**ΕΛΛΗΝΙΚΟ ΑΝΟΙΚΤΟ ΠΑΝΕΠΙΣΤΗΜΙΟ**

**Πρόγραμμα Σπουδών**

**Μεταπτυχιακή Εξειδίκευση στα Πληροφοριακά Συστήματα**

**Θεματική Ενότητα**

**ΠΛΣ50 – Βασικές εξειδικεύσεις σε Θεωρία και Λογισμικό**

Ακαδημαϊκό Έτος: 2016-2017

**Πρώτη Γραπτή Εργασία**

**Γλώσσα C (Α)**

Φοιτητής: Λιάγκος Αθανάσιος

Τμήμα: ΑΘΗ-2

Αριθμός Μητρώου: 123575

Καθηγητής: Μαυρομμάτης Γεώργιος

**Αθήνα Νοέμβριος 2016**

# ΕΙΣΑΓΩΓΗ

Οι ασκήσεις έχουν λυθεί και μεταγλωττιστεί με το CodeBlocks 16.01 σε περιβάλλον Windows 10 x64 με χρήση του gcc 5.4.0 και GNU make 4.2.1 του πακέτου Cygwin64[[1]](#footnote-1) (έκδοση 2.6.0), σε συνδυασμό με το CLion[[2]](#footnote-2) της JetBrains με τη χρήση του ίδιου compiler και make. Το CLion προτιμήθηκε και για τη δημιουργία των screenshots.

# ΑΣΚΗΣΗ 1

Στην άσκηση αυτή το ζητείται να φτιάξουμε μια μικρή αριθμομηχανή. Η αριθμομηχανή θα κάνει τις βασικές πράξει μεταξύ δύο δεκαδικών αριθμών, και στη συνέχεια το πρόγραμμα θα σταματάει, ενώ υπάρχει πρόβλεψη για διαίρεση με το 0 και εισαγωγή λάθος συντελεστή πράξης.

Ορίζουμε αρχικά τις απαιτούμενες μεταβλητές ως δεκαδικούς **float** οι οποίες θα περιέχουν τα νούμερα που έδωσε ο χρήστης καθώς και το αποτέλεσμα. Μετά ως **char** ορίζουμε το σύμβολο της πράξης.



Στη συνέχεια εμφανίζουμε ένα μήνυμα στο χρήστη με σύντομες οδηγίες για τη λειτουργία του προγράμματος.



Διαβάζουμε τα νούμερα από το πληκτρολόγιο στις μεταβλητές **num1** και **num2**, καθώς και το σύμβολο της πράξης στην μεταβλητή **praxi.**

****

Με ένα **if** ελέγχουμε αν ο δεύτερος αριθμός **num2** είναι μηδέν **και** η πράξη είναι διαίρεση, ώστε να εμφανίσουμε το κατάλληλο μήνυμα λάθους και το πρόγραμμα να σταματήσει με **return 0**.



Χρησιμοποιούμε τη **switch** για να ελέγξουμε αρχικά το σύμβολο της πράξης, και ανάλογα το σύμβολο, δίνουμε στη μεταβλητή **result** το αποτέλεσμα της αντίστοιχης πράξης.



Για να εμφανίσουμε ένα κατάλληλο μήνυμα σε περίπτωση που η πράξη δεν είναι κάποια από τις προβλεπόμενες, χρησιμοποιούμε το **default** της **switch**, η οποία εμφανίζει το κατάλληλο μήνυμα λάθους και σταματάει την εκτέλεση του προγράμματος με **return 0**. Το **break;** μετά το **return 0;** είναι περιττό αφού το πρόγραμμα θα έχει ήδη σταματήσει να εκτελείται.

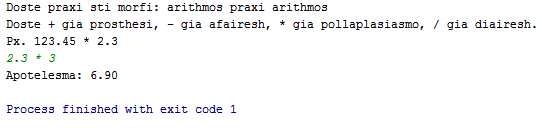


Σε αυτό το σημείο, έξω από τη **switch**, ξέρουμε πως το σύμβολο της πράξης είναι σωστό, δεν γίνεται διαίρεση με το μηδέν και η πράξη έχει εκτελεστεί. Άρα εμφανίζουμε το αποτέλεσμα με δύο δεκαδικά ψηφία από τη μεταβλητή **result** και σταματάμε την εκτέλεση του προγράμματος με **return 1**.

