται η δυνατότητα στα μαθήματα με βάση τον τύπο του μαθήματος, να οριστεί ένα μάθημα ως θεωρία ώστε να μην διδαχτεί την ίδια ώρα άλλο μάθημα του ίδιου εξαμήνου.

Κλείνοντας, σε κάθε περίπτωση η τελική επιλογή αφορά τους υπευθύνους της σχολής, οι οποίοι θα πρέπει να λαμβάνουν υπόψιν τους περιορισμούς, τις επιθυμίες και της διαθεσιμότητες της σχολής και των εμπλεκόμενων.

5.4 Μελλοντικές επεκτάσεις

Όπως έχει προαναφερθεί, για λόγους διατήρησης χαμηλής πολυπλοκότητας οι περιορισμοί που χρησιμοποιήθηκαν είναι οι βασικοί. Μελλοντικά το πρόγραμμα θα μπορούσε να επεκταθεί ώστε να γίνει πιο χρήσιμο και λειτουργικό στα παρακάτω σημεία:

Διαχωρισμός των χρονοθυρίδων σε μονόωρες συνεδρίες ώστε να μπορεί να προσαρμόζεται και σε άλλες σχολές που ίσως χρησιμοποιούν μονόωρες ή τρίωρες συνεδρίες μαθημάτων.

Προσθήκη χαλαρών περιορισμών για τους καθηγητές, για παράδειγμα το πόσες ή ποιες ώρες ή μέρες ένας καθηγητής μπορεί να βρίσκεται στην σχολή είτε διδάσκει εκείνη την ώρα είτε όχι.

Προσθήκη χαλαρών περιορισμών που θα μειώνουν τις κενές ώρες μέσα στην μέρα για του καθηγητές και για τους φοιτητές ανά εξάμηνο.

Προσθήκη χαλαρών περιορισμών με βάση τις ημέρες που θα είναι επιθυμητές να γίνονται ή να αποφεύγονται τα μαθήματα. Με αυτόν τον τρόπο θα υπάρχει η δυνατότητα ανά εξάμηνο είτε να συσσωρευτεί το πρόγραμμα και να μειωθούν οι μέρες παρουσίας ανά εβδομάδα για τους φοιτητές είτε κάποιες συγκεκριμένες ημέρες να προτιμώνται για να χρονοπρογραμματίζονται τα μαθήματα.

Επέκταση για το παραγόμενο αρχείο εξόδου ώστε να υπάρχει η δυνατότητα επιλογής για το πως θα βγαίνει το πρόγραμμα, είτε σε μορφή excel ή pdf είτε σε μορφή ημερολογίου (.ics) ξεχωριστά για τις παρακάτω περιπτώσεις:

- Ανά εξάμηνο φοίτησης
- Ανά φοιτητή προσωπικά με βάση την δήλωση μαθημάτων
- Ανά καθηγητή με βάση τα μαθήματα που θα διδάξει
- Ανά αίθουσα με βάση τα μαθήματα που θα φιλοξενεί

Οι περιπτώσεις επεκτάσεων ποικίλλουν αναλόγως των αναγκών της κάθε σχολής. Οι παραπάνω προτάσεις αποτελούν βασικές βελτιώσεις για ποιοτικότερα αποτελέσματα και για την δυνατότητα πιο γενικής χρήσης του προγράμματος.

ΒΙΒΛΙΟΓΡΑΦΙΑ

Βιβλιογραφία

[1] Νενεκούμη, Α. (2016). Μοντέλα Γραμμικού Προγραμματισμού για το Πρόβλημα του Περιοδεύοντος Πωλητή, Διπλωματική εργασία, ΑΠΘ, Θεσσαλονίκη Ιανουάριος 2016

[2] Αρουσαλίδου, Σ. (2019). Αλγόριθμοι για προβλήματα ακέραιου γραμμικού προγραμματισμού και εφαρμογές, Διπλωματική εργασία, ΕΑΠ, Κιλκίς Μάϊος 2019

[3] Παπαδόπουλος, Α. (2013). Ακέραιος Προγραμματισμός Μέθοδοι και Εφαρμογές. Διπλωματική Εργασία, ΕΜΠ, Ιούλιος 2013

[4] Sierksma, G. & Yori, Zwols. (2015). Linear and Integer Optimization: Theory and Practice, Third Edition. CRC Press. p. 1.

[5] Alexander, Schrijver. (1998). Theory of Linear and Integer Programming. John Wiley & Sons. pp. 221–222

[6] Dantzig, G. (1982). "Reminiscences about the origins of linear programming". Operations Research Letters. 1(2): 43–48

[7] Leonid, Khachiyan. (1979). "A Polynomial Algorithm for Linear Programming". DokladyAkademiiNauk SSSR. 224 (5): 1093–1096

[8] Dantzig, G. B. (1951b), Application of the simplex method to a transportation problem, in: Activity Analysis of Production and Allocation (Tj. C. Koopmans, ed.), Wiley, New York, 1951, pp. 359--373. [98,142,275,378]

[9] Dantzig, G., Fulkerson, R., and Johnson, S. (1954), Solution of a large-scale traveling salesman problem, Operations Research 2 (1954) 393- 410. [37 1,3781.

[10] Hirsch, W. M. and Dantzig, G. B. (1954), Thefixed charge problem, Report P-648, The RAND Corporation, Santa Monica, Cal., 1954. [378]

[11] Dantzig, G. B., Fulkerson, D. R., and Johnson, S. M. (1959), On a linear-programming, combinatorial approach to the traveling-salesman problem, Options Research 7 (1959) 58-66. [370,378]

[12] Heller, I. (1955c), Neighbor relations on the convex of cyclic permutations (Abstract), Bulletin of the American Mathematical Society 61 (1955) 440 [full article: Pacific Journal of Mathematics 6 (1956) 467-4771. [378]

[13] Heller, I. (1953a), On the problem of shortest path between points. I, Bulletin of the American Mathematical Society 59 (1953) 55 1. [378]

[14] Heller, I. (1953b), On the problem of shortest path between points. 11, Bulletin of the American Mathematical Society 59 (1953) 551-552. [378]

[15] Heller, I. (1955a), On the travelling salesman's problem, in: Proceedings of the Second Symposium in Linear Programming, Vol.ZZ (H. A. Antosiewitz, ed.), National Bureau of Standards, Washington, 1955, pp. 643-665. [378]

[16] Heller, I. (1955b), Geometric characterization of cyclic permutations (Abstract), Bulletin of the American Mathematical Society 61 (1955) 227. [378]

[17] Kuhn, H. W. (1955b), On certain convex polyhedral (Abstract), Bulletin of the American Mathematical Society 61 (1955) 557-558. [378]

[18] Norman, R. Z. (1955), On the convex polyhedral of the symmetric traveling salesman problem (Abstract), Bulletin of the American Mathematical Society 61 (1955) 559. [378]

[19] Gaddum, J. W., Hoffman, A. J., and Sokolowsky, D. (1954), On the solution of the caterer problem, Naval Research Logistics Quarterly (1954) 223-229. [378]

[20] Jacobs, W. (1954), The caterer problem, Naval Research Logistics Quarterly 1 (1954) 154- 165. [378]

[21] Prager, W. (1956-7), On the caterer problem, Management Science 3 (1956-7) 15-23. C3781

