Οντοκεντρικός Προγραμματισμός ΙΙ

Εργασία για το Ακαδημαϊκό έτος 2007-2008

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ

11 Φεβρουαρίου 2008

Σύνταξη από:
Αλκαλάι Ερρίκος (3889) - Β' Έτος
Καλάργαρης Χαράλαμπος (3929) - Β' Έτος
Κωτσόγιαννης - Τεφτσόγλου Ιωάννης (3961) - Β' Έτος

Οντοκεντρικός Προγραμματισμός II

Εργασία για το Ακαδημαϊκό έτος 2007-2008

Περιεχόμενα

Μέρος Α

Σκοπός

Σκοπός του μέρους Α είναι η υλοποίηση του αλγορίθμου της Ταχείας Ταξινόμησης, πιο γνωστή ως quicksort. Η quicksort είναι ένας αλγόριθμος ταξινόμησης του οποίου ο χρόνος εκτέλεσης χειρότερης περίπτωσης για έναν πίνακα ν στοιχείων είναι $\Theta(n^2)$. Παρά το μεγάλο αυτό χρόνο χειρότερης περίπτωσης, συνήθως η quicksort αποτελεί στην πράξη την καλύτερη επιλογή για ταξινόμηση, καθώς είναι εξαιρετικά ταχεία κατά μέσο όρο, ο αναμενόμενος χρόνος εκτέλεσης της είναι Θ(nlogn). Επιπλέον έχει το πλεονέκτημα να είναι επιτόπια (δηλαδή τα στοιχεία της να αναδιατάσσονται εντός του πίνακα), ενώ ταυτόχρονα λειτουργεί καλά ακόμα και σε περιβάλλοντα εικονικής μνήμης.

Σχεδιασμός

Για την υλοποίηση του αλγορίθμου χρησιμοποιούνται κάποιες βασικές δομικές συναρτήσεις. Αυτές είναι :

void Quicksort(int *a, int first, int last)	Αναλαμβάνει την αναδρομική επίλυση του προβλήματος.
int Pivot(int *a, int first, int last)	Βρίσκει και επιστρέφει την θέση του στοιχείου οδηγού και αναδιατάσσει ένα πίνακα.
void Swap(int &v1, int &v2)	Εναλλάσσει τις τιμές δύο μεταβλητών.
void PrintArray(int *A, int nElements)	Εκτυπώνει τις τιμές ενός πίνακα ακέραιων.
bool isSorted(int *A, int nElements)	Ελέγχει αν ένας πίνακας είναι ταξινομημένος ή όχι.

Ανάλυση

Η quicksort βασίζεται στο υπόδειγμα διαίρει-και-βασίλευε. Τα τρία μέρη της διαδικασίας για την ταξινόμηση ενός πίνακα A[p..r] έχουν ως εξής:

Διαίρει : Διαμέριση – αναδιάταξη του πίνακα A[p..r] σε δύο υπό-πίνακες A[p..q-1] και A[q+1..r] τέτοιους ώστε το κάθε στοιχείο του A[p..q-1] να είναι μικρότερο ή ίσο κάθε στοιχείου του A[q+1..r]. Στο πλαίσιο αυτής της διαδικασίας διαμέρισης υπολογίζεται ο αύξων αριθμός q.

Βασίλευε : Ταξινόμηση των δυο πινάκων A[p..q-1] και A[q+1..r] με αναδρομικές κλήσεις της quicksort.

Συνδύασε : Δεδομένου ότι οι υπό-πίνακες ταξινομούνται επί τόπου, ο συνδυασμός τους δεν απαιτεί καμία επιπλέον εργασία: ο πίνακας A[p..r] είναι ήδη ταξινομημένος στο σύνολο του.