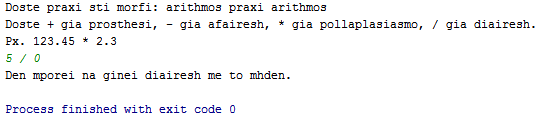


Ακολουθούνε **screenshots** από την εκτέλεση του προγράμματος:

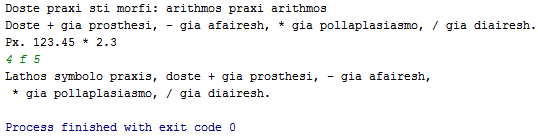
**Screenshot 1.1 – Κανονική εκτέλεση του προγράμματος**



**Screenshot 1.2 – Διαίρεση με το μηδέν**



**Screenshot 1.3 – Λάθος σύμβολο πράξης**



# ΑΣΚΗΣΗ 2

Σε αυτή την άσκηση θα υπολογίσουμε την περίμετρο τετραγώνου, παραλληλογράμμου, ορθογωνίου τριγώνου και κύκλου. Ο χρήστης θα επιλέγει από ένα μενού με επιλογές 1 έως 4 για τα αντίστοιχα σχήματα, και 0 για έξοδο. Το πρόγραμμα θα εκτελείται συνεχώς μέχρι ο χρήστης να επιλέξει 0. Οι υπολογισμοί των αντίστοιχων περιμέτρων θα γίνονται μέσα σε κατάλληλες **functions** οι οποίες θα επιστρέφουν το αποτέλεσμα.

Θα πρέπει να συμπεριλάβουμε τη βιβλιοθήκη **math.h** ώστε να χρησιμοποιήσουμε τη συνάρτηση **sqrt** (τετραγωνική ρίζα, απαραίτητη για το ορθογώνιο τρίγωνο) και τη σταθερά **M\_PI** (ο αριθμός **π**, απαραίτητος για τον κύκλο).



Βάζουμε την εμφάνιση του μενού σε μία **function** η οποία καλείται από το κυρίως πρόγραμμα μέχρι ο χρήστης να επιλέξει μηδέν.



Στη συνέχεια φτιάχνουμε τις συναρτήσεις που υπολογίζουν τις αντίστοιχες περιμέτρους. Όλες οι συναρτήσεις επιστρέφουν τιμές τύπου **float** και παίρνουν ως παράμετρο μία **float** μεταβλητή για τετράγωνο και κύκλο, και δύο **float** μεταβλητές για παραλληλόγραμμο και ορθογώνιο τρίγωνο.

Στο τετράγωνο επιστρέφουμε την παράμετρο επί 4 φορές.



Στο παραλληλόγραμμο επιστρέφουμε το διπλάσιο του αθροίσματος των παραμέτρων (μήκος και πλάτος).



Στο ορθογώνιο τρίγωνο βρίσκουμε την υποτείνουσα από τις δύο κάθετες πλευρές σύμφωνα με το Πυθαγόρειο Θεώρημα, και την αθροίζουμε μαζί με τις κάθετες πλευρές που γνωρίζουμε. Για την τετραγωνική ρίζα χρησιμοποιούμε τη συνάρτηση **sqrt** από τη βιβλιοθήκη **math.h**.



Στον κύκλο βρίσκουμε την περίμετρο πολλαπλασιάζοντας την ακτίνα με το 2 και τη σταθερά **π**. Την τιμή της σταθεράς **π** την παίρνουμε από την **M\_PI** που είναι ορισμένη στη βιβλιοθήκη **math.h**.



Στη συνέχεια ξεκινάει η **main()** με το κυρίως πρόγραμμα. Ορίζουμε τις μεταβλητές **factor1** και **factor2** ως τύπου **float** για τους αριθμούς που θα δίνει ο χρήστης και τη μεταβλητή **choice** τύπου **char** για την επιλογή από το μενού.



Χρησιμοποιούμε μετά έναν βρόγχο **do..while** για να τρέχει συνεχώς το πρόγραμμα όσο ο χρήστης δεν επιλέγει μηδέν για έξοδο, ακόμα και αν πατήσει λάθος επιλογή.



Μέσα στον βρόγχο εμφανίζεται το μενού με κλήση της αντίστοιχης συνάρτησης και διαβάζουμε από το πληκτρολόγιο στην **choice** την επιλογή του χρήστη.



Μετά με διαδοχικά **if…else if…else if…** ελέγχουμε την επιλογή του χρήστη και του λέμε να δώσει τα απαραίτητα νούμερα, μήκος πλευράς και ακτίνας για τετράγωνο και κύκλο αντίστοιχα, μήκος και πλάτος πλευρών για παραλληλόγραμμο και μήκος καθέτων πλευρών για ορθογώνιο τρίγωνο. Αφού διαβάσουμε τα δεδομένα από τον χρήστη, εμφανίζουμε το αποτέλεσμα χρησιμοποιώντας τις αντίστοιχες συναρτήσεις που έχουμε ορίσει παραπάνω.



Το τελευταίο **else if** το χρησιμοποιούμε για την περίπτωση που χρήστης έδωσε μια λανθασμένη επιλογή ώστε να εμφανίσουμε ένα μήνυμα. Εδώ έχουμε αποκλείσει την πιθανότητα να είναι η επιλογή 1 έως 4, ενώ αν είναι 0 το πρόγραμμα θα τερματίσει στο τέλος του βρόγχου. Άρα το μήνυμα εμφανίζεται μόνο αν η επιλογή είναι διαφορετική από 0, 1, 2, 3, 4.