[22] Dantzig, G. B. and Fulkerson, D. R. (1956), On the max-flow min-cut theorem of networks, in: Linear Inequalities and Related Systems (H. W. Kuhn and A. W. Tucker, eds.), Princeton University Press, Princeton, N.J., 1956, pp. 215-221. C274.3781
[23] Elias, P., Feinstein, A., and Shannon, C. E. (1956), A note on the maximum flow through a network, ZRE Transactions on Information Theory IT 2 (1956) 117-119. [97, 115, 275, 378)
[24] Ford, Jr, L. R. and Fulkerson, D. R. (1956-7), Solving the transportation problem, Management Science 3 (1956-7) 24-32. 113781
[25] Ford, Jr, L. R. and Fulkerson, D. R. (l956), Maximal flow through a network, Canadian Journal of Mathematics 8 (1956) 399--404. [97, 1 15, 275, 3781
[26] Ford, Jr. L. R. and Fulkerson, D. R. (1957), A simple algorithm for finding maximal network flows and an application to the Hitchcock problem, Canadian Journal of Mathematics 9 (1957)210-218. [154, 279, 3781]
[27] Egerviry E. (1958), Bemerkungen zum Transportproblem, MTW Mitteilungen 5 (1958) 278- 284. [3781]
[28] Koopmans, Tj. C. and Beckmann, M. (1957), Assignment problems and the location of economic activities, Econometrica 25 (1957) 53-76. [378]
[29] Kuhn, H. W. (1955a), The Hungarian method for the assignment problem, Naval Research Logistics Quarterly 2 (1955) 83-97. [378]
[30] Kuhn, H. W. (1955b), On certain convex polyhedral (Abstract), Bulletin of the American Mathematical Society 61 (1955) 557-558. [378]
[31] Motzkin, T. S. (1956), The assignment problem, in: Numerical Analysis (Proceedings of Symposia in Applied Mathematics Vol. VI; J. H. Curtiss, ed.), McGraw-Hill, New York, 1956, pp. 109- 125 [reprinted in: Theodore S. Motzkin: Selected Papers (D. Cantor, B. Gordon and B. Rothschild, eds.), Birkhauser, Boston, 1983, pp. I21 -1 371. [273,378]
[32] Motzkin, T. S. and Straus, E. G. (1956), Some combinatorial extremum problems, Proceedings of' the American Mathematical Society 7 (1956) 1014-1 021 [reprinted in: Theodore S. Motzkin: Selected Papers (D. Cantor, B. Gordon and B. Rothschild, eds.), Burkhouse, Boston, 1983, pp. 279-2861. [378]
[33] Munkres, J. (1957), Algorithms for the assignment and transportation problems, Journal of the Society, for Industrial and Applied Mathematics 5 (1957) 32-38. [378]
[34] Land, A. and Doig, A. (1960) An Automatic Method of Solving Discrete Programming Problems. Ecometrics, 28, 497-520.
[35] Paull, A. E. and Walter, J. R. (1955), The trim problem: an application of linear programming to the manufacture of newsprint paper (Abstract), Econometrical 23(1955) 336. [378]
[36] Παπότη, Α. (2003). Χρονοπρογραμματισμός Ωρολόγιού Προγράμματος Μαθημάτων Πανεπιστημιακού Τμήματος, Διπλωματική Εργασία, ΑΠΘ Τμήμα Πληροφορικής
[37] Δήμου, Ε. (2008). Υποδείγματα μαθηματικού προγραμματισμού για το σχεδιασμό του ωρολογίου προγράμματος ενός εκπαιδευτικού ιδρύματος, Διπλωματική Εργασία, Πανεπιστήμιο Πατρών, Τμήμα Διοίκησης Επιχειρήσεων
[38] Cacchiani, V., Caprara, A., Roberti, R., and Toth, P. (2013), A new lower bound for curriculum-based course timetabling, Computers & Operation Research, 40(10), 2466- 2476.
[39] Bellio, R., Ceschia, S., Gaspero, L. D., Schaerf, A., and Urli, T. (2016), Feature-based tuning of simulated annealing applied to the curriculum-based coursetimetabling problem, Computers & Operation Research, 65, 84-92.
[40] Schaerf, A. (1999). A Survey of Automated Timetabling. Artificial Intelligence Review, 13, 87-127.
[41] Yuan, Y. (2023). A Mixed-Integer Linear Programming Model for University Course Timetabling Problems, MSc Thesis, Wageningen University - Department of Social Sciences
[42] Daskalaki, S., Birbas, T. & Housos, E. (2004). An integer programming formulation for a case study in university timetabling. European Journal of Operational Research, 153(1), 117-135.
[43] Prabodanie, R. A. R. (2017). An Integer Programming Model for a Complex University Timetabling Problem: A Case Study. *Industrial Engineering and Management Systems*. Korean Institute of Industrial Engineers.

[44] Ismayilova, A. A., Sağir, M., and Gasimov, R. N. (2007), A multiobjective faculty-course-time slot assignment problem with preferences, Mathematical and Computer Modelling, 46(7-8), 1017-1029.
[45] Broek, J. V. D., Hurkens, C., and Woeginger, G. (2009), Timetabling problems at the TU Eindhoven, European Journal of Operational Research, 196(3), 877- 885
[46] Miranda, J., Rey, P. A., and Robles, J. M. (2012), UdpSkeduler: A web architecture-based decision support system for course and classroom scheduling, Decision Support Systems, 52(2), 505-513.
[47] Vermuyten, H., Lemmens, S., Marques, I., and Belien, J. (2015), Developing compact course timetables with optimized student flows, European Journal of Operational Research, 251(2), 651-661.
[48] Bakir, M. A. and Akshop, C. (2008), A 0–1 integer programming approach to a university timetabling problem, Hacettepe Journal of Mathematics and Statistics, 37(1), 41-55.
[49] Babaei, H., Karimpour, J. & Hadidi, A. (2015). A survey of approaches for university course timetabling problem. Computers and Industrial Engineering, 86, 43-59.
[50] Mirhassani, S. & Habibi, F. (2013). Solution approaches to the course timetabling problem. Artificial Intelligence Review, 39(2), 133-149.
[51] "The full PuLP function documentation" Accessed: April 20, 2024 [Online]. Available: https://coin-or.github.io/pulp/
[52] "Pyomo documentation and examples" Accessed: April 20, 2024 [Online]. Available: https://www.pyomo.org/documentation
[53] "SciPy Fundamental algorithms for scientific computing in Python" Accessed: April 20, 2024 [Online]. Available: https://scipy.org
[54] "CVXPY Convex optimization, for everyone." Accessed: April 20, 2024 [Online]. Available: https://www.cvxpy.org
[55] "GUROBI optimization" Accessed: April 20, 2024 [Online]. Available: https://www.gurobi.com
[56] "IBM ILOG CPLEX Optimization Studio" Accessed: April 20, 2024 [Online]. Available: https://www.ibm.com/products/ilog-cplex-optimization-studio
[57] "PANDAS Documentation" Accessed: April 20, 2024 [Online]. Available: https://pandas.pydata.org/docs/index.html
[58] "OPENPYXL - A Python library to read/write Excel xlsx/xlsm files" Accessed: April 20, 2024 [Online]. Available: https://openpyxl.readthedocs.io/en/stable/#documentation

ΠΑΡΑΡΤΗΜΑ Α : ΔΕΔΟΜΕΝΑ

Όλα τα δεδομένα αρχεία εισόδου είναι καταχωρημένα αρχείο excel, το οποίο αποτελείται από 5 φύλλα

Φύλλο ROOMS: Πληροφορίες σχετικά με τις αίθουσες την σχολής:

Παράρτημα δεδομένα: Πίνακας 1 ROOMS

ROOM_ID	ROOM_NAME	ROOM_TYPE	CAPACITY	DIRECTION	ROOM_CATEGORY
LP_201	201	Lab	25	Pliroforikis	LP
LP_202	202	Lab	25	Pliroforikis	LP
LP_208	208	Lab	25	Pliroforikis	LP
LP_210	210	Lab	30	Pliroforikis	LP
LP_211	211	Lab	25	Pliroforikis	LP
LH_A1	A1	Lab	25	Ilektronikon	LH
LH_A2	A2	Lab	25	Ilektronikon	LH
LH_A3	A3	Lab	25	Ilektronikon	LH
LH_A5	A5	Lab	25	Ilektronikon	LH
LH_Γ1	G1	Lab	25	Ilektronikon	LH
LH_Γ2	G2	Lab	25	Ilektronikon	LH
LH_Γ4	G4	Lab	25	Ilektronikon	LH
LH_Δ1	D1	Lab	25	Ilektronikon	LH
LH_Δ2	D2	Lab	25	Ilektronikon	LH
LH_Δ4	D4	Lab	25	Ilektronikon	LH
CK_101	101	Th	70	Koin	CK
CK_102	102	Th	70	Koin	CK
CK_109	109	Th	70	Koin	CK
AK_AMF1	AMF1	Amfitheater	120	Koin	CA
AK_AMF2	AMF2	Amfitheater	120	Koin	CA
CK_B1	B1	Th	70	Koin	CK
CK_B2	B2	Th	70	Koin	CK
CK_B3	B3	Th	70	Koin	CK
CK_B4	B4	Th	70	Koin	CK
CK_B5	B5	Th	70	Koin	CK
CK_B6	B6	Th	70	Koin	CK

Φύλλο SEMESTERS: Αφορά τις πληροφορίες των τρέχων εξαμήνων:

Παράρτημα δεδομένα: Πίνακας 2 SEMESTERS

SEMESTER	TMHMA	SEMESTER_ID
2	A	S2A
2	B	S2B
4		S4
6		S6
8		S8

Φύλλο DAYS: Αφορά της τις πληροφορίες για τις ημέρες:

Παράρτημα δεδομένα: Πίνακας 3 DAYS

DAY_NAME	DAY_ID
Monday	D1
Tuesday	D2
Wednesday	D3
Thursday	D4
Friday	D5

Φύλλο HOURS: Αφορά της τις πληροφορίες για τις ζώνες-ωρών (χρονοθυρίδες) της ημέρας:

Παράρτημα δεδομένα: Πίνακας 4 HOURS

START	END	HOUR_ID	HR_WEIGHT
9	11	H_09_11	1
11	13	H_11_13	1
14	16	H_14_16	2
16	18	H_16_18	3
18	20	H_18_20	4

Φύλλο PROFESSORS: Ο πίνακας των καθηγητών που θα διδάσκουν:

Παράρτημα δεδομένα: Πίνακας 5 PROFESSORS

PROFESSOR_NAME	PROFESSOR_ID
Ioannidis L.	P1
Ji Chenyang	P2
Amanatiadis D.	P3
Ampatzis Z.	P4
Antoniou E.	P5
Asdre K.	P6
Vassios V.	P7
Vitsas V.	P8
Giakoumis A.	P9
Goulianas K.	P10
Delimaras V.	P11
Ilioudis Ch.	P12
Intzes I.	P13
Ioannidou M.	P14
Iosifidis A.	P15
Karagiannis I.	P16
Karamitropoulos L.	P17
Keramopoulos E.	P18
Kioskeridis I.	P19
Kokkonis G.	P20
Kostoglou V.	P21
Kostakis R.	P22
Marmorkos I.	P23
Mesodiakaki A.	P24
Mizeli X.	P25

Mpamnios	P26
Mpratsas X.	P27
Ougiaroglou S.	P28
Papadopoulou	P29
Papakostas	P30
Salampasis A.	P31
Sidiropoulos A.	P32
Stamatis D.	P33
Tektonidis D.	P34
Tzekis	P35
Tokatlidis X.	P36
Touliou E.	P37
Tsiak	P38
Tsirogiannis A.	P39
Charalampidis Ch.	P40
Chatzimisios P.	P41
Chatzopoulos A.	P42

Φύλλο SUBJECTS: Ο πίνακας όλων των μαθημάτων που θα πρέπει να χρονοπρογραμματιστουν:

Παράρτημα δεδομένα: Πίνακας 6 SUBJECTS

SEMESTER	SUBJECT_ID	SUBJECT	Κατεύθυνση	CHARACTER	TYPE	DURATION	THEORY	LAB	TMHMA	ROOM_CATEGORY	PROFESSOR_N	PROFESSOR_ID	SEMESTER_ID
2	1201A	Μαθηματικά ΙΙ		Υποχ	Th	2	4		A	CA	Tzekis	P35	S2A
2	1202A	Μετρήσεις και Κυκλώματα Εναλλασσόμενου Ρεύματος		Υποχ	Th	2	4		A	CK	Kioskeridis I.	P19	S2A
2	1202L1A	Μετρήσεις και Κυκλώματα Εναλλασσόμενου Ρεύματος			Lab	1		2	A	LH	Ampatzis Z.	P4	S2A
2	1202L2A	Μετρήσεις και Κυκλώματα Εναλλασσόμενου Ρεύματος			Lab	1		2	A	LH	Ampatzis Z.	P4	S2A
2	1202L3A	Μετρήσεις και Κυκλώματα Εναλλασσόμενου Ρεύματος			Lab	1		2	A	LH	Mizeli X.	P25	S2A
2	1202L4A	Μετρήσεις και Κυκλώματα Εναλλασσόμενου Ρεύματος			Lab	1		2	A	LH	Mizeli X.	P25	S2A
2	1202L5A	Μετρήσεις και Κυκλώματα Εναλλασσόμενου Ρεύματος			Lab	1		2	A	LH	Tokatlidis X.	P36	S2A
2	1202L6A	Μετρήσεις και Κυκλώματα Εναλλασσόμενου Ρεύματος			Lab	1		2	A	LH	Tokatlidis X.	P36	S2A
2	1202L7A	Μετρήσεις και Κυκλώματα Εναλλασσόμενου Ρεύματος			Lab	1		2	A	LH	Charalampidis Ch.	P40	S2A
2	1203A	Τεχνική Συγγραφή, Παρουσίαση και Ορολογία Ξένης Γλώσσας		Υποχ	Th	2	4		A	CA	Chatzimisios P.	P41	S2A
2	1203L1A	Τεχνική Συγγραφή, Παρουσίαση και Ορολογία Ξένης Γλώσσας			Lab	1		2	A	LP	Ji Chenyang	P2	S2A
2	1203L2A	Τεχνική Συγγραφή, Παρουσίαση και Ορολογία Ξένης Γλώσσας			Lab	1		2	A	LP	Ji Chenyang	P2	S2A
2	1203L3A	Τεχνική Συγγραφή, Παρουσίαση και Ορολογία Ξένης Γλώσσας			Lab	1		2	A	LP	Ji Chenyang	P2	S2A
2	1203L4A	Τεχνική Συγγραφή, Παρουσίαση και Ορολογία Ξένης Γλώσσας			Lab	1		2	A	LP	Ji Chenyang	P2	S2A
2	1203L5A	Τεχνική Συγγραφή, Παρουσίαση και Ορολογία Ξένης Γλώσσας			Lab	1		2	A	LP	Touliou E.	P37	S2A
2	1203L6A	Τεχνική Συγγραφή, Παρουσίαση και Ορολογία Ξένης Γλώσσας			Lab	1		2	A	LP	Touliou E.	P37	S2A
2	1204A	Σχεδίαση Ψηφιακών Συστημάτων		Υποχ	Th	2	4		A	CA	Mpamnios	P26	S2A
2	1204L1A	Σχεδίαση Ψηφιακών Συστημάτων			Lab	1		2	A	LH	Vassios V.	P7	S2A
2	1204L2A	Σχεδίαση Ψηφιακών Συστημάτων			Lab	1		2	A	LH	Vassios V.	P7	S2A
2	1204L3A	Σχεδίαση Ψηφιακών Συστημάτων			Lab	1		2	A	LH	Vassios V.	P7	S2A
2	1204L4A	Σχεδίαση Ψηφιακών Συστημάτων			Lab	1		2	A	LH	Tsiak	P38	S2A
2	1204L5A	Σχεδίαση Ψηφιακών Συστημάτων			Lab	1		2	A	LH	Tsiak	P38	S2A
2	1204L6A	Σχεδίαση Ψηφιακών Συστημάτων			Lab	1		2	A	LH	Tsiak	P38	S2A
2	1205A	Αντικειμενοστρεφής Προγραμματισμός		Υποχ	Th	2	4		A	CA	Asdre K.	P6	S2A
2	1205L1A	Αντικειμενοστρεφής Προγραμματισμός			Lab	1		2	A	LP	Tektonidis D.	P34	S2A
2	1205L2A	Αντικειμενοστρεφής Προγραμματισμός			Lab	1		2	A	LP	Tektonidis D.	P34	S2A