Υλοποίηση

Η ταξινόμηση υλοποιείται μέσω της συνάρτησης Quicksort( ):

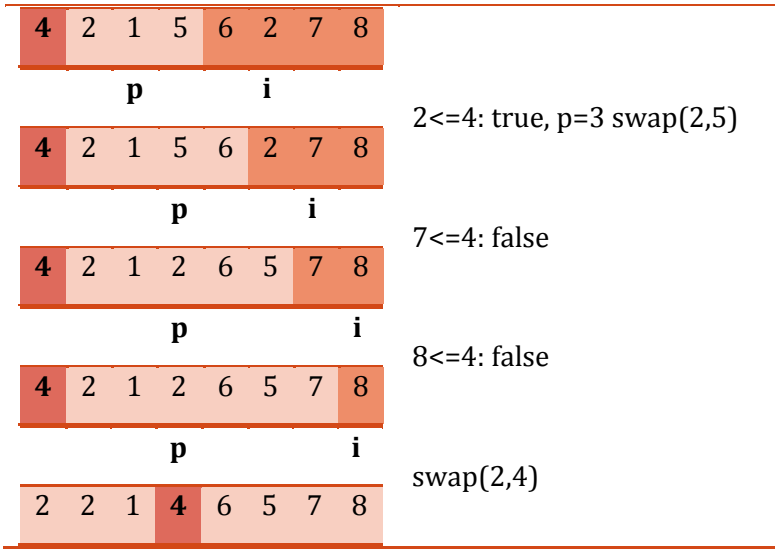
```
void Quicksort( int *a, int first, int last ) {
    int pivot;
    if( first < last ) {
        pivot = Pivot( a, first, last );
        Quicksort( a, first, pivot-1 );
        Quicksort( a, pivot+1, last );
    }
}
```

Η βασική συνιστώσα του αλγορίθμου είναι η διαδικασία Pivot, η οποία αναδιατάσσει τον πίνακα A[p..r] επί τόπου.

```
int Pivot( int *a, int first, int last ) {
    int  p = first;
    int pivot = a[first];
    for( int i = first+1 ; i <= last ; i++ ) {
        if( a[i] <= pivot ) {
            p++;
            Swap( a[i], a[p] );
        }
    }
    Swap( a[p], a[first] );
    return p;
}
```

Στον παρακάτω πίνακα φαίνεται η λειτουργιά της συνάρτησης Pivot για έναν πίνακα 8 στοιχείων. Η Pivot επιλέγει πάντα το στοιχείο pivot =a[first] ως οδηγό γύρω από τον οποίο θα διαμερίσει τον πίνακα A[p..r]. Καθώς εκτελείται η συνάρτηση, ο πίνακας διαμερίζεται σε τρεις περιοχές.

p **i**	pivot=4, p=0, i=1
4 2 5 1 6 2 7 8	2<=4: true, p=1,swap(2,2)
p **i**	5<=4: false
4 2 5 1 6 2 7 8	
p **i**	1<=4: true, p=2,swap(1,5)
4 2 5 1 6 2 7 8	
p **i**	6<=4: false



Στο τέλος εκτέλεσης της συνάρτησης Pivot αριστερά του p θα βρίσκονται όλα τα στοιχεία μικρότερα αυτού, ενώ δεξιά του όλα αυτά που είναι μεγαλύτερα. Μετά θα κληθεί και πάλι αναδρομικά η συνάρτηση quicksort με ορίσματα τους υπό – πίνακες από την αρχή μέχρι το στοιχείο p-1 και από το p+1 μέχρι το τέλος. Η αναδρομή θα γίνεται εωσότου έχουμε υπό-πίνακες με ένα στοιχείο που είναι από μόνοι τους διατεταγμένοι.

Μέσω της συνάρτησης Swap αλλάζουμε τις τιμές δύο μεταβλητών δίνοντας ως ορίσματα τις διευθύνσεις αυτών.

Στη συνέχεια η συνάρτηση PrintArray παίρνει ως ορίσματα την αρχική διεύθυνση ενός πίνακα και το σύνολο των στοιχείων του για να τον εκτυπώσει στην μορφή $[a_0, a_1, a_2,\ldots,a_n]$

Τέλος η συνάρτηση isSorted επιστρέφει μια μεταβλητή τύπου Boolean, true αν ο πίνακας είναι ταξινομημένος ενώ false σε διαφορετική περίπτωση. Εδώ υπάρχει ένας βρόχος for που συγκρίνει το i στοιχείο με το προηγούμενο του. Αν το προηγούμενο στοιχείο είναι μεγαλύτερο τότε σταματάμε τις επαναλήψεις και επιστρέφουμε false. Αν ολοκληρωθούν όλες οι επαναλήψεις σημαίνει ότι όλα τα στοιχεία είναι διατεταγμένα σε αύξουσα σειρά, άρα μπορεί η συνάρτηση να επιστρέψει true.