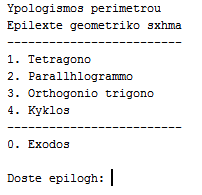


Μετά τελειώνει ο βρόγχος, ελέγχουμε αν η επιλογή δεν είναι μηδέν και το πρόγραμμα ξεκινάει από την αρχή. Διαφορετικά τερματίζει με **return 1**.

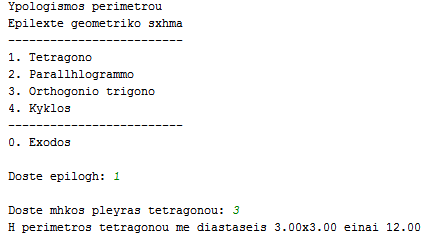


Ακολουθούνε **screenshots** από την εκτέλεση του προγράμματος:

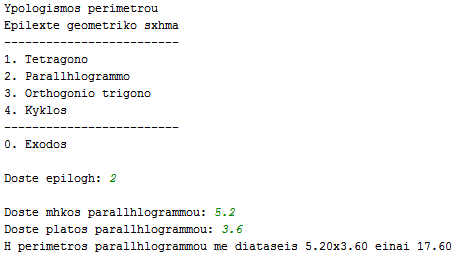
**Screenshot 2.1 – Εμφάνιση μενού**

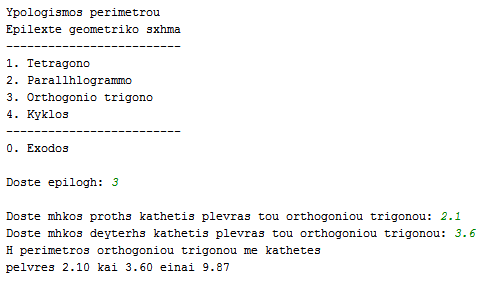


**Screenshot 2.2 – Περίμετρος τετραγώνου**

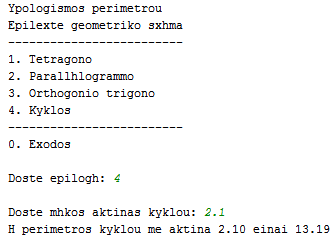


**Screenshot 2.3 – Περίμετρος παραλληλογράμμου**

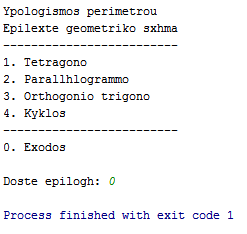


**Screenshot 2.4 – Περίμετρος ορθογωνίου τριγώνου**

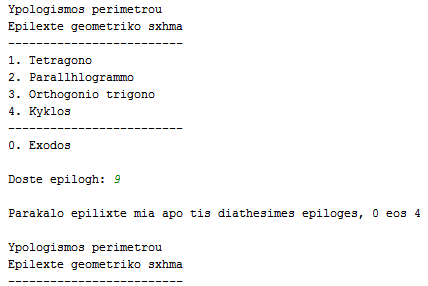
**Screenshot 2.5 – Περίμετρος κύκλου**



**Screenshot 2.6 – Έξοδος από το πρόγραμμα**



**Screenshot 2.7 – Λανθασμένη επιλογή**



# ΑΣΚΗΣΗ 3

Στην άσκηση αυτή θα διαβάζουμε έναν θετικό ακέραιο από το πληκτρολόγιο, και θα τον μετατρέπουμε σε αντίστροφη δυαδική αναπαράσταση (το λιγότερο σημαντικό ψηφίο αριστερά). Το πρόγραμμα θα υπολογίζει το υπόλοιπο της διαίρεσης του τρέχοντος αριθμού με το 2, και θα γράφει 1 αν το αποτέλεσμα είναι μηδέν ή 0 αν το αποτέλεσμα είναι 1. Στη συνέχεια θα θέτει ως τρέχον αριθμό το ακέραιο πηλίκο της διαίρεσης του τρέχοντος αριθμού με τη βάση (2). Όταν ο τρέχον αριθμός γίνει 0, το πρόγραμμα θα τερματίζεται.

Ορίζουμε ως σταθερά την **base** ως ακέραιο με την τιμή 2.



Ορισμός τύπου μεταβλητών **dec** (τρέχον αριθμός) και **bin** (τρέχον υπόλοιπο διαίρεσης) ως **int**.



Εμφάνιση τίτλου προγράμματος.



Βρόγχος που εκτελείται όσο ο χρήστης δεν δίνει θετικό αριθμό. Αφού εισαχθεί ο αριθμός, γίνεται έλεγχος αν είναι θετικός, διαφορετικά εμφανίζει το κατάλληλο μήνυμα. Ο αριθμός που δίνει ο χρήστης αποθηκεύεται στη μεταβλητή **dec**.



Εμφάνιση τίτλου αποτελέσματος.