2	1205L3A	Αντικειμενοστρεφής Προγραμματισμός		Lab	1		2	A	LP	Stamatis D.	P33	S2A
2	1205L4A	Αντικειμενοστρεφής Προγραμματισμός		Lab	1		2	A	LP	Stamatis D.	P33	S2A
2	1201B	Μαθηματικά ΙΙ	Υποχ	Th	2	4		B	CA	Tzekis	P35	S2B
2	1202B	Μετρήσεις και Κυκλώματα Εναλλασσόμενου Ρεύματος	Υποχ	Th	2	4		B	CK	Kioskeridis I.	P19	S2B
2	1202L1B	Μετρήσεις και Κυκλώματα Εναλλασσόμενου Ρεύματος		Lab	1		2	B	LH	Ampatzis Z.	P4	S2B
2	1202L2B	Μετρήσεις και Κυκλώματα Εναλλασσόμενου Ρεύματος		Lab	1		2	B	LH	Ampatzis Z.	P4	S2B
2	1202L3B	Μετρήσεις και Κυκλώματα Εναλλασσόμενου Ρεύματος		Lab	1		2	B	LH	Mizeli X.	P25	S2B
2	1202L4B	Μετρήσεις και Κυκλώματα Εναλλασσόμενου Ρεύματος		Lab	1		2	B	LH	Mizeli X.	P25	S2B
2	1202L5B	Μετρήσεις και Κυκλώματα Εναλλασσόμενου Ρεύματος		Lab	1		2	B	LH	Tokatlidis X.	P36	S2B
2	1202L6B	Μετρήσεις και Κυκλώματα Εναλλασσόμενου Ρεύματος		Lab	1		2	B	LH	Tokatlidis X.	P36	S2B
2	1202L7B	Μετρήσεις και Κυκλώματα Εναλλασσόμενου Ρεύματος		Lab	1		2	B	LH	Charalampidis Ch.	P40	S2B
2	1203B	Τεχνική Συγγραφή, Παρουσίαση και Ορολογία Ξένης Γλώσσας	Υποχ	Th	2	4		B	CA	Chatzimisios P.	P41	S2B
2	1203L1B	Τεχνική Συγγραφή, Παρουσίαση και Ορολογία Ξένης Γλώσσας		Lab	1		2	B	LP	Ji Chenyang	P2	S2B
2	1203L2B	Τεχνική Συγγραφή, Παρουσίαση και Ορολογία Ξένης Γλώσσας		Lab	1		2	B	LP	Ji Chenyang	P2	S2B
2	1203L3B	Τεχνική Συγγραφή, Παρουσίαση και Ορολογία Ξένης Γλώσσας		Lab	1		2	B	LP	Ji Chenyang	P2	S2B
2	1203L4B	Τεχνική Συγγραφή, Παρουσίαση και Ορολογία Ξένης Γλώσσας		Lab	1		2	B	LP	Ji Chenyang	P2	S2B
2	1203L5B	Τεχνική Συγγραφή, Παρουσίαση και Ορολογία Ξένης Γλώσσας		Lab	1		2	B	LP	Touliou E.	P37	S2B
2	1203L6B	Τεχνική Συγγραφή, Παρουσίαση και Ορολογία Ξένης Γλώσσας		Lab	1		2	B	LP	Touliou E.	P37	S2B
2	1204B	Σχεδίαση Ψηφιακών Συστημάτων	Υποχ	Th	2	4		B	CA	Mpamnios	P26	S2B
2	1204L1B	Σχεδίαση Ψηφιακών Συστημάτων		Lab	1		2	B	LH	Vassios V.	P7	S2B
2	1204L2B	Σχεδίαση Ψηφιακών Συστημάτων		Lab	1		2	B	LH	Vassios V.	P7	S2B
2	1204L3B	Σχεδίαση Ψηφιακών Συστημάτων		Lab	1		2	B	LH	Mizeli X.	P25	S2B
2	1204L4B	Σχεδίαση Ψηφιακών Συστημάτων		Lab	1		2	B	LH	Mizeli X.	P25	S2B
2	1204L5B	Σχεδίαση Ψηφιακών Συστημάτων		Lab	1		2	B	LH	Tsiak	P38	S2B
2	1204L6B	Σχεδίαση Ψηφιακών Συστημάτων		Lab	1		2	B	LH	Tsiak	P38	S2B
2	1205B	Αντικειμενοστρεφής Προγραμματισμός	Υποχ	Th	2	4		B	CA	Asdre K.	P6	S2B
2	1205L1B	Αντικειμενοστρεφής Προγραμματισμός		Lab	1		2	B	LP	Tektonidis D.	P34	S2B
2	1205L2B	Αντικειμενοστρεφής Προγραμματισμός		Lab	1		2	B	LP	Tektonidis D.	P34	S2B
2	1205L3B	Αντικειμενοστρεφής Προγραμματισμός		Lab	1		2	B	LP	Stamatis D.	P33	S2B
2	1205L4B	Αντικειμενοστρεφής Προγραμματισμός		Lab	1		2	B	LP	Stamatis D.	P33	S2B

4	1304	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων	Υποχ	Th	2	4		CA	Kokkonis G.	P20	S4
4	1304L1	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Amanatiadis D.	P3	S4
4	1304L2	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Amanatiadis D.	P3	S4
4	1304L3	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Amanatiadis D.	P3	S4
4	1304L4	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Delimaras V.	P11	S4
4	1304L5	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Delimaras V.	P11	S4
4	1304L6	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Karagiannis I.	P16	S4
4	1304L7	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Karagiannis I.	P16	S4
4	1304L8	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Karagiannis I.	P16	S4
4	1304L9	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Karagiannis I.	P16	S4
4	1304L10	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Kokkonis G.	P20	S4
4	1304L11	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Kokkonis G.	P20	S4
4	1304L12	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Kokkonis G.	P20	S4
4	1304L13	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Tsirogiannis A.	P39	S4
4	1304L14	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Tsirogiannis A.	P39	S4
4	1304L15	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Tsirogiannis A.	P39	S4
4	1304L16	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Karagiannis I.	P16	S4
4	1304L17	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Kokkonis G.	P20	S4
4	1304L18	Οργάνωση και Αρχιτεκτονική Υπολογιστικών Συστημάτων		Lab	1		2	LP	Tsiak	P38	S4
4	1401	Συστήματα Διαχείρισης Βάσεων Δεδομένων	Υποχ	Th	2	4		CA	Ougiaroglou S.	P28	S4
4	1401L1	Συστήματα Διαχείρισης Βάσεων Δεδομένων		Lab	1		2	LP	Karamitropou los L.	P17	S4
4	1401L2	Συστήματα Διαχείρισης Βάσεων Δεδομένων		Lab	1		2	LP	Karamitropou los L.	P17	S4
4	1401L3	Συστήματα Διαχείρισης Βάσεων Δεδομένων		Lab	1		2	LP	Karamitropou los L.	P17	S4
4	1401L4	Συστήματα Διαχείρισης Βάσεων Δεδομένων		Lab	1		2	LP	Karamitropou los L.	P17	S4
4	1401L5	Συστήματα Διαχείρισης Βάσεων Δεδομένων		Lab	1		2	LP	Karamitropou los L.	P17	S4
4	1401L6	Συστήματα Διαχείρισης Βάσεων Δεδομένων		Lab	1		2	LP	Karamitropou los L.	P17	S4

4	1401L7	Συστήματα Διαχείρισης Βάσεων Δεδομένων			Lab	1		2	LP	Ougiaroglou S.	P28	S4
4	1402	Τηλεπικοινωνιακά Συστήματα		Υποχ	Th	2	4		CK	Ioannidou M.	P14	S4
4	1403	Εισαγωγή στα Λειτουργικά Συστήματα		Υποχ	Th	2	4		CK	Sidiropoulos A.	P32	S4
4	1403L1	Εισαγωγή στα Λειτουργικά Συστήματα			Lab	1		2	LP	Sidiropoulos A.	P32	S4
4	1403L2	Εισαγωγή στα Λειτουργικά Συστήματα			Lab	1		2	LP	Sidiropoulos A.	P32	S4
4	1403L3	Εισαγωγή στα Λειτουργικά Συστήματα			Lab	1		2	LP	Ioannidis L.	P1	S4
4	1403L4	Εισαγωγή στα Λειτουργικά Συστήματα			Lab	1		2	LP	Ioannidis L.	P1	S4
4	1403L5	Εισαγωγή στα Λειτουργικά Συστήματα			Lab	1		2	LP	Ioannidis L.	P1	S4
4	1403L6	Εισαγωγή στα Λειτουργικά Συστήματα			Lab	1		2	LP	Ioannidis L.	P1	S4
4	1403L7	Εισαγωγή στα Λειτουργικά Συστήματα			Lab	1		2	LP	Ioannidis L.	P1	S4
4	1403L8	Εισαγωγή στα Λειτουργικά Συστήματα			Lab	1		2	LP	Ioannidis L.	P1	S4
4	1404	Ηλεκτρονικά Κυκλώματα		Υποχ	Th	2	4		CK	Papakostas	P30	S4
6	1601	Τεχνητή Νοημοσύνη	ΚΟΙΝ	Υποχ	Th	2	4		CK	Mpratsas X.	P27	S6
6	1602	Ενσωματωμένα συστήματα	ΚΟΙΝ	Υποχ	Th	2	4		CK	Papadopoulou	P29	S6
6	1602L1	Ενσωματωμένα συστήματα			Lab	1		2	LH	Giakoumis A.	P9	S6
6	1602L2	Ενσωματωμένα συστήματα			Lab	1		2	LH	Giakoumis A.	P9	S6
6	1602L3	Ενσωματωμένα συστήματα			Lab	1		2	LH	Giakoumis A.	P9	S6
6	1602L4	Ενσωματωμένα συστήματα			Lab	1		2	LH	Delimaras V.	P11	S6
6	1602L5	Ενσωματωμένα συστήματα			Lab	1		2	LH	Delimaras V.	P11	S6
6	1602L6	Ενσωματωμένα συστήματα			Lab	1		2	LH	Delimaras V.	P11	S6
6	1602L7	Ενσωματωμένα συστήματα			Lab	1		2	LH	Charalampidis Ch.	P40	S6
6	1602L8	Ενσωματωμένα συστήματα			Lab	1		2	LH	Charalampidis Ch.	P40	S6
6	1602L9	Ενσωματωμένα συστήματα			Lab	1		2	LH	Charalampidis Ch.	P40	S6
6	1611	Σύνθεση ηλεκτρονικών κυκλωμάτων	ΗΛΕΣ	Υπ-Επ	Th	2	4		CK	Chatzopoulos A.	P42	S6
6	1611L1	Σύνθεση ηλεκτρονικών κυκλωμάτων			Lab	1		2	LH	Delimaras V.	P11	S6
6	1611L2	Σύνθεση ηλεκτρονικών κυκλωμάτων			Lab	1		2	LH	Delimaras V.	P11	S6
6	1611L3	Σύνθεση ηλεκτρονικών κυκλωμάτων			Lab	1		2	LH	Delimaras V.	P11	S6