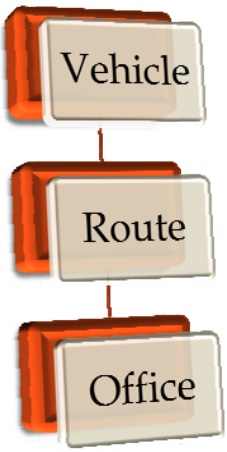
Μέρος Β

Σκοπός

Σκοπός του μέρους Β είναι η δημιουργία ενός ταξιδιωτικού γραφείου το οποίο θα διαχειρίζεται δρομολόγια ανά προορισμό και ανά μεταφορικό μέσο. Θα έχει την δυνατότητα μνα ελέγχει την πληρότητα των δρομολογίων και να θέτει τα άτομα σε λίστες αναμονής. Επίσης θα εμφανίζει στην οθόνη συγκεντρωτικά αποτελέσματα τα οποία θα μπορεί να τα αποθηκεύει και σε αρχείο.

Σχεδιασμός

Το πρόγραμμα είναι γραμμένο σε C++ και για την κατανόηση των εννοιών του αντικειμενοστραφούς προγραμματισμού χρησιμοποιούνται ως δομικά συστατικά σχέσεις κλάσεων καθώς και την κληρονομικότητα, η οποία παίζει καθοριστικό ρόλο για την υλοποίηση του. Παρακάτω φαίνεται το βασικό σχεδιάγραμμα των κλάσεων που χρησιμοποιούνται.



Σαν βασική κλάση ορίζω την Vehicle, η οποία έχει τα μεταφορικά μέσα. Υποκλάση της είναι η κλάση Route η οποία ασχολείται με τις διαδρομές – δρομολόγια. Τέλος υποκλάση της Route είναι η κλάση Office η οποία διαχειρίζεται όλες τις ζητούμενες λειτουργίες :

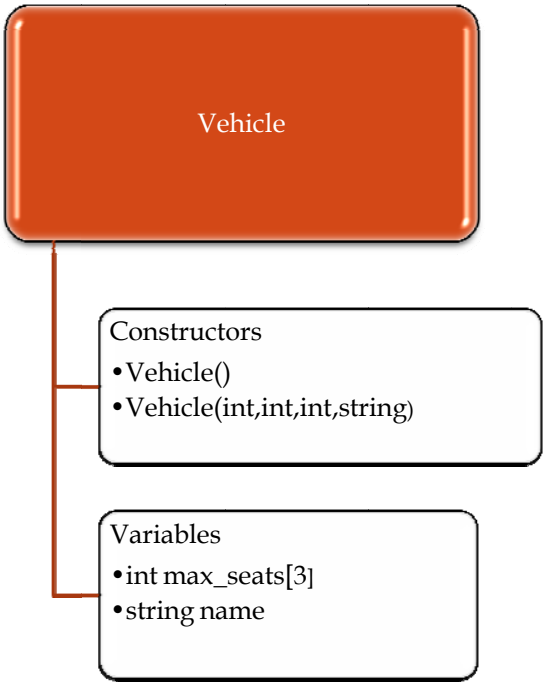
1. Κράτηση θέση, δίνοντας δρομολόγιο και κατηγορία θέσης.
2. Ακύρωση κράτησης όπου απλώς θα δίνονται στοιχεία για το δρομολόγιο και την κατηγορία.
3. Εκτύπωση στην οθόνη της τρέχουσας κατάστασης σύμφωνα με την ζητούμενη μορφή.
4. Αποθήκευση της συνολικής κατάστασης όλων των δρομολογίων σε ASCII αρχείο, με όνομα kratiseis.dat.
5. Έλεγχος πληρότητας δρομολογίου:
    a. Πληρότητα μιας κατηγορίας θέσεων.
    b. Συνολική πληρότητα.
6. Λίστα αναμονής.