Βρόγχος που εκτελεί τους ουσιαστικούς υπολογισμούς για να εμφανίσει τον κατάλληλο χαρακτήρα, 0 ή 1. Εκτελείται όσο ο τρέχον αριθμός **dec** είναι μεγαλύτερος του μηδέν. Υπολογίζεται το υπόλοιπο του τρέχοντος αριθμού με τη βάση **base** και αν αυτό είναι 0 γράφει τον χαρακτήρα 1 αλλιώς γράφει το 0, με τη βοήθεια του τριαδικού τελεστή ?:. Στη συνέχεια θέτει ως νέο τρέχον αριθμό το ακέραιο πηλίκο της διαίρεσης του τρέχοντος αριθμού με τη βάση. Χρησιμοποιείται ο τελεστής /= στην έκφραση **dev /= base** ο οποίος είναι ο ίδιος με την έκφραση **dec = dec / base**.

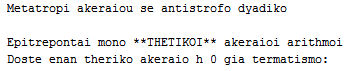


Στη συνέχεια το πρόγραμμα τερματίζεται.

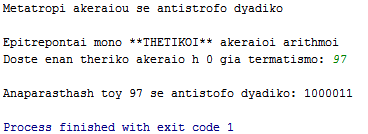


Ακολουθούνε **screenshots** από την εκτέλεση του προγράμματος:

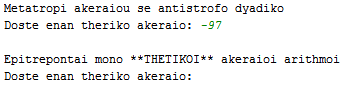
**Screenshot 3.1 – Εμφάνιση αρχικής οθόνης**



**Screenshot 3.2 – Υπολογισμός δυαδικού**



**Screenshot 3.3 – Είσοδος αρνητικού αριθμού**



# ΑΣΚΗΣΗ 4

Στην άσκηση αυτή θα φτιάξουμε ένα απλό σύστημα κρατήσεως θέσεων λεωφορείου. Το πρόγραμμα θα διαβάζει από το αρχείο **bus.txt** την πινακίδα του λεωφορείου η οποία θα αποτελείται από 7 χαρακτήρες και τον αριθμό Ν+5 των θέσεων του λεωφορείου. Το Ν πρέπει να ικανοποιεί τη συνθήκη 4xN+5 <= 53 για N>=0.

Το πρόγραμμα αποτελείται από επιμέρους συναρτήσεις οι οποίες ικανοποιούν τις 8 διαφορετικές διαθέσιμες επιλογές του προγράμματος. Επιπλέον υπάρχει μία η οποία θα δείχνει τις διαθέσιμες επιλογές, και 3 βοηθητικές οι οποίες ελέγχουν αν η τρέχουσα θέση είναι παράθυρο. Για λόγους αισθητικής στο στην επιλογή 1 (εμφάνιση κενών θέσεων), κρίθηκε σκόπιμο η αρχική συνάρτηση isWindow (η οποία επιστρέφει αν η τρέχουσα θέση είναι στο παράθυρο ή όχι) να διαχωριστεί σε 2 διαφορετικές, οι οποίες ελέγχουν αν είναι συγκεκριμένα αριστερό ή δεξιό παράθυρο. Έτσι η isWindow επιστρέφει με μία συνθήκη **OR** το αποτέλεσμα των επιμέρους συναρτήσεων.

Όλες οι συναρτήσεις του προγράμματος, εκτός του μενού και των συναρτήσεων που ελέγχουν τη θέση αν είναι σε παράθυρο, παίρνουν ως παραμέτρους τον πίνακα των θέσεων και τον μέγιστο αριθμό των θέσεων, των οποίων οι τιμές αρχικοποιούνται στη **main()** και σε όσες χρειάζεται επιπλέον περνάμε και τη συγκεκριμένη θέση του λεωφορείου για την οποία ενδιαφερόμαστε. Οι συναρτήσεις που ελέγχουν αν η θέση είναι σε παράθυρο, παίρνουν ως παραμέτρους τις συνολικές θέσεις και την τρέχουσα θέση.



Η συνάρτηση **isLeftWindow** επιστρέφει 1 αν η τρέχουσα θέση **seat** ανήκει σε τετράδα και το υπόλοιπο της διαίρεσης με το 4 είναι 1, **Ή** αν είναι σε πεντάδα και είναι η πρώτη θέση.



Η συνάρτηση **isRightWindow** επιστρέφει 1 αν η τρέχουσα θέση **seat** ανήκει σε τετράδα και το υπόλοιπο της διαίρεσης με το 4 είναι 0, **Ή** αν είναι σε πεντάδα και είναι η πέμπτη θέση.



Η συνάρτηση **isWindow** αποτελεί τον συνδυασμό των παραπάνω συναρτήσεων, και επιστρέφει 1 αν η θέση είναι είτε σε αριστερό είτε σε δεξιό παράθυρο.



Η συνάρτηση **showMenu** εμφανίζει το μενού με τις επιλογές στον χρήστη.