6	1611L4	Σύνθεση ηλεκτρονικών κυκλωμάτων			Lab	1		2		LH	Charalampidis Ch.	P40	S6
6	1611L5	Σύνθεση ηλεκτρονικών κυκλωμάτων			Lab	1		2		LH	Charalampidis Ch.	P40	S6
6	1611L6	Σύνθεση ηλεκτρονικών κυκλωμάτων			Lab	1		2		LH	Charalampidis Ch.	P40	S6
6	1612	Κβαντική Υπολογιστική	ΗΛΕΣ	Επ	Ep	2	4			CK	Marmorkos I.	P23	S6
6	1613	Μεθοδ. Σχεδ. Μικροηλ. Κυκλ.ΜΣΜΚ	ΗΛΕΣ	Επ	Ep	2	4			CK	Intzes I.	P13	S6
6	1641	Αριθμητικές Μέθοδοι	ΠΛΗΡ	Υπ-Επ	Th	2	4			CK	Antoniou E.	P5	S6
6	1642	Προηγμένα Θέματα Αλληλεπίδρασης (Προγραμματισμός Κινητών Συσκευών)	ΠΛΗΡ	Επ	Ep	2	4			CK	Keramopoulos E.	P18	S6
6	1643	Διοίκηση Έργων	ΠΛΗΡ	Επ	Ep	2	4			CK	Kostoglou V.	P21	S6
6	1671	Μικροκυμ. Τεχν. και Τηλεπισκόπηση	ΚΟΙΝ	Επ	Ep	2	4			CK	Ioannidou M.	P14	S6
6	1672	Οπτοηλεκτρονική και Οπτικές Επικ.	ΚΟΙΝ	Επ	Ep	1	2			CK	Papadopoulou	P29	S6
6	1672L	Οπτοηλεκτρονική και Οπτικές Επικ.			Lab	1		2		LP	Papadopoulou	P29	S6
6	1673	Συστήματα Μέσων Μαζικής Επικοινωνίας	ΚΟΙΝ	Επ	Ep	1	2			CK	Kostakis R.	P22	S6
6	1673L	Συστήματα Μέσων Μαζικής Επικοινωνίας			Lab	1		2		LH	Kostakis R.	P22	S6
8	1801	Ασφάλεια πληροφοριακών συστ.	ΚΟΙΝ	Υποχ	Th	2	4			CK	Ilioudis Ch.	P12	S8
8	1802	Αρχές και Μέθοδοι Μηχαν. Μάθησης	ΚΟΙΝ	Υποχ	Th	2	4			CK	Goulianas K.	P10	S8
8	1803	Διαδίκτυο των Πραγμάτων	ΚΟΙΝ	Υποχ	Th	2	4			CK	Chatzimisios P.	P41	S8
8	1812	Μετατροπείς Ισχύος	ΗΛΕΣ	Επ	Ep	1	2			CK	Kioskeridis I.	P19	S8
8	1812L1	Μετατροπείς Ισχύος			Lab	1		2		LH	Kioskeridis I.	P19	S8
8	1812L2	Μετατροπείς Ισχύος			Lab	1		2		LH	Vassios V.	P7	S8
8	1916	*Συστήματα Μετρήσεων Υποβοηθούμενων από Η/Υ	ΗΛΕΣ	Επ	Ep	1	2			CK	Intzes I.	P13	S8
8	1916L	*Συστήματα Μετρήσεων Υποβοηθούμενων από Η/Υ			Lab	1		2		LH	Intzes I.	P13	S8
8	1842	Διαδικτ. Υπηρ. Προστιθέμενης Αξίας	ΠΛΗΡ	Επ	Ep	2	4			CK	Asdre K.	P6	S8
8	1948	Ανάπτυξη Ολοκληρωμένων Πληρ.Συστ.	ΠΛΗΡ	Επ	Ep	2	4			CK	Salampasis A.	P31	S8
8	1871	Ασύρματα Δίκτυα	ΚΟΙΝ	Επ	Ep	2	4			CK	Vitsas V.	P8	S8
8	1873	Προηγμένα Θέματα Δικτύων	ΚΟΙΝ	Επ	Ep	2	4			CK	Vitsas V.	P8	S8
8	1874	Συστήματα Κινητών Επικοινωνιών	ΚΟΙΝ	Επ	Ep	2	4			CK	Iosifidis A.	P15	S8
8	1898	Ελεύθερη Επιλογή Β	ΚΟΙΝ	Επ	Ep	2	4			CA	Kostakis R.	P22	S8
8	1972	*Δικτύωση Καθορισμένη από Λογισμικό	ΚΟΙΝ	Επ	Ep	2	4			CK	Mesodiakaki A.	P24	S8