Όλες οι κλάσεις έχουν γραφτεί στο αρχείο office.h. Ενώ το κυρίως πρόγραμμα βρίσκεται στο αρχείο main.cpp. Στο μεν office.h υπάρχουν όλοι οι ορισμοί των κλάσεων και των χρησιμοποιούμενων συναρτήσεων, στο δε main.cpp γίνονται οι αρχικοποιήσεις των μεταφορικών μέσων, δρομολογίων και η "έναρξη" του προγράμματος.

Ανάλυση

Κλάση Vehicle.

Παρακάτω φαίνεται το σχεδιάγραμμα της κλάσης Vehicle:



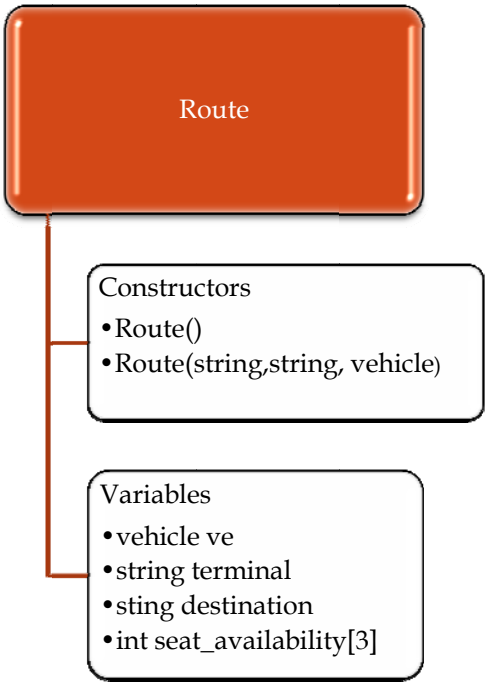
Για την κλάση Vehicle υπάρχουν δύο constructors. Ο πρώτος χωρίς ορίσματα απλά δημιουργεί ένα αυθαίρετο αντικείμενο για να μην υπάρχει κάποιο λάθος στη ροή του προγράμματος. Ο δεύτερος constructor παίρνει σαν ορίσματα τα τρεις ακεραίους τους οποίους τους τοποθετεί σε πίνακα και ένα string.

Ο πίνακας ακεραίων 3 θέσεων max_seats περιλαμβάνει τον μέγιστο αριθμό θέσεων ανά κατηγορία για το κάθε μεταφορικό μέσο. Όταν ένα μεταφορικό μέσο δεν διαθέτει μια κατηγορία θέσεων, όπως πχ στο ζητούμενο πρόγραμμα το τρένο έχει μόνο θέσεις τύπου Α και Β και όχι Γ, τότε απλά θέτουμε την αντίστοιχη τιμή στο 0 (μηδέν). Έτσι με αυτό τον τρόπο επιτυγχάνουμε την περαιτέρω επέκταση του προγράμματος σε περίπτωση που οι υπεύθυνοι των μεταφορικών μέσων αποφασίσουν να βάλουν μια επιπλέον κατηγορία θέσης.

Η μεταβλητή name τύπου string περιέχει το όνομα του μεταφορικού μέσου ώστε να μπορεί να χρησιμοποιηθεί από το πρόγραμμα για εκτύπωση ή αποθήκευση δεδομένων.

Κλάση Route.

Παρακάτω φαίνεται το σχεδιάγραμμα της κλάσης Route:



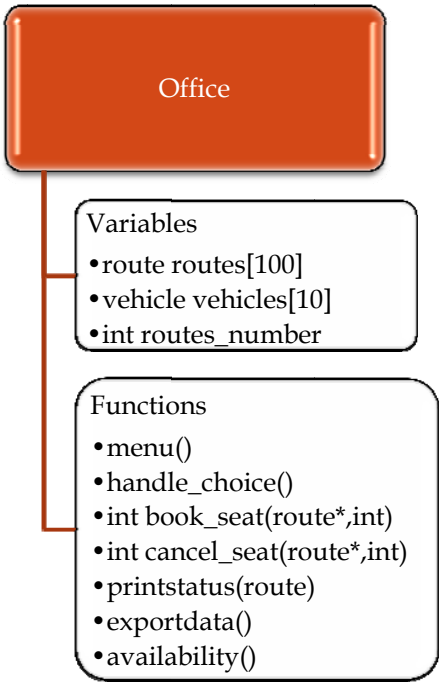
Επίσης για την κλάση Route υπάρχουν δύο constructor, ο ένας εκ των δύο αυθαίρετος και ο δεύτερος παίρνει ορίσματα 2 string και ένα όρισμα τύπου vehicle.