Η συνάρτηση **showBlanks** εμφανίζει συνεχόμενες τις κενές θέσεις του λεωφορείου καθώς και το σύνολο αυτών. Η μεταβλητή **i** χρησιμοποιείται για τον βρόγχο που διαβάζει όλες τις θέσεις του λεωφορείου και η μεταβλητή **blanks**,την οποία αρχικοποιούμε με 0, για να ανεβαίνει κατά 1 κάθε φορά που η θέση είναι κενή. Ο βρόγχος εκτελείται για όλες τις θέσεις **maxSeats** μέχρι το **i** να πάρει την τιμή **maxSeats – 1** επειδή οι το πρώτο στοιχείο του πίνακα είναι στη θέση 0. Στη συνέχεια ελέγχουμε αν στη θέση **i** του πίνακα έχουμε 0 (=κενή θέση), και αν ναι αυξάνουμε την **blanks** κατά 1, ενώ έχουμε ήδη τυπώσει ένα κόμμα για να ξεχωρίζουν οι θέσεις μεταξύ τους. Ενώ είμαστε μέσα στο **if** που ελέγχουμε αν η θέση είναι σε παράθυρο τυπώνουμε τη θέση μέσα σε αγκύλες, αλλιώς χωρίς αγκύλες. Αφού έχουμε βγει από τον έλεγχο για το παράθυρο, αν έχει συμπληρωθεί δεκάδα θέσεων (διαίρεση **blanks** με το 10 δίνει υπόλοιπο 0), τυπώνει το χαρακτήρας νέας γραμμής. Τέλος, αφού έχει τελειώσει ο βρόγχος, τυπώνουμε τη μεταβλητή **blanks** η οποία περιέχει το πλήθος των κενών θέσεων.



Η συνάρτηση **reserveSpecific** κρατάει μια συγκεκριμένη θέση. Εδώ χρησιμοποιούμε τη μεταβλητή **seat** επικουρικά, αν δηλαδή έχει την τιμή 0 το πρόγραμμα μας ζητάει να δώσουμε συγκεκριμένη θέση, διαφορετικά σημαίνει πως η θέση έχει ήδη υπολογισθεί και προχωράει χωρίς την ερώτηση. Με αυτόν τον τρόπο χρησιμοποιούμε την ίδια συνάρτηση και στην περίπτωση που θέλουμε να κλείσουμε την πρώτη θέση σε παράθυρο με την **reserveFreeWindow**, όπου όταν την καλέσουμε η **seat** θα έχει τιμή διαφορετική του 0. Η συνάρτηση ελέγχει αν η θέση που ζητήθηκε είναι μέσα στο αποδεκτό εύρος (<=maxSeats), διαφορετικά εμφανίζει μήνυμα. Αν η θέση είναι μέσα στο αποδεκτό εύρος, ελέγχει αν η θέση είναι διαθέσιμη, και αν είναι τη δεσμεύει θέτοντας στην αντίστοιχη θέση του πίνακα την τιμή 1. Σε κάθε περίπτωση, εμφανίζεται μήνυμα για το αν η θέση κρατήθηκε ή είναι ήδη κρατημένη.



Η συνάρτηση **reserveFreeWindow** κρατάει την πρώτη διαθέσιμη θέση σε παράθυρο. Χρησιμοποιεί τη μεταβλητή **i** για τον βρόγχο που ελέγχει όλες τις θέσεις. Μέσα στον βρόγχο, ελέγχουμε αν η θέση είναι ελεύθερη (=τιμή 0) **και** αν είναι σε παράθυρο χρησιμοποιώντας την **isWindow**. Αν ικανοποιείται αυτή η συνθήκη, καλείται η **reserveSpecific** με παράμετρο τη συγκεκριμένη θέση και ο βρόγχος τερματίζει με την εντολή **break**. Ελέγχοντας τέλος την τιμή της μεταβλητής **i**, αν αυτή έχει τιμή μεγαλύτερη ή ίση με τον αριθμό των θέσεων, σημαίνει πως ο βρόγχος έχει ολοκληρωθεί χωρίς να βρεθεί διαθέσιμη θέση και εμφανίζεται το αντίστοιχο μήνυμα.



Η συνάρτηση **cancelSpecific** ακυρώνειμια συγκεκριμένη θέση και λειτουργεί με αντίστοιχο τρόπο με τη **reserveSpecific**. Το πρόγραμμα αρχικά ζητάει να δώσουμε τη θέση, η οποία αποθηκεύεται στη μεταβλητή **seat**. Η συνάρτηση ελέγχει αν η θέση που ζητήθηκε είναι μέσα στο αποδεκτό εύρος (<=maxSeats), διαφορετικά εμφανίζει μήνυμα. Αν η θέση είναι μέσα στο αποδεκτό εύρος, ελέγχει αν η θέση είναι δεσμευμένη, και αν είναι την αποδεσμεύει θέτοντας στην αντίστοιχη θέση του πίνακα την τιμή 0. Σε κάθε περίπτωση, εμφανίζεται μήνυμα για το αν η θέση αποδεσμεύθηκε ή είναι ήδη ελεύθερη.