8	1974	*Δορυφορικές Επικοινωνίες	KOIN	Επ	Ερ	2	4		CK	Mesodiakaki A.	P24	S8	

ΠΑΡΑΡΤΗΜΑ Β : ΚΩΔΙΚΑΣ

Παρακάτω ακολουθεί ο κώδικας από το αρχείο “auxiliary.py”

```
# -*- coding: utf-8 -*-

import pandas as pd
from openpyxl import Workbook # Library for writing to excel workbook
from openpyxl import load_workbook


def upload_data(excel_file_path):
    """
    Anevazei ta dedomena pou uparxoun sto excel
    param: h diadromi tou arxeiou
    return: ta fulla excel me tin morfi dataframes opou key --> onoma fullou excel kai value --> o pinakas
    """

    # anoigma arxeiou excel me ti vivliothiki Pandas
    excel_file = pd.ExcelFile(excel_file_path)
    # Dimiourgia adeiou pinaka pou tha periexei ta dedomena apo ta filla tou excel arxeiou
    df_dict = {}
    # prospelasi sta fulla tou excel kai metafora ton dedomenon
    for sheet_name in excel_file.sheet_names:
        df = pd.read_excel(excel_file, sheet_name=sheet_name)
        df_dict[sheet_name] = df
    # kleisimo tou arxeioy excel
    excel_file.close()

    return df_dict
```

```python
def add_potential_rooms(df_subjects, df_rooms):
    """
    Prosthetei mia stili sta SUBJECTS
    pou periexei mia lista me ROOMS simfona me to room_category gia to sigkekrimeno mathima.
    :param: 2 dataframes
    :return: SUBJECTS me mia prostheti stili 'POTENTIAL_ROOMS'.
    """
    # prosthiki neas stilis kai arxikopoihsh kenhs listas gia tin apothikeusi pithanwn aithouswn
    df_subjects['POTENTIAL_ROOMS'] = [[] for _ in range(len(df_subjects))]

    # Gia kathe grammi tou DataFrame twn Subjects
    for index_s, subject_row in df_subjects.iterrows():

        # Gia kathe grammi tou Dataframe twn Rooms
        for _, room_row in df_rooms.iterrows():
            # Check an i katigoria tou Room tairiazei me tin antistixi katigoria gia to mathima
            if (room_row['ROOM_CATEGORY'] == subject_row['ROOM_CATEGORY']):
                df_subjects.at[index_s, 'POTENTIAL_ROOMS'].append(room_row['ROOM_ID'])

    return df_subjects


def add_professor_hours(df_subjects, df_professors):
    """
    Prosthetei mia stili sto PROFESSORS pou periexei upologismenes xronothirides (1= 2wres) ana kathigiti
    simfona me to programma
    :param: 2 dataframes
    :return: PROFESSORS me mia prostheti stili 'HOURS_PER_WEEK'.
    """
```

```
    # Arxikopoihsh stilis me 0
    df_professors['HOURS_PER_WEEK']= 0

    # Gia kathe mathima vriskei ton kathigiti kai tis ores pou apaitountai
    for _, subject_row in df_subjects.iterrows():
        prof_id = subject_row['PROFESSOR_ID']
        duration = subject_row['DURATION']
        # Prosthetei ston kathe kathigiti tis ores pou didaskei stin nea stili
        df_professors.loc[df_professors['PROFESSOR_ID'] == prof_id, 'HOURS_PER_WEEK'] += duration
        # prosvasi stis times me xrisi etiketwn

    return df_professors


def print_dataframes(dictionary_of_dataframes):
    # Print ta arxeia pou fortothikan apo to excel sto DataFrame mazi me tis stiles pou prostethikan
    print ("**************** COMPLETE DATA ****************")
    for sheet_name, df in dictionary_of_dataframes.items():
        print(f"\nDataFrame from sheet '{sheet_name}':")
        print(df)


def set_up_timetable(df_rooms, df_days, df_hours):
    """
    Orizei ton pinaka orologiou programmatos kai to leksiko pou apothikeuei ta eleuthera pedia apo ton pinaka opou:
        -matrix: lista me stixeia xronothiridon days_times_hours. Ta stoixia einai listes me arithmo stoixion analoga
        me ton arithmo aithouswn
        - free: mia lista apo pedia (x,y), opou x metritis apo 0 - (d_times_hours -1)
        kai y metritis apo 0 - (numbers of rooms -1)

    :param num_of_rooms, arithmos twn classrooms
```

```
    :      days_times_hours,  days multiplied with hours/day
    :return: matrix, free
    """
    num_of_rooms = len(df_rooms)
    days_times_hours = len(df_days) * len(df_hours)
    matrix = [[None for x in range(num_of_rooms)] for y in range(days_times_hours)]
    free = []

    # arxokopoiisi ths free listas ws ola ta pedia apo to matrix
    for i in range(len(matrix)):
        for j in range(len(matrix[i])):
            free.append((i, j))

    return matrix, free


def populate_subjects_timetable (matrix, free, df_subjects, df_rooms, df_days, df_hours, variable_list):
    """
    Simplironei to orologio programma me ta mathimata.
    Akoma diagrafei tis mi diathesimes theseis apo ton pinaka free.
    :param: matrix, free, df_subjects, df_rooms, df_days, df_hours
            variable_list: mia lista pou periexei zeugaria (variable_name, variable_value),
            simfona me tin lish tou montelou
    :return: matrix, free
    """
    initial_slots = len(free)
    slots = initial_slots

    for v in variable_list: # (variable_name, variable_value)
        # Gia oles tis meres, ores, aithouses k mathimata krataei osa parousiazontai sth variable_list
        for index_d, row in df_days.iterrows():
```

```
            d = row['DAY_ID']
            if d in v[0]: # an i mera d perilamvanetai sto 1o stixeio v opou x = (name, value)
                for index_h, row in df_hours.iterrows():
                    h = row['HOUR_ID']
                    if str(h) in v[0]:
                        for index_r, row in df_rooms.iterrows():
                            r = row['ROOM_ID']
                            if str(r) in v[0]:
                                for index_s, row in df_subjects.iterrows():
                                    s = row['SUBJECT_ID']
                                    # anazitisi mathimatos s na einai to idio apo to zeugos v prin tin _
                                    if str(s) == v[0][:v[0].find('_')]:
                                        # elegxos an to sigkekrimeno pedio einai keno
                                        if matrix[index_d * len(df_hours) + index_h][index_r] is None:
                                            #an einai kataxorise ton arithmo mathimatos s sto matrix kai antistoixa aferesi apo to free
                                            matrix[index_d * len(df_hours) + index_h][index_r] = s
                                            free.remove((index_d * len(df_hours) + index_h, index_r))
                                            slots -= 1
                                        # ean to pedio tou pinaka den einai keno tote uparxei sigkrousi kai tha tipothei minima sfalmatos
                                        else:
                                            print('Potential conflict:', matrix [index_d*len(df_days['DAY_ID'])+ index_h][index_r], ' vs ',
s, ' on day ', d, ', hour ', h, ', room ', r, '. *********')
    print ('********************** TOTAL WEAKLY TIMESLOTS: ', initial_slots-slots, ' **********************')
    return matrix, free


def populate_professors_timetable (matrix, free, df_subjects, df_rooms, df_days, df_hours, variable_list):
    """
    Simplironei to orologio programma gia tous kathigites
    :param: matrix, free, df_subjects, df_rooms, df_days, df_hours
            variable_list: mia lista pou periexei zeugaria (variable_name, variable_value), simfona me tin lish tou montelou
```

```
    :return: matrix, free
    """
    initial_slots = len(free)
    slots = initial_slots
    for v in variable_list:
        # Gia oles tis meres, ores, aithouses k kathigites krataei osa parousiazontai sth variable_list
        for index_d, row in df_days.iterrows():
            d = row['DAY_ID']
            if d in v[0]:
                for index_h, row in df_hours.iterrows():
                    h = row['HOUR_ID']
                    if str(h) in v[0]:
                        for index_r, row in df_rooms.iterrows():
                            r = row['ROOM_ID']
                            if r in v[0]:
                                for index_s, row in df_subjects.iterrows():
                                    p = row['PROFESSOR_ID']
                                    # variables 1  1643_D1_H_11_13_CK_101_P21_S6: 1.0
                                    # *****anazitisi tou kathigiti p na einai to idio apo to zeugos v apo tin proteleutaia thesi
                                    if str(p) == v[0][v[0].rfind('_', 0, v[0].rfind('_')) + 1 : v[0].rfind('_')]:
                                        # an h sigkekrimeni thesi sto matrix einai keni tote grafei ton kwdiko tou kathigiti ston
matrix
                                        if matrix [index_d*len(df_hours['HOUR_ID'])+ index_h][index_r] ==
None:
                                            matrix [index_d*len(df_hours['HOUR_ID'])+ index_h][index_r] = p
                                            free.remove((index_d*len(df_hours['HOUR_ID'])+ index_h, index_r))
                                            slots = slots -1
                                        # diaforetika ektiponei to parakatw minima sfalmatos

    print ('********************** TOTAL WEAKLY TIMESLOTS FOR PROFESSORS: ', initial_slots-slots, ' **********************')
    return matrix, free
```

```
def show_timetable(matrix, df_days, df_hours, df_rooms):
    """
    Ektiponei ton pinaka me to orologio programma
    Dimiourgei ektiposi tou pinaka se morfi pinaka opou oi stiles einai to room_id
    kai oi grammes me vasi a) day_name kai b) hour_id
    """
    days = []
    hours = []
    rooms = []
    for day in df_days['DAY_NAME']:
        days.append(day)
    for hour in df_hours['HOUR_ID']:
        hours.append(hour)
    for room in df_rooms['ROOM_ID']:
        rooms.append(room)

    # Diamorfosi kefalidas
    headlineD = '|' + ' ' * 9
    headlineH = '|' + ' ' * 7
    headlineA = '|' + ' ' * 5
    headline = headlineD + headlineH + headlineA # Diamorfosi kenou xwrou prin tin ektiposi
    # Prosthese se kathe aithousa stin eksodo me aristeri stixisi se apostasi 13 spaces
    for room in rooms:
        headline += '{:<13}'.format(room)
    print (headline)
    print()

    # Simplirosi tou orologiou programmatos
    day_counter = 0
```

```
    hour_counter = 0
    for i in range(len(matrix)):
        day = days[day_counter]
        hour = hours[hour_counter]
        # ektiposi meres, ores kai ena velos stin arxi tis grammis
        print('{:<10} {:<8} ->  '.format(day, hour), end='')
        for j in range(len(matrix[i])):
            print('{:<12} '.format(str(matrix[i][j])), end='')
        print()
        hour_counter += 1
        if hour_counter == len(hours):
            hour_counter = 0
            day_counter += 1
            print()


# Eggrafi lisis se arxeio ********************************************************
def write_timetable_subjects(matrix, df_days, df_hours, df_rooms):
    """
    Apothikeush orologiou programmatos se arxeio excel, me onoma TIMETABLE sto fullo SUBJECTS
    """
    days = df_days['DAY_NAME'].tolist()
    hours = df_hours['HOUR_ID'].tolist()
    rooms = df_rooms['ROOM_ID'].tolist()

    # Dimiourgeia arxeiou
    wb = Workbook()
    ws = wb.active
    ws.title = 'SUBJECTS'

    # Dimiourgia kefalidas
```

```
blank = ' '
headline = ['{:<20} '.format(blank)]
headline.append('{:<20} '.format(blank))
headline.append('{:<12} '.format(blank))

for room in rooms:
    headline.append('{:<13}'.format(room))
ws.append(headline)
# prosthiki kenis grammis
ws.append([''] * (len(headline) + 1))

# eggrafi tou programmatos
day_counter = 0
hour_counter = 0
for i in range(len(matrix)):
    day = days[day_counter]
    hour = hours[hour_counter]
    row = ['{:<20} '.format(day)]
    row.append('{:<20} '.format(hour))
    row.append('  -->  ')
    for j in range(len(matrix[i])):
        row.append('{:<12} '.format(str(matrix[i][j])))
    ws.append(row)
    hour_counter += 1
    if hour_counter == len(hours):
        hour_counter = 0
        day_counter += 1
        ws.append([''] * (len(headline) + 1))

# apothikeusi arxeiou
wb.save('TIMETABLE.xlsx')
```

```
def write_timetable_professors(matrix, df_days, df_hours, df_rooms):
    """
    Apothikeush orologiou programmatos kathigitwn se arxeio excel, me onoma TIMETABLE sto fullo PROFESSORS
    """
    #  Metatropi dataframes se antistixes listes python
    days = df_days['DAY_NAME'].tolist()
    hours = df_hours['HOUR_ID'].tolist()
    rooms = df_rooms['ROOM_ID'].tolist()

    # Fortosi tou uparxontos arxeiou
    wb = load_workbook('TIMETABLE.xlsx')

    # Dimiourgia neou fillou PROFESSOR sto arxeio
    ws = wb.create_sheet('PROFESSORS')

    # Dimiourgia kefalidas
    blank = ' '
    headline = ['{:<20} '.format(blank)]
    headline.append('{:<20} '.format(blank))
    headline.append('{:<12} '.format(blank))

    for room in rooms:
        headline.append('{:<13}'.format(room))
    ws.append(headline)
    ws.append([''] * (len(headline) + 1))

    # Eggrafi tou programmatos
    day_counter = 0
    hour_counter = 0
```

```
for i in range(len(matrix)):
    day = days[day_counter]
    hour = hours[hour_counter]
    row = ['{:<20} '.format(day)]
    row.append('{:<20} '.format(hour))
    row.append('  -->  ')
    for j in range(len(matrix[i])):
        row.append('{:<12} '.format(str(matrix[i][j])))
    ws.append(row)
    hour_counter += 1
    if hour_counter == len(hours):
        hour_counter = 0
        day_counter += 1
        ws.append([''] * (len(headline) + 1))
# Apothikeusi sto arxeio
wb.save('TIMETABLE.xlsx')
```