Οι μεταβλητές τύπου string αποθηκεύονται στις μεταβλητές terminal και destination, οι οποίες χρησιμοποιούνται για την εκτύπωση των κατάλληλων αποτελεσμάτων στην οθόνη. Η μεταβλητή ve τύπου vehicle έχει σκοπό να γνωρίζουμε τι είδους μεταφορικό μέσο εξυπηρετεί το συγκεκριμένο δρομολόγιο.

Τέλος ο πίνακας 3 ακεραίων seat_availability έχει κάθε στιγμή το πόσες θέσεις υπάρχουν διαθέσιμες για το κάθε δρομολόγιο. Οι 3 θέσεις του πίνακα αντιστοιχούν στις κατηγορίες Α, Β και Γ. Κάθε φορά που κρατάμε μια θέση, τότε αφαιρούμε ένα από την αντίστοιχη κατηγορία θέσης.

Κλάση Office.

Παρακάτω φαίνεται το σχεδιάγραμμα της κλάσης Office:



Η κλάση office περιέχει έναν πίνακα διαδρομών και έναν με τα μεταφορικά μέσα που διαθέτει το ταξιδιωτικό γραφείο. Για χάριν ευχρηστίας του προγράμματος η επιλογή του μεγέθους των πινάκων είναι μεγαλύτερη από το ζητούμενο της άσκησης για πιθανή επέκταση του. Επιπλέον υπάρχει η μεταβλητή routes_number η οποία περιέχει τον ακριβή αριθμό των διαδρομών που διαθέτει το γραφείο.

Οι συναρτήσεις που χρησιμοποιεί η Office έχουν σκοπό στην επίλυση των ζητούμενων ερωτημάτων. Η συνάρτηση menu() εμφανίζει στην οθόνη το κεντρικό μενού επιλογών, ενώ η συνάρτηση handle_choice διαχειρίζεται την επιλογή του χρήστη και καλεί την αντίστοιχη συνάρτηση. Οι υπόλοιπες συναρτήσεις αναλύονται παρακάτω στην ενότητα Υλοποίηση.

Υλοποίηση

Για την υλοποίηση του προγράμματος έχουν δημιουργηθεί 2 αρχεία. Το main.cpp και το office.h. Το μεν πρώτο αρχείο main.cpp περιέχει κάποιες αρχικοποιήσεις και ορισμούς των στιγμιότυπων των βασικών κλάσεων. Το δε δεύτερο office.h έχει τους ορισμούς των κλάσεων και των συναρτήσεων που χρησιμοποιούνται.

Αρχικά στο αρχείο main.cpp ορίζω ένα αντικείμενο τύπου office και τρία αντικείμενα τύπου vehicle ένα για το κάθε διαθέσιμο όχημα του γραφείου ταξιδιών. Έπειτα αρχικοποιώ και τα διαθέσιμα δρομολόγια. Τέλος καλώ την συνάρτηση menu() της κλάσης office για να ξεκινήσει η ροή του προγράμματος.

Στο αρχείο office.h όπως έχει αναφερθεί περιέχονται οι βασικές κλάσεις και συναρτήσεις. Η book_seat() όταν καλείτε δίνονται σαν ορίσματα η διαδρομή και ο τύπος της θέσης που θέλουμε να κλείσουμε. Στη συνέχεια ρωτάται ο χρήστης πόσες θέσεις θέλει να κλείσει και γίνεται ο έλεγχος αν υπάρχουν τόσες διαθέσιμες θέσεις. Σε θετική ανταπόκριση του

συστήματος τότε αφαιρείται ο αριθμός των ζητούμενων θέσεων από την μεταβλητή των υπολειπόμενων θέσεων του δρομολογίου.