Η συνάρτηση **showReserved** μας εμφανίζει μια λίστα με τις θέσεις που είναι ήδη κρατημένες. Χρησιμοποιούμε τη μεταβλητή **i** για τον βρόγχο που ελέγχει όλες τις θέσεις, και τη μεταβλητή **c**, την οποία αρχικοποιούμε με 0, για να μετρήσουμε τον αύξον αριθμό της τρέχουσας ελεύθερης θέση. Μέσα στον βρόγχο ελέγχουμε αν η τρέχουσα θέση **i** είναι δεσμευμένη, αν είναι αυξάνουμε τη **c** κατά 1 και εμφανίζουμε τον αύξον αριθμό και τον αριθμό της, ο οποίος αριθμός θέσης εμφανίζεται μέσα σε αγκύλες αν είναι σε παράθυρο με τη χρήση της **isWindow**. Έξω από τον βρόγχο, ελέγχουμε την τιμή του **c** και αν είναι 0 σημαίνει πως δεν βρέθηκε καμία δεσμευμένη θέση, οπότε εμφανίζεται το αντίστοιχο μήνυμα.



Η συνάρτηση **checkReserverdSpecific** ελέγχει αν μια συγκεκριμένη θέση είναι κρατημένη. Λειτουργεί με παρόμοιο τρόπο με την **reserveSpecific** με τη διαφορά πως δεν γίνεται δέσμευση θέσης. Το πρόγραμμα αρχικά ζητάει να δώσουμε τη θέση, η οποία αποθηκεύεται στη μεταβλητή **seat**. Η συνάρτηση ελέγχει αν η θέση που ζητήθηκε είναι μέσα στο αποδεκτό εύρος (<=maxSeats), διαφορετικά εμφανίζει μήνυμα. Αν η θέση είναι μέσα στο αποδεκτό εύρος και ελέγχει αν η θέση είναι δεσμευμένη (η τιμή στη θέση **seat-1** στον πίνακα θέσεων έχει την τιμή 1) ή όχι, και τυπώνει το αντίστοιχο μήνυμα.



Η συνάρτηση **showPlan** δείχνει το διάγραμμα των θέσεων του λεωφορείου και την πινακίδα (μεταβλητή **plate**), εμφανίζοντας τον χαρακτήρα **\_** για τις κενές θέσεις και τον χαρακτήρα **\*** για τις κρατημένες. Η συνάρτηση έχει τις μεταβλητές **i** και **c** τιςοποίες χρησιμοποιούμε για τους τρεις βρόγχους τις συνάρτησης. Οι πρώτοι δύο βρόγχοι είναι ο ένας μέσα στον άλλον, και τους χρησιμοποιούμε για τις θέσεις **maxSeats – 4**, δηλαδή για όλες τις θέσεις εκτός από την τελευταία σειρά. Ο βρόγχος ξεκινάει από τη θέτοντας την τιμή 0 στη μεταβλητή **i** και αυξάνει την **i** κατά 4 σε κάθε πέρασμα, δηλαδή τρέχει τόσες φορές όσες και οι διαθέσιμε σειρές στο λεωφορείο πλην μίας. Μέσα σε αυτόν το βρόγχο εκτελούμε έναν δεύτερο θέτοντας στη μεταβλητή **c** την τιμή 0 και την αυξάνουμε κατά 1 μέχρι να εκτελεστεί 4 φορές, δηλαδή για όλες τις θέσεις της σειράς. Μέσα στον βρόγχο αυτό, η μεταβλητή **i** έχει τον αριθμό της σειράς και η **c** τον αριθμό της θέσης, οπότε η **i+c** είναι η θέση στον πίνακα **layout** της συγκεκριμένης θέσης. Ελέγχουμε λοιπόν την τιμή αυτή της θέσης με τον τριαδικό συντελεστή και αν είναι κρατημένη εμφανίζουμε τον χαρακτήρα **\*** ήτον χαρακτήρα **\_** για τις ελεύθερες, με τη χρήση της **putchar**. Αν η **c** έχει την τιμή 1 (=έχουμε ήδη τυπώσει τη 2ηθέση της σειράς), τότε τυπώνει ένα κενό. Στο τέλος κάθε γραμμής τυπώνουμε τον χαρακτήρα αλλαγής γραμμής. Αφού έχει τελειώσει η εκτέλεση των δύο βρόγχων, κάνουμε έναν επιπλέον για τις τελευταίες 5 θέσεις, θέτοντας στην **i** την τιμή εκκίνησης **maxSeats-5** και την εκτελούμε 5 φορές. Σε κάθε πέρασμα η **i** θα έχει τη θέση στον πίνακα **layout** της τρέχουσας θέσης, και με αντίστοιχο τρόπο όπως και με τις άλλες θέσεις τυπώνουμε με την **putchar** τον αντίστοιχο χαρακτήρα, και στο τέλος του βρόγχου τυπώνουμε και τον χαρακτήρα αλλαγής γραμμής.