Παρακάτω ακολουθεί ο κώδικας από το αρχείο “schedule.py”

```
# -*- coding: utf-8 -*-

from pulp import *
from auxiliary import *

# Epilogi arxeiou excel gia DataInput. To arxeio apoteleitai apo 6 fulla: ROOMS, PROFESSORS, SUBJECTS, GROUPS, DAYS, HOURS.
excel_file_path = 'InputDataFile.xlsx'

# Metafora dedomenwn sto programma
df_dict = upload_data(excel_file_path)

# Prosthiki pithanwn aithouswn apo to auxiliary.py
df_dict['SUBJECTS'] = add_potential_rooms(df_dict['SUBJECTS'], df_dict['ROOMS'])

# Prosthiki twn sinolikwn evdomadiaiwn wrwn didaskalias se kathigites apo to auxiliary.py
df_dict['PROFESSORS'] = add_professor_hours(df_dict['SUBJECTS'], df_dict['PROFESSORS'])

# Ektiposi olwn twn fortomenwn dedomewn sto df_dict gia arxiko elegxo
print("1st Check print all dataframes from df_dict")
print_dataframes(df_dict)

# Dimiourgia tou timetable matrix
timetable, free = set_up_timetable(df_dict['ROOMS'], df_dict['DAYS'], df_dict['HOURS'])

# Dimiourgia LP provlimatos elaxistopoiisis***
prob = LpProblem("University_Timetabling", LpMinimize)

# Dimiourgia diadikis meta gia kathe subject, day, and hour***
```

```
x = {}  # x einai ena pinakas opou ta klidia einai upopinakes (s, d, h, r, p, sid) kai oi times einai ta onomata twn metavlitwn
for _, subject_row in df_dict['SUBJECTS'].iterrows(): #gia kathe grammi sto subjects dld gia kathe mathima
    s = subject_row['SUBJECT_ID'] # apothikeusi anagnoristikou mathimatos sto s
    sid = subject_row['SEMESTER_ID']  # apothikeusi anagnoristikou eksaminou
    for d in df_dict['DAYS']['DAY_ID']: # gia oles tis meres (D1 - D5)
        for h in df_dict['HOURS']['HOUR_ID']: # gia oles tis ores #
            for r_list in [subject_row['POTENTIAL_ROOMS']]: # gia kathe pithani aithousa
                for r in r_list:
                    for p in [subject_row['PROFESSOR_ID']]: # gia kathe kathigiti
                        variable_name_x = "{}_{}_{}_{}_{}_{}".format(s, d, h, r, p, sid)  # dimiourgia format metavlitis
(1201B_D2_H_16_18_AK_AMF1_P35_S2B)
                        x[(s, d, h, r, p, sid)] = LpVariable(variable_name_x, cat='Binary')  # dimiourgia diadikis metavlitis meso pulp
'LpVariable'

#elegxos metavlitwn
#print ('check leksikou x ***********************************************')
#for key, var in x.items():
#    print(f"Variable {key}: {var}")  #7400 metavlites gia to sigkekrimeno arxeio endeiktika

#************* orismos antikeimenikis sinartisis (objective function) ****************************
objective_function = 0
for _, subject_row in df_dict['SUBJECTS'].iterrows(): # gia kathe mathima
    s = subject_row['SUBJECT_ID']    # anagnoristiko mathimatos
    r_list = subject_row['POTENTIAL_ROOMS'] # aithouses pou mporoun na xrisimopoiithoun gia ena mathima
    p = subject_row['PROFESSOR_ID'] # anagnoristiko kathigiti
    sid = subject_row['SEMESTER_ID'] # anagnoristiko eksaminou
    for _, day_row in df_dict['DAYS'].iterrows(): #gia oles tis meres
        d = day_row['DAY_ID']  #anagnoristiko imeras
        for _, hour_row in df_dict['HOURS'].iterrows(): # gia oles tis ores
            h = hour_row['HOUR_ID'] #anagnoristiko wrwn
            hr_weight = hour_row['HR_WEIGHT'] # sintelestis varous gia sigkekrimenes wres
```

```
            for r in r_list: # gia kathe pirhani aithousa
                objective_function += lpSum(hr_weight * x[(s, d, h, r, p, sid)])



#************* Orismos Sklirwn Periorismwn (hard constaints) ***************************

# Constraint C1): kathe mathima prepei na emfanizetai sto orologio programma toses fores oses einai i diarkeia tou
for _, subject_row in df_dict['SUBJECTS'].iterrows(): # gia kethe mathima
    s = subject_row['SUBJECT_ID']
    duration = subject_row['DURATION']
    variables_for_subject_s = [x[key] for key in x.keys() if key[0] == s] #format(s, d, h, r, p, sid) key[0] einai to subject_id
    prob += lpSum(variables_for_subject_s) == duration # prosthiki periorismou sto provlima
    # to mathima s tha topothetithei sto programma akrivos gia ton arithmo oron pou xreiazetai


# Constraint C2): kathe aithousa mporei na anatethei mono se ena mathima se mia sigkekrimeni wra mias hmeras
for _, room_row in df_dict['ROOMS'].iterrows():
    r = room_row['ROOM_ID']
    for _, day_row in df_dict['DAYS'].iterrows():
        d= day_row['DAY_ID']
        for _, hour_row in df_dict['HOURS'].iterrows():
            h = hour_row['HOUR_ID']
            variables_for_rooms = [x[key] for key in x.keys() if key[3] == r and key[1]== d and key[2]== h]
            prob += lpSum(variables_for_rooms) <= 1
            # i sigkekrimeni aithousa r de mporei na anatethei se perissotero apo ena mathima s tautoxrona


# Constraint C3): Kathe kathigitis mporei na didaksei mono ena mathima se mia sigkekrimeni wra mias hmeras
for _, day_row in df_dict['DAYS'].iterrows():
    d= day_row['DAY_ID']
    for _, hour_row in df_dict['HOURS'].iterrows():
```

```
        h = hour_row['HOUR_ID']
        for _, professor_row in df_dict['PROFESSORS'].iterrows():
            p = professor_row['PROFESSOR_ID']
            variables_for_professors = [x[key] for key in x.keys() if key[4] == p and key[1]== d and key[2]== h]
            prob += lpSum(variables_for_professors) <= 1
            # o kathigitis de tha anatethei se prissotero apo ena mathima gia sigkekrimeni xroniki stigmi


# Constraint C4): Gia sigkekrimeni omada foititwn (eksamino) mono ena upoxrewtiko mathima mporei na didaskete se sigkekrimeni wra mia hmeras
# ypoxreotiko mathima orizete ws theory, ta upoloipa einai epilogis kai ergasthria
for semester in df_dict['SEMESTERS']['SEMESTER_ID']:
    # Vriskoume oles tis theories twn eksaminwn
    semester_theory_subjects =  df_dict['SUBJECTS'][(df_dict['SUBJECTS']['SEMESTER_ID'] == str(semester)) & (df_dict['SUBJECTS']['TYPE'] ==
'Th')]

    for day in df_dict['DAYS']['DAY_ID']:
        for hour in df_dict['HOURS']['HOUR_ID']:
            # Kataskeui keys gia theoritika mathimata se auto to eksamino gia kathe mera kai wra