Π.χ. ότι υπάρχουν 5 διαθέσιμες θέσεις για την κατηγορία Α, ο χρήστης μας ζητάει 6 θέσεις, δυστυχώς για αυτόν δεν υπάρχουν τόσες διαθέσιμες θέσεις και για αυτό του εμφανίζεται αντίστοιχο μήνυμα. Στην περίπτωση που ζητήσει 5 θέσεις, τότε θα αφαιρεθούν αυτές οι θέσεις από τις διαθέσιμες και η τιμή της μεταβλητής θα είναι πλέον 1 θέση.

Αντίστοιχη είναι και η λειτουργία cancel_seat, μόνο που στην συγκεκριμένη περίπτωση ελέγχεται αν ο ζητούμενος αριθμός θέσεων προς ακύρωση ξεπερνάει τον μέγιστο αριθμό θέσεων του αντίστοιχου οχήματος.

Η συνάρτηση printstatus() παίρνει σαν όρισμα ένα αντικείμενο τύπου route. Εδώ θέλουμε να εκτυπωθεί η κατάσταση του κάθε δρομολογίου σύμφωνα με το δοσμένο πρότυπο. Αρχικά γίνεται ένας έλεγχος για το αν το δρομολόγιο έχει κατηγορίες θέσεων Α και Β ή Β και C. Μετά θα εκτυπωθούν κανονικά στην οθόνη πληροφορίες όπως: όνομα δρομολογίου, αρχικό και τελικός προορισμός, συνολικές, διαθέσιμες και υπολειπόμενες θέσεις του δρομολογίου και για την κάθε κατηγορία θέσης.

Η αντίστοιχη συνάρτηση exportdata() εκτυπώνει για όλα τα δρομολόγια την κατάσταση του στο αρχείο kratiseis.dat που βρίσκεται μέσα στον κατάλογο του προγράμματος. Αν αυτό το αρχείο δεν υπάρχει τότε το δημιουργεί. Ενώ όταν υπάρχει απλά το ανανεώνει.

Τέλος η συνάρτηση availability() έχει δύο λειτουργίες για τον έλεγχο πληρότητας των δρομολογίων. Η πρώτη είναι για έλεγχο για το κάθε δρομολόγιο και η άλλη για την κατηγορία της θέσης. Ο έλεγχος ανά δρομολόγιο γίνεται επιλέγοντας το δρομολόγιο και αθροίζουμε τις διαθέσιμες θέσεις σε όλες τις κατηγορίες. Από την άλλη ο έλεγχος ανά κατηγορία γίνεται επιλέγοντας την κατηγορία που θέλουμε να βρούμε διαθέσιμο δρομολόγιο και τότε εκτυπώνονται οι διαθέσιμες θέσεις για όλα τα δρομολόγια στην συγκεκριμένη κατηγορία.

Βιβλιογραφία

- T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Εισαγωγή στους Αλγορίθμους*, ελληνική μετάφραση, Πανεπιστημιακές Εκδόσεις Κρήτης, 2006
- Bjarne Stroustrup, Η Γλώσσα Προγραμματισμού C++ 3η έκδοση, ελληνική μετάφραση, Κλειδάριθμος, 2006
- Kris Jamsa, Εισαγωγή στη…C++ - Ο πιο εύκολος τρόπος για να μάθετε τη C++, Κλειδάριθμος, 2004
- David Chapman, Teach Yourself Visual C++ 6 in 21 Days, Sams,1988
- Chuck Sphar, Learn Microsoft Visual C++ 6.0 Now, Microsoft Press, 1999
- http://mmlab.ceid.upatras.gr/courses/language_C++/
- http://www.cplusplus.com
- http://en.wikipedia.org/wiki/Quicksort
- Bruce Eckel, Thinking in C++, http://www.mindview.net/Books/TICPP/ThinkingInCPP2e.html
- Συμπληρωματικές σημειώσεις του μαθήματος από την ιστοσελίδα: http://mmlab.ceid.upatras.gr/courses/language_C++/lab/C++-Notes.pdf