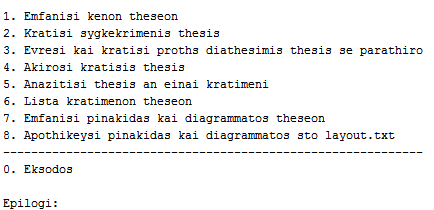
Η συνάρτηση **writePlanFile** είναι όμοια με την **showPlan**, με τη διαφορά πως η έξοδος είναι πλέον στο αρχείο **layout.txt** και όχι στην οθόνη. Για να δημιουργήσουμε το αρχείο ορίζουμε τη μεταβλητή **\*plan** ως τύπου **FILE** και ανοίγουμε το αρχείο με τη **fopen** για γράψιμο (με τον συντελεστή **w**). Στη συνέχεια ακολουθούμε τα ίδια ακριβώς βήματα όπως και στη συνάρτηση **showPlan**, με τη διαφορά πως όπου υπάρχει η **printf** την αντικαθιστούμε με την **fprintf** και βάζοντας ως **πρώτη** παράμετρο τη μεταβλητή **plan**, και την **putchar** με την **putc** και βάζοντας ως **τελευταία** παράμετροτη μεταβλητή **plan**, όπου η **plan** υποδηλώνει το αρχείο που έχουμε ανοίξει με την **fopen**. Επιπλέον εκτελούμε την **fclose** για να κλείσουμε το αρχείο που έχουμε δημιουργήσει, και εμφανίζουμε το αντίστοιχο μήνυμα.



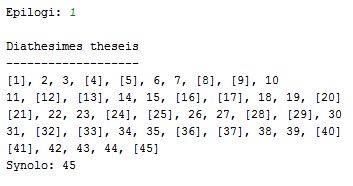
Η **main** τέλος, περιέχει τους βασικούς ορισμούς των μεταβλητών του προγράμματος, κάνει τους βασικούς ελέγχους ορθότητας των παραμέτρων και διαχειρίζεται το μενού οδηγώντας τη ροή του προγράμματος στις κατάλληλες συναρτήσεις. Ορίζουμε 2 μεταβλητές τύπου **char**, μία με μέγεθος 7 για την πινακίδα του λεωφορείου και μία με μέγεθος 1 για την επιλογή του μενού. Ορίζουμε 3 μεταβλητές τύπου **int**, μία για την αποθήκευση του επιθυμητού αριθμού θέσεων που θα έχει το λεωφορείο, μία ως πίνακα 53 θέσεων που θα καταχωρούνται όλες οι κινήσεις κράτησης/ακύρωσης, και μία για τον απαραίτητο βρόγχο αρχικοποίησης του πίνακα αυτού. Τέλος, ορίζουμε μια μεταβλητή τύπου **FILE** για να διαβάσουμε το αρχείο που περιέχει ως δεδομένα την πινακίδα και τον αριθμό των θέσεων, χωρισμένα με κενό. Ανοίγουμε λοιπόν με τη **fopen** το αρχείο **bus.txt** για διάβασμα (συντελεστής **r**), και με την **fscanf** διαβάζουμε ένα string, ένα κενό και έναν ακέραιο από το αρχείο **bus**, τα οποία περιέχουν την πινακίδα και τον αριθμό θέσεων τα οποία αποθηκεύουμε στις μεταβλητές **plate** και **maxSeats** αντίστοιχα, και κλείνουμε το αρχείο με την **fclose**. Στη συνέχεια ελέγχουμε αν αριθμός θέσεων δόθηκε είναι έγκυρος, ελέγχοντας αν η **maxSeats-5** διαιρούμενες με το 4 **δεν** βγάζουν υπόλοιπο 0. Αν δεν είναι έγκυρος εμφανίζεται αντίστοιχο μήνυμα, ενώ αν είναι ελέγχουμε αν επίσης ο αριθμός θέσεων είναι μεγαλύτερος από 53, οπότε εμφανίζουμε αντίστοιχο μήνυμα. Στο επόμενο **else** το πρόγραμμα θα μπει **μόνο** όταν ο αριθμός των θέσεων είναι έγκυρος. Εκεί το πρόγραμμα με τη χρήση της **for** θέτει σε όλες τις θέσεις του πίνακα **layout** την τιμή 0 (=ελεύθερη θέση) και για όσες θέσεις έχουν ορισθεί στην **maxSeats**. Μετά, σε έναν βρόγχο **do..while** εμφανίζουμε το μενού και διαβάζουμε την επιλογή του χρήστη στη μεταβλητή **option**, όσο (while) η επιλογή του χρήστη δεν είναι 0 (=έξοδος). Με τη χρήση της **switch** ελέγχουμε τις επιλογές του χρήστη, και καλούμε κάθε φορά την αντίστοιχη συνάρτηση με τις αντίστοιχες παραμέτρους που έχουμε αναφέρει στην αρχή αυτής της άσκησης. Σε περίπτωση που ο χρήστης επιλέξει 0, παρατηρούμε πως έχουμε μόνο ένα **break** το οποίο μας εξυπηρετεί διότι έτσι το πρόγραμμα θα βγει από την **switch** και θα ικανοποιηθεί η συνθήκη του βρόγχου **do..while** και το πρόγραμμα θα σταματήσει με **return 1**. Με τη χρήση της **default** εμφανίζουμε ένα μήνυμα σε περίπτωση που ο χρήστης πάτησε κάποια μη διαθέσιμη επιλογή, και το μενού εμφανίζεται από την αρχή. Έξω από όλα τα **if** υπάρχει η **return 0**, η οποία θα εκτελεστεί μόνο αν κάποιο από τα δύο αρχικά **if** επαληθευθεί (θέσεις που δεν ικανοποιούν τον αριθμό (4xN + 5) <= 53) και το πρόγραμμα τερματίζει.