            theory_keys = [(s, d, h, r, p, sid) for (s, d, h, r, p, sid) in x.keys() if s in semester_theory_subjects['SUBJECT_ID'].values
and d == day and h == hour]
            # epilogi antistoixwn metavlitwn apo to x
            theory_variables = [x[key] for key in theory_keys] # = b1
            #print(f"Test Check C4 - Theory variables for semester {semester}, day {day}, hour {hour}: {theory_variables}")
            prob += lpSum(theory_variables) <= 1
            #gia kathe timeslot (d,h) enos eksaminou semester den tha anatethei perissotero apo mia theoria


# Constraint C5): Gia sigkekrimeni omada foititwn (eksamino) ta ypoxreotika mathimata de prepei na simpiptoun me ta mathimata ergastirio h
epilogis
# 'Ep'=epilogis 'Lab'= ergastirio 'Th'=ypoxreotiko theoria
for semester in df_dict['SEMESTERS']['SEMESTER_ID']:
    # Vriskoume oles tis theories, ola ta ergasthria kai ta mathimata epilogis (ta vazoyme stin idia katigoria)
```

```
    semester_lab_subjects = df_dict['SUBJECTS'][(df_dict['SUBJECTS']['SEMESTER_ID'] == str(semester)) & ((df_dict['SUBJECTS']['TYPE'] ==
'Lab') | (df_dict['SUBJECTS']['TYPE'] == 'Ep'))]
    semester_theory_subjects = df_dict['SUBJECTS'][(df_dict['SUBJECTS']['SEMESTER_ID'] == str(semester)) & (df_dict['SUBJECTS']['TYPE'] ==
'Th')]

    for day in df_dict['DAYS']['DAY_ID']:
        for hour in df_dict['HOURS']['HOUR_ID']:
            # kataskeui keys gia ta ergastiria se auto to eksamino gia kathe mera kai wra
            lab_keys = [(s, d, h, r, p, sid) for (s, d, h, r, p, sid) in x.keys() if s in semester_lab_subjects['SUBJECT_ID'].values and d ==
day and h == hour]
            # kataskeui keys gia tiw theories se auto to eksamino gia kathe mera kai wra
            theory_keys = [(s, d, h, r, p, sid) for (s, d, h, r, p, sid) in x.keys() if s in semester_theory_subjects['SUBJECT_ID'].values
and d == day and h == hour]
            # epilogi antistoixwn metavlitwn apo to x
            lab_variables = [x[key] for key in lab_keys] # = b2+b3
            theory_variables = [x[key] for key in theory_keys] # = b1
            m = 12 #megistos arithmos ergastiriwn kai epilogis pou mporoun na didaxtoun sto idio timeslot gia to idio ekamino
            # an theoria einai 1, ta ergastiria einai 0, otan i theoria einai 0 tote M ergastiria mporoun na yparxoun sto idio timeslot
            prob += lpSum(lab_variables) <= (1-lpSum(theory_variables))*m
            #ta theoritika mathimata de tha simpiptoun me ergasthria kai epilogis gia kathe eksamino gia sigkekrimeno timeslot


# Constraint C6): Kathe 4wro mathima (2 timeslots) prepei na spaei se 2 dialekseis mesa stin evdomada
for _, subject_row in df_dict['SUBJECTS'].iterrows():
    s = subject_row['SUBJECT_ID']
    duration = subject_row['DURATION']
    #an to mathima apoteleite apo 4 ores (2 duora)
    if duration == 2:
        for _, day_row in df_dict['DAYS'].iterrows():
            d = day_row['DAY_ID']
            #den prepei na emfanistei ksana tin idia mera
```

```
            variables_for_day_d = [x[key] for key in x.keys() if key[0] == s and key[1] == d]
            prob += lpSum(variables_for_day_d) <=1
            # to mathima s den tha emfanistei 2h fora tin idia mera (d)



# Prosthiki antikeimenikis sinartisis sto montelo
# To prob periexei idi tous periorismous C1-C6
prob += objective_function, "Objective Function" #anagnoristiko antikeimenikis sinartisis

#******************************** Epilisi tou provlimatos veltistopoiisis ****************************************
prob.writeLP("University_Timetabling.lp") # apothikeusi provlimatos LP se arxeio .lp to opoio mporei na epeksergastei me notepad

#prob.solve()
#CBC (Coin-or branch and cut) methodos tis pulp, timeLimit xroniko orio se seconskds gia tin evresi kaliteris lisis
#Ektelesi tis epilisis tou montelou
prob.solve(pulp.PULP_CBC_CMD(timeLimit=300))


print('\n********************** STATUS *********************************')
# Elegxos katastasis tis lisis
print(f"status: {prob.status}, {LpStatus[prob.status]}")


print('\n************************ VARIABLES ******************************\n')
# Ektiposi twn metavlitwn apofasis xwris tis imeres twn eksaminwn
print ('********************** DECISION VARIABLES CHECK WITH VALUE 1 **********************')
print("Decision Variables:")
for var in prob.variables():
    if var.varValue == 1 and not var.name.startswith('S'):
        print(f"{var.name}: {var.value()}")
```

```
print('\n************************ VARIABLES INTO variable_list *******************************')
#metafora twn parapanw metavlitwn stin variable_list
variable_list = [(var.name, var.varValue) for var in prob.variables() if var.varValue == 1 and not var.name.startswith('S')]
print (variable_list)


print('\n********************** OBJECTIVE FUNCTION ************************')
# Timi synartisis stoxou meta tin epilush to provlimatos
print("Objective Function Value:", value(prob.objective),'\n')


print('\n************************ TIMETABLE *********************************')
# Simplirosi, emfanisi kai apothikeusi tou orologiou programmatos twn mathimatwn sto TIMETABLE.xlsx/SUBJECTS
timetable, free = populate_subjects_timetable (timetable, free, df_dict['SUBJECTS'], df_dict['ROOMS'], df_dict['DAYS'], df_dict['HOURS'],
variable_list)
print('SUBJECTS TIMETABLE')
show_timetable(timetable, df_dict['DAYS'], df_dict['HOURS'], df_dict['ROOMS'])
write_timetable_subjects(timetable, df_dict['DAYS'], df_dict['HOURS'], df_dict['ROOMS'])

# Simplirosi, emfanisi kai apothikeusi tou orologiou programmatos twn kathigitwn in TIMETABLE.xlsx/PROFESSORS
timetable, free = set_up_timetable(df_dict['ROOMS'], df_dict['DAYS'], df_dict['HOURS'])
timetable, free = populate_professors_timetable (timetable, free, df_dict['SUBJECTS'], df_dict['ROOMS'], df_dict['DAYS'], df_dict['HOURS'],
variable_list)
print('PROFESSORS TIMETABLE')
show_timetable(timetable, df_dict['DAYS'], df_dict['HOURS'], df_dict['ROOMS'])
write_timetable_professors(timetable, df_dict['DAYS'], df_dict['HOURS'], df_dict['ROOMS'])
```