Στέλιος Πολυχρόνης

ΑΜ 123603, Τμήμα ΘΕΣ-1

2η Γραπτή Εργασία

**Θέμα 1ο**

Στο θέμα αυτό επέλεξα όταν εισάγονται από τον χρήστη τα στοιχεία του επιβάτη αυτό να γίνεται σταδιακά και όχι όλα μαζί, δηλαδή το πρόγραμμα να ζητάει πρώτα το «επίθετο», μετά το «όνομα» και τέλος το «τηλέφωνο». Στη συνέχεια μόνο του το πρόγραμμα τα προσθέτει με τον σωστό τρόπο στα αντίστοιχα πεδία του πίνακα θέσεων. Έτσι αποφεύγουμε το πρόβλημα, να τα δώσει ο χρήστης με λάθος σειρά (πρώτα όνομα και μετά επίθετο) ή να μην αφήσει ανάμεσά τους «κενό».

Στις γραμμές 193-203 και 207-222 με την βοήθεια της μεταβλητής z η υλοποίηση που επέλεξα ξεπερνά το πρόβλημα να έχουν κρατηθεί δύο ή περισσότερες θέσεις στο ίδιο ονοματεπώνυμο ή στο ίδιο τηλέφωνο. Έτσι τις εμφανίζει όλες και δεν σταματά στην πρώτη καταχώρηση που ταιριάζει με την αναζήτηση του χρήστη.

**Θέμα 2ο**

Στις γραμμές 219-226 και 233-248 με την βοήθεια της μεταβλητής z μετράμε πόσες θέσεις είναι κρατημένες στο ίδιο ονοματεπώνυμο ή στο ίδιο τηλέφωνο, αντίστοιχα. Όποτε μπορούμε να εμφανίσουμε όλες αυτές τις θέσεις. Αν δεν χρησιμοποιούσαμε αυτή την υλοποίηση, τότε το πρόγραμμα θα εμφάνιζε μόνο την πρώτη θέση που θα εντόπιζε με το δοθέν ονοματεπώνυμο ή τηλέφωνο, και θα αγνοούσε τις υπόλοιπες.

Στη γραμμή 374 καλείται η συνάρτηση emfanisikomvwn που ορίζεται στις γραμμές 401-416 και εμφανίζει στην οθόνη όλους τους κόμβους της απλά συνδεδεμένης λίστας. Κάτι τέτοιο δεν ζητείται από την άσκηση. Απλά το πρόσθεσα ως επαλήθευση ότι η λίστα έχει δημιουργηθεί σωστά πριν αποθηκευτεί στο αρχείο εξόδου.

Στις γραμμές 383-398 ορίζεται η συνάρτηση prosthikikomvou, που είναι τύπου struct komvos και προσθέτει με αναδρομικό αλγόριθμο νέους κόμβους στο τέλος της απλά συνδεδεμένης λίστας. Στην αρχή προσπάθησα να γράψω την ίδια συνάρτηση ως τύπου void, αλλά αντιμετώπισα το εξής πρόβλημα: επειδή η συνάρτηση καλείται μέσα από μια for που σαρώνει τον πίνακα θέσεων και εντοπίζει τις κρατημένες θέσεις, ουσιαστικά ο κάθε καινούριος κόμβος «κάλυπτε» τον προηγούμενο που είχε εγγραφεί. Οπότε στο τέλος η λίστα είχε μόνο την τελευταία εγγραφή. Γι’ αυτό το λόγο άλλαξα τον τύπο της συνάρτησης σε struct komvos, ώστε να επιστρέφει στον βρόγχο που την καλεί τον τελευταίο «νέο κόμβο» και έτσι η επόμενη εγγραφή να γίνεται μετά από αυτόν.

**Θέμα 3ο**

Στη γραμμή 17 ορίζουμε έναν μονοδιάστατο πίνακα kartes 81 θέσεων που θα περιέχει και τις 81 κάρτες όλων των πιθανών συνδυασμών, και έναν διδιάστατο πίνακα tablo (3 γραμμών και 4 στηλών) που θα περιέχει μόνο τις κάρτες που εμφανίζονται στην οθόνη. Το βασικό σκεπτικό μου για την υλοποίηση της άσκησης είναι να υπάρχουν αποθηκευμένες οι 81 πιθανές κάρτες στον πίνακα kartes (διαδικασία που εφαρμόζεται στις γραμμές 43-64) και κάθε φορά που χρειαζόμαστε νέα 12άδα καρτών να καλώ την συνάρτηση getNextCard, που θα μου επιστρέφει 12 τυχαίους και μοναδικούς αριθμούς (μέσω του πίνακα tuxaioi). Έτσι οι 12 κάρτες που βρίσκονται στις θέσεις του πίνακα kartes που αντιστοιχούν στους 12 τυχαίους αριθμούς θα μεταφέρονται στον διδιάστατο πίνακα tablo και στη συνέχεια θα εμφανίζονται στην οθόνη. Αυτή η διαδικασία γίνεται στις γραμμές 81-92 του κώδικα.

Στη γραμμή 146 που ελέγχει αν κάποιος παίκτης έπιασε τους 10 βαθμούς, έβαλα “>=” και όχι μόνο το “=” γιατί υπάρχει περίπτωση ένας παίκτης να έχει 9 βαθμούς και με μια νίκη να πάει στους 11, οπότε πάλι είναι νικητής.