Ακολουθούν screenshots από την εκτέλεση του προγράμματος. Το αρχείο **bus.txt** έχει τη γραμμή **ΝΒΑ6677 45**, άρα οι θέσεις είναι 45.

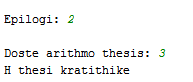
**Screenshot 4.1.1 – Μενού επιλογών**



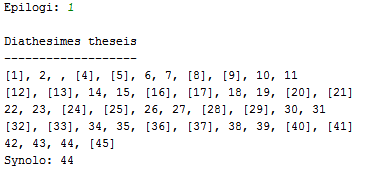
**Screenshot 4.1.2 – Κενές θέσεις (όλες κενές)**



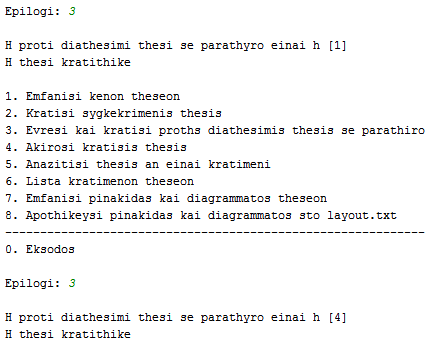
**Screenshot 4.1.3 – Κράτηση θέσης 3**



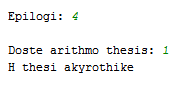
**Screenshot 4.1.4 – Κενές θέσεις (αφού έχει κρατηθεί η 3)**



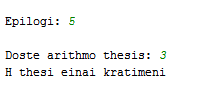
**Screenshot 4.1.5 – Κράτηση 2 συνεχόμενων θέσεων σε παράθυρο**



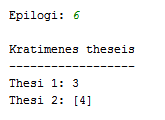
**Screenshot 4.1.6 – Ακύρωση θέσης 1**



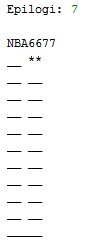
**Screenshot 4.1.7 – Έλεγχος αν η θέση 3 είναι κρατημένη**



**Screenshot 4.1.8 – Εμφάνιση κρατημένων θέσεων (3 & 4)**



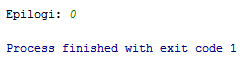
**Screenshot 4.1.9 – Εμφάνιση διαγράμματος**



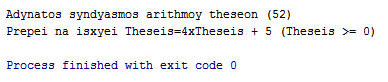
**Screenshot 4.1.10 – Αποθήκευση διαγράμματος σε αρχείο**



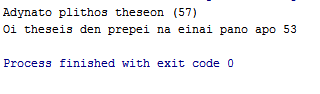
**Screenshot 4.1.11 – Έξοδος**



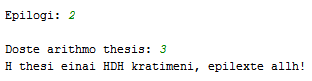
**Screenshot 4.2.1 – Μη έγκυρος αριθμός θέσεων στο αρχείο bus.txt (52)**



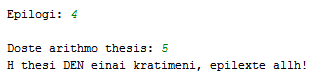
**Screenshot 4.2.2 – Έγκυρος αριθμός θέσεων αλλά >53 στο αρχείο bus.txt (57)**



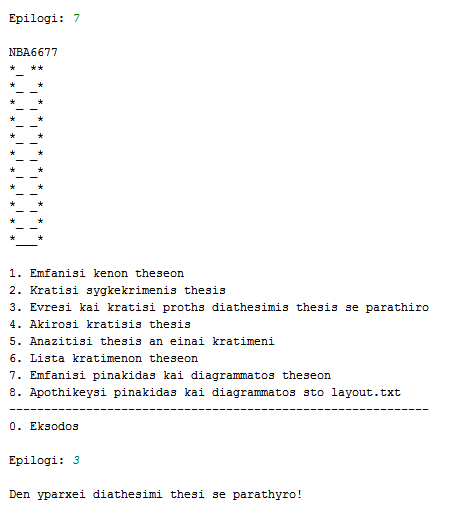
**Screenshot 4.2.3 – Απόπειρα κράτησης μη ελεύθερης θέσης**



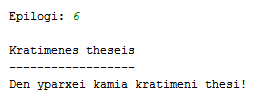
**Screenshot 4.2.4 – Απόπειρα ακύρωσης μη δεσμευμένης θέσης**



**Screenshot 4.2.5 – Κράτηση θέσης σε παράθυρο ενώ δεν υπάρχει διαθέσιμο**



**Screenshot 4.2.5 – Λίστα κρατημένων θέσεων ενώ είναι όλες ελεύθερες**



1. https://www.cygwin.com [↑](#footnote-ref-1)
2. https://www.jetbrains.com/clion [↑](#footnote-ref-2)