Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Αλγόριθμοι και Πολυπλοκότητα

Διδάσκοντες: Σ. Ζάχος, Δ. Φωτάκης

1η Σειρά Προγραμματιστικών Ασκήσεων - Ημ/νία Παράδοσης 2/11/2009

Άσκηση 1: Ταξινόμηση με Εισαγωγή

Το πρόβλημα της ταξινόμησης n αριθμών είναι γνωστό: δοσμένων n αριθμών σε τυχαία σειρά, πρέπει να τοποθετηθούν σε αύξουσα (ή φθίνουσα) σειρά. Ένας από τους αλγορίθμους για την επίλυσή του είναι ο αλγόριθμος *ταξινόμησης με εισαγωγή* (insertion sort). Περισσότερες πληροφορίες για την ταξινόμηση με εισαγωγή μπορείτε να βρείτε στο:

`http://ocw.mit.edu/OcwWeb/Electrical-Engineering-and-Computer-Science/6-046JFall-2005/LectureNotes/lec1.pdf`

Ζητείται να υλοποιήσετε τον αλγόριθμο ταξινόμησης με εισαγωγή και να πειραματιστείτε για διάφορες ακολουθίες αριθμών. Το πρόγραμμα που θα υλοποιήσετε πρέπει να διαβάζει τους αριθμούς από αρχείο με την εξής μορφή: Στην πρώτη γραμμή θα βρίσκεται το πλήθος των αριθμών προς ταξινόμηση, και στη δεύτερη γραμμή οι αριθμοί χωρισμένοι με κενό. Για παράδειγμα:

```
6
3 59 23 75 657 9
```

Η έξοδος του προγράμματος θα είναι οι αριθμοί σε αύξουσα σειρά χωρισμένοι με κενό. Για την παραπάνω είσοδο, η έξοδος θα είναι η εξής:

```
3 9 23 59 75 657
```

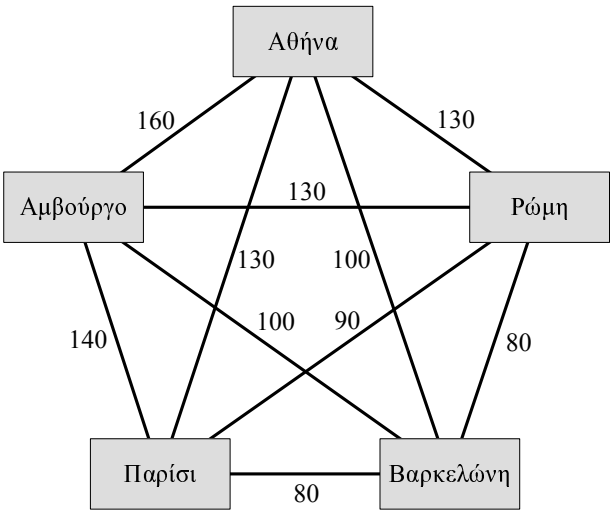
Το πρόγραμμά πρέπει επίσης να εκτυπώνει τον αριθμό των συγκρίσεων (μόνο όσες λαμβάνουν χώρα μεταξύ αριθμών που ταξινομούνται) που εκτελεί η ταξινόμηση με εισαγωγή για κάθε αρχείο εισόδου. Το συμπληρωματικό αρχείο `sort.c` περιέχει ήδη υλοποιημένες τις συναρτήσεις για είσοδο και έξοδο του προγράμματος.

Πειραματιστείτε με μεγάλο πλήθος αριθμών ($n \geq 1.000$) και δείτε πως μεταβάλλεται το πλήθος των συγκρίσεων ως συνάρτηση του n. Ποιο είναι το πλήθος των συγκρίσεων όταν οι αριθμοί της εισόδου είναι ταξινομημένοι σε αύξουσα σειρά και ποιο όταν είναι ταξινομημένοι σε φθίνουσα σειρά;

Άσκηση 2: Πρόβλημα Πλανόδιου Πωλητή

Έστω ότι θέλετε να επισκεφθείτε τις πόλεις Αθήνα, Ρώμη, Βαρκελώνη, Παρίσι και Αμβούργο (με όποια σειρά θέλετε) και να επιστρέψετε μετά στην πόλη από την οποία ξεκινήσατε. Θέλετε όμως να βρείτε την *περιοδεία* (δηλ. την κυκλική διαδρομή πού διέρχεται από όλες τις πόλεις) με το ελάχιστο συνολικό κόστος μετάβασης. Το κόστος μετάβασης μεταξύ των πόλεων είναι συμμετρικό, δηλ. η μετάβαση από την πόλη Α στην πόλη Β κοστίζει όσο και η μετάβαση από την πόλη Β στην πόλη Α. Για το συγκεκριμένο παράδειγμα, το κόστος μετάβασης μεταξύ των πόλεων δίνεται στο Σχήμα 1.

1. Γράψετε πρόγραμμα το οποίο δέχεται σαν είσοδο 5 πόλεις και τα κόστη μετάβασης μεταξύ τους, και υπολογίζει την συντομότερη περιοδεία που διέρχεται από τις 5 πόλεις.
2. Επεκτείνετε το πρόγραμμα σας ώστε να δέχεται ως είσοδο μέχρι 24 πόλεις, και εφαρμόστε το στο παράδειγμα του αρχείου `24cities.txt`.



Σχήμα 1. Στιγμιότυπο του Προβλήματος του Πλανόδιου Πωλητή με 5 πόλεις και συμμετρικά κόστη μετάβασης. Ποια είναι η περιοδεία ελάχιστου κόστους;

Το πρόγραμμα θα διαβάζει τα κόστη μετάβασης μεταξύ των πόλεων από ένα αρχείο. Στην πρώτη γραμμή του αρχείου θα βρίσκεται το πλήθος των πόλεων. Οι επόμενες γραμμές θα περιέχουν τον κάτω αριστερά τριγωνικό υποπίνακα του πίνακα γειτνίασης του γράφου (αυτό αρκεί, καθώς τα κόστη είναι συμμετρικά). Για παράδειγμα, το αρχείο για το παράδειγμα του Σχήματος 1 θα περιλαμβάνει τα εξής:

```
5
0
130 0
100 80 0
130 90 80 0
160 130 100 140 0
```

Το πρόγραμμα θα εκτυπώνει στην πρώτη γραμμή το συνολικό κόστος της βέλτιστης περιοδείας, και στην δεύτερη γραμμή τη σειρά με την οποία επισκέπτεται τις πόλεις η βέλτιστη περιοδεία ξεκινώντας και καταλήγοντας στην πόλη 1. Για παράδειγμα,

```
2654
1 2 3 4 5 1
```

Παρατηρήσεις:

- Το πρόβλημα, γνωστό ως Travelling Salesman Problem (TSP), είναι **NP**-complete, που σημαίνει ότι, κατά κοινή πεποίθηση, δεν υπάρχει αποδοτικός (πολυωνυμικός) αλγόριθμος που να το λύνει. Μια προφανής λύση είναι να ελεγχθούν όλες οι μεταθέσεις των n πόλεων, που όμως είναι $(n-1)!$ το πλήθος. Μπορείτε να βρείτε πιο αποδοτική λύση;
- Στο συμπληρωματικό αρχείο `tsp.c` θα βρείτε έτοιμη υλοποίηση για την ανάγνωση του αρχείου εισόδου και τη δημιουργία του πίνακα γειτνίασης.

Web Links:

- `http://en.wikipedia.org/wiki/Travelling_salesman_problem`
- `http://en.wikipedia.org/wiki/Permutation`